Семинар 3.
ММП, весна 2013
5 марта

Илья Толстихин
iliya.tolstikhin@gmail.com

Темы семинара:

- Композиции;
- Gradient Boosting

Задачка для бустинга — что минимизирует истинные экспоненциальные потери? Бишоп 661 Задача 14.9 Замена квадрата на модуль

1 Gradient Boosting

Рассмотрим способ построения взвешенных композиций базовых функций для решения задачи регрессии $\mathbb{Y} = \mathbb{R}$ с обучающей выборкой X^ℓ:

$$F_M(x) = c + \sum_{m=1}^{M} \alpha_m h_m(x), \quad h_m \in \mathcal{H} = \{h \colon \mathbb{X} \to \mathbb{R}\},\ m = 1, \ldots, M,$$

где c — константа, а $\mathcal{H}$ — базовый класс функций.

Мы рассмотрим широко применяемый на практике метод Gradient Boosting Machines (GBM) [3, 2, 1], который стремится минимизировать математическое ожидание $\mathbb{E}_{x,y} L\big(y, F(y)\big)$ заранее выбранной дифференцируемой функции потерь $L\big(y, F(x)\big)$. Как и рассмотренный нами ранее бустинг, GBM решает эту задачу жадным пошаговым образом: начиная с константного начального приближения

$$F_0(x) \equiv c = \arg\min_{\alpha} \sum_{i=1}^{\ell} L(y_i, \alpha),$$

на каждом t-ом шаге он пытается выбрать очередную базовую функцию h_t с соответствующим ей весом α_t, решая подзадачу

$$(h_t, \alpha_t) = \arg\min_{h_t, \alpha_t} \sum_{i=1}^{\ell} L\big(y_i, F_{t-1}(x_i) + \alpha_t h_t(x_i)\big). \tag{1}$$

Найденная базовая функция h_t добавляется к композиции $F_t(x) = F_{t-1}(x) + \alpha_t h_t(x)$ и итереции повторяются. Проблема в том, что для произвольных функций потерь L и базовых классов $\mathcal{H} = \{h\colon \mathbb{X} \to \mathbb{Y}\}$ задачи (1) могут оказаться очень сложными.

Вспомним, что один из способов замены задачи (1) на более простую предлагался в рамках алгоритма AnyBoost. Метод GBM предлагает еще один способ, который ведет к очень интересной интерпретации всей итерационной процедуры.

Представим, что наша единственная задача — минимизировать среднее значение потерь на обучающей выборке X^ℓ. В этом случае предлагается рассматривать $L\big(y, F(x)\big)$ не как функционал, зависящий от F, а как функцию ℓ аргументов $F(x_1), \ldots, F(x_\ell)$ — ответов функции F на точках обучающей выборки. Таким образом мы переходим от рассмотрения функций $F(x)$ к ℓ-мерным векторам их значений на обучающей выборке.

Теперь задачу (1) можно интерпретировать следующим образом: мы находимся в точке $\boldsymbol{f}_{t-1} = (F_{t-1}(x_1), \ldots, F_{t-1}(x_\ell))$ и хотим сделать шаг $\boldsymbol{\delta}_t$ в новую точку

$$\boldsymbol{f}_t = \boldsymbol{f}_{t-1} + \boldsymbol{\delta}_t = \big(F_{t-1}(x_1) + \boldsymbol{\delta}_t^1, \ldots, F_{t-1}(x_\ell) + \boldsymbol{\delta}_t^\ell\big)\,,$$

так чтобы минимизировать значение функции

$$\mathcal{L}(\boldsymbol{f}) = \sum_{i=1}^{\ell} L\big(y_i, \boldsymbol{f}^i\big) = \sum_{i=1}^{\ell} L\big(y_i, F(x_i)\big)$$

в этой точке. Для этого мы можем найти градиент $\boldsymbol{g}_t = \nabla\mathcal{L}(\boldsymbol{f}_{t-1})$ функции $\mathcal{L}(\boldsymbol{f})$ в точке $\boldsymbol{f} = \boldsymbol{f}_{t-1}$ и сделать шаг $\boldsymbol{\delta}_t = -\alpha_t \boldsymbol{g}_t,\ \alpha_t > 0$, в направлении, противоположном градиенту. Легко видеть, что i-я координата градиента $\boldsymbol{g}_t$ выражается в виде

$$\boldsymbol{g}_t^i = \left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=F_{t-1}(x_i)}.$$

Величину шага α_t мы найдем, решая следующую задачу одномерной оптимизации:

$$\alpha_t = \arg\min_{\alpha} \mathcal{L}(\boldsymbol{f}_{t-1} - \alpha\boldsymbol{g}_t) = \arg\min_{\alpha} \sum_{i=1}^{\ell} L(y_i, \boldsymbol{f}_{t-1}^i - \alpha\boldsymbol{g}_t^i).$$

Обновленное решение записывается в виде

$$\boldsymbol{f}_t = \boldsymbol{f}_{t-1} - \alpha_t\boldsymbol{g}_t.$$

Проблема заключается в том, что мы хотим получить композицию, способную давать ответы на новых, не содержащихся в обучающей выборке точках. А пока что мы построили функцию, определенную только на точках обучающей выборки. Чтобы преодолеть эту сложность, воспользуемся следующим приемом. На каждой итерации мы будем выбирать ту базовую функцию h_t, вектор ответов которой на обучающей выборке $\{h_t(x_i)\}_{i=1}^\ell$ наиболее «сонаправлен» с отрицательным градиентом $-\boldsymbol{g}_t$. Мерить степень «сонаправленности» мы будем с помощью МНК:

$$h_t = \arg\min_{\beta,h} \sum_{i=1}^{\ell} \big(-\boldsymbol{g}_t^i - \beta h(x_i)\big)^2 = \arg\min_{\beta,h} \sum_{i=1}^{\ell} \left(-\left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=F_{t-1}(x_i)} - \beta h(x_i)\right)^2.$$

После этого остается найти вес базовой функции h_t:

$$\alpha_t = \arg\min_{\alpha} \sum_{i=1}^{\ell} L\big(y_i, F_{t-1}(x_i) + \alpha h_t(x_i)\big)$$

и получить обновленную композицию

$$F_t(x) = F_{t-1}(x) + \alpha_t h_t(x).$$

Коротко подитожем полученный алгоритм:

Алгоритм GBM:

1. Инициализация композиции константной функцией

$$F_0(x) = \arg\min_{\alpha} \sum_{i=1}^{\ell} L(y_i, \alpha);$$

2. Для $t = 1, \ldots, M$:

   (a) Для $i = 1, \ldots, \ell$ вычислить

$$\boldsymbol{g}_t^i = \left.\frac{\partial L(y_i, z)}{\partial z}\right|_{z=F_{t-1}(x_i)};$$

   (b) Выбираем очередную базовую функцию с помощью МНК:

$$h_t = \arg\min_{\beta, h} \sum_{i=1}^{\ell} \big(-\boldsymbol{g}_t^i - \beta h(x_i)\big)^2;$$

   (c) Вычисляем вес нового базового классификатора, решая задачу одномерной оптимизации:

$$\alpha_t = \arg\min_{\alpha} \sum_{i=1}^{\ell} L\big(y_i, F_{t-1}(x_i) + \alpha h_t(x_i)\big)$$

   (d) Обновляем композицию:

$$F_t(x) = F_{t-1}(x) + \alpha_t h_t(x).$$

Описанный выше метод — ни что иное как жадная пошаговая оптимизация средних потерь на обучающей выборке, где на каждой итерации шаг против градиента оптимизируемой функции в текущей точке мы заменяем на шаг вдоль базовой функции, ответы которой наиболее сонаправлены с отрицательным градиентом.

При этом роль градиента можно особо наглядно продемонстрировать в случае, когда мы используем квадратичные потери $L\big(y, F(x)\big) = \big(y - F(x)\big)^2$.

Задача. *Чему в этом случае равен градиент $\boldsymbol{g}_t$?*

В этом случае градиент $\boldsymbol{g}_t$, как несложно убедиться, задается своими координатам $\boldsymbol{g}_t^i = 2\big(y_i - F_{t-1}(x_i)\big)$ — то есть это вектор остатков ответов текущей модели на обучающей выборке. Таким образом, на каждом шаге GBM в этом случае ищет базовую функцию, которая лучше всего приближает остатки текущей модели.

Stochastic Gradient Boosting Существует модификация метода GBM, известная как Stochastic GBM [2], которая на практике существенно улучшает качество получаемой композиции. Модификация заключается в шаге 2.b. Очередная базовая функция h_t выбирается не по всей обучающей выборке, а по ее случайному подмножеству $\tilde{X}_t \subset X^\ell$, вытянутому без возвращений. Рекомендуется использовать подвыборки вдвое меньшего размера. Подобная рандомизация, во-первых, ведет к значительной устойчивости алгоритма к переобучению, и во-вторых, ускоряет решение МНК на шаге 2.b.

Gradient Boosting для классификации Описанный метод GBM применим не только для решения задачи регрессии, но и для классификации. Достаточно заметить, что задачу классификации можно свести к задачи регрессии, приближая функцию апостериорной вероятности $P(y = +1|x)$ на обучающей выборке, используя, например, логарифмическую функцию потерь (очень похоже на то, что делает логистическая регрессия). С помощью полученной таким способом функции h классификацию можно получать в виде $a(x) = [h(x) > 0.5]$.

Decision Trees Gradient Boosting Считается, что gradient boosting, примененный к решающим деревьям — один из самых универсальных и сильных классификаторов, существующих на сегодняшний день. Он во многих случаях работает лучше, чем упомянутый Random Forest. В частности, в основе поискового алгоритма компании Yandex [5, 6, 7] лежит именно GBM над решающими деревьями определенного вида.

Список литературы

[1] *Hastie, T., R. Tibshirani, and J. H. Friedman.* The Elements of Statistical Learning: Data Mining, Inference and Prediction. — Springer, 2001.

[2] *Friedman, J.* (1999) Stochastic gradient boosting. Technical report, Stanford university.
http://www-stat.stanford.edu/~jhf/ftp/stobst.pdf.

[3] *Friedman, J.* (2001) Greedy function approximation: gradient boosting machine. *Annals of statistics.* — 29(5). — Pp. 1189–1232.
http://www-stat.stanford.edu/~jhf/ftp/trebst.pdf.

[4] Довольно наглядное видео о GBM с квадратичными потерями.
http://www.youtube.com/watch?v=sRktKszFmSk.

[5] Ненаучная наглядная статья про применение машинного обучения в поиске.
http://company.yandex.com/technologies/matrixnet.xml.

[6] Слайды лекции Яндекса о качестве поиска.
http://romip.ru/russir2009/slides/yandex/lecture.pdf.

[7] Слайды про MatrixNet в очень общих словах.
http://www.ashmanov.com/arc/searchconf2010/08gulin-searchconf2010.ppt.