О полных регрессионных решающих деревьях*

И. Е. Генрихов, Е. В. Дюкова, В. И. Журавлёв
ingvar1485@rambler.ru, edjukova@mail.ru, vadim091294@gmail.com
[1]ООО «Мобайл парк ИТ», г. Химки, ул. Панфилова, 21/1
[2]ФИЦ «Информатика и управление» РАН, г. Москва, ул. Вавилова, 44/2
[3]МГУ им. М. В. Ломоносова, г. Москва, Ленинские горы, 1

Рассматривается одна из центральных задач машинного обучения — задача восстановления регрессии. Предлагается качественно новая модель регрессионного решающего дерева (РРД), базирующаяся на понятии полного решающего дерева (ПРД). Ранее аналогичная конструкция решающего дерева (РД) была успешно апробирована на задаче классификации по прецедентам, которая по постановке близка к рассматриваемой задаче. Приведены результаты тестирования построенной модели полного РРД (ПРРД) на реальных данных.

Ключевые слова: *задача восстановления регрессии; регрессионные деревья; полное решающее дерево*

DOI: 10.21469/22233792.2.1.09

1 Введение

Одной из основных задач машинного обучения является задача обучения по прецедентам. Рассматривается следующая постановка этой задачи.

Исследуется множество объектов M. Объекты из M описываются системой признаков $\{x_1, \ldots, x_n\}$. Каждый объект S из M представим вектором длины n, в котором j-я координата равна значению признака x_j для объекта S. Задано некоторое числовое множество «ответов» Y и дана выборка объектов $T = \{S_1, ..., S_m\}$ из M такая, что для каждого объекта $S_i \in T$ известен «ответ» $y_i, y_i \in Y$. Объекты из T называются прецедентами или обучающими объектами. Требуется по выборке T построить алгоритм $A_T : M \to Y$, ставящий в соответствие каждому объекту S из M значение y из Y.

Актуальность рассматриваемой задачи заключается в том, что она возникает в целом ряде прикладных областей, таких как биология, геология, медицина, экономика, техника, банковская деятельность и др.

Выделяют два основных типа задач обучения по прецедентам.

1. Задача классификации (classification). В этом случае «ответ» y для объекта S из M называется меткой класса. Возможны следующие варианты:
   - $Y = \{-1; +1\}$ — классификация на 2 класса;
   - $Y = \{1, ..., N\}$ — классификация с N классами.
2. Задача восстановления регрессии (regression). В данном случае $Y = \mathbb{R}$ и «ответ» y для объекта S из M называется значением целевой переменной.

Одним из известных инструментов для решения задач обучения по прецедентам являются деревья решений.

Процедура построения классического РД представляет собой итерационный процесс. Как правило, для построения очередной вершины дерева выбирается признак, наилучшим

*Работа выполнена при финансовой поддержке РФФИ, проект № 16-01-00445.

образом удовлетворяющий некоторому критерию ветвления. По значениям этого признака и осуществляется ветвление, далее указанная процедура повторяется для каждой из ветвей. Однако если при построении дерева несколько признаков удовлетворяют критерию ветвления в равной или почти равной мере, то выбирается один из них (фактически случайным образом). При этом в зависимости от выбранного признака построенные деревья могут существенно отличаться как по составу используемых признаков, так и по своим распознающим качествам. Указанного недостатка лишена модель ПРД [1, 2]. В ПРД на каждой итерации строится так называемая полная вершина, которой соответствует набор признаков $\{x_{j_1}, \ldots, x_{j_q}\}, q \leqslant n$, где каждый признак удовлетворяет критерию ветвления. Затем для каждого признака $x_{j_i}, i \in \{1, ..., q\}$, строится «простая» внутренняя вершина, из которой осуществляется ветвление. По сравнению с классической конструкцией конструкция ПРД позволяет более полно использовать имеющуюся информацию, при этом описание распознаваемого объекта может порождаться не одной ветвью, как в классическом дереве, а несколькими ветвями. Каждая такая ветвь участвует в процедуре голосования (является голосующей).

Модель классического РД используется для решения обоих типов задач обучения по прецедентам. Модель ПРД разработана сравнительно недавно для решения задач классификации.

В настоящей работе рассматривается задача восстановления регрессии.

Одним из первых алгоритмов, использующих РРД, является алгоритм CART (classification and regression trees). Этот алгоритм строит бинарное РРД с критерием ветвления, основанным на вычислении статистик [3]. Похожую на CART конструкцию имеет алгоритм M5P. Алгоритм M5P, так же как и алгоритм CART, выполняет построение бинарных РД [4]. Более сложную конструкцию имеют алгоритмы классификации и восстановления регрессии Random Forest [4], REPTree [5] и Decision Stump [6]. Алгоритм Decision Stump базируется на построении k-арных РД, остальные алгоритмы строят бинарные РД.

Основной целью данной работы является построение и исследование ПРРД для задач с целочисленными данными.

В работе построены и протестированы алгоритмы NBRTree (nonbinary regression tree) и NBFRTree (nonbinary full regression tree), строящие k-арные регрессионные деревья, где k – максимальное число ребер, выходящих из простых вершин дерева. Алгоритм NBRTree строит классическое k-арное РРД. Алгоритм NBFRTree строит k-арное ПРРД. В обоих алгоритмах используется критерий ветвления, являющийся модификацией критерия ветвления алгоритма CART на случай k-арного дерева [3].

Проведено тестирование алгоритмов NBRTree и NBFRTree на реальных задачах. Показано, что на большинстве рассмотренных в работе задач алгоритм NBFRTree работает лучше других алгоритмов восстановления регрессии, участвовавших в тестировании, среди которых алгоритмы Random Forest, REPTree, M5P, CART, Decision Stump и NBRTree.

2 Основные понятия

Рассмотрим основные понятия, используемые при построении РРД, на примере бинарного РРД (БРРД).

Обозначим через $\breve{T}$ и $X(\breve{T}) \subseteq \{x_1, ..., x_n\}$ рассматриваемые на текущей итерации (шаге) построения РРД подмножество обучающих объектов и подмножество признаков соответственно.

На первом шаге $\breve{T} = T, X(\breve{T}) = \{x_1, ..., x_n\}$. На текущем шаге построения дерева для каждого признака x из $X(\breve{T})$ и каждого значения a признака x проводится разбиение $\breve{T}$

на две подвыборки (подвыборку $\breve{T}_R(x,a)$, для объектов которой выполняется неравенство $x \geqslant a$, и подвыборку $\breve{T}_L(x,a)$, для объектов которой выполняется неравенство $x < a$) и вычисляется оценка качества этого разбиения.

`Определение 1.` Оптимальным разбиением называется разбиение с наилучшей оценкой качества.

`Определение 2.` Признак удовлетворяет критерию ветвления, если оптимальное разбиение для этого признака имеет максимальную оценку качества разбиения среди оптимальных разбиений для всех других признаков.

Среди всех признаков, удовлетворяющих критерию ветвления, выбирается только один признак.

Различные алгоритмы БРРД отличаются критерием ветвления, а также правилом останова ветвления. Сложность построения БРРД очень велика при большом числе признаков, особенно если признаки многозначны.

На рис. 1 приведен пример ветвления из вершины x в БРРД. В этом дереве $x \in X(\breve{T}), a \in \{0,1,\ldots,k-1\}$, a — значение x.

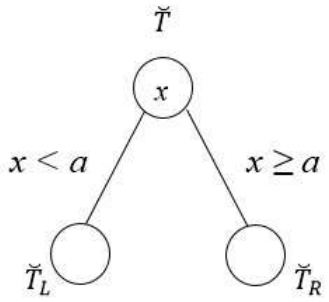

Рис. 1 Пример ветвления из x в БРРД

Алгоритм CART строит БРРД и при выборе оптимального разбиения использует статистический подход к оценке качества разбиения (критерий ветвления, основанный на вычислении статистик) [3]. Опишем этот критерий.

Пусть $\breve{T} = \{S_{i_1},\ldots,S_{i_u}\}, \breve{T}_R(x,a) = \{S_1^R,...,S_q^R\}, \breve{T}_L(x,a) = \{S_1^L,...,S_p^L\}$. При данном разбиении в правое и левое поддеревья попадает q и p объектов соответственно.

Пусть далее y_i^L и y_i^R — значения целевых переменных для объектов $S_i^L, i = 1,\ldots,q$, и $S_j^R, j = 1,...,p$, соответственно. Введем обозначения:

$$\bar{y}_{\breve{T}} = \frac{1}{u}\sum_{t=1}^{u} y_{i_t}\,;$$

$$V = \frac{1}{u}\sum_{t=1}^{u}(y_{i_t}^2) - \Big[\frac{1}{u}\sum_{t=1}^{u}(y_{i_t})\Big]^2\,;$$

$$\mathrm{SE}\,(x) = \frac{1}{u}\Big\{\sum_{i=1}^{p}(y_i^L - \bar{y}_{\breve{T}_L})^2 + \sum_{j=1}^{q}(y_j^R - \bar{y}_{\breve{T}_R})^2\Big\}\,;$$

$$C(x) = V - \mathrm{SE}\,(x)$$

Оптимальным считается разбиение с максимальным значением величины $C(x)$.

Построение очередной ветви в алгоритме CART прекращается, если величина $C(x)$ не превосходит наперед заданного числа δ.

Похожую на CART конструкцию имеет алгоритм M5P. Алгоритм M5P, так же как и алгоритм CART, выполняет построение бинарных РД, но использует энтропийный критерий ветвления [7].

Более сложную конструкцию имеют алгоритмы классификации и восстановления регрессии Random Forest [4], REPTree [5] и Decision Stump [6]. Алгоритм Decision Stump базируется на построении k-арных РД, остальные алгоритмы строят бинарные РД. Среди перечисленных алгоритмов наиболее используемым является алгоритм Random Forest.

Random Forest — алгоритм машинного обучения, заключающийся в использовании комитета (ансамбля) РД (предложен Л. Брейманом и А. Катлер в 2001 г.). В алгоритме Random Forest используется энтропийный критерий ветвления и процедура «бэггинг». Процедура бэггинга над РД заключается в использовании композиции РД, каждое из которых строится независимо. Для построения очередного дерева композиции случайным образом выбирается (с возвращением) некоторое подмножество обучающих объектов из исходной выборки. Результат распознавания определяется путем усреднения значений целевой переменной по всем построенным РД. Таким образом, деревья компенсируют ошибки друг друга.

3 Алгоритмы восстановления регрессии NBRTree и NBFRTree

3.1 Построение алгоритмов NBRTree и NBFRTree

Алгоритм NBRTree строит классическое РРД. Главная особенность алгоритма NBRTree — это его k-арная структура. Ветвление по выбранному признаку x разбивает обучающие объекты на k подвыборок, где k — число различных значений признака.

Рассмотрим более подробно схему ветвления из вершины $x, x \in X(\breve{T})$, в алгоритме NBRTree.

Не ограничивая общности, будем считать, что признак x имеет значения из $\{0, 1, \ldots, k-1\}, k \geqslant 2$. В этом случае при построении дерева решений из вершины x выходят k дуг, помеченные числами из $\{0, 1, \ldots, k-1\}$. Пусть σ — метка одной из дуг, выходящих из вершины $x, \sigma \in \{0, 1, \ldots, k-1\}$. Для формирования нового текущего подмножества объектов и нового текущего множества признаков удаляются те объекты из $\breve{T}$, для которых значение признака x не равно σ, а также из множества признаков удаляется сам признак x. Для улучшения качества распознавания при построении ветви используется правило останова, которое описано в конце данного параграфа.

Положим

$$x^{\sigma} = \begin{cases} 1, & \text{если } x = \sigma\,; \\ 0, & \text{если } x \neq \sigma. \end{cases}$$

Пусть v — висячая вершина, порожденная ветвью дерева с внутренними вершинами $x_{j_1}, \ldots, x_{j_r}$ и пусть дуга, выходящая из вершины $x_{j_i}, i \in \{1, \ldots, r\}$, имеет метку σ. Пусть далее $\breve{T}(v)$ — текущее множество объектов, которые попали в вершину v. Вершине v ставится в соответствие пара $(B, w(v))$, где $w(v)$ равно среднему значению целевой переменной по всем объектам из $\breve{T}(v)$, а B — элементарная конъюнкция вида $x_{j_1}^{\sigma_1}, \ldots, x_{j_r}^{\sigma_r}$. Интервал истинности элементарной конъюнкции B обозначим через N_B.

Пусть S — распознаваемый объект. Для каждой висячей вершины $(B, w(v))$ выполняется проверка принадлежности описания распознаваемого объекта S интервалу истинности N_B. Если описание S принадлежит N_B, то объекту S ставим в соответствие значение

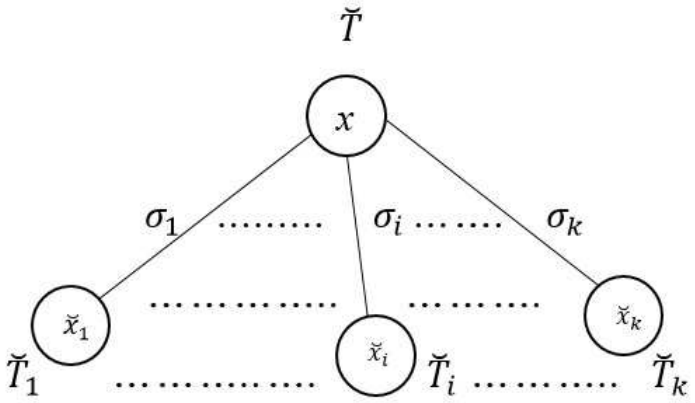

Рис. 2 Ветвление из вершины x в алгоритме NBRTree

целевой переменной $w(v)$. По построению описание может попасть только в одну из висячих вершин.

На рис. 2 показано ветвление из вершины x в алгоритме NBRTree для $\breve{T} = \breve{T}_1 \cup \breve{T}_2 \cup ... \cup \breve{T}_k$, где k — множество различных значений признака x.

В алгоритме NBFRTree используется идея ПРД, т. е. при возникновении ситуации, когда два или более признака удовлетворяют критерию ветвления в равной или почти равной мере, то ветвление проводится по каждому из этих признаков независимо.

Процедура распознавания объекта выполняется следующим образом. Пусть $V = v_1, \ldots, v_l$ — множество висячих вершин построенного дерева с соответствующими парами $(B_i, w(v_i)), i = 1, 2, ..., l, l \geqslant 1$. Для каждой висячей вершины v_i осуществляется проверка принадлежности описания объекта S интервалу истинности N_{B_i}.

Положим

$$I_{B_i} = \begin{cases} 1, & \text{если описание объекта } S \in N_{B_i}; \\ 0 & \text{в противном случае.} \end{cases}$$

Объекту S ставится в соответствие значение целевой переменной

$$W = \frac{\sum_{i=1}^{l} w(v_i) I_{B_i}}{\sum_{i=1}^{l} I_{B_i}}.$$

На рис. 3 показано ветвление из полной вершины $\{x_{j_1}, \ldots, x_{j_r}\}$ в алгоритме NBFRTree. Ветвление из простых вершин $x_{j_1}, ..., x_{j_r}$ производится как в алгоритме NBRTree.

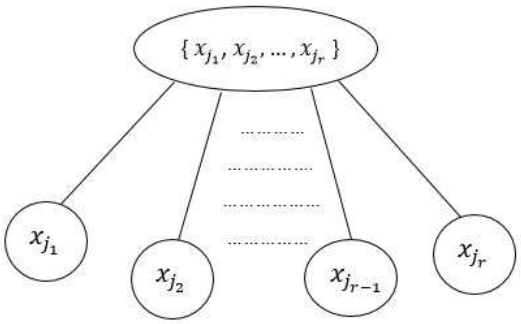

Рис. 3 Ветвление из полной вершины $\{x_{j_1}, \ldots, x_{j_r}\}$ в алгоритме NBFRTree

Опишем критерий ветвления, используемый в алгоритмах NBRTree и NBFRTree.

Пусть $\breve{T}_i = S^i_1, \ldots, S^i_{u_i}, y^i_t$ — значение целевой переменной обучающего объекта $S^i_t, t \in \{1, 2, \ldots, u_i\}$, и пусть рассматриваемый признак x принимает k значений. Обучающая выборка $\breve{T}_i$ разбивается по этому признаку на k подвыборок $\breve{T}_{i_1}, \ldots, \breve{T}_{i_k}$. Вычисляются величины

$$\mathrm{SE}\,(x,k) = \frac{1}{u_i}\Big\{ \sum_{S^i_t \in \breve{T}_{i_1}} (y^i_t - \bar{y}_{\breve{T}_{i_1}})^2 + \cdots + \sum_{S^i_t \in \breve{T}_{i_k}} (y^i_t - \bar{y}_{\breve{T}_{i_k}})^2 \Big\};$$

$$C(k,x) = V - \mathrm{SE}\,(x,k)\ .$$

При $k = 2$ описанный критерий совпадает с критерием ветвления алгоритма CART (см. разд. 2).

В алгоритме NBRTree для ветвления выбирается один признак, для которого $C(k,x) = \max\limits_{x \in X(\breve{T})} C(k,x)$.

В алгоритме NBFRTree признаки для ветвления выбираются иначе. Пусть $C_{\min} = \min\limits_{x \in X(\breve{T})} C(k,x)$ и $C_{\max} = \max\limits_{x \in X(\breve{T})} C(k,x)$. Сначала для каждого признака $x \in X(\breve{T}_i)$ вычисляется величина $C(k,x) = V - \mathrm{SE}\,(x,k)$. Далее значение $C(k,x)$ нормируется и вычисляется

$$C^*(k,x) = \frac{C(k,x) - C_{\min}}{C_{\max} - C_{\min}}\,.$$

Для построения полной вершины выбираются те признаки из $X(\breve{T}_i)$, для которых $0{,}75 \leqslant C^*(k,x) \leqslant 1$. В случае, когда $C_{\max} = C_{\min}$, разбиение производится по всем признакам из $X(\breve{T}_i)$.

Построение ветви прекращается, если разность между минимальной и максимальной целевыми переменными в данной вершине не превосходит наперед заданного ε (параметр останова).

4 Тестирование алгоритмов NBRTree и NBFRTree

Алгоритмы были протестированы на 18 реальных задачах из ресурса UCI [8]. Список задач, на которых производилось тестирование алгоритмов: Data1 — Servo; Data2 — Computer Hardware; Data3 — Yacht Hydrodynamics; Data4 — Concrete Slump Test; Data5 — Fertility; Data6 — Breast Cancer Wisconsin breast-cancer-wisconsin; Data7 — Concrete Compressive Strength; Data8 — Housing; Data9 — Airfoil Self-Noise; Data10 — Combined Cycle Power Plant; Data11 — Forest Fires; Data12 — White Wine Quality; Data13 — Red Wine Quality; Data14 — Student Performance; Data15 — Geographical Original of Music Data Set Geographical Original of Music Data Set latitude; Data16 — Geographical Original of Music Data Set longitude; Data17 — Breast Cancer Wisconsin wdbc; Data18 — Breast Cancer Wisconsin wpbc.

В задачах, в которых присутствовали признаки, принимающие вещественнозначные значения, была применена процедура перекодирования вещественнозначных значений признака в целочисленные. Производилась она следующим образом.

Пусть $\{c_1, \ldots, c_u\}$ – множество различных значений признака $x, c_{i+1} \geqslant c_i, 1 \leqslant i \leqslant u-1$. Выбирается t порогов, $5 \leqslant t \leqslant 10$, для признака x, делящих обучающую выборку по этому признаку на t равных частей.

Таблица 1 Оптимальное значение ε

Data	ε	Data	ε
Data1	0,6	Data10	0,5
Data2	0	Data11	2
Data3	0,1	Data12	1
Data4	0,5	Data13	1
Data5	0,7	Data14	17
Data6	1,1	Data15	60
Data7	0,5	Data16	170
Data8	0,1	Data17	0
Data9	0	Data18	0

Значение параметра останова ε для каждой задачи определялось эмпирически. Для разных значений ε производилась кросс-валидация. В результате выбиралось то значение ε, при котором достигался наилучший результат алгоритма. В табл. 1 приведено оптимальное значение ε для каждой из рассмотренных задач.

Для оценки качества работы алгоритмов была применена кросс-валидация по k фолдам. Исходные данные разбивались на k подвыборок, $k \geqslant 2$. Затем на $k-1$ подвыборке производилось обучение алгоритма, а оставшаяся подвыборка использовалась для тестирования. Процедура повторялась k раз. В итоге каждая из k подвыборок использовалась для тестирования.

Для оценки эффективности алгоритмов использовались величины MAE (Mean Absolute Error — средняя абсолютная ошибка) и RMSE (Root Mean Squared Error — корень среднеквадратичной ошибки), вычисляемые соответственно следующим образом:

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m}|y_i - h_i|\,; \qquad \mathrm{RMSE} = \frac{1}{\sqrt{m}}\sqrt{\sum_{i=1}^{m}(y_i - h_i)^2},$$

где y_i — значения целевых переменных, а h_i — значения, выданные алгоритмом.

Алгоритмы NBRTree и NBFRTree сравнивались с алгоритмами CART и Random Forest из библиотеки sklearn языка Python, а также с алгоритмами Decision Stump, M5P и REPTree из свободного программного обеспечения для анализа данных WEKA.

Если число объектов в выборке не превышало 350, использовалась кросс-валидация Leave One Out. Для выборок, в которых больше 350 объектов, применялась кросс-валидация по 10 фолдам. Для большей надежности эксперимента кросс-валидация по 10 фолдам производилась 10 раз, после каждой итерации выборка перемешивалась.

В табл. 2–5 приведены результаты тестирования. В табл. 2 и 3 приведены результаты тестирования по методу Leave One Out на шести реальных задачах. В табл. 4 и 5 приведены результаты кросс-валидации (10 раз по 10 фолдам) на 12 задачах.

Из табл. 2–5 видно, что на 11 из 18 реальных задач с функционалом качества MAE наилучшие результаты показал алгоритм NBFRTree, на 4-х — Random forest, на двух — CART и на одной — NBRTree.

На 7 из 18 реальных задач с функционалом качества RMSE наилучшие результаты показал алгоритм NBFRTree, на 7 — Random Forest, на одной — алгоритмы CART, Desicion Stump, NBRTree и M5P.

Таблица 2 Качество работы оценивается функционалом качества MAE

Задачи	Размер $m \times n$	NBRTree	NBFRTree	Decision Stump	M5P	REPTree	CART	Random Forest
Data1	167×4	0,277	0,277	0,645	0,500	0,356	**0,181**	0,219
Data2	209×5	8,391	**7,313**	15,311	14,162	13,306	8,352	8,220
Data3	308×6	0,886	0,662	4,940	2,277	0,800	0,672	**0,511**
Data4	103×7	3,476	3,392	6,013	4,751	4,029	3,313	**2,902**
Data5	100×10	0,150	**0,120**	0,199	0,213	0,219	0,265	0,215
Data6	198×33	0,354	**0,237**	0,336	0,339	0,329	0,298	0,248

Таблица 3 Качество работы оценивается функционалом качества RMSE

Задачи	Размер $m \times n$	NBRTree	NBFRTree	Decision Stump	M5P	REPTree	CART	Random Forest
Data1	167×4	0,505	0,511	1,013	0,840	0,750	**0,402**	0,448
Data2	209×5	22,254	**16,563**	25,770	23,407	26,483	21,480	18,378
Data3	308×6	1,417	**0,877**	7,240	4,218	1,567	1,521	1,060
Data4	103×7	5,160	4,795	7,633	6,108	5,384	4,823	**4,160**
Data5	100×10	0,387	0,346	**0,318**	0,328	0,347	0,512	0,369
Data6	198×33	0,595	0,487	0,421	**0,411**	0,417	0,546	0,498

Таблица 4 Качество работы оценивается функционалом качества MAE

Задачи	Размер $m \times n$	NBRTree	NBFRTree	Decision Stump	M5P	REPTree	CART	Random Forest
Data7	1030×7	4,932	4,672	11,572	6,876	5,613	**4,489**	5,731
Data8	506×13	3,492	**3,106**	5,203	3,607	3,415	3,467	3,857
Data9	1503×5	2,651	**2,647**	5,018	3,318	2,753	2,670	3,404
Data10	9568×4	**3,710**	3,786	7,494	3,871	3,746	3,718	3,723
Data11	517×7	18,597	**18,563**	19,342	18,653	18,626	27,151	30,065
Data12	4898×11	0,490	**0,433**	0,671	0,582	0,563	0,499	0,467
Data13	1599×11	0,436	**0,396**	0,560	0,523	0,510	0,463	0,440
Data14	649×30	2,157	**2,073**	2,201	2,137	2,132	2,783	2,091
Data15	1059×68	16,844	13,895	13,989	14,246	13,929	15,932	**12,768**
Data16	1059×68	42,901	37,466	40,074	37,788	38,996	44,816	**34,166**
Data17	699×9	0,136	**0,113**	0,282	0,244	0,163	0,123	0,125
Data18	569×30	0,076	**0,065**	0,182	0,148	0,105	0,073	0,074

Наилучшие результаты показали алгоритмы NBFRTree и Random Forest. На задачах, на которых наилучшие результаты показал NBFRTree, алгоритм NBFRTree оказался лучше в среднем на 23% (функционал MAE) и на 22% (функционал RMSE) по сравнению с алгоритмом Random Forest. На задачах, на которых наилучшие результаты показал Random Forest, алгоритм NBFRTree оказался хуже в среднем на 16% (функционал MAE) и на 15% (функционал RMSE) по сравнению с алгоритмом Random Forest.

5 Заключение

Разработаны новые алгоритмы для задачи восстановления регрессии, основанные на построении РД (алгоритмы NBRTree и NBFRTree). В обоих алгоритмах используется критерий ветвления, который является модификацией критерия, используемого в хорошо из-

Таблица 5 Качество работы оценивается функционалом качества RMSE

Задачи	Размер $m \times n$	NBRTree	NBFRTree	Decision Stump	M5P	REPTree	CART	Random Forest
Data7	1030×7	7,334	**6,24**	14,508	8,755	7,454	6,698	7,484
Data8	506×13	5,390	**4,664**	6,949	5,216	5,113	5,355	5,205
Data9	1503×5	3,394	**3,369**	6,341	4,210	3,520	3,403	4,252
Data10	9568×4	**4,812**	4,876	9,135	4,966	4,855	4,817	4,838
Data11	517×7	45,738	**45,681**	64,018	63,825	64,470	86,587	77,133
Data12	4898×11	0,852	0,766	0,813	0,745	0,747	0,869	**0,668**
Data13	1599×11	0,778	0,665	0,734	0,670	0,681	0,787	**0,616**
Data14	649×30	2,965	**2,819**	2,908	2,900	2,933	3,794	2,833
Data15	1059×68	23,237	17,435	17,451	17,645	17,675	23,290	**16,766**
Data16	1059×68	54,920	47,427	50,263	47,948	50,844	61,821	**44,370**
Data17	699×9	0,479	0,446	0,549	0,421	0,449	0,484	**0,358**
Data18	569×30	0,272	0,242	0,325	0,236	0,244	0,272	**0,188**

вестном алгоритме CART, на случай k-арного дерева. Алгоритм NBRTree строит классическое k-арное дерево, в котором ветвление проводится только по одному признаку.

Алгоритм NBFRTree строит k-арное ПРРД, в котором ветвление на каждом шаге синтеза проводится по всем признакам, удовлетворяющим критерию ветвления в равной или почти равной мере. Проведено сравнение алгоритмов NBRTree и NBFRTree с алгоритмами CART и Decision Stump, M5P, а также с алгоритмами, имеющими более сложную конструкцию: Random Forest и REPTree. Тестирование проводилось на 18 реальных задачах из ресурса UCI. Эффективность алгоритмов оценивалась стандартными функционалами качества MAE и RMSE. Показано, что качество алгоритма NBFRTree выше качества алгоритмов NBRTree, Decision Stump, REPTree, M5P и CART и не уступает качеству алгоритма Random Forest, а в некоторых случаях показывает и лучшие результаты.

Таким образом, исследованный в данной работе подход к синтезу РД для решения задачи восстановления регрессии — построение ППРД — может быть успешно использован наравне с другими известными подходами к синтезу РД.

Литература

[1] *Djukova E. V., Peskov N. V.* A classification algorithm based on the complete decision tree // J. Pattern Recogn. Image Anal., 2007. Vol. 17. P. 363–367.

[2] *Генрихов И. Е., Дюкова Е. В.* Классификация на основе полных решающих деревьев // Ж. вычисл. мат. мат. физ., 2012. Т. 52. № 4. С. 750–761.

[3] *Breiman L., Friedman J. H., Olshen R. A., Stone C. J.* Classification and regression trees. — CRC Press, 1984. 368 p.

[4] *Breiman L.* Random forests // Machine Learning, 2001. Vol. 45. Iss. 1. P. 5–32.

[5] *Elomaa T., Kääriäinen M.* An analysis of reduced error pruning // J. Artificial Intelligence Res., 2001. Vol. 15. P. 163–187.

[6] *Ai W. I., Langley P.* Induction of one-level decision trees // 9th Conference (International) on Machine Learning Proceedings. — Morgan Kaufmann, 1992. P. 233–240.

[7] *Quinlan J. R.* Learning with continuous classes // Australian Joint Conference on Artificial Intellegence Proceedings. — World Scientific, 1992. P. 343–348.

[8] *Lichman M.* UCI machine learning repository, 2013. `http://archive.ics.uci.edu/ml`.

Поступила в редакцию 18.06.2016

About full regression decision trees*

I. E. Genrikhov, E. V. Djukova, and V. I. Zhuravlyov
ingvar1485@rambler.ru, edjukova@mail.ru, vadim091294@gmail.com
[1]LLC Mobile park IT, 21/1 Panfilova Str., Khimki, Moscow region, Russia
[2]FRC "Computer Science and Control"of RAS, 44/2 Vavilova Str., Moscow, Russia
[3]Lomonosov Moscow State University, 1 Leninskie Gory, Moscow, Russia

Background: The regression restoration problem is considered. The approach based on the construction of regression trees is highlighted among the existing approaches. The most known among algorithms of regression trees synthesis (e. g., algorithms CART and Random Forest) are based on use of the elementary trees, namely, binary regression trees. Rarely, k-ary regression trees are used. Only one feature which is meeting the selected criteria of branching is selected in the synthesis of such trees, and the branching is carried out using this feature. However, only one feature is chosen (randomly) in case when several features are equally or almost equally meeting the selected criteria in construction of regression trees. Thus, the constructed trees depending on the selected feature can significantly vary both on structures of the used features and on its recognition qualities.

Methods: A new approach to the construction of regression trees based on the construction of the so-called full decision tree is applied. Originally, the approach to the synthesis of full decision trees was investigated only on the precedents classification problems and presented improved quality in comparison with the known methods of synthesis of decision trees. So-called full node is built on each iteration in the full decision tree. A set of features corresponds to the full node, and each feature meets the selected branching criterion. Then, the simple internal node from which the branching is carried out is built for each feature of this set. In comparison with the classical construction, the full decision tree allows to use more fully the available information. Herewith, the description of the recognizable object may be generated not by only one branch, as in a classical tree, but by several branches.

Results: Two synthesis algorithms of regression trees — NBFRTree and NBRTree — are developed. The NBRTree algorithm builds classic k-ary regression tree using a statistical criterion selection feature. The NBFRTree algorithm is an improvement of the NBRTree with the approach to the full decision trees synthesis — at each step, a set of features, which are equally or nearly equally meet a statistical criterion, on which branching is carried out is selected. It is shown that the best results were received when the NBFRTree algorithm was used. A comparison of 18 real problems of NBFRTree and NBRTree algorithms with known regression trees synthesis algorithms, such as the Random Forest, Decision Stump, REPTree, and CART, is carried out. It is shown that the quality of the NBFRTree algorithm is higher than the quality of the Decision Stump, REPTree, and CART algorithms, and it does not concede on quality to the Random Forest algorithm and, in some cases, also shows the best results.

Concluding Remarks: It is shown that the applied in this work approach to the regression trees synthesis for the solving of the regression restoration problem — full regression trees — can be successfully used on an equal basis with other known approaches to regression trees synthesis.

Keywords: *regression restoration problem; regression trees; full decision tree*

DOI: 10.21469/22233792.2.1.09

*The research was supported by the Russian Foundation for Basic Research (grant 16-01-00445.).

References

[1] Djukova, E. V., and N. V. Peskov. 2007. A classification algorithm based on the complete decision tree. *J. Pattern Recogn. Image Anal.* 17:363–367.

[2] Genrikhov, I. E., and E. V. Djukova. 2012. Klassifikatsiya na osnove polnykh reshayushchikh derev'ev [Classification based on full decision trees]. *Zh. Vychisl. Matem. Matem. Fiz.* [J. Comput. Math. Math. Phys.] 52(4):653–663.

[3] Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone. 1984. *Classification and regression trees.* CRC Press. 368 p.

[4] Breiman, L. 2001. Random forests. *Machine Learning* 45(1):5–32.

[5] Elomaa, T., and M. Kääriäinen. 2001. An analysis of reduced error pruning. *J. Artificial Intelligence Res.* 15:163–187.

[6] Ai, W. I., and P. Langley. 1992. Induction of one-level decision trees. *9th Conference (International) on Machine Learning Proceedings.* Morgan Kaufmann. 233–240.

[7] Quinlan, J. R. 1992. Learning with continuous classes. *Australian Joint Conference on Artificial Intellegence Proceedings.* World Scientific. 343–348.

[8] Lichman, M. 2013. *UCI machine learning repository.* Available at: `http://archive.ics.uci.edu/ml` (accessed August 14, 2016).

Received June 18, 2016