ПОЧЕМУ ЛИСТ ВЫДАЁТ ИМЕННО ЭТО ЧИСЛО?

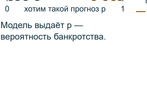
Читать по блокам: в блоке — по шагам слева направо, шаг — сверху вниз.
Номера (1), (2)… — для ссылок. Условия: CPU, Plain · числовые признаки · веса 1
λ > 0 — коэффициент регуляризации (№04) · в листе один шаг Ньютона

01 Бинарная классификация: какой прогноз хорош и как измерить это одним числом?

$$P = \prod_i p_i^{y_i} (1-p_i)^{1-y_i}$$

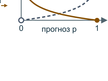
$$LL = \sum_{y_i=1} |\, y_i \ln p_i + \underbrace{(1-y_i)\ln(1-p_i)}_{y_i=0} \,| \to \max$$

i — номер клиента; y_i — его известный ответ: 1 — банкрот, 0 — нет; p_i — прогноз модели: вероятность банкротства. Ответы даны; менять можно только прогнозы.
P — правдоподобие: вероятность получить именно такие ответы при этих прогнозах, если клиенты независимы (у банкрота множитель p, у остальных 1 − p).
∏ — знак произведения: множители всех клиентов перемножаются; для двух клиентов в P два множителя, Σ — знак суммы по всем клиентам.
LL — логарифм правдоподобия: LL = ln P; логарифм превращает произведение в сумму. ln растёт вместе с аргументом ⇒ LL больше, когда у банкротов $p \to 1$, у остальных $p \to 0$.



Модель выдаёт p — вероятность банкротства.

считаем P, затем LL = ln P



LogLoss = −LL → min: это сумма штрафов L всех клиентов.
L — штраф одного клиента: кривые на рисунке.
В выводе используем сумму, а не среднее: коэффициент λ ниже относится к этой записи.
LogLoss — потеря по умолчанию в CatBoostClassifier; бывают и другие.

Слова для вывода ниже. Дерево — ветвящаяся схема вопросов о признаках клиента, например о его доходе. Каждый клиент проходит по ней один путь и приходит в лист — конец пути, где записано одно число. Разбиение — то, как вопросы дерева делят клиентов по листьям. Подробнее — в №02 и №04.

ВЫВОД. Вся дальнейшая работа — найти такие деревья (разбиения и числа в листьях), при которых LogLoss минимален.

02 На чём считает бустинг: логит z вместо вероятности p

ВВОДИМ логит z и сигмоиду σ

Модель CatBoost — набор деревьев. Дерево задаёт вопросы о признаках клиента — например, «доход не больше 50?» — и приводит его в лист; в листе записано число. Числа из листьев складывают — выходит любое число z, а вероятность p живёт только от нуля до единицы. Мост между ними — сигмоида σ: функция, переводящая z в p. Сигмоида — принятое устройство модели: её не выводим.

$$p = \sigma(z) = \frac{e^z}{1+e^z}$$

$$z = \ln\frac{p}{1-p}$$



z — логит: любое число.
При любом z выходит 0 < p < 1.
e ≈ 2,718 — основание логарифма ln.

ПОДСТАВЛЯЕМ сигмоиду в штраф

$$L = -y\ln p - (1-y)\ln(1-p) \qquad (1)$$

L — штраф одного клиента: слагаемое LL из №01 с обратным знаком. Клиент один, поэтому номер i не пишем: y — его ответ, p — его прогноз.
В формуле два логарифма: ln p и ln(1 − p).

$$\ln p = \ln\frac{e^z}{1+e^z} = z - \ln(1+e^z)$$

$$\ln(1-p) = \ln\frac{1}{1+e^z} = -\ln(1+e^z)$$

Вместо p ставим дробь σ(z) из шага «ВВОДИМ»;
1 − p — дробь с тем же знаменателем и единицей в числителе.
Логарифм дроби — разность логарифмов; ln e^z = z, ln 1 = 0.

ПОЛУЧАЕМ штраф клиента через логит z

$$L = -y\left[z - \ln(1+e^z)\right] + (1-y)\ln(1+e^z)$$

Подставили оба логарифма в (1). Раскрываем скобки:
ln(1 + e^z) входит с множителем +y и с множителем −y — эти два слагаемых сокращаются.

$$L(y,z) = \ln(1+e^z) - y\,z \qquad (2)$$

L(y, z) — тот же штраф клиента, записанный через логит z и ответ y. Вероятность p из формулы ушла: остался логит z — его и меняют деревья.
Дальше работаем со штрафом в виде (2).

03 Куда и насколько менять логит z: наклон и штраф за шаг

СЧИТАЕМ производные штрафа (2) по логиту z; ответ y при этом не меняется

$L(y,z) = \ln(1+e^z) - y\,z$
штраф клиента: повтор формулы (2) из №02

производная по z: $(\ln(1+e^z))' = p$

$$g = \frac{\partial L}{\partial z} = p - y \qquad (3)$$

g — наклон штрафа по z при неизменном y; здесь p = σ(z)

ещё раз по z: dp/dz = p(1 − p)

$$h = \frac{\partial^2 L}{\partial z^2} = p(1-p) \qquad (4)$$

h — изгиб: как быстро меняется наклон

$$r = y - p = -g \qquad (5)$$

r — остаток: r > 0 — прогноз p ниже ответа y, r < 0 — выше. Равен −g.

СТРОИМ параболу в нашей точке

$$q(t) = L(y, z+t) + b\,t + c\,t^2$$

Наша точка — нынешний логит z клиента. t — пробный сдвиг от него: было z, станет z + t.
q(t) — парабола, которой заменяем кривую штрафа рядом с нашей точкой.
L(y, z) — штраф (2) при нынешнем логите z: число, от t не зависит. b и c — пока неизвестные числа.
При t = 0 значение уже совпадает: q(0) = L(y, z).
Подберём b и c так, чтобы совпали ещё наклон и изгиб.

ТРЕБУЕМ те же наклон и изгиб при сдвиге t = 0

$$q'(t) = b + 2ct \quad\Rightarrow\quad q'(0) = b = g$$

$$q''(t) = 2c \quad\Rightarrow\quad q''(0) = 2c = h$$

В каждой строке: производная параболы; её значение при t = 0; последнее равенство — наше требование. Требуем, чтобы наклон параболы равнялся наклону g (3) кривой штрафа, а изгиб — изгибу h (4): оба взяты в нашей точке, то есть при нынешнем z.
Отсюда b = g, c = h / 2.

ПОЛУЧАЕМ параболу и её дно

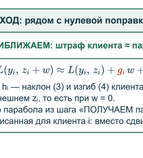
$$L(y, z+t) \approx q(t) = L(y,z) + g\,t + \tfrac{1}{2}h\,t^2$$

$$q'(t) = g + h\,t = 0 \quad\Rightarrow\quad t = -\frac{g}{h}$$

Слева — штраф после сдвига t, справа — парабола.
Основание знака ≈ — формула Тейлора: штраф после сдвига равен q(t) плюс ошибка. Третья производная штрафа по z равна p(1 − p)(1 − 2p); по модулю она не больше ¼. Поэтому ошибка не больше |t|³ / 24: при малом сдвиге она мала, при большом точность не обещаем.
Вторая строка — дно параболы: там её наклон равен нулю. Изгиб h = p(1 − p) положителен, поэтому ветви идут вверх и дно — минимум.
В №04 вместо сдвига t появится w — общая поправка логита для клиентов одного листа.

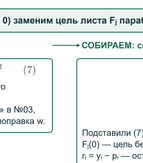
СМОТРИМ на графике: штраф одного клиента-банкрота (y = 1) как кривая от его логита z

Кадр 1. Где мы сейчас



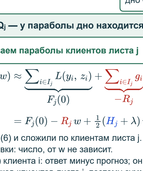
Кривая — штраф клиента (2). Точка — его нынешний логит z. Двигать точку можно только по кривой, меняя z.

Кадр 2. Наклон g: в какую сторону



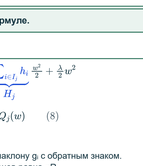
g — наклон касательной в этой точке. Здесь g < 0: кривая спускается вправо, значит, z увеличиваем. Правило: идём против знака g.

Кадр 3. Изгиб h: на сколько



Серая кривая — штраф (2), синяя — парабола. Она прилежит к кривой в нашей точке: те же значение, наклон g и изгиб h. Это парабола q(t) из ряда выше. Шагаем в её дно, а не в дно самой кривой: синяя точка — дно параболы; настоящий штраф после такого шага — на серой кривой над тем же z.

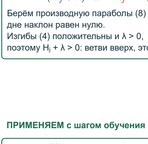
Кадр 4. Больше изгиб — короче шаг



Условное сравнение двух парабол с одинаковыми значением и наклоном в точке: чем больше изгиб h, тем ближе дно и короче шаг.

04 Шаг бустинга: какое число w поставить в лист нового дерева?

ДАНО: логит клиента — сумма по деревьям



● — наш клиент; остальные кружки — другие клиенты
Он попал в лист 1 ⇒ прибавим: z = 0,8 → 0,8 + w_1

Логит клиента z — начальное значение плюс вклады прежних деревьев: клиент проходит каждое дерево до листа и забирает его число. На рисунке сделали за одним клиентом — кружок в чёрном кольце: жирные линии — его путь. Начальный логит до первого дерева здесь 0; числа в листьях прежних деревьев — готовые вклады, уже умноженные на шаг обучения (№05).
Бустинг добавляет деревья по одному. Новое дерево уже разложило клиентов по листьям j = 1, 2 (как выбирают разбиение — страница 2). w_1, w_2 — числа его листьев, пока неизвестные: это полные поправки, до умножения на шаг обучения; число листа — одно на всех его клиентов. Наш клиент попал в лист 1, значит, к его логиту прибавят w_1.
Какие числа поставить в листья? В этом блоке строим цель листа и её параболу; само число найдём в №05.
Кружки под листьями нового дерева — клиенты: закрашенные — банкроты, пустые — остальные, как в №01.

ПРОБУЕМ: поставить в лист j поправку w

$$z_i \;\longrightarrow\; z_i + w \qquad (i \in I_j)$$

z_i — нынешний логит клиента i: сумма по прежним деревьям, как на рисунке слева.
j — номер листа нового дерева. I_j — клиенты, попавшие в лист j; запись $i \in I_j$ значит «клиент i из листа j».
w — число в листе (поправка): его прибавят к логиту z каждого клиента листа, одно на всех. В №03 сдвиг t был у одного клиента; теперь поправка w — одна на всех клиентов листа. Пока прибавляем w целиком. Позже, в №05, прибавим только её долю: η — шаг обучения, число от 0 до 1, на которое умножают поправку.

СЧИТАЕМ потерю листа после поправки

$$\sum_{i\in I_j} L(y_i,\, z_i + w)$$
$$= \sum_{i\in I_j} \left[\, \ln(1+e^{z_i+w}) - y_i\,(z_i+w) \,\right]$$

Потеря листа — сумма штрафов его клиентов. Берём штраф клиента (2), вместо z ставим сумму z + w и складываем по клиентам листа j. Это вклад листа в LogLoss (№01).

ДОБАВЛЯЕМ регуляризацию

$$\tfrac{\lambda}{2}\,w^2$$

Регуляризация — плата за большую поправку w: чем дальше w от нуля, тем плата больше (растёт как w^2). Квадрат — принятое правило CatBoost, а не вывод.
λ > 0 — её коэффициент (в CatBoost — параметр l2_leaf_reg).
Зачем: если в листе мало клиентов, потеря может требовать огромного w; плата сдерживает поправку.
½ — для удобства: сократится при дифференцировании.

ХОТИМ: наименьшую цель листа

$$F_j(w) = \sum_{i\in I_j} L(y_i,\, z_i + w) + \tfrac{\lambda}{2}w^2 \;\to\; \min_w \qquad (6)$$

$F_j(w)$ — цель листа j: потеря его клиентов после поправки w плюс регуляризация. Ищем w, при котором F_j наименьшая.
Разбиение уже задано: меняем только w.
Каждый клиент попадает ровно в один лист, и число листа влияет только на его клиентов. Поэтому числа разных листьев ищем по отдельности; дальше — один лист j.

УПИРАЕМСЯ: точно решать не будем

$$F_j'(w) = \sum_{i\in I_j} \sigma(z_i + w) - y_i \,] + \lambda w = 0$$

Условие минимума: наклон F_j равен нулю. Берём производную цели (6) по w. У каждого клиента это наклон штрафа (3), взятый при сдвинутом логите: σ(z_i + w) − y_i.
Регуляризация даёт λw.
При λ > 0 минимум есть, и он один.
Точно решать это уравнение не будем: w стоит под сигмоидой σ у каждого клиента, и такие уравнения решают численно, шаг за шагом. Здесь делаем один шаг Ньютона вместо точного решения: заменяем цель параболой и берём её дно — блоки никак.

ОБХОД: рядом с нулевой поправкой (w = 0) заменим цель листа F_j параболой Q_j — у параболы дно находится по формуле.

ПРИБЛИЖАЕМ: штраф клиента = парабола

$$L(y_i,\, z_i + w) \approx L(y_i,\, z_i) + g_i\,w + \tfrac{1}{2}h_i\,w^2 \qquad (7)$$

g_i и h_i — наклон (3) и изгиб (4) клиента i при его нынешнем z_i, то есть при w = 0.
Это парабола из шага «ПОЛУЧАЕМ параболу» в №03, записанная для клиента i: вместо сдвига t — поправка w.

СОБИРАЕМ: складываем параболы клиентов листа j

$$F_j(w) \approx \underbrace{\sum_{i\in I_j} L(y_i,\, z_i)}_{F_j(0)} + \underbrace{\sum_{i\in I_j} g_i}_{-R_j}\,w + \underbrace{\sum_{i\in I_j} h_i}_{H_j}\,\frac{w^2}{2} + \frac{\lambda}{2}w^2$$

$$= F_j(0) - R_j\,w + \tfrac{1}{2}(H_j + \lambda)\,w^2 \;=\; Q_j(w) \qquad (8)$$

Подставили (7) в цель (6) и сложили по клиентам листа j.
$F_j(0)$ — цель без поправки: число, от w не зависит.
$r_i = y_i - p_i$ — остаток (5) клиента i: ответ минус прогноз; он равен наклону g_i с обратным знаком.
$R_j = \Sigma\, r_i$ — сумма остатков клиентов листа j, поэтому сумма наклонов равна −R_j.
$H_j = \Sigma\, h_i$ — сумма изгибов (4) клиентов листа.
$Q_j(w)$ — парабола, которой заменяем F_j рядом с w = 0.

05 Ответ листа: дно параболы Q_j

НАХОДИМ ДНО параболы Q_j

$$Q_j'(w) = -R_j + (H_j + \lambda)\,w = 0$$
$$\Downarrow$$
$$(H_j + \lambda)\cdot w = R_j \qquad (9)$$

Берём производную параболы (8) по w; на дне наклон равен нулю.
Изгибы (4) положительны и λ > 0, поэтому $H_j + \lambda > 0$: ветви вверх, это минимум.

ДЕЛИМ (9) на сумму изгибов плюс λ

$$w_j^* = \frac{R_j}{H_j + \lambda} = \frac{\sum_{i\in I_j}(y_i - p_i)}{\sum_{i\in I_j} p_i(1-p_i) + \lambda} \qquad (10)$$

w_j^* — дно параболы Q_j: полная поправка листа j, один шаг Ньютона от w = 0. В статье её называют O_value.
Вторая дробь — то же самое через прогнозы: остатков (5) и изгиб (4).
К логиту прибавят долю этой поправки — шаг «ПРИМЕНЯЕМ» ниже.

ЧИТАЕМ формулу

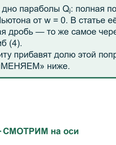
$R_j > 0$: в сумме прогнозы p клиентов листа меньше их ответов y ⇒ $w_j^* > 0$ ⇒ z и p у всех клиентов листа вырастут.
При том же ненулевом R_j: больше H_j или λ — меньше $|w_j^*|$.
w_j^* — точное дно параболы Q_j, но не обязательно минимум цели F_j: при малых w парабола близка к цели, дальше точность не гарантируется. Такой шаг — и дно параболы — называют шагом Ньютона.
Проверка: для одного клиента без регуляризации формула (10) даёт −g / h — дно параболы одного клиента из №03.

ПРИМЕНЯЕМ с шагом обучения η

$$z_i^{new} = z_i + \eta\, w_j^* \qquad (i \in I_j)$$

Слева — новый логит клиента (пометка new): после добавления дерева. Вклад нового дерева в логит — $\eta \cdot w_j^*$.
η — шаг обучения; в CatBoost это параметр learning_rate.
На этой карте рассматриваем 0 < η ≤ 1.
Если η меньше единицы, к логиту прибавляют часть поправки, чтобы не доверять одному дереву целиком; если η равно единице — всю.

СМОТРИМ на оси



Логит клиента на оси: до поправки; после прибавления $\eta \cdot w_j^*$ — так и делают; после полной поправки w_j^* — для сравнения. На схеме η = 0,5.

ПРОВЕРЯЕМ числами

Пример. Лист из трёх клиентов.
Ответы: $(y_1, y_2, y_3) = (1, 1, 0)$;
у всех z = 0, то есть p = 0,5; λ = 1.
Остатки (5): $(r_1, r_2, r_3) = (0{,}5;\ 0{,}5;\ -0{,}5)$,
их сумма $R_j = 0{,}5$.
Изгибы (4): h = 0,25 у каждого, их сумма $H_j = 0{,}75$.
Ответ (10): $w_j^* = 0{,}5\ /\ (0{,}75 + 1) = 2/7 \approx 0{,}29$.
С η = 0,5 к логиту прибавят $\eta \cdot w_j^* = 1/7 \approx 0{,}143$:
новый z = 1/7, новый p = σ(1/7) ≈ 0,536.

ИТОГ СТРАНИЦЫ 1. Когда разбиение задано, полная поправка листа j равна $w_j^* = R_j\ /\ (H_j + \lambda)$, где R_j — сумма остатков клиентов листа, H_j — сумма их изгибов, λ — коэффициент регуляризации. К текущему логиту z каждого клиента листа j прибавляют её долю: z + η·w_j^*, где η — шаг обучения. Как выбрать само разбиение — страница 2.

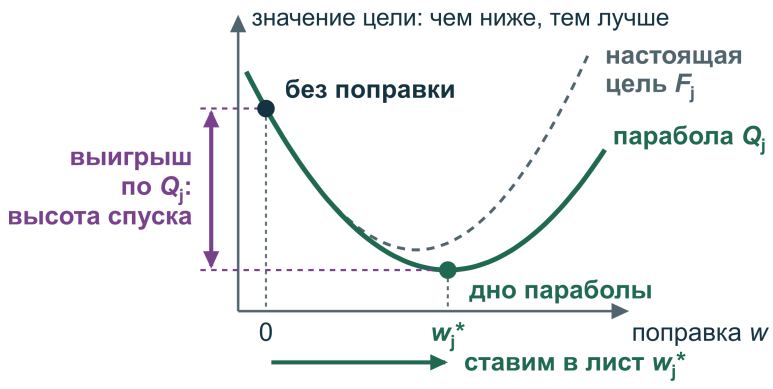
КАКОЕ РАЗБИЕНИЕ ВЫБРАТЬ: СКОР

06 Выигрыш листа: на сколько опустится парабола Q_j, если поставить в лист полную поправку w_j^*

ПОДСТАВЛЯЕМ поправку w_j^* в параболу (8) → **СМОТРИМ на параболе** → **ЗАЧЕМ это число**

$$Q_j(0) - Q_j(w_j^*) = R_j\,w_j^* - \tfrac{1}{2}(H_j+\lambda)\,w_j^{*2}$$
$$= R_j\,w_j^* - \tfrac{1}{2}R_j\,w_j^* = \tfrac{1}{2}R_j\,w_j^*$$
$$= \frac{R_j^2}{2(H_j+\lambda)} \qquad (11)$$

Из (8): $Q_j(0) = F_j(0)$, остальное — два слагаемых с w.
Во второй строке $(H_j + \lambda)\cdot w_j^*$ заменили на R_j — это уравнение дна (9).
В третьей подставили w_j^* из (10).



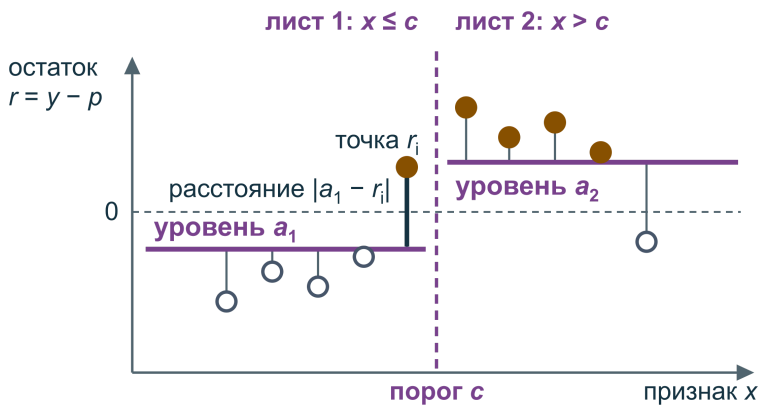
Выигрыш — высота спуска по параболе Q_j от w = 0 до её дна. У настоящей цели F_j (пунктир) в точке w = 0 те же значение, наклон и изгиб; дальше кривые расходятся.

Выигрыш (11) — на сколько опустится парабола Q_j, если поставить в лист w_j^*. Для настоящей цели F_j это оценка, а не гарантия: фактическое изменение может быть другим. Одно число оценивает пользу поправки в этом листе — по параболе Q_j.
Считается для полной поправки w_j^*, до умножения на шаг обучения η.

А КАКОЕ РАЗБИЕНИЕ ВЗЯТЬ? До сих пор разбиение было задано. Теперь перебираем кандидатов, каждому считаем одно число — скор — и берём кандидата с наибольшим скором. Скор считают по отдельному правилу: оно не выводится из выигрыша (11). Ниже два таких правила: скор L2 и скор Cosine.

07 Скор L2: насколько хорошо листья кандидата приближают остатки клиентов

ДАНО: кандидат — правило «признак ≤ порог» → **МЕРЯЕМ: уровень a вместо остатков** → **НАХОДИМ ДНО**



Кандидат — правило «признак ≤ порог» на очередном шаге роста дерева: x — значение признака у клиента, c — порог. Скор считают по листьям j = 1, 2, …, которые получатся; на рисунке — первое разбиение, два листа.
Точки — клиенты: закрашенные — банкроты, у них остаток r > 0; пустые — остальные, у них r < 0.
I_j — клиенты листа j, n_j — их число, $R_j = \Sigma\, r_i$ — сумма их остатков (5).
Фиолетовые уровни a_1, a_2 — вспомогательные числа: каждым приближаем остатки клиентов своего листа, чтобы посчитать скор. Сами остатки при сравнении кандидатов не меняются. На рисунке уровни уже найдены; как — шаг «НАХОДИМ ДНО» в этом ряду. Жирный отрезок — расстояние от точки одного клиента до уровня его листа.

$$J_j(a) = \sum_{i\in I_j}(a - r_i)^2 + \lambda a^2 \qquad (12)$$

Остаток $r_i = y_i - p_i = -g_i$, формула (5), — расхождение ответа и прогноза; он же — наклон штрафа по логиту с обратным знаком. Это не готовая поправка логита: её считают отдельно, по (10).
Формулу скора L2 принимаем как правило этого режима CatBoost; объясним её через вспомогательную цель.
a — пробный уровень: одно число, которым приближаем остатки всех клиентов листа j; уровень на рисунке.
$J_j(a)$ — вспомогательная цель листа j при уровне a: сумма квадратов расстояний от остатков до уровня плюс плата $\lambda\cdot a^2$ с тем же коэффициентом λ.
Расстояние от остатка r_i до уровня равно $|a - r_i|$: жирный отрезок на рисунке; в цели стоит его квадрат.
Меньше J_j — меньше вся эта сумма; отдельные расстояния при этом могут и вырасти.
J_j — отдельная цель: она проще, чем F_j, и из неё не выводится.

$$\tfrac{1}{2}J_j'(a) = (n_j+\lambda)\,a - R_j = 0$$
$$\Downarrow$$
$$a_j = \frac{R_j}{n_j+\lambda} \qquad (13)$$

J_j — парабола по a, ветви вверх; на дне наклон равен нулю.
a_j — уровень, при котором J_j наименьшая: лучший из пробных уровней a для листа j.
Для непустого листа a_j — среднее остатков R_j / n_j, сжатое к нулю множителем $n_j / (n_j + \lambda)$.
В пустом листе $R_j = 0$, и формула даёт $a_j = 0$.
Сравни с ответом листа (10): та же форма, но вместо суммы изгибов H_j — число клиентов n_j.

СЧИТАЕМ ВЫИГРЫШ листа → **СКЛАДЫВАЕМ по листьям: скор** → **ПОМНИМ**

$$\Delta J_j = J_j(0) - J_j(a_j) = 2a_jR_j - (n_j+\lambda)\,a_j^2$$
$$= a_jR_j = \frac{R_j^2}{n_j+\lambda} \qquad (14)$$

ΔJ_j — выигрыш листа j по вспомогательной цели: на сколько J_j на дне меньше, чем $J_j(0)$ — при нулевом уровне a = 0.
Раскрыли скобки в (12); на дне $(n_j + \lambda)\cdot a_j = R_j$.
Та же форма, что у выигрыша (11): вместо H_j — n_j, и нет множителя ½.

$$S_{L2} = \sum_j \Delta J_j = \sum_j a_jR_j \qquad (15)$$

S_L2 — скор L2 кандидата: суммарный выигрыш (14) его листьев. Сумма $J_j(0)$ по листьям — сумма квадратов всех остатков: у всех кандидатов она одна. Поэтому больше скор — меньше сумма J_j на дне: кандидат лучше по этой вспомогательной цели. Это не гарантия наибольшего снижения LogLoss.

a_j нужно только для сравнения разбиений. Когда разбиение выбрано, в лист ставят w_j^* из (10).
J_j — не LogLoss и не F_j: так устроен выбор разбиения в этом режиме CatBoost (условия — внизу страницы).

08 Скор Cosine — второй вариант скора: сравнивать направление, а не длину

СМОТРИМ на числа как на векторы → **ПЕРЕПИСЫВАЕМ скор L2 и длину вектора A** → **ЗАМЕЧАЕМ: скор L2 растёт с длиной A**

$$\mathbf{r} = (r_1,\ \ldots,\ r_N)$$
$$\mathbf{A} = (A_1,\ \ldots,\ A_N), \quad A_i = a_{j(i)}$$

N — число всех клиентов. Жирное r — набор остатков всех клиентов: вектор из N чисел. Жирное A — такой же набор из N чисел. A_i — число вектора A для клиента i: оно равно уровню a его листа. j(i) — номер листа клиента i; уровни a_j найдены в (13).

$$\mathbf{A}\cdot\mathbf{r} = \sum_{i=1}^{N} A_i\,r_i = \sum_j a_j \sum_{i\in I_j} r_i = \sum_j a_jR_j = S_{L2}$$
$$\|\mathbf{A}\|^2 = \sum_{i=1}^{N} A_i^2 = \sum_j n_j\,a_j^2$$

Группируем клиентов по листьям: в листе j у всех одно и то же a_j, а клиентов там n_j.
A·r — скалярное произведение векторов; A в двойных чертах — длина вектора A.

$$S_{L2} = \mathbf{A}\cdot\mathbf{r} = \|\mathbf{A}\|\,\|\mathbf{r}\|\cos\theta$$

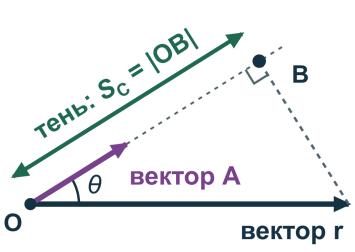
θ — угол между векторами A и r. Пока сравниваем кандидатов, модель не меняется: остатки r у всех кандидатов одни и те же, а вектор A у каждого свой. При том же направлении более длинный A даёт больший S_L2; у разных кандидатов меняются и длина, и угол.

НОРМИРУЕМ: делим на длину вектора A → **СМОТРИМ: тень** → **УДВАИВАЕМ вектор A** → **ИТОГ**

$$S_C = \frac{\mathbf{A}\cdot\mathbf{r}}{\|\mathbf{A}\|} = \frac{\sum_j a_jR_j}{\sqrt{\sum_j n_j\,a_j^2}} = \|\mathbf{r}\|\cos\theta \qquad (16)$$

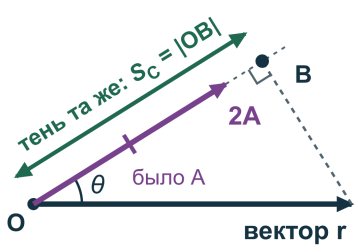
S_C — скор Cosine. Рассматриваем только ненулевые A: хотя бы одно a_j не ноль — иначе в формуле скора деление на ноль; этот случай здесь не разбираем. Длина r у всех кандидатов одна, поэтому наибольший S_C — у кандидата с наименьшим углом θ: его A точнее всех направлен вдоль остатков.

Кадр 1. Тень r на A



O — общее начало векторов r и A. B — основание перпендикуляра, опущенного из конца r на направление A. Отрезок OB — тень вектора r на направление A; её длина — длина r, умноженная на cos θ. Это и есть S_C.

Кадр 2. Удвоили A



Мысленный опыт: растянем A вдвое — стрелка 2A из того же начала O; засечка на ней — конец прежнего A. Точка B, угол θ и тень OB остались на месте — S_C не изменился, а A·r вырос вдвое.
Настоящие a_j — из (13).

Разбиение выбирают по скору: S_L2 — скор L2 (15) или S_C — скор Cosine (16). В CatBoost это параметр score_function.
Пока дерево растёт, прогнозы и остатки не меняются: сравнивают кандидатов, берут лучшее разбиение и растят дерево дальше.
Когда дерево готово, в каждый его лист ставят w_j^* — полную поправку (10), к логитам прибавляют её долю $\eta\cdot w_j^*$, где η — шаг обучения, и пересчитывают прогнозы перед следующим деревом.

Условия вывода подробно: CatBoost на CPU, boosting_type = Plain · только числовые признаки · веса клиентов равны 1 · λ > 0 · один шаг Ньютона в листе · без bootstrap и без случайной добавки к скору (random_strength = 0).