

Машинное обучение

Лекция 4

Построение ансамблей, Random Forest и
Gradient Boosting

Виктор Кантор

План

- I. Обзор методов построения композиций
- II. Ансамбли решающих деревьев

I. Обзор методов построения композиций

Bagging

Bagging = Bootstrap aggregation

Бутстреп

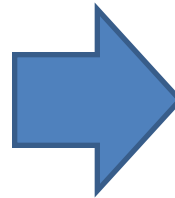
Выборка из некоторого
распределения:

№	значение
1	
2	
3	
N	

Бутстреп

Выборка из некоторого
распределения:

№	значение
1	
2	
3	
N	



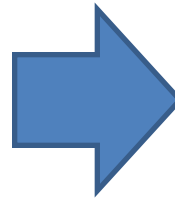
Хотим вычислить какую-то величину X по данным наблюдениями.

Было бы здорово вычислить X на многих выборках из распределения, а потом усреднить, но их у нас нет

Бутстреп

Выборка из некоторого распределения:

№	значение
1	
2	
3	
N	



Хотим вычислить какую-то величину X по данным наблюдениями.

Было бы здорово вычислить X на многих выборках из распределения, а потом усреднить, но их у нас нет

Решение:

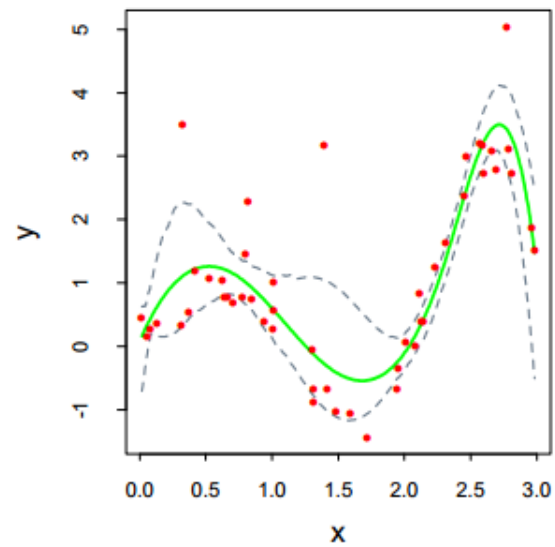
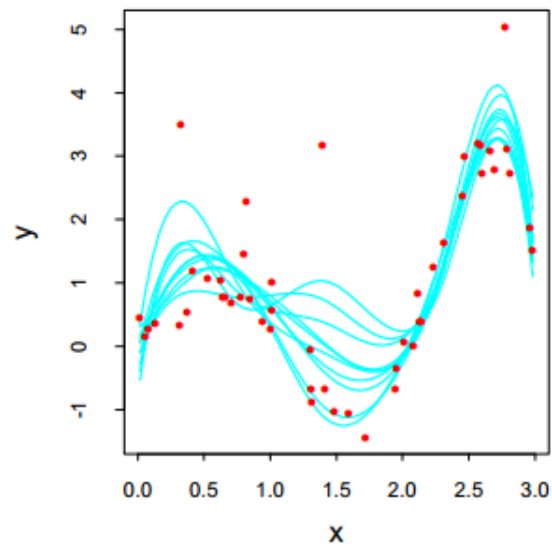
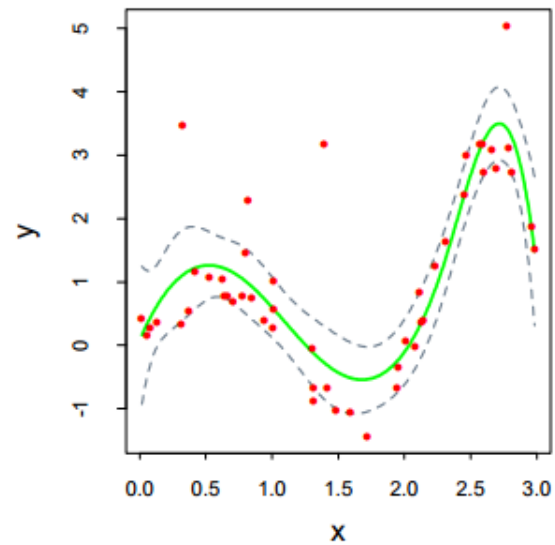
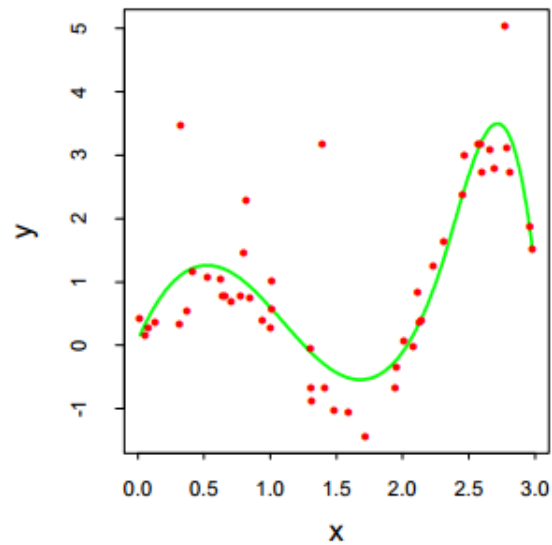
1. Выбираем наугад одно наблюдение из имеющихся.
2. Повторяем пункт 1 столько раз, сколько у нас есть наблюдений. При этом некоторые из них мы можем выбрать повторно
3. Считаем интересующие нас величины по новой выборке. Запоминаем результат.
4. Повторяем пункты 1-3 много раз и усредняем

Bagging

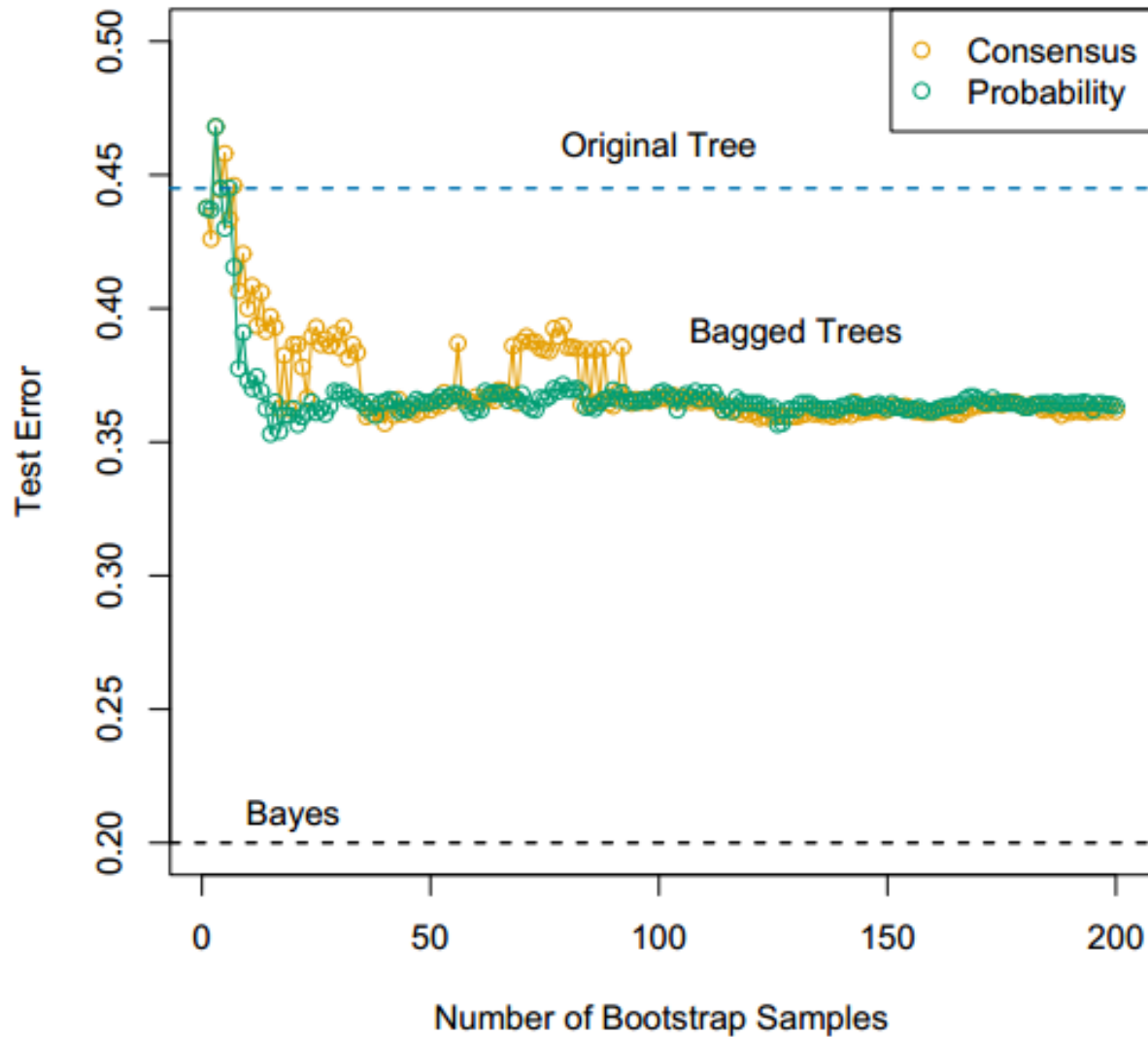
Bagging = Bootstrap aggregation

По схеме выбора с возвращением,
генерируем M обучающих выборок такого же
размера, обучаем на них модели и усредняем

Bagging



Бэггинг в классификации



Вариации: Pasting, RSM

- RSM – Random Subspace Method, выбираем не объекты, а признаки
- Pasting – выбираем объекты без возвращения

Stacking

Обучающая выборка:

[illegible]

Stacking

Обучающая выборка:

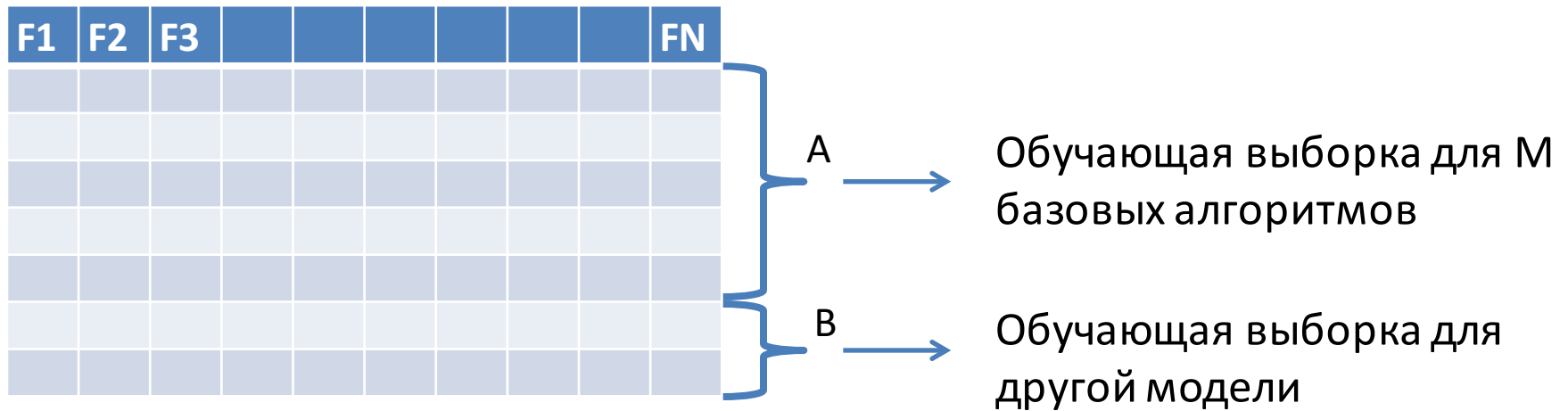
F1	F2	F3							FN

A

Обучающая выборка для M
базовых алгоритмов

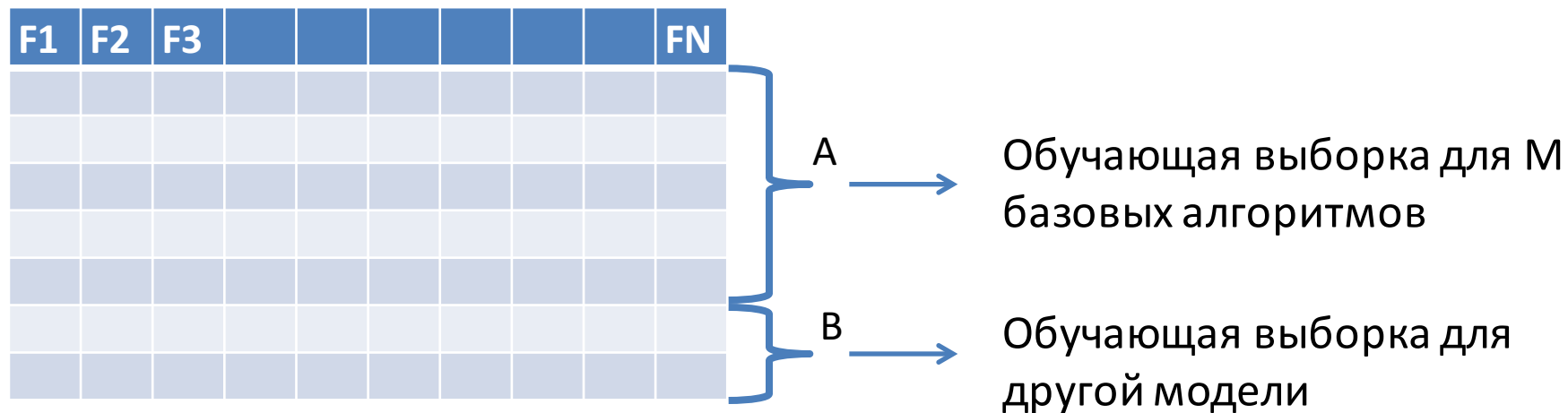
Stacking

Обучающая выборка:



Stacking

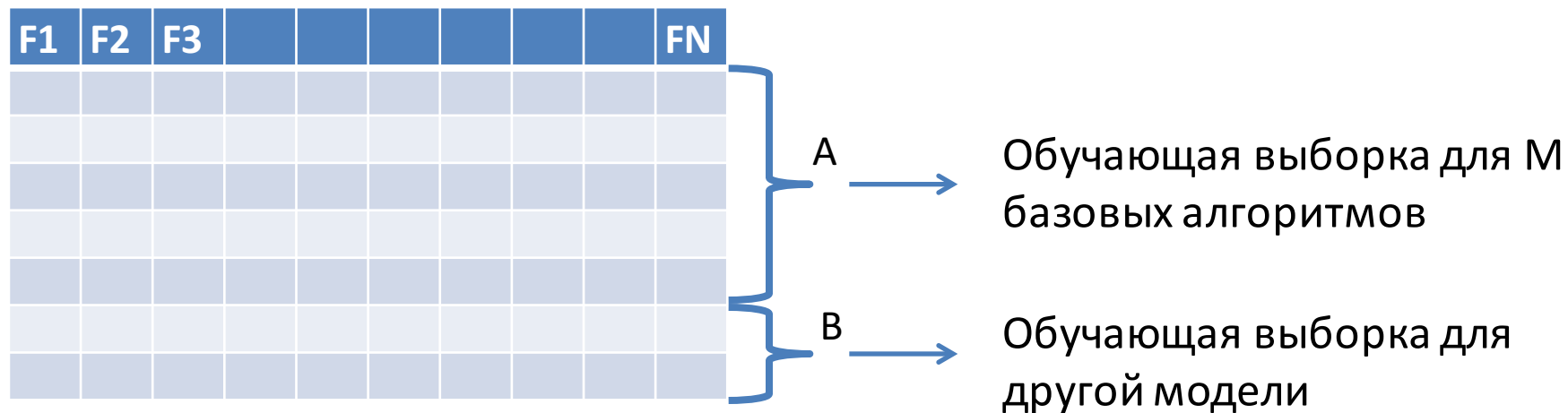
Обучающая выборка:



Обучаем M
базовых
алгоритмов на
выборке A

Stacking

Обучающая выборка:



Обучаем M
базовых
алгоритмов на
выборке A



Считаем их
прогнозы на
выборке B

Stacking

Обучающая выборка:

F1	F2	F3							FN

A

Обучающая выборка для M базовых алгоритмов

B

Обучающая выборка для другой модели

Обучаем M базовых алгоритмов на выборке A

Считаем их прогнозы на выборке B

B1	B2			BM

Stacking

Обучающая выборка:

F1	F2	F3							FN

A

Обучающая выборка для M базовых алгоритмов

B

Обучающая выборка для другой модели

Обучаем M базовых алгоритмов на выборке A

Считаем их прогнозы на выборке B

B1	B2			BM

Обучаем другую модель (например, линейную регрессию с $w_0 = 0$)

Stacking

Обучающая выборка:

F1	F2	F3							FN

A

Обучающая выборка для M базовых алгоритмов

B

Обучающая выборка для другой модели

Обучаем M базовых алгоритмов на выборке A

Считаем их прогнозы на выборке B

B1	B2			BM

Обучаем другую модель (например, линейную регрессию с $w_0 = 0$)

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

Blending

Смесь нескольких сильных классификаторов:

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

+ веса неотрицательны и дают в сумме единицу

Blending

Смесь нескольких сильных классификаторов:

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

+ веса неотрицательны и дают в сумме единицу

Преимущества и недостатки:

- Очень прост идейно, хорошо работает, логичен

Blending

Смесь нескольких сильных классификаторов:

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

+ веса неотрицательны и дают в сумме единицу

Преимущества и недостатки:

- Очень прост идейно, хорошо работает, логичен
- Иногда надо перебирать веса или использовать дискретную оптимизацию

Blending

Смесь нескольких сильных классификаторов:

$$a(x) = \sum_{t=1}^T \alpha_t b_t(x)$$

+ веса неотрицательны и дают в сумме единицу

Преимущества и недостатки:

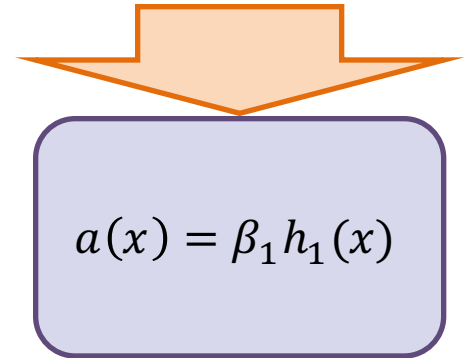
- Очень прост идейно, хорошо работает, логичен
- Иногда надо перебирать веса или использовать дискретную оптимизацию
- Не всегда композиция в виде взвешенной суммы – то, что надо. Иногда нужна более сложная композиция

Boosting

Бустинг – жадное построение
взвешенной суммы базовых
алгоритмов $h_k(x)$

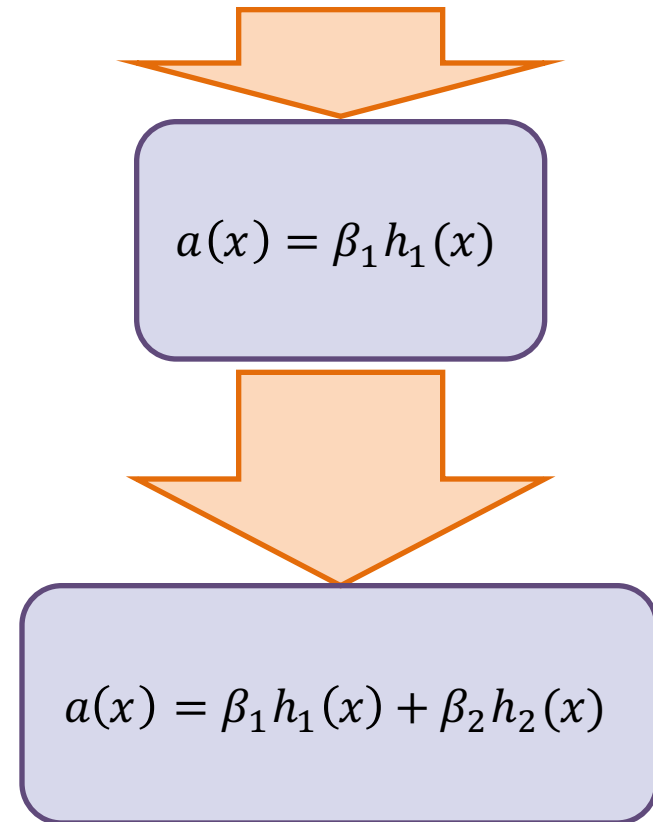
Boosting

Бустинг – жадное построение
взвешенной суммы базовых
алгоритмов $h_k(x)$



Boosting

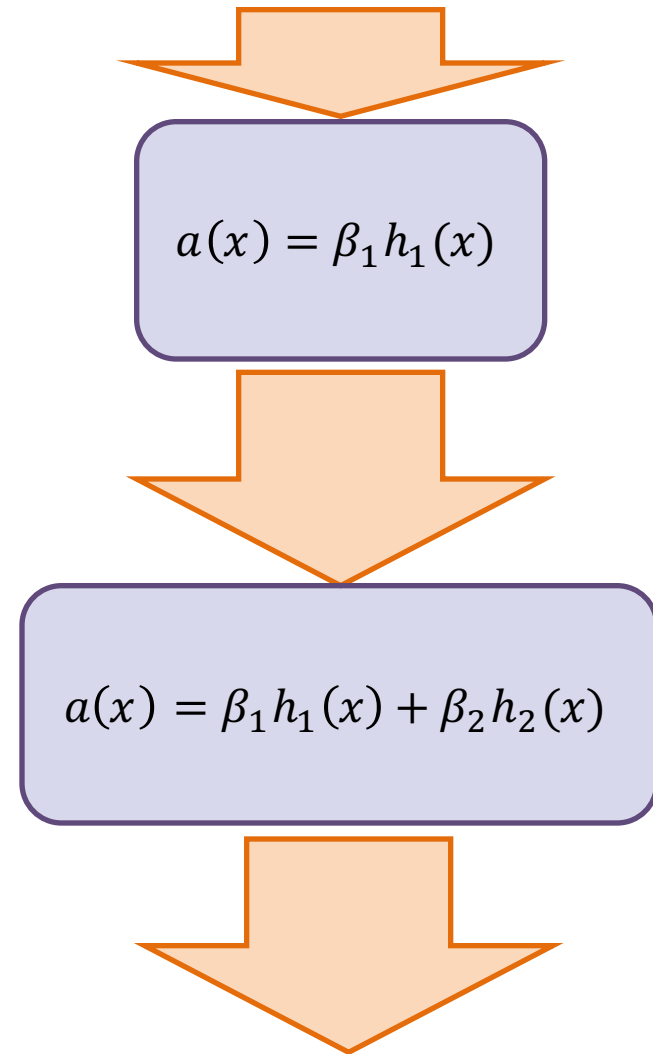
Бустинг – жадное построение
взвешенной суммы базовых
алгоритмов $h_k(x)$



Boosting

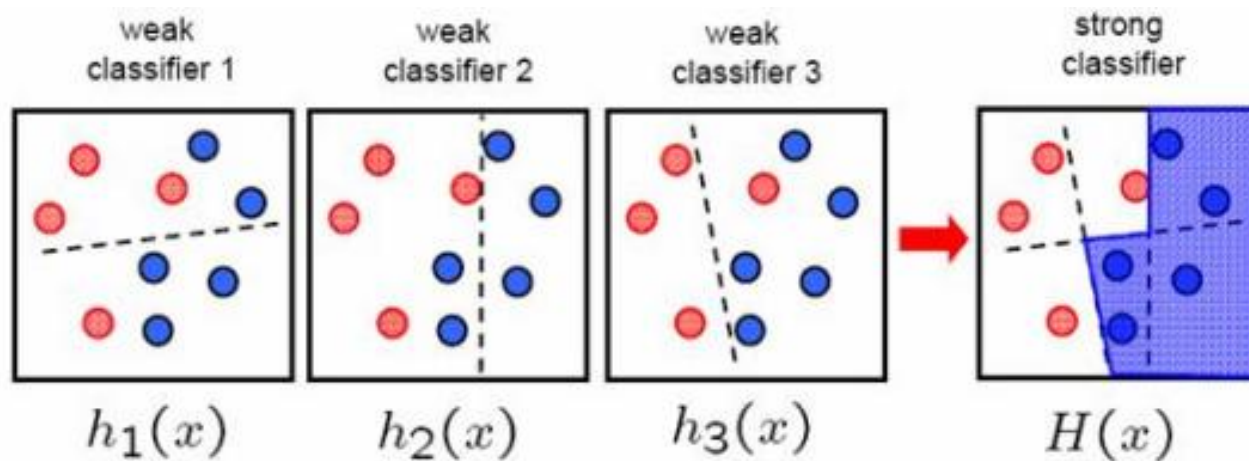
Бустинг – жадное построение взвешенной суммы базовых алгоритмов $h_k(x)$

$$a(x) = \sum_{t=1}^T \beta_t h_t(x)$$



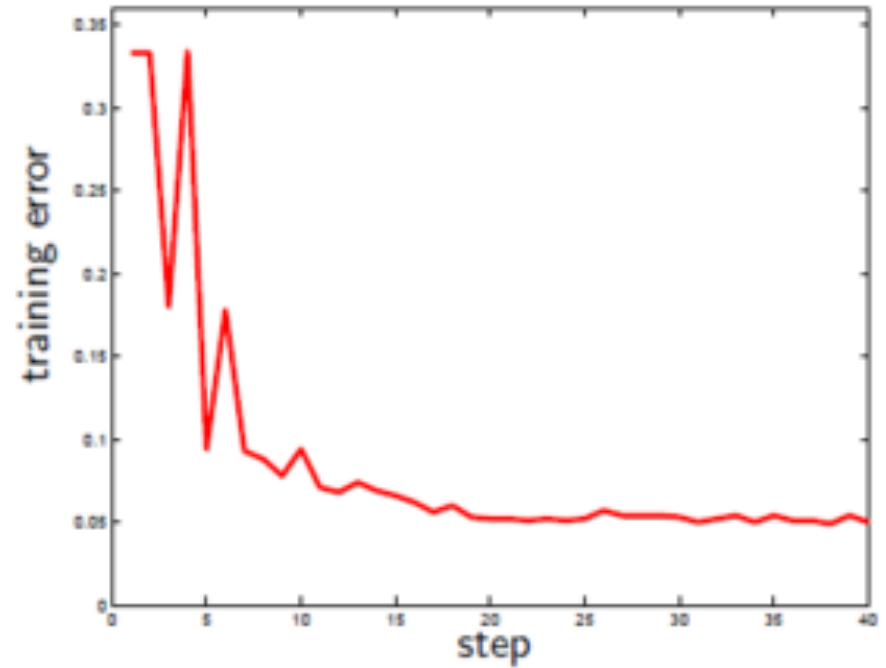
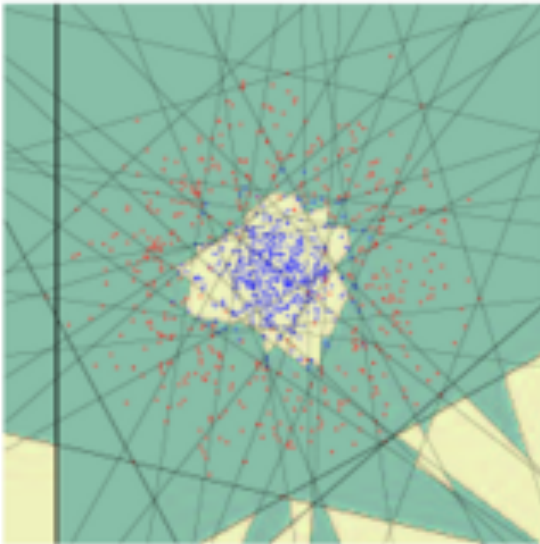
«Слабые» алгоритмы

$h_k(x)$ – как правило, решающие деревья небольшой глубины или линейные модели



Пример: бустинг над линейными классификаторами

$t = 40$

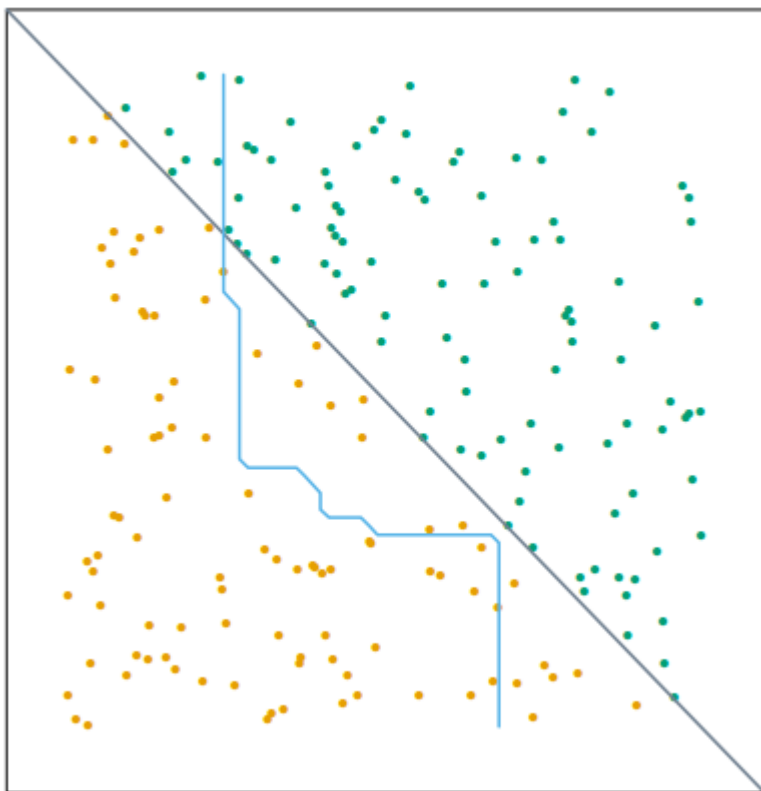


Алгоритмы бустинга

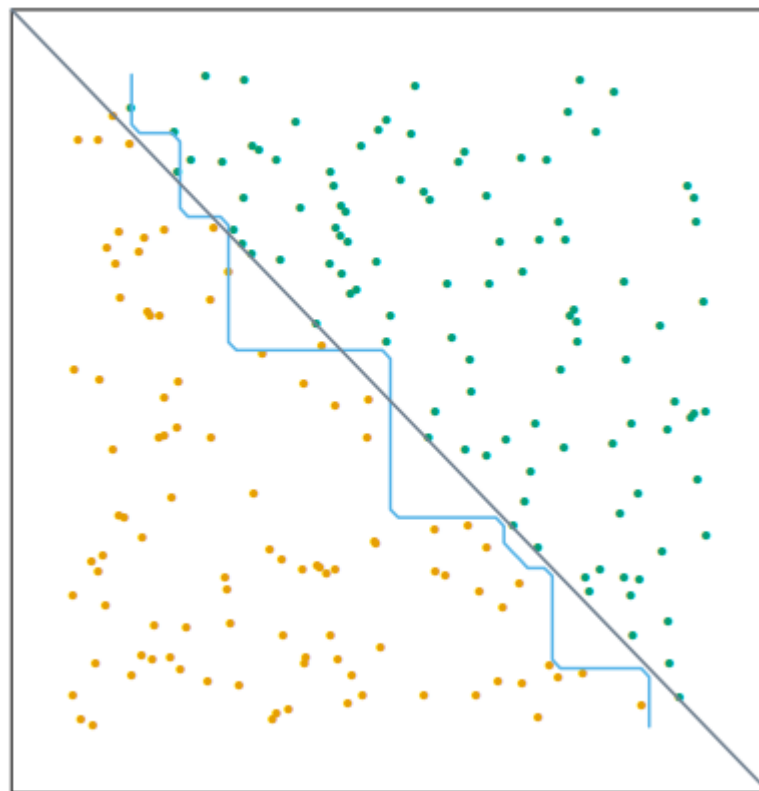
- Основные алгоритмы:
 - Градиентный бустинг
 - Адаптивный бустинг (AdaBoost)
- Вариации AdaBoost:
 - AnyBoost (произвольная функция потерь)
 - BrownBoost
 - GentleBoost
 - LogitBoost
 -

Бэггинг и бустинг

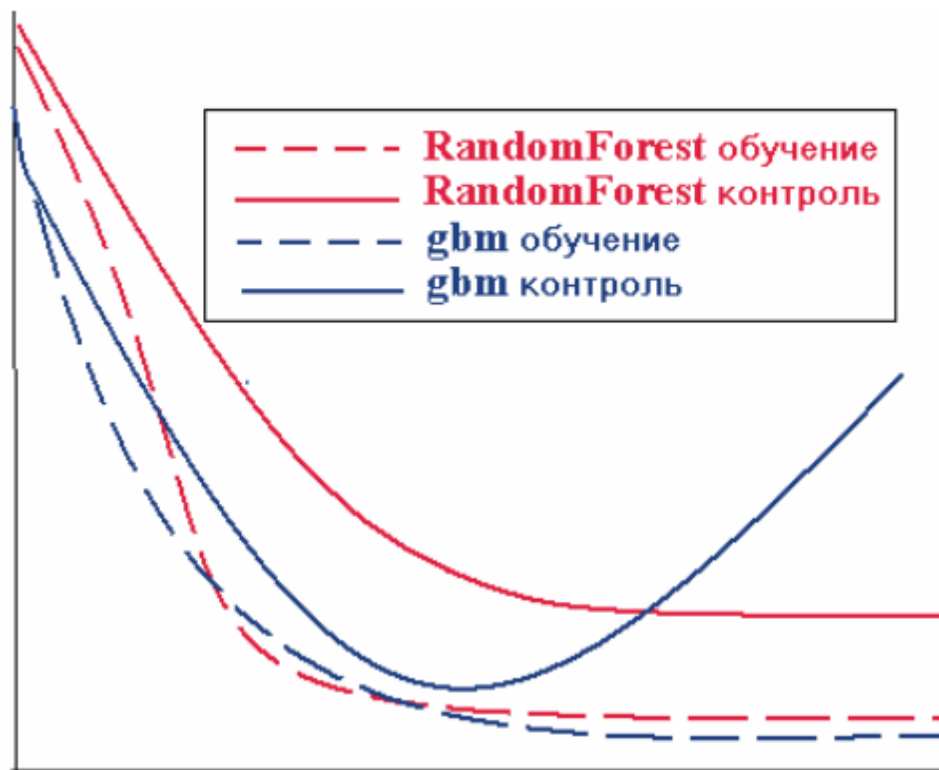
Bagged Decision Rule



Boosted Decision Rule



Бэггинг и бустинг: переобучение



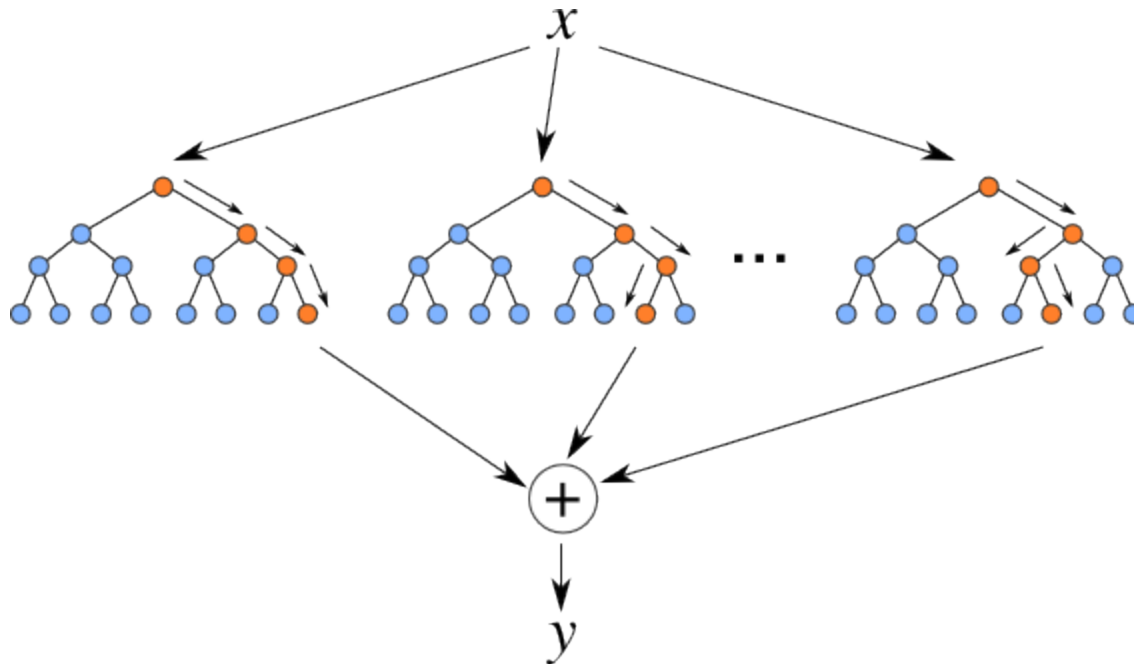
Преимущества и недостатки бустинга

- Позволяет очень точно приблизить восстанавливаемую функцию или разделяющую поверхность классов
- Плохо интерпретируем
- Композиции могут содержать десятки тысяч базовых моделей и долго обучаться
- Переобучение на выбросах при избыточном количестве классификаторов

II. Ансамбли решающих деревьев

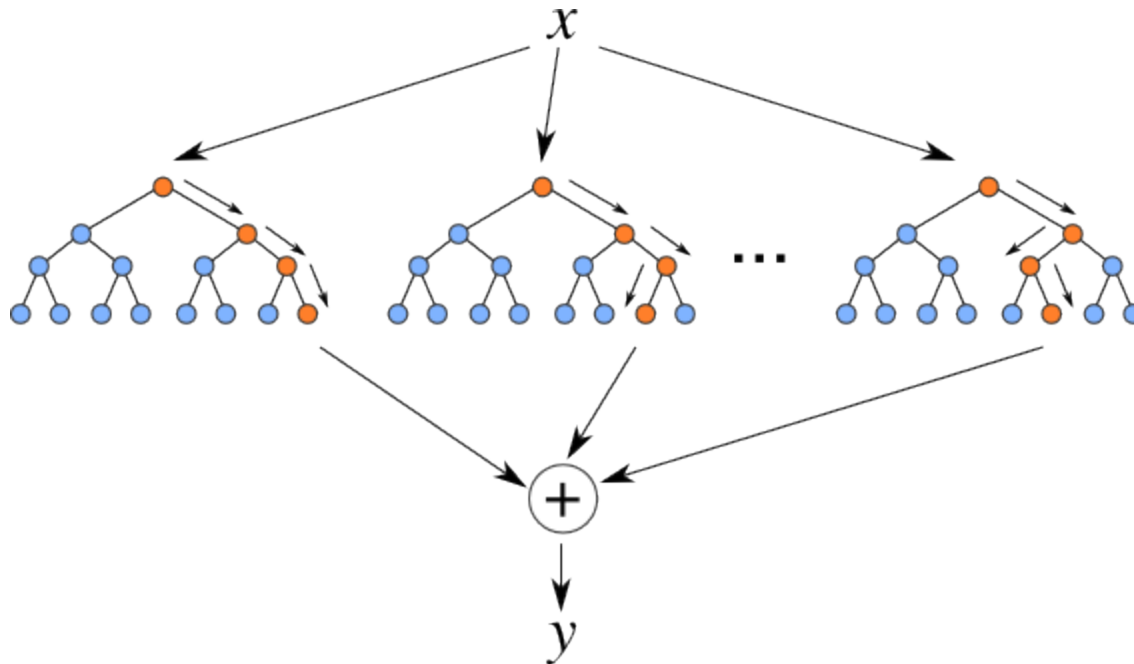
Леса решающих деревьев

Random Forest



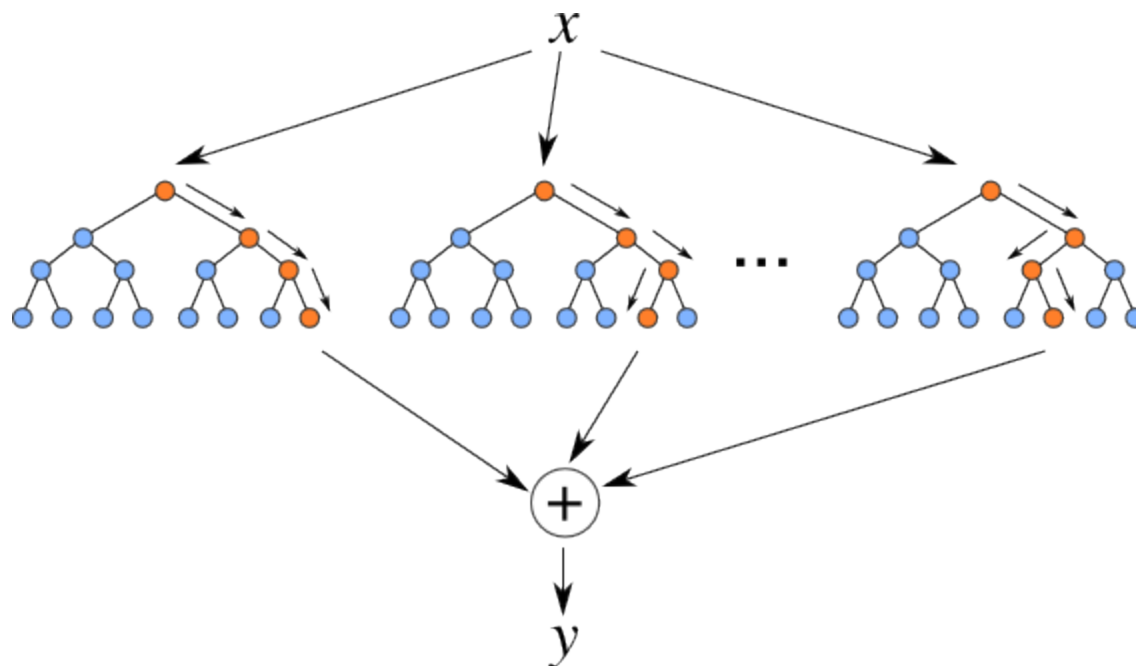
1. Бэггинг над деревьями
2. Рандомизированные разбиения в деревьях: выбираем k случайных признаков и ищем наиболее информативное разбиение по ним

Extreemly Randomized Trees



1. Бэггинг над «сильно рандомизированными» деревьями
2. При разбиении в дереве выбираем k случайных признаков и случайные пороги по ним, затем ищем наиболее информативное из этих разбиений

Extreemly Randomized Trees



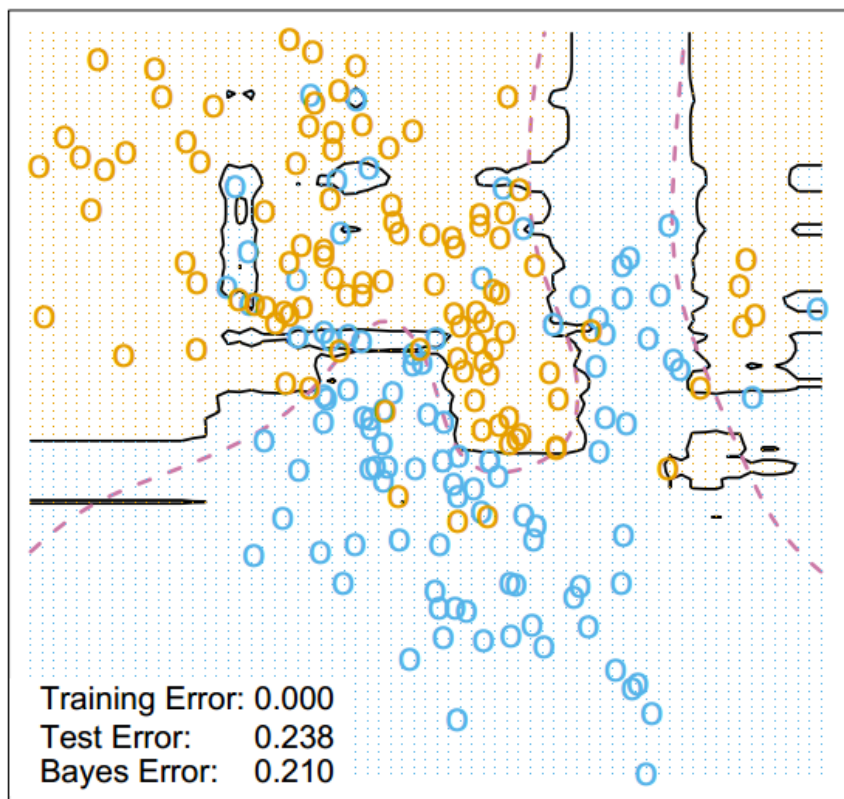
1. Бэггинг над «сильно рандомизированными» деревьями
2. При разбиении в дереве выбираем k случайных признаков и случайные пороги по ним, затем ищем наиболее информативное из этих разбиений

Нестандартные применения случайного леса

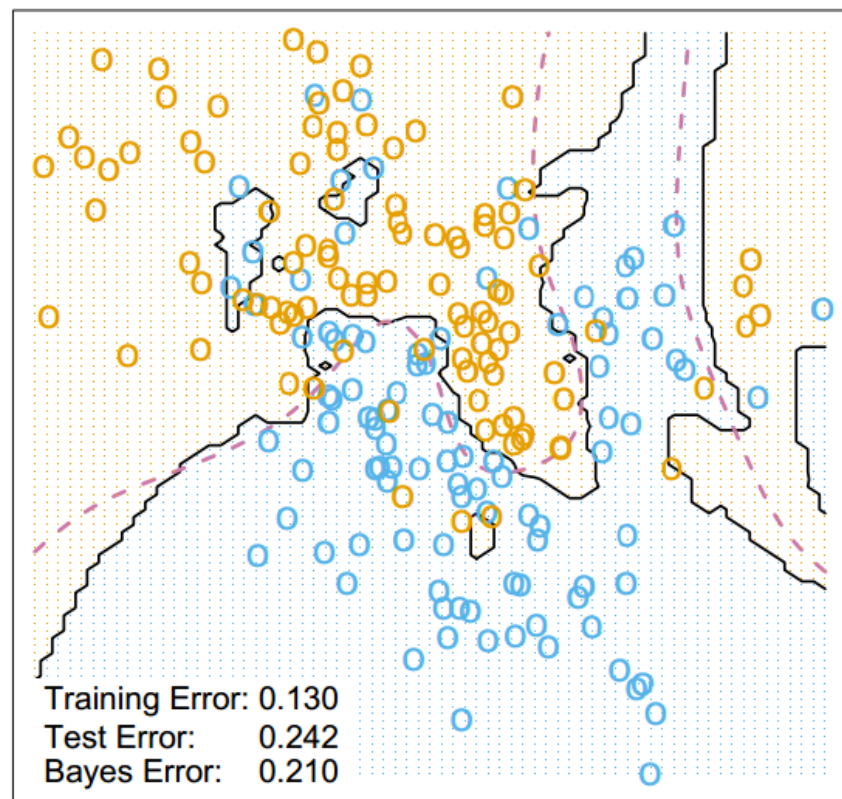
- Метрика и поиск похожих объектов
- Преобразование признаков

Может ли *** работать лучше RF

Random Forest Classifier



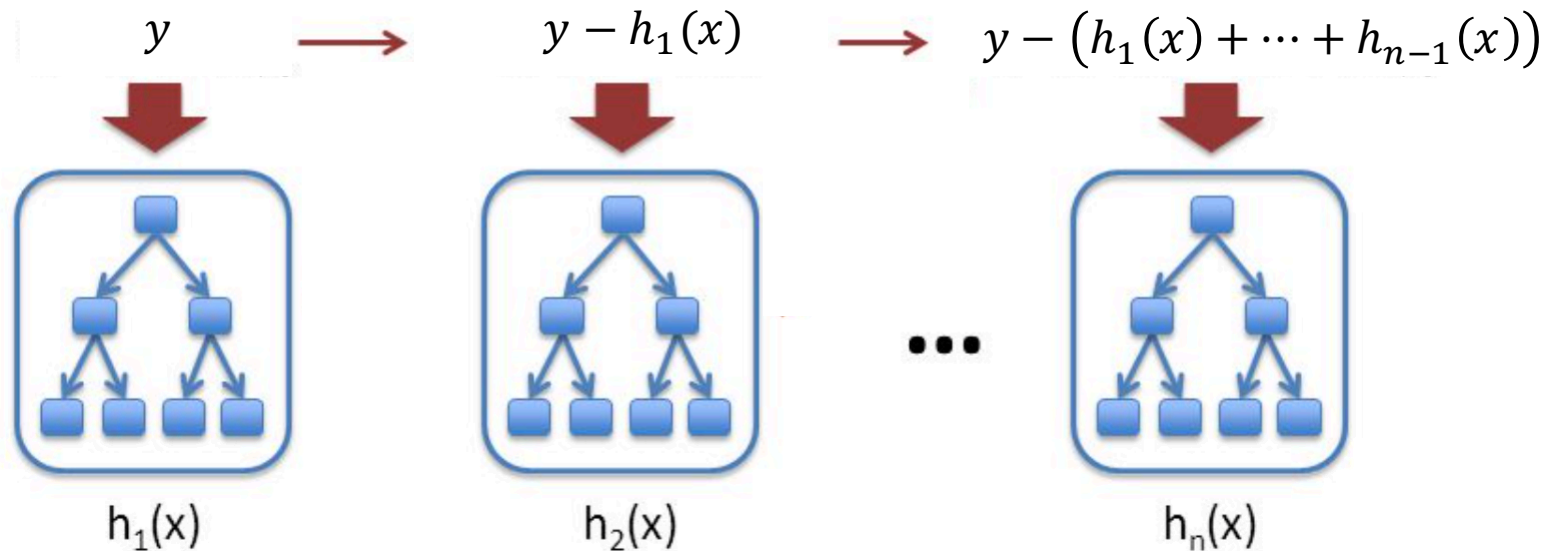
3-Nearest Neighbors



Градиентный бустинг над деревьями

Идея Gradient Boosted Decision Trees (с прошлой лекции)

$$a_n(x) = h_1(x) + \dots + h_n(x)$$



Аналогия с численной оптимизацией

Нам нужно минимизировать ошибку:

$$Q(\hat{y}, y) = \sum_{i=1}^l (\hat{y}_i - y_i)^2 \rightarrow \min \qquad \hat{y}_i = a(x_i)$$

Аналогия с численной оптимизацией

Нам нужно минимизировать ошибку:

$$Q(\hat{y}, y) = \sum_{i=1}^l (\hat{y}_i - y_i)^2 \rightarrow \min \quad \hat{y}_i = a(x_i)$$

Если бы мы подбирали ответы \hat{y} итеративно, можно было бы это делать градиентным спуском

Аналогия с численной оптимизацией

Нам нужно минимизировать ошибку:

$$Q(\hat{y}, y) = \sum_{i=1}^l (\hat{y}_i - y_i)^2 \rightarrow \min \quad \hat{y}_i = a(x_i)$$

Если бы мы подбирали ответы \hat{y} итеративно, можно было бы это делать градиентным спуском

Но нам нужно подобрать не ответы, а функцию $a(x)$

Градиентный бустинг и градиент

В бустинге

$$a(x) = \sum_{t=1}^T \beta_t h_t(x)$$

Идея: будем каждый следующий алгоритм выбирать так, чтобы он приближал антиградиент ошибки

$$h_t(x) \approx -\frac{\partial Q(\hat{y}, y)}{\partial \hat{y}}$$

Градиентный бустинг и градиент

Если $h_t(x) \approx -\frac{\partial Q(\hat{y}, y)}{\partial \hat{y}}$ и $Q(\hat{y}, y) = \sum_{i=1}^l (\hat{y}_i - y_i)^2$

$$h_t(x_i) \approx -\frac{\partial Q(\hat{y}_i, y_i)}{\partial \hat{y}_i} = -2(\hat{y}_i - y_i) \propto y_i - \hat{y}_i$$

GBM с квадратичными потерями

1. Обучаем первый базовый алгоритм h_1 , $\beta_1 = 1$
2. Повторяем в цикле по t от 2 до T :

обучаем h_t на ответы $y_i - a_{t-1}(x_i)$

выбираем β_t

GBM с квадратичными потерями

1. Обучаем первый базовый алгоритм h_1 , $\beta_1 = 1$
2. Повторяем в цикле по t от 2 до T :

обучаем h_t на ответы $y_i - a_{t-1}(x_i)$

выбираем β_t

Стратегии выбора β_t :

- всегда равен небольшой константе
- как в методе наискорейшего спуска
- уменьшая с ростом t

GBM с произвольными потерями

1. Обучаем первый базовый алгоритм h_1 , $\beta_1 = 1$
2. Повторяем в цикле по t от 2 до T :

обучаем h_t на $-\frac{\partial Q(\hat{y}_i, y_i)}{\partial \hat{y}_i} = -\frac{\partial L(\hat{y}_i, y_i)}{\partial \hat{y}_i}$

выбираем β_t

Здесь $Q(\hat{y}, y) = \sum_{i=1}^l L(\hat{y}_i, y_i)$ $\hat{y}_i = a_{t-1}(x_i)$

GBM в наиболее общем виде

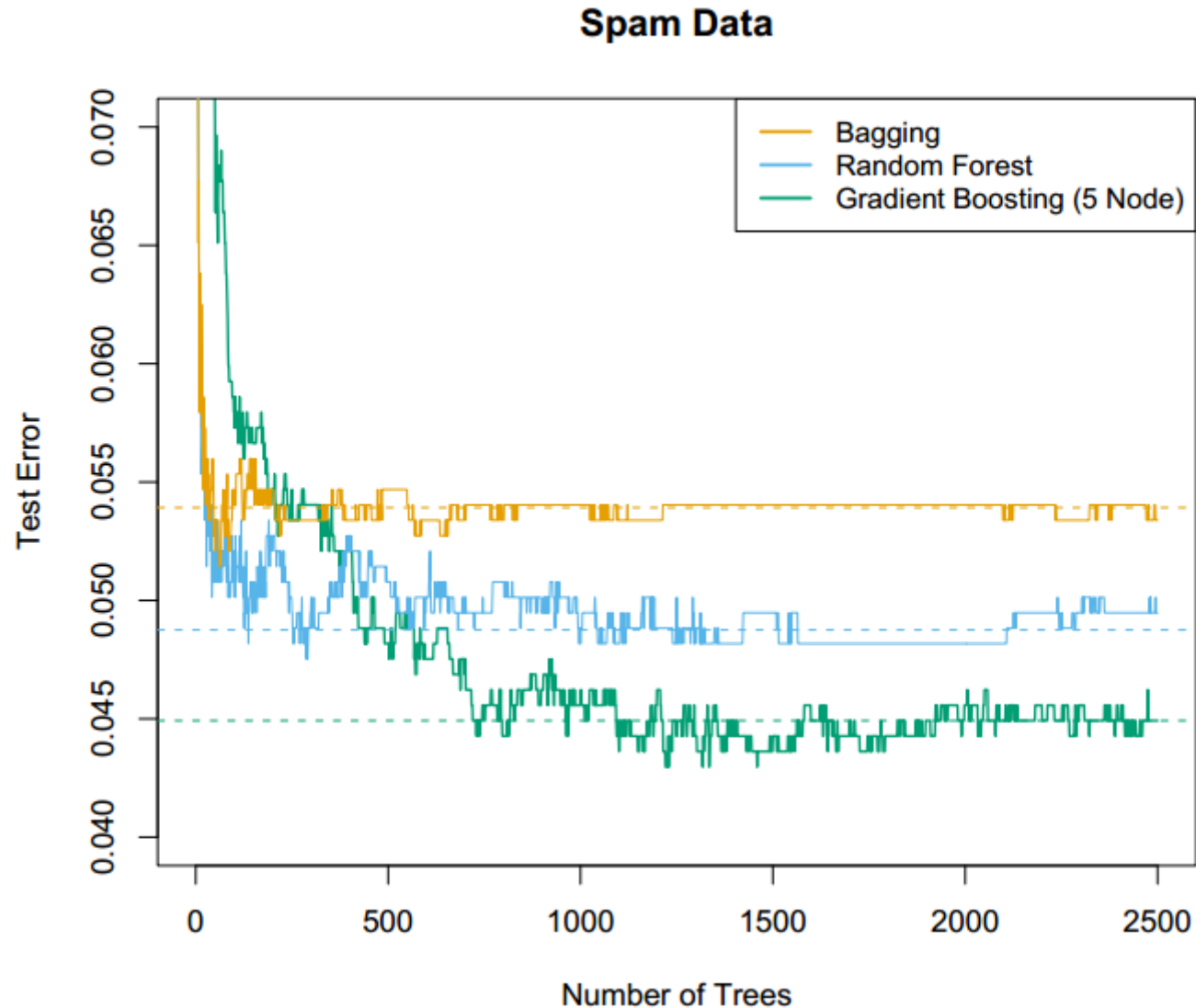
1. Обучаем первый базовый алгоритм h_1 , $\beta_1 = 1$
2. Повторяем в цикле по t от 2 до T :

$$h_t = \operatorname{argmin}_h \sum_{i=1}^l \tilde{L} \left(h(x_i), -\frac{\partial L(\hat{y}_i, y_i)}{\partial \hat{y}_i} \right)$$

выбираем β_t

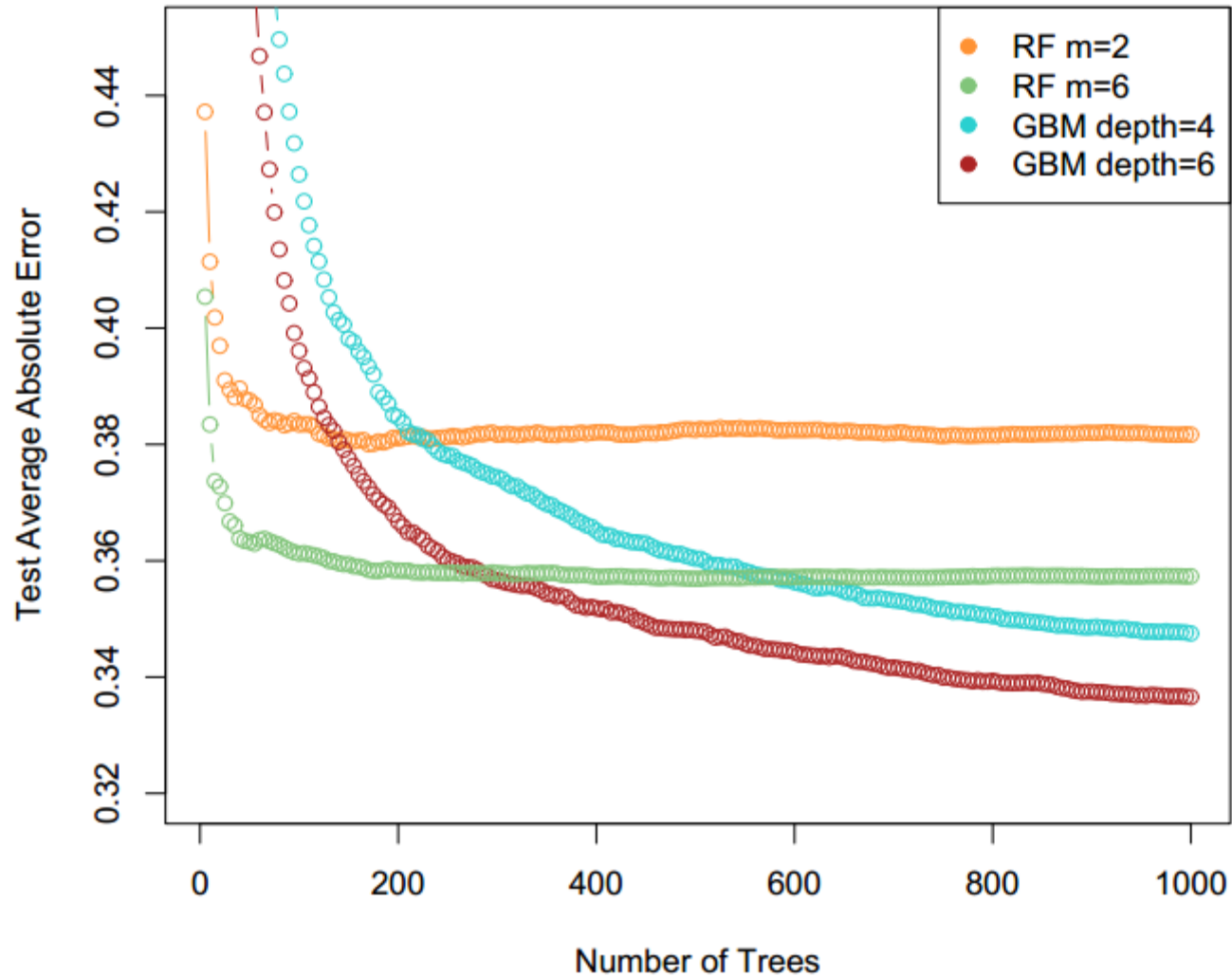
$$\text{Здесь } Q(\hat{y}, y) = \sum_{i=1}^l L(\hat{y}_i, y_i) \qquad \hat{y}_i = a_{t-1}(x_i)$$

Bagging, Random Forest, GBDT



GTBM и RF

California Housing Data



Распараллеливание

Вопрос для обсуждения:

Какой из ансамблей деревьев больше подходит для распараллеливания? Как это делать в одном и в другом случае?

Резюме

I. Обзор методов построения композиций

- Bagging
- Stacking
- Blending
- Boosting

II. Ансамбли решающих деревьев

- Random Forest
- Gradient Boosted Decision Trees

Отзывы

Отзывы о прошедших лекциях и семинарах
можно и нужно оставлять здесь:

<https://ml-mipt.github.io/2017part1/>