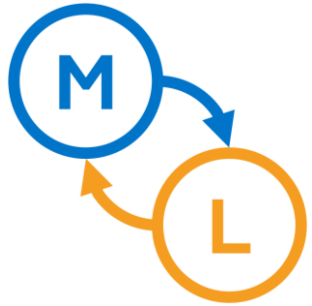
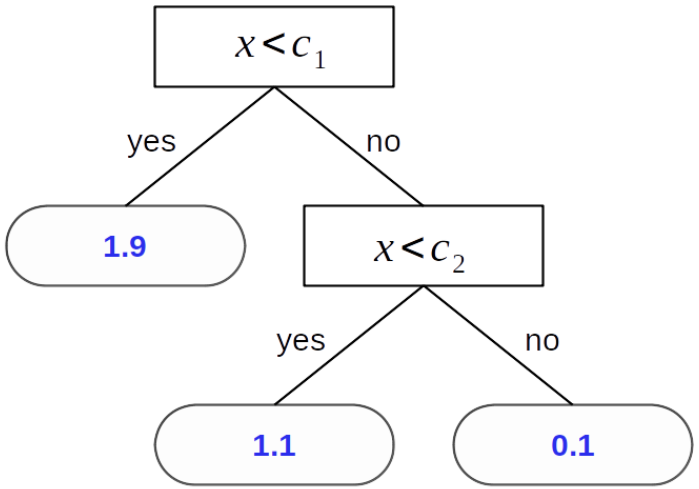


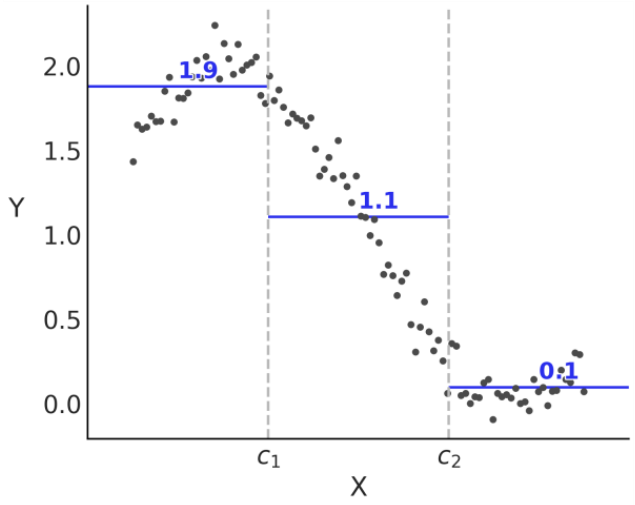
Bayesian Additive Regression Trees

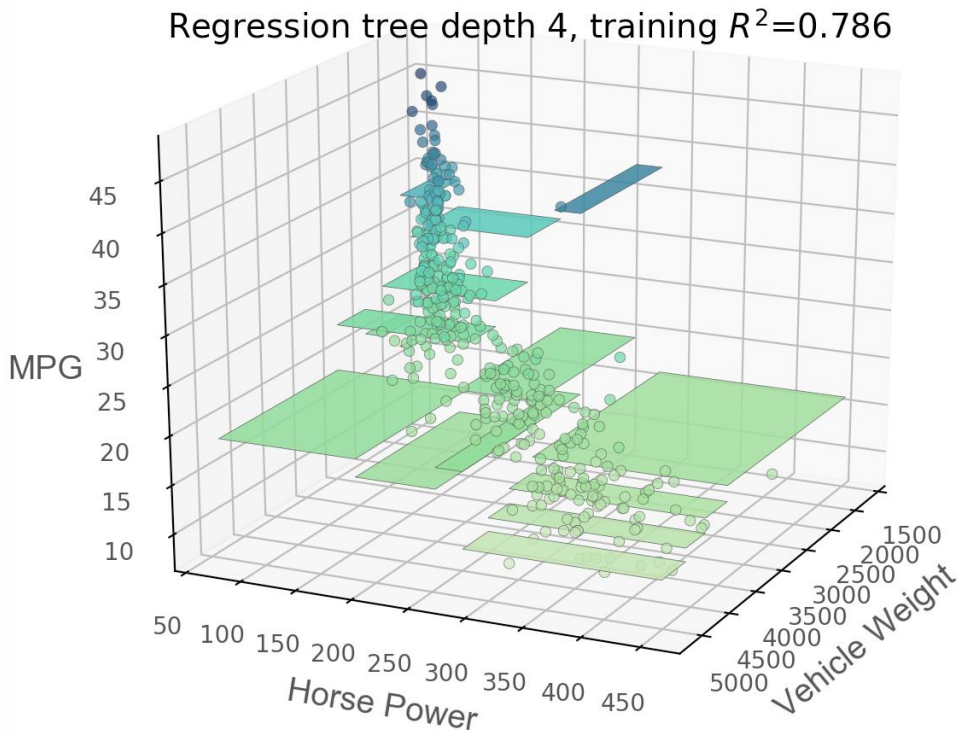
Максим Николаев

СПбГУ, 19 ноября 2025

Деревья решений

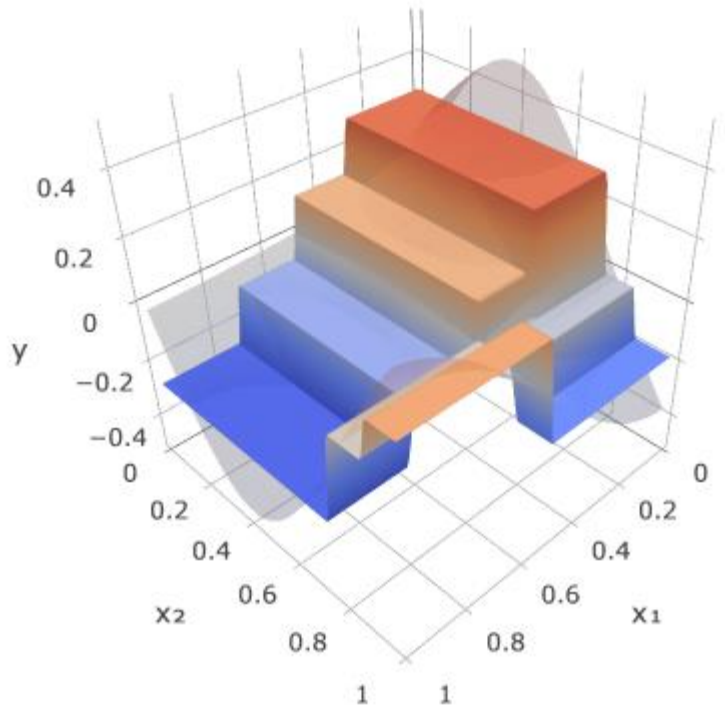




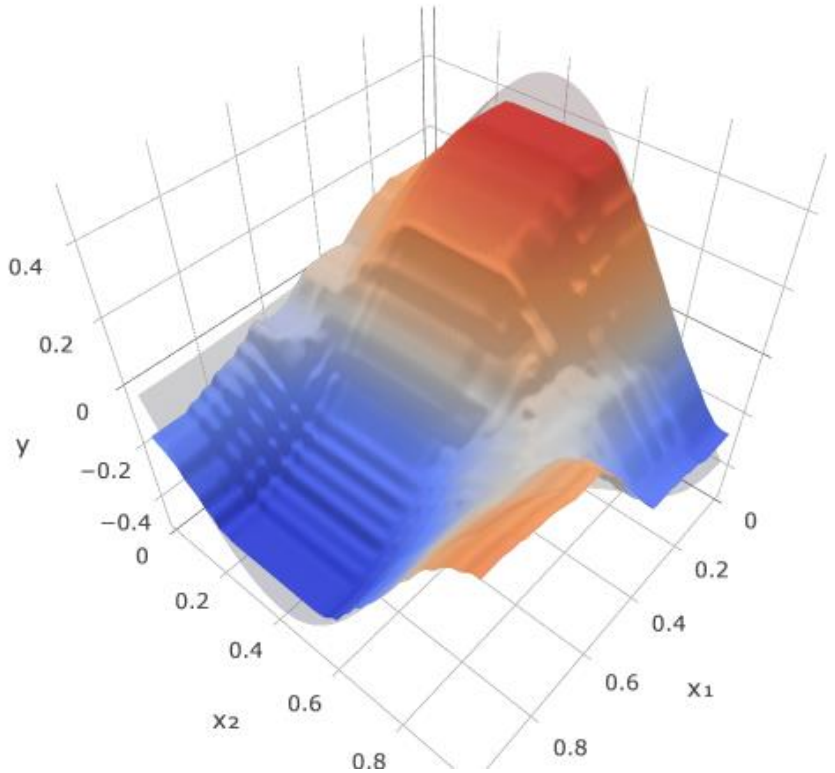


Ансамбль деревьев

1 дерево глубины 3



100 деревьев глубины 3



Bayesian Additive Regression Trees

BART, **Bayesian Additive Regression Trees** — является непараметрическим методом, который объединяет в себе лучшие черты древесных ансамблей и байесовского подхода: выразительность и оценку неопределенности.

Если кратко, то идея очень простая:

1. Задаем априорное распределение на случайном лесе, то есть распределение, из которого можно генерировать леса, совместимые с *форматом* наших данных
2. По имеющейся обучающей выборке считаем апостериорное распределение, из которого можно генерировать леса, *описывающие* наши данные
3. Прогоняем интересующие нас данные через апостериорные леса, получая апостериорное распределение предсказаний

Более формально

$$Y_i = f(X_i, Z) + \mathcal{E}_i, \qquad \mathcal{E}_i \sim \mathcal{N}(0, \sigma^2),$$

$$f(x, z) = g(x, z, T_1, M_1) + g(x, z, T_2, M_2) + \cdots + g(x, z, T_m, M_m).$$

Здесь g это функция, которая выдает значение дерева на данных, T_h описывает структуру дерева h, а $M_h = (\mu_{h1}, \ldots, \mu_{hb_h})$ описывает оценки средних в листьях дерева.

Если мы считаем m фиксированным, то для проведения байесовского вывода нам нужно задать:

- априорное распределение на T
- априорное распределение на M
- априорное распределение на σ^2

Априорное распределение на T

Априорное распределение на одном дереве состоит из трех частей:

- Вероятность того, что вершина на глубине d не является **листом**. Обычно берут в виде $\frac{\alpha}{(1+d)^\beta} \cdot [\text{мы можем разделиться на две вершины}]$, $\alpha \in (0, 1)$, $\beta > 0$
- **Распределение на множестве переменных**, по которому будет проводиться разделение в данной вершине. Обычно берут равномерное.
- **Распределение значения переменной**, в котором будет проводиться разделение. Обычно берут равномерное на значениях из данных

Априорное распределение на M

Для того, чтобы задать априорное распределение на M, делают следующее:

1. Стандартизируют переменную Y, чтобы ее значения были между -0.5 и 0.5
2. Задают априорное распределение на $\mu_i \sim \mathcal{N}\left(0, \left(\frac{0.5}{k\sqrt{m}}\right)^2\right)$. В этом случае сумма m независимых деревьев будет иметь распределение $\mathcal{N}\left(0, \left(\frac{0.5}{k}\right)^2\right)$, которое накрывает интервал $(-0.5, 0.5)$ с большой вероятностью, которую можно выбирать за счет k

Априорное распределение на σ

Для того, чтобы задать априорное распределение на σ, делают следующее:

1. Считают $RSE = \sqrt{\frac{1}{n}\sum_i e_i^2}$ линейной регрессии (e_i это остатки)
2. Задают на σ распределение Inv-Gamma так, чтобы оно было меньше RSE с большой вероятностью, например, 90%.

Аппроксимация апостериорного распределения

Как это часто бывает со сложными моделями, получить апостериорное распределение аналитически невозможно.

Вместо этого используют метод **Монте-Карло на марковских цепях** (MCMC), чтобы сгенерировать достаточно большую выборку из апостериорного распределения. Все необходимые оценки делаются по этой выборке.

Markov Chain Monte Carlo

MCMC

Мы не будем особо погружаться в теорию марковских цепей, и просто рассмотрим следующую структуру:

- Есть некоторое множество состояний $\mathfrak{X}$,
- Для каждого состояния $x \in \mathfrak{X}$ есть **распределение перехода** $T(x'|x)$ на $\mathfrak{X}$.

Имея такую структуру мы можем начать **блуждать** по $\mathfrak{X}$: стартуем в точке x_0, потом генерируем x_1 из $T(\cdot\,|x_0)$, потом генерируем x_2 из $T(\cdot\,|x_1)$ и так далее: x_{i+1} генерируется из $T(\cdot\,|x_i)$.

MCMC

Последовательность $x_0, x_1, x_2, \ldots$ называется **траекторией**. Если семейство распределений переходов обладают некоторыми специальными свойствами, то

1. Точки траектории, находящиеся далеко друг от друга, будут практически независимы.
2. Эти далекие точки будут иметь некоторое фиксированное распределение $\pi(x)$, называющееся **стационарным**.

MCMC

Последовательность $x_0, x_1, x_2, \ldots$ называется **траекторией**. Если семейство распределений переходов обладают некоторыми специальными свойствами, то

1. Точки траектории, находящиеся далеко друг от друга, будут практически независимы.
2. Эти далекие точки будут иметь некоторое фиксированное распределение $\pi(x)$, называющееся **стационарным**.

Цель MCMC — выбрать такие T, чтобы стационарное распределение было равно нужному нам распределению $p(x)$, и чтобы точки траекторий становились достаточно независимыми достаточно быстро.

Если это выполнено, то мы можем сгенерировать выборку из $p(x)$, сгенерировав длинную траекторию и потом выбрав достаточно далеко отстоящие точки.

Алгоритм Метрополиса—Гастингса

Имеются достаточные условия на T для существования единственного стационарного распределения, равного $p(x)$:

1. **Принцип детального равновесия**: $p(x)T(x'|x) = p(x')T(x|x')$
2. $T(x'|x) > 0$ для всех x' и x

Заметим, что из первого условия следует, что

$$\frac{T(x'|x)}{T(x|x')} = \frac{p(x')}{p(x)}.$$

Представим $T(x'|x)$ в виде $T(x'|x) = g(x'|x)A(x'|x)$, где $g(x'|x)$ это распределение, которое предлагает кандидатуру для очередного перехода, а $A(x'|x)$ это вероятность, с которой такой переход одобряется. Если переход не одобряется, то стоим на месте.

Алгоритм Метрополиса—Гастингса

$$\frac{T(x'|x)}{T(x|x')} = \frac{p(x')}{p(x)}.$$

Представим $T(x'|x)$ в виде $T(x'|x) = g(x'|x)A(x'|x)$, где $g(x'|x)$ это распределение, которое предлагает кандидатуру для очередного перехода, а $A(x'|x)$ это вероятность, с которой такой переход одобряется. Если переход не одобряется, то стоим на месте.

Получаем

$$\frac{A(x'|x)}{A(x|x')} = \frac{p(x')}{p(x)}\frac{g(x|x')}{g(x'|x)}.$$

Заметим, что если взять $A(x'|x) = \min\left(1, \frac{p(x')}{p(x)}\frac{g(x|x')}{g(x'|x)}\right)$, то все получится!

Алгоритм Метрополиса—Гастингса

$$\frac{T(x'|x)}{T(x|x')} = \frac{p(x')}{p(x)}.$$

Представим $T(x'|x)$ в виде $T(x'|x) = g(x'|x)A(x'|x)$, где $g(x'|x)$ это распределение, которое предлагает кандидатуру для очередного перехода, а $A(x'|x)$ это вероятность, с которой такой переход одобряется. Если переход не одобряется, то стоим на месте.

Получаем

$$\frac{A(x'|x)}{A(x|x')} = \frac{p(x')}{p(x)}\frac{g(x|x')}{g(x'|x)}.$$

Заметим, что если взять $A(x'|x) = \min\left(1, \frac{p(x')}{p(x)}\frac{g(x|x')}{g(x'|x)}\right)$, то все получится!

Нам достаточно знать $p(x)$ с точностью до множителя!

Алгоритм Метрополиса—Гастингса

Алгоритм:

1. Выбираем x_0
2. Пока не надоест:
    1. Генерируем x' из $g(\cdot\,|x_i)$
    2. Считаем $A = \min\left(1, \frac{p(x')}{p(x_i)}\frac{g(x_i|x')}{g(x'|x_i)}\right)$
    3. С вероятностью A берем $x_{i+1} = x'$, а с вероятностью $1 - A$ берем $x_{i+1} = x_i$.

Если $g(x|x') = g(x'|x)$, то A считать еще проще: $A = \min\left(1, \frac{p(x')}{p(x)}\right)$

Сэмплирование по Гиббсу

В многомерных пространствах сложно предлагать хорошие новые состояния для всего вектора $x = (x_1, \dots, x_n)$ сразу.

Решение: итеративно обновлять по одной переменной за раз, фиксируя все остальные. Это частный случай Метрополиса—Гастингса, в котором в качестве g выступает распределение $p(\cdot \mid x_{-i})$, где i может быть случайным, а может последовательно обходить все координаты. Распределение перехода не зависит от текущего значения x_i, и нетрудно показать, что в этом случае A всегда будет равно 1.

Когда применяется: совместное распределение случайных величин неизвестно явно, но условные вероятности известны и из них легко генерировать.

Сэмплирование для BART

Сэмплирование из n деревьев на верхнем уровне является сэмплированием по Гиббсу: мы проходим по деревьям по очереди и сэмплируем дерево из условного распределения. Это сэмплирование можно проводить разными способами, например, с помощью еще одного Метрополиса—Гастингса.

Algorithm 1 Bayesian backfitting MCMC for posterior inference in BART

Inputs: Training data $(\boldsymbol{X}, Y)$, BART hyperparameters $(\nu, q, k, m, \alpha_s, \beta_s)$
Initialization: For all j, set $\mathcal{T}_j^{(0)} = \{\mathsf{T}_j^{(0)} = \{\epsilon\}, \boldsymbol{\tau}_j^{(0)} = \boldsymbol{\kappa}_j^{(0)} = \emptyset\}$ and sample $\boldsymbol{\mu}_j^{(0)}$
for $i = 1 :$ max_iter **do**
    Sample $\sigma^{2(i)} | \mathcal{T}_{1:m}^{(i-1)}, \boldsymbol{\mu}_{1:m}^{(i-1)}$ ▷ *sample from inverse gamma distribution*
    for $j = 1 : m$ **do**
        Compute residual $R_j^{(i)}$ $\quad R_j = Y - \sum_{j'=1, j' \neq j}^{m} g(\boldsymbol{X}; \mathcal{T}_{j'}, \mu_{j'})$.
        Sample $\mathcal{T}_j^{(i)} | R_j^{(i)}, \sigma^{2(i)}, \mathcal{T}_j^{(i-1)}$ ▷ *using CGM, GrowPrune or PG*
        Sample $\boldsymbol{\mu}_j^{(i)} | R_j^{(i)}, \sigma^{2(i)}, \mathcal{T}_j^{(i)}$ ▷ *sample from Gaussian distribution*

Направления для исследований

1. Другие структуры деревьев. Например, интересно посмотреть, что будет в случае таблиц решений — полных двоичных деревьев, у которых на каждом уровне стоит один и тот же предикат
2. Асимптотическое поведение. При возрастании числа деревьев BART сходится к гауссовскому процессу.
3. Эффективные алгоритмы для сэмплирования.