РЕАЛИЗАЦИЯ И РАСПАРАЛЛЕЛИВАНИЕ АЛГОРИТМА ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ, ОСНОВАННОГО НА ДЕРЕВЬЯХ РЕШЕНИЙ

Е.С. Лепшова, В.А. Биллиг

Тверской госуниверситет

Исследуется алгоритм, основанный на деревьях решений. Обсуждается его использование для решения задач классификации и интеллектуального анализа данных с выявлением скрытых знаний. Целью работы является повышение эффективности, быстродействия и адаптивности метода формирования дерева решений, основанного на алгоритме C4-5. Также рассматриваются возможности модификации алгоритма с использованием распаралеливания в процессе построения дерева.

Стремительное развитие информационных технологий, в частности прогресс в методах сбора, хранения и обработки данных, позволил многим организациям собирать огромные массивы данных, которые необходимо анализировать. Объемы этих данных настолько велики, что возможностей экспертов уже не хватает.

На сегодняшний день интенсивно развивается направление, связанное с интеллектуализацией методов обработки и анализа данных. Интеллектуальные системы анализа данных (ИСАД) призваны минимизировать усилия лица, принимающего решения (ЛПР), в процессе анализа данных, а также в настройке алгоритмов анализа. Многие ИСАД позволяют не только решать классические задачи принятия решения, но и способны выявлять причинно-следственные связи, скрытые закономерности в системе, подвергаемой анализу. Точность результатов исследования прямо пропорциональна объему тестовой выборки и обратно пропорциональна временным затратам. Суммарное время проведения исследований оказывается просто неприемлемым. Поэтому основной целью данной работы является актуальная научно-техническая задача реализации алгоритма и его распараллеливание.

Обоснование выбранного метода. Деревья решений

Впервые деревья решений были предложены Ховилендом и Хантом (Hoveland, Hunt) в конце 50-х годов XX века. Самая ранняя и известная работа Ханта и др., в которой излагается суть деревьев решений – “Эксперименты в индукции” (“Experiments in Induction”) – была опубликована в 1966 году.

Дерево решений – это способ представления правил в иерархической, последовательной структуре. Основа такой структуры – ответы “Да” или “Нет” в случае бинарных деревьев (алгоритм CART), или на ряд вопросов, определимых динамически в случае деревьев с произвольным числом потомков (алгоритм C4-5).

Чтобы решить задачу, т.е. принять решение, требуется ответить на ряд вопросов, которые находятся в узлах дерева, начиная с его корня. При положительном ответе на вопрос осуществляется переход к левой части дерева, называемой левой ветвью, при отрицательном – к правой части дерева. Таким образом, внутренний узел дерева является узлом проверки определенного условия. Далее идет следующий вопрос и т.д., пока не будет достигнут конечный узел дерева, являющийся узлом решения, т.е. листом дерева. Для каждой конкретной задачи существуют свои типы конечного узла в количестве n+1.

Модель, представленная в виде дерева решений, является интуитивной и упрощает понимание решаемой задачи. Это свойство деревьев решений не только важно при отнесении к определенному классу нового объекта, но и полезно при интерпретации модели классификации в целом. Дерево решений позволяет понять и объяснить, почему конкретный объект относится к тому или иному классу. Деревья решений дают возможность извлекать правила из базы данных на естественном языке.

Деревья решений позволяют создавать классификационные модели в тех областях, где аналитику достаточно сложно формализовать знания. Алгоритм конструирования дерева решений не требует от пользователя выбора входных атрибутов (независимых переменных). На вход алгоритма можно подавать все существующие атрибуты, алгоритм сам выберет наиболее значимые среди них, и только они будут использованы для построения дерева.

Точность моделей, созданных при помощи деревьев решений, сопоставима с другими методами построения классификационных моделей (статистические методы, нейронные сети). Разработан ряд масштабируемых алгоритмов, которые могут быть использованы для построения деревьев решения на сверхбольших базах данных.

Большинство алгоритмов конструирования деревьев решений имеют возможность специальной обработки пропущенных значений. Многие классические статистические методы, при помощи которых решаются задачи классификации, могут работать только с числовыми данными, в то время как деревья решений работают и с числовыми, и с категориальными типами данных.

Многие статистические методы являются параметрическими, и пользователь должен заранее владеть определенной информацией, например, знать вид модели, иметь гипотезу о виде зависимости между переменными, предполагать, какой вид распределения имеют данные. Деревья решений, в отличие от таких методов, строят непараметрические модели. Таким образом, деревья решений способны решать такие задачи Data Mining (раскопки данных), в которых отсутствует априорная информация о виде зависимости между исследуемыми данными.

На сегодняшний день существует большое число алгоритмов, реализующих деревья решений: CART, C4.5, CHAID, CN2, NewId, ITrule и другие. Деревья с произвольным числом потомков, как одна из разновидностей деревьев решений, благодаря своим свойствам являются более универсальным методом решения задач классификации. Именно такой алгоритм выбран в качестве области исследования.

Алгоритм C4.5

Алгоритм C4.5 моделирует дерево решений с неограниченным количеством ветвей у узла. Данный алгоритм может работать только с дискретным зависимым атрибутом и поэтому может решать только задачи классификации. C4.5 считается одним из самых известных и широко используемых алгоритмов построения деревьев классификации.

Для работы алгоритма C4.5 необходимо соблюдение следующих требований:

Каждая запись набора данных должна быть ассоциирована с одним из предопределенных классов, т.е. один из атрибутов набора данных должен являться меткой класса. Классы должны быть дискретными. Каждый пример должен однозначно относиться к одному из классов. Количество классов должно быть значительно меньше количества записей в исследуемом наборе данных. Алгоритм C4.5 медленно работает на сверхбольших и зашумленных наборах данных.

Псевдокод Algorithm C4-5

1. Input: database D
2. Tree = {}
3. If D is pure OR other stopping criteria Then

4. Terminate
5. End If
6. For all attribute a ϵ D do
7. Entropy for root
8. End For
9. abest = Best attribute entropy
10. Tree = Create a node then test a best in the root
11. Dv = sud_datasets from D
12. For all Dv do
13. Tree v= C4.5(Dv)
14. End for
15. Return Tree

Математическое обеспечение алгоритма

Задача алгоритма заключается в построении иерархической классификационной модели в виде дерева из множества примеров *T*. Процесс построения дерева будет происходить сверху вниз. Сначала создается корень дерева, затем потомки корня и т.д.

На первом шаге дерево пустое (имеется только корень) и исходное множество *T* (ассоциированное с корнем). Требуется разбить исходное множество на подмножества. Для этого выбрать один из атрибутов в качестве проверки. Тогда в результате разбиения получаются *n* (по числу значений атрибута) подмножеств и, соответственно, создаются n потомков корня, каждому из которых поставлено в соответствие свое подмножество, полученное при разбиении множества *T*. Затем эта процедура рекурсивно применяется ко всем подмножествам (потомкам корня) и т.д.

Очевидно, что из m (по числу атрибутов) возможных вариантов, требуется выбрать самый подходящий. Некоторые алгоритмы исключают повторное использование атрибута при построении дерева. Любой из атрибутов можно использовать неограниченное количество раз при построении дерева.

Пусть мы имеем проверку *X* (в качестве проверки может быть выбран любой атрибут), которая принимает n значений A_1, A_2 … A_n. Тогда разбиение *T* по проверке *X* даст нам подмножества T_1, T_2 … T_n, при *X*, равном соответственно A_1, A_2 … A_n. Единственная доступная информация – то, каким образом классы распределены во множестве *T* и его подмножествах, получаемых при разбиении по *X*. Именно это играет роль при определении критерия.

Пусть $freq(C_i, S)$ – количество примеров из некоторого множества *S*, относящихся к одному и тому же классу C_j. Тогда вероятность того, что случайно выбранный пример из множества *S* будет принадлежать к классу C_j, такова:

$$P = \frac{freq(C_i, S)}{|S|}.$$

Согласно теории информации, количество содержащейся в сообщении информации зависит от ее вероятности

$$log_2\left(\frac{1}{P}\right). \quad (1)$$

Выражение дает оценку среднего количества информации, необходимого для определения класса примера из множества *T*. В терминологии теории информации выражение

$$Info(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} * log_2(\frac{freq(C_j, T)}{|T|}) \quad (2)$$

называется энтропией множества T.

Ту же оценку, но только уже после разбиения множества T по X, дает выражение:

$$Info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} * Info(T_i). \quad (3)$$

Тогда критерием для выбора атрибута будет являться формула:

$$Gain(X) = Info(T) - Info_x(T). \quad (4)$$

Критерий (4) считается для всех атрибутов. Выбирается атрибут, максимизирующий данное выражение. В случае с числовыми атрибутами следует выбрать порог, с которым должны сравниваться все значения атрибута. Пусть числовой атрибут имеет конечное число значений $\{v_1, v_2 \dots v_n\}$, предварительно отсортированное. Тогда любое значение, лежащее между v_i и v_{i+1}, делит все примеры на два множества: те, которые лежат слева от этого значения $\{v_1, v_2 \dots v_i\}$, и те, что справа $\{v_{i+1}, v_{i+2} \dots v_n\}$. В качестве порога можно выбрать среднее между значениями v_i и v_{i+1}:

$$TH_i = \frac{v_i + v_{i+1}}{2}.$$

Оценка сложности алгоритма решения задачи. Параллелизм

Алгоритм решающего дерева имеет сложность, линейную по длине выборки. Если множество предикатов настолько богато, что на шаге всегда находится предикат, разбивающий выборку S на непустые подмножества S_0 и S_1, то алгоритм строит бинарное решающее дерево, безошибочно классифицирующее выборку X. Параллелизм (Concurrency) – это свойство систем, при котором несколько вычислений выполняются одновременно, и при этом, возможно, взаимодействуют друг с другом. Вычисления могут выполняться на нескольких ядрах одного чипа с вытесняющим разделением времени потоков вычислений на одном процессоре, либо выполняться на физически отдельных процессорах.

Алгоритм деревьев решений построен на индукции, является примером поглощающего «жадного» алгоритма и на сегодняшний день наиболее распространенной стратегией деревьев решений для данных, но это не только стратегия. Как отмечалось ранее, общая схема построения дерева принятия решений выглядит следующим образом.

Выбирается очередной атрибут, помещается в корень. Затем для всех его значений рекурсивно строится дерево в этом потомке. Начиная со второго шага, алгоритм может быть распараллелен (рис. 1).

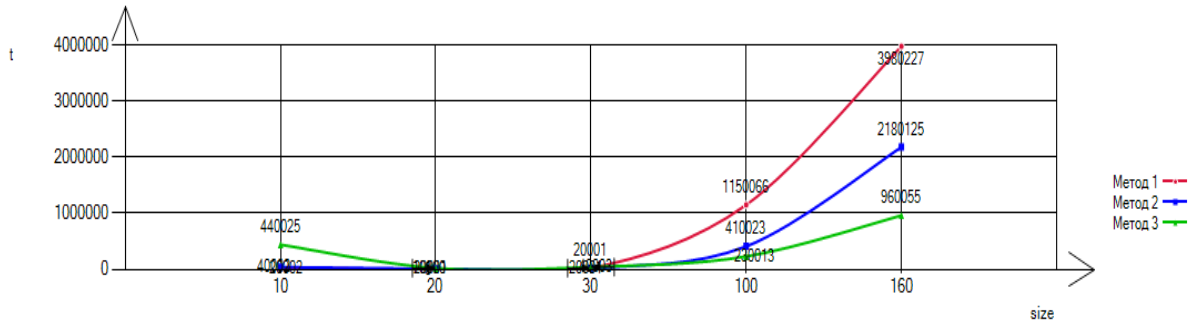

Рис. 1. Зависимость времени работы алгоритма от объема обучающей выборки (t, тик / size)

Метод 1 – Линейные вычисления (t, тик / size, количество строк шт. в базе)

Метод 2 – Распараллеливание в один поток (блокирование внешних потоков) (t, тик / size, количество строк шт. в базе)

Метод 3 – Полностью распараллеленный алгоритм на 2 потока (t, тик / size, количество строк шт. в базе).

Анализ полученных результатов распараллеливания

Из рис. 1 видно, что распараллеливание алгоритма деревьев решений имеет смысл проводить только на больших объемах данных. Для случаев, где выборка данных меньше 50 элементов, распараллеливание избыточно, линейный вариант алгоритма работает быстрее. Это связано с затратами приложения на процесс создания и работы с потоками. Распараллеливание в 1 поток значительно проще реализуется, однако дает результат, превосходящий линейный алгоритм, а значит, также является применимым для реализации данной задачи на практике. В результате, уже при объеме базы более ста строк алгоритм в распараллеленной версии позволяет экономить минуты. С увеличением базы этот результат будет подтверждаться, а разрыв во времени работы методов увеличиваться.

Выводы

Деревья решений – иерархическое, гибкое средство предсказания принадлежности объектов к определенному классу или прогнозирования значений числовых переменных. Среди большого числа разнообразных алгоритмов классификации деревья решений отличаются тем, что позволяют проводить интеллектуальный анализ данных, т.е. представлять в явном и доступном для понимания и интерпретации экспертами причинно-следственные связи в процессе отнесения объекта классификации к тому или иному классу.

Задача формирования дерева решений представляет собой оптимизационную задачу. Качество работы рассмотренного метода деревьев решений зависит как от выбора алгоритма, так и от набора исследуемых данных. Несмотря на все преимущества данного метода, следует помнить, что для того чтобы построить качественную модель, необходимо понимать природу взаимосвязи между зависимыми и независимыми переменными и подготовить достаточный набор данных.

Сравнивая алгоритм C4-5 с другими методами, можно отметить, что алгоритм деревьев решений превосходит по эффективности современные алгоритмы классификации. Кроме того, он является инструментом интеллектуального анализа данных для обнаружения скрытых знаний. В ходе апробации алгоритма на практических задачах классификации получены деревья решающих правил, представляющие интерес для специалистов в соответствующих проблемных областях.

Литература

1. Чубуков И.А. Data Mining. Учебный курс – [http://www.intuit.ru/department/database/datamining/].

2. Биллиг В.А. Основы программирования на C#. Бином. Лаборатория знаний, Интернет-университет информационных технологий. Серия: Основы информационных технологий, 2009. – [http://www.intuit.ru/department/se/prcsharp08/].

3. Шилдт Г. Теория и практика C#. СПб.: BHV-Санкт-Петербург, 1996. – 312 с.

4. Breiman, L., Friedman, J., Olshen, R., and Stone, C. Classification and Regression Trees. Wadsworth International Group, Belmont, CA. 1984.

5. Joydeep Ghosh and Alexander Liu, K-Means, The Top Ten Algorithms in Data Mining.

6. Dan Steinberg, CART: Classification and Regression Trees, The Top Ten Algorithms in Data Mining.