«Искусственный интеллект»

Использование деревьев решений для интеллектуального анализа данных и извлечения правил из нейронных сетей[1]

Евдокимов И.А., Солодовников В.И., Филипков С.В.
Учреждение Российской академии наук
Центр информационных технологий в проектировании РАН
Россия, Московская область, г. Одинцово, ул. Маршала Бирюзова, д.7а
Тел.: (495) 596-02-19, E-mail: info@ditc.ras.ru

Введение

Данные являются ценным ресурсом, который хранит в себе большие потенциальные возможности по извлечению полезной аналитической информации. Поэтому, все большую актуальность приобретают задачи выявления скрытых закономерностей, выработки стратегий принятия решений, прогнозирования. Внедрение средств автоматизации в системы интеллектуального анализа данных способно сократить сроки, повысить качество и эффективность принимаемых решений. В связи с этим представляет интерес совместное использование нейросетевых технологий и деревьев решения, где нейронные сети выступают в роли основного средства обработки данных, а деревья решения помогают интерпретировать полученные результаты и представить их в доступном для понимания виде.

1. Деревья решений

Логические методы, к которым можно отнести системы рассуждений на основе аналогичных случаев, деревья решений и алгоритмы ограниченного перебора, позволяют выявить логические закономерности в данных. Основное требование к математическому аппарату заключается в интерпретации полученных результатов. Деревья решений обладают свойством наглядности и позволяют анализировать сочетания признаков, реализуя последовательный многошаговый процесс принятия решений. Они предоставляют возможность прогнозировать и связывать различные параметры изучаемого явления в единое целое и объясняют связи, которые нередко бывают далеко не очевидны.

1.1. Структурные характеристики

Дерево решений состоит из вершин двух типов. Вершины решений, содержащие вопросы, обозначаются окружностями. Цели или логические выводы обозначаются прямоугольниками. Вершины нумеруются и на дугах задаются условия. Каждая вершина может иметь не более одного входа. Пути движения по дереву с верхнего уровня на самые нижние определяют логические правила в виде цепочек конъюнкций.

[1] Работа выполняется при поддержке Совета по грантам Президента РФ № МК-3702.2011.9.

На рисунке 1 приведен пример такого дерева решений, и соответствующий логический вывод, где $\theta_1, \theta_2, \theta_3$ - предикаты, x, y, z – переменные, α, β, χ - константы.

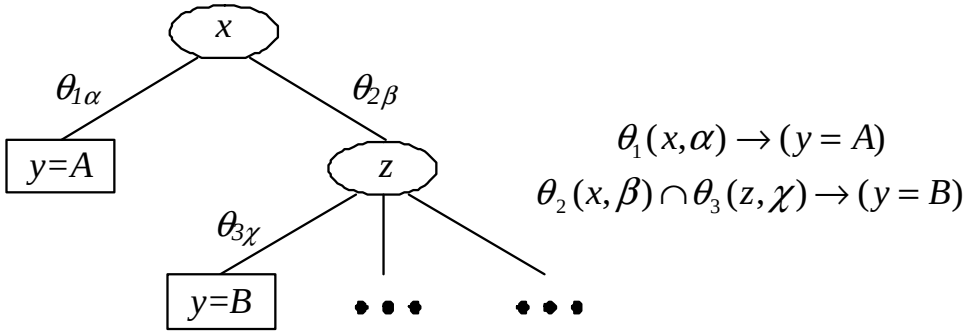


Рис. 1. Пример дерева решений

Правила, выражающие закономерности, формулируются в виде продукций: «ЕСЛИ *A* ТО *B*» или в случае множества условий: «ЕСЛИ (условие 1) ∧ (условие 2) ∧ … ∧ (условие N) ТО (Значение вершины вывода)».

2. Подходы к построению деревьев решений

Построение деревьев решений обычно осуществляется на основе экспертных оценок или с использованием алгоритмов обработки примеров (CLS, ID3 (Interactive Dichotomizer), C4.5, CART (classification and regression trees) и др.). Каждый из этих подходов имеет свои особенности и может использоваться для решения конкретных задач.

2.1. Построение деревьев решений на основе экспертных оценок

Данный подход свойственен экспертным системам, которые ориентированы на обработку данных с помощью некоторых правил вывода, которые предполагается извлекать у экспертов в той или иной области знаний. Такая экспертная система реализует цепочку рассуждений, имитирующую анализ ситуации экспертом-человеком. Деревья решений в данном случае позволяют хранить информацию о данных в компактной форме.

2.2. Построение деревьев решений с использованием алгоритмов обработки примеров

Общий принцип построения деревьев решений, заключается в рекурсивном разбиении объектов из обучающей выборки на подмножества, содержащие объекты, относящиеся к одному классу. Относительно обучающей выборки T и множества классов C возможно три ситуации:

- множество T содержит один или более объектов, относящихся к одному классу C_j. Тогда дерево решений для T – это лист, определяющий этот класс.
- Множество T не содержит ни одного объекта (пустое множество). Тогда это снова лист, и класс, ассоциированный с листом, выбирается из другого множества, отличного от T, например, из множества ассоциированного с родителем
- Множество T содержит объекты, принадлежащие к разным классам. В этом случае следует разбить множество T на некоторые подмножества. Для этого выбирается одна из независимых переменных X, имеющая два, и более различных значений $X_1,\ldots,X_l$. Множество T разбивается на подмножества

$T_1,\ldots,T_l$, где каждое подмножество T_i содержит все объекты, имеющие значения X_i для заданного признака. Эта процедура будет продолжаться рекурсивно до тех пор, пока в конечном множестве не окажутся объекты одного класса

Общее правило для выбора переменной можно сформулировать следующим образом: выбранная переменная должна разбить множество так, чтобы получаемые в итоге подмножества состояли из объектов, принадлежащих к одному классу, или были максимально приближены к этому, то есть, чтобы количество объектов из других классов в каждом из этих подмножеств было минимально. В данном случае критерием может служить количество взаимной информации между множеством значений рассматриваемой независимой переменной и набором целевых классов. Лучшей для разделения считается переменная X, которая позволяет максимизировать информацию о классах $Gain(X)$. Ее поиск осуществляется по формуле вычисления количества информации:

$Gain(X) = Info(T) - Info_X(T)$

Здесь $Info(T)$ - это количество информации, необходимое для идентификации очередного примера из множества T и отнесения его к определенному классу из C, $Info_X(T)$-аналогичное значение, но после разбиения множества T по X. В терминах энтропии данное выражение примет вид:

$I(T \mid X) = H(T) - H(T \mid X)$

где $H(T)$ - энтропия множества T, $H(T \mid X)$ - средняя условная энтропия множества T при известном множестве X. Значения этих величин вычисляются по формулам:

$$H(T) = -\sum_j P(C_j) \cdot \log_2 P(C_j),$$

$$H(T \mid X) = \sum_i P(X_i) \cdot H(T \mid X_i).$$

$P(C_j)$ - вероятность того, что случайно выбранный пример из множества T будет принадлежать к классу C_j. Если $|T|$ - общее число примеров множества T, а количество примеров из T, относящихся к C_j, обозначить как $freq(C_j, T)$ то

$$P(C_j) = \frac{freq(C_j, T)}{|T|}.$$

Тогда формула для вычисления количества информации для идентификации примеров из T примет вид:

$$Info(T) = -\sum_j \frac{freq(C_j, T)}{|T|} \cdot \log_2 \left(\frac{freq(C_j, T)}{|T|} \right).$$

Рассмотрим выражение для условной энтропии $H(T \mid X)$. $P(X_i)$ - вероятность выбора примера из подмножества T_i, которое содержит объекты, имеющие значения X_i для заданного признака X. $H(T \mid X_i)$ - условная энтропия множества T, при условии, что для X выбрано значение X_i. Тогда для $Info_X(T)$ получаем:

$$Info_X(T) = \sum_i \frac{|T_i|}{|T|} \cdot Info(T_i)$$

Критерий $Gain(X)$ считается для всех независимых переменных X и выбирается та, которая максимизирует данное выражение. Этот признак будет являться проверкой в текущем узле дерева, и дальнейшее движение будет производиться в зависимости от полученного ответа. Такие же рассуждения можно применить к полученным подмножествам $T_1,\ldots,T_l$ и продолжить рекурсивно процесс построения дерева, до тех пор, пока в узле не окажутся примеры из одного класса.

Использование данной методики основано на построении дерева решений сверху вниз. Большинство таких процедур являются «жадными алгоритмами». Это значит, что если одна переменная была выбрана, и по ней было проведено разбиение на подмножества, то алгоритм не может вернуться назад и выбрать другую переменную, которая давала бы лучшее разбиение.

Другой проблемой является проблема останова. Обычно используются следующие правила:

- Использование статистических методов для оценки целесообразности дальнейшего разбиения, так называемая ранняя остановка
- Ограничение глубины дерева

Возникает необходимость осуществлять принятие решений о выборе той или иной переменной для дальнейшего разбиения множества, иметь возможность оценить целесообразность дальнейшего разбиения, а также определить наиболее оптимальное и перспективное дерево кандидат.

3. Использование деревьев решений для интеллектуального анализа данных

Методы выделения закономерностей с помощью деревьев решений позволяют находить такие связи, которые заключены не только в отдельных признаках, но и в сочетании признаков, что во многих случаях дает этим методам значительное преимущество по сравнению с классическими методами многомерного анализа.

Однако данный подход и не лишен недостатков, т.к. в задаче поиска логических закономерностей они не способны находить наиболее полные и точные правила в данных и реализуют принцип последовательного просмотра признаков, формируют лишь фрагменты закономерностей. При больших объемах многомерных данных алгоритмы построения деревьев могут выдавать очень сложную структуру, которая имеет много узлов и ветвей. Такие деревья бывает трудно анализировать, т.к. происходит разбиение обучающего множества на большое количество подмножеств, состоящих из малого количества объектов. Тогда как гораздо предпочтительнее иметь дерево, состоящее из малого количества узлов, которым соответствует большое количество объектов из обучающей выборки. Для решения данной проблемы часто применяются алгоритмы отсечения ветвей, но они не всегда могут привести к желаемому результату.

С другой стороны деревья решений являются основным подходом, применимым в системах поддержки принятия решений, что позволяет использовать их для решения задачи выбора архитектуры и вычисления параметров нейросетевых структур в зависимости от решаемой задачи и имеющихся исходных данных [7]. Построение таких деревьев решений осуществляется на основе экспертных оценок и правил, характеризующих объекты предметной области. Это влечет за собой необходимость проведения анализа различных нейросетевых парадигм, направленного на формулировку правил построения и критериев применимости.

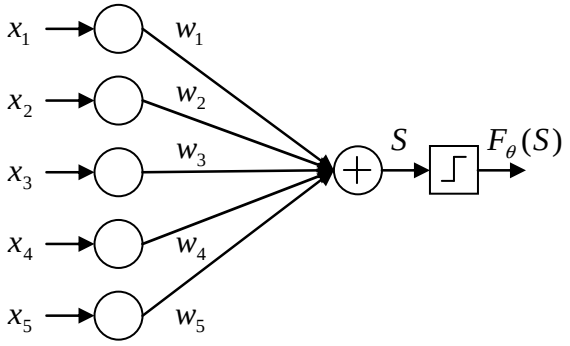
Еще одним направлением использование деревьев решений для интеллектуального анализа данных является их применение для извлечения правил из нейронных сетей. Совместное использование нейросетевых технологий с методами логического вывода способно улучшить понимание структуры изучаемого явления за счет предоставления результата, полученного в ходе обучения нейронной сети, в виде иерархической, последовательной структуры правил типа «если-то».

4. Извлечение правил из обученных нейронных сетей.

Особенностью алгоритмов и методов, применимых при интеллектуальном анализе данных, является отсутствие ограничительных рамок априорных предположений о структуре выборки и виде распределений значений анализируемых показателей, чему наилучшим образом соответствует использование подхода, основанного на нейросетевых технологиях обработки данных. Это обусловлено способностью нейронных сетей к моделированию нелинейных процессов, работе с зашумленными данными, адаптивностью (обучение и самообучение), способностью обобщать и извлекать существенные особенности из поступающей информации. Однако, существует и ряд сложностей использования данного подхода. Например, в случае использования сетей класса многослойный персептрон возможно возникновение проблем с интерпретацией полученного результата и его предпосылок. Нейросеть, по сути, выступает «черным ящиком», на вход которого подаются исходные данные и на выходе получается некоторый результат, однако обоснования, почему было принято именно такое решение, не предоставляется. Правила содержатся в весовых коэффициентах, функциях активации и связях между нейронами, но обычно их структура слишком сложна для восприятия. Более того, в многослойной сети, эти параметры могут представлять собой нелинейные, немонотонной отношения между входными и целевыми значениями. Таким образом, как правило, не представляется возможным отделить влияние определенного признака на целевое значение, потому что этот эффект может быть опосредован значениями других параметров.

Пусть задача состоит в классификации некоторого набора данных с помощью персептрона и последующего анализа полученной сети с целью нахождения классифицирующих правил, характеризующий каждый из классов.

Сначала рассмотрим данную задачу на примере однослойного персептрона, в котором пять булевых нейронов на входе и один на выходе. Данная сеть может быть в точности интерпретирована конечным числом правил «если-то», так как для нее определено конечное число возможных входных векторов.



$$S = \sum_i w_i \cdot x_i$$

$$y = F_\theta(S) = \begin{cases} 1 & if\ S > \theta \\ 0 & else \end{cases}$$

Рис. 2. Однослойный персептрон с пятью булевыми входами и одним выходом.

Пусть веса принимают значения: $w_1 = 6$, $w_2 = 4$, $w_3 = 4$, $w_4 = 0$, $w_5 = -4$, а порог $\theta = 9$. В этом случае из сети можно извлечь следующий набор правил:

$x_1 \wedge x_2 \wedge x_3 \to y$, $x_1 \wedge x_2 \wedge \neg x_5 \to y$, $x_1 \wedge x_3 \wedge \neg x_5 \to y$.

Таким образом, процедура принятия решений заключается в предсказании значения $y = true$, если активация выходного нейрона равна 1, и $y = false$, когда активация равна 0.

Если для выходного нейрона вместо пороговой функции активации использовать логистическую функцию (сигмоиду), тогда решение $y = true$ будет приниматься в случае, когда значение активации нейрона превышает определенное значения, например 0.5. При решении задачи классификации возможно использование отдельного выходного нейрона для каждого класса. В этом случае решение принимается в пользу нейрона с наибольшей активацией. Таким образом, полученное правило должно характеризовать набор входных параметров, при которых обученная нейросеть в сочетании с процедурой вывода предсказывает появление определенного класса.

Вообще говоря, можно выделить два подхода к извлечению правил из многослойных нейронных сетей. Первый подход заключается в извлечении набора глобальных правил, которые характеризуют классы на выходе непосредственно через значения входных параметров. Альтернативой является извлечение локальных правил, разделяя многослойную сеть на совокупность однослойных сетей. Каждое извлекаемое локальное правило характеризует отдельный скрытый или выходной нейрон с учетом элементов, которые имеют с ним взвешенные соединения. Затем правила объединяются в набор, который определяет поведение всей сети в целом.

Рассмотрим задачу извлечения правил в более общем виде.

Пусть X обозначает набор из n свойств $X_1, X_2, ..., X_n$, а $\{x_i\}$- множество возможных значений, которое может принимать свойство X_i. Обозначим через C множество классов $c_1, c_2, ..., c_m$. Для обучающей выборки известны ассоциированные пары векторов входных и выходных значений $(x_1, ..., x_n, c_j)$, где $c_j \in C$.

Одним из алгоритмов извлечения правил из нейронных сетей, обученных решению задачи классификации, является метод NeuroRule [3]. Данный алгоритм включает три основных этапа:

Этап 1. Обучение нейронной сети;

Этап 2. Прореживание нейронной сети;

Этап 3. Извлечение правил.

Однако, данный алгоритм устанавливает довольно жесткие ограничения на архитектуру нейросети, числу элементов, связей и виду функций активации. Так для промежуточных нейронов используется гиперболический тангенс и их состояния изменяются в интервале [-1,1], а для выходных нейронов применяется функция Ферми с интервалом состояний [0,1].

К недостаткам большинства алгоритмов извлечения правил можно отнести отсутствие универсальности и масштабируемости. В связи с этим, наибольший интерес представляет алгоритм TREPAN [4], который лишен этих недостатков и не предъявляет никаких требований к архитектуре сети, входным и выходным значениям, алгоритму обучения и т.д. Данный подход осуществляет построение дерева решений на основе знаний, заложенных в обученную нейросеть, причем достаточно того, что сеть является неким «черным ящиком» или «оракулом»,

которому можно задавать вопросы и получать от него ответы. Более того, алгоритм является достаточно универсальным и может применяться к широкому кругу других обученных классификаторов. Он также хорошо масштабируется и не чувствителен к размерности пространства входных признаков и размеру сети.

Алгоритм построения дерева решения, аппроксимирующего работу обученной нейронной сети состоит из двух этапов.

Предварительный этап:

1. Построить и обучить нейронную сеть, которая в дальнейшем будет выступать в роли «Оракула».
2. Инициализировать корень дерева *R* в виде листа.
3. Использовать все обучающее множество примеров *S* для конструирования модели M_R распределения входных векторов, достигающих узла *R*.

   Вычислить значение *q = max(0, minSamples - |S|)*, где *minSamples* – минимальное число обучающих примеров, используемое в каждом узле дерева, *S* – текущая обучающая выборка (|*S*|-объем обучающей выборки). Таким образом, *q* – количество дополнительных примеров, которые необходимо сгенерировать.
4. На основе оценки распределения признаков из *S*, случайным образом, генерируются *q* новых обучающих примеров.

   $query_R$ - множество из q примеров, генерируемых моделью M_R.
5. Использовать нейронную сеть «Оракула» для классификации, как новых $query_R$, так и старых примеров из множества *S* к тому или иному классу. Для каждого вектора признаков $x \in (S \cup query_R)$ выставить метку класса $x = Oracle(x)$.
6. Инициализировать очередь *Queue*, поместив в нее набор $\langle R, S, query_R, \{empty\_constr\}\rangle$.

Основной этап:

7. Взять очередной набор $\langle N, S_N, query_N, constr_N\rangle$ из начала очереди *Queue*, где N - узел дерева, S_N - обучающая выборка в узле N, $constr_N$ - набор ограничений на определенные признаки обучающих примеров для достижения узла N.
8. Использовать $F, S_N, query_N$ для конструирования в узле N разветвления T.

   Здесь F - функция, оценивающая узел *N*. Она имеет вид $F(N) = R(N) \cdot (1 - f(N))$, где $R(N)$ - вероятность достижения узла *N* примером, а $f(N)$ - оценка правильности обработки этих примеров деревом. Таким образом, выбирается наилучший узел, разветвление которого оказывает наибольшее влияние на точность классификации генерируемого дерева.

   Разделение примеров, достигающих данный внутренний узел дерева, осуществляется в зависимости от $m - of - n$ теста. Такой тест считается пройденным, когда выполняются, по меньшей мере, m из n условий.

   С другой стороны, возможно расщепление множества *S* как в обычном алгоритме построения дерева решений.
9. Для каждой дуги *t* разветвления T создать узлы следующего поколения:
   a. Создать C - новый дочерний узел по отношению к N.
   b. $constr_C = constr_N \cup \{T = t\}$ - добавить ограничение с дуги *t*.

c. Сформировать S_C = примеры из множества S_N, которые удовлетворяют условию на дуге *t*.
d. Сконструировать модель M_C распределения примеров, достигающих узла C.

   Подсчитать значение $q = max(0, minSamples - |S_C|)$, т.е. количество примеров, которые необходимо сгенерировать
e. На основе оценки распределения признаков из S_C и значения ограничений $constr_C$, случайным образом, сгенерировать q новых обучающих примеров. $query_C$ – множество из q примеров, сгенерированных моделью M_C и ограничением $constr_C$
f. Использовать нейронную сеть «Оракула» для классификации новых примеров $x \in query_C$ и выставить метку класса $x = Oracle(x)$.
g. Изначально предполагается, что узел C является листом. Использовать S_C и $query_C$ для определения метки класса для C.
h. Проверить необходимость дальнейшего расщепления узла C. Если локальный критерий остановки не удовлетворен, то поместить набор $< C, S_C, query_C, constr_C >$ в очередь $Queue$. Локальным критерием в данном случае выступает величина, которая характеризует вероятность, что в данном узле встречаются экземпляры одного класса.

10. Если очередь $Queue$ не пуста и не выполнен глобальный критерий остановки, то перейти к шагу 7, иначе вернуть дерево с корнем R.

В качестве глобального критерия завершения алгоритма используется максимальный размер дерева и общую оценку качества классификации примеров деревом.

Основное преимуществом данного подхода заключается в обобщающей способности искусственных нейронных сетей, которая позволяет получать более простые деревья решений. К тому же, использование такого «Оракула» позволяет компенсировать недостаток данных, наблюдающийся при построении деревьев решений на нижних уровнях.

Таким образом, возможно извлечение структурированных знаний не только из чрезвычайно упрощенных нейронных сетей, но и из произвольных классификаторов, что делает возможным применение данного алгоритма в широком круге практических задач.

Список литературы

1. Комарцова Л.Г., Максимов А.В. Нейрокомпьютеры. – М.: Изд-во МГТУ, 2004
2. Рассел С., Норвиг П. Искусственный интеллект: современный подход – М.: Издательский дом «Вильямс», 2006
3. Ежов А., Шумский С., Нейрокомпьютинг и его применение в экономике и бизнесе,1998.
4. Craven, M. W. & Shavlik, J. W. (1996). Extracting tree-structured representations of trained networks. In Touretzky, D., Mozer, M., & Hasselmo, M., editors, Advances in Neural Information Processing Systems (volume 8). MIT Press, Cambridge, MA.

5. Джонс М.Т. Программирование искусственного интеллекта в приложениях. – М.: ДМК Пресс, 2004
6. Евдокимов И.А., Солодовников В.И. Совместное использование нейросетевых технологий и деревьев решений для анализа информации, содержащейся в приложениях баз данных // Материалы тринадцатого научно–практического семинара «Новые информационные технологии в автоматизированных системах», Москва, 2010. - С. 13-18.
7. Солодовников И.В., Солодовников В.И. Подход к созданию подсистемы автоматизации проектирования нейросетевых структур обработки данных с использованием деревьев решений // Информационные технологии в проектировании и производстве - 2006 - № 2, стр. 62-66