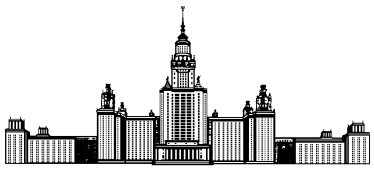


Московский государственный университет имени М. В. Ломоносова
Факультет Вычислительной математики и кибернетики
Кафедра Математических методов прогнозирования

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Обработка множеств логических закономерностей с помощью дисперсионного критерия

Выполнил:
студент 417 группы
Лисяной Александр Евгеньевич

Научный руководитель:
д.ф-м.н., профессор
Рязанов Владимир Васильевич

Москва, 2015

Содержание

1 Введение

Одной из основных задач машинного обучения является задача классификации. Существует множество разнообразных алгоритмов классификации: логистическая регрессия [2], метод опорных векторов [10], нейронные сети [8]. В этой работе будут рассмотрены алгоритмы, основанные на поиске логических закономерностей в данных [16, 13, 12]. Результатом работы таких алгоритмов является набор логических закономерностей (правил), используя которые можно принимать решение о том, к какому классу принадлежит тот или иной объект. Обычно такие правила имеют вид конъюнкции по признакам объекта, где каждый терм конъюнкции представляет из себя допустимый диапазон для значений признака, при котором объект может быть *принят* или *покрыт* данным правилом.

Поиск логических закономерностей в данных — довольно трудоемкая задача. Каждый алгоритм использует различные предположения при построении логических закономерностей. Например, основная идея алгоритма «логические закономерности», реализованного в системе «Распознавание» [17], заключается в следующем: для каждого объекта записывается дискретная оптимизационная задача, решение которой однозначно определяет логическую закономерность. В результате решения данных оптимизационных задач находятся необходимые параметры логических закономерностей, определяющие левые и правые границы допустимых диапазонов для каждого признака. В итоге получается правило, которое как бы «натянуто» на некоторые объекты обучающей выборки.

При использовании логических закономерностей возникают разного рода трудности. Во-первых, множество логических закономерностей может быть большим, тогда пользователю сложно интерпретировать это множество. Во-вторых, множество логических закономерностей может содержать похожие или даже вырожденные закономерности, что ухудшает качество классификации с использованием этого множества.

Таким образом, отдельный интерес представляет задача по обработке полученных множеств логических закономерностей. Эта задача заключается в том, чтобы по исходному множеству логических закономерностей построить множество меньшей мощности, что должно упростить пользователю задачу интерпретации полученных правил. Вторая цель при обработке заключается в том, чтобы построенное множество логических закономерностей меньшей мощности имело качество классификации, сравнимое с исходным множеством.

Целью данной работы является реализовать и применить метод обработки множества правил, основанный на кластеризации с помощью дисперсионного критерия, к результату работы алгоритма «логические закономерности», представленного в системе «Распознавание». В этой работе предлагается использовать описание логических закономерностей с помощью вектора левых и правых границ. Также целью работы является сравнить этот метод с бинаризованным описанием логических закономерностей, предложенным в [14].

Обработанные множества логических закономерностей сравниваются как между собой (в ходе обработки используются различные способы описания правил и критерии информативности), так и с исходным множеством логических закономерностей, полученным в результате настройки алгоритма системы «Распознавания». В результате предложенный в данной работе подход с использованием вектора левых и правых границ в целом работает лучше, чем подход с бинаризованным описанием из [14]. На основе относительно большого числа исходных правил вычисляется существенно меньшее число правил. При их использовании по схеме простого голосования получается классификатор, качество которого сравнимо с качеством алгоритма «логические закономерности» из системы «Распознавание».

2 Задача поиска логических закономерностей

2.1 Определения и обозначения

Основные определения и обозначения будем вводить согласно [15]. Пусть имеется пространство объектов X и конечное множество имен классов $Y = \{1, \ldots, M\}$. Пусть также имеется *обучающая выборка* $X^l = (x_i, y_i)_{i=1}^l$, в которой для каждого объекта x_i известен его класс $y_i \in Y$. Для восстановления целевой зависимости $y^*(x_i) = y_i$ построим алгоритм классификации $a\colon X \to Y$, аппроксимирующий y^* на всём пространстве объектов X.

Для решения такой задачи классификации (восстановления зависимости y^* по обучающей выборке X^l) существует множество разнообразных алгоритмов. Некоторые из них основаны на поиске логических закономерностей.

Определение 1 *Функция* $\varphi(x)\colon X \to \{0, 1\}$ *называется* предикатом. *Говорят, что предикат* φ выделяет *или* покрывает *объект* $x \in X$, *если* $\varphi(x) = 1$.

Формализуем понятие *закономерности*. Для этого введем несколько обозначений:

- Пусть P_c — число объектов класса c в выборке X^l.
- Пусть $p_c(\varphi)$ — число объектов x класса c, на которых $\varphi(x) = 1$
- Пусть N_c — число объектов всех остальных классов $Y \setminus \{c\}$
- Пусть $n_c(\varphi)$ — число объектов x не из объектов класса c, на которых $\varphi(x) = 1$
- Обозначим долю ошибочно выделяемых объектов среди всех объектов, покрываемых *закономерностью*, через $E_c(\varphi, X^l) = \frac{n_c(\varphi)}{p_c(\varphi)+n_c(\varphi)}$
- Обозначим долю верно выделяемых объектов через $D_c(\varphi, X^l) = \frac{p_c(\varphi)}{l}$

Определение 2 *Предикат $\varphi(x)$ называется ε,δ-закономерностью для класса $c \in Y$, если $E_c(\varphi, X^l) \leqslant \varepsilon$ и $D_c(\varphi, X^l) \geqslant \delta$ при заданных ε и δ из отрезка $[0, 1]$*

Такие ε,δ-закономерности также называют просто *закономерностями* или *правилами*. Закономерность называется *чистой* или *непротиворечивой*, если она не покрывает объекты чужого класса. В противном случае закономерность называется *частичной*

Другой способ определения понятия *закономерности* основан на понятии *энтропии*. Будем считать появление объекта класса c исходом ω_0, а появление объекта любого другого класса — исходом ω_1. Вспомним, что функция энтропии для дискретной случайной величины с двумя исходами, вероятности которых p и $q = 1 - p$, имеет вид (основание логарифма выбрано равным 2):

$$\mathbb{H}(p, q) = -p \log_2(p) - q \log_2(q)$$

Пусть далее в выборке X^l имеется P_c объектов из класса c и N_c объектов не из класса c (таким образом, $P_c + N_c = l$). Тогда можно оценить энтропию выборки по формуле:

$$\tilde{\mathbb{H}}(P_c, N_c) = \mathbb{H}\left(\frac{P_c}{P_c + N_c}, \frac{N_c}{P_c + N_c}\right) \tag{1}$$

Когда предикат $\varphi(x)$ покрывает $p_c \leqslant P_c$ объектов из класса c и $n_c \leqslant N_c$ объектов не из класса c, то он разбивает исходную выборку на две подвыборки: $\left\{x \in X^l | \varphi(x) = 1\right\}$ и $\left\{x \in X^l | \varphi(x) = 0\right\}$, энтропии которых можно оценить аналогично 1 как $\tilde{\mathbb{H}}(p_c, n_c)$ и $\tilde{\mathbb{H}}(P_c - p_c, N_c - n_c)$ соответственно. Вероятность

того, что объект принадлежит одной из подвыборок, оценивается, как $\frac{p_c+n_c}{P_c+N_c}$ и $\frac{P_c+N_c-p_c-n_c}{P_c+N_c}$ соответственно. Тогда после получения информации от предиката $\varphi(x)$ энтропия всей выборки оценивается так:

$$\tilde{\mathbb{H}}(P_c, N_c, p_c, n_c) = \frac{p_c + n_c}{P_c + N_c}\tilde{\mathbb{H}}(p_c, n_c) + \frac{P_c + N_c - p_c - n_c}{P_c + N_c}\tilde{\mathbb{H}}(P_c - p_c, N_c - n_c)$$

Уменьшение энтропии при этом составляет:

$$\mathrm{IGain}_c(\varphi, X^l) = \tilde{\mathbb{H}}(P_c, N_c) - \tilde{\mathbb{H}}(P_c, N_c, p_c, n_c)$$

Эта величина называется *информационным выигрышем* и отражает количество информации, которое содержится в предикате $\varphi(x)$. Это позволяет ввести альтернативное определение *закономерности*:

Определение 3 *Предикат $\varphi(x)$ называется* закономерностью *по энтропийному критерию информативности, если $IGain_c(\varphi, X^l) > G_0$ при некотором G_0.*

Вообще говоря, предикаты $\varphi(x)\colon X \to Y$ могут иметь довольно сложный вид. Например, если в качестве объектов из X выступают точки на числовой прямой, то для классификации на рациональные и иррациональные числа можно взять аналог функции Дирихле в качестве предиката:

$$\varphi(x) = \begin{cases} 1 & x \in \mathbf{Q} \\ 0 & x \in \mathbf{R} \setminus \mathbf{Q} \end{cases}$$

Для того, чтобы предикаты были более простыми, введем ограничение на их вид так, как было сделано в [16].

Определение 4 *Пусть $c_1, c_2 \in R$, $f_j(x)$ — j-ый признак объекта x, тогда* параметрическим элементарным предикатом *будем называть*

$$P^{c_1,c_2}(f_j(x)) = \begin{cases} 1 & \textit{при } c_1 \leqslant f_j(x) \leqslant c_2 \\ 0 & \textit{иначе} \end{cases}$$

Поскольку в предикате используется лишь один признак из всего признакового описания объекта, то такой предикат называется элементарным. Параметры $c_1, c_2 \in R$ для каждого отдельного признака $f_j(x)$ выбираются, вообще говоря, свои, поэтому предикат называют параметрическим.

Определение 5 *Пусть каждый объект выборки $x \in X^l$ имеет размерность D и пусть $\Omega \subseteq \{1, 2, \ldots, D\}$. Предикат*

$$\varphi(x) = P^{\Omega, \mathbf{c_1}, \mathbf{c_2}}(x) = \bigwedge_{j \in \Omega} P^{c_1^j, c_2^j}(f_j(x))$$

называется логической закономерностью класса c, если выполнено:

1. $\exists x \in c\colon \varphi(x) = 1$
2. $\forall x \notin c\colon \varphi(x) = 0$
3. $\varphi(x) = \arg\max_{\varphi^*(x)} \Phi(\varphi^*(x))$, *где* Φ — *критерий качества предиката.*

Если предикат удовлетворяет только первым двум условиям, то предикат $\varphi(x)$ называется *допустимым*. Если выполнены все условия, кроме второго, то такая логическая закономерность называется *частичной*. Индексы j над параметрами c_1^j, c_2^j означают, что для каждого признака могут быть выбраны свои границы. Для логических закономерностей такого вида есть простая геометрическая интерпретация — это построенные по объектам обучающей выборки X прямоугольные гиперпараллелепипеды. Пример для случая объектов с двумя признаками на рисунке 1. Стандартным критерием качества согласно [13] и [16] называется:

$$\Phi(\varphi(x)) = |\{x \in c\colon \varphi(x) = 1\}|$$

2.2 Методы поиска логических закономерностей

Пусть множество $\mathcal{P}$ параметрических элементарных предикатов конечно. Тогда множество $\mathcal{K}_K$ логических закономерностей, состоящих из K термов, имеет вид:

$$\mathcal{K}_K = \{\varphi(x) = P_1(x) \& P_2(x) \& \ldots \& P_K(x) \mid P_1, \ldots, P_K \in \mathcal{P}\}$$

Количество логических закономерностей: $|\mathcal{K}_K| = |\mathcal{P}|^K$. Существует большое количество алгоритмов, которые по-разному подходят к построению множества логических закономерностей. К таким алгоритмам относятся «КОРА» [12], «IREP» и «RIPPER» [5], «SLIPPER» [1], «ID3» [6], «C4.5rules» [4], [3]. В

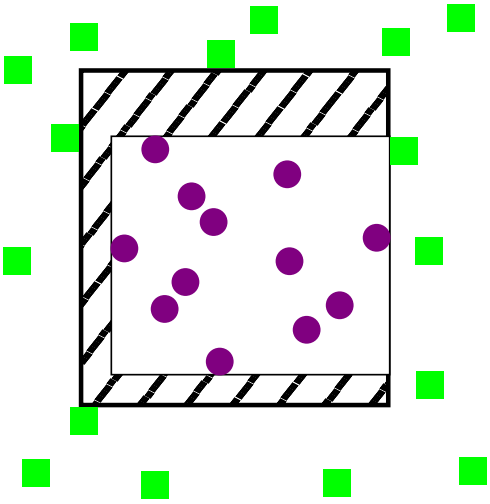


Рис. 1: Два класса (фиолетовые круги и зеленые квадраты) и область, в которой можно построить логическую закономерность (диагональная штриховка).

качестве классического примера 1 такого алгоритма можно привести двухклассовую версию «IREP». Это жадный алгоритм построения правил, работающий по принципу «разделяй и властвуй» (divide and conquer). Этот принцип проявляется в том, что правила создаются по одному, а объекты, которые покрываются созданным правилом, исключаются из рассмотрения. Многие алгоритмы («КОРА», «ТЭМП» [15]) по смыслу очень похожи и отличаются только критериями останова и способами обработки полученных правил.

В системе «Распознавание» реализован алгоритм «логические закономерности», который подробно описан в работах [13], [16], [17]. Основная идея этого алгоритма заключается в следующем: для каждого объекта записывается дискретная оптимизационная задача, решение которой однозначно определяет логическую закономерность. В результате решения данных оптимизационных задач находятся необходимые параметры логических закономерностей, определяющие левые и правые границы интервалов. В итоге получается правило, которое как бы «натянуто» на некоторые объекты обучающей выборки. Точный комбинаторный метод основан на методе ветвей границ.

Многие алгоритмы поиска логических закономерностей имеют следующие недостатки:

1. Количество найденных правил может быть довольно большим. Из-за этого пользователю может быть сложно интерпретировать эти правила.

Algorithm 1: Incremental Reduced Error Pruning (IREP)

Вход : Pos — множество элементов класса +1
Neg — множество элементов класса −1
Выход: Rules — множество правил
Функция `IREP(`*Pos, Neg*`)`
  Завести пустой набор правил Ruleset;
  Пока *Pos не опустеет*
    `/* Нужно обучить и упростить очередное правило Rule */`
    Разделить (Pos, Neg) на (GrowPos, GrowNeg) и (PrunePos, PruneNeg);
    Обучить правило Rule по паре множеств (GrowPos, GrowNeg);
    Упростить правило Rule по паре множеств (PrunePos, PruneNeg);
    Если *доля ошибок Rule на (PrunePos, PruneNeg) больше 50%* **то**
      Вернуть *Ruleset*;
    Иначе
      Добавить правило Rule в Ruleset;
      Убрать из (Pos, Neg) объекты, которые покрывает правило Rule;
  Вернуть *Ruleset*;

2. Найденные логические закономерности могут получиться очень похожими или даже вырожденными. В случае алгоритма из системы «Распознавание» это связано с близостью решаемых оптимизационных задач.

Первая проблема ставит интересную задачу поиска множества логических закономерностей, которое содержит меньшее число элементов и при этом имеет качество классификации, которое хотя бы не хуже, чем у исходного множества правил. Вторая проблема хорошо иллюстрируется примером 1, взятом из [15].

Пример 1 *Предположим, что в некоторой экспертной комиссии есть два эксперта, мнения которых всегда (или почти всегда) совпадают по всем вопросам. Интуитивно понятно, что если убрать одного из экспертов, то качество работы новой комиссии будет таким же (или почти таким же).*

Поэтому понятно, что способ обработки логических закономерностей должен по-возможности оставлять различные правила. Наконец, от вырожден-

ных правил, которые покрывают малое число объектов (например, один) стоит и вовсе избавляться.

3 Обработка логических закономерностей

Пусть в результате работы некоторого метода поиска логических закономерностей было получено множество правил $\{\varphi_1(x), \ldots, \varphi_n(x)\}$. Для того, чтобы обработать это множество, предлагается провести его кластеризацию с помощью дисперсионного критерия. Основная схема этого подхода, предложенного В. В. Рязановым, выглядит следующим образом:

1. Каждой логической закономерности φ_j^i из множества $\mathcal{K}_i = \{\varphi_1^i(x), \ldots, \varphi_t^i(x)\}$ логических закономерностей, покрывающих класс y_i, поставить в соответствие некоторый вектор $\boldsymbol{z}_j$, описывающий эту закономерность.

2. Провести кластеризацию векторов $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t$ на $k \leqslant t$ кластеров и найти центры этих кластеров $\boldsymbol{z}_1^*, \ldots, \boldsymbol{z}_k^*$.

3. По центрам кластеров восстановить $\varphi_1^*(x), \ldots, \varphi_k^*(x)$ — некоторые, вообще говоря новые, логические закономерности для класса y_i.

3.1 Способы представления логических закономерностей

В этой работе рассмотрено два способа представить логические закономерности: с помощью бинарного вектора и с помощью вектора левых и правых границ.

Представление с помощью бинарного вектора было исследовано в [14]. Идея заключается в том, чтобы каждой логической закономерности $\varphi(x)$ поставить в соответствие вектор $\boldsymbol{z} \in \{0, 1\}^l$ так, что

$$\boldsymbol{z}_i = \begin{cases} 1 & \text{если } \varphi(x_i) = 1 \\ 0 & \text{иначе} \end{cases}$$

В данной работе предлагается использовать представление с помощью вектора левых и правых границ, которое использует введенное в 2.1 представление логической закономерности $\varphi(x)$ в виде параметрических элементарных предикатов (5). При таком подходе вектор $\boldsymbol{z}$ принимает вид:

$$\boldsymbol{z} = (c_1^1, c_2^1, c_1^2, c_2^2, \ldots, c_1^D, c_2^D)$$

3.2 Алгоритм кластеризации логических закономерностей

Нужно разбить полученное описание множества логических закономерностей $\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_t$ на $k \leqslant t$ непересекающихся кластеров $\boldsymbol{S} = \{S_1, S_2, \ldots, S_k\}$ таким образом, чтобы минимизировать функционал 2.

$$\boldsymbol{S}^* = \arg\min_{\boldsymbol{S}} \sum_{i=1}^{k} \sum_{\boldsymbol{z}_j \in S_i} \|\boldsymbol{z}_j - \boldsymbol{\mu}_i\|^2 \tag{2}$$

Тогда алгоритм кластеризации (алгоритм Ллойда [7], алгоритм K-средних [9]) примет вид алгоритма 2.

Algorithm 2: Алгоритм кластеризации (Ллойда, K-средних)

Вход : Описание закономерностей $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t$

Выход: $\boldsymbol{S}^* = S_1, \ldots, S_k$ — разбиение на непересекающиеся кластеры

Функция *Кластеризовать*$(\boldsymbol{z}_1, \ldots, \boldsymbol{z}_t)$

Инициализировать центры кластеров $\boldsymbol{\mu}_1^1, \ldots, \boldsymbol{\mu}_k^1$;

Пока *кластеризация не стабилизируется*

`/* распределить объекты по кластерам, при этом */`

`/* минимизировать функционал качества 2 */`

$S_i^t = \left\{\boldsymbol{z}_p : \|\boldsymbol{z}_p - \boldsymbol{\mu}_i^t\|^2 \leqslant \|\boldsymbol{z}_p - \boldsymbol{\mu}_j^t\|^2, \forall j : 1 \leqslant j \leqslant k\right\}$;

`/* пересчитать центры кластеров */`

$\boldsymbol{\mu}_i^{t+1} = \frac{1}{|S_i^t|} \frac{\sum_{\boldsymbol{z}_j \in S_i^t} \beta_j \boldsymbol{z}_j}{\sum_{\boldsymbol{z}_j \in S_i^t} \beta_j}$;

Вернуть $S_1^t, \ldots, S_k^t$;

3.3 Построение логических закономерностей по результату кластеризации

После того, как алгоритм кластеризации находит разбиение $\boldsymbol{S}^*$, необходимо по центрам кластеров $\hat{\boldsymbol{z}}_1^*, \ldots, \hat{\boldsymbol{z}}_k^*$ построить логические закономерности $\varphi_1^*(x), \ldots, \varphi_k^*(x)$.

В случае, когда логические закономерности исходно были представлены в виде бинарных векторов, полученный центр кластера $\hat{\boldsymbol{z}}_i^*$ уже, вообще говоря, не является бинарным вектором. Поэтому для каждого вектора $\hat{\boldsymbol{z}}_i^*, i = 1, \ldots, k$ выбирается некоторый порог бинаризации θ_i таким образом, чтобы логическая закономерность $\varphi_i^*(x)$, отвечающая представлению:

$$z_i^* = \begin{cases} 1 & \text{если } \hat{z}_i^* \geqslant \theta_i \\ 0 & \text{иначе} \end{cases}$$

была наиболее информативной среди всех рассматриваемых значений порога θ_i. Информативность логической закономерности рассчитывается с помощью известных критериев информативности, таких как энтропийный критерий IGain. Подробный обзор использования разных критериев для этой задачи представлен в [14].

В случае, когда логические закономерности исходно описывались с помощью вектора левых и правых границ, полученный центр кластера $\hat{z}_i^*$ можно напрямую использовать в качестве описания z_i^* некоторой новой логической закономерности. То есть нет необходимости проводить отдельную процедуру по восстановлению описания.

4 Вычислительные эксперименты

Основной целью вычислительных экспериментов было реализовать подход к обработке множества логических закономерностей с помощью кластеризации по дисперсионному критерию, описанного в разделе 3. Второй целью вычислительного эксперимента было сравнить способы представления логических закономерностей, описанные в разделе 3.1. Наконец, третьей целью вычислительного эксперимента было понять, возможно ли с помощью описанного подхода получить множество правил с меньшим числом элементов и сравнимым качеством классификации.

4.1 Исходные данные и условия экспериментов

Исходные данные были взяты из репозитория UCI [11]. Ниже представлена сводная таблица 1 по использованным данным.

Выборка	Всего объектов	Объекты по классам	Признаки
Iris[1]	150	50/50/50	4
Wine[2]	178	59/71/48	13
Climate[3]	540	46/494	11
Ionosphere[4]	351	126/255	34

Таблица 1: Сводная таблица по использованным данным

Выборка Iris[1] ставит задачу классификации растений семейства ирисов. Выборка Wine[2] содержит данные химического анализа вина, полученного от трех разных итальянских виноделов. Выборка Climate[3] содержит информацию об удачных и неудачных запусках симулирования погоды. Наконец, выборка Ionosphere[4] исследует свободные электроны в ионосфере.

4.2 Результаты экспериментов

Для выполнения основной цели вычислительного эксперимента было сделано следующее:

1. Был найден способ работать в системе «Распознавание» под операционной системой семейства Linux. Детали смотрите в Приложении А.

2. Система «Распознавание» использует собственный формат файла для исходных выборок с данными. Было найдено описание этого формата и написана программа по преобразованию исходных выборок в этот формат. Детали смотрите в Приложении Б.

3. Результаты работы алгоритма «логические закономерности» системы «Распознавание» представляются в виде отчетов. Была написана программа для извлечения построенных правил из этих отчетов.

4. Для реализации подхода к обработке логических закономерностей была написана программа, которая:

   (a) Представляет построенные правила в виде описаний из раздела 3.1

   (b) Производит кластеризацию правил с помощью алгоритма 2

   (c) В зависимости от выбранного способа описания правил строит логические закономерности на основе результатов кластеризации.

Для выполнения второй и третьей целей вычислительного эксперимента написанный код был запущен на выборках, представленных в таблице 1. При этом использовалась следующая схема эксперимента:

[1] `http://archive.ics.uci.edu/ml/datasets/Iris`
[2] `http://archive.ics.uci.edu/ml/datasets/Wine`
[3] `https://archive.ics.uci.edu/ml/datasets/Climate+Model+Simulation+Crashes`
[4] `https://archive.ics.uci.edu/ml/datasets/Ionosphere`

1. Исходная выборка делилась пополам на подвыборки обучения и контроля. При этом для подвыборок сохранялись одинаковые пропорции классов.

2. На подвыборке обучения в системе «Распознавание» настраивался классификатор «логические закономерности».

3. Логические закономерности настроенного классификатора импортировались в программу, реализующую описанный подход к обработке множеств логических закономерностей.

4. По этому набору логических закономерностей с помощью алгоритма «простого голосования» [15] проводилась классификация контрольной подвыборки.

5. Далее проводилось описание логических закономерностей одним из двух способов из раздела 3.1. Полученные описания проходили кластеризацию на $t = 2, 3, \ldots, n, \ldots$ кластеров.

6. По полученным на очередном шаге t центрам кластеров восстанавливались новые логические закономерности, которые затем использовались в алгоритме «простого голосования» для классификации контрольной подвыборки.

7. Полученные значения доли правильно классифицированных объектов для разных способов представления и для разного числа кластеров t выносилось на график.

Результаты запусков различных конфигураций метода обработки множества логических закономерностей с помощью кластеризации представлены на графиках 2a, 2b, 3a, 3b, 4a и 4b. Сплошная линия красного цвета обозначает качество классификации исходного множества логических закономерностей. Синяя линия с кружками обозначает предложенный в данной работе подход, зеленая линия с треугольниками и бирюзовая с квадратами — подходы с бинаризованным описанием логических закономерностей, построенных с помощью критериев информативности IGain и Stat соответственно [15].

Критерий информативности IGain — это энтропийный критерий информативности правила $\varphi(x)$, отвечающего классу y_i, который имеет вид:

$$\text{IGain} = H\left(\frac{P}{P+N}, \frac{N}{P+N}\right) - \frac{p+n}{P+N} H\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - \frac{P+N-p-n}{P+N} H\left(\frac{P-p}{P+N-p-n}, \frac{N-n}{P+N-p-n}\right) \quad (3)$$

В формуле (3) введено обозначение энтропии $H(p,q) = -p\log_2(p) - q\log_2(q)$, P, N — число объектов класса y_i и не y_i соответственно, а p, n — число объектов из класса y_i и не из класса y_i, которые покрывает логическая закономерность $\varphi(x)$.

Критерий Stat имеет следующий вид:

$$\text{Stat} = -\ln \frac{C_{P_1}^{p_1} \dots C_{P_K}^{p_K}}{C_l^{p_1+\dots+p_K}} \quad (4)$$

В формуле (4) используется обозначение $P_1, \dots, P_K$ — количество объектов в классе $1, \dots, K$ и $p_1, \dots, p_K$ — количество объектов класса, покрываемых закономерностью $\varphi(x)$. Наконец, $C_n^k = \frac{n!}{k!(n-k)!}$ обозначает биномиальный коэффициент.

На выборках Iris и Wine (графики 2a и 2b) в обоих случаях способ обработки множества логических закономерностей с помощью вектора левых и правых границ, предложенный в данной работе, оказался заметно лучше способа, изложенного в [14]. Оба графика показывают, что обработка с помощью вектора левых и правых границ достигает качества классификации исходного множества закономерностей, в то время как методы из [14] имеют заметно худшее качество.

Из графика 2a видно, что достаточно всего 20 логических закономерностей вместо 29 исходных. В то же время график 2b говорит о том, что после обработки методом левых и правых границ можно оставить всего 6 логических закономерностей вместо исходных 36.

На выборке Climate (графики 3a и 3b) наблюдается противоположная картина. В случае чистых логических закономерностей (график 3a) имеется участок любопытно высокого качества классификации для логических закономерностей, построенных с помощью бинарного представления и критерия IGain. С увеличением числа логических закономерностей в кластере этот эффект пропадает и подход из [14] становится сравним с подходом из данной

работы. Аналогичное поведение можно видеть на графике с частичными закономерностями 3b, на котором правила, построенные с помощью бинарного представления и критерия IGain, снова показывают лучшее качество классификации при сравнительно малом числе логических закономерностей в кластере. Этот эффект опять пропадает с ростом числа закономерностей в кластере и методы из [14] становятся хуже метода из данной работы. Стоит отметить, что в выборке Climate сильно несбалансированные классы, что может быть причиной наблюдаемого поведения.

В случае графика 3a можно выбрать около 20–25 закономерностей вместо исходных 35 и добиться сравнимого качества классификации. При этом в случае использования бинарного представления и критерия IGain можно выбрать всего 4–9 правил. В то же время для графика 3a и подхода с вектором левых и правых границ можно остановиться на 35 логических закономерностях вместо исходных 63. При этом в случае использования бинарного представления и критерия IGain это количество можно снизить до 25.

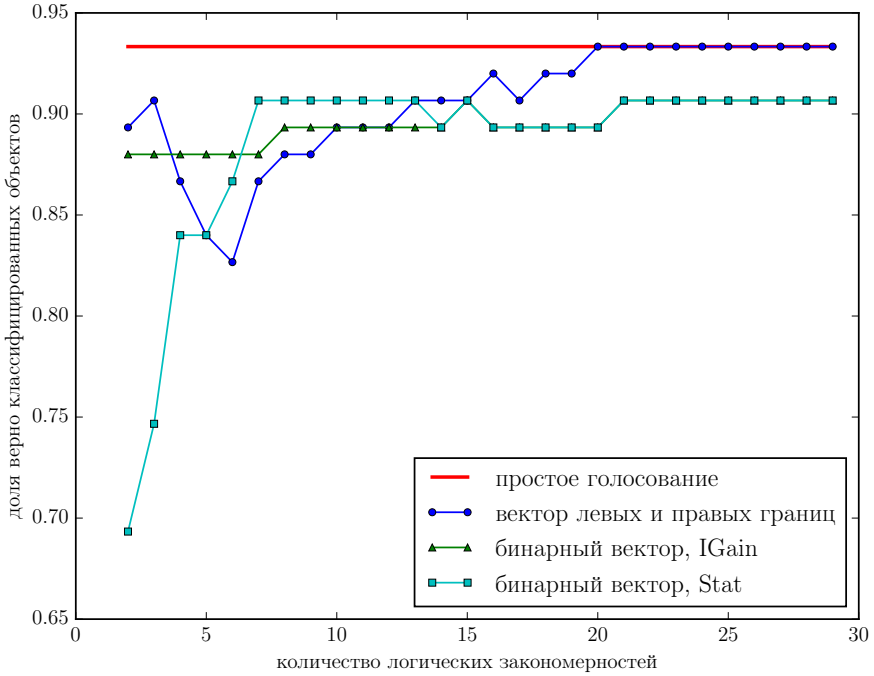
На выборке Ionosphere (графики 4a и 4b), в которой классы также не являются сбалансированными, проводилась последняя серия экспериментов. В случае, когда строились чистые логические закономерности (график 4a), методы из [14] показывают более хорошее качество классификации. С другой стороны, при кластеризации частичных логических закономерностей (график 4b) уже метод левых и правых границ демонстрирует лучшее качество классификации.

В случае графика 4a можно выбрать всего 1–2 правила вместо исходных 48 и добиться качества классификации, которое заметно выше исходного множества логических закономерностей. Это возможно скорее всего потому, что данные легко разделимы. В то же время для графика 4b можно выбрать около 30 правил вместо исходных 88 и при этом добиться качества классификации, которое превосходит качество классификации с помощью исходного множества логических закономерностей.

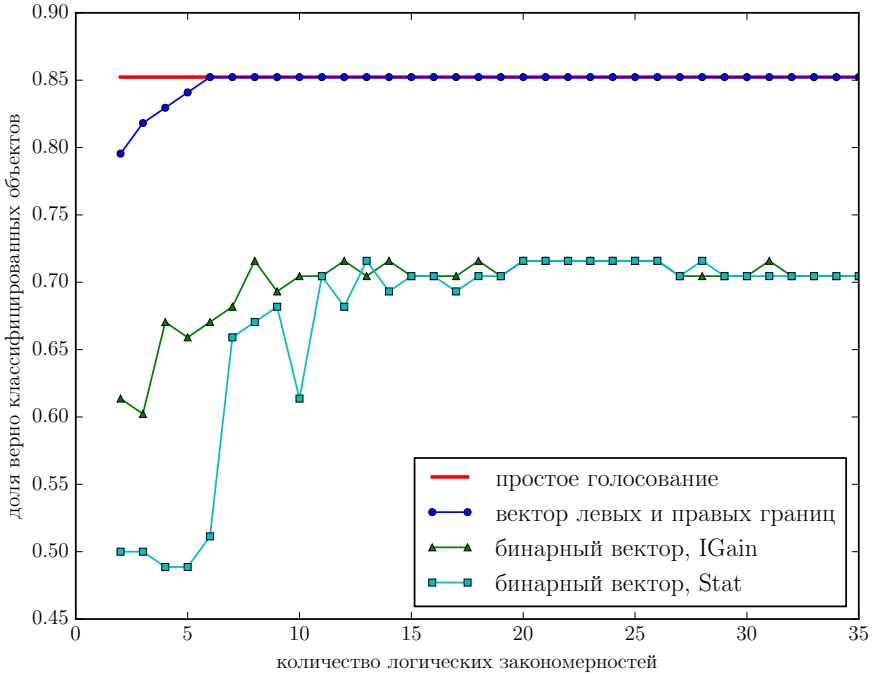
4.3 Обсуждение и выводы

1. С помощью обработки множеств логических закономерностей удается получить множество меньшей мощности, которое проще интерпретировать.

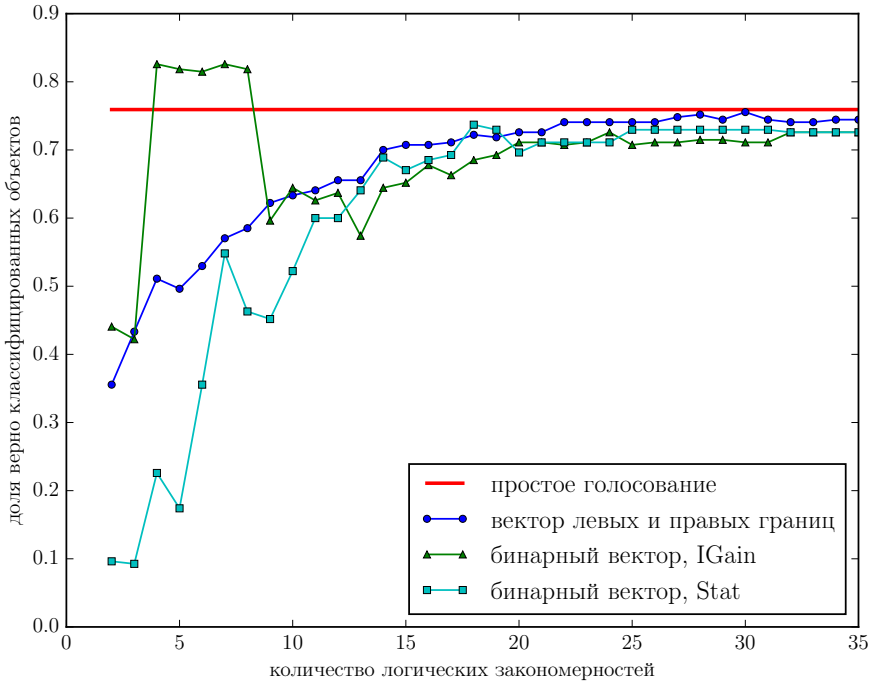
2. Обработка логических закономерностей, использующая описание правил



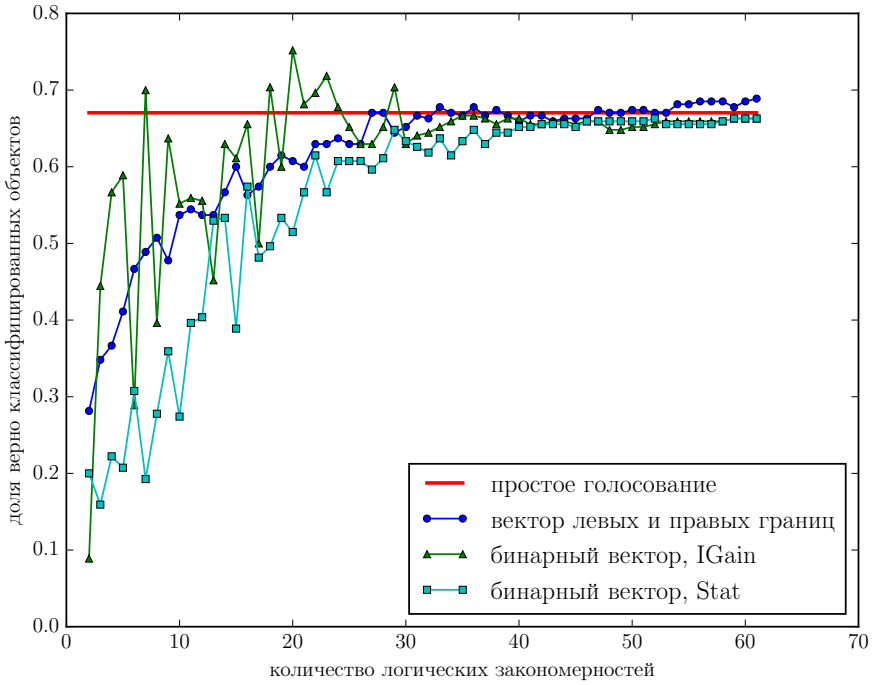
(a) Выборка Iris. Анализ графика в разделе 4.2.



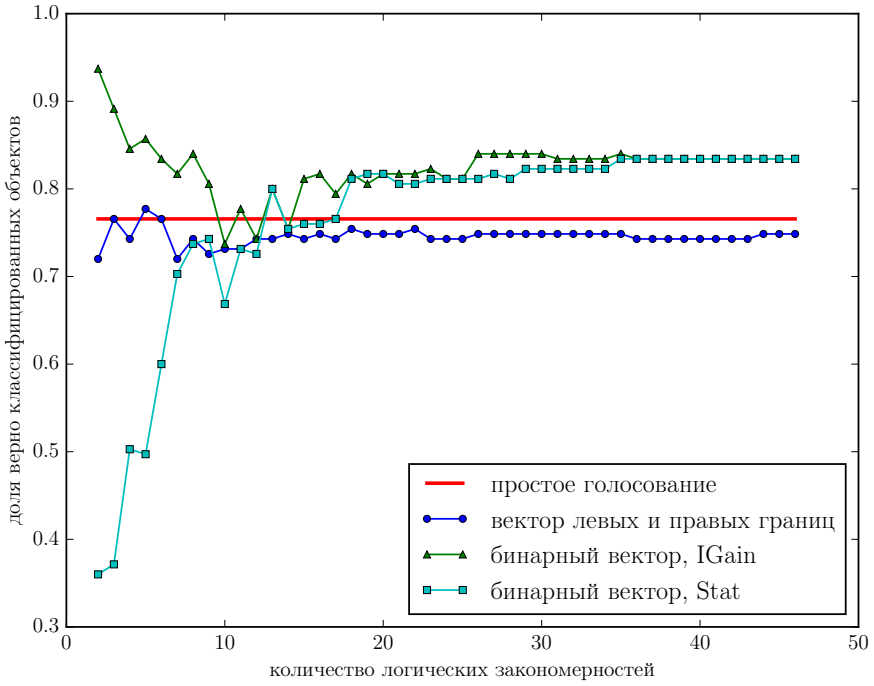
(b) Выборка Wine. Анализ графика в разделе 4.2.



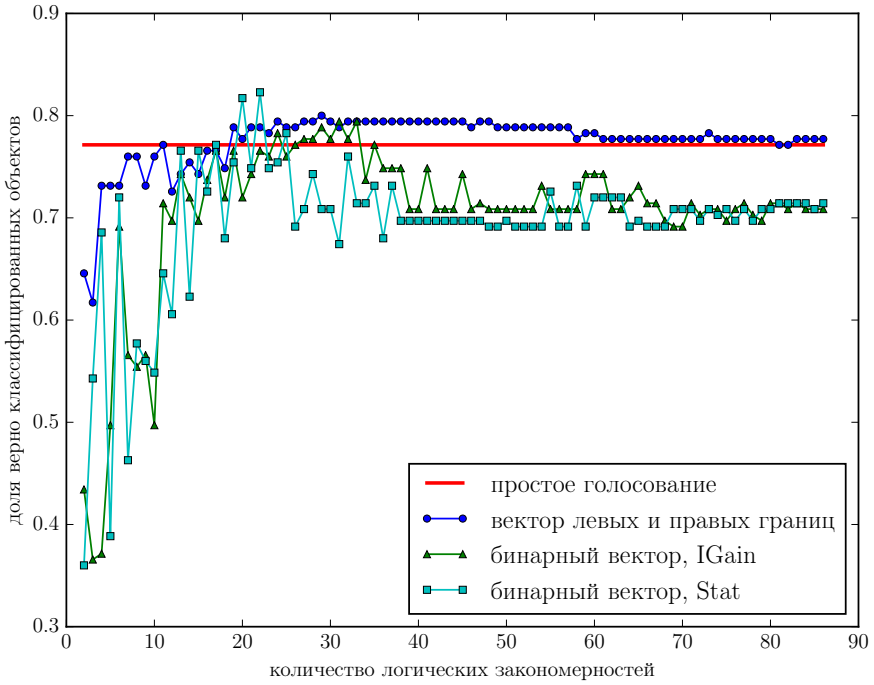
(a) Выборка Climate, исходные логические закономерности чистые. Анализ графика в разделе 4.2.



(b) Выборка Climate, исходные логические закономерности частичные. Анализ графика в разделе 4.2.



(a) Выборка Ionosphere, исходные логические закономерности чистые. Анализ графика в разделе 4.2.



(b) Выборка Ionosphere, исходные логические закономерности частичные. Анализ графика в разделе 4.2.

с помощью вектора левых и правых границ, строит логические закономерности, которые в целом показывают результаты как лучше (2a, 2b, 4b), так и хуже (4a), чем результаты обработки с использованием бинаризованных описаний правил из [14]. Также есть случаи, в которых поведение двух подходов примерно одинаково: 3a, 3b.

3. Обработка с использованием бинаризованных представлений правил иногда работает лучше 4a. Если же исходно используются не чистые, а частичные закономерности, то представление с помощью вектора левых и правых границ более устойчиво (3b, 4b) в том плане, что качество классификации оказывается сравнимым с исходным множеством логических закономерностей.

4. В результате обработки множества логических закономерностей как с помощью бинаризованного представления, так и с помощью вектора левых и правых границ, можно получить сравнимое с исходным качество классификации при меньшем количестве использованных правил.

5 Заключение

1. В данной работе был предложен подход к обработке множеств логических закономерностей с помощью кластеризации на основе дисперсионного критерия. В разделе 3 была сформулирована общая схема подхода, а в разделе 4 были поставлены эксперименты по его исследованию.

2. Получены обработанные множества логических закономерностей для задач классификации выборок Iris, Wine, Climate, Ionosphere.

3. Проведено сравнение метода обработки с использованием вектора левых и правых границ с методом обработки, в котором используется бинаризованное описание логических закономерностей из [14]

4. На практических задачах показано, что использование кластеризации при обработке множества логических закономерностей позволяет получить меньшее количество правил. При этом классификатор по схеме простого голосования имеет качество, сравнимое с исходным алгоритмом из системы «Распознавание».

A Система «Распознавание»

Система «Распознавание» [17] работает под управлением операционной системы семейства Microsoft Windows XP и выше. В этом приложении будет показано, как систему «Распознавание» можно использовать на UNIX-подобных операционных системах. Для этого нужно:

1. Установить свободное программное обеспечение Wine.
2. Настроить 32-битный префикс командой

   ```
   WINEARCH=win32 WINEPREFIX=~/.wine32 winecfg
   ```

3. Установить в созданный префикс недостающую библиотеку

   ```
   WINEPREFIX=~/.wine32 winetricks mfc42
   ```

4. Запустить систему «Распознавание» командой

   ```
   LC_CTYPE=ru_RU.utf8 WINEARCH=win32 WINEPREFIX=~/.wine32 \
   wine Recognition.exe
   ```

B Формат файлов *.tab

Система «Распознавание» [17] использует собственный формат файлов описания выборки. Здесь приведено описание этого формата:

Первая строка — заголовок. Представляет из себя числа, разделенные пробелом. Первое число — количество признаков каждого объекта выборки. Второе — количество классов. Далее идет число 0, за которым идет число объектов первого класса, затем сумма числа объектов первых двух классов, затем сумма числа объектов первых трех классов и так далее. Завершает заголовок число, обозначающее пропуск измерения в данных.

Каждая последующая строка содержит признаковое описание отдельного объекта, разные классы разделены пустой строкой.

Список литературы

Cohen W. W., *Singer Y.* A Simple, Fast and Effective Rule Learner // Proc. of the 16 National Conference on Artificial Intelligence. — 1999. — С. 335—342.

Hosmer D. W., *Lemeshov S.*, *Sturdivant R. X.* Applied Logistic Regression. — John Wiley & Sons, Inc., 2013.

Quinlan J. R. Bagging, Boosting, and C4.5 // AAAI/IAAI, Vol. 1. — 1996. — С. 725—730.

Quinlan J. R. C4.5: Programs for machine learning. — Morgan Kaufmann, San Francisco, CA, 1993.

Cohen W. W. Fast Effective Rule Induction // Proc. of the 12th International Conference on Machine Learning, Tahoe City, CA. — Morgan Kaufmann, 1995. — С. 115—123.

Quinlan J. R. Induction of Decision Trees // Machine Learning. — 1986. — Т. 1, № 1. — С. 81—106.

Lloyd S. Least Squares Quantization in PCM // IEEE Trans. Inf. Theor. — Piscataway, NJ, USA, 2006. — Сент. — Т. 28, № 2. — С. 129—137.

Bishop C. M. Neural Networks for Pattern Recognition. — Clarendon Press, 1995. — (Advanced Texts in Econometrics).

Macqueen J. B. Some methods for classification and analysis of multivariate observations. — 1967.

Cortes C., *Vapnik V.* Support-vector networks // Machine Learning. — 1995. — Т. 20, № 3. — С. 273—297.

Lichman M. UCI Machine Learning Repository. — 2013. — URL: `http://archive.ics.uci.edu/ml`.

Вайнцвайг М. Н. Алгоритм обучения распознаванию образов «Кора» // Алгоритмы обучения распознаванию образов / под ред. В. Н. Вапник. — М.: Советское радио, 1973. — С. 110—116.

Ковшов Н. В., *Моисеев В. Л.*, *Рязанов В. В.* Алгоритмы поиска логических закономерностей в задачах распознавания // Ж. вычисл. матем. и матем. физ. — 2008. — Т. 48, № 2. — С. 329—344.

Новиков М. С. Исследование задачи кластеризации логических закономерностей, представленных булевыми векторами. — 2014.

Воронцов К. В. Лекции по логическим алгоритмам классификации. — 2010. — URL: `http://www.machinelearning.ru/wiki/images/3/3e/Voron-ML-Logic.pdf`.

Рязанов В. В. Логические закономерности в задачах распознавания (параметрический подход) // Ж. вычисл. матем. и матем. физ. — 2007. — Т. 47. — С. 1793—1808.

Журавлёв Ю. И., *Рязанов В. В.*, *Сенько О. В.* «Распознавание». Математические методы. Программная система. Практические применения. — М.: Фазис, 2006.