Введение в искусственный интеллект. Машинное обучение

Тема: Ансамблирование моделей. Три метода на букву “Б”

Бабин Д.Н., Иванов И.Е.

кафедра Математической Теории Интеллектуальных Систем

План лекции

1. Стековое обобщение

План лекции

1. Стековое обобщение
   - Блендинг

План лекции

1. Стековое обобщение
   - Блендинг
   - Стекинг

План лекции

1. Стековое обобщение
   - Блендинг
   - Стекинг
2. Бутстрэп и пэстинг

План лекции

1. Стековое обобщение
   - Блендинг
   - Стекинг
2. Бутстрэп и пэстинг
3. Бэггинг

План лекции

1. Стековое обобщение
   - Блендинг
   - Стекинг
2. Бутстрэп и пэстинг
3. Бэггинг
4. Бустинг с дискретными базовыми алгоритмами

Ансамблирование

Ансамбль методов

Это способ использования нескольких обучающих алгоритмов с целью получения лучшей эффективности предсказания (классификации или регрессии), чем могли бы получить от каждого обучающего алгоритма по отдельности

Ансамблирование

Ансамбль методов

Это способ использования нескольких обучающих алгоритмов с целью получения лучшей эффективности предсказания (классификации или регрессии), чем могли бы получить от каждого обучающего алгоритма по отдельности

Замечание. Ансамбль методов не бесконечен: состоит из конкретного конечного множества альтернативных моделей.

Ансамблирование

Ансамбль методов

Это способ использования нескольких обучающих алгоритмов с целью получения лучшей эффективности предсказания (классификации или регрессии), чем могли бы получить от каждого обучающего алгоритма по отдельности

Замечание. Ансамбль методов не бесконечен: состоит из конкретного конечного множества альтернативных моделей.

Основные представители:

1. Стековое обобщение (stacked generalization)
2. Бэггинг (bagging)
3. Бустинг (boosting)

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования:
$a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования: $a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).
Если $y^*(x)$ – истинная функция ответа, то среднеквадратичная ошибка для базового алгоритма: $E(b_t(x) - y^*(x))^2 = E\varepsilon_t^2(x)$.

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования: $a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).
Если $y^*(x)$ – истинная функция ответа, то среднеквадратичная ошибка для базового алгоритма: $E(b_t(x) - y^*(x))^2 = E\varepsilon_t^2(x)$.
Средняя среднеквадратичная ошибка по всем базовым алгоритмам: $E_{avg} = \frac{1}{T} E \sum_{t=1}^{T} \varepsilon_t^2(x)$.

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования:
$a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).
Если $y^*(x)$ – истинная функция ответа, то среднеквадратичная ошибка для базового алгоритма: $E(b_t(x) - y^*(x))^2 = E\varepsilon_t^2(x)$.
Средняя среднеквадратичная ошибка по всем базовым алгоритмам:
$E_{avg} = \frac{1}{T} E \sum_{t=1}^{T} \varepsilon_t^2(x)$.
Предположим, что ошибки несмещены и некоррелированы: $E\varepsilon_t(x) = 0, E\varepsilon_t\varepsilon_u = 0, t \neq u$.

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования: $a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).
Если $y^*(x)$ – истинная функция ответа, то среднеквадратичная ошибка для базового алгоритма: $E(b_t(x) - y^*(x))^2 = E\varepsilon_t^2(x)$.
Средняя среднеквадратичная ошибка по всем базовым алгоритмам:
$E_{avg} = \frac{1}{T} E\sum_{t=1}^{T} \varepsilon_t^2(x)$.
Предположим, что ошибки несмещены и некоррелированы: $E\varepsilon_t(x) = 0, E\varepsilon_t\varepsilon_u = 0, t \neq u$.
Найдем среднеквадратичную ошибку для $a(x)$:
$E_{ens} = E(a(x) - y^*(x))^2 = E(\frac{1}{T}\sum_{t=1}^{T} b_t(x) - y^*(x))^2 = E(\frac{1}{T}\sum_{t=1}^{T} \varepsilon_t)^2 =$
$\frac{1}{T^2}E(\sum_{t=1}^{T} \varepsilon_t^2 + \sum_{t\neq u} \varepsilon_t\varepsilon_u) = \frac{1}{T^2}E\sum_{t=1}^{T} \varepsilon_t^2(x) = \frac{1}{T}E_{avg}$.

Зачем нужно ансамблирование

Рассмотрим часто применяемый на практике метод простого голосования:
$a(x) = \frac{1}{T}\sum_{t=1}^{T} b_t(x)$, где T – число базовых алгоритмов регрессии, $b_t(x)$ – сами базовые алгоритмы (для классификации с sign все рассматривается аналогично).
Если $y^*(x)$ – истинная функция ответа, то среднеквадратичная ошибка для базового алгоритма: $E(b_t(x) - y^*(x))^2 = E\varepsilon_t^2(x)$.
Средняя среднеквадратичная ошибка по всем базовым алгоритмам:
$E_{avg} = \frac{1}{T}E\sum_{t=1}^{T}\varepsilon_t^2(x)$.
Предположим, что ошибки несмещены и некоррелированы: $E\varepsilon_t(x) = 0, E\varepsilon_t\varepsilon_u = 0, t \neq u$.
Найдем среднеквадратичную ошибку для $a(x)$:
$E_{ens} = E(a(x) - y^*(x))^2 = E(\frac{1}{T}\sum_{t=1}^{T} b_t(x) - y^*(x))^2 = E(\frac{1}{T}\sum_{t=1}^{T}\varepsilon_t)^2 =$
$\frac{1}{T^2}E(\sum_{t=1}^{T}\varepsilon_t^2 + \sum_{t\neq u}\varepsilon_t\varepsilon_u) = \frac{1}{T^2}E\sum_{t=1}^{T}\varepsilon_t^2(x) = \frac{1}{T}E_{avg}$.
Таким образом, простое голосование позволило уменьшить средний квадрат ошибки в T раз!

Стековое обобщение[1]

Предположим, что мы можем обучить T базовых алгоритмов.
После этого мы обучаем комбинирующий алгоритм верхнего уровня (**мета-алгоритм**), *входом* для которого являются *выходы* базовых.

[1]Wolpert D. (1992) "Stacked Generalization"

Стековое обобщение[1]

Предположим, что мы можем обучить T базовых алгоритмов.
После этого мы обучаем комбинирующий алгоритм верхнего уровня (**мета-алгоритм**), *входом* для которого являются *выходы* базовых.

Схема стекового обобщения

1. Обучаем по отдельности каждый базовый алгоритм $b_t(x), t = 1, \dots, T$
2. Фиксируем алгоритмы $b_t(x)$
3. Обучаем комбинирующий алгоритм верхнего уровня $a(x) = a(b_1(x), \dots, b_T(x))$

[1] Wolpert D. (1992) "Stacked Generalization"

Стековое обобщение[1]

Предположим, что мы можем обучить T базовых алгоритмов.
После этого мы обучаем комбинирующий алгоритм верхнего уровня (**мета-алгоритм**), *входом* для которого являются *выходы* базовых.

Схема стекового обобщения

1. Обучаем по отдельности каждый базовый алгоритм $b_t(x), t = 1, \ldots, T$
2. Фиксируем алгоритмы $b_t(x)$
3. Обучаем комбинирующий алгоритм верхнего уровня $a(x) = a(b_1(x), \ldots, b_T(x))$

Замечание 1. Простое (или взвешенное) голосование является частным случаем стекового обобщения с необучаемым комбинирующим алгоритмом верхнего уровня.

[1] Wolpert D. (1992) “Stacked Generalization”

Стековое обобщение[1]

Предположим, что мы можем обучить T базовых алгоритмов.
После этого мы обучаем комбинирующий алгоритм верхнего уровня (**мета-алгоритм**), *входом* для которого являются *выходы* базовых.

Схема стекового обобщения

1. Обучаем по отдельности каждый базовый алгоритм $b_t(x), t = 1, \ldots, T$
2. Фиксируем алгоритмы $b_t(x)$
3. Обучаем комбинирующий алгоритм верхнего уровня $a(x) = a(b_1(x), \ldots, b_T(x))$

Замечание 1. Простое (или взвешенное) голосование является частным случаем стекового обобщения с необучаемым комбинирующим алгоритмом верхнего уровня.
Замечание 2. Стековое обобщение - один из главных методов достижения успехов на Kaggle :)

[1] Wolpert D. (1992) "Stacked Generalization"

Блендинг (Blending)

Ранее мы рассмотрели общую схему, теперь рассмотрим конкретные варианты реализации на практике.

Блендинг (Blending)

Ранее мы рассмотрели общую схему, теперь рассмотрим конкретные варианты реализации на практике.

1. Разбиваем обучающую выборку на две части

Блендинг (Blending)

Ранее мы рассмотрели общую схему, теперь рассмотрим конкретные варианты реализации на практике.

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы

Блендинг (Blending)

Ранее мы рассмотрели общую схему, теперь рассмотрим конкретные варианты реализации на практике.

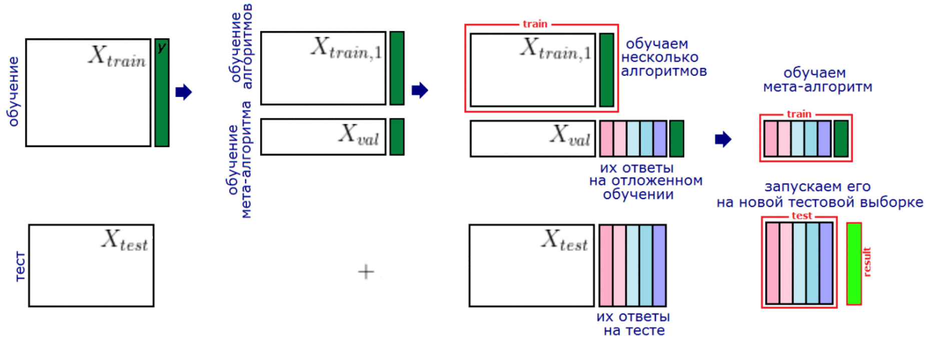
1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй получаем ответы базовых алгоритмов и обучаем мета-алгоритм

Блендинг (Blending)

Ранее мы рассмотрели общую схему, теперь рассмотрим конкретные варианты реализации на практике.

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй получаем ответы базовых алгоритмов и обучаем мета-алгоритм
4. На тесте сначала получаем выходы базовых алгоритмов, к которым затем применяем мета-алгоритм

Рассмотрим схему[2] блендинга:



[2]https://dyakonov.org

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход "вширь").

1. Разбиваем обучающую выборку на две части

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход "вширь").

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход “вширь”).

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй части получаем их ответы (**мета-признаки**)

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход “вширь”).

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй части получаем их ответы (**мета-признаки**)
4. Обучаем мета-алгоритм

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход “вширь”).

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй части получаем их ответы (**мета-признаки**)
4. Обучаем мета-алгоритм
5. Повторяем пп. 2-4 $M \geq 2$ раз для других разбиений обучающей выборки

Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход “вширь”).

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй части получаем их ответы (**мета-признаки**)
4. Обучаем мета-алгоритм
5. Повторяем пп. 2-4 $M \geq 2$ раз для других разбиений обучающей выборки
6. На тесте сначала получаем M наборов выходов базовых алгоритмов, обученных на разных разбиениях, затем для каждого набора запускаем соответствующий мета-алгоритм

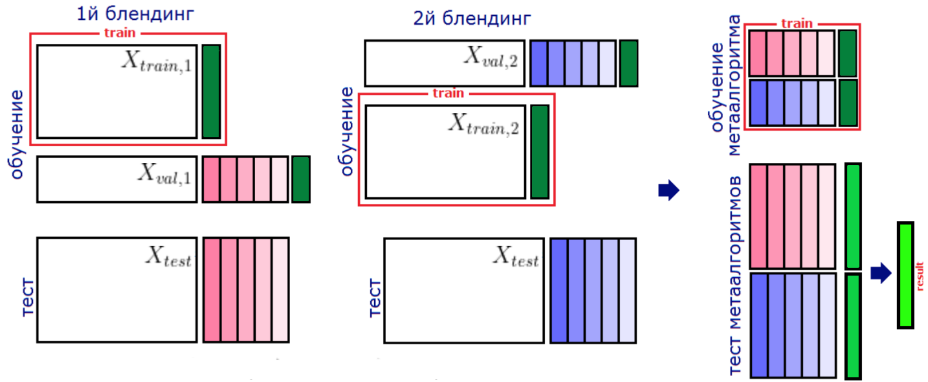
Блендинг – улучшение

Проблема классического блендинга: ни базовые алгоритмы, ни мета-алгоритм не видят всей обучающей выборки. Поэтому можно немного усовершенствовать подход (подход “вширь”).

1. Разбиваем обучающую выборку на две части
2. На одной части обучаем базовые алгоритмы
3. На второй части получаем их ответы (**мета-признаки**)
4. Обучаем мета-алгоритм
5. Повторяем пп. 2-4 $M \geq 2$ раз для других разбиений обучающей выборки
6. На тесте сначала получаем M наборов выходов базовых алгоритмов, обученных на разных разбиениях, затем для каждого набора запускаем соответствующий мета-алгоритм
7. После чего усредняем M ответов мета-алгоритмов

Блендинг – визуализация усовершенствования

Рассмотрим схему усовершенствованного [3] блендинга:



[3] https://dyakonov.org

Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки.

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). "Super learner"

Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки. Опишем алгоритм **Super Learner**[4]:

1. Разбиваем обучающую выборку на k частей

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). “Super learner”

Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки. Опишем алгоритм **Super Learner**[4]:

1. Разбиваем обучающую выборку на k частей
2. Обучаем базовые алгоритмы с помощью кросс-валидации

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). “Super learner”

Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки. Опишем алгоритм **Super Learner**[4]:

1. Разбиваем обучающую выборку на k частей
2. Обучаем базовые алгоритмы с помощью кросс-валидации
   - Обучаем каждый алгоритм на $k-1$ частях и тестируем на hold-out части,
   - Повторяем так для каждой из k частей,
   - Обучаем каждый базовый алгоритм на всей обучающей выборке,

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). "Super learner"

Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки. Опишем алгоритм **Super Learner**[4]:

1. Разбиваем обучающую выборку на k частей
2. Обучаем базовые алгоритмы с помощью кросс-валидации
   - Обучаем каждый алгоритм на $k-1$ частях и тестируем на hold-out части,
   - Повторяем так для каждой из k частей,
   - Обучаем каждый базовый алгоритм на всей обучающей выборке,
3. Обучаем мета-алгоритм на hold-out выходах (мета-признаках) базовых алгоритмов по всей обучающей выборке

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). "Super learner"

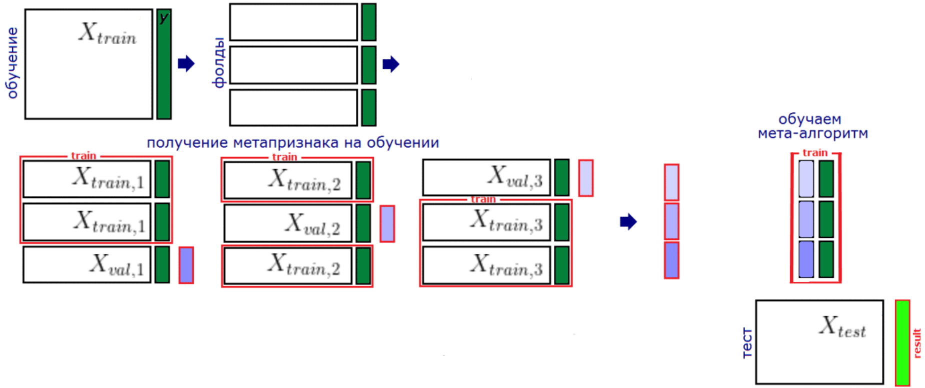
Стекинг (Stacking)

Попытка решения той же проблемы: алгоритмы не видят всей обучающей выборки. Опишем алгоритм **Super Learner**[4]:

1. Разбиваем обучающую выборку на k частей
2. Обучаем базовые алгоритмы с помощью кросс-валидации
   - Обучаем каждый алгоритм на $k-1$ частях и тестируем на hold-out части,
   - Повторяем так для каждой из k частей,
   - Обучаем каждый базовый алгоритм на всей обучающей выборке,
3. Обучаем мета-алгоритм на hold-out выходах (мета-признаках) базовых алгоритмов по всей обучающей выборке
4. На тесте сначала получаем выходы базовых алгоритмов, к которым применяем мета-алгоритм

[4] Van der Laan, M. J., Polley, E. C., and Hubbard, A. E. (2007). “Super learner”

Рассмотрим схему[5] стекинга:



[5] https://dyakonov.org

Замечания о стековых обобщениях

- Блендинг очень прост в реализации

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке
- Стекинг решает проблему обучения базовых алгоритмов на всем обучающем множестве, однако не всегда от этого есть ощутимый прирост

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке
- Стекинг решает проблему обучения базовых алгоритмов на всем обучающем множестве, однако не всегда от этого есть ощутимый прирост
- Стековое обобщение подходит для использования алгоритмов разной природы

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке
- Стекинг решает проблему обучения базовых алгоритмов на всем обучающем множестве, однако не всегда от этого есть ощутимый прирост
- Стековое обобщение подходит для использования алгоритмов разной природы
- В качестве мета-алгоритмов проще всего использовать регрессоры

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке
- Стекинг решает проблему обучения базовых алгоритмов на всем обучающем множестве, однако не всегда от этого есть ощутимый прирост
- Стековое обобщение подходит для использования алгоритмов разной природы
- В качестве мета-алгоритмов проще всего использовать регрессоры
- Выходы базовых алгоритмов часто коррелируют, поэтому лучше использовать недообученные версии этих алгоритмов

Замечания о стековых обобщениях

- Блендинг очень прост в реализации
- Усовершенствованный блендинг зачастую не дает прироста по сравнению с обычным усреднением ответов базовых алгоритмов, обученных на всей обучающей выборке
- Стекинг решает проблему обучения базовых алгоритмов на всем обучающем множестве, однако не всегда от этого есть ощутимый прирост
- Стековое обобщение подходит для использования алгоритмов разной природы
- В качестве мета-алгоритмов проще всего использовать регрессоры
- Выходы базовых алгоритмов часто коррелируют, поэтому лучше использовать недообученные версии этих алгоритмов
- Можно для обучения мета-алгоритма использовать не только выходы базовых алгоритмов, но и исходные данные; однако так лучше не делать

Время для вопросов

Бутстрэп (Bootstrap)

Бутстрэп (Bootstrap)

Английская поговорка: “*To pull oneself over a fence by one’s bootstraps*”.
Русский аналог: *Мюнхгаузен, вытаскивающий себя за волосы из болота*.

Бутстрэп

Определение

Бутстрэп - это методика тестирования на основе случайного семплирования (либо само это семплирование) из выборки **с возвращением**.

[6] Efron, B. (1979). "Bootstrap methods: Another look at the jackknife"

Бутстрэп

Определение

Бутстрэп - это методика тестирования на основе случайного семплирования (либо само это семплирование) из выборки **с возвращением**.

- Бутстрэп[6] позволяет оценивать параметры алгоритмов (такие как смещение, разброс, доверительный интервал и т.п.) на основе семплированных выборок

[6] Efron, B. (1979). "Bootstrap methods: Another look at the jackknife"

Бутстрэп

Определение

Бутстрэп - это методика тестирования на основе случайного семплирования (либо само это семплирование) из выборки **с возвращением**.

- Бутстрэп[6] позволяет оценивать параметры алгоритмов (такие как смещение, разброс, доверительный интервал и т.п.) на основе семплированных выборок
- Многократная генерация выборок происходит методом Монте-Карло на базе имеющейся выборки (т.о., из одной выборки генерируем любое число выборок)

[6] Efron, B. (1979). "Bootstrap methods: Another look at the jackknife"

О семплировании с возвращением

Теорема

При использовании бутстрэпа для генерации выборки той же мощности N, что и исходная выборка, доля объектов, не попавших в сгенерированную выборку, стремится к e^{-1} при $N \to \infty$.

О семплировании с возвращением

Теорема

При использовании бутстрэпа для генерации выборки той же мощности N, что и исходная выборка, доля объектов, не попавших в сгенерированную выборку, стремится к e^{-1} при $N \to \infty$.

Доказательство. На каждом шаге все объекты попадают в новую выборку с возвращением равновероятно, т.е отдельный объект – с вероятностью $\frac{1}{N}$.
Вероятность того, что объект не попадёт в новую выборку после N шагов: $(1 - \frac{1}{N})^N$.
Вспоминаем второй замечательный предел:
$\lim_{N\to\infty}(1 - \frac{1}{N})^N = \lim_{N\to\infty}\left((1 - \frac{1}{N})^{-N}\right)^{-1} = e^{-1}$. **Ч.т.д.**

О семплировании с возвращением

Теорема

При использовании бутстрэпа для генерации выборки той же мощности N, что и исходная выборка, доля объектов, не попавших в сгенерированную выборку, стремится к e^{-1} при $N \to \infty$.

Доказательство. На каждом шаге все объекты попадают в новую выборку с возвращением равновероятно, т.е отдельный объект – с вероятностью $\frac{1}{N}$.
Вероятность того, что объект не попадёт в новую выборку после N шагов: $(1 - \frac{1}{N})^N$.
Вспоминаем второй замечательный предел:
$\lim_{N\to\infty}(1 - \frac{1}{N})^N = \lim_{N\to\infty}\left((1 - \frac{1}{N})^{-N}\right)^{-1} = e^{-1}$. **Ч.т.д.**
Замечание. Т.о. можно тестировать алгоритм на оставшихся $e^{-1} \approx 37\%$ данных.

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

[7] Breiman, L. (1999). “Pasting small votes for classification in large databases and on-line”

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

- Дано: обучающая выборка, состоящая из $N = 5$ объектов $X = \{1, 2, 3, 4, 5\}$,

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

- Дано: обучающая выборка, состоящая из $N = 5$ объектов $X = \{1, 2, 3, 4, 5\}$,
- Проводим процедуру семплирования $n = 3$ объектов,

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

- Дано: обучающая выборка, состоящая из $N = 5$ объектов $X = \{1, 2, 3, 4, 5\}$,
- Проводим процедуру семплирования $n = 3$ объектов,
- Бутстрэп: $\{2, 2, 5\}, \{1, 2, 5\}, \{2, 4, 4\}, \ldots$

[7] Breiman, L. (1999). “Pasting small votes for classification in large databases and on-line”

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

- Дано: обучающая выборка, состоящая из $N = 5$ объектов $X = \{1, 2, 3, 4, 5\}$,
- Проводим процедуру семплирования $n = 3$ объектов,
- Бутстрэп: $\{2, 2, 5\}, \{1, 2, 5\}, \{2, 4, 4\}, \ldots$
- Пэстинг: $\{2, 3, 5\}, \{1, 3, 5\}, \{2, 3, 4\}, \ldots$

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Пэстинг

- **Вопрос**. Что будет, если мы **запретим** при процедуре бутстрэп-семплирования **возвращать** объекты назад в выборку для их возможного выбора еще раз?
- **Ответ**. Получим т.н. процедуру **пэстинга**[7] (Pasting).

Приведем пример для сравнения различных процедур семплирования.

- Дано: обучающая выборка, состоящая из $N = 5$ объектов $X = \{1, 2, 3, 4, 5\}$,
- Проводим процедуру семплирования $n = 3$ объектов,
- Бутстрэп: $\{2, 2, 5\}, \{1, 2, 5\}, \{2, 4, 4\}, \ldots$
- Пэстинг: $\{2, 3, 5\}, \{1, 3, 5\}, \{2, 3, 4\}, \ldots$

Замечание. Пэстинг имеет очевидную реализацию: сначала случайно перемешиваем объекты, затем берем из них n первых.

[7] Breiman, L. (1999). "Pasting small votes for classification in large databases and on-line"

Сравнение пэстинга и бутстрэпа

- При бутстрэпе мы можем получить практически неограниченное количество подвыборок (за счет разрешения семплирования объектов с повторением),

Сравнение пэстинга и бутстрэпа

- При бутстрэпе мы можем получить практически неограниченное количество подвыборок (за счет разрешения семплирования объектов с повторением),
- При пэстинге мы можем получить гораздо меньше подвыборок (поскольку все элементы должны быть различны),

Сравнение пэстинга и бутстрэпа

- При бутстрэпе мы можем получить практически неограниченное количество подвыборок (за счет разрешения семплирования объектов с повторением),
- При пэстинге мы можем получить гораздо меньше подвыборок (поскольку все элементы должны быть различны),
- Пэстинг при размере выборки, совпадающей по порядку с размером исходной обучающей выборки ($n \sim N$), практически не имеет никакого смысла,

Сравнение пэстинга и бутстрэпа

- При бутстрэпе мы можем получить практически неограниченное количество подвыборок (за счет разрешения семплирования объектов с повторением),
- При пэстинге мы можем получить гораздо меньше подвыборок (поскольку все элементы должны быть различны),
- Пэстинг при размере выборки, совпадающей по порядку с размером исходной обучающей выборки ($n \sim N$), практически не имеет никакого смысла,
- Пэстинг имеет смысл применять, когда нам важно, чтобы объекты не повторялись.

Время для вопросов

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$.

[8] Breiman L. (1994). “Bagging Predictors”.

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$. Также мы видели, что простое усреднение T алгоритмов позволяет теоретически уменьшить разброс в T раз, при этом не влияя на смещение.

[8] Breiman L. (1994). “Bagging Predictors”.

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$.
Также мы видели, что простое усреднение T алгоритмов позволяет теоретически уменьшить разброс в T раз, при этом не влияя на смещение.
Это и есть главная идея бэггинга[8]:

[8] Breiman L. (1994). "Bagging Predictors".

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$.
Также мы видели, что простое усреднение T алгоритмов позволяет теоретически уменьшить разброс в T раз, при этом не влияя на смещение.
Это и есть главная идея бэггинга[8]:

- уменьшить разброс алгоритма,

[8] Breiman L. (1994). “Bagging Predictors”.

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$.
Также мы видели, что простое усреднение T алгоритмов позволяет теоретически уменьшить разброс в T раз, при этом не влияя на смещение.
Это и есть главная идея бэггинга[8]:

- уменьшить разброс алгоритма,
- как следствие, бороться с переобучением.

[8] Breiman L. (1994). “Bagging Predictors”.

Бэггинг

Вспомним разложение ошибки на разброс и смещение: $\sigma^2 + variance(a) + bias^2(f, a)$.
Также мы видели, что простое усреднение T алгоритмов позволяет теоретически уменьшить разброс в T раз, при этом не влияя на смещение.
Это и есть главная идея бэггинга[8]:

- уменьшить разброс алгоритма,
- как следствие, бороться с переобучением.

Определение

Бэггинг (**B**ootstrap **AGG**regat**ING**) - это метод ансамблирования, основанный на:

1. бутстрэп-семплировании для каждого обучения базового алгоритма,
2. последующем усреднении ответов уже обученных базовых алгоритмов методом простого голосования.

[8] Breiman L. (1994). “Bagging Predictors”.

Алгоритм бэггинга

- **Дано**: обучающая выборка X^m мощности m.
- **Цель**: обучить ансамбль из T классификаторов $b_t(x), t = 1, \ldots, T$.

Алгоритм бэггинга

- **Дано**: обучающая выборка X^m мощности m.
- **Цель**: обучить ансамбль из T классификаторов $b_t(x), t = 1, \ldots, T$.

Алгоритм

1. Формируем T выборок $X_t^m, t = 1, \ldots, T$ мощности m с помощью бутстрэп-семплирования,

Алгоритм бэггинга

- **Дано**: обучающая выборка X^m мощности m.
- **Цель**: обучить ансамбль из T классификаторов $b_t(x), t = 1, \dots, T$.

Алгоритм

1. Формируем T выборок $X_t^m, t = 1, \dots, T$ мощности m с помощью бутстрэп-семплирования,
2. На каждой выборке $X_t^m, t = 1, \dots, T$ обучаем свой алгоритм $b_t(x)$,

Алгоритм бэггинга

- **Дано**: обучающая выборка X^m мощности m.
- **Цель**: обучить ансамбль из T классификаторов $b_t(x), t = 1, \ldots, T$.

Алгоритм

1. Формируем T выборок $X_t^m, t = 1, \ldots, T$ мощности m с помощью бутстрэп-семплирования,
2. На каждой выборке $X_t^m, t = 1, \ldots, T$ обучаем свой алгоритм $b_t(x)$,
3. Результат применения – усреднение (для регрессии) или голосование (для классификации).

Пример бэггинга

Оказывается, можно использовать бутстрэп-семплирование не только для обучающей выборки, но и для признаков!

[9] Ho T. K. (1998). "The Random Subspace Method for Constructing Decision Forests"

Пример бэггинга

Оказывается, можно использовать бутстрэп-семплирование не только для обучающей выборки, но и для признаков!
Это – **метод случайных подпространств**[9].

[9] Ho T. K. (1998). “The Random Subspace Method for Constructing Decision Forests”

Пример бэггинга

Оказывается, можно использовать бутстрэп-семплирование не только для обучающей выборки, но и для признаков!

Это – **метод случайных подпространств**[9].

Т.о., случайные деревья из прошлой лекции – это объединение:

- Бэггинга для работы с выборкой и алгоритмами,

[9] Ho T. K. (1998). “The Random Subspace Method for Constructing Decision Forests”

Пример бэггинга

Оказывается, можно использовать бутстрэп-семплирование не только для обучающей выборки, но и для признаков!

Это – **метод случайных подпространств**[9].

Т.о., случайные деревья из прошлой лекции – это объединение:

- Бэггинга для работы с выборкой и алгоритмами,
- Метода случайных подпространств для работы с признаковым пространством.

[9] Ho T. K. (1998). “The Random Subspace Method for Constructing Decision Forests”

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,
- Каждый базовый алгоритм видит всего 63% обучающих данных,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,
- Каждый базовый алгоритм видит всего 63% обучающих данных,
- Не очень хорошо работает для стабильных алгоритмов (метод K-ближайших соседей),

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,
- Каждый базовый алгоритм видит всего 63% обучающих данных,
- Не очень хорошо работает для стабильных алгоритмов (метод K-ближайших соседей),
- Нужно держать в памяти T базовых алгоритмов,

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,
- Каждый базовый алгоритм видит всего 63% обучающих данных,
- Не очень хорошо работает для стабильных алгоритмов (метод K-ближайших соседей),
- Нужно держать в памяти T базовых алгоритмов,
- Нужно делать T запусков.

Плюсы и минусы бэггинга

Плюсы бэггинга

- Уменьшает разброс и, как следствие, борется с переобучением,
- Ошибки базовых алгоритмов взаимно компенсируются,
- Объекты-выбросы могут не попасть в некоторые обучающие подвыборки,
- Хорошо работает для нестабильных алгоритмов (нейронные сети),
- Легко распараллеливается.

Минусы бэггинга

- Не борется со смещением,
- Каждый базовый алгоритм видит всего 63% обучающих данных,
- Не очень хорошо работает для стабильных алгоритмов (метод K-ближайших соседей),
- Нужно держать в памяти T базовых алгоритмов,
- Нужно делать T запусков.

Замечание. Последние два минуса характерны в целом для ансамблирования.

Время для вопросов

Бустинг

Попробуем теперь бороться не только с разбросом, но и со смещением.

[10] Schapire R. E. (1990). “The Strength of Weak Learnability”.

Бустинг

Попробуем теперь бороться не только с разбросом, но и со смещением.
Главная идея бустинга[10]:

[10] Schapire R. E. (1990). “The Strength of Weak Learnability”.

Бустинг

Попробуем теперь бороться не только с разбросом, но и со смещением.
Главная идея бустинга[10]:

- Отвечает на вопрос: “Может ли набор слабых обучающих алгоритмов создать сильный обучающий алгоритм?”,

[10] Schapire R. E. (1990). “The Strength of Weak Learnability”.

Бустинг

Попробуем теперь бороться не только с разбросом, но и со смещением.
Главная идея бустинга[10]:

- Отвечает на вопрос: “Может ли набор слабых обучающих алгоритмов создать сильный обучающий алгоритм?”,
- Борется не только с разбросом, но и со смещением алгоритма.

[10] Schapire R. E. (1990). “The Strength of Weak Learnability”.

Бустинг

Попробуем теперь бороться не только с разбросом, но и со смещением.
Главная идея бустинга[10]:

- Отвечает на вопрос: "Может ли набор слабых обучающих алгоритмов создать сильный обучающий алгоритм?",
- Борется не только с разбросом, но и со смещением алгоритма.

Определение

Бустинг (Boosting) - это метод ансамблирования, основанный на:

1. взвешенном голосовании композиции,
2. последовательном выборе нового классификатора на основе ошибок предыдущих.

[10] Schapire R. E. (1990). "The Strength of Weak Learnability".

Типы бустинга

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,

Типы бустинга

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,
- Множества выходных значений базовых классификаторов.

Типы бустинга

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,
- Множества выходных значений базовых классификаторов.

Обозначим взвешенную сумму выходов базовых классификаторов $b_t(x)$ как $a(x) = \sum_{t=1}^{T} \alpha_t b_t, \alpha_t \in \mathbb{R}$.

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,
- Множества выходных значений базовых классификаторов.

Обозначим взвешенную сумму выходов базовых классификаторов $b_t(x)$ как $a(x) = \sum_{t=1}^{T} \alpha_t b_t, \alpha_t \in \mathbb{R}$.

AdaBoost

- Базовые алгоритмы $b_t(x)$ принимают значения из дискретного множества (например, $\{-1, +1\}$),
- Функция потерь: $e^{-y_i a(x_i)}$

Типы бустинга

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,
- Множества выходных значений базовых классификаторов.

Обозначим взвешенную сумму выходов базовых классификаторов $b_t(x)$ как $a(x) = \sum_{t=1}^{T} \alpha_t b_t, \alpha_t \in \mathbb{R}$.

AdaBoost

- Базовые алгоритмы $b_t(x)$ принимают значения из дискретного множества (например, $\{-1, +1\}$),
- Функция потерь: $e^{-y_i a(x_i)}$

AnyBoost

- Базовые алгоритмы $b_t(x)$ принимают значения из $\mathbb{R}$,
- Функция потерь $L : \mathbb{R} \to \mathbb{R}$ – гладкая функция от $y_i a(x_i)$

Типы бустинга

Разные виды бустинга можно описать в зависимости от:

- Функции потерь,
- Множества выходных значений базовых классификаторов.

Обозначим взвешенную сумму выходов базовых классификаторов $b_t(x)$ как $a(x) = \sum_{t=1}^{T} \alpha_t b_t, \alpha_t \in \mathbb{R}$.

AdaBoost

- Базовые алгоритмы $b_t(x)$ принимают значения из дискретного множества (например, $\{-1, +1\}$),
- Функция потерь: $e^{-y_i a(x_i)}$

AnyBoost

- Базовые алгоритмы $b_t(x)$ принимают значения из $\mathbb{R}$,
- Функция потерь $L : \mathbb{R} \to \mathbb{R}$ – гладкая функция от $y_i a(x_i)$

Gradient Boosting

- Базовые алгоритмы $b_t(x)$ принимают значения из $\mathbb{R}$,
- Функция потерь $L : \mathbb{R}^2 \to \mathbb{R}$ – гладкая функция от пары $(y_i, a(x_i))$

Бустинг для бинарной классификации

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

Бустинг для бинарной классификации

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

- Алгоритм классификации – взвешенное голосование: $a(x) = \text{sign}(\sum_{t=1}^T \alpha_t b_t(x))$,

Бустинг для бинарной классификации

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

- Алгоритм классификации – взвешенное голосование: $a(x) = \text{sign}(\sum_{t=1}^T \alpha_t b_t(x))$,
- Эмпирический риск – число ошибок на X^m: $R_T = \sum_{i=1}^m [y_i \sum_{t=1}^T \alpha_t b_t(x_i) < 0]$

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

- Алгоритм классификации – взвешенное голосование: $a(x) = \text{sign}(\sum_{t=1}^T \alpha_t b_t(x))$,
- Эмпирический риск – число ошибок на X^m: $R_T = \sum_{i=1}^m [y_i \sum_{t=1}^T \alpha_t b_t(x_i) < 0]$

Основные идеи обучения:

- Заморозка $\alpha_1 b_1(x_i), \ldots, \alpha_{t-1} b_{t-1}(x_i)$ при добавлении $\alpha_t b_t(x_i)$,

Бустинг для бинарной классификации

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

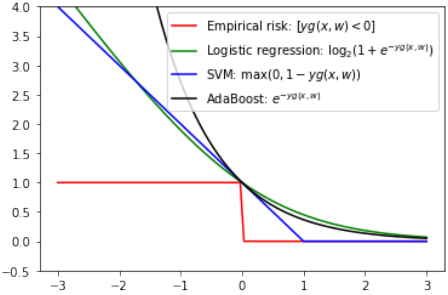
- Алгоритм классификации – взвешенное голосование: $a(x) = \operatorname{sign}(\sum_{t=1}^{T} \alpha_t b_t(x))$,
- Эмпирический риск – число ошибок на X^m: $R_T = \sum_{i=1}^{m} [y_i \sum_{t=1}^{T} \alpha_t b_t(x_i) < 0]$

Основные идеи обучения:

- Заморозка $\alpha_1 b_1(x_i), \ldots, \alpha_{t-1} b_{t-1}(x_i)$ при добавлении $\alpha_t b_t(x_i)$,
- Использовать аппроксимированный Э.Р.

Пусть $X^m = \{(x_i, y_i)_{i=1}^m\}, y_i \in Y = \{+1, -1\}, b_t : X \to \{-1, 0, +1\}$. Значение $b_t(x) = 0$ вводится для сигнализации неопределенности в классификации (аналогия: нахождение внутри полосы для SVM).

- Алгоритм классификации – взвешенное голосование: $a(x) = \text{sign}(\sum_{t=1}^T \alpha_t b_t(x))$,
- Эмпирический риск – число ошибок на X^m: $R_T = \sum_{i=1}^m [y_i \sum_{t=1}^T \alpha_t b_t(x_i) < 0]$



Основные идеи обучения:

- Заморозка $\alpha_1 b_1(x_i), \dots, \alpha_{t-1} b_{t-1}(x_i)$ при добавлении $\alpha_t b_t(x_i)$,
- Использовать аппроксимированный Э.Р.

Обозначения

Аппроксимация Э.Р. с помощью функции потерь $e^{-y_i a(x_i)}$:

$$R_T \leq \widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)}$$

Обозначения

Аппроксимация Э.Р. с помощью функции потерь $e^{-y_i a(x_i)}$:

$$R_T \leq \widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)}$$

- Вектор весов (взвешиваем объекты) $W^m = (w_1, \ldots, w_m)$: $w_i = e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} \Rightarrow \widetilde{R}_{T-1} = \sum_{i=1}^{m} w_i$,
- Нормировка: $\widetilde{w}_i = \frac{w_i}{\sum_{j=1}^{m} w_j} \Rightarrow \sum_{i=1}^{m} \widetilde{w}_i = 1, \widetilde{w}_i \geq 0$

Обозначения

Аппроксимация Э.Р. с помощью функции потерь $e^{-y_i a(x_i)}$:

$$R_T \le \widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)}$$

- Вектор весов (взвешиваем объекты) $W^m = (w_1, \ldots, w_m)$: $w_i = e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} \Rightarrow \widetilde{R}_{T-1} = \sum_{i=1}^{m} w_i$,
- Нормировка: $\widetilde{w}_i = \frac{w_i}{\sum_{j=1}^{m} w_j} \Rightarrow \sum_{i=1}^{m} \widetilde{w}_i = 1, \widetilde{w}_i \ge 0$
- Вероятностный вектор $U^m = (u_1, \ldots, u_m)$: $\sum_{i=1}^{m} u_i = 1, u_i \ge 0$,
- Взвешенное число правильных классификаций алгоритма $b(x)$ по вектору U^m: $P(b; U^m) = \sum_{i=1}^{m} u_i [b(x) = y_i]$
- Взвешенное число ошибочных классификаций алгоритма $b(x)$ по вектору U^m: $N(b; U^m) = \sum_{i=1}^{m} u_i [b(x) = -y_i]$
- Взвешенное число отказов от классификации: $1 - P - N$.

Основная теорема бустинга

Пусть A – достаточно богатое семейство базовых алгоритмов.

Теорема

Если для любого вероятностного вектора U^m существует алгоритм $b \in A$, т.ч. $P(b; U^m) > N(b; U^m)$, то минимум аппроксимированного Э.Р. $\widetilde{R}_T$ достигается на:

- $b_T = \arg\max_{b \in A} \sqrt{P(b; \widetilde{W}^m)} - \sqrt{N(b; \widetilde{W}^m)}$
- $\alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$

Основная теорема бустинга

Пусть A – достаточно богатое семейство базовых алгоритмов.

Теорема

Если для любого вероятностного вектора U^m существует алгоритм $b \in A$, т.ч. $P(b; U^m) > N(b; U^m)$, то минимум аппроксимированного Э.Р. $\widetilde{R}_T$ достигается на:

- $b_T = \arg\max_{b \in A} \sqrt{P(b; \widetilde{W}^m)} - \sqrt{N(b; \widetilde{W}^m)}$
- $\alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$

Замечание. В этом случае $\alpha_T > 0$.

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.
$\widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$
$\sum_{i=1}^{m} w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.
$\widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$
$\sum_{i=1}^{m} w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$
$e^{-\alpha_T} \sum_{i=1}^{m} w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^{m} w_i[b_T(x_i) = -y_i] + \sum_{i=1}^{m} w_i[b_T(x_i) = 0] =$

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.

$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$

$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$

$e^{-\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = -y_i] + \sum_{i=1}^m w_i[b_T(x_i) = 0] =$

$(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = 0]) \sum_{i=1}^m w_i =$

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.
$\widetilde{R}_T = \sum_{i=1}^{m} e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$
$\sum_{i=1}^{m} w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$
$e^{-\alpha_T} \sum_{i=1}^{m} w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^{m} w_i[b_T(x_i) = -y_i] + \sum_{i=1}^{m} w_i[b_T(x_i) = 0] =$
$(e^{-\alpha_T} \sum_{i=1}^{m} \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^{m} \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^{m} \widetilde{w}_i[b_T(x_i) = 0]) \sum_{i=1}^{m} w_i =$
$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N)\widetilde{R}_{T-1}$.

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.
$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$
$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$
$e^{-\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = -y_i] + \sum_{i=1}^m w_i[b_T(x_i) = 0] =$
$(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = 0]) \sum_{i=1}^m w_i =$
$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N)\widetilde{R}_{T-1}$.
$\widetilde{R}_T \to \min_{\alpha_T} \Rightarrow \frac{\partial \widetilde{R}_T}{\partial \alpha_T} = 0$.

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b=1] + e^{\alpha}[b=-1] + [b=0]$.

$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$

$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$

$e^{-\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = -y_i] + \sum_{i=1}^m w_i[b_T(x_i) = 0] =$

$(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = 0]) \sum_{i=1}^m w_i =$

$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N)\widetilde{R}_{T-1}$.

$\widetilde{R}_T \to \min_{\alpha_T} \Rightarrow \frac{\partial \widetilde{R}_T}{\partial \alpha_T} = 0$.

$\frac{\partial \widetilde{R}_T}{\partial \alpha_T} = (-e^{-\alpha_T} P + e^{\alpha_T} N)\widetilde{R}_{T-1} = 0 \Rightarrow e^{-\alpha_T} P = e^{\alpha_T} N \Rightarrow e^{2\alpha_T} = \frac{P}{N} \Rightarrow \alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$.

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.
$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$
$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$
$e^{-\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = -y_i] + \sum_{i=1}^m w_i[b_T(x_i) = 0] =$
$\left(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = 0]\right) \sum_{i=1}^m w_i =$
$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N)\widetilde{R}_{T-1}$.
$\widetilde{R}_T \to \min_{\alpha_T} \Rightarrow \frac{\partial \widetilde{R}_T}{\partial \alpha_T} = 0$.
$\frac{\partial \widetilde{R}_T}{\partial \alpha_T} = (-e^{-\alpha_T} P + e^{\alpha_T} N)\widetilde{R}_{T-1} = 0 \Rightarrow e^{-\alpha_T} P = e^{\alpha_T} N \Rightarrow e^{2\alpha_T} = \frac{P}{N} \Rightarrow \alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$.
Для поиска $b_T(x)$ подставим найденное α_T в формулу для $\widetilde{R}_T$:

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b=1] + e^{\alpha}[b=-1] + [b=0]$.

$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$

$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$

$e^{-\alpha_T} \sum_{i=1}^m w_i [b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i [b_T(x_i) = -y_i] + \sum_{i=1}^m w_i [b_T(x_i) = 0] =$

$\left(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i [b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i [b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i [b_T(x_i) = 0]\right) \sum_{i=1}^m w_i =$

$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N) \widetilde{R}_{T-1}$.

$\widetilde{R}_T \to \min_{\alpha_T} \Rightarrow \frac{\partial \widetilde{R}_T}{\partial \alpha_T} = 0$.

$\frac{\partial \widetilde{R}_T}{\partial \alpha_T} = (-e^{-\alpha_T} P + e^{\alpha_T} N) \widetilde{R}_{T-1} = 0 \Rightarrow e^{-\alpha_T} P = e^{\alpha_T} N \Rightarrow e^{2\alpha_T} = \frac{P}{N} \Rightarrow \alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$.

Для поиска $b_T(x)$ подставим найденное α_T в формулу для $\widetilde{R}_T$:

$\widetilde{R}_T = \left(\sqrt{\frac{N}{P}} P + \sqrt{\frac{P}{N}} N + 1 - P - N\right) \widetilde{R}_{T-1} = (1 - (P - 2\sqrt{PN} + N)) \widetilde{R}_{T-1} =$

$(1 - (\sqrt{P} - \sqrt{N})^2) \widetilde{R}_{T-1} \to \min_{b_T} \Rightarrow$

Доказательство

Если $b \in \{-1, 0, +1\}$, то верно тождество $e^{-\alpha b} = e^{-\alpha}[b = 1] + e^{\alpha}[b = -1] + [b = 0]$.

$\widetilde{R}_T = \sum_{i=1}^m e^{-y_i \sum_{t=1}^{T-1} \alpha_t b_t(x_i)} e^{-y_i \alpha_T b_T(x_i)} =$

$\sum_{i=1}^m w_i \left(e^{-\alpha_T}[b_T(x_i) = y_i] + e^{\alpha_T}[b_T(x_i) = -y_i] + [b_T(x_i) = 0]\right) =$

$e^{-\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m w_i[b_T(x_i) = -y_i] + \sum_{i=1}^m w_i[b_T(x_i) = 0] =$

$(e^{-\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = y_i] + e^{\alpha_T} \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = -y_i] + \sum_{i=1}^m \widetilde{w}_i[b_T(x_i) = 0]) \sum_{i=1}^m w_i =$

$(e^{-\alpha_T} P + e^{\alpha_T} N + 1 - P - N)\widetilde{R}_{T-1}$.

$\widetilde{R}_T \to \min_{\alpha_T} \Rightarrow \frac{\partial \widetilde{R}_T}{\partial \alpha_T} = 0$.

$\frac{\partial \widetilde{R}_T}{\partial \alpha_T} = (-e^{-\alpha_T} P + e^{\alpha_T} N)\widetilde{R}_{T-1} = 0 \Rightarrow e^{-\alpha_T} P = e^{\alpha_T} N \Rightarrow e^{2\alpha_T} = \frac{P}{N} \Rightarrow \alpha_T = \frac{1}{2} \ln \frac{P(b_T; \widetilde{W}^m)}{N(b_T; \widetilde{W}^m)}$.

Для поиска $b_T(x)$ подставим найденное α_T в формулу для $\widetilde{R}_T$:

$\widetilde{R}_T = (\sqrt{\frac{N}{P}} P + \sqrt{\frac{P}{N}} N + 1 - P - N)\widetilde{R}_{T-1} = (1 - (P - 2\sqrt{PN} + N))\widetilde{R}_{T-1} =$

$(1 - (\sqrt{P} - \sqrt{N})^2)\widetilde{R}_{T-1} \to \min_{b_T} \Rightarrow$

$b_T = \arg\max_{b \in A} \sqrt{P(b; \widetilde{W}^m)} - \sqrt{N(b; \widetilde{W}^m)}$ (т.к. $N < P \leq 1$ по условию Теоремы). **Ч.т.д.**

Следствие 1: Сходимость

Теорема

Если на каждом шаге t можно добиться выполнения $\sqrt{P(b_t; \widetilde{W}^m)} - \sqrt{N(b_t; \widetilde{W}^m)} = \beta_t \geq \beta$ при некотором $0 < \beta \leq 1$, то за конечное число шагов будет построен алгоритм, не допускающий ни единой ошибки на обучающем множестве.

Следствие 1: Сходимость

Теорема

Если на каждом шаге t можно добиться выполнения $\sqrt{P(b_t; \widetilde{W}^m)} - \sqrt{N(b_t; \widetilde{W}^m)} = \beta_t \geq \beta$ при некотором $0 < \beta \leq 1$, то за конечное число шагов будет построен алгоритм, не допускающий ни единой ошибки на обучающем множестве.

Доказательство.

$R_T \leq \widetilde{R}_T = (1 - (\sqrt{P} - \sqrt{N})^2)\widetilde{R}_{T-1} \leq (1 - \beta^2)\widetilde{R}_{T-1} \leq \dots \leq (1 - \beta^2)^{T-1}\widetilde{R}_1.$

Следствие 1: Сходимость

Теорема

Если на каждом шаге t можно добиться выполнения $\sqrt{P(b_t; \widetilde{W}^m)} - \sqrt{N(b_t; \widetilde{W}^m)} = \beta_t \geq \beta$ при некотором $0 < \beta \leq 1$, то за конечное число шагов будет построен алгоритм, не допускающий ни единой ошибки на обучающем множестве.

Доказательство.
$R_T \leq \widetilde{R}_T = (1 - (\sqrt{P} - \sqrt{N})^2)\widetilde{R}_{T-1} \leq (1 - \beta^2)\widetilde{R}_{T-1} \leq \cdots \leq (1 - \beta^2)^{T-1}\widetilde{R}_1$.
Для любого $0 < \beta \leq 1$ и любого $\widetilde{R}_1$ будет существовать такое T, что $R_T < 1$.

Следствие 1: Сходимость

Теорема

Если на каждом шаге t можно добиться выполнения $\sqrt{P(b_t; \widetilde{W}^m)} - \sqrt{N(b_t; \widetilde{W}^m)} = \beta_t \geq \beta$ при некотором $0 < \beta \leq 1$, то за конечное число шагов будет построен алгоритм, не допускающий ни единой ошибки на обучающем множестве.

Доказательство.
$R_T \leq \widetilde{R}_T = (1 - (\sqrt{P} - \sqrt{N})^2)\widetilde{R}_{T-1} \leq (1 - \beta^2)\widetilde{R}_{T-1} \leq \dots \leq (1 - \beta^2)^{T-1}\widetilde{R}_1$.
Для любого $0 < \beta \leq 1$ и любого $\widetilde{R}_1$ будет существовать такое T, что $R_T < 1$.
Э.Р. R_T – это число ошибок на обучающем множестве (т.е. неотрицательное целое число)
$\Rightarrow R_T = 0$. **Ч.т.д.**

Следствие 2: Классический AdaBoost

Рассмотрим более частную ситуацию, когда базовый алгоритм не сигнализирует о неопределенности: $b_t : X \to \{-1, +1\}$. Тогда $P + N = 1$.

[11] Freund Y. and Schapire R.E (1997). "A decision-theoretic generalization of on-line learning and an application to boosting"

Рассмотрим более частную ситуацию, когда базовый алгоритм не сигнализирует о неопределенности: $b_t : X \to \{-1, +1\}$. Тогда $P + N = 1$.
В этом случае конкретный алгоритм бустинга называется AdaBoost[11] (**Ada**ptive **Boost**ing).

[11] Freund Y. and Schapire R.E (1997). "A decision-theoretic generalization of on-line learning and an application to boosting"

Рассмотрим более частную ситуацию, когда базовый алгоритм не сигнализирует о неопределенности: $b_t : X \to \{-1, +1\}$. Тогда $P + N = 1$.
В этом случае конкретный алгоритм бустинга называется AdaBoost[11] (**Ada**ptive **Boost**ing).

Теорема

Если для любого вероятностного вектора U^m существует алгоритм $b \in A$, т.ч. $N(b; U^m) < \frac{1}{2}$, то минимум аппроксимированного Э.Р. $\widetilde{R}_T$ достигается на:

- $b_T = \arg\min_{b \in A} N(b; \widetilde{W}^m)$
- $\alpha_T = \frac{1}{2} \ln \frac{1 - N(b; \widetilde{W}^m)}{N(b; \widetilde{W}^m)}$

[11] Freund Y. and Schapire R.E (1997). “A decision-theoretic generalization of on-line learning and an application to boosting”

Алгоритм AdaBoost

Алгоритм

- Инициализация весов: $w_i = \frac{1}{m}, i = 1, \ldots, m,$

Алгоритм AdaBoost

Алгоритм

- Инициализация весов: $w_i = \frac{1}{m}, i = 1, \ldots, m,$

Для $t = 1, \ldots, T$

- Обучение базового алгоритма $b_t = \arg\min_{b \in A} N(b; \widetilde{W}^m),$

Алгоритм AdaBoost

Алгоритм

- Инициализация весов: $w_i = \frac{1}{m}, i = 1, \ldots, m,$

Для $t = 1, \ldots, T$

- Обучение базового алгоритма $b_t = \arg\min_{b \in A} N(b; \widetilde{W}^m),$
- Вычисление нового веса $\alpha_t = \frac{1}{2} \ln \frac{1 - N(b_t; \widetilde{W}^m)}{N(b_t; \widetilde{W}^m)},$

Алгоритм AdaBoost

Алгоритм

- Инициализация весов: $w_i = \frac{1}{m}, i = 1, \dots, m,$

Для $t = 1, \dots, T$

- Обучение базового алгоритма $b_t = \arg\min_{b \in A} N(b; \widetilde{W}^m),$
- Вычисление нового веса $\alpha_t = \frac{1}{2} \ln \frac{1 - N(b_t; \widetilde{W}^m)}{N(b_t; \widetilde{W}^m)},$
- Обновление весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}, i = 1, \dots, m,$

Алгоритм AdaBoost

Алгоритм

- Инициализация весов: $w_i = \frac{1}{m}, i = 1, \ldots, m,$

Для $t = 1, \ldots, T$

- Обучение базового алгоритма $b_t = \arg\min_{b \in A} N(b; \widetilde{W}^m),$
- Вычисление нового веса $\alpha_t = \frac{1}{2} \ln \frac{1 - N(b_t; \widetilde{W}^m)}{N(b_t; \widetilde{W}^m)},$
- Обновление весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}, i = 1, \ldots, m,$
- Перенормировка весов $w_i := \frac{w_i}{\sum_{j=1}^m w_j}, i = 1, \ldots, m.$

Об обновлении весов

Замечание относительно шага обновления весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}$, $i = 1, \ldots, m$.

Об обновлении весов

Замечание относительно шага обновления весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}$, $i = 1, \ldots, m$.

- Вес объекта x_i увеличивается в e^{α_t} раз, когда b_t допускает на нем ошибку (т.к. $\alpha_t > 0$),

Об обновлении весов

Замечание относительно шага обновления весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}$, $i = 1, \dots, m$.

- Вес объекта x_i увеличивается в e^{α_t} раз, когда b_t допускает на нем ошибку (т.к. $\alpha_t > 0$),
- Вес объекта x_i уменьшается в e^{α_t} раз, когда b_t правильно его классифицирует,

Об обновлении весов

Замечание относительно шага обновления весов $w_i := w_i e^{-\alpha_t y_i b_t(x_i)}$, $i = 1, \ldots, m$.

- Вес объекта x_i увеличивается в e^{α_t} раз, когда b_t допускает на нем ошибку (т.к. $\alpha_t > 0$),
- Вес объекта x_i уменьшается в e^{α_t} раз, когда b_t правильно его классифицирует,
- Т.о. наибольший вес накапливается у тех объектов, которые чаще оказывались трудными для предыдущих алгоритмов.

Время для вопросов

Источники

На основе материалов сайта http://www.machinelearning.ru.