КОНСТРУКТИВНЫЕ АЛГОРИТМЫ СИНТЕЗА ЛОКАЛЬНО-ОПТИМАЛЬНЫХ ДЕРЕВЬЕВ И МУЛЬТИДЕРЕВЬЕВ РЕШЕНИЙ

Шибзухов З.М., Тезадов С.М.

Институт прикладной математики и автоматизации КБНЦ РАН, РОССИЯ
360000, КБР, г.Нальчик, Шортанова 89А
sz@zmail.ru

ABSTRACT

Multiple-valued decision trees and decision multi-trees are considered. Recursive algorithm for constructing decision trees from data tables are defined. This algorithm use 2-level local optimization of variables selection on the base of various functionals for inclusion into tree structure. It was implemented in package PyDecisionTree in extensible object-oriented language Python.

ВВЕДЕНИЕ

Пусть $\boldsymbol{y}=\{y_1,\ldots,y_m\}$ – набор целевых переменных, $y_j\in\mathbf{Y}_j$, $j=1,\ldots,m$, $\mathbf{Y}_j$ – конечное множество значений j-ой целевой переменной; $\boldsymbol{x}=\{x_1,\ldots,x_n\}$ – набор основных переменных, $x_i\in\mathbf{X}_i$, $i=1,\ldots,n$, $\mathbf{X}_i$ – конечное множество значений i-ой основной переменной. Между $\boldsymbol{y}$ и $\boldsymbol{x}$ имеет место качественная зависимость $\mathbf{F}=\{\mathrm{F}_1,\ldots,\mathrm{F}_m\}$, которая позволяет для каждого набора значений основных переменных получить набор значений целевых переменных:

$$[s_{j1}/p_{j1},\ldots,s_{jv(j)}/p_{jv(j)}] = \mathrm{F}_j(x_1,\ldots,x_n),\quad j=1,\ldots,m,$$

где $s_{j1},\ldots, s_{jv(j)}$ – значения j-ой целевой переменной с "достоверностями" $p_{j1},\ldots, p_{jv(j)}$, соответственно. В качестве "достоверности" решения может, например, выступать вероятность правильности или правдоподобия решения, вычисленные по обучающей информации.

Качественные зависимости $\mathrm{F}_1,\ldots,\mathrm{F}_m$ как по отдельности, таки все в целом представляются в форме дерева решений (ДР) или мультидерева решений (МДР), которые будут определены ниже. Предполагается, что имеется таблица вида:

$$\mathbf{D} = \{\langle\mathbf{y}_k,\mathbf{x}_k\rangle{:}k=1,\ldots,\mathrm{N}\},$$

где $\mathbf{y}_k,\mathbf{x}_k$ – соответственно, список значений целевых и основных переменных в k-ой строке, N – количество строк таблице. Требуется построить качественную зависимость **F** в форме ДР или МДР на основе информации, содержащейся в таблице **D**.

СТРУКТУРА ДЕРЕВА РЕШЕНИЙ

Логическая структура ДР представляется при помощи составного логического терма[1], который будем называть термом ДР. Структура терма ДР включает в себя подтермы, представляющие поддеревья ДР. Каждому такому подтерму соответствует узел ДР, являющийся корневым в поддереве. Имеются несколько видов узлов.

Первый вид – это узлы *решений*, который представляют листовую часть ДР:

$$\texttt{решение}(y, [s_1/p_1,\ldots,s_k/p_k]), \qquad (1')$$

$$\texttt{решение}(y, [s_1/p_1,\ldots,s_k/p_k], \mathrm{t}'), \qquad (1'')$$

который определяет решение по целевой переменной y со значениями $s_1, \ldots, s_k$ с достоверностью $p_1, \ldots, p_k$, соответственно. Если в ДР имеются решения по другим целевым переменным, то t' – корневой узел поддерева решений для нахождения их значений.

[1] Здесь и далее термы и правила для удобства будут записываться в нотации языка логического программирования PROLOG.

Второй вид – это узлы *выбора* пути поиска решений по значениям некоторой переменной:

выбор(x, [v_1,…,v_k,], [t_1,…,t_k]), (2')

выбор(x, [v_1,…,v_k,], [t_1,…,t_k, t_{k+1}]), (2")

где x – переменная по значениям которой осуществляется выбор дальнейшего пути поиска решений, v_1,…,v_k – значения x, t_1,…,t_k – соответственно, корневые узлы поддеревьев, в которых может быть продолжен процесс поиска альтернативных решений.

Третий вид – это узлы *альтернатив*, которые определяют альтернативные пути поиска решений:

альтернативы([t_1,…,t_m]), (3)

где t_1,…,t_m обозначают корневые узлы поддеревьев, которые могут содержать альтернативные решения.

Таким, образом терм ДР является таким термом, в котором каждый составной подтерм имеет вид (1')-(1"), (2')-(2"), (3).

Отметим подкласс ДР, который не содержит узлов альтернатив. В этом случае поиск по ДР приводит не более чем к одному решению. При последовательной реализации процедуры поиска по такому ДР не возникает необходимость возвратов (backtracking) или циклов перебора вариантов. Такие деревья будем называть безальтернативными ДР.

Каждое ДР, содержащее узлы альтернатив можно разбить на несколько безальтернативных ДР. Каждое такое безальтернативное дерево получается путем замещения каждого узла альтернатив с поддеревьями t_1,…,t_m на любой узел t_i, $1 \le i \le m$. В результате получается список безальтернативных ДР, который является эквивалентным исходному ДР в следующем смысле: множество решений исходного ДР и множество решений всех безальтернативных ДР в кортеже совпадают.

Под МДР будем понимать дерево, в котором в качестве внутренних узлов выступают ДР с “висячими” вершинами, а в качестве листовых узлов – ДР без “висячих” вершин. Для идентификации узлов МДР используется некоторое множество символов – меток. Логическая структура ДР с “висячими” узлами представляется при помощи термов вида (1')-(1"), (2')-(2"), (3), в которые дополнительно могут входить еще и символы меток. Если заменить все символы меток в таком терме на термы деревьев решений, то получится терм ДР.

Таким образом, логическую структуру МДР можно представить в виде совокупности термов вида:

мдр(τ, f(τ_1,…, τ_m)),

где выражение f(τ_1,…,τ_m) обозначает составной логический терм, в котором присутствуют символы τ_1,…,τ_m, идентифицирующие “потомков” узла МДР.

МДР можно построить из ДР, путем разбиения терма ДР на подтермы. Необходимость в МДР возникает при работе с ДР большой сложности.

ПОСТРОЕНИЕ ДЕРЕВА РЕШЕНИЙ

Синтез ДР осуществляется при помощи рекурсивного алгоритма, основанного на принципе декомпозиции задач с локальной оптимизацией на каждом шаге рекурсии выбора основных переменных и наборов их значений, которые будут использоваться при формировании узлов выбора и узлов альтернатив. Опишем процедуру декомпозиции.

Обозначим $\Lambda(y_j|\mathbf{D})$ функционал "достоверности" значения целевой переменной y_j на произвольной таблице $\mathbf{D}$: максимальное значение функционала соответствует наиболее достоверным значениям, а минимальное – наименее достоверным. В качестве Λ, например, можно выбрать следующие функционалы:

– $\max \{P(y_j=s_1|\mathbf{D}), \ldots, P(y_j=s_k|\mathbf{D})\}$,
– $1 + (P(y_j=s_1|\mathbf{D})\log P(y_j=s_1|\mathbf{D}) + \ldots + P(y_j=s_k|\mathbf{D})\log P(y_j=s_k|\mathbf{D}))/\log k$;
– $1 - (1 - P(y_j=s_1|\mathbf{D}))\ldots(1 - P(y_j=s_k|\mathbf{D}))$,
– $(P(y_j=s_1|\mathbf{D})^2 + \ldots + P(y_j=s_k|\mathbf{D})^2)/k$

где $P(y_j=s_j|\mathbf{D})$ – частота появления j-го значения s_j целевой переменной y_j в таблице $\mathbf{D}$. Естественно, что список функционалов не исчерпывается приведенными.

Пусть на некотором шаге алгоритма необходимо построить ДР $\mathbf{T}_0$ по таблице $\mathbf{D}_0$. Обозначим $\boldsymbol{x}_0 \subseteq \boldsymbol{x}$ – подмножество основных переменных, которые на таблице $\mathbf{D}_0$ принимают более одного значения, $\boldsymbol{y}_0 \subseteq \boldsymbol{y}$ – подмножество целевых переменных для которых ищется решение. Обозначим $\Delta=|\mathbf{D}_0|/|\mathbf{D}|$ – относительный размер таблицы $\mathbf{D}_0$ по отношению к первоначальной таблице $\mathbf{D}$, по которой осуществляется синтеза ДР. Обозначим $\Phi(y_j|\mathbf{D})$ – функционал "достоверности" значения целевой переменной y_j на таблице $\mathbf{D}_0$ с учетом относительного ее размера.

В качестве примера можно привести следующие:

- $\Lambda(y_j|\mathbf{D}_0)^{\alpha}\Delta^{\beta}$;
- $\Lambda(y_j|\mathbf{D}_0)^{\alpha} + \Delta^{\beta}$;
- $\Lambda(y_j|\mathbf{D}_0)^{\alpha} + (1 - \Lambda(y_j|\mathbf{D}_0))^{\gamma}\Delta^{\beta}$;
- $\Lambda(y_j|\mathbf{D}_0)^{\alpha} + (1 - \Lambda(y_j|\mathbf{D}_0))^{\gamma}(1 - \Delta)^{\beta}$.

Возможны два тривиальных случая:

- $\boldsymbol{x}_0 \neq \varnothing$.

Вычисляется значения $\Phi_j=\Phi(y_j,|\mathbf{D}_0)$ для всех целевых переменных $y_j \in \boldsymbol{y}_0$. Если для некоторого j $\Phi_j \geq \Phi^*$, где Φ^* – пороговое значение ("близкое" к максимальному), то строится узел решения. Если $|\boldsymbol{y}_0|>1$, то возвращается терм ⟨решение(y_j, Vs)⟩, иначе рекурсивно вызывается процедура построения ДР T' для нахождения значений целевых переменных из $\boldsymbol{y}_0 \setminus \{y_j\}$ по той же таблице $\mathbf{D}_0$ и возвращается терм ⟨решение(y_j, Vs, T')⟩. Здесь список Vs состоит из пар (s,p), s пробегает значения y_j, $p=P(y_j=s|\mathbf{D}_0)$.

- $\boldsymbol{x}_0 = \varnothing$.

Возвращаются термы ⟨решение(y_j, Vs)⟩ для каждой целевой переменной $y_j \in \boldsymbol{y}_0$, которые определяют значения целевых переменных с их частотами на таблице $\mathbf{D}_0$.

В других случаях выбирается некоторое подмножество пар (x_i,a_i), $x_i \in \boldsymbol{x}_0$, $a_i \in \mathbf{X}_{0i}$, $\mathbf{X}_{0i}$ – множество значений x_i на таблице $\mathbf{D}_0$. Выбор пар (x_i,a_i) осуществляется из условия

$$\Phi(y_j|\mathbf{D}_0(x_i,a_i)) > \Phi^* - \varepsilon(\Phi^* - \Phi_*),$$

где Φ^* – максимальное значение $\Phi(y_j|\mathbf{D}_0(x_i,a_i))$ по всем парам (x_i,a_i) на таблице $\mathbf{D}_0$, Φ^* – минимальное; $\mathbf{D}_0(x_i,a_i) = \{\langle \mathbf{y},\mathbf{x}\rangle \in \mathbf{D}_0 : x_i=a_i\}$ – подмножество строк таблицы $\mathbf{D}_0$, в которых значение основной переменной x_i равно a_i; $0<\varepsilon<1$. Это первый этап отбора основных переменных для включения в структуру дерева..

Множество отобранных пар разбивается на группы вида (x_i,A_i), A_i – множество значений x_i, которые встречались в отобранных парах. Если групп оказалось более одной, то необходимо выбрать M ($M \geq 1$) “лучших” групп. Для этого они упорядочиваются в порядке убывания значения заранее выбранного функционала $\Psi(x_i,A_i)$. В качестве такого функционала можно привести следующие:

- $\Psi(x_i,A_i) = \max \{\Phi(y_j|\mathbf{D}_0(x_i,a_i)) : a_i \in A_i \text{ и } y_j \in \boldsymbol{y}_0\}$;
- $\Psi(x_i,A_i) = \min \{\Phi(y_j|\mathbf{D}_0(x_i,a_i)) : a_i \in A_i \text{ и } y_j \in \boldsymbol{y}_0\}$.

Если число отобранных групп оказалось $1<m\leq M$, строится терм ⟨альтернативы($[t_1,\ldots,t_m]$)⟩ и термы для выбора ⟨выбор(x_i, A_i', Ts_i)⟩, где $A_i' = A_i \cup \{\texttt{else}\}$, Ts_i – список термов вида $[t_{i1},\ldots,t_{iv(i)},t_{\texttt{else}}]$, которые строятся рекурсивным применением описанной здесь процедуры по подтаблицам $\mathbf{D}_0(a_{i1}),\ldots,$ $\mathbf{D}_0(a_{iv(i)})$, $\mathbf{D}_0(\texttt{else})$, где $\mathbf{D}_0(a_{it})=\{\langle \mathbf{y},\mathbf{x}\rangle \in \mathbf{D}_0 : x_i=a_t\}$, $\mathbf{D}_0(\texttt{else})=\{\langle \mathbf{y},\mathbf{x}\rangle \in \mathbf{D}_0 : x_i \notin A_i\}$. Если была выбрана одна группа, то строится только один терм выбора.

Полное дерево решений строится рекурсивным применением описанной выше процедуры, начиная с исходной таблицы $\mathbf{D}$.

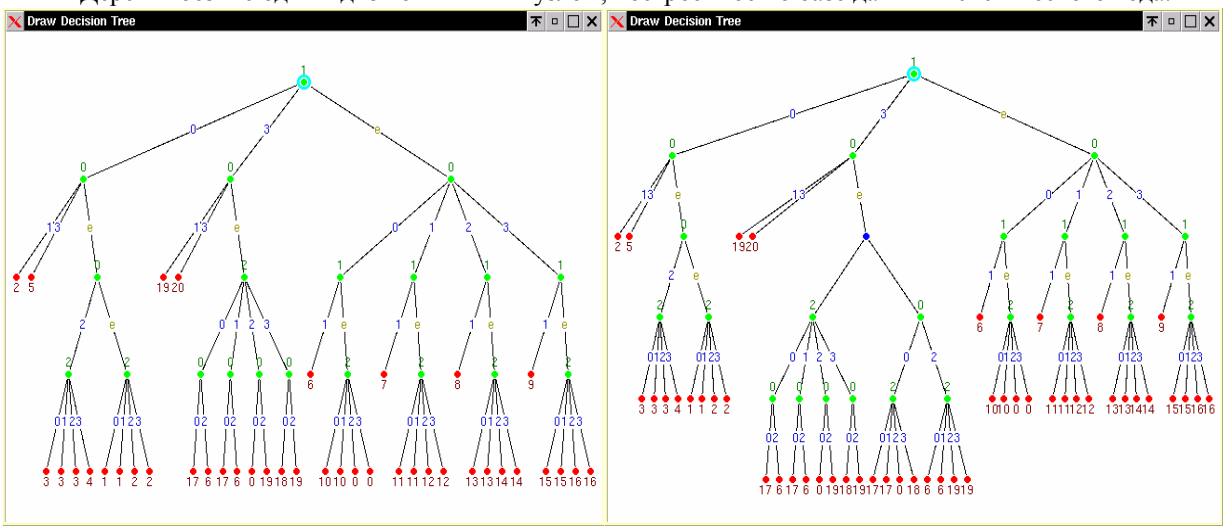
ПАКЕТ PYDECISIONTREE

На основе языка объектно-ориентированного языка Python (версия 1.5.2) разработана пакет PyDecisionTree (версия 2.0 pre-α) для построения, визуализации деревьев решений и принятия по ним решений. В нем непосредственно реализован описанный рекурсивный метод построения деревьев решений с локальной оптимизацией узлов выбора на основе различных функционалов Ф.
Пакет представляет собой набор классов для описания деревьев решений общего вида с возможностью расширения их структуры и функциональности на основе объектно-ориентированных возможностей языка Python.

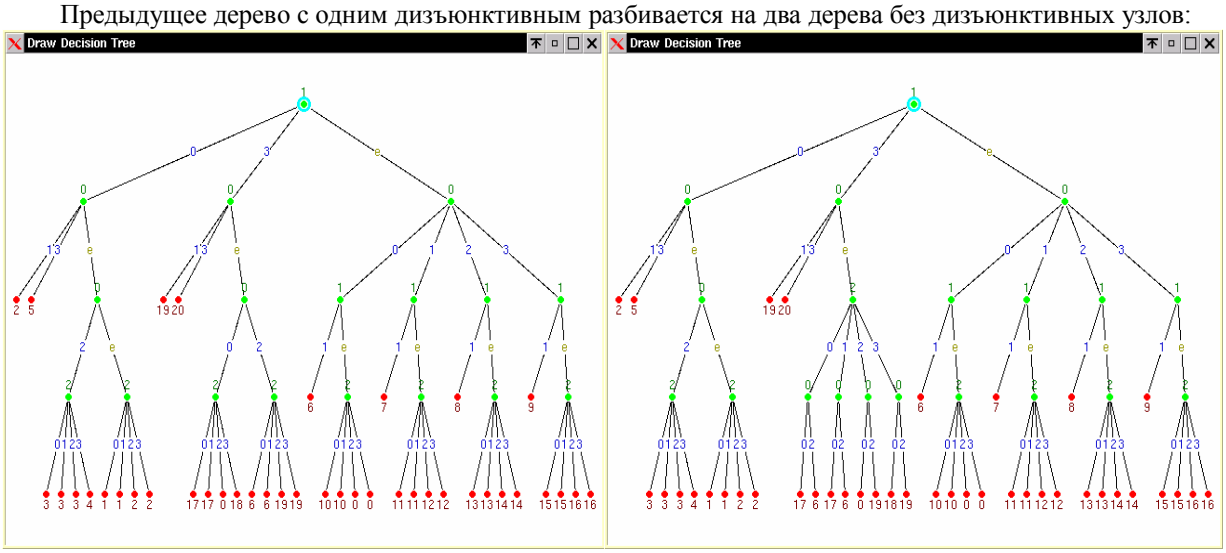
ПРИЛОЖЕНИЕ: ПРИМЕР ПОСТРОЕНИЯ ДР ПРИ ПОМОЩИ ПАКЕТА PYDECISIONTREE

Разметка: зеленым цветом выделены селекторные узлы, красным – решающие, синим – дизъюнктивные. Над каждым селекторным узлом указан номер признака, по которому селекторный узел производит разветвление. Под каждым решающим узлом подписано значение целевого признака, соответствующее ветке от корня до решающего узла. Ребра, исходящие из селекторных узлов помечены номером значения признака селекторного узла; метка ‘e’ ребра означает, что это else-ребро.

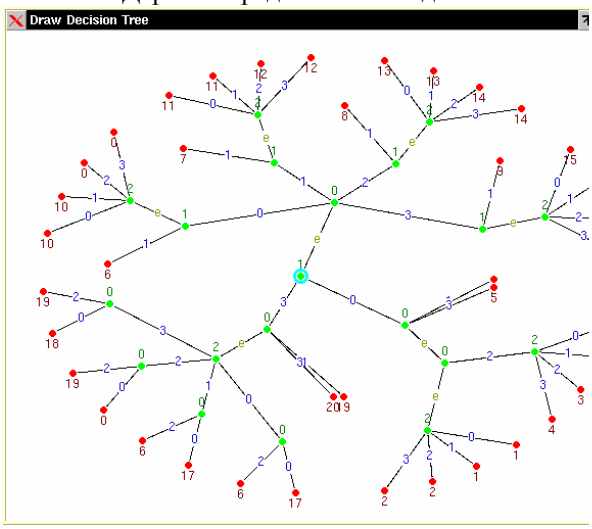
Деревья без и с одним дизъюнктивным узлом, построенное по базе данных генетического кода.



Предыдущее дерево с одним дизъюнктивным разбивается на два дерева без дизъюнктивных узлов:



Дерево в “радиальном” виде:



База данных генетического кода

[1],[0,0,0]	[6],[0,1,0]	[10],[0,2,0]	[17],[0,3,0]
[1],[0,0,1]	[6],[0,1,1]	[10],[0,2,1]	[17],[0,3,1]
[2],[0,0,2]	[6],[0,1,2]	[0],[0,2,2]	[0],[0,3,2]
[2],[0,0,3]	[6],[0,1,3]	[0],[0,2,3]	[18],[0,3,3]
[2],[1,0,0]	[7],[1,1,0]	[11],[1,2,0]	[19],[1,3,0]
[2],[1,0,1]	[7],[1,1,1]	[11],[1,2,1]	[19],[1,3,1]
[2],[1,0,2]	[7],[1,1,2]	[12],[1,2,2]	[19],[1,3,2]
[2],[1,0,3]	[7],[1,1,3]	[12],[1,2,3]	[19],[1,3,3]
[3],[2,0,0]	[8],[2,1,0]	[13],[2,2,0]	[6],[2,3,0]
[3],[2,0,1]	[8],[2,1,1]	[13],[2,2,1]	[6],[2,3,1]
[3],[2,0,2]	[8],[2,1,2]	[14],[2,2,2]	[19],[2,3,2]
[4],[2,0,3]	[8],[2,1,3]	[14],[2,2,3]	[19],[2,3,3]
[5],[3,0,0]	[9],[3,1,0]	[15],[3,2,0]	[20],[3,3,0]
[5],[3,0,1]	[9],[3,1,1]	[15],[3,2,1]	[20],[3,3,1]
[5],[3,0,2]	[9],[3,1,2]	[16],[3,2,2]	[20],[3,3,2]
[5],[3,0,3]	[9],[3,1,3]	[16],[3,2,3]	[20],[3,3,3]

ЛИТЕРАТУРА

1. Тимофеев А.В., Шибзухов З.М. Методы синтеза и оптимизации баз знаний по базам данных на основе локально-оптимальных логико-вероятностных алгоритмов. – International journal "Information theories and applications". 1995. Vol. 3. № 2. Pp 12-19.
2. Тимофеев А.В., Шибзухов З.М. Методы синтеза и оптимизации баз знаний по базам данных на основе локально-оптимальных алгоритмов. – Proc. XXII International conference CAD-95 (Grimea, Gurzuff, may, 1995). Part I. Pp.28.