Повышение эффективности регрессионных моделей за счет исключения аномальных данных

В. Г. Мосин

Аннотация

Исследован метод, определяющий влияние выбросов на производительность регрессионных моделей. На примере данных о потреблении контента одного из ведущих хостингов показано, что исключение аномальных объектов способно существенно повысить их прогнозирующую способность. Даны рекомендации по применению метода исключения аномальных данных.

Содержание

1 Введение

Выбросы представляют собой наблюдения, которые значительно отклоняются от остальных значений в наборе данных. Хорошо известно, что выбросы оказывают существенное влияние на прогнозирующую способность моделей машинного обучения, поэтому важно учитывать их в анализе данных и принимать меры для их предварительной обработки с целью нивелировать влияние аномальных значений на производительность и эффективность моделей машинного обучения (см. [6], [7]).

Одним из основных негативных факторов наличия выбросов и их влияния на модели машинного обучения является искажение статистических показателей данных, таких как среднее значение, медиана и стандартное отклонение. Это приводит к неправильному определению зависимостей между признаками и целевой переменной, ведь выбросы могут внести существенное искажение в статистику данных. Кроме того, выбросы могут сильно влиять на основные алгоритмы машинного обучения, такие как линейная регрессия или метод ближайших соседей, так как эти модели основываются на расстоянии между наблюдениями, и выбросы могут занимать отдельные области пространства наблюдений.

Это может приводить к искажению предсказаний или повышению ошибки модели.

В целом, выбросы в данных могут серьезно повлиять на прогнозирующую способность моделей машинного обучения, поэтому важно учитывать их при анализе данных и выборе подходящих методов обработки выбросов.

1.1 Теоретическая часть

Итак, если в данных есть существенные выбросы, то модель, обучаясь на таких данных, после обучения дает некорректные прогнозы даже на нормальных, не являющихся аномальными, объектах. И наоборот: обучившись на нормальных объектах, модель, получая для прогноза аномальный объект, может дать некорректный прогноз. Это замечание позволяет нам сформулировать два вполне очевидных принципа работы с нетипичными данными:

1. Обучать модели следует только на типичных объектах.
2. Применять модели следует только к типичным объектам.

Другой вопрос, какие объекты считать типичными, другими словами, что есть норма и что есть аномалия в имеющихся для обучения и во вновь поступающих для прогноза данных. Если обучающие данные размечены по принципу «норма/аномалия», то определение выбросов — это просто задача бинарной классификации. Если же такой разметки нет, то ситуация существенно усложняется. Один и способов детекции аномалий на неразмеченных данных состоит в применении сингулярных разложений: методом SVD определяются главные направлении в облаке обучающих данных, вычисляются проекции объектов подпространства, порожденные одним, двумя или несколькими главными направлениями[В настоящей работе мы используем одно главное направление.] и вычисляется так называемая реконструкционная ошибка, то есть расстояние от объекта до его проекции. Объекты с наибольшей реконструкционной ошибкой признаются выбросами.

1.2 Постановки задачи

1.2.1 Предмет исследования

Исследуется прогнозирующая способность нескольких регрессионных моделей до и после удаления выбросов.

1.2.2 Методика исследования

Применяя метод сингулярных разложений, мы вычисляем реконструкционные ошибки объектов обучающих данных и устанавливаем порог отсечения на уровне определенного перцентиля (99%, 98%, 97% и так далее, до 90%). Объекты, превысившие порог отсечения, признаются выбросами и удаляются, после чего на очищенных данных обучается регрессионная модель. В качестве ее метрики эффективности мы используем коэффициент детерминации, который вычисляем на тестовых данных, после того, как к ним применяется тот же метод очистки от выбросов.

1.2.3 Цель исследования

Наша цель — выяснить, приводит ли удаление выбросов к существенному повышению коэффициента детерминации.

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

Мы используем данные о потреблении контента одного из ведущих хостингов за период 500 дней, с 2021-08-20 по 2023-01-01. В качестве показателя востребованности контента мы берем общее количество просмотров за сутки, в качестве показателя удовлетворенности — количество лайков. Кроме того, мы используем еще один вспомогательный признак: количество показов значков канала в течение суток для того, чтобы иметь избыточную размерность в пространстве объектов и возможность редукции данных к первому главному направлению.

3 Алгоритм

3.1 Чтение данных

Используя функцию `read_csv` из библиотеки `pandas`, загружаем набор данных о потреблении контента и создаем дата-фрейм в текущей среде

исполнения программы.

	Просмотры	Показы	Лайки
0	475.0	5176.0	16.0
1	174.0	2222.0	4.0
2	490.0	5584.0	3.0
...	...	...	...
498	222.0	2976.0	4.0
499	209.0	2880.0	1.0

В данных присутствуют 500 объектов. Все признаки относятся к типу с плавающей запятой и не содержат пропущенных значений.

3.2 Установка начальных значений

В данном алгоритме мы реализуем вложенный цикл по трем уровням:

1. На верхнем уровне мы будем варьировать признаки, используя их поочередно в качестве целевой функции для регрессионной задачи, а оставшиеся два признака — в качестве предикторов модели.
2. На втором уровне мы будем варьировать объем данных, которые мы признаем выбросами, постепенно увеличивая их число.
3. Наконец, на третьем, самом глубоком уровне вложенности мы будем несколько раз воспроизводить случайные разбиения данных на обучающую и тестовую выборки для того, чтобы после усреднения результатов нивелировать эффект случайности.

Присваиваем стартовое значение переменной, под именем которой будет выступать целевая функция: `target='Просмотры'`. Во вложенном цикле мы будем отсеивать выбросы, используя $(100 - K)$-й перцентиль от общего объема данных (подробнее об этом см. ниже). На старте цикла устанавливаем `K=0`.

3.3 Разбиение данных на train и test

Пользуемся модулем `model_selection` из библиотеки `sklearn` и применяем метод `train_test_split` с параметром `test_size=0.2`. В результате получаем два датафрейма: `df_train` с 400 объектами и `df_test` с 100 объектами.

3.4 Нормализация на train'е

Применяя метод `describe` из библиотеки `pandas` к обучающей выборке, получаем информацию о статистических показателях всех признаков:

	min	mean	max	std
Просмотры	159.00	936.51	2200.00	418.144
Показы	1938.00	8093.78	39479.00	3816.08
Лайки	−6.00	15.80	70.00	9.13

Диапазоны значений для признаков 'Показы' и 'Лайки' сильно отличаются друг от друга, отличие составляет несколько порядков. Чтобы избежать неравномерности в масштабах, применяем процесс нормализации к данным и приводим их к стандартному виду:

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

После преобразования, все признаки получаются центрированными с нулевыми средними значениями, а их дисперсии становятся единичными. Таким образом, дисбаланс в размерностях исчезает:

	min	mean	max	std
Просмотры	−1.85	0.00	3.02	1.00
Показы	−1.61	0.00	8.22	1.00
Лайки	−2.38	0.00	5.93	1.00

3.5 Нормализация на test'е

Применяя метод `describe` из библиотеки `pandas` к тестовой выборке, мы получаем сведения о статистиках по всем признакам на тесте:

	min	mean	max	std
Просмотры	126.00	916.10	1805.00	422.82
Показы	1701.00	8418.39	69432.00	6887.19
Лайки	0.00	17.86	63.00	12.48

Для приведения облака тестовой выборки к облаку обучающей выборки, требуется преобразовать данные тестовой выборки с использованием средних и дисперсий обучающей выборки:

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

Важно отметить, что после этой процедуры средние значения признаков тестовой выборки отличаются от нуля, а их дисперсии не являются единичными:

	min	mean	max	std
Просмотры	–1.93	–0.04	2.07	1.01
Показы	–1.67	0.08	16.07	1.809
Лайки	–1.73	0.22	5.16	1.36

Это происходит потому, что центр и рассеяние облака были настроены на обучающей выборке заранее.

3.6 Регрессия до удаления выбросов

Пользуясь методом `drop` библиотеки `pandas`, удаляем столбец с именем `target` как в обучающем, так и в тестовом датафреймах. Затем, пользуясь методом `to_numpy` библиотеки `pandas`, преобразуем датафреймы к парам массивов: `X_train`, `y_train` и `X_test`, `y_test`. Здесь `X` представляют собой двумерные массивы со значениями предикторов, а `y` — одномерные массивы со значениями целевой функции. Пользуясь методом `LinearRegression` библиотеки `sklearn`, формируем объект `model`. Затем применяем к нему метод `fit` на обучающих данных `X_train`, `y_train`. Эффективность модели вычисляем на тестовых данных `X_test`, `y_test`, используя метод `score` библиотеки `sklearn`. Результат записываем в переменную `score_before`.

На этом решение регрессионной задачи без выполнения каких-либо действия по очистке данных закончено. Далее мы будем избавляться от выбросов за счет сингулярных разложений матрицы обучающих данных и решать ту же регрессионную задачу на очищенных данных для того, чтобы сравнить результаты и выяснить, приводит ли удаление выбросов к повышению прогнозирующей способности регрессионной модели.

3.7 SVD на train'e

Сингулярное разложение (см. [3], [8], [9]) матрицы X — это способ представления матрицы X в виде произведения двух ортогональных матриц U и V и диагональной матрицы Σ следующим образом:

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

При этом столбцы матриц U и V являются левым и правым сингулярными базисами соответственно, а диагональные элементы матрицы Σ являются сингулярными значениями матрицы X. Теорема о сингулярном разложении утверждает, что такое разложение существует для любой матрицы X (а значит, и для нашей матрицы обучающих данных).

3.7.1 Вывод данных в массив

Данные были разбиты на `train` и `test` и нормализованы по средним и дисперсиям обучающей части выборки. Для осуществления сингулярного разложения данных обучающей выборки мы будем использовать методы библиотеки `numpy`, поэтому перед тем, как приступить к разложению, мы преобразовываем датафрейм `df_train_norm` в массив `numpy`, используя метод `to_numpy` из библиотеки `pandas`. Таким образом, мы получаем двумерный массив `X_train` нужного нам формата.

3.7.2 Разложение SVD

Получение сингулярных базисов U, V и сингулярных значений Σ осуществляется с помощью метода `svd` из модуля `linalg` библиотеки `numpy`. Применение этого метода к матрице `X_train` возвращает три объекта:

1. двумерный массив `U`, столбцы которого представляют собой левый сингулярный базис,
2. одномерный массив `Sigma`, содержащий значения сингулярных значений и служащий диагональю матрицы Σ.
3. двумерный массив `V`, представляющий собой обращенную матрицу V из теоремы о сингулярном разложении, его строки 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`.].

3.7.3 Train в сингулярном базисе

Если мы умножим матричное равенство из теоремы о сингулярном разложении на матрицу V справа, то оно изменится следующим образом:

$$XV = U\Sigma$$

Отметим, что после нормализации данных они становятся центрированными. Поэтому левая часть этого равенства представляет собой теорему о замене базиса (см. [3]). Другими словами, строки матрицы, которые

находятся в левой части, представляют собой координаты точек облака обучающих данных в правом сингулярном базисе.

Для того чтобы получить координаты облака обучающих данных в правом сингулярном базисе, мы создаем массив `S_train` и, с применением метода `dot` из библиотеки `numpy`, присваиваем ему результат матричного произведения массивов `X_train` и `V.T`. Здесь важно отметить, что мы транспонировали правый множитель из-за особенностей реализации алгоритма сингулярного разложения в библиотеке `numpy` (см. выше).

3.8 Редукция train'а к главному направлению

Для получения координат редуцированных данных в сингулярном базисе, мы применяем следующий подход. Возьмем массив `S_train`, который был получен ранее, и заменим все столбцы, кроме первого, нулями. Полученному массиву присваиваем имя `S_trian_reduce`. В этом массиве строки представляют собой координаты редуцированных данных в сингулярном базисе, но нас интересуют их координаты в исходном базисе. Для этого нам нужно снова воспользоваться теоремой о замене базиса и выполнить обратный переход. Однако на этот раз мы применяем его не к полной матрице `S_trian`, а к редуцированной матрице `S_trian_reduce`.

Результат матричного произведения массивов `S_train_reduce` и `V` присваиваем массиву `X_train_reduce`, который получается с использованием метода `dot` из библиотеки `numpy`. Координаты проекций исходных данных на первое главное направление теперь содержатся в строках массива `X_train_reduce`.

3.9 Отклонения на train'е

Выбросами мы будем считать объекты с высокой реконструкционной ошибкой, которая представляет собой норму разности между редуцированными данными и исходными данными. Другими словами, это расстояние от проекции точки облака данных на первое главное направление до самой точки.

Для вычисления реконструкционной ошибки мы используем модуль `linalg` из библиотеки `numpy` и его метод `norm`. Мы применяем этот метод к разности двумерных массивов `X_train_reduce - X_train`, указывая значение атрибута `axis=1`, чтобы норма разности вычислялась именно для строк. Результат сохраняем в массиве `reconstruction_error_train`, после чего в нем оказываются записаны реконструкционные ошибки всех объектов обучающей выборки.

3.10 Критический порог отклонения

Мы будем определять выбросы в данных путем установления порога для реконструкционной ошибки, причем, этот порог будет зависеть от объема обучающей выборки. Если мы готовы пожертвовать $K\%$ данных при обучении модели, чтобы получить более устойчивую модель, то порог отсечения будет выбран как $(100 - K)$-й перцентиль массива реконструкционных ошибок. Для этого мы используем метод `pecentil` из библиотеки `numpy` для массива `reconstruction_error_train` и результат применения этого метода записываем в переменную `threshold_train`.

3.11 Выбросы на train'e

Для получения аномальных объектов обучающей выборки пользуемся методом `where` из библиотеки `numpy`. В данном случае условие метода будет следующим:

```
reconstruction_error_train > threshold_train
```

Результат записываем в массив `anomaly_indices_train`, содрежащий индексы объектов, которые мы посчитали выбросами, это некий набор номеров.

3.12 Отклонения на test'e

Получение точек тестовой выборки в сингулярном базисе происходит аналогичным тому, как это было сделано выше, когда мы проводили эту процедуру по отношению к точкам обучающей выборки. Отметим, что на этом шаге мы не выполняем новое сингулярное разложение для тестовых данных, мы используем правый сингулярный базис из разложения, полученного для обучающей выборки. Умножаем массив `X_test` на массив `V.T`, в результате получаем двумерный массив `S_test`.

Затем мы сокращаем этот массив до первого столбца, получая массив `S_test_reduce`, и возвращаемся к исходному базису, умножая сокращенные сингулярные координаты на матрицу `V`. В итоге получается массив `X_test_reduce`, в котором строки представляют собой проекции точек тестовой выборки на первое главное направление обучающей выборки.

После этого массив реконструкционных ошибок тестовых данных `reconstruction_error_test` как и выше получается за счет метода `norm` из модуля `linalg` библиотеки `numpy`.

3.13 Выбросы на test'e

Выше мы получили порог отсечения реконструкционной ошибки, используя $(100\breve{}K)$-й перцентиль на обучающей выборке. Здесь мы не вы-

числяем новый перцентиль на тестовой выборке. Наоборот: тот же порог применяется к тестовой выборке:

`reconstruction_error_test > threshold_train`

По этому условию индексы нетипичных объектов возвращаются методом `np.where` и записываются в массив `anomaly_indices_test`.

3.14 Удаление выбросов

3.14.1 Удаление выбросов на train'е

Итак, мы получили обучающую выборку `df_train`, привели ее к стандартному виду `df_train_norm`, и получили индексы аномальных объектов в виде массива `anomaly_indices_train`. Теперь, используя метод `drop` библиотеки `pandas`, мы удаляем из нормализованных обучающих данных `df_train_norm` строки с найденными номерами выбросов.

3.14.2 Удаление выбросов на test'е

Точно также мы получили тестовую выборку `df_test`, нормализовали ее к стандартному виду `df_test_norm` и получили массив с индексами аномалий на тесте `anomaly_indices_test`. Теперь, так же как и выше мы удаляем аномальные объекты из тестовой выборки.

3.15 Регрессия после удаления выбросов

Снова обучаем модель на `X_train`, `y_train` и вычисляем ее метрику эффективности на `X_test`, `y_test`. Результат записываем в переменную `score_after`.

3.16 Усреднение случайных разбиений

Повторяем все эти шаги достаточное число раз и усредняем значения переменных `score_before` и `score_after` для того, чтобы сгладить случайные артефакты и получить устойчивые результаты. На иллюстрациях в разделе 4 приведены результаты с повторением 25 раз.

3.17 Увеличение процента выбросов

Увеличиваем значений переменной `K` на 1 и снова повторяем все шаги. Этот шаг повторяем 10 раз, значения переменных `score_before` и `score_after` записываем в массивы `score_before_list`, в котором оказываются записаны коэффициенты детерминации моделей до удаления выбросов, и `score_after_list`, в который, соответственно, попадают коэффициенты детерминации, получившиеся после их удаления.

3.18 Перебор целевых функций

Изменяем значение переменной `target` еще дважды: сначала полагаем `target='Показы'`, а затем `target='Лайки'` и еще дважды воспроизводим все шаги.

4 Результаты

Мы исследовали поведение всех трех признаков, используя их поочередно в качестве целевой функции регрессионной задачи с оставшимися двумя признаками в качестве предикторов. Нас интересовал вопрос: что будет, если при решении регрессионной задачи удалять из данных некоторое количество нетипичных объектов, постепенно увеличивая их число, и не приведет ли это к повышению прогнозирующей способности регрессионной модели. Результаты исследования таковы.

4.1 Целевая функция 'Просмотры'

Процент объектов, которые мы готовы признать аномальными (это горизонтальная переменная на всех иллюстрациях), изменяется от 0% до 10%. Коэффициент детерминации (это вертикальная переменная) на неочищенных данных приблизительно постоянен (что и неудивительно, так как мы использовали множественный прогон случайных разбиений, и это нивелировало возможные случайные отклонения) и находится в районе 0.6.

После удаления выбросов коэффициент детерминации составил более 0.8, что говорит об огромной разнице в прогнозирующей способности моделей до и после удаления выбросов.

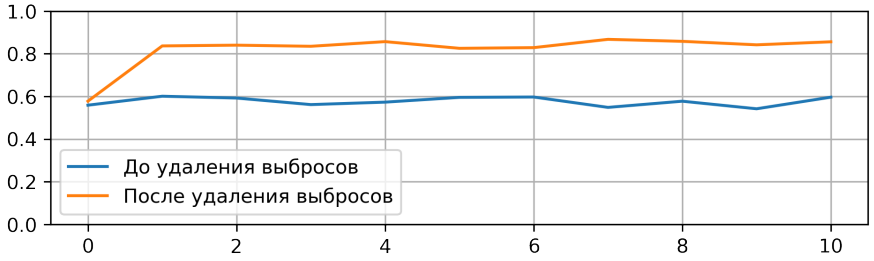

Рис. 1: Целевая функция 'Просмотры'

Поведение кривой коэффициента детерминации таково, что достаточно пожертвовать 1% обучающих данных для качественного повышения производительности регрессионной модели. Дальнейшее увеличения

числа отбракованных объектов не приводит к росту прогнозирующей способности.

4.2 Целевая функция 'Показы'

В целом, на этой целевой функции ситуация очень похожая. Процент объектов, которые мы готовы признать аномальными, изменяется от 0% до 10%. Коэффициент детерминации на неочищенных данных приблизительно постоянен и находится в районе 0.5.

После удаления выбросов коэффициент детерминации находится в районе 0.7, и это тоже огромная разница в прогнозирующей способности моделей до и после удаления выбросов.

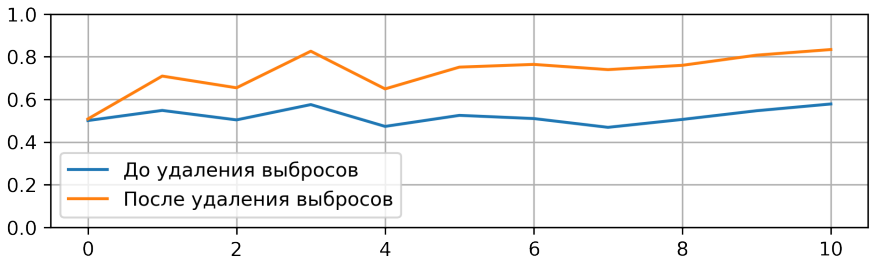

Рис. 2: Целевая функция 'Показы'

4.3 Целевая функция 'Лайки'

Здесь ситуация принципиально иная. Мы видим, что кривые коэффициентов детерминации практически совпадают (и видимо, при бесконечном увеличении числа прогонов совпадут до неразличимости), а это значит, что удаление выбросов никак не влияет на повышение прогнозирующей способности регрессионной модели.

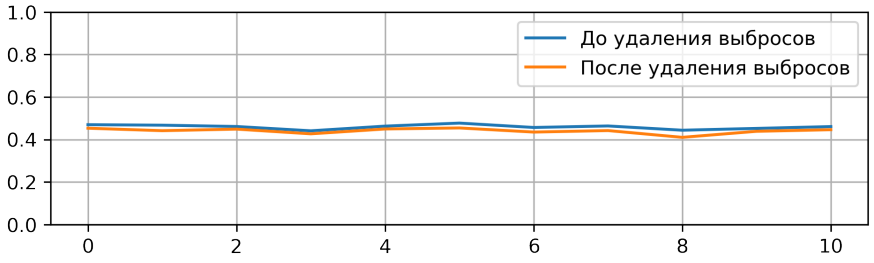

Рис. 3: Целевая функция 'Лайки'

5 Выводы

Напомним принципы, сформулированные нами в самом начале, во введении к этой работе:

1. Обучать модели следует только на типичных объектах.
2. Применять модели следует только к типичным объектам.

Соотнесем их с полученными результатами. С одной стороны, наши принципы не нашли своего подтверждения, так как мы выяснили, что они не универсальны, и возможны ситуации, когда удаление выбросов не приводит к повышению производительности (см. рис. 3). С другой стороны, возможны и оптимистичные сценарии (см. рис. 1 и 2). Поэтому будет верным скорректировать формулировку наших принципов, добавив к ним условие их справедливости:

1. Если исключение выбросов приводит к повышению производительности, то обучать модели следует только на типичных объектах.
2. Если исключение выбросов приводит к повышению производительности, то применять модели следует только к типичным объектам.

Условие, при котором принципы очистки данных оказываются рабочими и эффективными методами моделирования, требуют предварительного анализа данных и не всегда проявляются. Но то, что есть ситуации, в которых удаление аномальных объектов приводит к существенному эффекту — несомненно.

6 Литература

1. Хейдт М. Изучаем Pandas / М. Хейдт; — Москва: ДМК Пресс, 2018. — 438 с. — ISBN 978-5-97060-625-4. — Текст: непосредственный.
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.
8. Ахметшина Л.Г. Адаптивная фильтрация шумов методом сингулярного разложения автоморфного отображения // Вестник компьютерных и информационных технологий. 2006. № 8 (26). С. 12–21.
9. Бурнашев Д.С., Залыгаева М.Е. Реализация рекомендательной системы пользовательских предпочтений на основе сингулярного разложения матриц на языке Python // В сборнике: Труды молодых учёных факультета компьютерных наук ВГУ. Сборник статей. Воронеж, 2023. С. 11–18.