ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПАРАЛЛЕЛЬНОГО АЛГОРИТМА ГРАДИЕНТНОГО БУСТИНГА ДЕРЕВЬЕВ РЕШЕНИЙ

П.Н. Дружков

Нижегородский госуниверситет им. Н.И. Лобачевского

Рассматривается задача обучения с учителем, приводится краткое описание параллельного алгоритма обучения дерева решений, а также алгоритма градиентного бустинга деревьев решений, применимых для случая распределенного хранения данных, предназначенных, в первую очередь, для решения больших задач машинного обучения. Приводятся результаты вычислительных экспериментов с помощью созданной программной реализации рассматриваемых алгоритмов.

Введение

Обучение с учителем, или обучение по прецедентам, заключается в восстановлении неизвестной общей зависимости между набором входных переменных и некоторым целевым (выходным) показателем на основании конечного набора пар вида «вход–выход». Чем больше прецедентов доступно на этапе обучения, тем большим объемом информации о восстанавливаемой зависимости располагает алгоритм, и, следовательно, может построить более точную модель зависимости. Таким образом, сама возможность применения алгоритма машинного обучения в случае наличия большой обучающей выборки может повысить качество решения задачи. Данная тенденция поддерживается и с другой стороны. В последнее время появляются все более и более объемные наборы данных, связанные с важными прикладными задачами: от извлечения информации из поисковых Интернет-запросов и Интернет-рекламы [1] до интеллектуального анализа изображений и видео [2]. В связи с этим возникает необходимость модификации существующих алгоритмов или разработки новых, ориентированных именно на обработку больших наборов данных. Данная работа посвящена модификации, позволяющей обрабатывать распределенно хранящиеся данные, одного из наиболее перспективных алгоритмов решения задачи обучения с учителем: градиентного бустинга деревьев решений [3].

1. Постановка задачи

Одной из задач, изучаемой в машинном обучении, является *задача обучения с учителем*. В рамках этой задачи дано некоторое множество *объектов* (пространство признаков) X. Каждому объекту $x \in X$ поставлена в соответствие величина y, называемая *выходом*, *ответом* или *целевой переменной* и принадлежащая множеству допустимых ответов Y. Упорядоченная пара «объект–ответ» (x,y), где $x \in X$, $y \in Y$, называется *прецедентом*. Требуется восстановить зависимость между входом и выходом, основываясь на данных о конечном наборе прецедентов, называемом *обучающей выборкой*: $\{(x_i, y_i) \mid x_i \in X,\ y_i \in Y,\ i = 1, ..., n\}$. Другими словами, задача состоит в построении функции f из некоторого множества K, которая, получив на вход x, предсказала бы значение ответа y как можно точнее. В случае конечного Y говорят о *задаче классификации*, если $Y = \mathbf{R}$ – *задаче восстановления регрессии* [4]. Процесс нахождения f называется *обучением* (*тренировкой*, *настройкой*) модели, процесс определения выхода по некоторому входу с помощью уже построенной модели – *предсказанием*.

2. Описание алгоритма

Одними из наиболее популярных и перспективных на данный момент подходов в машинном обучении являются алгоритмы, основанные на построении деревьев решений и их ансамблей. Дерево решений строит разбиение пространства признаков на непересекающиеся области с помощью рекурсивной процедуры [5]. Фактически каждому узлу дерева соответствует некоторая область пространства признаков и правило, по которому осуществляется ее разделение на две, приписанные к дочерним вершинам. Вид правила разбиения зависит от типа участвующей в нем переменной: для количественного признака s это неравенство вида $x^{(s)} < v^{(s)}$, где $x^{(s)}$ – s-я координата вектора входных переменных x, а $v^{(s)}$ – некоторое пороговое значение; для номинального признака правило имеет вид $x^{(s)} \in P$, где P – подмножество множества возможных значений переменной $x^{(s)}$. Для обучения данной модели используется жадная стратегия максимального уменьшения функции, описывающей неоднородность данных $I(D)$, где D есть область, соответствующая некоторому узлу. В качестве $I(D)$ могут применяться различные функции: ошибка классификации, индекс Джини и т.д.

Вход: $D = \{(x_i, y_i), i = \overline{1,n}\}$ – обучающая выборка.

Выход: Обученное дерево решений T.

T = **Обучить_дерево_решений**(D)

1. Если не выполнен критерий остановки:
   a. Найти оптимальное правило разбиения D на DL и DR, максимизирующее I(D)-(I(D_L)+I(D_R)).
   b. Выполнить разбиение DL.
   c. Выполнить разбиение DR.
2. Иначе
   a. Найти значение, приписанное листу дерева.

Следует отметить, что поиск оптимального разбиения по количественной переменной производится для всех $v^{(s)} \in V^{(s)} = \left\{ \frac{\left(\hat{x}_j^{(s)} + \hat{x}_{j+1}^{(s)}\right)}{2} : \hat{x}_j^{(s)} \le \hat{x}_{j+1}^{(s)}, j = \overline{1, n-1} \right\}$, где $\hat{x}^{(s)}$ – упорядоченные значения s-й переменной. В данной работе рассматривается алгоритм построения регрессионного дерева и квадратичная функция $I(D) = \sum_{i:x_i \in \mathrm{D}} (y_i - \bar{y})^2$. В данном случае выражение для уменьшения функции неоднородности упрощается до $\frac{\left(\sum_{i:x_i \in D_L} y_i\right)^2}{|D_L|} + \frac{\left(\sum_{i:x_i \in D_R} y_i\right)^2}{|D_R|}$, а каждому листу приписывается значение, равное среднему арифметическому значений целевой переменной попавших в него прецедентов.

Пусть обучающая выборка разделена на несколько частей $R_1, R_2, \ldots, R_T$ и каждая часть обрабатывается отдельным процессом независимо от остальных, в то время как для синхронизации их работы служит один управляющий процесс. Рабочий процесс для каждой переменной s и значения $v^{(s)}$ вычисляет пару $\alpha_t^{(s)} = \left(\sum_{i:x_i \in D \cap R_t} 1, \sum_{i:x_i \in D \cap R_t} y_i\right)$, которая передается главному процессу для выбора глобально-оптимального разбиения в данной вершине дерева. Следует отметить, что поскольку каждому отдельно взятому процессу не доступна вся выборка, вычислить все элементы множества $V^{(s)}$ для количественной переменной s не представляется возможным. В связи с этим предлагается перед процессом обучения вычислить для каждой количественной переменной элементы множества $V^{(s)}$ как квантили уровней $\frac{i}{q}$, $i = \overline{1, q-1}$, для относительно небольшого фиксированного числа q. Вычисление значений в листьях дерева, а именно вычисление среднего арифметического, легко выполняется в случае распределенного хранения данных.

Как уже отмечалось ранее, практический интерес представляет не только обучение одиночных деревьев решений, но и построение моделей в виде их ансамблей. Так, бустинг-алгоритмы последовательно строят базовые модели (в настоящей работе – деревья решений) таким образом, что каждая следующая корректирует ошибки предыдущих, тем самым улучшая качество модели. Одним из наиболее общих методов этого направления является алгоритм градиентного бустинга деревьев решений, который использует аналогию с методом градиентного спуска для построения ансамбля деревьев решений, минимизирующего ошибку на обучающей выборке, заданную дифференцируемой неотрицательной функцией потерь (штрафа) $L(y, y')$. Данный алгоритм представляет собой итерационный процесс, на каждом шаге которого строится одиночное дерево решений, наилучшим образом предсказывающее не исходные значения целевой переменной, а антиградиент функции потерь $-\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F=h_{m-1}}$, зависящий от истинных значений выходного признака и предсказаний модели, построенной к текущей итерации. Следует отметить, что каждое дерево решений обучается с использованием квадратичного штрафа, что не требует сильной модификации алгоритма его построения, за исключением вычисления значений, приписанных листьям уже построенного дерева. Листовые константы бустинг-дерева находятся с учетом функции потерь, используемой в алгоритме градиентного бустинга, из соображений минимизации потерь в листе: $\arg\min_\rho \sum_{i:\, x_i \in \text{leaf}} L(y_i, \rho)$. Следует отметить, что, в отличие от отыскания градиента, вычисление значений в листовых вершинах задача гораздо более сложная, в связи с чем зачастую применяются приближенные решения.

Таким образом, применение алгоритма градиентного бустинга деревьев решений, в случае когда данные расположены распределенно, требует, помимо алгоритма обучения одиночных деревьев решений, также модифицированных методов пересчета значений в листовых вершинах деревьев и вычисления антиградиента на каждой итерации. В общем случае обе эти процедуры зависят от конкретного вида функции потерь.

3. Программная реализация

В соответствии с описанными выше схемами были созданы параллельные программные реализации как алгоритма обучения одиночного дерева решений, так и алгоритма градиентного бустинга деревьев решений. Для последнего были реализованы следующие функции потерь [3] для задач восстановления регрессии: абсолютная, квадратичная, функция Хьюбера – и для классификации: логистическая и кросс-энропия. Как уже отмечалось выше, алгоритм вычисления антиградиента и листовых констант зависит от используемой функции потерь, и даже для реализованного набора целиком осуществить эти операции независимо над каждой частью данных не всегда представляется возможным. В таких случаях приходится пересылать необходимую информацию на управляющий процесс, где целиком производится вычисление необходимых значений. В остальных случаях б*о*льшая часть работы осуществляется параллельно рабочими процессами, после чего результирующее значение вычисляется управляющим процессом.

Для обеспечения параллелизма используется технология MPI.

4. Вычислительный эксперимент

Для оценки эффективности описанной программной реализации был проведен ряд вычислительных экспериментов. Для экспериментов использовался набор данных Million Song Dataset, взятый с репозитория UCI [6], и искусственно сгенерированный набор данных. Все эксперименты проводились на вычислительном кластере МГУ «Ломоносов». Модель градиентного бустинга деревьев решений обучалась с применением

функции потерь Хьюбера, строился ансамбль из 500 деревьев, на каждой итерации использовалось 70% от всей обучающей выборки.

Million Song Dataset представляет собой набор данных для предсказания года создания музыкальной композиции по ее характеристикам. Набор данных состоит из 515345 прецедентов, каждый из которых содержит 90 предикативных числовых переменных и числовую целевую переменную. Для каждой переменной было вычислено по 9 статистических квантилей значений в обучающей выборке. Несмотря на то, что данные предразбиты на обучающую и тестовую части, для оценки эффективности программной реализации параллельного алгоритма обучения модели градиентного бустинга обе выборки использовались для обучения, что позволило увеличить объем обрабатываемых данных. График масштабируемости алгоритма приведен на рис. 1(a). Как видно из приведенных результатов, параллельный алгоритм обеспечивает ускорение, близкое к линейному, или даже выше. Сверхлинейное ускорение в данном случае можно объяснить нелинейным характером трудоемкости последовательного алгоритма, а также более эффективным использованием памяти процессами при уменьшении объема обрабатываемых ими данных.

Для генерации искусственной выборки использовалась функциональная зависимость целевого признака от 100 предикативных $f(x_1, \dots, x_{100}) = \sum_{i=1}^{15} \exp(-2x_i^2) + \sum_{i=16}^{100} x_i$. Также, для повышения размерности пространства, было добавлено 2400 фиктивных (независимых с целевой) переменных. Для формирования обучающей выборки было сгенерировано 50000 точек, независимо и равномерно распределенных в 2500-мерном гиперкубе. Так как все признаки являются числовыми, для использования параллельного алгоритма обучения модели градиентного бустинга деревьев решений было подсчитано по 9 статистических квантилей для каждой переменной. Время работы параллельного алгоритма на разном количестве процессов приведено на рис. 1(b). Также следует отметить, что рассматриваемая параллельная схема и последовательный алгоритм градиентного бустинга не являются полностью эквивалентными из-за сужения множества рассматриваемых разбиений по количественным переменным в деревьях решений. Для сравнения качества решения задачи вся выборка была разбита на 2 части в соотношении 70 к 30 и были обучены бустинг-модели с помощью последовательного [7, 8] и параллельного алгоритмов, после чего была вычислена средняя квадратичная ошибка на тестовой выборке. Для данной задачи ошибка сохранилась на уровне 6.4, а следовательно, можно говорить о сохранении качества решения задачи с помощью параллельной модификации алгоритма.

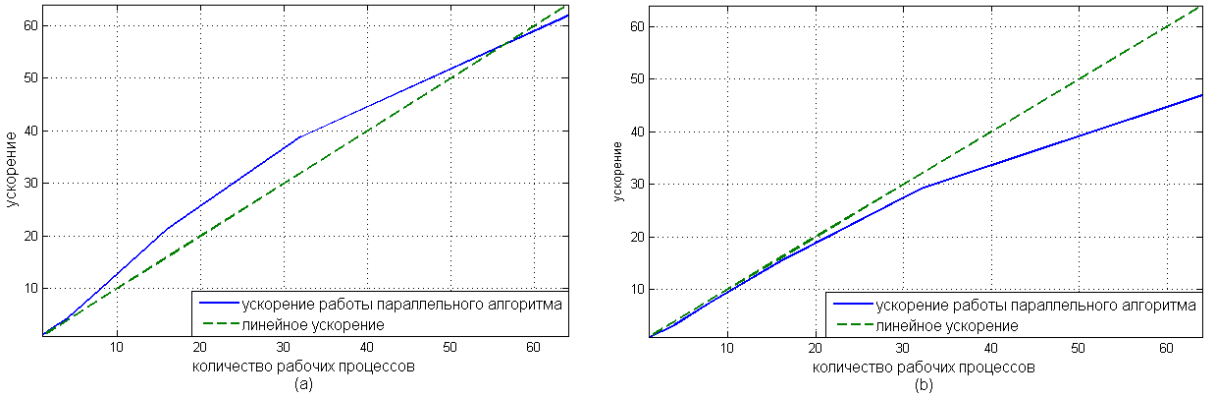

Рис. 1. Ускорение работы параллельного алгоритма обучения модели градиентного бустинга деревьев решений с ростом количества рабочих процессов (a) на данных Million Song Dataset, (b) на искусственных данных

Полученный результат свидетельствует о хорошей масштабируемости даже для относительно небольшой обучающей выборки, что позволяет использовать данный алгоритм не только для больших наборов данных, но также и для параллельного обучения модели градиентного бустинга. Разница в получаемых показателях ускорения для рассмотренных задач обусловлена не только размером выборки (количеством прецедентов), но и количеством предикативных переменных: случай, когда количество прецедентов значительно больше количества переменных, является наиболее подходящим для рассматриваемого алгоритма.

Работа выполнена в рамках программы «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы», государственный контракт № 11.519.11.4015.

Литература

1. Panda B., Herbach J.S., Basu S., Bayardo R.J. PLANET: Massively parallel learning of tree ensembles with MapReduce // Proceedings of the 35th International Conference on Very Large Data Base (VLDB). 2009. P. 1426–1437.
2. Dollar P., Wojek C., Schiele B., Perona P. Pedestrian Detection: A Benchmark // Computer Vision and Pattern Recognition (CVPR'09). 2009.
3. Friedman J. Greedy function approximation: the gradient boosting machine // Annals of Statistics. 2001. V. 29, N. 5. P. 1189–1232.
4. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning. – Springer, 2008.
5. Breiman L., Friedman J.H., Olshen R.A., Stone C.J. Classification and Regression Trees. Wadsworth & Brooks, 1984.
6. UCI Machine Learning Repository – [http://archive.ics.uci.edu/ml].
7. Дружков П.Н., Золотых Н.Ю., Половинкин А.Н. Программная реализация алгоритма градиентного бустинга деревьев решений // Вестник Нижегородского государственного университета им. Н.И. Лобачевского. 2011. № 1. С. 193–200.
8. Druzhkov P. N., Eruhimov V. L., Kozinov E. A., Kustikova V. D., Meyerov I. B., Polovinkin A. N., Zolotykh N. Yu. On some new object detection features in OpenCV Library // Pattern Recognition and Image Analysis. 2011. V. 21, № 2. P. 377–379.