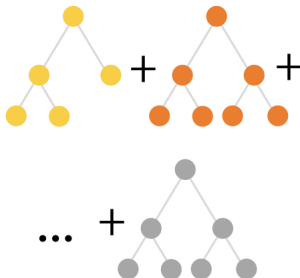


Бустинг

Виктор Китов

victorkitov.github.io



Курс поддержан
фондом
'Интеллект'



Победитель
конкурса VK среди
курсов по IT



Бустинг¹

Строится линейная композиция:

$$G_M(x) = f_0(x) + c_1 f_1(x) + \dots + c_M f_M(x)$$

Регрессия: $\hat{y}(x) = G_M(x)$

Классификация: $g_y(x) = G_M(x)$, $\hat{y}(x) = \text{sign } G_M(x)$

- $f_1(x), \dots, f_M(x)$ - базовые модели (base learners, base models).
- Сложно оптимизировать $f_0(x), f_1(x), \dots, f_M(x)$ и c_1, \dots, c_M одновременно.
- Упрощение: настраиваем $f_0(x)$, потом последовательно $(f_m(x), c_m)$ для $m = 1, 2, \dots, M$.

¹Какую модель получим, если применим бустинг к линейным моделям?

Алгоритм бустинга

- 1 Настраиваем начальное приближение

$$f_0(x) = \arg \min_f \sum_{n=1}^N \mathcal{L}(f(x_n), y_n)$$

- 2 Для $m = 1, 2, \dots, M$:

- находим коррекцию:

$$(c_m, f_m) := \arg \min_{f, c} \sum_{n=1}^N \mathcal{L}(G_{m-1}(x_n) + cf(x_n), y_n)$$

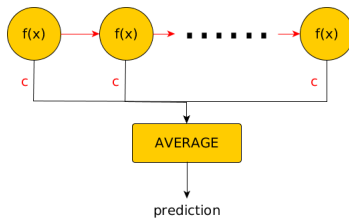
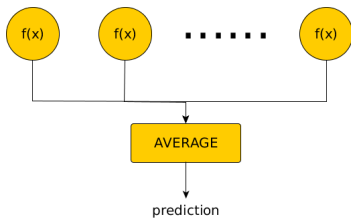
- обновляем ансамбль:

$$G_m(x) := G_{m-1}(x) + c_m f_m(x)$$

- 3 Возвращаем $G_M(x)$.

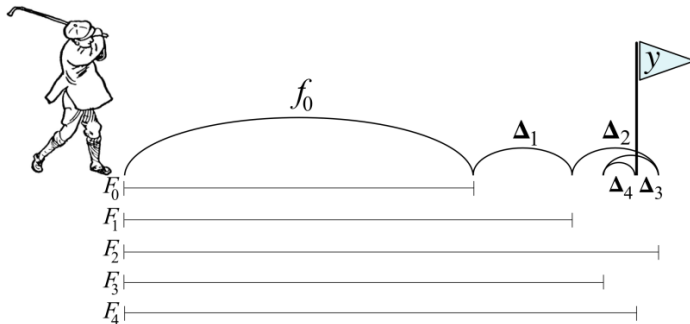
Бэггинг и бустинг

Бэггинг и бустинг



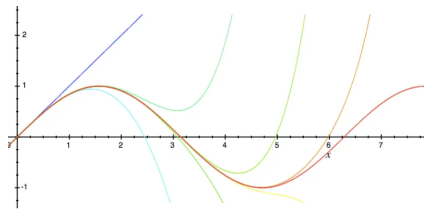
Аналогия с игрой в гольф

Аналогия с игрой в гольф



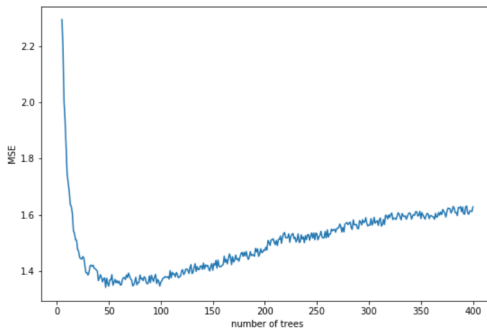
Аналогия с разложением в ряд Тейлора

Аналогия с разложением в ряд Тейлора



Зависимость от M

Зависимость средних потерь от M :



M контролирует сложность ансамбля. Настраиваем #базовых моделей стратегией ранней остановки (early stopping), когда качество на валидации стало уменьшаться.

Комментарии

- Точнее - усреднение по большому числу $f_m(x)$.
- Каждый $f_m(x)$ должен быть простым, чтобы оставлять возможности последующим моделям.
 - $f_0(x) \equiv 0$ или $f_0(x) \equiv \text{const}$
 - настройка $f_m(x)$ может быть неточной
 - $f_m(x)$ должны быть простыми моделями

Содержание

- 1 Adaboost
- 2 Градиентный бустинг
- 3 Расширения
- 4 Градиентный бустинг деревьев

Adaboost (дискретная версия)

Предположения:

- бинарная классификация $y \in \{+1, -1\}$
- функция потерь $\mathcal{L}(G(x), y) = e^{-yG(x)}$
 - неустойчива к выбросам, их нужно фильтровать по весам
- $f_m(x) \in \{+1, -1\}$, способны настраиваться на взвешенной обучающей выборке.
- настраивается $\hat{y}(x) = \text{sign} \left(\sum_{m=1}^M c_m f_m(x) \right)$
- M - внешний параметр.

Возможно аналитическое решение.

Adaboost (дискретная версия): алгоритм

- ❶ Инициализируем веса объектов $w_n = 1/N$, $n = 1, 2, \dots, N$.
- ❷ Для каждого $m = 1, 2, \dots, M$:
 - ❶ настраиваем $f_m(x)$ по выборке с весами $\{w_n\}_{n=1}^N$
 - ❷ вычисляем взвешенную частоту ошибок:

$$E_m = \frac{\sum_{n=1}^N w_n \mathbb{I}[f_m(x_n) \neq y_n]}{\sum_{n=1}^N w_n}$$

- ❸ если $E_m > 0.5$ либо $E_m = 0$: останавливаем построение ансамбля.
- ❹ вычисляем $c_m = \frac{1}{2} \ln((1 - E_m)/E_m)$ $E_m < 0.5 \Rightarrow c_m > 0$
- ❺ увеличиваем веса объектов, где $f_m(x)$ ошиблась:

$$w_n \leftarrow w_n e^{2c_m} = w_n \left(\frac{1 - E_m}{E_m} \right), \text{ для } n : f_m(x_n) \neq y_n$$

Вывод Adaboost

Задаем $G_0(x) \equiv 0$.

Для каждого $m = 1, 2, \dots, M$:

$$\begin{aligned}(c_m, f_m) &= \arg \min_{c_m, f_m} \sum_{n=1}^N \mathcal{L}(G_{m-1}(x_n) + c_m f_m(x_n), y_n) \\&= \arg \min_{c_m, f_m} \sum_{n=1}^N e^{-y_n G_{m-1}(x_n)} e^{-c_m y_n f_m(x_n)} \\&= \arg \min_{c_m, f_m} \sum_{i=1}^N w_n^m e^{-c_m y_n f_m(x_n)}, \quad w_n^m := e^{-y_n G_{m-1}(x_n)}\end{aligned}$$

Вывод Adaboost

$$\frac{\partial L(c_m)}{\partial c_m} = - \sum_{n=1}^N w_n^m e^{-c_m y_n f_m(x_n)} y_n f_m(x_n) = 0$$

$$- \sum_{n: f_m(x_n)=y_n} w_n^m e^{-c_m} + \sum_{n: f_m(x_n) \neq y_n} w_n^m e^{c_m} = 0$$

$$e^{2c_m} = \frac{\sum_{n: f_m(x_n)=y_n} w_n^m}{\sum_{n: f_m(x_n) \neq y_n} w_n^m}$$

$$c_m = \frac{1}{2} \ln \frac{\left(\sum_{n: f_m(x_n)=y_n} w_n^m \right) / \left(\sum_{n=1}^N w_n^m \right)}{\left(\sum_{n: f_m(x_n) \neq y_n} w_n^m \right) / \left(\sum_{n=1}^N w_n^m \right)} = \frac{1}{2} \ln \frac{1 - E_m}{E_m} > 0,$$

где

Вывод Adaboost

$$\begin{aligned}
 \sum_{n=1}^N w_n^m e^{-c_m y_n f_m(x_n)} &= \sum_{n: f_m(x_n)=y_n} w_n^m e^{-c_m} + \sum_{n: f_m(x_n) \neq y_n} w_n^m e^{c_m} \\
 &= e^{-c_m} \sum_{n: f_m(x_n)=y_n} w_n^m + e^{c_m} \sum_{n: f_m(x_n) \neq y_n} w_n^m \\
 &= e^{-c_m} \sum_{n=1}^N w_n^m - e^{-c_m} \sum_{n: f_m(x_n) \neq y_n} w_n^m + e^{c_m} \sum_{n: f_m(x_n) \neq y_n} w_n^m \\
 &= e^{-c_m} \sum_n w_n^m + (e^{c_m} - e^{-c_m}) \sum_{n: f_m(x_n) \neq y_n} w_n^m
 \end{aligned}$$

Поскольку $c_m > 0$, $f_m(\cdot)$ находится из условия

$$f_m(\cdot) = \arg \min_f \sum_{n=1}^N w_n^m \mathbb{I}[f(x_n) \neq y_n]$$

Вывод Adaboost

Пересчет весов:

$$w_n^{m+1} \stackrel{\text{def}}{=} e^{-y_n G_m(x_n)} = e^{-y_n G_{m-1}(x_n)} e^{-y_n c_m f_m(x_n)} = w_n^{m+1} e^{-y_n c_m f_m(x_n)}$$

Т.к. $y_n f_m(x_n) = 1 - 2\mathbb{I}[f_m(x_n) \neq y_n]$, получим:

$$\begin{aligned} w_n^{m+1} &= e^{-y_n G_{m-1}(x_n)} e^{c_m(2\mathbb{I}[f_m(x_n) \neq y_n] - 1)} = w_n^m e^{2c_m \mathbb{I}[f_m(x_n) \neq y_n]} e^{-c_m} \\ &\propto w_n^m e^{2c_m \mathbb{I}[f_m(x_n) \neq y_n]} = w_n^m \left(\frac{1 - E_m}{E_m} \right)^{\mathbb{I}[f_m(x_n) \neq y_n]} \geq w_n^m \end{aligned}$$

- Веса нормируются \Rightarrow сократили общий множитель.
- $w_n^{m+1} = w_n^m$ для $n : f_m(x_n) = y_n$.
- $w_n^{m+1} > w_n^m$ для $n : f_m(x_n) \neq y_n$.
 - последующие модели уделят объектам повышенное внимание

Использование AdaBoost

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import brier_score_loss

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
model = AdaBoostClassifier(n_estimators=50)
# для регрессии есть AdaBoostRegressor
model.fit(X_train, Y_train) # обучение модели
Y_hat = model.predict(X_test) # построение прогнозов
print(f'Точность прогнозов: \
      {100*accuracy_score(Y_test, Y_hat):.1f}%')

P_hat = model.predict_proba(X_test)
loss = brier_score_loss(Y_test, P_hat[:,1])
print(f'Ошибка Бриера: {loss:.2f}')
```

Больше информации. Полный код.

Содержание

- 1 Adaboost
- 2 Градиентный бустинг
- 3 Расширения
- 4 Градиентный бустинг деревьев

Мотивация

- Проблема: для ϕ -ции потерь общего вида пересчет модели/весов не может быть решен аналитически.
- Аналогия с минимизацией ϕ -ций: если нет аналитического решения, находим численное решение
- Градиентный бустинг: аналогия градиентного спуска в пространстве функций.

Интуиция метода

- Решаем задачу подбора $f_{m+1}(x)$ (пока без c_m):

$$\mathcal{L}(f_m) = \frac{1}{N} \sum_{n=1}^N \mathcal{L}(G_m(x_n) + f_m(x_n), y_n) \rightarrow \min_{f_m}$$

- Если $[f_m(x_1), f_m(x_2), \dots, f_m(x_N)] \rightarrow [u_1, u_2, \dots, u_N] \in \mathbb{R}^N$:

$$\mathcal{L}(u) = \mathcal{L}(u_1, \dots, u_N) = \frac{1}{N} \sum_{n=1}^N \mathcal{L}(G_m(x_n) + u_n, y_n) \rightarrow \min_{u_1, u_2, \dots, u_N}$$

- В линейном приближении можно решить, положив

$$u_1 = \frac{\partial \mathcal{L}(G_m(x_1), y_1)}{\partial G}; \dots \quad u_N = \frac{\partial \mathcal{L}(G_m(x_N), y_N)}{\partial G}$$

- Следовательно, следует выбирать $f_m(x)$ так, чтобы

$$f_m(x_1) \approx \frac{\partial \mathcal{L}(G_m(x_1), y_1)}{\partial G}; \dots \quad f_m(x_N) \approx \frac{\partial \mathcal{L}(G_m(x_N), y_N)}{\partial G}$$

Интуиция метода

- Шагу градиентного спуска при минимизации $\mathcal{L}(u)$:

$$u := u - \varepsilon \nabla \mathcal{L}(u)$$

будет приближённо соответствовать обновлению ансамбля:

$$G_{m+1}(x) := G_m(x) + \varepsilon f_m(x)$$

- $\nabla \mathcal{L}(u) \in \mathbb{R}^N$ соответствует $[f_m(x_1), f_m(x_2), \dots, f_m(x_N)]$.
- ε - шаг обучения (learning rate).

Локальная линейная аппроксимация

Линейная аппроксимация \mathcal{L} с $g(x) = \left. \frac{\partial \mathcal{L}(G, y)}{\partial G} \right|_{G=G(x)}$:

Локальная линейная аппроксимация

Линейная аппроксимация \mathcal{L} с $g(x) = \left. \frac{\partial \mathcal{L}(G, y)}{\partial G} \right|_{G=G(x)}$:

$$\mathcal{L}(G(x) + f(x), y) \approx \mathcal{L}(G(x), y) + g(x)f(x)$$

$$\arg \min_{f(x)} \sum_{n=1}^N \mathcal{L}(G(x_n) + f(x_n), y_n)$$

$$\approx \arg \min_{f(x)} \sum_{n=1}^N \mathcal{L}(G(x_n), y_n) + g(x_n)f(x_n)$$

$$= \arg \min_{f(x)} \sum_{n=1}^N g(x_n)f(x_n) = \arg \max_{f(x)} \sum_{n=1}^N g(x_n)f(x_n)$$

$\Rightarrow f(x)$ должна настраиваться на $-g(x)$, т.к.

$$\arg \min_{f: \|f\| \leq \|g\|} \langle f, g \rangle = -g$$

Пример: регрессия

$$\sum_{n=1}^N \left(f_m(x_n) + \frac{\partial \mathcal{L}(G, y_n)}{\partial G} \Big|_{G=G_{m-1}(x_n)} \right)^2 \rightarrow \min_{f_m}$$

Пример: регрессия

$$\sum_{n=1}^N \left(f_m(x_n) + \frac{\partial \mathcal{L}(G, y_n)}{\partial G} \Big|_{G=G_{m-1}(x_n)} \right)^2 \rightarrow \min_{f_m}$$

$$\mathcal{L} = \frac{1}{2} (G - y)^2 : f(x) \approx -\frac{\partial \mathcal{L}(G, y)}{\partial G} = -(G - y)$$

$$G_m(x_n) := G_{m-1}(x_n) + \varepsilon f(x) \approx G_{m-1}(x_n) + \varepsilon (y_n - G_{m-1}(x_n))$$

Пример: классификация

$$\sum_{n=1}^N \left(f_m(x_n) + \frac{\partial \mathcal{L}(G, y_n)}{\partial G} \Big|_{G=G_{m-1}(x_n)} \right)^2 \rightarrow \min_{f_m}$$

Пример: классификация

$$\sum_{n=1}^N \left(f_m(x_n) + \frac{\partial \mathcal{L}(G, y_n)}{\partial G} \Big|_{G=G_{m-1}(x_n)} \right)^2 \rightarrow \min_{f_m}$$

$$\mathcal{L} = [-Gy]_+ : f_m(x) \approx -\frac{\partial \mathcal{L}(G, y)}{\partial G} = \begin{cases} y, & Gy < 0 \\ 0, & Gy \geq 0 \end{cases}$$

$$G_m(x_n) := G_{m-1}(x_n) + \varepsilon f_m(x) \approx G_{m-1}(x_n) + \begin{cases} \varepsilon y_n, & G(x_n)y_n < 0 \\ 0, & G(x_n)y_n \geq 0 \end{cases}$$

Алгоритм градиентного бустинга

Вход: обучающая выборка X , $Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

- 1 Настраиваем начальную модель $G_0(x)$ по X, Y .

Алгоритм градиентного бустинга

Вход: обучающая выборка X , $Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

- 1 Настраиваем начальную модель $G_0(x)$ по X, Y .
- 2 Для каждого $m = 1, 2, \dots, M$:

Алгоритм градиентного бустинга

Вход: обучающая выборка X , $Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

- ❶ Настраиваем начальную модель $G_0(x)$ по X, Y .
- ❷ Для каждого $m = 1, 2, \dots, M$:
 - ❶ вычисляем градиенты: $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$

Алгоритм градиентного бустинга

Вход: обучающая выборка $X, Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

- ❶ Настраиваем начальную модель $G_0(x)$ по X, Y .
- ❷ Для каждого $m = 1, 2, \dots, M$:
 - ❶ вычисляем градиенты: $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$
 - ❷ настраиваем $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, например

$$\sum_{n=1}^N (f_m(x_n) + g_n)^2 \rightarrow \min_{f_m}$$

Алгоритм градиентного бустинга

Вход: обучающая выборка $X, Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

❶ Настраиваем начальную модель $G_0(x)$ по X, Y .

❷ Для каждого $m = 1, 2, \dots, M$:

❶ вычисляем градиенты: $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$

❷ настраиваем $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, например

$$\sum_{n=1}^N (f_m(x_n) + g_n)^2 \rightarrow \min_{f_m}$$

❸ настраиваем шаг (в sklearn - константный):

$$\varepsilon_m = \arg \min_{\varepsilon > 0} \sum_{n=1}^N \mathcal{L}(G_{m-1}(x_n) + \varepsilon f_m(x_n), y_n)$$

Алгоритм градиентного бустинга

Вход: обучающая выборка $X, Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

❶ Настраиваем начальную модель $G_0(x)$ по X, Y .

❷ Для каждого $m = 1, 2, \dots, M$:

❶ вычисляем градиенты: $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$

❷ настраиваем $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, например

$$\sum_{n=1}^N (f_m(x_n) + g_n)^2 \rightarrow \min_{f_m}$$

❸ настраиваем шаг (в sklearn - константный):

$$\varepsilon_m = \arg \min_{\varepsilon > 0} \sum_{n=1}^N \mathcal{L}(G_{m-1}(x_n) + \varepsilon f_m(x_n), y_n)$$

❹ обновляем $G_m(x) = G_{m-1}(x) + \varepsilon_m f_m(x)$

Алгоритм градиентного бустинга

Вход: обучающая выборка $X, Y = \{(x_n, y_n)\}_{n=1}^N$; функция потерь $\mathcal{L}(f, y)$ и число базовых моделей M .

❶ Настраиваем начальную модель $G_0(x)$ по X, Y .

❷ Для каждого $m = 1, 2, \dots, M$:

❶ вычисляем градиенты: $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$

❷ настраиваем $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, например

$$\sum_{n=1}^N (f_m(x_n) + g_n)^2 \rightarrow \min_{f_m}$$

❸ настраиваем шаг (в sklearn - константный):

$$\varepsilon_m = \arg \min_{\varepsilon > 0} \sum_{n=1}^N \mathcal{L}(G_{m-1}(x_n) + \varepsilon f_m(x_n), y_n)$$

❹ обновляем $G_m(x) = G_{m-1}(x) + \varepsilon_m f_m(x)$

Выход: композиция $G_M(x)$.

Содержание

- 1 Adaboost
- 2 Градиентный бустинг
- 3 Расширения**
- 4 Градиентный бустинг деревьев

Сжатие, обучение на подвыборках

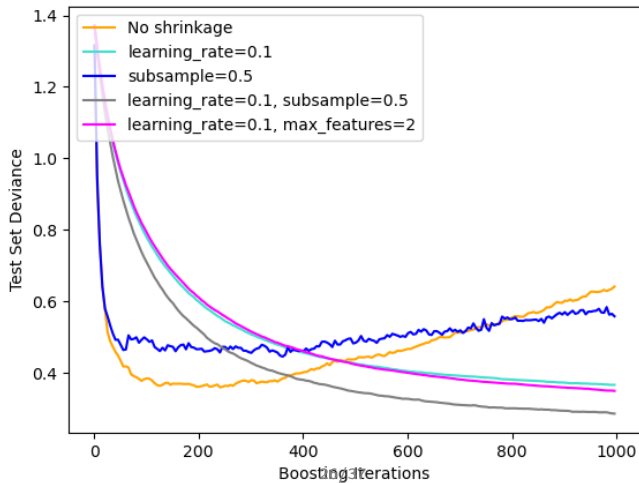
- Перенастройка линейной регрессией весов по признакам $f_1(x), \dots, f_M(x)$.
- Сжатие шага (shrinkage):

$$G_m(x) = G_{m-1}(x) + \alpha \varepsilon_m f_m(x)$$

- $\alpha \downarrow \implies M \uparrow (\alpha M \approx \text{const})$
- $\alpha \in (0, 1]$ - искусственно увеличиваем $\#$ шагов и $\#$ базовых моделей для \uparrow точности.
- настраиваем все параметры, потом используем α и $M := M/\alpha$.
- Обучение $f_m(x)$ на подвыборках (subsampling)
 - объекты и признаки сэмплятся без возвращения
 - ускоряет настройку
 - повышает разнообразие $f_m(x)$ и точность композиции.

Эффект сжатия и обучения на подвыборках

Эффект сжатия и обучения на подвыборках



Локальная квадратичная аппроксимация

Квадратичная аппроксимация \mathcal{L} , используя $g(x) = \frac{\partial \mathcal{L}(G(x), y)}{\partial G}$,
 $h(x) = \frac{\partial^2 \mathcal{L}(G(x), y)}{\partial G^2}$:

Локальная квадратичная аппроксимация

Квадратичная аппроксимация \mathcal{L} , используя $g(x) = \frac{\partial \mathcal{L}(G(x), y)}{\partial G}$,
 $h(x) = \frac{\partial^2 \mathcal{L}(G(x), y)}{\partial G^2}$:

$$\mathcal{L}(G(x) + f(x), y) \approx \mathcal{L}(G(x), y) + g(x)f(x) + \frac{1}{2}h(x)(f(x))^2 =$$

$$\frac{1}{2}h(x) \left(f(x) + \frac{g(x)}{h(x)} \right)^2 + \text{const}(f(x))$$

$$\arg \min_{f(x)} \sum_{n=1}^N \frac{1}{2} h(x_n) \left(f(x_n) + \frac{g(x_n)}{h(x_n)} \right)^2 + \text{const}(f(x))$$

$$= \arg \min_{f(x)} \sum_{n=1}^N h(x_n) \left(f(x_n) + \frac{g(x_n)}{h(x_n)} \right)^2$$

Следовательно $f(x_n) \approx -g(x_n)/h(x_n)$ с весом $h(x_n)$.

- $h(x) \geq 0$ в окрестности локального минимума L .

Содержание

- 1 Adaboost
- 2 Градиентный бустинг
- 3 Расширения
- 4 Градиентный бустинг деревьев

Преимущества решающих деревьев

Преимущества решающих деревьев:

- нелинейная модель с гибкой настройкой сложности
 - по глубине и др. критериям
- вычислительная эффективность прогнозов
- встроенный обзор признаков
- инвариантны к масштабу признаков
- инвариантны к монотонным преобразованиям признаков
- обладают универсальной применимостью к признакам разной природы
 - бинарные, вещественные, порядковые категориальные
 - категориальные \rightarrow бинарные (one-hot) или вещественные (mean-value encoding)
 - категориальные \rightarrow порядковые, упорядочив категории по \bar{y} при условии категории
- позволяют вычислять важность признаков

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

- 1 вычисляем градиенты $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

- 1 вычисляем градиенты $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$
- 2 настраиваем решающее дерево $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$,
получаем разбиение пр-ва признаков $\{R_k\}_{k=1}^K$.

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

- 1 вычисляем градиенты $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$
- 2 настраиваем решающее дерево $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$,
получаем разбиение пр-ва признаков $\{R_k\}_{k=1}^K$.
- 3 для каждого прямоугольника R_k , $k = 1, 2, \dots, K$
пересчитываем прогнозы:

$$\gamma_k = \arg \min_{\gamma > 0} \sum_{x_n \in R_k} \mathcal{L}(F_{m-1}(x_n) + \gamma, y_n)$$

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и # базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

- 1 вычисляем градиенты $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$
- 2 настраиваем решающее дерево $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, получаем разбиение пр-ва признаков $\{R_k\}_{k=1}^K$.
- 3 для каждого прямоугольника R_k , $k = 1, 2, \dots, K$ пересчитываем прогнозы:

$$\gamma_k = \arg \min_{\gamma > 0} \sum_{x_n \in R_k} \mathcal{L}(F_{m-1}(x_n) + \gamma, y_n)$$

- 4 обновляем $G_m(x) := G_{m-1}(x) + \sum_{k=1}^K \gamma_k \mathbb{I}[x \in R_k]$

Градиентный бустинг деревьев

Вход: обучающая выборка (x_n, y_n) , $n = 1, 2, \dots, N$; ф-ция потерь $\mathcal{L}(f, y)$ и $\#$ базовых моделей M .

- 1 Начальная аппроксимация-константа:

$$G_0(x) = \arg \min_{\gamma} \sum_{n=1}^N \mathcal{L}(\gamma, y_n)$$

- 2 Для каждого $m = 1, 2, \dots, M$:

- 1 вычисляем градиенты $g_n = \frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G}$
- 2 настраиваем решающее дерево $f_m(\cdot)$ на $\{(x_n, -g_n)\}_{n=1}^N$, получаем разбиение пр-ва признаков $\{R_k\}_{k=1}^K$.
- 3 для каждого прямоугольника R_k , $k = 1, 2, \dots, K$ пересчитываем прогнозы:

$$\gamma_k = \arg \min_{\gamma > 0} \sum_{x_n \in R_k} \mathcal{L}(F_{m-1}(x_n) + \gamma, y_n)$$

- 4 обновляем $G_m(x) := G_{m-1}(x) + \sum_{k=1}^K \gamma_k \mathbb{I}[x \in R_k]$

Выход: композиция $G_M(x)$.

Градиентный бустинг деревьев

- Индивидуальная настройка прогноза в каждом R_j^m .
 - подбирать общий множитель ε_m уже не нужно
 - повышает гибкость настройки
- Охватываем взаимодействие K признаков, если
 - макс. глубина дерева K :
 - либо макс. # листьев $K + 1$
- Обычно подбирают $2 \leq K \leq 8$ по валидации.
- Англ. gradient boosting on decision trees (GBDT).
- Один из самых точных алгоритмов для неоднородных данных (признаки разной природы)
- Для однородных данных (текст, звук, изображения, видео)
- нейросети, обычно, лучше.

Случай $y \in \{1, 2, \dots, C\}$

Используем обобщение: один-против-всех, один-против-одного, коды, исправляющие ошибки.

Случай $y \in \{1, 2, \dots, C\}$

Используем обобщение: один-против-всех, один-против-одного, коды, исправляющие ошибки.

Альтернативно, можно оптимизировать $\mathcal{L}(G(x), y)$, $G(x) \in \mathbb{R}^C$.

Пример:

- $G(x) = \{p(y = c|x)\}_{c=1}^C$, y - one-hot закодированный класс,
- Решаем задачу C -мерной регрессии:
 - Минимизация потерь $\mathcal{L}(\cdot)$:

$$f_m(x_n) \approx -\frac{\partial \mathcal{L}(G_{m-1}(x_n), y_n)}{\partial G} \in \mathbb{R}^C$$

Максимизация качества $S(\cdot)$:

$$f_m(x_n) \approx \frac{\partial S(G_{m-1}(x_n), y_n)}{\partial G} \in \mathbb{R}^C$$

Случай $y \in \{1, 2, \dots, C\}$

$$S(G_{m-1}(x), y) = p(y|x) = \sum_{c=1}^C \mathbb{I}[y = c] p_c$$

$$f_m(x_n) \approx \begin{pmatrix} \mathbb{I}[y = 1] \\ \mathbb{I}[y = 2] \\ \dots \\ \mathbb{I}[y = C] \end{pmatrix}$$

$$S(G_{m-1}(x), y) = \ln p(y|x) = \sum_{c=1}^C \mathbb{I}[y = c] \ln p_c$$

$$f_m(x_n) \approx \begin{pmatrix} \frac{1}{p_1} \mathbb{I}[y = 1] \\ \frac{1}{p_2} \mathbb{I}[y = 2] \\ \dots \\ \frac{1}{p_C} \mathbb{I}[y = C] \end{pmatrix}$$

Использование градиентного бустинга (регрессия)

```
from sklearn.ensemble import
    GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error

X_train, X_test, Y_train, Y_test =
    get_demo_regression_data()
# инициализация (базовые модели—всегда деревья)
model = GradientBoostingRegressor(
    n_estimators=1000, # число базовых моделей
    learning_rate=0.1, # шаг обучения
    subsample=1.0,      # доля объектов для обучения
    max_features=1.0)   # доля признаков для обучения
model.fit(X_train, Y_train) # обучение модели
Y_hat = model.predict(X_test) # прогнозирование
print(f'Средний модуль ошибки (MAE): \
    {mean_absolute_error(Y_test, Y_hat):.2f}')
```

Использование градиентного бустинга (класс-ция)

```
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import brier_score_loss

X_train, X_test, Y_train, Y_test =
    get_demo_classification_data()
model = GradientBoostingClassifier(n_estimators=1000,
    learning_rate=0.1, subsample=1.0, max_features=1.0)
model.fit(X_train, Y_train) # обучение модели
Y_hat = model.predict(X_test) # построение прогнозов
print(f'Точность прогнозов: {100*accuracy_score(Y_test,
    Y_hat):.1 f}%')

P_hat = model.predict_proba(X_test) # вер-ти классов
loss = brier_score_loss(Y_test, P_hat[:,1])
print(f'Средняя ошибка прогноза вероятностей (по мере
    Бриера): {loss:.2 f}%')
```

Больше информации. [Полный код.](#)

Продвинутые реализации бустинга

- Продвинутые реализации бустинга:
 - CatBoost
 - разработано Яндексом, документация на русском
 - специальная обработка категориальных признаков
 - xgBoost
 - аппроксимация 2го порядка, дискретизация признаков
 - гибкая регуляризация деревьев: $L_0 + L_2$
 - деревья настраиваются на пользовательскую \mathcal{L}
 - LightGBM
 - ускорение: оптимизация на меньшем #объектов и признаков
- Эффективная реализация с параллелизацией на ядрах процессора и видеокарте.

Заключение

- Бустинг - линейная композиция последовательно настраиваемых алгоритмов.
- В редких случаях есть аналитическое решение (AdaBoost).
- В общем случае используется градиентный бустинг.
- Градиентный бустинг над решающими деревьями - один из самых точных методов ML.
 - менее гибок, чем нейросети, но
 - проще, не нужно подбирать архитектуру сети
 - меньше предобработки данных (универсальность деревьев)
 - интерпретируемый (можно считать важности признаков)
- Полезные надстройки:
 - сжатие (shrinking) - больше базовых алгоритмов
 - обучение на подвыборках (subsampling) - повышение разнообразия
- Можно использовать квадратичную, а не линейную аппроксимацию.