Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)
Факультет управления и прикладной математики
Кафедра Информатики

Замятина Екатерина Сергеевна

Решение задачи многоклассовой классификации с помощью вложенных дихотомий на основе кластеризации

010900 — Прикладные математика и физика

Выпускная квалификационная работа бакалавра

Научный руководитель:
Профессор, академик РАЕН ,
д. ф.-м. н.
Владимир Васильевич Рязанов

Москва

2018

Содержание

1 Введение. 4

1.1 Вводная часть 6

1.2 Постановка задачи 7

1.3 Гипотеза 8

2 Методы многоклассовой классификации. 9

2.1 Метод вложенных дихотомий. 9

2.2 Методы бинарной классификации. 13

3 Индексы 16

3.1 C индекс 17

3.2 Dunn индекс 18

3.3 Trace_W индекс 19

4 Предлагаемые алгоритмы 19

4.1 Расстояние 20

4.2 Алгоритм NDCIC 21

4.3 Алгоритм NDCID 22

4.4 Проблемы методов NDCID, NDCIC 24

4.5 Алгоритм NDCIW 25

5 Сравнение алгоритмов 27

5.1 Результаты с использованием логистической регрессии как базового классификатора 27

5.2 Результаты с использованием решающих деревьев 29

5.3 Результаты с использованием больших наборов данных 29

6 Заключение 32

Список литературы 33

Аннотация

Задача многоклассовой классификации — это задача классификации данных на более двух классов. Существует множество методов ее решения, важным классом которых является метод бинарного иерархического разделение, то есть расположение классов в виде двоичного дерева. Дерево создается таким образом, что классы в каждом дочернем узле делятся на 2 кластера. Процесс построения структуры продолжается до тех пор, пока листовые узлы не будут содержать только один класс. При этом в каждом узле дерева находится двоичный классификатор. Большинство методов построения вложенных дихотомий или дерева используют случайную стратегию, которая не гарантирует нахождения хорошего оптимального разбиения. В данной работе предлагаются новые неслучайные методы построения вложенных дихотомий, использующие идею уменьшения ошибок классификации путем разделения на верхних уровнях дерева классов в два макрокласса таким образом, что в одном макроклассе группируются наиболее схожие классы. Чтобы оценить эффективность предлагаемых методов, произведено сравнение их с уже существующими методами вложенных дихотомий. Выявлена их эффективность по метрики accuracy.

Ключевые слова: *Вложенные дихотомии, многоклассовая классификация, кластеризация.*

1 Введение.

Актуальность темы. Многоклассовая классификация встречается повсеместно. Часто ее сводят к более простой задаче бинарной классификации, так как для ее решения уже предложено много успешных алгоритмов. Популярные методы адаптации бинарных классификаторов к задачам с несколькими классами включают в себя парную классификацию (AVA), классификацию один-против-всех (OVA) и метод CORRECTING OUTPUT CODES. Однако в относительно недавних исследованиях было показано, что другой метод, под названием "ансамбли вложенных дихотомий является более эффективной заменой выше описанных методов. Сейчас активно разрабатываются новые алгоритмы, основанные на методе вложенных дихотомий, которые работают быстрее и точнее.

Цели работы.

- Анализ текущих методов вложенных дихотомий.
- Разработка алгоритмов многоклассовой классификации на основе иерархических деревьев с использованием кластеризации.
- Сравнение существующих и предложенных методов.

Методы исследований. При написании алгоритмов использовались элементы комбинаторики и линейной алгебры. Для программной реализации использовался язык программирования python 2.7.

Научная новизна. Разработаны эффективные методы бинoризации. Произведено сравнение с текущими методами по скорости и качеству работы на основе метрики accuracy. Доказана их эффективность.

Работа организована следующим образом. В главе 1 описывается решаемая задача. В главе 2 обсуждается существующие методы решения. В главе 3 описывается использованные индексы кластеризации. В главе 4 описываются предлагаемые новые методы. В 5 главе произведено сравнение новых моделей с существующими.

1.1 Вводная часть

Задача многоклассовой классификации — это задача присвоения данным метки класса, когда число классов больше двух. При наборе данных обучения вида (x_i, y_i), где $x_i \in R^n$ - i-й пример, а $y_i \in \{1, ..., K\}$ - метка класса i, цель задачи найти модель обучения H такую, что $H(x_i) = y_i$ для новых тестовых данных. Проблема просто формулируется в случае двух классов, где метка y_i равна 1 или -1, где $i \in \{0, 1\}$. Для таких задач было предложено несколько успешных методов, некоторые из которых могут быть расширены на многоклассовый случай путем разбиения исходной многоклассовой задачи на набор бинарных задач. Было показано, что метод "ансамбль вложенных дихотомий"является наиболее эффективным методом биноризации. Так результаты [1, 2] показывают, что подход AVA "все против всех"в целом лучше, чем подход OVA «один против всех». А результаты доказывают [3], что "ансамбль вложенных дихотомий"в зависимости от используемого базового классификатора может превзойти по качеству методы OVA, AVA, ERROR CORRECTING OUTPUT CODES. Метод "ансамбль вложенных дихотомий"предполагает, что все иерархические деревья, которые он строит, равновероятны и нет никакого резона выбирать какое-то конкретное. Но тогда получается, что когда двоичный классификатор во вложенной дихотомии совершает ошибку,то она распространяется на более низкие уровни дерева, где уже не может быть исправлена. По этой причине важно уменьшить количество ошибок на верхних уровнях дерева. Следуя этой идее, в этой работе предлагается построить вложенные дихотомии, в которых классы, которые легче отделяются, рассматриваются сначала, то есть на верхних уровнях дерева, а те классы, которые сложнее отделить, откладываются до нижнего уровня.

1.2 Постановка задачи

Дана выборка $X = \{x_i | i \in \{1, 2, ..., m\}\}$, с помощью обучающих данных нужно на тестовой выборке предсказать к какому классу $P_j \in \{P_i | i \in \{1, 2, ..c\}\}$ принадлежит объект. Будем строить бинарное дерево следующим образом: поместим в корень дерева все классы, а в каждой последующей вершине оставим только часть из них. Таким образом в любом листе дерева в конечном итоге будет содержаться только один класс. Множество классов в каждом узле назовем макроклассом. Тогда получается, что на одном уровне дерева всегда содержатся два макрокласса - K_1, K_2. Если I_1 - множество номеров классов из макрокласса K_1, а I_2 - из K_2, то

$$K_1 = \bigcup_{i \in I_1} P_i \tag{1.1}$$

$$K_2 = \bigcup_{i \in I_2} P_i \tag{1.2}$$

Задача посвящена локальной оптимизации: добавлению класса P_i к K_1 , при его исключении из K_2 или наоборот. После построения на каждом узле дерева локально оптимальной бинаризации будет сформировано N локально-оптимальных дихотомий. Для каждой дихотомии строится алгоритм бинарной классификации A_i. Таким образом, для каждого обьекта сформурется бинарный вектор $\vec{beta} = (\beta_1, \beta_2, ..., \beta_m)^T$, который представляет собой путь от корня дерева до листа – конкретного класса. По этому вектору можно однозначно определить конечный класс объекта.

1.3 Гипотеза

Основная гипотеза состоит в том, что разбиение на макроклассы случайным и неслучайным образом дают разное качество модели. Выгоднее разбивать множество всех классов на такие допустимые макроклассы, которые дают «хорошую» дихотомию, с низким количеством ошибок. То есть создать два макрокласса из исходных, которые будут оптимизировать выбранный «внутренний критерий» или индекс.

2 Методы многоклассовой классификации.

Было предложено несколько алгоритмов для решения проблемы многоклассовой классификации. Первая категория включает в себя решающие деревья, нейронные сети, k-Nearest Neighbor, классификатор Naive Bayes и метод опорных векторов. Вторая категория состоит из подходов, использующих преобразование проблемы многоклассовой классификации в набор задач двоичной классификации, которые эффективно решаются с использованием двоичных классификаторов. Рассмотрим лишь часть методов из последней группы, а именно методы вложенных дихотомий, так как они наиболее близки к поставленной задаче.

2.1 Метод вложенных дихотомий.

Во вложенной дихотомии набор классов разбивается на два подмножества или макрокласса рекурсивно, пока в каждом макроклассе не останется только один класс. Вложенные дихотомии можно представить в виде двоичных древовидных структур (рис.1). В каждом узле дерева построен двоичный классификатор для классификации экземпляров, принадлежащих макроклассам. Хорошей особенностью вложенных дихотомий является то, что оценки вероятности принадлежности класса объекту могут быть вычислены естественным образом, если бинарный классификатор, используемый на каждом узле, может выводить двухклассовые оценки вероятностей.

2.1.1 Существующие методы, основанные на методе вложенных дихотомий

Первоначальная структура ансамблей вложенных дихотомий Фрэнка и Крамера была предложена в 2004 году [3]. В этой структуре каждое двоичное дерево из ансамбля выбирается из множества всевозможных деревьев, исходя из предположения, что вложенные дихотомии не отличаются друг от друга. Таким

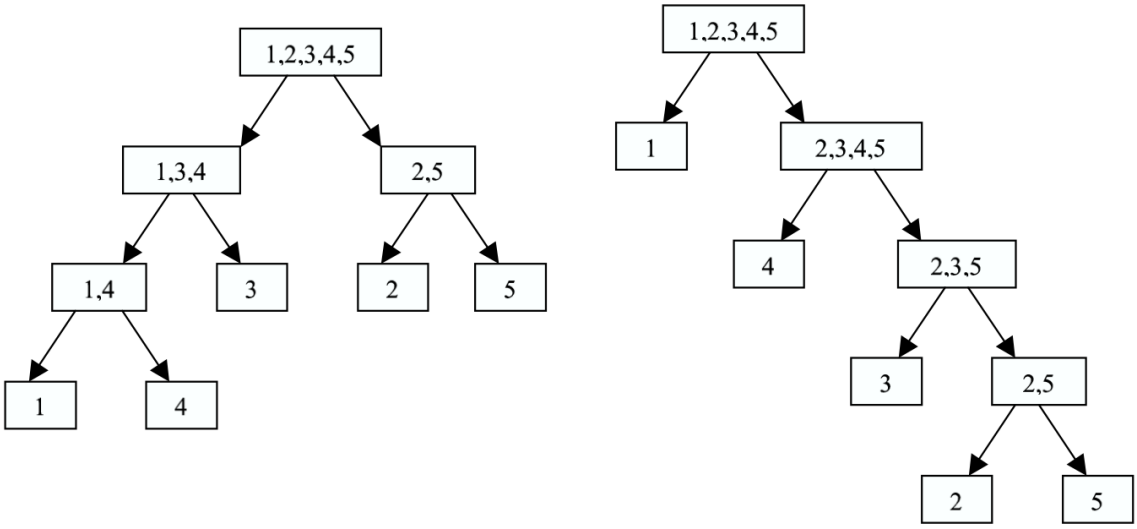


Рис. 1: Пример сбалансированного и не сбалансированного дерева

образом, создав ансамбль вложенных дихотомий, Фрэнк и Крамер достигли результатов, которые являются конкурентоспособными с другими методами бинаризации с использованием решающих деревьев и логистической регрессии, как модели классификаторов двух классов для каждого узла. Недостатком вложенной дихотомии, даже по сравнению с парной классификацией, является значительное увеличение времени выполнения. Хотя для парной классификации AVA требуется выполнить $\frac{c(c-1)}{2}$ итераций бинарного классификатора для проблемы обучения с 'c' классами, каждая проблема намного меньше исходной задачи, поскольку рассматриваются только данные из двух классов.

Кроме этого, отрицательно на время выполнения задачи может повлиять использование глубоких деревьев. Чтобы этого избежать, был проведен ряд адаптаций. В 2005 году Донг, Фрэнк и Крамер [4] предлагают ограничить пространство вложенных дихотомий только множеством сбалансированных деревьев, то есть при построении вложенной дихотомии всё также случайным образом разбиваются классы в макроклассы на каждом внутреннем узле, но при этом сохраняется равновесие: либо между количеством классов на каждом дочернем узле — этот метод получил название ECBND, либо между количеством данных — такой метод получил название EDBND. Описанные выше методы регули-

рует глубину деревьев, что уменьшает размер данных обучения для каждого бинарного классификатора и, таким образом, положительно влияет на время выполнения. При этом, было эмпирически показано, что качество классификации значительно не меняется.

Чтобы построить ансамбль вложенных дихотомий, необходимо выбрать набор бинарных деревьев из пространства всех бинарных деревьев. Размер этого пространства растет очень быстро по мере увеличения количества классов. Фрэнк и Крамер вычислили, что число потенциальных вложенных дихотомий есть (2c - 3) !! для задачи 'c' классами [3]. Для примера: в задаче с 11 классами это означает 34,459,425 различных деревьев. Количество сбалансированных деревьев при этом также по-прежнему очень велико, что дает 113 400 возможных вложенных дихотомий для проблемы с 10 классами [4].

Исходная структура ансамблей вложенных дихотомий использует рандомизацию не только для построения деревьев, но и для построения ансамбля, т. е. структура каждой вложенной дихотомии в ансамбле выбирается случайным образом. В 2010 году Ровридес, Гарса-Орио и Маудес [5] исследовали использование других методов выбора ансамбля в сочетании с вложенными дихотомиями. Авторы обнаружили, что улучшение точности может быть достигнуто путем использования мешков, AdaBoost и MultiBoost по сравнению со случайным выбором деревьев для ансамбля.

Абсолютно иным методом увеличения качества является влияние на структуру кластеров, а не ансамбля. Тим Лезарт, Бернхард Фанрингер и Эйб Франк [6] предложили стратегию разбиения подмножеств во вложенной дихотомии, которая получила название "выбор случайной пары". Метод использует прямой подход к обнаружению подобных классов, используя базовый классификатор, чтобы решить, какие классы легче отделять. При построении нового уровня дерева, из классов родительйской вершины выбираются произвольные два базовых класса c_1, c_2, после чего происходит обучение на данных с метками c_1, c_2 базового классификатора и применение его на всех данных из той же роди-

тельской вершины. Таким образом, получается разбиение на 2 неслучайных кластера, где на разделение влияла только "похожесть"на класс c_1 или c_2. При таком подходе все еще возможно строить ансамбль дихотомий, так как остается элемент случайности в выборе базовых классов. Количество вложенных дихотомий T при k классовой классификации оценивается рекуррентной формулой $T(k) = T(\frac{k}{3}) \cdot T(\frac{2 \cdot k}{3})$.

Методы выбора подмножества, которые описываются в данной работы, основаны на кластеризации, доводят эту идею до крайности и дают только одну вложенную дихотомию для любого заданного числа классов, потому что выбор подмножества класса является детерминированным. Эти методы не создают ансамбль, а используют только одно иерархическое бинарное дерево. Было предложено 2 таких метода [7]:

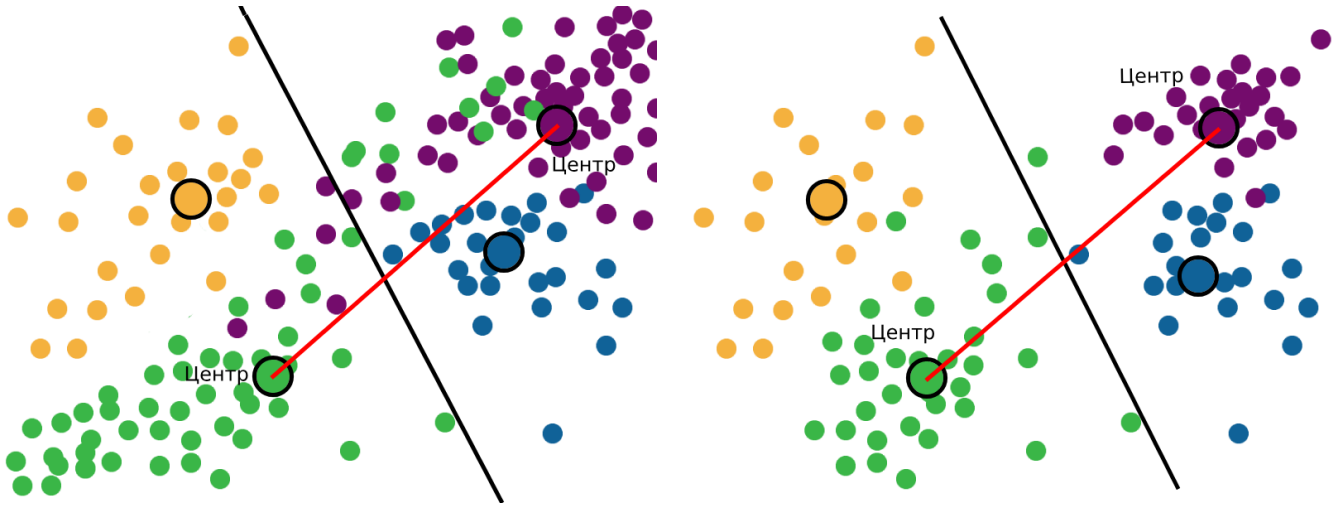
Метод NDC Рассмотрим множество объектов выборки. Каждый такой объект можно представить точкой в пространстве признаков. Объединив все точки, принадлежащие одному классу, вычисляется центроид данного класса. Метод NDC предполагает, что легче отделять те классы, у которых центроиды расположены дальше всего друг от друга. Поэтому ищется максимальное расстояние, и классы, на которых оно достигается, называемые групповыми центрами, служат центрами будущих кластеров. Остальные классы примыкают к одному из кластеров, в зависимости от их удаленности от них (рис. 2b). Недостатком метода NDC является то, что могут быть классы, которые, несмотря на то, что их центроиды далеки, трудно отделяются друг от друга из-за значительного перекрытия (рис. 2a). Возможен и противоположный случай: классы с близкими центроидами, не имеющие перекрытия и, следовательно, легко разделяемые.

Метод NDCR По этой причине был предложен усовершенствованный вариант метода NDC, который включает в себя вычисление радиуса класса. Радиус класса – это расстояние между центроидом класса и элементом внутри клас-

са, наиболее удаленным от центра тяжести. Вместо расстояния вычисляется функция $D(C_1, C_2)$:

$$D(C_1, C_2) = \frac{d(m_1, m_2)}{(r_1 + r_2)} \tag{2.1}$$

Таким образом, для измерения расстояния между двумя классами C_1 и C_2, учитывая центроид каждого класса – m_1 и m_2 соответственно, расстояние между ними – $d(m_1, m_2)$ и радиус каждого класса – r_1 и r_2 соответственно.



(a) плохой случай (b) хороший случай

Рис. 2: Иллюстрация метода NDC в пространстве двух признаков

2.2 Методы бинарной классификации.

Задача бинарной классификации решает вопрос: к какому классу из двух возможных принадлежит объект. Признаки объекта, или элементы строки матрицы 'объекты-признаки' A, могут быть как числовыми и не числовыми, непрерывными или детерминированными. При этом для представителей исходного множества известна метка класса – детерминированный признак. Классовая принадлежность остальных объектов неизвестна. Необходимо построить алгоритм, способный определить для произвольного объекта из тестовой выборки метку класса $\{P_j, j \in \{0, 1\}\}$.

Для решения сформулированной задачи могут быть использовано множе-

ство методов бинарной классификации, однако в этой работе будут рассматриваться только логистическая регрессия [8] и решающие деревья [9].

2.2.1 Логистическая регрессия

Логистическая регрессия – это статическая модель бинарной классификации, предсказывающая вероятность принадлежности метки 0 или 1 для объекта путём подгонки данных к логистической кривой. В алгоритме к множеству всех вещественных признаков $\{x_i | i \in \{1..N\}\}$ вводиться дополнительный единичный признак x_0. Делается предположение о том, что вероятность наступления события y=1 равна:

$$\mathbb{P}\{y = 1 \mid x\} = f(z),$$

где $z = \theta^T x = \theta_0 + \theta_1 x_1 + \ldots + \theta_n x_n$

$\vec{\theta}$ – подбираемые коэффициенты регрессии — вещественные числа, а f(z) – логистическая функция (иногда ее также называют сигмоидой или логит-функцией):

$$f(z) = \frac{1}{1 + e^{-z}}$$

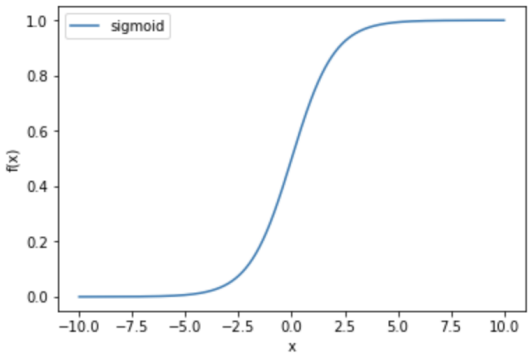


Рис. 3: Сигмоида

Выбор сигмоиды (рис. 3) обусловлен тем, что эта гладкая функция, ограниченная снизу нулем, а сверху единицей, поэтому она отлично, без дополнительных манипуляций, отображается в вероятностное пространство.

Для подбора параметров $\vec{\theta}$ обычно используется метод максимального правдоподобия, максимизирующий значение функции правдоподобия на обучающей выборке:

$$\hat{\theta} = \operatorname{argmax}_\theta L(\theta) = \operatorname{argmax}_\theta \prod_{i=1}^{m} \mathbb{P}\{y = y^{(i)} \mid x = x^{(i)}\}$$

Прологарифмировав выражение получим эквивалентное:

$$\ln L(\theta) = \sum_{i=1}^{m} \log \mathbb{P}\{y = y^{(i)} \mid x = x^{(i)}\} = \sum_{i=1}^{m} y^{(i)} \ln f(\theta^T x^{(i)}) + (1 - y^{(i)}) \ln(1 - f(\theta^T x^{(i)}$$

$$\theta^T x^{(i)} = \theta_0 + \theta_1 x_1^{(i)} + \cdots + \theta_n x_n^{(i)}$$

Для максимизации этой функции может быть применён, например, метод градиентного спуска.

2.2.2 Решающие деревья

Решающие деревья – семейство моделей, которые позволяют восстанавливать нелинейные зависимости произвольной сложности. Рассмотрим бинарное дерево, в котором:

- каждой внутренней вершине приписан предикат
- каждой листовой вершине приписано либо точное значение метки класса, либо вектор вероятности принадлежности каждому классу.

Рассмотрим теперь алгоритм a(x), называемый бинарным решающим деревом. Он стартует из корневой вершины и вычисляет значение предиката β во всех остальных вершинах. Если значение равно 0, то алгоритм переходит в левую дочернюю вершину, иначе – в правую. Процесс продолжается пока не будет достигнута листовая вершина, которой приписана единственная метка класса. Такой алгоритм называется бинарным решающим деревом.

На практике в большинстве случаев используются одномерные предикаты β, которые также называются решающими пнями. Они сравнивают значение

одного из признаков с порогом:

$$\beta(x_j, t) = [x_j < t]$$

[ ] - нотация Айверсона

Существуют и многомерные предикаты, например:

- линейные $\beta(x) = [(w, x) < t]$
- метрические $\beta(x, x_v) = [(x, x_v) < t]$, где точка x_v является одним из объектов выборки.

Опишем базовый алгоритм построения бинарного решающего дерева. Прежде всего ищется во всей обучающей выборке X наилучшее ее разбиение на две части ($R_1(j,t) = \{x|x_j < t\}$ и $R_2(j,t) = \{x|x_j > t\}$) с точки зрения заранее заданного функционала качества Q(X, j, t). Найдя наилучшие значения j и t, создается корневая вершина дерева, которой в соответствие ставят предикат $[x_j < t]$. Таким образом объекты разбиваются на две части – одни попадут в левое поддерево, другие – в правое. Для каждой из этих подвыборок рекурсивно повторяется процедура – происходит построение дочерней вершины. В каждой вершине проверяется выполнение некоторого условия останова и, если оно выполнилось, то прекращается рекурсия и объявляется вершина листом. Когда дерево построено, каждому листу ставится в соответствие единственный ответ.

3 Индексы

Основная идея этой работы – улучшение метода вложенных дихотомий, основываясь на метриках, используемых при кластеризации. Как и в методах NDC, NDCR [7] предполагается, что чем дальше классы по какой-либо метрике, тем их легче между собой отделить. Чтобы определить расстояние между классами, можно воспользоваться не только базовыми понятиями, такими как

расстояние между центроидами классов, а и различными метриками или составными индексами, которые используются в задачах кластеризации. Ниже приведена таблица некоторых таких индексов и правила принятия решения: max, min означают, что нужно выбрать соответственно наибольшее или наименьшее значение индекса.

Таблица 1: Внутренние метрики

Индекс	**Правило**
Trace_W	min
C index	min
Dunn	max
Baker-Hubert Gamma	max
G plus	min
Point-Biserial	max

Все последующие индексы были подсчитаны для разбиения на 2 кластера.

3.1 C индекс

Обозначим через A матрицу объектов-признаков размером N x p: каждая i-ая строка представляет собой наблюдение O_i, а каждый столбец представляет собой наблюдаемую переменную. Данные разбиваются на 2 кластера. Обозначим буквой $\vec{p}$ целочисленный вектор со значениями нуль и единица: 0 – если строка отнесена к кластеру 0, 1 – иначе. Размер $\vec{p}$ равен числу наблюдений N. Наблюдения, или строки матрицы A, могут быть представлены точками M_i где $i \in \{1..N\}$ в пространстве R^p. Обозначим за n_k количество точек в кластере k, где $k \in \{0, 1\}$. В кластере C_k найдутся $\frac{n_k \cdot (n_k - 1)}{2}$ пары различных точек (порядок точек не имеет значения). Обозначим через N_W общее число таких пар:

$$N_W = \sum_0^1 \frac{n_k(n_k - 1)}{2} = \frac{1}{2}(\sum_0^1 n_k^2 - \sum_0^1 n_k) = \frac{1}{2}(\sum_0^1 n_k^2 - N) \tag{3.1}$$

Общее количество пар точек обозначим как N_T

$$N_T = \frac{N(N-1)}{2} = \frac{1}{2}(\sum_{k=0}^{1} n_k)^2 - \frac{1}{2}\sum_{k=0}^{1} n_k = \frac{1}{2}(\sum_{k=0}^{1} n_k^2 + 2n_0 \cdot n_1) - \frac{1}{2}\sum_{k=0}^{1} n_k = N_W + n_0 \cdot n_1 = N_W + N_B \tag{3.2}$$

Рассмотрим расстояния между парами точек внутри каждого кластера.

- Обозначим как S_W сумму N_W расстояний внутри каждого кластера.
- S_{min} – сумма наименьших расстояний N_W между всеми парами точек во всем наборе данных. То есть из N_T возможных расстояний рассмотрим сумму наименьших N_W расстояний.
- S_{max} – сумма наибольших N_W расстояний между всеми парами точек во всем наборе данных. Аналогично: из N_T возможных расстояний рассмотрим сумму наибольших N_W расстояний.

Тогда индекс выражается формулой [**?**]:

$$C_c = \frac{S_W - S_{min}}{S_{max} - S_{min}} \tag{3.3}$$

3.2 Dunn индекс

Обозначим через d_{min} минимальное расстояние между точками различных кластеров, а через d_{max} - наибольшее расстояние внутри кластера. Расстояние между кластерами I_0 и I_1 измеряется расстоянием между ближайшими точками:

$$d_{min} = \min_{i \in I_0, j \in I_1} ||M_i^0 - M_j^1||_2 \tag{3.4}$$

$$d_{max}^k = \max_{i,j \in I_k} ||M_i^k - M_j^k||_2, k \in \{0,1\} \tag{3.5}$$

$$d_{max} = \max_{k \in \{0,1\}} d_{max}^k \tag{3.6}$$

Тогда индекс выражается формулой:

$$C_{Dunn} = \frac{d_{min}}{d_{max}} \tag{3.7}$$

3.3 Trace_W индекс

В тех же обозначениях, что использовались для построения индекса C

$$C_{Trace_W} = trace(WG) = \sum_{k=0}^{1} WG^k = \sum_{k=0}^{k=1} \sum_{i \in I_k} ||M_i^k - G^k|| \tag{3.8}$$

Где G^k - центр масс k-ого кластера.

4 Предлагаемые алгоритмы

Абстрагируемся от задачи классификации и рассмотрим задачу разделения классов в два макрокласса. Переформулируем проблему в локальную задачу кластеризации.

Локальная задача кластеризации Рассмотрим произвольный узел дерева, в котором содержиться m классов. Объединим объекты из одной группы в объект-класс. Для определенности будем рассматривать группу i. Присвоим объекту-классу признаки $\{b|b = med(a|a \in O_i)\}$. Таким образом будет построено m объектов-классов. Задача состоит в том, чтобы разбить их на 2 макрокласса, используюя индексы кластеризации, описанные выше.

Пользуясь переформулировкой выше, были построены новые методы многоклассовой классификаци с использованием бинарной кластеризации на каждом узле дерева.

4.1 Расстояние

Использование индексов предполагает, что в пространстве признаком введена мера расстояния. Выбор меры и весов очень важен для описываемых методов, так как от этих процедур зависят состав формируемых кластеров. Алгоритмы, предложенные ниже, предполагают, что пользователь сможет сам выбирать меру или комбинацию мер и передавать её в функцию, которая будет применена для создания дерева. В опытах, приведенных ниже использовались:

4.1.1 Взвешенное Евклидово расстояние

$$P = \sqrt{\sum_{i=1}^{N} w_i (A_i - B_i)^2} \tag{4.1}$$

Где A,B – объекты w_i – вес или важность признака. Если этот параметр положить единице, то получиться обычное евклидово расстояние. Применение евклидова расстояния оправдано в следующих случаях:

- Признаки однородны по физическому смыслу.
- Признаковое пространство совпадает или очень похоже на геометрическое пространство.

4.1.2 Процент несогласия

Эта мера расстояния хорошо показывает себя в тех случаях, когда свойства объекта являются категориальными неупорядоченными:

$$P = VALUE(A_i = B_i) \tag{4.2}$$

Где A,B - объекты Например, первый признак объекта – цвет, второй – название вида. Представим значения свойств объекта в виде вектора значений. Первый

вектор – (коричневый, конь), второй вектор – (коричневый, курица). Процент несогласия равен 1/2. Эти вектора различаются на 50%.

4.2 Алгоритм NDCIC

Алгоритм NDCIC основывается на C индексе.

Описание алгоритма

- Создайте корневой узел для дерева.
- Создайте множество объект-классов:
  - Объедините объекты из одного класса.
  - Для каждого признака посчитайте медиану по группе.
- Поместите все объект-классы в корневой узел.
- Если объект-класс C содержит только один класс, то создайте листовой узел.
- В противном случае, если он содержит k классов, разделите C на два подмножества S_1 и S_2 следующим образом:
  - Постройте множество всевозможных подмножеств.
  - Для каждого варианта разбития S_{1j} и S_{2j} из множества:
    * Посчитайте среднюю S_W сумму расстояний между объект-классами для S_{1j} и S_{2j}.
    * Посчитайте S_{min} сумму наименьших N_W расстояний между всеми N_T парами точек во всем наборе данных.
    * Посчитайте S_{max} сумму наибольших N_W расстояний между всеми парами точек во всем наборе данных.

    * Вычислите индекс $C = \dfrac{S_W - S_{min}}{S_{max} - S_{min}}$

  – Найдите максимальный C индекс. Пусть для определенности он достигается на вариантах разбития S_{1s} и S_{2s}.

  – Присвойте $S_{1j} = S_{1s}$ и $S_{2j} = S_{2s}$

- Обучите бинарный классификатор на каждом узле построенного бинарного дерева.

- Процесс классификации:

  – Объект перемещается по дереву начиная с корня по двоичным классификаторам до тех пор, пока не будет достигнут лист.

  – Класс, связанный с листом дерева, присваивается экземпляру.

.

4.3 Алгоритм NDCID

Алгоритм NDCID основывается на Dunn индексе.

Описание алгоритма

- Создайте корневой узел для дерева.

- Создайте множество объект-классов:

  – Объедините объекты из одного класса.

  – Для каждого признака посчитайте медиану по группе.

- Поместите все объект-классы в корневой узел.

- Если объект-класс C содержит только один класс, то создайте листовой узел.

- В противном случае, если он содержит k классов, разделите C на два подмножества S_1 и S_2 следующим образом:
  - Постройте множество всевозможных подмножеств.
  - Для каждого варианта разбития S_{0j} и S_{1j} из множества:
    * Посчитайте
    $$d_{min} = \min_{i \in S_{0j}, j \in S_{1j}} ||M_i^0 - M_j^1||_2$$
    * Посчитайте
    $$d_{max}^k = \max_{i,j \in S_{kj}} ||M_i^k - M_j^k||_2, k \in \{0, 1\}$$
    * Посчитайте
    $$d_{max} = \max_{k \in \{0,1\}} d_{max}^k$$
    * Вычислите индекс
    $$C_{Dunn} = \frac{d_{min}}{d_{max}}$$
  - Найдите минимальный C индекс. Пусть для определенности он достигается на вариантах разбития S_{0s} и S_{1s}.
  - Присвойте $S_{0j} = S_{0s}$ и $S_{1j} = S_{1s}$
- Обучите бинарный классификатор на каждом узле построенного бинарного дерева.
- Процесс классификации:
  - Объект перемещается по дереву начиная с корня по двоичным классификаторам до тех пор, пока не будет достигнут лист.
  - Класс, связанный с листом дерева, присваивается экземпляру

.

4.4 Проблемы методов NDCID, NDCIC

Из-за того, что индексы Dunn и C плохо интерпретируемы, не получилось придумать способа как посчитать их каким-либо иным способом, как напрямую. То есть поиск минимального индекса для всех возможных разбиений классов на 2 кластера. Количество разбиений выражается формулой:

$$2^k - k \tag{4.3}$$

Асимптотика подсчета метода NDCID и NDCIC:

$$\sum_{k=l+l'}(\sum_{i}^{l}\sum_{k=i}^{l}O(1)+\sum_{i}^{l}{}'\sum_{k=i}^{l}{}'O(1)) = \sum_{k=l+l'}(\sum_{i=1}^{l}O(l-i)+\sum_{i=1}^{l'}O(l'-i)) =$$
$$\sum_{k=l+l'}(\sum_{1}^{l-1}O(p)+\sum_{1}^{l'-1}O(p)) = \sum_{k=l+l'}(O(l^2)+O(l'^2)) = \sum_{l}(O(l^2)+O((k-l)^2)) =$$
$$O(k^2 \cdot 2^k) \tag{4.4}$$

Поэтому алгоритмы работают медленнее своих аналогов, хотя и дают хорошее, часто лучшее, качество:

Таблица 2: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на коллекции с 4 классами

Название	**Кол-во данных**	**Точность**	**Скорость**
NDCID	10000	0.535	750 ms
NDCIC	10000	0.5257	783 ms
NDR	10000	0.5178	1250 ms
ECBND	10000	0.5149	1270 ms
NDWRPS	10000	0.5204	1030 ms
NDCR	10000	0.5332	1480 ms
NDC	10000	0.5257	827 ms

Из таблиц выше видно, что несмотря на то, что метод может давать наилуч-

Таблица 3: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на коллекции с 15 классами

Название	**Кол-во данных**	**Точность**	**Скорость**
NDCID	10000	0.22458	5min 4s
NDCIC	10000	0.21562	6min 55s
NDR	10000	0.15523	1min
ECBND	10000	0.18509	55.8s
NDWRPS	10000	0.16523	19s
NDCR	10000	0.20092	3min 33s
NDC	10000	0.18852	33.3s

шую точность, время выполнения алгоритма быстро увеличивается с ростом количества классов. Поэтому был рассмотрен еще один индекс Trace_W для минимизации которого уже предложен алгоритм кластеризации k-means [10].

4.5 Алгоритм NDCIW

Алгоритм NDCIW основывается на Trace_W индексе.

4.5.1 Описание алгоритма

- Создайте корневой узел для дерева.
- Создайте множество объект-классов:
  - Объедините объекты из одного класса.
  - Для каждого признака посчитайте медиану по группе.
- Поместите все объект-классы в корневой узел.
- Если объект-класс C содержит только один класс, то создайте листовой узел.
- В противном случае, если он содержит k классов, разделите C на два подмножества S_1 и S_2 используя алгоритм кластеризации k-means.

- Обучите бинарный классификатор на каждом узле построенного бинарного дерева.
- Процесс классификации:
  - Объект перемещается по дереву начиная с корня по двоичным классификаторам до тех пор, пока не будет достигнут лист.
  - Класс, связанный с листом дерева, присваивается экземпляру.

.

4.5.2 k-means

K-means – это алгоритм кластеризации, который разбивает исходное множество элементов признакового пространства на заранее известное число кластеров k. При этом, на каждой итерации перевычисляется центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике. Алгоритм завершается когда на какой-то итерации не происходит изменения внутрикластерного расстояния. Алгоритм сходиться, так как количество возможных разбиений конечного множества конечно, при этом на каждой итерации суммарное квадратичное отклонение (3.8) уменьшается, поэтому зацикливание невозможно.

При этом сложность алгоитма для разбиения на 2 кластера оценивается как O(nd), где n - количество обьектов, а d - количество признаков [11]. Таким образом, скорость алгоритма станет линейной по числу классов, но при этом, так как k-mean использует (3.8) в качестве функции ошибки, то точной минимизации индекса не получается.

5 Сравнение алгоритмов

Прозведено сравнения методов NDC, NDCR, [7] NDR [3], ECBND, EDBND [4], NDWRPS (Nested Dichotomies with Random-pair-selection) с методами NDCID, NDCIC на искуственных и открытых наборах данных (Таб. 4) с использованием базовых классификаторов: логистическая регрессия и решающие деревья.

Таблица 4: Наборы данных используемых в эксперементе

Название	**Кол-во данных**	**Кол-во классов**	**Кол-во признаков**
RD_digits	5620	10	64
RD_CIFAR10	10000	10	3072
uci-cardio	2126	10	21
rd_breast	106	6	9

В открытом доступе была найдена реализация для методов ECBND, EDBND, NDWRPS [12], остальные методы NDC, NDCR, NDCID, NDCIC были реализованы самомстоятельно.

Для борьбы с переобучением была использована улучшенная схема оценивания качества алгоритмов. Данные были разбиты на 2 части. Одна часть под кросс-валидацию, другая под контрольную выборку. При таком подходе каждый из алгоритмов в зависимости от гиперпараметров обучается на обучающей выборке, а его качество измеряется на валидационной с помощью кросс-валидации. Алгоритм с наилучшим качеством будет проверен на тестовой выборке, чтобы исключить переобучение и проверить алгоритм на адекватность. [13].

5.1 Результаты с использованием логистической регрессии как базового классификатора

На датасете RD_digits предложенный алгоритм NDCID показал наилучшие результаты. (Таб. 5)

Таблица 5: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете RD_digits с использованием базового классификатора линейной регрессии

Название	**Точность**	**Скорость**
NDCID	**0.9249**	5.74 s
NDCIC	0.9026	2.45 s
NDCIW	0.9132	1.56 s
NDR	0.8820	1.02 s
ECBND	0.8492	1.07 s
NDWRPS	0.9087	1.04 s
NDCR	0.9132	10.2 s
NDC	0.8987	836 ms

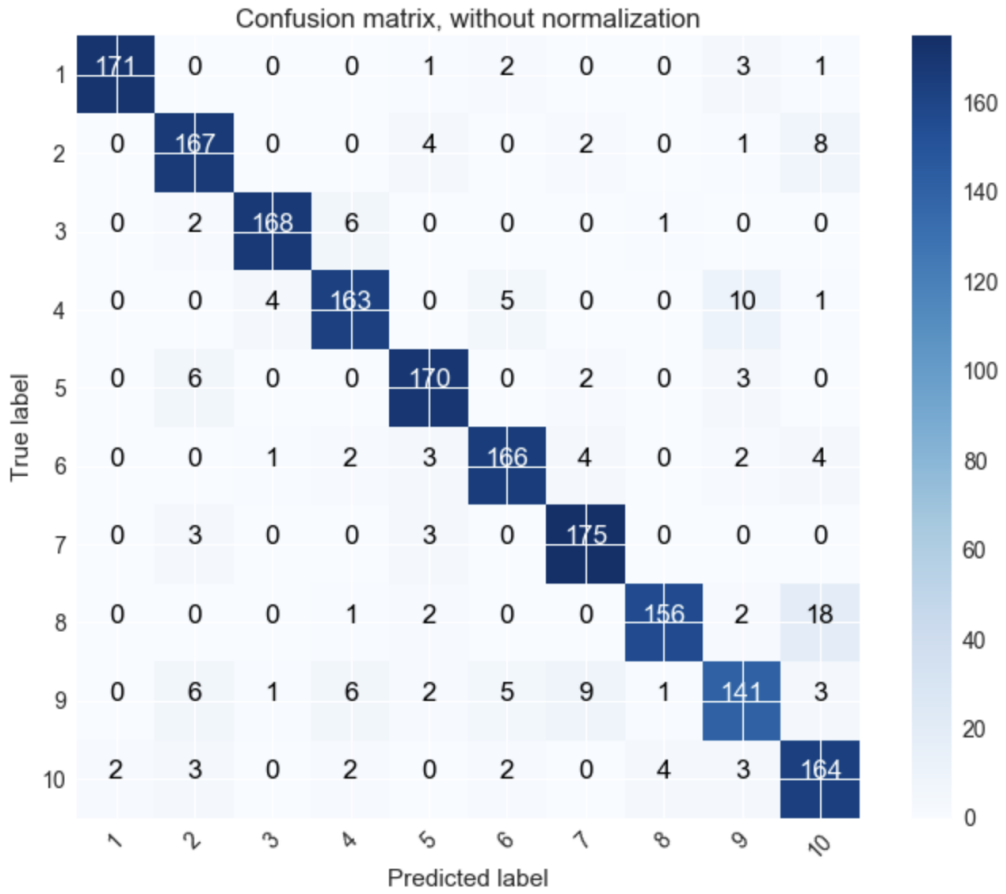


Рис. 4: Матрица ошибок для метода NDCID на датасете RD_digits с использованием базового классификатора линейной регрессии

На наборе данных uci-cardiotocography алгоритм NDCID показал второй по качеству результат, уступая лишь NDCR, но по скорости обогнал его более чем в 2 раза (Таб. 6). Аналогично и для данных из набора RD_Breast (Таб. 7).

Таблица 6: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете uci-cardiotocography с использованием базового классификатора линейной регрессии

Название	**Точность**	**Скорость**
NDCID	**0.9619**	1.72 s
NDCIC	0.9398	1.19 s
NDCIW	0.9417	664 ms
NDR	0.9430	481 ms
ECBND	0.9610	407 ms
NDCR	0.9638	3.7 s
NDC	0.9384	492 ms

Таблица 7: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете RD_Breast с использованием базового классификатора линейной регрессии

Название	**Точность**	**Скорость**
NDCID	**0.3207**	**91.1 ms**
NDCIC	0.2358	64 ms
NDCIW	0.2736	156 ms
NDR	0.3018	55.5 ms
NDWRPS	0.3396	67.3 ms
ECBND	0.2264	51.5 ms
NDCR	**0.3491**	**416 ms**
NDC	0.2830	54.9 ms

5.2 Результаты с использованием решающих деревьев

Как видно из таблиц 8 9 10 наилучшее качество классификации достигается на методах NDCIC и NDCR. То есть на методах, основанных на кластеризации.

5.3 Результаты с использованием больших наборов данных

В предыдущей главе были рассмотрены датасеты небольшого объема. В данном разделе будет рассматриваться набор данных CIFAR-10. Набор данных состоит из 60000 32x32 цветных изображений в 10 классах с 6000 изображениями на класс. Содержит 50000 учебных изображений и 10000 тестовых изображений. Классы: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck.

Таблица 8: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете RD_Breast с использованием базового классификатора 'решающие деревья'

Название	**Точность**	**Скорость**
NDCID	0.2642	70.4 ms
NDCIC	**0.3585**	61.2 ms
NDCIW	0.3208	273 ms
NDR	0.2453	41.8 ms
ECBND	0.3301	41.6 ms
NDWRPS	0.2453	58 ms
NDCR	0.3584	10.2 s
NDC	0.2830	42.9 ms

Таблица 9: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете RD_digits с использованием базового классификатора 'решающие деревья'

Название	**Точность**	**Скорость**
NDCID	0.8414	6.39 s
NDCIC	0.8392	2.47 s
NDCIW	0.8397	2.08 s
NDR	0.8057	751 ms
NDWRPS	0.8269	805 ms
ECBND	0.7818	739 ms
NDCR	**0.8953**	7.11 s
NDC	0.8375	1.71 s

Таблица 10: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете RD_Cardio с использованием базового классификатора 'решающие деревья'

Название	**Точность**	**Скорость**
NDCID	0.9817	1.6 s
NDCIC	0.9864	1.02 s
NDCIW	0.9854	443 ms
NDR	0.982	194 ms
NDWRPS	0.8269	805 ms
ECBND	0.9811	185 ms
NDCR	**0.9929**	1.4 s
NDC	0.9864	169 ms

Наилучшие результаты показал предложенный метод NDCID по соотношению время-качество.

Таблица 11: Сравнение алгоритмов NDCID, NDCIC, NDR, ECBND, NDWRPS, NDCR, NDC на датасете CIFAR-10 с использованием базового классификатора 'решающие деревья'

Название	**Точность**	**Скорость**
NDCID	0.2447	21min 5s
NDCIC	**0.2489**	8min 29s
NDCIW	0.2441	443 ms
NDR	0.2337	16min 23s
NDWRPS	0.2471	18min 24s
ECBND	0.2418	13min 21s
NDCR	0.3307	3h 3min 29s
NDC	0.2483	18min 36s

6 Заключение

В данной работе рассматривалась проблема классификации и сравнивались предложенные методы, базируемые на идее построения дерева бинарных классификаторов – nested dichotomies. Это методы NDC, NDCR, ECBND, NDWRPS, NDR. Также были предложены новые методы многоклассовой классификации NDCIC, NCID, NDCIW нацеленные на улучшение качества предсказаний. Методы NDC, NDCR, реализация которых не была найдена в открытых репозиториях, были запрограммированы и выложены в открытый доступ. На основе приведенных выше эксперементов можно сделать вывод, что лучше всего по качеству предсказаний себя показывают методы, основанные на кластеризации: NDC, NDCR, NDCID, NDCIC, NDCIW. При этом в большинстве случаев лучше всего по метрике accuracy показал себя метод NDCID. Стоит однако заметить, что NDC, NDCR, NDCID, NDCIC работают дольше своих аналогов. Это происходит из-за того, что на каждмо узле производятся дополнительные вычисления. Методы NDCID и NDCIC ищут наилучшее разделение с помощью полного перебора по всем возможным разделениям, поэтому они менее всего эффективны по скорости. Методы NDC, NDCR не используют полный перебор, упростив способ вычисления наилучшего разбиения, поэтому, в целом, работают быстрее. Однако на больших данных метод NDCR работал более чем в 10 раз медленнее, чем аналоги. При этом для методов NDC, NDCID, NDCIC время работы не сильно зависит от количества данных в коллекции, но сильно зависит от количества классов. Когда количество классов больше 15, а целью задачи является улучшение качества предсказаний, лучше использовать метод NDCIW.

Список литературы

[1] Erin Allwein Robert Shapire Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. // Journal of Machine Learning Research. — 2000. — P. 113–141.

[2] Chih-Wei Hsu Chih-Jen Lin. A comparison of methods for multiclass support vector machines. In IEEE TRANSACTIONS ON NEURAL NET- WORKS // IEEE TRANSACTIONS ON NEURAL NETWORKS. — March 2002. — Vol. 13. — P. 415–425.

[3] Frank E. Kramer S. Ensembles of nested dichotomies for multi-class problems. // Proceedings of the twenty-first international conference on Machine learning. — 2004. — P. 39.

[4] Dong L. Frank E.-Kramer S. Ensembles of balanced nested dichotomies for multi-class problems. // Knowledge Discovery in Databases. — PKDD 2005. — P. 84–95.

[5] Rodrguez J.J. Garca-Osorio C. Maudes J. Forests of Nested Dichotomies. // Pattern Recognition Letters. — Vol. 31. — P. 125–132.

[6] Rodrguez J.J. Garca-Osorio C. Maudes J. Forests of Nested Dichotomies. // Proceedings of Joint European Conference on Machine Learning and Knowledge Discovery in Databases. — 2016.

[7] Miriam Mónica Duarte-Villaseñor Jesús Ariel Carrasco-Ochoa José Francisco Martínez-Trinidad Marisol Flores-Garrido. Nested Dichotomies Based on Clustering // Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. — 2012.

[8] Воронцов К.В. Линейные методы классификации и регрессии: метод стохастического градиента. — http://www.machinelearning.ru/wiki/images/5/53/Voron-ML-Lin-SG.pdf.

[9] Воронцов К.В. Лекции по логическим алгоритмам классификации. — http://www.ccas.ru/voron/download/LogicAlgs.pdf.

[10] R.C.Dubes A.K.Jain. Forests of Nested Dichotomies. // Algorithmsnfor Clustering Data. — 1988.

[11] R.C.Dubes A.K.Jain. An Efficient K-Means Clustering Algorithm // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 07 August 2002.

[12] Leathart Tim. — https://github.com/timleathart/Random-Pair-Nested-Dichotomies.

[13] Payam Refaeilzadeh Lei Tang Huan Liu // Encyclopedia of Database Systems. — 2009. — P. 532–538.