Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра Математических Методов Прогнозирования

Журавлёв Вадим Игоревич

Построение и исследование полных решающих деревьев для задачи восстановления регрессии

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
д.ф-м.н., доцент
Е.В. Дюкова

Москва 2016

Содержание

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 $\{x_1, \dots, x_n\}$. 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 $T = \{S_1, \dots, S_m\}$ 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 т.д.

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.
   - $Y = \{1, \dots, N\}$ – классификация с N непересекающимися классами.
   - $Y = \{0, 1\}^N$ – классификация с N пересекающимися 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 решающего дерева (РД) представляет собой итерационный процесс. Как правило, для построения очередной вершины дерева выбирается признак, наилучшим 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 выбор одного из них происходит 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 полного решающего дерева (ПРД) [1, 2, 3, 4, 11]. В ПРД на каждой итерации строится так называемая полная вершина, которой соответствует набор признаков $\{x_{j_1}, \dots, x_{j_q}\}$, $q \leq n$, в котором каждый признак удовлетворяет критерию ветвления. Затем для каждого признака x_{j_i}, $i \in \{1, \dots, q\}$, строится внутренняя вершина, из которой осуществляется ветвление.

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 решения этой задачи является алгоритм CART (Classification And Regression Trees), строящий бинарное решающее регрессионное дерево (РРД) с критерием ветвления, основанным на вычислении статистик [10, 12,

15]. Современные модели алгоритмов используют более сложные конструкции РРД.

Основной целью данной работы является построение и исследование полных регрессионных решающих деревьев (ПРРД) для задач с целочисленными данными.

В работе построены и протестированы алгоритмы NBRTree (Non Binary Regression Tree) и NBFRTree (Non Binary Full Regression Tree), строящие k-арные регрессионные деревья, где k – максимальное число ребер, выходящих из обычных (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 [12, 15].

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, NBRTree [8, 9, 12, 13, 18, 19, 23, 27, 28, 29].

2. Задача восстановления регрессии: основные понятия и полученные ранее результаты

Рассмотрим важные понятия, используемые при построении регрессионных решающих деревьев (РРД), на примере бинарного регрессионного решающего дерева (БРРД).

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

На первом шаге $\check{T} = T$, $X(\check{T}) = \{x_1, \dots, x_n\}$. На текущем шаге построения дерева для каждого признака x *из* $X(\check{T})$ и каждого значения a признака x проводится разбиение $\check{T}$ на две подвыборки (подвыборку $\check{T}_R(x, a)$, для объектов которой выполняется неравенство $x \geq a$, и подвыборку $\check{T}_L(x, a)$, для объектов которой выполняется неравенство $x < a$) и вычисляется оценка качества этого разбиения. Среди всех возможных разбиений выбирается разбиение с наилучшей оценкой качества. Такое разбиение называется оптимальным для признака x. Говорят, что признак удовлетворяет критерию ветвления, если оптимальное разбиение для этого признака имеет максимальную оценку качества разбиения. Среди всех признаков удовлетворяющих критерию ветвления, выбирается только один признак.

Различные алгоритмы БРРД отличаются способом оценки качества разбиения для признака x (критерием ветвления), а также правилом останова ветвления.

Сложность построения БРРД очень велика при большом числе признаков, особенно если признаки многозначны.

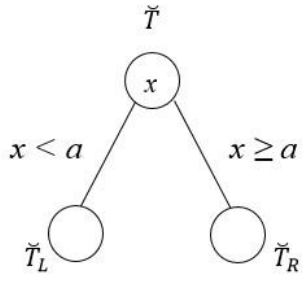

Рисунок 1. Пример БРРД

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

Ниже описываются основные алгоритмы, использующие деревья решений для задачи восстановления регрессии.

Алгоритм *CART* (Leo Breiman, Jerome Friedman, Richard Olshen, Stone, 1983) строит БРРД и при выборе оптимального разбиения использует статистический подход к оценке качества разбиения (критерий ветвления, основанный на вычислении статистик) [10, 12]. Опишем этот критерий.

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

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

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

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

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

$$C(x) = V - SE(x).$$

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

Похожую на CART конструкцию имеет алгоритм M5P. Алгоритм M5P, так же как и алгоритм CART, выполняет построение бинарных решающих деревьев. Во время построения дерева решений алгоритм M5P использует энтропийный критерий ветвления. Алгоритм M5P является модификацией (для решения задач регрессии) алгоритма M5, построенного Quinlan в 1992 году [12, 19, 28].

Существуют и другие известные алгоритмы, позволяющие строить более сложные конструкции РРД, а именно: Decision Stump, REPTree, Random Forest.

Алгоритм DecisionStump представляет собой одноуровневое дерево со статистическим критерием ветвления [13, 27]. Это дерево с корневой вершиной, которая соединена ребром с каждой из висячих вершин. Decision Stump последовательно рассматривает каждый признак x и строит для этого признака отдельное дерево. Возможны варианты: 1) для каждого значения признака x строится висячая вершина; 2) выбирается число a (порог) и строятся две вершины, в одной из которых $x < a$, а во второй $x \geq a$; 3) множество значений признака x разбивается на интервалы и строится дерево с числом вершин, равным числу этих интервалов.

Алгоритм REPTree (Reduced Error Pruning Tree) строит бинарные деревья для задач классификации и регрессии, используя соответственно энтропийный и статистический критерии ветвления. Этот алгоритм впервые был предложен Quinlan в 1987 [18, 23, 29].

Random Forest – алгоритм машинного обучения, заключающийся в использовании комитета (ансамбля) решающих деревьев (предложен Лео Брейманом и Адель Катлер в 2001 г.).

Алгоритм применяется для задач классификации, регрессии и кластеризации. В алгоритме Random Forest используется энтропийный критерий ветвления и процедура «бэггинг» [8, 9]. 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 определяется путем усреднения значений целевой функции по всем построенным РД. Таким образом, деревья компенсируют ошибки друг друга.

3. Алгоритмы восстановления регрессии NBRTree и NBFRTree

В данном разделе описаны алгоритмы восстановления регрессии NBRTree и NBFRTree, построенные в работе. Приведены результаты тестирования этих алгоритмов на реальных задачах. Оба алгоритма предназначены для обработки целочисленной информации.

3.1. Построение алгоритмов NBRTree и NBFRTree

Алгоритм NBRTree строит классическое регрессионное решающее дерево, то есть если несколько признаков удовлетворяют критерию ветвления в равной мере, то выбирается один из них.

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

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.

Не ограничивая общности, будем считать, что признак x имеет значения из $\{0, 1, \dots, k-1\}$, $k \geq 2$. В этом случае при построении дерева решений из вершины x выходят k дуг, помеченные числами из $\{0, 1, \dots, k-1\}$. Пусть σ – метка одной из дуг, выходящих из вершины x, $\sigma \in \{0, 1, \dots, k-1\}$. Для формирования нового текущего подмножества объектов и нового текущего множества признаков удаляются те объекты из $\check{T}$, для которых значение признака x не равно σ, а также из множества признаков удаляется сам признак x.

Положим

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

Пусть v – висячая вершина, порожденная ветвью дерева с внутренними вершинами $x_{j_1}, \dots, x_{j_r}$ и пусть дуга, выходящая из вершины x_{j_i}, $i \in \{1, \dots, r\}$, имеет метку σ_i. 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 $x_{j_1}^{\sigma_1} \dots x_{j_r}^{\sigma_r}$. Если вершина v не является висячей, то ей в соответствие поставим конъюнкцию $B = x_{j_1}^{\sigma_1} \dots x_{j_r}^{\sigma_r}$. Интервал истинности элементарной конъюнкции B обозначим через N_B.

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

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

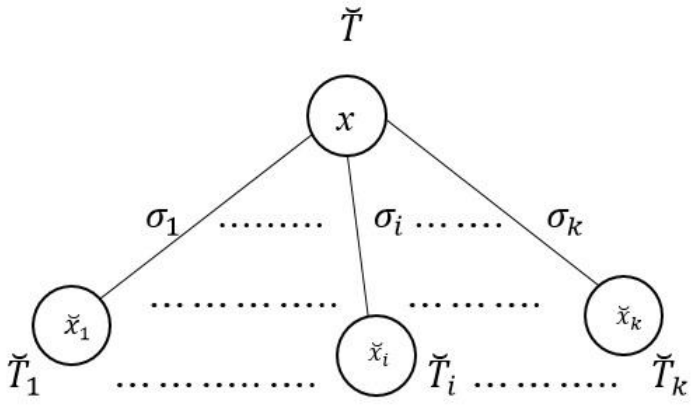

Рисунок 2. NBRTree

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

Процедура распознавания объекта S выполняется следующим образом. Пусть $V = \{v_1, \dots, v_l\}$ – множество висячих вершин построенного дерева с соответствующими парами $(B_i, w(v_i))$, $i = 1, \dots, l, l \geq 1$. 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 $\{x_{j_1}, \dots, x_{j_r}\}$ в алгоритме NBFRTree. Ветвление из простых вершин $x_{j_1}, \dots, x_{j_r}$ производится как в алгоритме NBRTree.

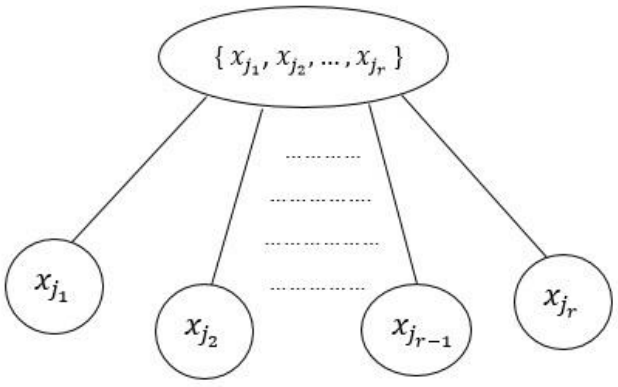

Рисунок 3. Пример NBFRTree

Опишем критерий ветвления, используемый в алгоритмах NBRTree и NBFRTree.

Пусть $\check{T}_i = \{S_1^i, \dots, S_{u_i}^i\}$, y_t^i – значение целевой переменной обучающего объекта S_t^i, $i \in \{1, 2, \dots, u_i\}$.

$$\bar{y}_{\check{T}_i} = \frac{1}{u_i}\sum_{t=1}^{u_i} y_t^i,$$

$$V = \sum_{t=1}^{u_i}(y_t^i)^2 - [\sum_{t=1}^{u_i}(y_t^i)]^2,$$

Пусть рассматриваемый признак x принимает k значений. Обучающую выборку $\check{T}_i$ можно разбить по этому признаку на k подвыборок $\check{T}_{i_1}, \dots, \check{T}_{i_k}$.

$$SE(x,k) = \frac{1}{u_i}\left\{\sum_{S_t^i \in \check{T}_{i_1}}\left(y_t^i - \bar{y}_{\check{T}_{i_1}}\right)^2 + \dots + \sum_{S_t^i \in \check{T}_{i_k}}\left(y_t^i - \bar{y}_{\check{T}_{i_k}}\right)^2\right\}.$$

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

Наилучшее разбиение в алгоритме NBRTree выбирается следующим образом. Для каждого признака $x \in X(\check{T}_i)$ вычисляется величина $C(k,x) = V - SE(x,k)$. Наилучшим признаком для ветвления считается тот, для которого значение $C(k,x)$ максимально.

В алгоритме NBFRTree наилучшее разбиение выбирается иначе. Пусть C_{min} и C_{max} – минимальное и максимальное значения $C(k,x)$ соответственно. Сначала для каждого признака $x \in X(\check{T}_i)$ вычисляется величина $C(k,x) = V - SE(x,k)$. Далее значение $C(k,x)$ нормируется и вычисляется по формуле

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

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

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

3.2. Тестирование алгоритмов NBRTree и NBFRTree

Алгоритмы были протестированы на 18 реальных задачах из ресурса UCI [16]. 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*, $c_{i+1} \geq c_i, 1 \leq i \leq u-1$. Далее выбирается *t* порогов для признака *x*, делящих обучающую выборку по этому признаку на *t* равных частей $5 \leq t \leq 10$.

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

Таблица 1. Оптимальное значение ε

Data	Data1	Data2	Data3	Data4	Data5	Data6	Data7	Data8	Data9
ε	0.6	0	0.1	0.5	0.7	1.1	0.5	0.1	0
Data	Data10	Data11	Data12	Data13	Data14	Data15	Data16	Data17	Data18
ε	0.5	2	1	1	17	60	170	0	0

Для оценки качества работы алгоритмов была применена перекрёстная проверка по k частям. Исходные данные разбивались на k подвыборок, $k \geq 2$. 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:

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

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

Алгоритмы NBRTree и NBFRTree сравнивались с алгоритмами CART и Random Forest (RF) из библиотеки sklearn языка Python, а

также с алгоритмами DecisionStump (DS), M5P и REPTree из свободного программного обеспечения для анализа данных WEKA [26, 27, 28].

Если число объектов в выборке не превышало 350, использовался метод тестирования Leave One Out ($k = m$, где m – число обучающих объектов). Для выборок, в которых больше 350 объектов применялась перекрёстная проверка по 10 частям (k = 10). Для большей надежности эксперимента перекрёстная проверка по 10 частям производилась 10 раз, после каждой итерации выборка перемешивалась.

В табл. 2, 3, 4, 5 приведены результаты тестирования. В табл. 2, 3 приведены результаты тестирования по методу Leave One Out на шести реальных задачах. В табл. 4, 5 приведены результаты перекрёстной проверки (10 раз по 10 частям) на 12 задачах.

Таблица 2. Качество работы оценивается функционалом качества MAE

Задачи	Размер $m * n$	NBRTree	NBFRTree	DC	M5P	REPTree	CART	RF
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 $m * n$	NBRTree	NBFRTree	DS	M5P	REPTree	CART	RF
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 $m * n$	NBRTree	NBFRTree	DS	M5P	REPTree	CART	RF
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

Таблица 5. Качество работы оценивается функционалом качества RMSE

Задачи	Размер $m * n$	NBRTree	NBFRTree	DS	M5P	REPTree	CART	RF
Data7	1030*7	7.334	**6.624**	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**

4. Заключение

Получены следующие результаты:

1) Модифицирован на случай k-арного дерева критерий ветвления алгоритма CART и построен алгоритм NBRTree, строящий классическое k-арное регрессионное дерево;
2) На базе алгоритма NBRTree построен алгоритм NBFRTree, строящий полное решающее регрессионное дерево;
3) Проведено тестирование алгоритмов NBRTree и NBFRTree на реальных задачах. На большинстве задач наилучшие результаты показал алгоритм NBFRTree. Достаточно хорошие результаты показали алгоритмы Random Forest и CART, строящие бинарные регрессионные деревья.

5. Список литературы:

1. Генрихов И. Е., Дюкова Е. В. Классификация на основе полных решающих деревьев // Ж. вычисл. матем. и матем. физ. – 2012. – Т. 52, № 4. – С. 750761.
2. Генрихов И. Е., Дюкова Е. В. Построение и исследование распознающих процедур на основе полных решающих деревьях // Междунар. конф. Интеллектуализация обработки информации-8. – М.: МАКС Пресс, 2010. – С. 117-120.
3. Генрихов И. Е., Дюкова Е. В. Усовершенствование алгоритма С4.5 на основе использования полных решающих деревьев // Всеросс. конф. Математические методы распознавания образов-14. – М.: МАКС Пресс, 2009. – С. 104-107.
4. Дюкова Е. В., Песков Н. В. Об алгоритме классификации на основе полного решающего дерева // Всеросс. конф. Математические методы распознавания образов-13. – М.: МАКС Пресс, 2007. – С. 125-126.
5. Дюкова Е.В. и др. Обработка веществennозначной информации логическими процедурами распознавания // Искусств. интеллект. 2004. № 2. С. 80–85.
6. Журавлёв Ю.И. “Об алгебраическом подходе к решению задач распознавания или классификации”
7. Журавлёв Ю.И., Рязанов В.В., Сенько О.В. “Распознавание. Математические методы. Программная система, практические применения”. Москва: ФАЗИС, 2006.
8. Breiman, L. Consistency for a simple model of Random Forests. Technical report, University of California at Berkeley, 2004.
9. Breiman, L. Random Forests. Machine Learning, 45(1):5–32, 2001.
10. Clifton D. "Classification and Regression Trees, Bagging, and Boosting" // Handbook of Statistics, 2005, Vol. 24, pp. 303-329.
11. Djukova E.V., Peskov N.V. A classification algorithm based on the complete decision tree // J. Pattern Recognition and Image Analysis. 2007. V. 17. № 3. P. 363–367.
12. Guvenir, HA. Uysal, I, " An overview of regression techniques for knowledge discovery" // The Knowledge Engineering Review, Vol. 14:4, 1999, 319-340.
13. Iba, Wayne; and Langley, Pat (1992); Induction of One-Level Decision Trees, *in* ML92: Proceedings of the Ninth International Conference on Machine Learning,

Aberdeen, Scotland, 1–3 July 1992, San Francisco, CA: Morgan Kaufmann, pp. 233–240.

14. Kuncheva L. I. "Combining pattern classifiers methods and algorithms" // John Wiley & Sons, Inc., Hoboken, New Jersey, 2004. – Pp. 154-163.
15. L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone "Classification and Regression Trees" // CRC Press, 1984.
16. Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
17. Luís Fernando Raínho Alves Torgo «Inductive learning of tree-based regression models», 1999
18. M. Zontul, F. Aydin, G. Dogan, S. Sener, O. Kaynar “Wind speed forecasting using reptree and bagging methods in kirklareli-turkey” in Journal of Theoretical and Applied Information Technology 10th October 2013. Vol. 56 No.1
19. Quinlan, J. R. (1992), “Learning with continuous classes,” in Proceedings of AI’92 Australian National Conference on Artificial Intelligence, 343–348, Singapore: World Scientific.
20. Quinlan, J.R. Induction of decision trees. Machine Learning 1(1) 81-106, 1986
21. Quinlan, J.R., "Combining instance-based and model-based learning", Proc. ML'93 (ed P.E. Utgoff), San Mateo: Morgan Kaufmann 1993
22. R. Timofeev “Classification and Regression Trees (CART) Theory and Applications”, 2004
23. T. Elomaa and M. Kaariainen. “An Analysis of Reduced Error Pruning” Journal of Artificial Intelligence Research (2001) Volume 15, pages 163-187
24. Wang ,Y., Witten, I. H.: Induction of model trees for predicting continuous classes. In : Poster papers of the 9th European Conference on Machine Learning, 1997
25. Wei-Yin Loh «Logistic Regression Tree Analysis» in Handbook of Engineering Statistics, H. Pham, ed., 537–549, Springer, 2006
26. WEKA: suite of machine learning software, developed at the University of Waikato, New Zealand, 2009. REPTree. URL: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/REPTree.html (дата обращения: 15.04.2016)

27. WEKA: suite of machine learning software, developed at the University of Waikato, New Zealand, 2009. DecisionStump. URL: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/DecisionStump.html (дата обращения: 15.04.2016)
28. WEKA: suite of machine learning software, developed at the University of Waikato, New Zealand, 2009. M5P. URL: http://weka.sourceforge.net/doc.dev/weka/classifiers/trees/M5P.html (дата обращения: 15.04.2016)
29. Wilkinson, L. 1998. Classification and regression trees in Systat 8.0 Statistics, SPSS, Inc., United States of America, 31–51.
30. Y. Yohannes, J. Hoddinott "Classification and regression trees: an introduction" International Food Policy Research Institute 2033 K Street, N.W. Washington, D.C., 1999
31. Y. Zhao, Y. Zhan "Comparison of decision tree methods for finding active objects", 2007