

Лекция 8

Структура ошибки выпуклых комбинаций,
комитетные методы, логическая коррекция

Лектор – Сенько Олег Валентинович

Курс «Методы машинного обучения и интеллектуальный анализ данных»
МФК

Использование различных методов прогнозирования (распознавания), а также различных обучающих выборок или подмножеств признаков позволяет получить набор прогнозирующих (распознающих) алгоритмов: A_1, \dots, A_r . Можно попытаться увеличить обобщающую способность за счёт выбора алгоритма с минимальной оценкой ошибки прогнозирования. Однако нередко более эффективной процедурой является вычисление прогноза с использованием всех алгоритмов из A_1, \dots, A_r . Использование коллектива (ансамбля) алгоритмов, которые строятся с помощью различных методов позволяет использовать при прогнозировании различные принципы экстраполяции, лежащих в основе этих методов. Статистическое обоснование использованию ансамбля алгоритмов даёт анализ ошибки выпуклой комбинации прогнозов, вычисляемых членами ансамбля. Предположим, что алгоритмы ансамбля A_1, \dots, A_r вычисляют прогноз переменной Y .

Пусть f_i - прогноз, вычисляемый алгоритмом A_i . Тогда

$$\Delta_i = E_{\Omega}(Y - f_i)^2$$

является математическим ожиданием квадрата ошибки прогнозирования для A_i . Введём обозначение $\rho_{i'i''}$ для математического ожидания квадрата отклонения друг от друга прогнозов, вычисляемых алгоритмами $A_{i'}$ и $A_{i''}$. То есть

$$\rho_{i'i''} = E_{\Omega}(f_{i'} - f_{i''})^2.$$

Пусть c_1, \dots, c_r - положительные коэффициенты такие, что $\sum_{i=1}^r c_i = 1$. Обозначим через \hat{f} **выпуклую комбинацию** прогнозов, вычисляемых алгоритмами ансамбля A_1, \dots, A_r . То есть

$$\hat{f} = \sum_{i=1}^r c_i f_i.$$

Для ошибки выпуклой комбинации справедливо выражение

$$\hat{\Delta} = E_{\Omega}(Y - \hat{f})^2 = \sum_{i=1}^r c_i \Delta_i - \frac{1}{2} \sum_{i'=1}^r \sum_{i''=1}^r c_{i'} c_{i''} \rho_{i' i''} \quad (1)$$

Принимая во внимание, что все квадратичные отклонения $\rho_{i' i''}$ всегда неотрицательны, а коэффициенты c_1, \dots, c_r положительны получаем неравенство

$$\hat{\Delta} \leq \sum_{i=1}^r c_i \Delta_i.$$

Иными словами математическое ожидание квадрата ошибки выпуклой комбинации всегда не превышает аналогичную выпуклую комбинацию математических ожиданий квадратов ошибок отдельных алгоритмов ансамбля.

Рассмотрим, случай, когда все алгоритмы участвуют в построении коллективного решения равноправно. В этом случае $c_i = \frac{1}{r}$, $i = 1, \dots, r$. Выпуклая комбинация становится просто средним значением

$$\hat{f} = \frac{1}{m} \sum_{i=1}^r f_i.$$

Математическое ожидание квадрата ошибки усреднённого по ансамблю прогноза вычисляется по формуле

$$\hat{\Delta} = E_{\Omega}(Y - \hat{f})^2 = \frac{1}{m} \sum_{i=1}^r \Delta_i - \frac{1}{2} \frac{1}{m^2} \sum_{i'=1}^r \sum_{i''=1}^r c_{i'} c_{i''} \rho_{i' i''} \quad (2)$$

Таким образом математическое ожидание квадрата ошибки усреднённого по ансамблю прогноза представляет собой разницу между средней по ансамблю величиной математического ожидания квадрата ошибки и средней величиной квадратичного отклонения между прогнозами вычисляемыми различными алгоритмами.

Рассмотрим сначала несколько простейших эвристических методов принятия коллективных решений. Предположим, что у нас есть ансамбль алгоритмов распознавания A_1, \dots, A_r , которые были использованы для классификации некоторого объекта s^* .

Голосование по большинству. Простейшим комитетным методом является метод голосования по большинству, относящий объект к тому классу, к которому он был присвоен относительным большинством алгоритмов.

Использование вещественных оценок за классы. Напомним, что произвольный распознающий алгоритм является комбинацией распознающего оператора, вычисляющего оценки за классы и решающего правила, производящего классификацию по оценкам, вычисленным распознающим оператором. Предположим, что $\Gamma_l^i(*)$ - оценка за класс l , вычисляемая алгоритмом A_i . Коллективное решение может строиться путём вычисления коллективных оценок за классы через оценки $\Gamma_l^i(*)$, соответствующие отдельным алгоритмам. При этом могут использоваться различные варианты вычисления

1) Коллективная оценка за класс K_l вычисляется как среднеарифметическое оценок, вычисляемых алгоритмами из ансамбля $\{A_1, \dots, A_r\}$:

$$\Gamma_l^{av}(s^*) = \sum_{j=1}^r \Gamma_l^j(s^*).$$

2) Коллективная оценка за класс K_l вычисляется как вычисляется как минимум всех оценок за данный класс полученных алгоритмами из ансамбля $\{A_1, \dots, A_r\}$:

$$\Gamma_l^{min}(s^*) = \min_{j \in \{1, \dots, r\}} \Gamma_l^j(s^*).$$

3) Коллективная оценка за класс K_l вычисляется как вычисляется как максимум всех оценок за данный класс полученных алгоритмами из ансамбля $\{A_1, \dots, A_r\}$:

$$\Gamma_l^{min}(s^*) = \max_{j \in \{1, \dots, r\}} \Gamma_l^j(s^*).$$

4) Еще одним употребительным способом построения комитетного решения является использование произведений оценок, вычисляемых алгоритмами из ансамбля $\{A_1, \dots, A_r\}$:

$$\Gamma_l^{av}(s^*) = \prod_{j=1}^r \Gamma_l^j(s^*).$$

К достоинствам комитетных методов относится их простота, высокая быстродействие. Для применения этих методов не требуется никакой дополнительной процедуры обучения, что позволяет сразу переходить к распознаванию объектов комитетом обученных алгоритмов.

Подобными же достоинствами обладает другой известный метод построения коллективных решений – «Наивный байесовский классификатор», который является статистическим методом, основанном на оценках вероятностей принадлежности объекта классам в зависимости от результатов классификации отдельными алгоритмами. Предположим, что алгоритмы $\{A_1, \dots, A_r\}$ отнесли объект s^* в классы $K_{J(1)}, \dots, K_{J(r)}$ соответственно. Факт отнесения объекта s в класс K_i алгоритмом A_j далее будем обозначать $A_j(s) = \text{Pt}_i(s)$, где $\text{Pt}_i(s)$ является предикатом, обозначающим отнесение s в класс K_i . Наибольшую точность распознавания обеспечивает байесовский классификатор, относящий объект в класс K_{i^*} , для которого максимальной является условная вероятность

$$P[s^* \in K_{i^*} \mid A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(r)}(s^*)] \quad (3)$$

Условная вероятность (1) для класса K_i может быть вычислена по формуле Байеса

$$\begin{aligned} & P[s^* \in K_i \mid A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(r)}(s^*)] = \\ &= \frac{P(K_i)P[A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(r)}(s^*) \mid s^* \in K_i]}{P[A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(1)}(s^*)]}. \end{aligned}$$

Условная вероятность

$$P[A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(1)}(s^*) \mid s^* \in K_i]$$

может быть оценена исходя из гипотезы о независимости входящих в ансамбль классификаторов. То есть

$$\begin{aligned} & P[A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(r)}(s^*) \mid s^* \in K_i] = \\ &= \prod_{j=1}^r P[A_1(s^*) = \text{Pt}_{J(j)}(s^*) \mid s^* \in K_i]. \end{aligned}$$

В качестве оценок вероятностей

$$P(K_1), \dots, P(K_l)$$

$$P[A_1(s^*) = \text{Pt}_{J(j)}(s^*) \mid s^* \in K_i],$$

при

$$j = 1, \dots, r, i = 1, \dots, r$$

могут быть использованы соответствующие доли объектов обучающей выборки. Отметим, что вероятность

$$P[A_1(s^*) = \text{Pt}_{J(1)}(s^*), \dots, A_r(s^*) = \text{Pt}_{J(r)}(s^*)]$$

является одинаковым для всех классов множителей, который может не учитываться при вычислении окончательного решения.

Комитетные методы и наивный байесовский классификатор являются простейшими методами коллективной коррекции, не учитывающих взаимодействие алгоритмов ансамбля или их относительную эффективность. Требование повышения обобщающей способности ансамбля за счёт более полного учёта его структуры и использования возможностей лежащих в его основе эвристик привело к созданию средств алгебраической и логической коррекции. Методы логической коррекции учитывают только окончательные результаты классификации. Пусть у нас имеется некоторая выборка $\tilde{S}_c = \{s_1, \dots, s_q\}$ объектов, принадлежащих классам K_1, \dots, K_L , по которой мы собираемся произвести коррекцию. Данной выборке может быть сопоставлена информационная матрица $\|\alpha_{li}\|_{L \times q}$, где $\alpha_{li} = 1$ при $s_i \in K_l$ и $\alpha_{ij} = 0$ в противном случае.

Выборке \tilde{S}_c может быть также сопоставлен набор матриц

$$\{\|\beta_{li}^j\|_{L \times q} \mid j = 1, \dots, r\}.$$

Элемент $\beta_{li}^j = 1$, если $A_j(s_i) = \text{Pt}_l(s_i)$, и $\beta_{li}^j = 0$ в противном случае. Поиск оптимального логического корректора сводится к поиску для каждого класса такой логической функции $F_l(z_1, \dots, z_r)$ от булевых переменных z_1, \dots, z_r , чтобы равенство

$$F_l[\beta_{li}^{g(1)}, \dots, \beta_{li}^{g(r)}] = \alpha_{li}$$

выполнялось для возможно большего числа объектов выборки \tilde{S}_c . Функция $g(i)$ устанавливает связь между переменными z_1, \dots, z_r и алгоритмами A_1, \dots, A_r . Использование g позволяет учитывать информативность алгоритмов для оценки принадлежности распознаваемых объектов классу K_l , через их место в логической функции.

Предположим, что отсутствуют противоречия типа существования в выборке \tilde{S}_c объектов $s_{i'}$ и $s_{i''}$ с неодинаковыми $\alpha_{li'}$ и $\alpha_{li''}$, которые однако одинаково классифицируются алгоритмами ансамбля. То есть $\beta_{li'}^j = \beta_{li''}^j$ при $j = 1, \dots, r$.

В случае, если задана какая-либо функция g , а множество векторов

$$\{(\beta_{li}^1, \dots, \beta_{li}^r) \mid i = 1, \dots, r\}$$

включает всё множество вершин единичного куба \mathbb{E}^r , логическая функция F_l оказывается полностью определённой. В противном случае задача построения логического корректора включает в себя задачу доопределения логической функции естественным путём заданной на выборке \tilde{S}_c на весь единичный куб \mathbb{E}^r .

Одним из способов логической коррекции является построение монотонных корректоров, которое сводится к поиску такой функции g , что логическая функция F_l , правильно вычисляющая элементы информационной матрицы, является монотонной. То есть ищется функция $g(i)$, которая

а) удовлетворяет равенству

$$F_l[\beta_{li}^{g(1)}, \dots, \beta_{li}^{g(r)}] = \alpha_{li}$$

при $i = 1, \dots, q$;

б) для любых векторов (z'_1, \dots, z'_r) и (z''_1, \dots, z''_r) , удовлетворяющих условию

$$(z'_1, \dots, z'_r) \succeq (z''_1, \dots, z''_r)$$

выполняется неравенство

$$F_l(z'_1, \dots, z'_r) \succeq F_l(z''_1, \dots, z''_r)$$

Построение монотонных корректоров сводится к следующей схеме. В исходном наборе A_1, \dots, A_r для каждого класса K_l выбирается поднабор $A_{f(1)}, \dots, A_{f(k)}$. Объект s относится монотонным логическим корректором в класс K_l в том и только в том случае, если он отнесён в K_l всеми алгоритмами из $A_{f(1)}, \dots, A_{f(k)}$ и ещё одним алгоритмом из набора A_1, \dots, A_r , который не принадлежит $A_{f(1)}, \dots, A_{f(k)}$.

Универсальным способом построения оптимального распознающего алгоритма по набору исходных алгоритмов A_1, \dots, A_r является использование алгебраических методов коррекции. В отличие от логических методов коррекции алгебраические методы используют не только окончательные результаты классификации, содержащиеся в матрицах $\|\beta_{li}^j\|_{L \times q}$, но также матрицы оценок $\|\gamma_{li}^j\|_{L \times q}$, вычисляемые операторами R_1, \dots, R_k , входящими в алгоритмы A_1, \dots, A_r . Элемент γ_{li}^j является оценкой объекта s_i за класс K_l , вычисляемая оператором R_j , $i = 1, \dots, q, l = 1, \dots, L, j = 1, \dots, r$. Основы теории алгебраической коррекции были разработаны Ю.И.Журавлёвым в 1976-1978 годах. Задача распознавания в алгебраической теории рассматривается как задача построения по начальной информации I о классах K_1, \dots, K_L для предъявленной для распознавания выборки $\tilde{S}_c = \{s_1, \dots, s_q\}$ правильной информационной матрицы $\|\alpha_{li}^j\|_{L \times q}$.

Последнюю задачу мы будем называть задачей $Z(I, \tilde{S}_c, Pt_1, \dots, Pt_L)$ или просто задачей Z . Примером начальной информации о классах является таблица признаков описаний эталонных объектов классов и их информационная матрица. Предположим, что у нас имеется множество алгоритмов $\{A\}$, переводящих пару (I, \tilde{S}_c) в матрицы $\|\beta_{li}^j\|_{L \times q}$, составленные из элементов $\{0, 1, \Delta\}$, где значения 1 и 0 как и раньше соответствуют истинности или ложности предикатов, вычисленными алгоритмами из множества $\{A\}$, значение Δ соответствует отказу от вычисления значения предиката.

Определение 1. Алгоритм A называется корректным для задачи Z , если выполнено равенство

$$A(I, \tilde{S}_c, Pt_1, \dots, Pt_L) = \|\alpha_{li}\|_{L \times q}.$$

Алгоритм, не являющийся корректным для задачи Z , называется некорректным.

Совокупность $\{A\}$ состоит из вообще говоря некорректных алгоритмов. Алгебраический подход к решению задач распознавания включает в себя введение алгебраических операций над алгоритмами из $\{A\}$, позволяющих строить корректные алгоритмы по наборам алгоритмов из $\{A\}$. Поскольку каждый распознающий алгоритм может быть представлен как последовательное выполнение распознающего оператора и решающего правила, множеству $\{A\}$ соответствуют множества операторов $\{R\}$ и множество решающих правил $\{C\}$. Каждый из операторов из множества $\{R\}$ вычисляет для задачи Z матрицу оценок за классы

$$R^*(I, \tilde{S}_c) = \|\gamma_{li}^*\|_{L \times q}$$

На множестве операторов $\{R\}$ вводятся операции сложения, умножения и умножения на скаляр.

Предположим, что R' и R'' являются операторами из $\{R\}$. При этом $R'(I, \tilde{S}_c) = \|\gamma'_{li}\|_{L \times q}$ и $R''(I, \tilde{S}_c) = \|\gamma''_{li}\|_{L \times q}$. Пусть b является некоторой скалярной величиной. Операция умножения на скаляр преобразует оператор R' в оператор $(b \bullet R')$, задаваемый формулой

$$(b \bullet R')(I, \tilde{S}_c) = \|b\gamma'_{li}\|_{L \times q}, \quad (4)$$

Сумма операторов $(R' + R'')$ задаётся формулой

$$(R' + R'')(I, \tilde{S}_c) = \|\gamma'_{li} + \gamma''_{li}\|_{L \times q}, \quad (5)$$

Произведение операторов $(R' \bullet R'')$ задаётся формулой

$$(R' \bullet R'')(I, \tilde{S}_c) = \|\gamma'_{li} \cdot \gamma''_{li}\|_{L \times q}, \quad (6)$$

Использование операций (4)-(6) позволяет строить новые распознающие операторы, являющиеся полиномами от операторов из исходного множества вида

$$\sum_{i=1}^{N_p} a_i [R_{t(1,i)} \bullet \dots \bullet R_{t(k(i),i)}]$$

Функция $t(j, i)$ указывает на оператор, находящийся в позиции j слагаемом с номером i , k_i - число сомножителей в слагаемом с номером i .

Одним из способов получения ансамбля является использование алгоритмов, обученных по разным обучающим выборкам, которые возникают в результате случайного процесса, лежащего в основе исследуемой задачи. Обычно при решении прикладной задачи в распоряжении исследователя имеется обучающая выборка $\tilde{S}_t = \{s_1, \dots, s_m\}$ ограниченного объёма. Однако процесс генерации семейства выборок из генеральной совокупности может быть имитирован с помощью процедуры бутстрэп (bootstrap), которая основана на выборках с возвращениями из \tilde{S}_t . В результате получаются выборки \tilde{S}_*^{bg} , включающие объекты из обучающей выборки \tilde{S}_t . Однако некоторые объекты \tilde{S}_t могут встречаться в \tilde{S}_*^{bg} более одного раза, а другие объекты отсутствовать. Предположим, что с помощью процедуры бутстрэп получено T выборок. С помощью заранее выбранного метода, используемого для обучения отдельных алгоритмов распознавания, получим множество, включающее T распознающих алгоритмов $\tilde{A}_{bg} = \{A_1^{bg}, \dots, A_T^{bg}\}$.

Для получения коллективного решения может быть использован простейший комитетный метод, относящий объект в тот класс, куда его отнесло большинство алгоритмов. Данная процедура носит название **бэггинг (bagging)**, что является сокращением названия Bootstrap Aggregating. Процедура бэггинг показывает высокий прирост обобщающей способности по сравнению с алгоритмом, обученным с помощью базового метода по исходной обучающей выборке \tilde{S}_t , в тех случаях, когда вариационная составляющая ошибки базового метода высока. К таким моделям относятся в частности решающие деревья и нейросетевые методы. При использовании в качестве базового метода решающих деревьев процедура бэггинг приводит к построению ансамблей решающих деревьев (решающих лесов).

Основной идеей алгоритма **бустинг** является пошаговое наращивание ансамбля алгоритмов. Алгоритм, который присоединяется к ансамблю на шаге k обучается по выборке, которая формируется из объектов исходной обучающей выборки \tilde{S}_t .

В отличие от метода бэггинг объекты выбираются не равноправно, а исходя из некоторого вероятностного распределения, заданного на выборке \tilde{S}_t . Данное распределение вычисляется по результатам классификации с помощью ансамбля, полученного на предыдущем шаге. Приведём схему одного из наиболее популярных вариантов метода бустинг AdaBoost (Adaptive boosting) более подробно. На первом шаге присваиваем начальные значения весов (w_1^1, \dots, w_m^1) объектам обучающей выборки. Поскольку веса имеют вероятностную интерпретации, то для них соблюдаются ограничения $\sum_{j=1}^m w_j^1 = 1$, $w_j^1 \in [0, 1]$. Обычно начальное распределение выбирается равномерным $w_j^1 = \frac{1}{m}$, $j = 1, \dots, m$. Выбираем число итераций T . На итерации k генерируем выборку \tilde{S}_k^{bs} из исходной выборки \tilde{S}_t согласно распределению задаваемому весами (w_1^k, \dots, w_m^k) . Обучаем распознающий алгоритм A_k^{bs} по выборке \tilde{S}_k^{bs} .

Вычисляем взвешенную ошибку по формуле $\varepsilon_k = \sum_{j=1}^m w_j^k e_j^k$, где $e_j^k = 1$, если алгоритм A_k^{bs} неправильно классифицировал объект s_j и $e_j^k = 0$ в противном случае. В том случае, если $\varepsilon_k \geq 0.5$ или $\varepsilon_k = 0$ игнорируем шаг и заново генерируем выборку \tilde{S}_k^{bs} исходя из весовых коэффициентов $w_j^1 = \frac{1}{m}$, $j = 1, \dots, m$. В случае если $\varepsilon_k \in (0, 0.5)$ вычисляем коэффициенты $\tau_k = \frac{\varepsilon_k}{1-\varepsilon_k}$ и пересчитываем веса объектов по формуле

$$w_j^{k+1} = \frac{w_j^k (\tau_k)^{1-e_j^k}}{\sum_{j=1}^m w_j^k (\tau_k)^{1-e_j^k}} \quad (7)$$

при $j = 1, \dots, m$.

Процесс, задаваемый формулой (1), продолжается до тех пор, пока не выполнено T итераций. В результате мы получаем совокупность из T распознающих алгоритмов $A_1^{bs}, \dots, A_T^{bs}$.

Предположим, что нам требуется распознать объект s^* . Пусть $\beta_l^k(s^*) = 1$, если s^* отнесён алгоритмом A_k^{bs} в класс K_l , и $\beta_l^k(s^*) = 0$ в противном случае. Оценка объекта s^* за класс K_l вычисляется по формуле

$$\Gamma_l(s^*) = \sum_{k=1}^T \ln \frac{1}{\tau_k} \beta_l^k(s^*).$$

Объект s^* будет отнесён к классу, оценка за который максимальна.

Описанный вариант метода носит название AdaBoost. M1.

Эффективность процедур бустинга подтверждается многочисленными экспериментами на реальных данных. В настоящее время существует большое количество вариантов метода, имеющих разное обоснование.

Вернёмся к методу минимизации эмпирического риска

$$Q(\tilde{S}_t, A) = \sum_{j=1}^m L[y_j, A(\mathbf{x}_j)]$$

Попробуем использовать метод градиентного спуска для минимизации $Q(\tilde{S}_t, A)$. Значения прогнозов в точках признавого пространства, соответствующего объектам \tilde{S}_t для алгоритма, получаемого на шаге k вычисляются через значения прогнозов в этих точках для алгоритма, полученного на шаге $k - 1$:

$$A^{(k)}(\mathbf{x}_j) = A^{(k-1)}(\mathbf{x}_j) - \eta_k \times \frac{dQ}{dA^{k-1}(\mathbf{x}_j)}$$

где $\eta_k > 0$, $j = 1, \dots, m$

При заданной функции потерь производные $\frac{dQ}{dA^{(k-1)}(\mathbf{x}_j)}$ могут быть легко вычислены. Например, при

$$L[y_j, A^{(k-1)}(\mathbf{x}_j)] = [y_j - A^{(k-1)}(\mathbf{x}_j)]^2$$

$$\frac{dQ}{dA^{(k-1)}(\mathbf{x}_j)} = -2y_j + A^{(k-1)}(\mathbf{x}_j)$$

Значения производных $\frac{dQ}{dA^{(k-1)}(\mathbf{x}_j)}$ могут рассматриваться как значения (h_1, \dots, h_m) некоторой переменной H . По обучающей выборке $\{(h_1, \mathbf{x}_1), \dots, (h_1, \mathbf{x}_m)\}$ с использованием какого-либо из методов МО построим алгоритм, позволяющий прогнозировать H в произвольной точке \mathbf{x} признакового пространства. Обозначим результат применения этого алгоритма в точке \mathbf{x} как $\hat{H}^{(k)}(\mathbf{x})$

В результате на шаге (k) получаем новый алгоритм

$$A^{(k)}(\mathbf{x}) = A^{(k-1)}(\mathbf{x}) - \eta_k \times \hat{H}^{(k)}(\mathbf{x})$$

Величину η_k можем подобрать из условия

$$\eta_k = \arg \min Q(\tilde{S}_t, A^{(k)})$$

Окончательный алгоритм, полученный за r шагов, принимает вид

$$A^{(r)}(\mathbf{x}) = A^{(0)}(\mathbf{x}) - \sum_{k=1}^r \eta_k \times \hat{H}^{(k)}(\mathbf{x})$$

"Градиентный бустинг для решающего леса".

Используются следующая модификация

1) Выходом исходного алгоритма $A^{(0)}(\mathbf{x})$ является

$$\gamma_0 = \arg \min Q(\tilde{S}_t, \gamma)$$

При использовании квадратичных потерь $\gamma_0 = \frac{1}{m} \sum_{j=1}^m y_j$.

2) Коррекция $A^{(k-1)}(\mathbf{x})$ производится отдельно внутри каждой области q_t , соответствующей концевой вершине дерева, задающего $\hat{H}^{(k)}(\mathbf{x})$:

$$A^{(k)}(\mathbf{x} \mid \mathbf{x} \in q_t) = A^{(k-1)}(\mathbf{x}) + \gamma_t$$

$$\gamma_t = \arg \min_{\gamma} \sum_{\mathbf{x}_j \in q_t} L[y_j, A^{(k-1)}(\mathbf{x}) + \gamma]$$

Эффективность процедур бустинга подтверждается многочисленными экспериментами на реальных данных. В настоящее время существует большое количество вариантов метода, имеющих разное обоснование.