

УДК 004.9

МЕТОДЫ БИНАРИЗАЦИИ АЛГОРИТМА СТАИ ЛАСТОЧЕК ДЛЯ РЕШЕНИЯ ЗАДАЧИ ОТБОРА ПРИЗНАКОВ

© 2021 г. А. О. Слезкин^{а,*}, И. А. Ходашинский^{а,**}, А. А. Шелупанов^{а,***}

^а Томский государственный университет систем управления и радиоэлектроники,
634050 Томск, проспект Ленина, д. 40, Россия

* E-mail: saotom724@gmail.com

** E-mail: hodashn@rambler.ru

*** E-mail: saa@tusur.ru

Поступила в редакцию 05.02.2021 г.

После доработки 16.03.2021 г.

Принята к публикации 12.05.2021 г.

Предложены шесть методов бинаризации алгоритма стаи ласточек для решения задачи отбора признаков по методу обертки. Эффективность выбранных подмножеств признаков оценивается двумя классификаторами: нечетким классификатором и классификатором на основе k -ближайших соседей. При поиске оптимального подмножества признаков учитывались количество признаков и точность классификации. Разработанные алгоритмы протестированы на наборах данных из репозитория KEEL. Для статистической оценки методов бинаризации использовался двухфакторный дисперсионный анализ Фридмана для связанных выборок. Лучшие способности к отбору признаков показал гибридный метод, основанный на методе модифицированных алгебраических операций и введенной нами операции MERGE. Лучшая точность классификации получена с использованием метода V-образной функции трансформации.

DOI: 10.31857/S013234742105006X

1. ВВЕДЕНИЕ

Отбор признаков — одна из важнейших тем исследований в области интеллектуального анализа данных и машинного обучения. Удаление нерелевантных и избыточных признаков позволяет уменьшить размерность исходных данных, ускорить процесс обучения, упростить разрабатываемую модель, повысить ее эффективность и обобщающие возможности, уменьшить риск переобучения.

Основная цель отбора признаков — исключить из набора данных признаки, которые слабо или даже негативно влияют на определение метки класса классификатором. Избавление от таких признаков, позволит как снизить сложность получаемой модели, так и, возможно, повысить точность классификации. Таким образом, отбор признаков можно рассматривать как двухкритериальную задачу. Однако часто указанная двухкритериальная задача сводится к однокритериальной задаче введением единой целевой функции, в которой с помощью весовых коэффициентов учтены ошибка классификации и отобранное число признаков.

Методы отбора признаков разделяются на три группы: фильтры, обертки и встроенные методы. Фильтры используют только сами данные, не

требуя построения классификатора. Преимущество этой группы методов состоит в низкой вычислительной сложности. Обертки требуют построения классификатора и включения его в процесс отбора признаков. По этой причине, получаемые результаты по качеству часто превосходят результаты, полученные методами фильтра, однако при этом возрастает и вычислительная сложность [1]. Встраиваемые методы интегрируются в процесс построения классификатора и производят отбор признаков в процессе его обучения.

В статье [2] авторы формулируют и доказывают теорему о NP-трудности проблемы отбора признаков, сводя эту проблему к задаче максимального покрытия, которая является NP-трудной. Таким образом, нахождение оптимального решения проблемы отбора признаков может быть гарантировано только полным перебором. Однако в современных наборах данных размер признакового пространства может достигать нескольких сотен, тысяч и даже больше [3]. Полный перебор средствами современных вычислительных средств является затратной или даже невыполнимой задачей. Для решения этой проблемы можно использовать метаэвристики, которые

позволяют найти оптимальное либо субоптимальное решение за приемлемое время.

Кратко остановимся на понятии “метаэвристика”, но сначала дадим рабочее определение понятию эвристика — это способ нахождения “достаточно хороших” решений без доказательства оптимальности найденных решений. Качество или эффективность решения могут быть выражены через точность, стабильность, экономию памяти или времени. Термин “мета” указывает на обобщенность и “методологию верхнего уровня”. Как правило, эвристики предназначены для решения конкретных задач, в то время как метаэвристики независимы от решаемых проблем. По мнению Шона Люка [4], метаэвристика — общее, но неудачное название стохастического алгоритма оптимизации, который применяется в качестве последней надежды на пути к решению задачи оптимизации. Применяются метаэвристики в том случае, когда 1) *неизвестен* способ поиска оптимального решения; 2) *недостаточно* имеющейся априорной информации; 3) *невозможен* полный перебор; 4) *можно* проверить качество решения, т.е. неизвестна аналитическая форма целевой функции, но ее значение может быть вычислено для конкретного решения.

Метаэвристики не используют вычисление производных целевой функции, однако, в отличие от детерминированных методов оптимизации, метаэвристики позволяют находить за приемлемое время удовлетворительные решения задач большой размерности. Широкому применению метаэвристики способствует увеличение вычислительной мощности компьютеров и развитие параллельных архитектур [5].

Отбор признаков ведется в бинарном пространстве поиска, когда возможны два состояния — признак используется или не используется в классификаторе. Большинство метаэвристических алгоритмов изначально разрабатывалось для работы в непрерывном пространстве поиска, которое задается множеством действительных чисел на определенном интервале для каждого признака; бинарное пространство поиска задается только двумя возможными значениями для каждого признака — 0 или 1. Поэтому для решения задачи отбора признаков алгоритмы, работающие в непрерывном пространстве поиска, необходимо адаптировать для работы в бинарном пространстве и разработать систему кодирования решения, сохранив при этом изначальную идею алгоритма. В процессе адаптации возникает ряд трудностей: *пространственная разобщенность*, возникающая, когда близкие решения, генерируемые метаэвристикой в непрерывном пространстве, не преобразуются в близкие решения в дискретном пространстве; *параметры по умолчанию*, найден-

ные для непрерывной метаэвристики, не всегда оптимальны для бинарной метаэвристики [6].

Предметом нашего исследования является метаэвристический алгоритм стаи ласточек (АСЛ). Алгоритм основан на поведенческих особенностях стаи ласточек. Он имеет некоторые общие черты с алгоритмом роящихся частиц (АРЧ) [7], но есть и отличия. В АСЛ популяция делится на субпопуляции, в которых есть свои частицы-лидеры или лучшие решения. Скорость частиц обновляется, как в АРЧ, но в обновлении учитывается позиция частицы-лидеры субпопуляции. Существенным отличием является наличие в АСЛ случайно сгенерированных решений, что позволяет АСЛ успешно осуществлять процесс диверсификации и успешнее преодолевать локальные оптимумы. Еще одним преимуществом АСЛ является тот факт, что во время одной итерации не изменяют свое положение частицы-лидеры. По этой причине снижается вычислительная сложность алгоритма, по сравнению с другими метаэвристиками, во время итерации которых изменяется положение всех частиц, как например АРЧ.

АСЛ доказал свою высокую эффективность при поиске оптимума мультимодальных функций. Алгоритм был протестирован на 19 функциях, показав высокую скорость сходимости, редкое застревание в точках локальных экстремумов и быстрое перемещение на участках со слабым изменением целевой функции. Результаты сравнивались со стандартным АРЧ, алгоритмом рыб и десятью различными модификациями АРЧ [8].

Адаптированная дискретная версия АСЛ использовалась авторами [9, 10] для решения задач комбинаторной оптимизации и по утверждению авторов оказалась более эффективной по сравнению с другими протестированными алгоритмами. В [11] описана бинарная версия АСЛ, выполненная с помощью разработанной авторами операции MERGE. Результаты экспериментов, выполненные на 30 наборах данных, показали перспективность предлагаемого метода для отбора признаков.

Метод на основе обертки ищет оптимальное подмножество признаков, адаптированное к конкретному классификатору и набору данных. В нашем исследовании используются два различных метода классификации: метрический — k -NN [12] и основанный на правилах — нечеткий классификатор [13, 14]. Эти классификаторы были выбраны потому, что они широко используются в машинном обучении, и, самое главное, решение каждого из них может быть интерпретируемо. В k -NN выбор решения обосновывается через прецеденты, в нечетком классификаторе — посредством лингвистического правила, нечеткие термы которого можно интерпретировать как нечеткие ве-

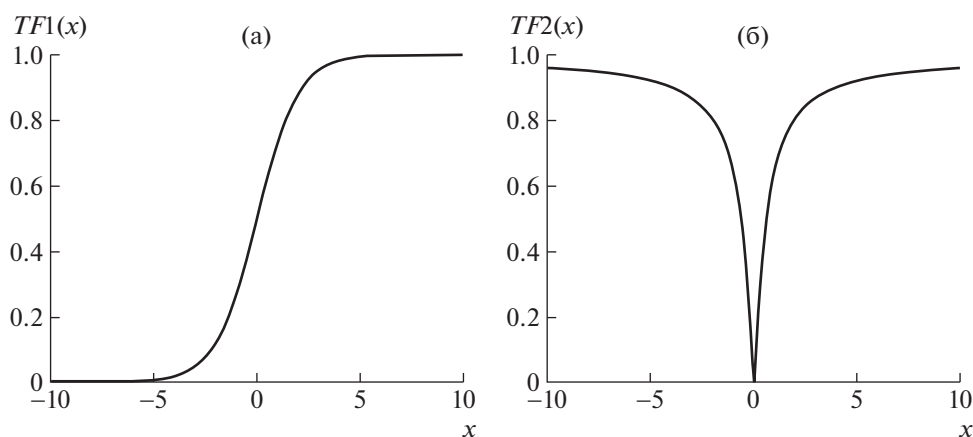


Рис. 1. Трансформационные функции: (а) S-образная, (б) V-образная.

личины, например, очень маленькие, маленькие, средние, большие, очень большие [15, 16].

Основная цель этой статьи — исследовать методы бинаризации метаэвристического алгоритма стаи ласточек для решения задач отбора признаков по методу обертки.

Метод обертки включает в себя два основных компонента: собственно метод отбора признаков и классификатор, который оценивает эффективность отобранных признаков. Предметом исследования в нашей статье является метод бинаризации АСЛ, классификатор в нашем исследовании играет вспомогательную роль и не является предметом исследования.

2. БЛИЗКИЕ РАБОТЫ

Принято выделять две основные группы методов бинаризации. Первая группа получила название “поэтапная бинаризация”. Методы этой группы позволяют работать с непрерывными метаэвристиками без модификации их операторов; здесь на каждой итерации вещественные векторы решений преобразуются в двоичные значения, используя заданные правила бинаризации. Вторая группа методов основана на введении оператора, выполняющая преобразование из непрерывного пространства поиска в бинарное пространство.

2.1. Поэтапная бинаризация

Метод ближайшего целого — простейшая одноэтапная процедура бинаризации, которая состоит в присвоении каждому элементу бинарного решения b ближайшего целого числа (0 или 1), полученного путем округления элемента непрерывного решения x по формуле:

$$b = \lfloor |x \bmod 2| \rfloor.$$

Авторы [17] использовали этот метод для бинаризации алгоритма дифференциальной эволюции. В работе [18] указанный метод применялся для бинаризации непрерывного алгоритма светлячков.

В двухэтапном методе бинаризация выполняется в два этапа, на этапе нормализации непрерывное решение $\mathfrak{R}^N$ трансформируется в решение $[0, 1]^N$, на втором этапе по заранее определенному правилу выполняется преобразование решения $[0, 1]^N$ в двоичное представление $\{0, 1\}^N$ [19]. В рамках этого метода рассмотрим метод трансформационных функций и метод угловой модуляции.

Метод трансформационных функций основывается на идее преобразования непрерывного решения к бинарному виду с помощью некоторой трансформационной функции, которая определяет вероятность изменения отдельной позиции решения и формирует промежуточное решение $[0, 1]^N$. Различают два типа трансформационных функций: S-образные и V-образные. На рисунке 1 представлены графики типичных представителей этих функций, S-образная функция описана формулой:

$$TF1(x) = \frac{1}{1 + e^{-x}},$$

V-образная функция задается выражением:

$$TF2(x) = \left| \frac{2}{\pi} \arctan\left(\frac{\pi}{2} x\right) \right|.$$

В [20] предложена линейная функция:

$$L(x) = \frac{x - R_{\min}}{R_{\max} - R_{\min}},$$

где $[R_{\min}, R_{\max}]$ — диапазон изменения действительной переменной x .

На втором этапе промежуточное решение, полученное на предыдущем этапе, по заданному правилу преобразуется в бинарное решение. В [22] для S-

образной функции приведено правило бинаризации следующего вида:

$$b = \begin{cases} 1 & \text{если } rand < TF1(x) \\ 0 & \text{иначе} \end{cases}.$$

Для линейной функции авторы [7] вводят аналогичное правило:

$$b = \begin{cases} 1 & \text{если } rand < L(x) \\ 0 & \text{иначе} \end{cases}.$$

Авторы [23] вместо случайно сгенерированного числа $rand$ используют константу:

$$b = \begin{cases} 1 & \text{если } L(x) > 0.5 \\ 0 & \text{иначе} \end{cases}.$$

В [23, 24] для V-образной функции предложено правило:

$$b(t+1) = \begin{cases} not(b(t)) & \text{если } rand < TF2(x(t+1)) \\ b(t) & \text{если } rand \geq TF2(x(t+1)) \end{cases}.$$

S-образные и V-образные функции трансформации были успешно применены для бинаризации метаэвристических алгоритмов серого волка [25], летучей мыши [26], стрекозы [27, 28], бабочки [29], дрозофилы [30], роящихся частиц [31, 32], гравитационного поиска [23, 24, 33].

Метод угловой модуляции на первом этапе использует тригонометрическую функцию с четырьмя параметрами [22, 34]:

$$g(x) = \sin(2\pi(x - a_1)a_2 \cos(2\pi(x - a_1)a_3)) + a_4.$$

Правило бинаризации имеет вид:

$$b = \begin{cases} 1 & \text{если } g(x) > 0 \\ 0 & \text{иначе} \end{cases}.$$

Метод угловой модуляции применялся для бинаризации алгоритма искусственной пчелиной колонии [35], роящихся частиц [36], опыления цветов [34], дифференциальной эволюции [37].

В [22] отмечается, что эффективность трансформационных и тригонометрических функций все еще неясна и не изучена, предстоит понять, зависит ли эффективность этих функций от используемого алгоритма и решаемой проблемы.

2.2. Переопределение исходных операторов метаэвристики

Метод модифицированных алгебраических операций. В этом методе арифметические операции заменяют на их логические аналоги. Вместо сложения используют логическое “или” (OR), вместо умножения — логическое “и” (AND), а вместо вычитания — исключающее “или” (XOR). Таким образом, пространство поиска изменяется на би-

нарное, однако исходная идея алгоритма, формально, сохраняется.

Данный метод имеет несколько недостатков, из-за которых он не может быть применен к некоторым метаэвристическим алгоритмам. Первый недостаток заключается в невозможности применения этого метода к алгоритмам, которые используют какие-либо операции кроме сложения, вычитания, умножения и деления. Вторым недостатком связан с тем, что многие метаэвристические алгоритмы используют вещественные коэффициенты. При переходе к бинарным векторам, эти коэффициенты должны быть тоже некоторым образом преобразованы или исключены из алгоритма. Так, в [38] при модификации алгоритма PSO, авторы отказались от использования коэффициента инерции, а вместо коэффициентов скорости использовали случайные бинарные вектора.

Метод модифицированных алгебраических операций был применен для бинаризации метаэвристических алгоритмов кошек [39], пчел [40], паукообразных обезьян [41, 42].

Квантовый подход. Двоичная природа принципов квантовых вычислений позволяет применять их для решения проблемы бинаризации. Наименьшая информация, которую можно сохранить в Q-бите, представлена следующим образом:

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}; \quad \forall \alpha, \beta \in [0, 1],$$

$$|\alpha|^2 + |\beta|^2 = 1.$$

В приведенном выше уравнении $|\alpha|^2$ и $|\beta|^2$ означают вероятности перехода Q-бита в состояние “0” и “1” соответственно. Большее значение $|\alpha|^2$ определяет вероятность того, что Q-бит окажется в состоянии “0”, $|\beta|^2$ — в состоянии “1”. Таким образом, Q-бит подчиняется принципу линейной суперпозиции:

$$|\phi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

Строка из NQ -битов задается следующим образом:

$$\begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_N \\ \beta_1 & \beta_2 & \dots & \beta_N \end{bmatrix},$$

$$|\alpha_i|^2 + |\beta_i|^2 = 1, \quad i = 1, 2, \dots, N.$$

Q-бит имеет равные шансы принять состояние “0” или “1” при $\alpha = \beta = 1/\sqrt{2}$. В этом случае, принцип линейной суперпозиции приводит к

$$|\phi\rangle = \sum_{k=1}^{2^N} \frac{1}{\sqrt{2^N}} |X_k\rangle,$$

где X_k описывает состояние k двоичной строкой $(x_1, x_2, \dots, x_N)$ [43].

Итоговое бинарное решение формируется следующим образом: для каждой j -й позиции генерируется случайное число $rand$ в диапазоне $[0, 1]$ и сравнивается с β_j ; если $rand < \beta_j$, то $x_j = 1$, в противном случае $x_j = 0$.

Состояние Q-бита может быть изменено квантовым вентилям (Q-вентилем). Q-вентиль является обратимым и может быть представлен как унитарный оператор U , работающий на базисных Q-битных состояниях и удовлетворяющий условию $U^+U = UU^+$, где U^+ эрмитово сопряжение U . Существует несколько Q-вентилей, таких как NOT-вентиль, управляемый NOT-вентиль, вентиль вращения, вентиль Адамара. Система из N Q-битов может содержать информацию о 2^N состояниях. Однако процесс измерения приведет к мгновенному выбору одного из этих состояний. Такое явление определяется волновой функцией ψ . Коэффициенты c_i называются амплитудами вероятности, а $|c_i|^2$ задает вероятность перехода $|\psi\rangle$ в состояние $|\phi\rangle$, если будет проведено измерение, причем $\sum_i |c_i|^2 = 1$ [44].

В [45] проблемы выбора и оптимизации параметров для нейронных сетей были решены с использованием метода оптимизации роя квантовых частиц (PSO), при этом функции были представлены в виде квантовых битов (Q-битов). Принципы квантовой суперпозиции и квантовой вероятности использовались для ускорения поиска оптимального набора признаков.

Исследователи в [46] предложили гибридный алгоритм интеллекта роя, основанный на квантовых вычислениях и комбинации алгоритма светлячков и PSO для выбора признаков. Квантовые вычисления обеспечили хороший компромисс между интенсификацией и диверсификацией поиска, в то время как комбинация алгоритма светлячка и PSO позволила эффективно исследовать сгенерированные подмножества признаков. Чтобы оценить актуальность этих подмножеств, была использована теория приближительных множеств.

Хан и Ким в [47] описали квантовый эволюционный алгоритм, основанный на концепции и принципах квантовых вычислений. Алгоритм представляет решение в виде Q-битовой хромосомы и использует квантовый оператор для его обновления, этот оператор является модифицированной версией оператора логического вращения.

В [48] для решения задач бинарной оптимизации был разработан алгоритм квантового гравитационного поиска путем добавления в базовую структуру процедуры гравитационного поиска следующих элементов квантовых вычислений: квантовый бит, квантовые измерения, суперпозиция и модифицированные операторы квантового вращения.

3. ПОСТАНОВКА ЗАДАЧИ

В основе классификации лежит таблица наблюдений:

$$T = \{t_1, t_2, \dots, t_M\},$$

где $t_k = (x_k, c_k)$ — k -й помеченный экземпляр таблицы наблюдений, $x_k = (x_1, x_2, \dots, x_n)$ — вектор значений входных переменных, c_k — метка класса, M — число наблюдений, n — число входных переменных.

Задача классификации — сопоставить каждому новому непомеченному экземпляру наиболее подходящий класс из множества классов $c = \{c_1, c_2, \dots, c_K\}$, K — количество классов.

Задача отбора признаков формулируется следующим образом: на заданном множестве признаков найти такое подмножество, которое не вызывает значительного снижения точности классификации или даже увеличивает ее при уменьшении количества признаков. Решение представлено в виде вектора $s = (s_1, s_2, \dots, s_n)$, где $s_i = 0$ означает, что i -й признак исключен из классификации, а $s_i = 1$ означает, что i -й признак используется классификатором. Эффективность отобранных признаков оценивается по ошибке классификации. Ошибка классификации — это количество неправильно классифицированных экземпляров по отношению к общему количеству экземпляров.

Задача алгоритма бинарной оптимизации — поиск подмножества признаков, на которых ошибка классификации будет минимальной.

4. КЛАССИФИКАТОРЫ

4.1. Классификатор на основе k -ближайших соседей

Алгоритм на основе k -ближайших соседей является одним из простых и наиболее часто используемых классификаторов. У алгоритма есть только один параметр k , который указывает количество соседей, используемых для определения метки класса для нового экземпляра данных. Алгоритм не требует этапа обучения и настройки параметров.

Пошаговое определение алгоритма приведено ниже.

1. Определить значение k .
2. Сохранить таблицу наблюдений T .
3. Для каждого нового экземпляра данных x :
 - 3.1. Рассчитать расстояния от x до каждого экземпляра T .
 - 3.2. Выбрать k ближайших экземпляров.
 - 3.3. Определить наиболее часто встречаемую метку класса c среди выбранных экземпляров.
 - 3.4. Присвоить метку c экземпляру x .

В нашей статье значение k выбрано равным 5.

4.2. Нечеткий классификатор

На основе таблицы наблюдений на области определения каждой входной переменной формируются нечеткие термы. Исходя из разбиения переменных на нечеткие термы, формируется база правил нечеткого классификатора. Правило нечеткого классификатора имеет вид [33, 42, 49]:

$$R_i : \text{IF } s_1 \wedge x_1 = A_{i1} \text{ AND } s_2 \wedge x_2 = A_{i2} \text{ AND } \dots \text{ AND } s_n \wedge x_n = A_{in} \text{ TO class} = c_j, \\ j = 1, \dots, R,$$

где A_{ki} — нечеткий терм, характеризующий k -й признак в i -м правиле ($k = 1, \dots, n$), R — число правил.

На заданной таблице наблюдений метка класса определяется следующим образом:

$$\text{class} = c_t, \quad t = \arg \max_{1 \leq j \leq m} \beta_j, \\ \beta_t(x_p) = \sum_{R_i: \text{class } t} \prod_{k=1}^n \mu_{A_{ki}}(x_{pk}),$$

где $\mu_{A_{ki}}(x_{pk})$ — значение функции принадлежности нечеткого термина A_{ki} в точке x_k .

Мера точности классификации задается как отношение количества правильно определенных меток классов к общему количеству экземпляров:

$$Acc(\theta, S) = \frac{\sum_{p=1}^Z \begin{cases} 1, & \text{if } c_p = \arg \max_{1 \leq j \leq m} f_j(x_p; \theta, S) \\ 0, & \text{otherwise} \end{cases}}{Z}$$

Ошибка классификации определяется как:

$$Error = 1 - Acc(\theta, S).$$

5. БИНАРИЗАЦИЯ АЛГОРИТМА СТАИ ЛАСТОЧЕК

5.1. Алгоритм стаи ласточек для непрерывного пространства поиска

Алгоритм стаи ласточек основан на поведении стаи ласточек при длительных перелетах и поиске пищи [8]. Основная идея алгоритма состоит в разделении всех частиц на три вида: частицы-лидеры, частицы-исследователи, бесцельные частицы. Роль каждой частицы зависит от ее положения в пространстве.

Частицы-лидеры делятся на глобального лидера и локальных лидеров. Глобальным лидером является частица с самым лучшим положением в исследуемом пространстве. Локальные лидеры — это l следующих за глобальным лидером частиц.

Бесцельные частицы — это k худших частиц популяции. Остальные частицы — исследователи.

Во время выполнения очередной итерации перемещаются только исследователи и бесцельные частицы. После каждой итерации роли частиц перераспределяются в зависимости от их нового положения. Таким образом, бесцельные частицы и исследователи могут становиться лидерами, а лидеры могут становиться исследователями или бесцельными частицами.

Перемещение исследователя определяют положение глобального лидера, положение ближайшего локального лидера и лучшее положение этой частицы за все время оптимизации. Формулы движения исследователя приведены ниже:

$$\theta_{e_{i+1}} = \theta_{e_i} + V_{i+1},$$

$$V_{i+1} = VHL_{i+1} + VLL_{i+1},$$

$$VHL_{i+1} = \text{rand}(0; 1) \times (\theta_{e_{best}} - \theta_{e_i}) + \text{rand}(0; 1) \times (HL - \theta_{e_i}),$$

$$VLL_{i+1} = \text{rand}(0; 1) \times (\theta_{e_{best}} - \theta_{e_i}) + \text{rand}(0; 1) \times (LL_i - \theta_{e_i}),$$

где θ_{e_i} — положение i -й частицы-исследователя, **HL** — положение глобального лидера, **LL_i** — положение локального лидера, наименее удаленного от i -й частицы исследователя, $\theta_{e_{best}}$ — лучшее положение i -й частицы-исследователя, **V** вектор скорости частицы, **VHL** — вектор скорости движения частицы к глобальному лидеру, **VLL** — вектор скорости движения частицы к ближайшему локальному лидеру, $\text{rand}(0; 1)$ — случайное число из диапазона $[0, 1]$.

Бесцельные частицы перемещаются случайным образом в пространстве поиска. Для этого используются формулы:

$$\theta_{o_{i+1}} = \text{rand}(0.5; 2) \times VSS,$$

$$VSS = \frac{\sum_{i=1}^{N-k} \theta_i}{N - k},$$

где θ_{o_i} — положение i -й бесцельной частицы, θ_i — положение i -й частицы популяции, N — количество всех частиц в популяции, k — количество бесцельных частиц.

5.2. Метод модифицированных алгебраических операций

Обозначим логические операции следующим образом: $\vee$ — логическое “или”, $\wedge$ — логическое “и”, $\oplus$ — исключающее “или”. После модификации АСЛ, формулы перемещения частицы исследователя будут выглядеть следующим образом:

$$\mathbf{S}_{e_{i+1}} = \mathbf{S}_{e_i} \vee \mathbf{V}_{i+1},$$

$$\mathbf{V}_{i+1} = \mathbf{VHL}_{i+1} \vee \mathbf{VLL}_{i+1},$$

$$\mathbf{VHL}_{i+1} = \text{randbinvector} \wedge (\mathbf{S}_{e_{best}} \oplus \mathbf{S}_{e_i}) \vee \vee \text{randbinvector} \wedge (\mathbf{HL} \oplus \mathbf{S}_{e_i}),$$

$$\mathbf{VLL}_{i+1} = \text{randbinvector} \wedge (\mathbf{S}_{e_{best}} \oplus \mathbf{S}_{e_i}) \vee \vee \text{randbinvector} \wedge (\mathbf{LL}_i \oplus \mathbf{S}_{e_i}),$$

где $\mathbf{S}_{e_i}$ — положение i -й частицы-исследователя, $\mathbf{HL}$ — положение глобального лидера, $\mathbf{LL}_i$ — положение ближайшего до $\mathbf{S}_{e_i}$ локального лидера, $\mathbf{S}_{e_{best}}$ — лучшее положение $\mathbf{S}_{e_i}$, $\mathbf{V}_i$ — вектор скорости $\mathbf{S}_{e_i}$, $\mathbf{VHL}$ — вектор скорости частицы относительно $\mathbf{HL}$, $\mathbf{VLL}$ — вектор скорости частицы относительно $\mathbf{LL}$, randbinvector — случайный бинарный вектор.

Изменение положения бесцельной частицы будет выглядеть следующим образом:

$$\mathbf{S}_{o_{i+1}} = \text{randbinvector}$$

5.3. Метод бинаризации, основанный на операции MERGE

Для решения проблемы отбора признаков нами введена авторская операция MERGE, объединяющая два бинарных вектора, путем сохранения совпадающих элементов и случайного выбора из несовпадающих элементов [10]. Определение значения C_i результирующего вектора $\mathbf{C}$ из соответствующих элементов векторов $\mathbf{A}$ и $\mathbf{B}$ может быть представлено следующим образом:

$$\text{MERGE}(A_i, B_i, C_i) = \begin{cases} C_i = A_i = B_i, & \text{если } A_i = B_i \\ C_i = A_i, & \text{если } A_i \neq B_i \text{ и } \text{rand}(0;1) \leq 0.5, \\ C_i = B_i, & \text{если } A_i \neq B_i \text{ и } \text{rand}(0;1) > 0.5 \end{cases}$$

где A_i — i -й элемент вектора $\mathbf{A}$, B_i — i -й элемент вектора $\mathbf{B}$, C_i — i -й элемент вектора $\mathbf{C}$, $\text{rand}(0; 1)$ — случайное число из диапазона $[0, 1]$.

Метод бинаризации, основанный на операции MERGE, очень похож на метод модифицированных алгебраических операций, отличие состоит только в том, что все оригинальные арифметиче-

ские операции необходимо заменить на операцию MERGE. Также, этот метод накладывает те же ограничения, что и метод модифицированных алгебраических операций.

После модификации АСЛ, формулы перемещения частицы исследователя будут выглядеть следующим образом:

$$\mathbf{S}_{e_{i+1}} = \text{merge}(\mathbf{S}_{e_i}, \mathbf{V}_{i+1}),$$

$$\mathbf{V}_{i+1} = \text{merge}(\mathbf{VHL}_{i+1}, \mathbf{VLL}_{i+1}),$$

$$\mathbf{VHL}_{i+1} = \text{merge}(\text{merge}(\text{randbinvector}, \text{merge}(\mathbf{S}_{e_{best}}, \mathbf{S}_{e_i})), \text{merge}(\text{randbinvector}, \text{merge}(\mathbf{HL}, \mathbf{S}_{e_i}))),$$

$$\mathbf{VLL}_{i+1} = \text{merge}(\text{merge}(\text{randbinvector}, \text{merge}(\mathbf{S}_{e_{best}}, \mathbf{S}_{e_i})), \text{merge}(\text{randbinvector}, \text{merge}(\mathbf{LL}_i, \mathbf{S}_{e_i}))).$$

Изменение положения бесцельной частицы будет выглядеть следующим образом:

$$\mathbf{S}_{o_{i+1}} = \text{randbinvector}$$

5.4. Гибридный метод бинаризации, основанный на методе модифицированных алгебраических операций и операции MERGE

Это авторский метод, он похож на метод модифицированных алгебраических операций, отличие состоит только в том, что операцию сложения нужно заменить на операцию MERGE. Также, этот метод накладывает те же ограничения, что и метод модифицированных алгебраических операций.

После модификации АСЛ, формулы перемещения частицы исследователя будут выглядеть следующим образом:

$$\mathbf{S}_{e_{i+1}} = \text{merge}(\mathbf{S}_{e_i}, \mathbf{V}_{i+1}),$$

$$\mathbf{V}_{i+1} = \text{merge}(\mathbf{VHL}_{i+1}, \mathbf{VLL}_{i+1}),$$

$$\mathbf{VHL}_{i+1} = \text{merge}(\text{randbinvector} \wedge (\mathbf{S}_{e_{best}} \oplus \mathbf{S}_{e_i}), \text{randbinvector} \wedge (\mathbf{HL} \oplus \mathbf{S}_{e_i})),$$

$$\mathbf{VLL}_{i+1} = \text{merge}(\text{randbinvector} \wedge (\mathbf{S}_{e_{best}} \oplus \mathbf{S}_{e_i}), \text{randbinvector} \wedge (\mathbf{LL}_i \oplus \mathbf{S}_{e_i})).$$

Изменение положения бесцельной частицы будет выглядеть следующим образом:

$$\mathbf{S}_{o_{i+1}} = \text{randbinvector}$$

Таблица 1. Описание наборов данных

Набор данных	Количество признаков	Количество классов	Количество экземпляров
australian	14	2	890
balance	4	3	625
banana	2	2	5300
bands	19	2	365
bupa	6	2	345
cleveland	13	5	297
coil2000	85	2	9822
contraceptive	9	3	1473
dermatology	34	6	358
ecoli	7	8	336
glass	9	7	214
haberman	3	2	306
heart	13	2	270
hepatitis	19	2	80
ionosphere	33	2	351
iris	4	3	150
magic	10	2	19020
newthyroid	5	3	215
optdigits	64	10	5620
page-blocks	10	5	5472
penbased	16	10	10992
phoneme	5	2	5404
pima	8	2	768
ring	20	2	7400
satimage	36	7	6435
segment	19	7	2310
sonar	60	2	208
spambase	57	2	4597
spectfheart	44	2	267
tae	5	3	151
texture	40	11	5500
thyroid	21	3	7200
titanic	3	2	2201
twonorm	20	2	7400
vehicle	18	4	846
wdbc	30	2	569
wine	13	3	178
wisconsin	9	2	683

Таблица 2. Обозначение методов бинаризации

Метод	Обозначение
Без отбора признаков	NoSelect
Метод, основанный на операции MERGE	Operations
Метод модифицированных алгебраических операций	Merge
Гибридный метод, основанный на методе модифицированных алгебраических операций и операции MERGE	OpMerge
Метод трансформационных функций с функцией трансформации F_S	Sigm
Метод трансформационных функций с функцией трансформации F_{V_1}	Tanh
Метод трансформационных функций с функцией трансформации F_{V_2}	VShaped

Таблица 3. Ранги методов бинаризации для ошибки классификации

	NoSelect	Operations	Merge	OpMerge	Sigm	Tanh	VShaped
Нечеткий классификатор	6.08	4.26	4.04	3.34	3.57	3.54	3.16
5NN	6.82	5.03	4.24	3.32	2.78	3.07	2.76

Таблица 4. Ранги методов бинаризации для количества признаков

	Operations	Merge	OpMerge	Sigm	Tanh	VShaped
Нечеткий классификатор	3.82	4.71	2.49	3.28	3.41	3.30
5NN	3.87	4.74	2.35	2.87	3.75	3.42

5.5. Метод трансформационных функций

Для бинаризации использованы три функции трансформации. Одна S-образная, которая описывается следующим образом:

$$F_S(v_{ij}) = \frac{1}{(1 + e^{-v_{ij}})}$$

а также две V-образные:

$$F_{V_1}(v_{ij}) = |\tanh(v_{ij})|,$$

$$F_{V_2}(v_{ij}) = \left| \frac{v_{ij}}{\sqrt{1 + v_{ij}^2}} \right|.$$

Для трансформации вещественного значения в бинарное будет использоваться следующее правило:

$$\text{IF rand}(0; 1) < F(v_{ij}), \\ \text{THEN } s_{ij} = 1, \quad \text{ELSE } s_{ij} = 0.$$

6. ОПИСАНИЕ ЭКСПЕРИМЕНТА

Вычислительный эксперимент проведен на наборах данных из репозитория KEEL (<https://>

sci2s.ugr.es/keel/datasets.php). Информация о наборах данных представлена в табл. 1.

Все наборы данных были разделены с помощью десятикратной кросс-валидации. В качестве классификаторов использовались нечеткий классификатор и классификатор kNN с 5 соседями (5NN). В качестве алгоритма отбора признаков использовался АСЛ, бинаризованный шестью различными способами, представленными в табл. 2.

Параметры алгоритма всегда были следующие: количество всех частиц популяции — 40, количество локальных лидеров — 3, количество бесцельных частиц — 6, количество итераций — 300. Для каждого набора данных эксперимент повторялся 30 раз, после чего результаты усреднялись.

Таблица 5. p-значения

	Ошибка классификации	Количество признаков
Нечеткий классификатор	4.66e-11	1.4e-06
5NN	1.49e-27	4.82e-8

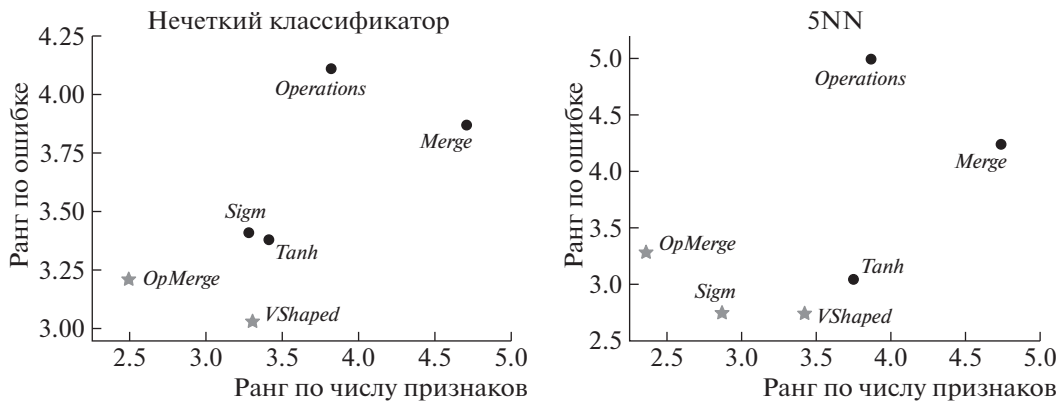


Рис. 2. Графическое представление результатов бинаризации.

При проведении эксперимента оценивалось два параметра — точность классификации на тестовых данных и количество признаков, оставленных алгоритмом. Программный код эксперимента и инструкции по воспроизведению тестов опубликованы на платформе GitLab [50].

Результаты эксперимента для нечеткого классификатора представлены в табл. П1 и П2, для классификатора 5NN — в табл. П3 и П4.

Для статистической оценки предложенных методов бинаризации использовался двухфакторный дисперсионный анализ Фридмана для связанных выборок (уровень значимости равен 0.05). В табл. 3 и 4 приведены ранги каждого из методов бинаризации для каждого классификатора, а в табл. 5 соответствующие им p -значения.

Все p -значения меньше уровня значимости 0.05, значит результаты являются статистически значимыми и отвергаются нулевые гипотезы о равенстве ошибок классификации и числа отобранных признаков для исследуемых методов бинаризации. На основе полученных результатов были построены графики в координатах ошибка-число_признаков и определен Парето-фронт.

На рис. 2 звездочками отмечены элементы, входящие в Парето-множество. Используя один из принципов оптимальности, а именно, принцип идеальной точки можно определить лучшее решение из Парето-множества. Координаты идеальной точки соответствуют минимальным рангам и равны (1, 1). Лучшей является альтернатива, максимально приближенная к идеальной точке. В качестве меры близости можно использовать евклидово расстояние. Для нечеткого классификатора и метода бинаризации OpMerge расстояние до идеальной точки равно:

$$\sqrt{(3.21 - 1)^2 + (2.49 - 1)^2} = 2.67,$$

а для нечеткого классификатора и метода бинаризации VShaped расстояние равно:

$$\sqrt{(3.03 - 1)^2 + (3.30 - 1)^2} = 3.07.$$

В этом случае лучшей является альтернатива OpMerge.

Для классификатора 5NN и метода бинаризации OpMerge расстояние до идеальной точки равно 2.64, для метода бинаризации VShaped — 2.98, для метода бинаризации Sigm — 2.56. В случае классификатора 5NN альтернатива OpMerge лишь немного уступает альтернативе Sigm.

Если рассматривать ошибку классификации и число отобранных признаков как отдельные критерии, то гибридный метод OpMerge позволил найти минимальное подмножество признаков для обоих классификаторов, однако лучшие результаты по точности классификации показал метод VShaped.

7. ЗАКЛЮЧЕНИЕ

В данной работе было рассмотрено шесть подходов к бинаризации алгоритма стаи ласточек для решения задачи отбора признаков при классификации 38 наборов данных. Лучшие способности к отбору признаков показал гибридный метод, основанный на методе модифицированных алгебраических операций и введенной нами операции MERGE. Данный метод бинаризации рекомендуется к использованию, если приоритетной задачей является максимальное упрощение получаемой модели. Наибольшую точность классификации показал метод с использованием V-образной функции трансформации. Этот метод рекомендуется к использованию, если приоритетом является точность классификации.

В дальнейших исследованиях предполагается проверить предложенные методы бинаризации на других классификаторах, а также использовать квантовые методы бинаризации и другие трансформационные функции с другими правилами бинаризации.

Таблица П1. Значения точности для нечеткого классификатора

Dataset	NoSelect	Operations	Merge	Opmerge	Sigm	Tanh	Vshaped
australian	49.42	59.41	59.56	59.71	59.56	59.56	59.71
balance	46.08	46.08	46.08	46.08	46.08	46.08	46.08
banana	46.66	58.91	58.91	58.91	58.91	58.91	58.91
bands	52.96	64.4	64.68	63.34	64.41	62.5	64.68
bupa	48.96	57.66	57.66	57.66	57.66	57.66	57.66
cleveland	42.31	53.13	54.07	53.13	54.07	54.07	54.07
coil2000	94.03	94.03	94.03	94.03	94.01	94.03	94.03
contraceptive	39.04	42.7	42.7	42.7	42.7	42.7	42.7
dermatology	74.81	82.88	83.42	81.57	86.23	86.81	84.32
ecoli	45.59	46.2	46.84	48.04	46.84	46.84	46.84
glass	52.29	58.24	57.35	56.76	56.16	56.16	56.16
haberman	54.27	54.96	54.96	54.96	54.96	54.96	54.96
heart	67.41	67.04	66.67	67.04	66.67	67.04	67.04
hepatitis	26.69	85.02	82.48	81.05	83.59	82.34	82.34
ionosphere	79.83	84.37	82.95	82.98	82.95	82.69	81.51
iris	94	96.67	96.67	96	96.67	96	96
magic	57.93	71.72	72.08	72.98	72.57	72.97	72.97
newthyroid	95.84	94.89	94.89	94.89	95.84	95.84	95.84
optdigits	10.07	26.58	26.81	27.1	29.65	30.58	30.92
page-blocks	48.77	90	90.13	90.28	90.28	90.26	90.28
penbased	56.85	56.76	57.8	56.98	57.79	57.83	57.39
phoneme	74.76	77.33	77.33	77.33	77.33	77.33	77.33
pima	68.76	70.18	70.05	70.57	70.57	71.22	71.22
ring	49.51	57.2	56.84	65.66	62.28	65.66	65.66
satimage	60.33	63.37	63.97	68.43	65.38	65.15	65.6
segment	79.74	84.16	84.5	84.94	85.11	84.72	84.68
sonar	56.08	57.01	54.22	68.22	55.01	55.2	54.13
spambase	39.44	60.26	63.45	78.6	70.17	73.26	76.79
spectfheart	80.56	79.78	78.75	77.88	76.8	78.33	81
tae	33.63	38.62	38.62	36.31	35.64	36.31	38.62
texture	69.82	71.6	71.56	73.31	72.36	72.33	72.55
thyroid	7.28	93.71	96.07	96.75	96.81	96.76	96.76
titanic	65.21	65.21	65.21	65.21	65.21	65.21	65.21
twonorm	96.05	96.05	95.95	96.05	96.03	95.96	96.04
vehicle	29.2	39.12	41.5	44.7	44.42	43.86	44.23
wdbc	92.97	94.55	95.26	95.08	95.43	95.43	96.48
wine	89.3	94.96	92.61	91.57	92.12	91.57	91.04
wisconsin	87.71	87.72	88.15	88.01	87.86	88.01	88.01

Таблица П2. Количество отобранных признаков для нечеткого классификатора

Dataset	Operations	Merge	OpMerge	Sigm	Tanh	Vshaped
australian	6.3	5.9	4.3	5.8	5.6	5.2
balance	4	4	4	4	4	4
banana	1	1	1	1	1	1
bands	9.1	9.7	4.9	7.1	9	8
bupa	2.6	2.6	2.6	2.6	2.6	2.6
cleveland	3.7	4	3.4	3.8	2.8	3.1
coil2000	85	85	81.1	51	68	73.9
contraceptive	3.5	3.7	4.1	3.7	3.2	3.4
dermatology	18.9	22.3	14.6	18.6	23.3	22.2
ecoli	5.3	5.3	5.3	5.1	5.3	5.1
glass	6.7	6.3	6.3	6.5	6.5	6.5
haberman	1.7	1.8	1.6	1.8	1.9	1.7
heart	6.9	7	7.3	7.2	6.8	6.9
hepatitis	8.7	8.8	5.9	9	7.5	8
ionosphere	18.4	19.5	15.8	18.1	19.9	18.4
iris	2.2	2.1	2.3	2.2	2.3	2.3
magic	4.9	4.6	1.4	2.3	1.5	1.5
newthyroid	2.8	3	3.2	3.3	2.9	3
optdigits	30.5	37.5	23.5	31.9	36.4	34.4
page-blocks	3.1	4	4	4.4	4.2	4.3
penbased	14.8	15	15.2	14.8	14.6	14.7
phoneme	4	4	4	4	4	4
pima	3.9	4.1	3.6	4	4.1	4.1
ring	4.9	4.9	2.8	3.6	2.8	2.8
satimage	16	13.2	4.9	11.1	12.5	10.5
segment	9.8	11.2	7.6	8.6	10.3	9.4
sonar	28.9	43.3	4.1	29.5	31.4	30.1
spambase	19.8	20.1	4.1	19.9	19.2	15.8
spectfheart	24.9	32	25	22.9	29.7	27
tae	2.6	2.8	2.4	2.4	2.5	2.9
texture	19.2	21.7	6.8	16.9	21.3	17.6
thyroid	8.5	9.2	5.3	8.4	8.3	7.2
titanic	3	3	3	3	3	3
twonorm	20	19.1	20	19.8	19	19.8
vehicle	8.5	10.1	4.8	6.5	6	6.5
wdbc	13.4	13.4	4.9	11.1	12.9	11.7
wine	6.5	7.7	5	6.2	6.4	6.6
wisconsin	5.1	5.7	5.1	4.9	5	5.2

Таблица ПЗ. Значения точности для классификатора 5NN

Dataset	NoSelect	Operations	Merge	OpMerge	Sigm	Tanh	Vshaped
australian	83.932	91.45	91.61	92.18	92.91	92.47	92.48
balance	83.202	83.20	83.20	83.20	83.20	83.20	83.20
banana	89.397	89.40	89.40	89.40	89.40	89.40	89.40
bands	67.65	84.41	87.15	87.41	87.67	88.24	89.32
bupa	61.429	79.11	79.70	79.70	79.70	79.70	79.70
cleveland	57.002	71.04	72.38	73.10	75.39	74.41	74.08
coil2000	93.667	94.23	94.19	94.35	94.35	94.28	94.29
contraceptive	46.981	57.84	57.91	58.46	58.52	58.52	58.52
dermatology	97.235	100.00	100.00	99.72	100.00	100.00	100.00
ecoli	82.206	86.36	86.36	86.36	86.36	86.36	86.36
glass	67.741	83.74	84.21	84.71	84.71	84.71	84.71
haberman	68.335	79.39	79.39	79.39	79.39	79.39	79.39
heart	81.111	92.59	94.07	95.19	94.82	94.07	94.82
hepatitis	85.794	98.89	98.89	100.00	98.89	98.89	98.89
ionosphere	84.912	94.30	93.73	98.87	95.99	96.01	96.58
iris	96	100.00	100.00	100.00	100.00	100.00	100.00
magic	83.722	84.82	85.06	85.03	85.23	85.07	85.23
newthyroid	94.004	98.18	98.18	98.18	98.18	98.18	98.18
optdigits	98.827	98.88	99.27	98.86	99.17	99.27	99.13
page-blocks	95.834	97.04	97.00	97.20	97.19	97.11	97.20
penbased	99.3	99.36	99.52	99.34	99.56	99.51	99.48
phoneme	88.693	88.99	88.99	88.99	88.99	88.99	88.99
pima	74.228	81.78	81.91	81.91	82.04	81.91	82.04
ring	69.012	87.15	87.28	87.81	88.04	88.14	88.09
satimage	91.098	92.23	92.62	92.40	92.79	92.81	92.75
segment	95.626	97.84	97.97	98.27	98.31	98.23	98.18
sonar	82.18	97.12	96.64	100.00	100.00	99.52	100.00
spambase	90.326	91.96	92.85	92.83	93.24	93.18	93.42
spectfheart	76.478	92.53	93.33	97.04	95.17	96.28	96.68
tae	49.735	63.50	63.50	63.50	63.50	63.50	63.50
texture	98.6	99.16	99.26	99.33	99.44	99.31	99.35
thyroid	94.167	98.50	98.49	98.99	99.00	98.96	99.00
titanic	75.476	78.39	78.39	78.39	78.39	78.39	78.39
twonorm	97	97.00	97.43	97.00	97.12	97.34	97.18
vehicle	71.614	79.31	79.89	81.66	81.19	81.55	81.31
wdbc	96.645	99.12	99.12	99.47	99.12	99.30	99.30
wine	95.512	100.00	100.00	100.00	100.00	100.00	100.00
wisconsin	97.66	98.54	98.98	99.12	99.12	99.12	99.12

Таблица П4. Количество отобранных признаков для классификатора 5NN

Dataset	Operations	Merge	OpMerge	Sigm	Tanh	Vshaped
australian	7	8.5	5.5	7.1	7.5	7.1
balance	4	4	4	4	4	4
banana	2	2	2	2	2	2
bands	8.5	13.1	7	10.6	11.9	10.2
bupa	3.1	3	2.9	3	3.1	2.9
cleveland	6.9	7.8	4.7	6.7	6	6
coil2000	43.8	55.2	21.2	46.2	46.6	49.3
contraceptive	5.2	5.1	5.1	5.1	5.4	5.4
dermatology	24.4	26.1	20.8	23.6	25.1	23.5
ecoli	5.2	5.4	5.2	5	5.1	5.2
glass	6	5.4	4.7	4.7	5	4.8
haberman	1.8	1.8	1.8	1.7	1.7	1.7
heart	6.2	7.1	5.4	7.2	7	7.2
hepatitis	11.9	13.5	11.2	11.7	12	11.1
ionosphere	14.6	15.3	4.8	12.9	11.4	13.4
iris	3.2	3.1	3.1	3.2	3.1	3.2
magic	6.7	6.6	5.5	6.3	6.6	6.3
newthyroid	3	2.8	2.9	2.9	2.8	3.2
optdigits	55.8	54.9	61.2	41.1	49.8	49.3
page-blocks	4.4	5	4.3	4.5	4.5	3.9
penbased	14.3	14.1	14.5	13.3	14	14
phoneme	4.6	4.6	4.6	4.6	4.6	4.6
pima	4.7	5.3	4.4	4.9	4.7	5.1
ring	8.4	8.4	7.8	8.5	8.3	8.3
satimage	21.7	28.5	23.2	20.4	21.6	22.4
segment	10.1	11.3	7.8	9.6	10.5	10
sonar	30.1	32.3	16.1	29	31.1	25.7
spambase	29.7	43.9	22.2	30.1	33.4	30.3
spectfheart	20.3	27.7	10.5	21.9	19	22.9
tae	2.4	2.7	2.6	2.4	2.8	2.8
texture	21.7	25.4	16.1	20.9	23	23.4
thyroid	8.3	10	6.4	7.7	9.5	7.8
titanic	1.9	1.8	2	1.9	1.9	1.8
twonorm	20	19.2	20	18.5	18.8	18.5
vehicle	8.6	10.4	7.6	8.9	10.4	10.1
wdbc	17.8	19.9	16.2	17.5	17.6	17.7
wine	9.3	9.4	8.9	9.2	9.5	9.4
wisconsin	7.6	5.9	6.1	5.7	6	6

8. БЛАГОДАРНОСТИ

Работа поддержана Министерством науки и высшего образования Российской Федерации (проект № FEWM-2020-0042).

СПИСОК ЛИТЕРАТУРЫ

1. *Bolon-Canedo V., Sanchez-Marono N., Alonso-Betanzos A.* Feature Selection for High-Dimensional Data. Heidelberg: Springer, 2015. 163 p.
2. *Qi Z., Wang H., He T., Li J., Gao H.* FRIEND: Feature selection on inconsistent data // *Neurocomputing*. 2020. V. 391. P. 52–64.
3. *Sha Z.-C., Liu Z.-M., Ma C., Chen J.* Feature selection for multi-label classification by maximizing full-dimensional conditional mutual information // *Applied Intelligence*. 2021. V. 51. P. 326–340.
4. *Luke S.* Essentials of Metaheuristics [Электронный ресурс]. Режим доступа: <https://cs.gmu.edu/~sean/book/metaheuristics/>
5. *Boussaid I., Lepagnot J., Siarry P.* A Survey on Optimization Metaheuristics // *Inf. Sci.* 2013. V. 237. P. 82–117.
6. *Garcia J., Crawford B., Soto R., Astorga G.* A clustering algorithm applied to the binarization of Swarm intelligence continuous metaheuristics // *Swarm and Evolutionary Computation*. 2019. V. 44. P. 646–664.
7. *Poli R., Kennedy J., Blackwell T.* Particle swarm optimization // *Swarm Intelligence*. 2007. V. 1. P. 33–57.
8. *Neshat M., Sepidnam G., Sargolzaei M.* Swallow swarm optimization algorithm: a new method to optimization // *Neural Computing and Application*. 2013. V. 23. P. 429–454.
9. *Bouzidi S., Riffi M.E., Bouzidi M., Moucouf M.* The Discrete Swallow Swarm Optimization for Flow-Shop Scheduling Problem // *Advances in Intelligent Systems and Computing*. 2019. V. 915. P. 228–236.
10. *Bouzidi S., Riffi M.E.* Discrete Swallow Swarm Optimization algorithm for Travelling Salesman Problem // *Proceedings of the 2017 International Conference on Smart Digital Environment*. 2017. P. 80–84.
11. *Hodashinsky I., Sarin K., Shelupanov A., Slezkin A.* Feature selection based on swallow swarm optimization for fuzzy classification // *Symmetry*. 2019. V. 11. № 11. P. 1423.
12. *Cover T., Hart P.* Nearest neighbor pattern classification // *IEEE Transactions on Information Theory*. 1967. V. 13. P. 21–27.
13. *Roh S.-Be., Pedrycz W., Ahn T.-C.* A design of granular fuzzy classifier // *Expert Systems with Applications*. 2014. V. 41. P. 6786–6795.
14. *Mekh M.A., Hodashinsky I.A.* Comparative analysis of differential evolution methods to optimize parameters of fuzzy classifiers // *Journal of Computer and Systems Sciences International*. 2017. V. 56. P. 616–626.
15. *Evsukoff A.G., Galichet S., de Lima B.S.L.P., Ebecken N.F.F.* Design of interpretable fuzzy rule-based classifiers using spectral analysis with structure and parameters optimization // *Fuzzy Sets and Systems*. 2009. V. 160. P. 857–881.
16. *Hodashinsky I.A., Gorbunov I.V.* Algorithms of the tradeoff between accuracy and complexity in the design of fuzzy approximators // *Optoelectronics, Instrumentation and Data Processing*. 2013. V. 49. P. 569–577.
17. *Jiang D., Peng C., Fan Z., Chen Y.* Modified binary differential evolution for solving wind farm layout optimization problems // *Proceedings of the IEEE Symposium on Computational Intelligence for Engineering Solutions (CIES)*. 2013. P. 23–28.
18. *Costa M.F.P., Rocha A.M.A.C., Francisco R.B., Fernandes E.M.G.P.* Heuristic-based firefly algorithm for bound constrained nonlinear binary optimization // *Advances in Operations Research*. 2014. Article ID 215182.
19. *Crawford B. et al.* Putting continuous metaheuristics to work in binary search spaces // *Complexity*. 2017. Article ID 8404231.
20. *Wang L., Wang X., Zhen F.J., Zhen L.* A novel probability binary particle swarm optimization algorithm and its application // *Journal of Software*. 2008. V. 3. № 9. P. 28–35.
21. *Kennedy J., Eberhart R.* A discrete binary version of the particle swarm algorithm // *Proceedings of the IEEE International Conference on Computational Cybernetics and Simulation*. 1997. P. 4104–4108.
22. *Dahi Z.A.E.M., Mezjoud C., Draa A.* Binary Bat Algorithm: On the Efficiency of Mapping Functions When Handling Binary Problems Using Continuous-variable-based Metaheuristics // *IFIP Advances in Information and Communication Technology*. V. 456. Cham: Springer, 2015. P. 3–14.
23. *Rashedi E., Nezamabadi-pour H., Saryazdi S.* BGSa: binary gravitational search algorithm // *Natural Computing*. 2009. V. 9. № 3. P. 727–745.
24. *Mirhosseini M., Nezamabadi-pour H.* BICA: a binary imperialist competitive algorithm and its application in CBIR systems // *International Journal of Machine Learning and Cybernetics*. 2018. V. 9. P. 2043–2057.
25. *Emary E., Zawbaa H.M., Hassanien A.E.* Binary grey wolf optimization approaches for feature selection // *Neurocomputing*. 2016. V. 172. P. 371–381.
26. *Qasim O.S., Algarni Z.Y.* Feature selection using different transfer functions for binary bat algorithm // *International Journal of Mathematical, Engineering and Management Sciences*. 2020. V. 5. № 4. P. 697–706.
27. *Sayed G.I., Tharwat A., Hassanien A.E.* Chaotic dragonfly algorithm: an improved metaheuristic algorithm for feature selection // *Applied Intelligence*. 2019. V. 49. P. 188–205.
28. *Too J., Mirjalili S.* A Hyper Learning Binary Dragonfly Algorithm for Feature Selection: A COVID-19 Case Study // *Knowledge-Based Systems*. 2021. V. 212. 106553.
29. *Arora S., Anand P.* Binary Butterfly Optimization Approaches for Feature Selection // *Expert Systems with Application*. 2019. V. 116. P. 147–160.
30. *Zhang X. et al.* Gaussian mutational chaotic fruit fly-built optimization and feature selection // *Expert Systems with Application*. 2020. V. 141. P. 112976.
31. *Ji B. et al.* Bio-inspired feature selection: An improved binary particle swarm optimization approach // *IEEE Access*. 2020. V. 8. P. 85989–86002.

32. *Mirjalili S., Lewis A.* S-shaped versus V-shaped transfer functions for binary Particle Swarm Optimization // *Swarm and Evolutionary Computation*. 2013. V. 9. P. 1–14.
33. *Bardamova M., Konev A., Hodashinsky I., Shelupanov A.* A fuzzy classifier with feature selection based on the gravitational search algorithm // *Symmetry*. 2018. V. 10. № 11. P. 609.
34. *Dahi Z.A.E.M., Mezjoud C., Draa A.* On the efficiency of the binary flower pollination algorithm: Application on the antenna positioning problem // *Applied Soft Computing*. 2016. V. 47. P. 395–414.
35. *Yavuz G., Aydin D.* Angle Modulated Artificial Bee Colony Algorithms for Feature Selection // *Applied Computational Intelligence and Soft Computing*. 2016. Article ID 9569161.
36. *Pampara G., Franken N., Engelbrecht A.* Combining particle swarm optimisation with angle modulation to solve binary problems // *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1. IEEE Press, Piscataway, NJ, 2005, P. 89–96.
37. *Pampara G., Engelbrecht A.P., Franken N.* Binary differential evolution // *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'06)*. IEEE, 2006. P. 1873–1879.
38. *Yuan X., Nie H., Su A., Wang L., Yuan Y.* An Improved Binary Particle Swarm Optimization for Unit Commitment Problem // *Expert Systems with Application*. 2009. V. 36. P. 8049–8055.
39. *Siqueira H., Figueiredo E., Macedo M., Santana C.J., Bastos-868 Filho C.J., Gokhale A.A.* Boolean binary cat swarm optimization algorithm // *2018 IEEE Latin American Conference on 870 Computational Intelligence (LA-CCI)*. IEEE, 2018. P. 1–6.
40. *Kiran M.S., Gunduz M.* XOR-Based Artificial Bee Colony Algorithm for Binary Optimization // *Turkish Journal of Electrical Engineering and Computer Sciences*. 2013. V. 21. P. 2307–2328.
41. *Singh U., Salgotra R., Rattan M.* A Novel Binary Spider Monkey Optimization Algorithm for Thinning of Concentric Circular Antenna Arrays // *IETE Journal of Research*. 2016. V. 62. P. 736–744.
42. *Hodashinsky I.A., Nemirovich-Danchenko M.M., Samsonov S.S.* Feature selection for fuzzy classifier using the spider monkey algorithm // *Business Informatics*. 2019. V. 13. № 2. P. 29–42.
43. *Srikanth K. et al.* Meta-heuristic framework: Quantum inspired binary grey wolf optimizer for unit commitment problem // *Computers and Electrical Engineering*. 2018. V. 70. P. 243–260.
44. *Manju A., Nigam M.J.* Applications of quantum inspired computational intelligence: a survey // *Artificial Intelligence Review*. 2014. V. 42. P. 79–156.
45. *Hamed H.N.A., Kasabov N.K., Shamsuddin S.M.* Quantum inspired particle swarm optimization for feature selection and parameter optimization in evolving spiking neural networks for classification tasks // *Evolutionary algorithms*. IntechOpen, 2011. P. 133–148.
46. *Zouache D., Abdelaziz F.B.* A cooperative swarm intelligence algorithm based on quantum-inspired and rough sets for feature selection // *Computers and Industrial Engineering*. 2018. V. 115. P. 26–36.
47. *Han K.-H., Kim J.-H.* Quantum-Inspired Evolutionary Algorithms with a New Termination Criterion, He Gate, and Two-Phase Scheme // *IEEE Transactions on Evolutionary Computation*. 2004. V. 8. P. 164–171.
48. *Nezamabadi-pour H.* A quantum-inspired gravitational search algorithm for binary encoded optimization problems // *Engineering Applications of Artificial Intelligence*. 2015. V. 40. P. 62–75.
49. *Ходашинский И.А., Мех М.А.* Построение нечеткого классификатора на основе методов гармонического поиска // *Программирование*. 2017. № 1. С. 54–65.
50. Программный код и инструкции [Электронный ресурс]. Режим доступа: https://gitlab.com/core_developers/fuzzy_core/-/tree/master/experiments/binarization_methods