Научная статья

УДК 004.8
DOI 10.25205/1818-7900-2023-21-1-46-61

Прозрачное уменьшение размерности с помощью генетического алгоритма

Никита Андреевич Радеев

Новосибирский государственный университет
Новосибирск, Россия

n.radeev@g.nsu.ru, orcid.org/0000-0002-4334-5725

Аннотация

Существуют предметные области, где все преобразования данных должны быть прозрачными и объяснимыми (например, медицина и финансы). Уменьшение размерности данных является важной частью предварительной обработки данных, но алгоритмы для него в настоящее время не являются прозрачными. В данной работе мы предлагаем генетический алгоритм для прозрачного уменьшения размерности числовых табличных данных. Алгоритм строит признаки в виде деревьев выражений на основе подмножества числовых признаков из исходных данных и обычных арифметических операций. Он спроектирован так, чтобы стремиться к достижению максимального качества в задачах бинарной классификации и генерировать признаки, объяснимые человеком, что достигается за счет использования в построении признаков операций, понятных человеку. Кроме того, преобразованные алгоритмом данные могут быть использованы в визуальном анализе, если уменьшить размерность до двух. В алгоритме используется многокритериальная динамическая фитнес-функция, предназначенная для построения признаков с высоким разнообразием.

Ключевые слова

генетический алгоритм, уменьшение размерности, построение признаков, интерпретируемость, объяснимый ИИ, символьная регрессия

Для цитирования

Радеев Н. А. Прозрачное уменьшение размерности с помощью генетического алгоритма // Вестник НГУ. Серия: Информационные технологии. 2023. Т. 21, № 1. С. 46–61. DOI 10.25205/1818-7900-2023-21-1-46-61

Transparent Reduction of Dimension with Genetic Algorithm

Nikita A. Radeev

Novosibirsk State University
Novosibirsk, Russian Federation

n.radeev@g.nsu.ru, orcid.org/0000-0002-4334-5725

Abstract

There are domain areas where all transformations of data must be transparent and interpretable (medicine and finance for example). Dimension reduction is an important part of a preprocessing pipeline but algorithms for it are not transparent at the current time. In this work, we provide a genetic algorithm for transparent dimension reduction of numerical data. The algorithm constructs features in a form of expression trees based on a subset of numerical features from the source data and common arithmetical operations. It is designed to maximize quality in binary classification tasks and generate features explainable by a human which achieves by using human-interpretable operations in a feature construction.

Also, data transformed by the algorithm can be used in a visual analysis. The multicriterial dynamic fitness function is provided to build features with high diversity.

Keywords

genetic algorithm, dimension reduction, feature construction, interpretability, explainable AI, symbolic regression

For citation

Radeev N. A. Transparent Reduction of Dimension with Genetic Algorithm. *Vestnik NSU. Series: Information Technologies*, 2023, vol. 21, no. 1, pp. 46–61. (in Russ.) DOI 10.25205/1818-7900-2023-21-1-46-61

Введение

Уменьшение размерности – одна из основных частей процесса анализа данных. Низкоразмерные данные могут быть эффективно визуализированы. Такие визуализации полезны в исследовательском анализе данных для изучения их природы. Меньшим количеством признаков удобно манипулировать и анализировать человеку. Сокращение размерности полезно при обработке больших данных, поскольку оно уменьшает объем данных с минимально возможной потерей информации и позволяет ускорить обработку больших данных [1].

Очень важно, чтобы признаки, генерируемые алгоритмами уменьшения размерности, были интерпретируемыми в таких областях, как медицина, финансы, управление персоналом, правосудие, образование и маркетинг. Другими словами, все области, где решения могут повлиять на человека не лучшим образом с опасностью для здоровья, жизни, финансов, нуждаются в интерпретируемых манипуляциях с данными [2, 3]. Эксперт должен уметь объяснить значение генерируемых признаков в терминологии предметной области. В противном случае решение, предоставляемое алгоритмом, не может быть использовано в производстве, поскольку отсутствие объяснимости приводит к непредвиденным ошибкам, которые недопустимы в таких предметных областях.

Проблему уменьшения размерности можно представить как поиск низкоразмерного представления исходного пространства. Эта идея лежит в основе Manifold Learning [4, 5], которая предполагает, что полезные данные содержатся в низкоразмерном многообразии, вложенном в высокоразмерное пространство. Таким образом, цель состоит в том, чтобы найти представление, которое сохраняет полезную информацию и не содержит бесполезной информации. Полезность информации может быть определена многими способами. Самый прямой способ – это экспертная оценка информации. В то же время это самый сложный способ, потому что он требует наличия эксперта, которого трудно найти. Кроме того, время специалиста стоит дорого. Также существует естественное ограничение на объем информации, который может оценить человек. Таким образом, экспертная оценка подходит в отдельных случаях данных с небольшим количеством признаков. В случае с автоматической оценкой полезности информации существует широкий спектр вариантов. Это может быть качество предсказания модели машинного обучения, линейная разделимость [6] кластеров, принадлежащих разным классам, мера расстояния между такими кластерами или какая-либо пользовательская бизнес-метрика, которая может быть рассчитана на данных.

Генетическое программирование [7–9] – это подход к решению задач оптимизации. У него есть несколько важных свойств, которые отличают этот тип оптимизации от других. В генетической оптимизации существует функция, называемая фитнес, которую необходимо оптимизировать. Требования к фитнес-функции гораздо мягче, чем, например, к функции, оптимизируемой при градиентном спуске [10]. Фитнес-функция может быть недифференцируемой, это самое интересное свойство генетической оптимизации, позволяющее легко создавать сложные фитнес-функции, совершенно не заботясь об их дифференцируемости. Существуют модификации генетической оптимизации, которые не требуют даже точного значения фитнес-функции. Таким алгоритмам нужны только ранги, т. е. отношения «больше-меньше» между фит-

нес-функциями индивидов, для сравнения их друг с другом. В этом случае лучшее решение имеет самый высокий (или самый низкий) ранг, но точное значение функции при этом может быть неизвестно.

Чтобы построить решение, необходимо выбрать такое представление признаков, которое может эффективно мутировать. Символическая (или символьная) регрессия [11, 12] – это подход к нахождению арифметического выражения, которое описывает закон, по которому производятся данные. Он может быть представлен в виде дерева выражений, где лист – это терминал (признак из исходных данных), а узел – операция над дочерними поддеревьями. Деревья могут быть закодированы различными способами: графы [13], обратная польская нотация [14], или менее известные методы, такие как генные выражения [15, 16], которые и используются в данной работе и будут рассмотрены далее.

В этой работе мы предлагаем прозрачный алгоритм уменьшения размеров под названием ГУРу (Генетическое Уменьшение Размерности). ГУРу является эволюционным алгоритмом и реализует общий конвейер генетического алгоритма. Он строит признаки, стремясь обеспечить линейную разделимость пространства признаков и максимизировать среднее расстояние между объектами различных классов.

Обзор существующих работ

Классические подходы строят непрозрачные решения, потому что они оперируют данными с позиции статистики и не используют никаких знаний о предметной области. Например, PCA [17] находит подпространство в исходном пространстве, в котором вдоль осей у данных наибольшая дисперсия. Для аналитика данных является сложной задачей объяснить значение таких признаков с точки зрения предметной области. Наиболее распространенным случаем решения задачи сокращения размерности является использование классических подходов, таких как PCA, когда признаков очень много и работать с таким большим количеством признаков просто невозможно, или t-SNE [18], когда необходимо сделать некоторые интуитивные выводы о распределении данных путем визуализации пространства признаков. С другой стороны, существует несколько работ, посвященных прозрачному сокращению размерности [19–22]. Они используют эволюционный подход для уменьшения размерности данных и получения интерпретируемых человеком признаков. Эти алгоритмы генерируют признаки в виде деревьев выражений, которые имеют тот же уровень интерпретируемости, что и исходные признаки, поскольку операции, применяемые к данным, легко интерпретировать. Поэтому, если имеется большой объем данных, который необходимо уменьшить без потери смысла, эти алгоритмы можно использовать вместо классических подходов, которые не могут уменьшить размерность данных без потери интерпретируемости и генерируют признаки, которые трудно объяснить с помощью терминологии предметной области.

Алгоритм GP-DR

В работе [20] авторы исследуют, насколько хорошо работает простой генетический алгоритм без настройки гиперпараметров (размер популяции и количество эпох) в задаче сокращения размерности. В статье авторы не дают ему названия, поэтому в данной работе назовем этот алгоритм GP-DR (Genetic Programming for Dimension Reduction). Существует несколько фитнес-функций: расстояние между объектами (А), сохранение ранга (на основе расстояния) (Б), функция, оценивающая качество дистилляции кодирующей части автоэнкодера (В), и качество дистилляции всего автоэнкодера (Г). Во всех случаях это алгоритм обучения без учителя.

Алгоритм (А) сравнивает расстояния между объектом в исходном высокоразмерном пространстве и его проекцией в сгенерированном низкоразмерном пространстве. Сумма таких расстояний должна быть минимизирована. В этом заключается суть фитнес-функции, основанной на расстоянии.

Фитнес-функция, сохраняющая ранг (Б), предназначена для сохранения порядка рангов расстояний. Для каждого объекта все остальные объекты могут быть отсортированы по расстоянию и ранжированы. В новом пространстве эти ранги должны быть такими же, как в исходном пространстве. Эта фитнес-функция предназначена для того, чтобы генетический алгоритм минимизировал количество неправильно упорядоченных пар объектов в низкоразмерном пространстве. Объекты с более высокими рангами более важны, чем объекты с более низкими рангами. Таким образом, в фитнес-функции используется взвешивание объектов в соответствии с их рангами.

Другая фитнес-функция (В) использует выходной сигнал кодирующей части автоэнкодера и предназначена для того, чтобы генетический алгоритм генерировал такой же выходной сигнал. Ее можно представить как задачу символьной регрессии на выходе кодировщика и исходном пространстве признаков в качестве цели.

Последняя фитнес-функция (Г) реализует тот же подход, но выход автоэнкодера в целом должен сравниваться с выходом генетического алгоритма. Деревья генетических алгоритмов делятся на деревья кодировщиков и деревья декодировщиков. При этом только кодирующие деревья используются для генерации низкоразмерного пространства после завершения подгонки алгоритма.

Алгоритм GP-MaL-MO

Этот генетический алгоритм основан на идее, что соседи в новом низкоразмерном пространстве должны иметь порядок, аналогичный порядку в исходном высокоразмерном пространстве. Этот алгоритм называется GP-MaL-MO (Genetic Programming for Manifold Learning using a Multi-objective Approach) [21], является расширением алгоритма GP-MaL [23]. Данный алгоритм использует многокритериальную функцию пригодности для построения фронта Парето из решений. Он использует многоцелевой эволюционный алгоритм на основе декомпозиции (MOEA/D) [24] для генерации решений. Алгоритм также генерирует решения в виде деревьев выражений с исходными признаками в листьях и операциями между ними в узлах дерева. Таким образом, GP-MaL-MO генерирует деревья выражений, которые могут быть интерпретированы человеком и могут быть использованы как прозрачный алгоритм уменьшения размерности.

Существует компромисс между качеством и количеством признаков (измерений). Таким образом, пользователь может выбрать решение, удовлетворяющее его требованиям. Алгоритм имеет операторы кроссинговера и мутации, разработанные для того, чтобы GP-MaL-MO мог генерировать решения как с одним деревом, так и со ста деревьями, что приводит к богатому фронту Парето на выходе алгоритма. Как показывают авторы, алгоритм работает так же хорошо, как и классические алгоритмы уменьшения размерности, такие как PCA, LLE и др. [25], MDS [26] и UMAP [27], и полученные с его помощью решения имеют сопоставимое качество.

Описание алгоритма ГУРУ

В ГУРУ реализовано несколько отличительных особенностей:

- Хромосома – это дерево вычислений, которое кодируется методом «генных выражений» [15, 16].
- Фитнес-функция является динамической и изменяется для каждого конструируемого признака.
- Конвейер алгоритма разделен на две части: фаза исследования и фаза эксплуатации.

Все эволюционные алгоритмы имеют схожую конструкцию. Это цикл, начинающийся с оценки стартовой популяции, выбора лучших особей, мутации, кроссинговера особей и создания новой популяции из их детей. Эта процедура повторяется, пока не срабатывают крите-

Таблица 1

Операторы, используемые в ГУРУ

Table 1

Operators Used in GURU

Операция	+	-	*	/	exp	ln	cos	модуль	среднее	max
Количество операндов	2	2	2	2	1	1	1	1	2	2

рии остановки. Выбор генетических операций влияет на стиль поиска алгоритма в пространстве поиска, его скорость и баланс между поведением разведки и эксплуатации.

Далее приводится подробное описание генетических операций и операндов в ГУРУ.

ГУРУ работает с индивидами, которые представляют собой деревья вычислений. Эти деревья содержат терминальные признаки в листьях и операции в узлах. Таким образом, если это дерево будет вычисляться, то на входе будет несколько терминалов, а на выходе – одна функция.

Алгоритм работает с числовыми данными, представленными столбцами входного датафрейма данных. Текущая реализация работает с датафреймами известной в среде аналитиков данных библиотеки pandas. Таким образом, столбцы исходного набора данных, содержащие числовые данные, являются терминальными признаками для алгоритма.

Популяция в ГУРУ – это массив определенного размера, содержащий индивидов. Ее размер меняется между фазами разведки и эксплуатации, но является константой для всех итераций в одной фазе.

В версии ГУРУ, представленной в данной работе, используются обычные арифметические операции из табл. 1. Они могут быть легко интерпретированы аналитиком (при условии, что в конкретной предметной области эти операции между признаками имеют смысл).

В алгоритме используются три типа мутации: инверсия, транспонирование последовательности вставки (IS), транспонирование корневой последовательности вставки (RIS).

- Инвертирующая мутация случайным образом выбирает подпоследовательность в массиве head и инвертирует ее.
- Мутация IS Transpose случайным образом выбирает подпоследовательность внутри головки и вставляет ее в другую позицию, кроме начальной.
- Мутация RIS Transpose делает то же самое, что и IS Transpose. Единственное отличие заключается в том, что RIS Transpose вставляет подпоследовательности только из корневой позиции.

В ГУРУ используются стратегии одноточечного и двухточечного кроссинговера.

- Одноточечный кроссинговер берет два гена, случайным образом выбирает точку и создает двух потомков, объединяя части родителей относительно выбранной точки.
- Двухточечный кроссинговер похож на одноточечный, но в нем выбираются две точки, и потомки строятся путем обмена материалами родителей между этими двумя точками.

В ГУРУ лучшие индивиды отбираются с помощью турнирного отбора. В нем случайным образом выбирается фиксированное число особей и сравниваются их показатели приспособленности. Лучшая особь из каждого турнира переходит в следующее поколение.

ГУРУ оценивает новые признаки по отношению к ранее созданным. Это заставляет алгоритм находить такие новые признаки, которые содержат информацию, не содержащуюся в ранее созданных признаках. Это приводит к построению выразительных компактных признаков. Стоит отметить, что каждый новый признак сам по себе все менее и менее важен, чем ранее сгенерированный, поскольку каждый новый признак зависит от всех предыдущих. Поэтому возможно, что какой-то признак высокого ранга (сгенерированный на поздней эпохе) не содер-

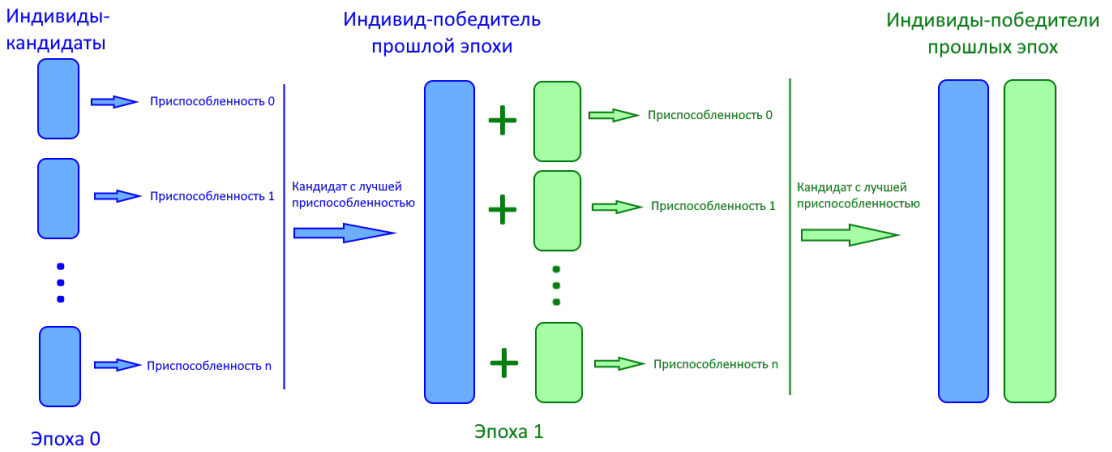

Рис. 1. Конвейер ГУРУ, генерирующий два признака
Fig. 1. GURU pipeline generating two features

жит никакой полезной информации для классификации, но содержит информацию об ошибках всех предыдущих признаков. Это может привести к чрезмерной подгонке (переобучению).

Первый признак оценивается с помощью подгонки к нему функции пригодности и 3-кратной кросс-валидации. Каждый следующий признак оценивается с помощью ранее созданных признаков. Например, второй признак оценивается с помощью оценивания функции пригодности на данных, которые содержат первый признак, полученный из предыдущей эпохи. Это позволяет ГУРУ генерировать признаки таким образом, что каждый новый признак исправляет недостатки всех ранее сгенерированных признаков. Данный процесс проиллюстрирован на рис. 1.

Программирование выражений генов – это разновидность генетического программирования. Отличительной особенностью генных выражений является линейное представление фиксированной длины. Такое представление может быть преобразовано в вычислительное дерево. Хромосома в генном выражении – это строка фиксированной длины. Она делится на головку и хвост, как показано на рис. 2. Голова – это список функций и терминалов постоянной длины. Хвост содержит только терминалы. Его длина зависит от того, сколько операндов требуется головным функциям. Листья дерева – это терминалы, а узлы – функции. Преимуществом генных выражений является простота реализации генетических операций (мутация, кроссинговер, оценка). Недостатком может быть недостаточная выразительность выражений из-за фиксированной длины, что с другой стороны также сокращает пространство поиска решений.

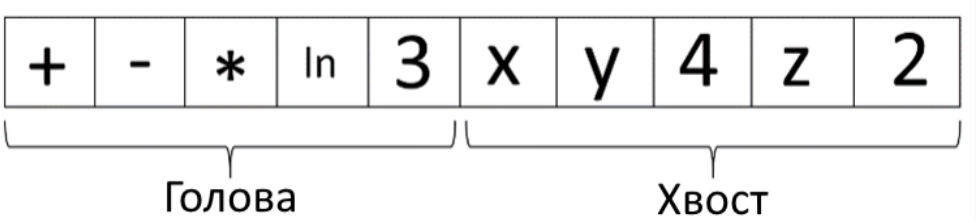

Рис. 2. Генное выражение
Fig. 2. Gene expression

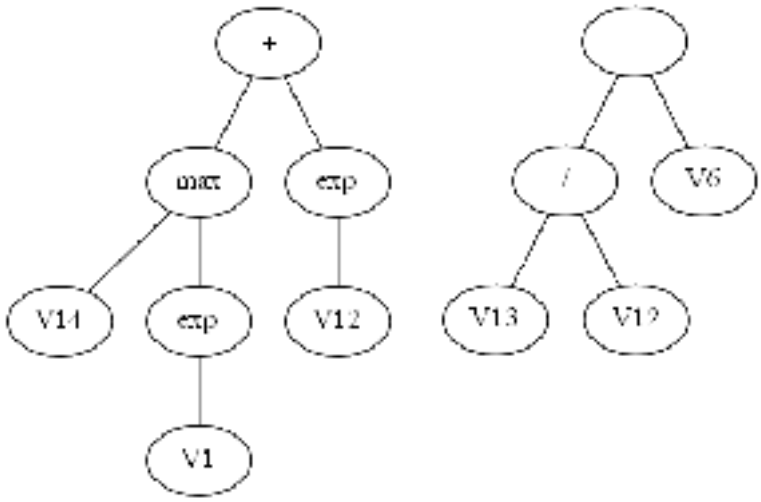

Рис. 3. Дерево, сгенерированное ГУРУ для набора данных «банковский маркетинг»
Fig. 3. Expression tree generated by GURU on “bank-marketing” dataset

Выражение гена однозначно отображается в дерево вычислений, как показано на рис. 3. Однако одно дерево может отображаться на более чем одно генное выражение, поскольку существуют генные выражения, содержащие неиспользуемые операции и терминалы.

Генные выражения – это один из многих методов кодирования дерева вычислений. Он прост в использовании и управлении: регулируя длину генного выражения, мы можем избежать чрезмерной подгонки и регулировать выразительность решения.

Функция пригодности оценивает качество особи. Это одна из самых важных частей генетического алгоритма, поскольку она определяет пространство поиска. Очень важно сделать фитнес-функцию, которая описывает решения с разных точек зрения, чтобы уменьшить вероятность перебора. В данной работе мы используем динамическую многокритериальную фитнес-функцию.

Многокритериальная фитнес-функция использует более одного критерия для оценки особи. В данной работе мы используем два критерия для каждой итерации генерации. Фитнес-функция представлена взвешенной суммой, как в (1). Один из критериев называется примесью, а другой – базовой фитнес-функцией. Базовая функция не изменяется в процессе работы алгоритма и является одной и той же для всех итераций алгоритма. Примесь изменяется на каждой итерации.

$$fitness = \frac{k_1 * fit_{\text{база}} + k_2 * fit_{\text{примесь}}}{k_1 + k_2} \quad (1)$$

В ГУРУ мы используем такую конфигурацию мультикритериальной фитнес-функции:

- Базовая функция – линейная машина опорных векторов [28].
- Примесь – это один из трех алгоритмов: логистическая регрессия, дерево решений или мера расстояния.

Интуиция, лежащая в основе такой мультикритериальной функции, заключается в том, чтобы построить различные признаки благодаря динамической примеси. И в то же время признаки должны строить линейно разделяемое пространство благодаря статической базовой функции, которая является линейной моделью. Такая комбинация приводит к стремлению алгоритмом создать линейно разделимое пространство даже при наличии нелинейной функции-примеси.

Фитнес-функция является динамической, поскольку алгоритм оценивает каждый новый признак с помощью другой фитнес-функции. Пример сокращения до двух измерений показан на рис. 1. Это делает пространство признаков результата более разнообразным и выразительным по сравнению со статической фитнес-функцией, когда существует одна и та же модель для оценки всех генерируемых признаков. В данной работе используются простые фитнес-функции из разных семейств алгоритмов:

- *Логистическая регрессия*, которая должна заставить алгоритм построить признак, делающтй пространство более линейно разделяемым.
- *Мера расстояния*, приводящая к построению признака, который смещает объекты разных классов на большее расстояние. Мы используем среднее попарное расстояние между объектами классов. Оно масштабируется по формуле (2) таким образом, что значение около 0,0 является наихудшим, а 1,0 – наилучшим:

$$distance_{\ scaled} = \frac{1 - e^{-distance}}{1 + e^{-distance}} \tag{2}$$

- *Дерево решений.* Это простая и быстрая модель, принцип работы которой сильно отличается от логистической регрессии и меры расстояния. Лучше использовать признаки, полученные с помощью нелинейной модели, чем с помощью другой линейной модели.

Компромисс между разведкой и эксплуатацией – известная дилемма в машинном обучении. Исследование – это такое поведение алгоритма, когда он совершает большие хаотические скачки в пространстве поиска и потенциально имеет возможность выскочить из локального экстремума и попасть в другой. Такое поведение увеличивает шансы найти глобальный экстремум. Тем не менее, разведка является плохой стратегией для поиска точной точки экстремума. Напротив, в случае поведения эксплуатации алгоритм делает небольшие шаги в пространстве поиска. Это помогает найти точную точку экстремума. Однако эксплуатирующий алгоритм, скорее всего, не найдет ни глобального, ни даже другого локального экстремума, а будет медленно и верно сходиться к тому экстремуму, возле которого находится.

Генетические алгоритмы также страдают от компромисса между разведкой и эксплуатацией. Популяция может содержать похожие особи, которые могут иметь малые вероятности мутации и кроссинговера. Алгоритм с такой конфигурацией будет реализовывать стратегию эксплуатации, поскольку особи меняются медленно. Поэтому алгоритм делает небольшие шаги в пространстве поиска. В других случаях вероятности мутации и кроссинговера достаточно высоки. Тогда алгоритм будет генерировать различные особи в популяции. Это поведение разведки, поскольку каждая особь имеет небольшой шанс выжить и дать потомство с похожими свойствами. Возможно, она мутирует и потеряет свои полезные свойства или потеряет их при скрещивании с другой особью.

В данной работе мы разделили процесс работы алгоритма на две фазы: разведка и эксплуатация. Идея заключается в том, чтобы дать алгоритму свободу в начале процесса поиска генерировать огромное количество различных особей. Они будут отсортированы по значению фитнес-функции, и алгоритм возьмет подмножество лучших особей. Затем алгоритм начнет фазу эксплуатации.

Сбор данных

Бенчмарк (набор данных для оценки качества работы алгоритмов) содержит 12 датасетов, загруженных с сайта проекта OpenML [29]: bank marketing [30], blood transfusion [31], breast cancer wisconsin [32], [33], credit-g [33], diabetes [33], hyperplane [29], ionosphere [34], madelon [35], sonar [36], bioresponse [29], christine [29], guillermo [29]. Была протестирована только задача бинарной классификации. Как видно из табл. 2, все наборы данных в целом имеют числовые признаки и только два набора данных имеют категориальные признаки, чтобы посмо-

треть, как алгоритмы будут работать на наборах данных, где важны категориальные признаки. Для проверки устойчивости алгоритмов есть несколько достаточно больших наборов данных, таких как hyperplane и guillermo. Баланс классов также различается, но нет наборов данных с огромным дисбалансом.

В наборе данных christine имеется 38 бинарных и унарных признаков. В данной работе мы используем их как числовые признаки, где False – 0.0, а True – 1.0.

В табл. 2 в колонке OpenML id представлены идентификаторы наборов данных на сайте.

Таблица 2

Набор данных, используемые в бенчмарке

Table 2

Datasets Used in the Benchmark

Название наборов данных	**OpenML id**	**#признаки числовые / категориальные**	**#образцы**	**Баланс классов**
bank marketing	1461	7/9	45211	7,5 = 39922/5289
blood transfusion	1464	4/0	748	3,2 = 570/178
breast cancer wisconsin	1510	30/0	569	1,7 = 357/212
credit-g	31	7/13	1000	2,3 = 700/300
diabetes	37	8/0	768	1,8 = 500/268
hyperplane	43122	10/0	500000	1,0 = 250000/250000
ionosphere	59	34/0	351	1,8 = 225/126
madelon	1485	500/0	2600	1,0 = 1300/1300
sonar	40	60/0	208	1,1 = 111/97
bioresponse	4134	1776/0	3751	1,2 = 2034/1717
christine	41142	1599/37	5418	1,0 = 2709/2709
guillermo	41159	4296/0	20000	1,5 = 11997/8003

Эксперимент

В этом разделе представлено сравнение с другими прозрачными алгоритмами уменьшения размерности. Все эксперименты запускались десять раз с различными случайными семенами из фиксированного набора. В бенчмарке наборы данных разбиваются на обучающий и тестовый наборы. Затем каждый алгоритм обучается на обучающем множестве. Настроенная модель преобразует как обучающий, так и тестовый наборы. Затем модель классификации обучается на преобразованном обучающем множестве и оценивается на преобразованном тестовом множестве. Используется метрика AUC-ROC. Все десять результатов, рассчитанных для каждого алгоритма на каждом наборе данных, усредняются, и этот средний балл используется для сравнения алгоритмов.

Конфигурация ГУРУ была одинаковой во всех бенчмарках. Она представлена в табл. 3. Гиперпараметры вероятности кроссинговера и мутации были предварительно настроены на бенчмарке. Все остальные гиперпараметры были определены интуитивно. Таким образом, возможно, что ГУРУ может достичь лучшего качества в другой конфигурации, что является предметом отдельного исследования.

Таблица 3

Конфигурация ГУРУ

Table 3

Configuration of GURU

Параметр	**Значение**
случайные состояния	[72, 73, 74, ..., 81]
режим слияния	категории
выходные признаки	2
поколения	21
население	1600
режим работы функций терминала	все
вероятность мутации	0,25
вероятность кроссинговера	0,25
размер турнира	4
максимальный размер выборки	8192
генное выражение длины головы	8
операции	+, -, *, /, exp, ln, cos, abs, mean, max
базовая фитнес-функция	Линейная SVM
фитнес-функции-примеси	[Логистическая регрессия, расстояние]

GP-DR и GP-MaL-MO были адаптированы для выполнения в нашем бенчмарке (это означает, что они должны реализовать интерфейс трансформера из библиотеки sklearn). После адаптации они были запущены в бенчмарке с конфигурациями по умолчанию. Единственное отличие в гиперпараметрах – это количество выходных признаков, которое было изменено на 2 там, где это было возможно.

Результатом GP-MaL-MO является фронт Парето с компромиссом между качеством решения и количеством измерений. В этом эксперименте использовано лучшее решение с двумя измерениями, если оно существовало, и объединение двух решений с одним измерением в противном случае.

В этом эксперименте мы сравниваем метрики линейной модели классификации (логистической регрессии), обученной и протестированной на признаках, сгенерированных различными алгоритмами уменьшения размерности. Все алгоритмы строят двумерное пространство признаков, поскольку это наиболее распространенный случай в реальной жизни, когда уменьшение размерности используется для того, чтобы сделать пространство признаков пригодным для визуализации (трехмерное пространство также может быть визуализировано, но построить понятную изометрическую визуализацию – более сложная задача). Линейная модель используется потому, что качество ее предсказаний показывает, насколько хорошо пространство признаков может быть линейно разделено. Линейно разделяемое пространство удобно для визуального анализа. Кроме того, линейные зависимости в целом более интерпретируемы и понятны человеку.

Генетические алгоритмы преобразуют пространство признаков с помощью преобразования, имеющего наибольшее значение фитнес-функции.

GP-DR не очень хорошо работает с достаточно большими наборами данных. Именно поэтому некоторые результаты для этого алгоритма отсутствуют. GP-MaL-MO также не может завершить вычисления на больших наборах данных.

Таблица 4

Сравнение прозрачных алгоритмов понижения размерности

Table 4

Comparison of the Transparent Dimension Reduction Algorithms

Набор данных	Baseline	GP-MaL-MO	GP-DR sammon euclid	GP-DR rank euclid	GP-DR sammon isomap	GP-DR rank isomap	GP-DR AE teacher	GP-DR AE fit	ГУРУ	Победитель
bank marketing	**0,589**	–	0,514	0,507	0,515	0,503	0,522	0,541	0,548	Baseline
bioresponse	0,514	0,571	0,500	0,500	0,507	0,508	0,498	–	**0,727**	ГУРУ
blood trans	0,500	0,509	0,523	0,513	0,517	0,517	0,518	0,523	**0,568**	ГУРУ
breast cancer wisconsin	0,867	0,851	0,785	0,702	0,774	0,649	0,886	0,870	**0,918**	ГУРУ
christine	0,583	0,574	0,527	0,530	0,529	0,534	0,577	–	**0,667**	ГУРУ
credit-g	**0,595**	0,562	0,571	0,560	0,571	0,560	0,581	0,581	**0,528**	Baseline
diabetes	0,707	0,604	0,552	0,546	0,624	0,548	0,610	0,579	**0,714**	ГУРУ
guillermo	0,507	–	–	–	–	–	–	–	**0,525**	ГУРУ
hyperplane	0,642	–	–	–	–	–	–	–	**0,672**	ГУРУ
ionosphere	0,653	0,578	0,607	0,583	0,579	0,552	0,544	0,559	**0,739**	ГУРУ
madelon	0,556	0,564	0,497	0,500	0,508	0,501	0,553	–	**0,602**	ГУРУ
sonar	**0,676**	0,583	–	–	–	–	0,589	0,602	**0,605**	Baseline

В качестве базового решения в этом эксперименте использовался отбор признаков с помощью линейной модели (величина коэффициентов логистической регрессии), поскольку выбор признаков можно рассматривать как примитивное прозрачное уменьшение размерности.

Анализ результатов

Как видно из табл. 4, ГУРУ превосходит другие генетические алгоритмы в этом эксперименте почти на всех наборах данных. Однако стоит отметить результаты на наборе данных credit-g, где ГУРУ показывает худший результат среди всех алгоритмов. Причиной этого, вероятно, является нетривиальная комбинация числовых и категориальных признаков, содержащих важную для классификации информацию. ГУРУ генерирует признаки без каких-либо знаний о категориальных признаках. Это приводит к эффектам, подобным этому, на тех наборах данных, где важна комбинация числовых и категориальных признаков.

Вероятно, такая же ситуация и с набором данных bank-marketing, поскольку в случае трехмерного пространства качество значительно лучше. В случае с набором данных sonar трудно делать предположения, потому что этот набор данных небольшой и имеет всего 208 объектов, и несколько других алгоритмов на таком малом датасете вообще не смогли успешно отработать.

Также стоит отметить, что ГУРУ – единственный из генетических алгоритмов смог успешно отработать на больших по количеству признаков датасетах guillermo и hyperplane.

Заключение

В этой работе мы попытались создать прозрачное решение для уменьшения размерности и представили ГУРУ. Алгоритм обеспечивает построение линейно сепарабельного низкоразмерного пространства признаков посредством конструирования признаков. Он работает с числовыми признаками, а сгенерированные признаки представлены в виде выражений генов, являющихся деревьями выражений, которые могут быть интерпретированы человеком. ГУРУ использует динамическую многокритериальную фитнес-функцию для оценки особей, которая позволяет строить разнообразные решения. Чтобы справиться с компромиссом между исследованием и эксплуатацией, конвейер алгоритмов разделен на две фазы: короткая фаза исследования с популяциями большого размера в начале и длинная фаза эксплуатации со значительно меньшими популяциями в дальнейшем. Эксперименты показали, что ГУРУ обеспечивает лучшее качество классификации по линейной модели среди аналогов в случае уменьшения размерности до двух измерений. Стоит отметить, что ГУРУ может быть использован как алгоритм инженерии признаков, который строит признаки, обеспечивающие высокое качество классификации и поддающиеся интерпретации. Он может быть использован аналитиком для генерации большого набора различных признаков и поиска среди них значимых, которые могут привести к пониманию данных. Исследование такого применения является одним из направлений будущей работы.

Список литературы

1. **M. H. ur Rehman, C. S. Liew, A. Abbas, P. P. Jayaraman, T. Y. Wah, S. U. Khan.** Big Data Reduction Methods: A Survey. Data Sci. Eng., 2016, vol. 1, no. 4, p. 265–284, DOI: 10.1007/s41019-016-0022-0.
2. **C. H. Yoon, R. Torrance, N. Scheinerman.** Machine learning in medicine: should the pursuit of enhanced interpretability be abandoned? J. Med. Ethics, 2022, vol. 48, no. 9, p. 581–585, DOI: 10.1136/medethics-2020-107102.

3. **P. Linardatos, V. Papastefanopoulos, S. Kotsiantis.** Explainable AI: A Review of Machine Learning Interpretability Methods. Entropy, 2021, vol. 23, no. 1, Art. no. 1, DOI: 10.3390/e23010018.
4. **Izenman.** Introduction to manifold learning. Wiley Interdiscip. Rev. Comput. Stat., 2012, vol. 4, DOI: 10.1002/wics.1222.
5. **H. Han, W. Li, J. Wang, G. Qin, X. Qin**. Enhance Explainability of Manifold Learning. Neurocomputing, 2022, vol. 500, DOI: 10.1016/j.neucom.2022.05.119.
6. **D. Elizondo, R. Birkenhead, M. Gámez, N. Rubio, E. Alfaro-Cortés.** Linear separability and classification complexity. Expert Syst. Appl., 2012, vol. 39, p. 7796–7807, DOI: 10.1016/j.eswa.2012.01.090.
7. **J. Koza, R. Poli.** Genetic Programming. in Search Methodologies, 2005, p. 127–164. DOI: 10.1007/0-387-28356-0_5.
8. **U.-M. O'Reilly, E. Hemberg.** Genetic programming: a tutorial introduction. 2021, p. 453. DOI: 10.1145/3449726.3461394.
9. **Vasuki.** Genetic Programming. 2020, p. 61–76. DOI: 10.1201/9780429289071-5.
10. **L. Kallel, B. Naudts, C. Reeves.** Properties of Fitness Functions and Search Landscapes. 2000, DOI: 10.1007/978-3-662-04448-3_8.
11. **M. Schmidt, H. Lipson.** Distilling Free-Form Natural Laws from Experimental Data. Science, 2009, vol. 324, no. 5923, p. 81–85, DOI: 10.1126/science.1165893.
12. **W. La Cava et al.** Contemporary Symbolic Regression Methods and their Relative Performance. in Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 2021, vol. 1.
13. L**. Sotto, P. Kaufmann, T. Atkinson, R. Kalkreuth, M. Basgalupp.** Graph representations in genetic programming. Genet. Program. Evolvable Mach., 2021, vol. 22, DOI: 10.1007/s10710-021-09413-9.
14. **P. Krtolica, P. Stanimirovic.** Reverse Polish notation method. Int. J. Comput. Math., 2004, vol. IJCM, p. 273–284, DOI: 10.1080/00207160410001660826.
15. **C. Ferreira.** Gene Expression Programming: a New Adaptive Algorithm for Solving Problems. arXiv, 2001. DOI: 10.48550/arXiv.cs/0102027.
16. **C. Ferreira.** Gene Expression Programming in Problem Solving. in Soft Computing and Industry: Recent Applications, Eds. London: Springer, 2002, p. 635–653. DOI: 10.1007/978-1-4471-0123-9_54.
17. **Jolliffe.** Principal Component Analysis. Springer: Berlin, Germany, 1986, vol. 87, p. 41–64, DOI: 10.1007/b98835.
18. **L. van der Maaten, G. Hinton.** Visualizing data using t-SNE. Journal of Machine Learning Research, 2008, vol. 9, p. 2579–2605.
19. **B. Hosseini, B. Hammer.** Interpretable Discriminative Dimensionality Reduction and Feature Selection on the Manifold. arXiv, arXiv:1909.09218, 2019 DOI: 10.48550/arXiv.1909.09218.
20. **T. Uriot, M. Virgolin, T. Alderliesten, P. Bosman.** On genetic programming representations and fitness functions for interpretable dimensionality reduction. arXiv, arXiv:2203.00528, 2022. DOI: 10.48550/arXiv.2203.00528.
21. **Lensen, M. Zhang, B. Xue.** Multi-Objective Genetic Programming for Manifold Learning: Balancing Quality and Dimensionality. Genet. Program. Evolvable Mach., 2020, vol. 21, no. 3, p. 399–431, DOI: 10.1007/s10710-020-09375-4.
22. **M. Virgolin, T. Alderliesten, P. A. N. Bosman.** On Explaining Machine Learning Models by Evolving Crucial and Compact Features. Swarm Evol. Comput., 2020 vol. 53, p. 100640, DOI: 10.1016/j.swevo.2019.100640.
23. **Lensen, B. Xue, M. Zhang.** Can Genetic Programming Do Manifold Learning Too? in Genetic Programming, Cham, 2019, p. 114–130. doi: 10.1007/978-3-030-16670-0_8.

24. **Q. Zhang, H. Li.** MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. IEEE Trans. Evol. Comput., 2007, vol. 11, no. 6, p. 712–731, DOI: 10.1109/TEVC.2007.892759.
25. **S. Roweis, L. Saul.** Nonlinear Dimensionality Reduction by Locally Linear Embedding. Science, 2001, vol. 290, p. 2323–6, DOI: 10.1126/science.290.5500.2323.
26. **B. K. Tripathy, S. Anveshrithaa, S. Ghela.** Multidimensional Scaling (MDS). 2021, p. 41–51. DOI: 10.1201/9781003190554-6.
27. **L. McInnes, J. Healy, J. Melville.** UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv, Sep. 17, 2020. DOI: 10.48550/arXiv.1802.03426.
28. **C. Cortes, V. Vapnik.** Support vector machines. Mach. Learn., 1995, vol. 20, p. 273–293.
29. **Vanschoren, J. N. van Rijn, B. Bischl, L. Torgo.** OpenML: networked science in machine learning. ACM SIGKDD Explor. Newsl., 2014, vol. 15, no. 2, p. 49–60, DOI: 10.1145/2641190.2641198.
30. **S. Moro, P. Cortez, R. Laureano.** Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. 2011.
31. **Yeh, K.-J. Yang, T.-M. Ting.** Knowledge discovery on RFM model using Bernoulli sequence. Expert Syst Appl, 2009, vol. 36, p. 5866–5871, DOI: 10.1016/j.eswa.2008.07.018.
32. **Bennett, O. L. Mangasarian.** Robust Linear Programming Discrimination Of Two Linearly Inseparable Sets. Optim. Methods Softw., 2002, vol. 1, DOI: 10.1080/10556789208805504.
33. **D. Dua, C. Graff.** UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2019. [Online]. Available: http://archive.ics.uci.edu/ml
34. **V. Sigillito, S. Wing, L. Hutton, K. Baker.** Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Tech. Dig. Appl. Phys. Lab., 1989, vol. 10.
35. **Guyon, S. Gunn, A. Ben-Hur, G. Dror.** Result Analysis of the NIPS 2003 Feature Selection Challenge, vol. 17. 2004.
36. **R. P. Gorman, T. Sejnowski.** Analysis of hidden units in a layered network trained to classify sonar targets. Neural Netw., 1988, vol. 1, p. 75–89, DOI: 10.1016/0893-6080(88)90023-8

References

1. **ur Rehman M. H., Liew C. S., Abbas A., Jayaraman P. P., T Wah. Y., Khan S. U.** Big Data Reduction Methods: A Survey // Data Sci. Eng. 2016. Vol. 1, no. 4. Pp. 265–284. DOI 10.1007/s41019-016-0022-0
2. **Yoon C. H., Torrance R., Scheinerman N..** Machine learning in medicine: should the pursuit of enhanced interpretability be abandoned? // J. Med. Ethics. 2022. Vol. 48, no. 9. Pp. 581–585. DOI 10.1136/medethics-2020-107102
3. **Linardatos P., Papastefanopoulos V., Kotsiantis S.** Explainable AI: A Review of Machine Learning Interpretability Methods // Entropy. 2021. Vol. 23, no. 1. Art. 1. DOI 10.3390/e23010018
4. **Izenman.** Introduction to manifold learning // Wiley Interdiscip. Rev. Comput. Stat. 2012. Vol. 4. DOI 10.1002/wics.1222
5. **Han H., Li W., Wang J., Qin G., Qin X..** Enhance Explainability of Manifold Learning // Neurocomputing. 2022. Vol. 500. DOI 10.1016/j.neucom.2022.05.119
6. **Elizondo D., Birkenhead R., Gámez M., Rubio N., Alfaro-Cortés E.** Linear separability and classification complexity // Expert Syst. Appl. 2012. Vol. 39. Pp. 7796–7807. DOI 10.1016/j.eswa.2012.01.090
7. **Koza J., Poli R..** Genetic Programming / Search Methodologies, 2005. Pp. 127–164. DOI 10.1007/0-387-28356-0_5
8. **O'Reilly U.-M., Hemberg E.** Genetic programming: a tutorial introduction. 2021, p. 453. DOI 10.1145/3449726.3461394
9. **Vasuki.** Genetic Programming. 2020. Pp. 61–76. DOI 10.1201/9780429289071-5
10. **Kallel L., Naudts B., Reeves C.** Properties of Fitness Functions and Search Landscapes. 2000. DOI 10.1007/978-3-662-04448-3_8

11. **Schmidt M., Lipson H.** Distilling Free-Form Natural Laws from Experimental Data // Science. 2009. Vol. 324, no. 5923. Pp. 81–85. DOI 10.1126/science.1165893
12. **La Cava W. et al.** Contemporary Symbolic Regression Methods and their Relative Performance / Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks, 2021, vol. 1.
13. **Sotto L., Kaufmann P., Atkinson T., Kalkreuth R., Basgalupp M.** Graph representations in genetic programming // Genet. Program. Evolvable Mach., 2021, vol. 22. DOI 10.1007/s10710-021-09413-9
14. Kr**tolica P., Stanimirovic P..** Reverse Polish notation method // Int. J. Comput. Math., 2004, vol. IJCM, p. 273–284. DOI 10.1080/00207160410001660826
15. **Ferreira C.** Gene Expression Programming: a New Adaptive Algorithm for Solving Problems. arXiv, 2001. DOI 10.48550/arXiv.cs/0102027
16. **Ferreira C.** Gene Expression Programming in Problem Solving. in Soft Computing and Industry: Recent Applications. London: Springer, 2002, pp. 635–653. DOI 10.1007/978-1-4471-0123-9_54
17. **Jolliffe.** Principal Component Analysis. Springer: Berlin, Germany, 1986. Vol. 87, pp. 41–64. DOI 10.1007/b98835
18. **van der Maaten L., Hinton G.** Visualizing data using t-SNE // Journal of Machine Learning Research, 2008. Vol. 9, pp. 2579–2605.
19. **Hosseini B., Hammer B.** Interpretable Discriminative Dimensionality Reduction and Feature Selection on the Manifold. arXiv, arXiv:1909.09218, 2019 doi: 10.48550/arXiv.1909.09218.
20. **Uriot T., Virgolin M., Alderliesten T., Bosman P.** On genetic programming representations and fitness functions for interpretable dimensionality reduction. arXiv, arXiv:2203.00528, 2022. DOI 10.48550/arXiv.2203.00528
21. **Lensen, M., Zhang, B. Xue.** Multi-Objective Genetic Programming for Manifold Learning: Balancing Quality and Dimensionality. Genet. Program. Evolvable Mach. 2020. Vol. 21, no. 3, pp. 399–431. DOI 10.1007/s10710-020-09375-4
22. **Virgolin M., Alderliesten T., Bosman P. A. N.** On Explaining Machine Learning Models by Evolving Crucial and Compact Features. Swarm Evol. Comput. 2020. vol. 53, p. 100640. DOI 10.1016/j.swevo.2019.100640
23. **Lensen, B. Xue, M. Zhang.** Can Genetic Programming Do Manifold Learning Too? / Genetic Programming, Cham, 2019, p. 114–130. DOI 10.1007/978-3-030-16670-0_8
24. **Zhang Q., Li H.** MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition // IEEE Trans. Evol. Comput., 2007, vol. 11, no. 6, p. 712–731. DOI 10.1109/TEVC.2007.892759
25. **Roweis S., Saul L..** Nonlinear Dimensionality Reduction by Locally Linear Embedding // Science, 2001, vol. 290, p. 2323–6. DOI 10.1126/science.290.5500.2323
26. **Tripathy B. K., Anveshrithaa S., Ghela S.** Multidimensional Scaling (MDS). 2021, p. 41–51. DOI 10.1201/9781003190554-6
27. **McInnes L., Healy J., Melville J.** UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. arXiv, Sep. 17, 2020. DOI 10.48550/arXiv.1802.03426
28. **Cortes C., Vapnik V.** Support vector machines // Mach. Learn., 1995, vol. 20, pp. 273–293.
29. **Vanschoren J., van Rijn N., Bischl B., Torgo L.** OpenML: networked science in machine learning. ACM SIGKDD Explor. Newsl., 2014, vol. 15, no. 2, p. 49–60. DOI 10.1145/2641190.2641198
30. **Moro S., Cortez P., Laureano R.** Using Data Mining for Bank Direct Marketing: An Application of the CRISP-DM Methodology. 2011.
31. **Yeh, Yang K.-J., Ting T.-M.** Knowledge discovery on RFM model using Bernoulli sequence. Expert Syst Appl, 2009, vol. 36, p. 5866–5871. DOI 10.1016/j.eswa.2008.07.018
32. **Bennett, Mangasarian O. L..** Robust Linear Programming Discrimination of Two Linearly Inseparable Sets. Optim. Methods Softw., 2002, vol. 1. DOI 10.1080/10556789208805504

33. **Dua D., Graff C.** UCI Machine Learning Repository. University of California, Irvine, School of Information and Computer Sciences, 2019. [Online]. URL: http://archive.ics.uci.edu/ml.
34. **Sigillito V., Wing S., Hutton L., Baker K.** Classification of radar returns from the ionosphere using neural networks. Johns Hopkins APL Tech. Dig. Appl. Phys. Lab., 1989, vol. 10.
35. **Guyon, Gunn S., Ben-Hur A., Dror G.** Result Analysis of the NIPS 2003 Feature Selection Challenge, 2004. Vol. 17.
36. **Gorman R. P., Sejnowski T.** Analysis of hidden units in a layered network trained to classify sonar targets. Neural Netw., 1988, vol. 1, p. 75–89. DOI 10.1016/0893-6080(88)90023-8

Информация об авторе

Радеев Никита Андреевич, магистрант кафедры общей информатики факультета информационных технологий НГУ

Information about the Author

Nikita A. Radeev, master of General Informatics Department, faculty of Information Technologies, Novosibirsk state University (Novosibirsk, Russian Federation)

Статья поступила в редакцию 29.03.2023;
одобрена после рецензирования 25.04.2023; принята к публикации 25.04.2023
The article was submitted 29.03.2023;
approved after reviewing 25.04.2023; accepted for publication 25.04.2023