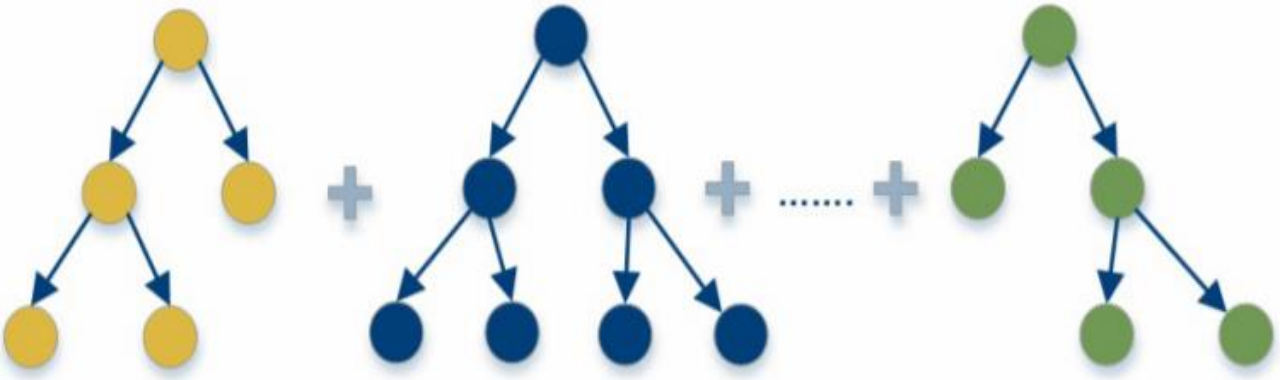
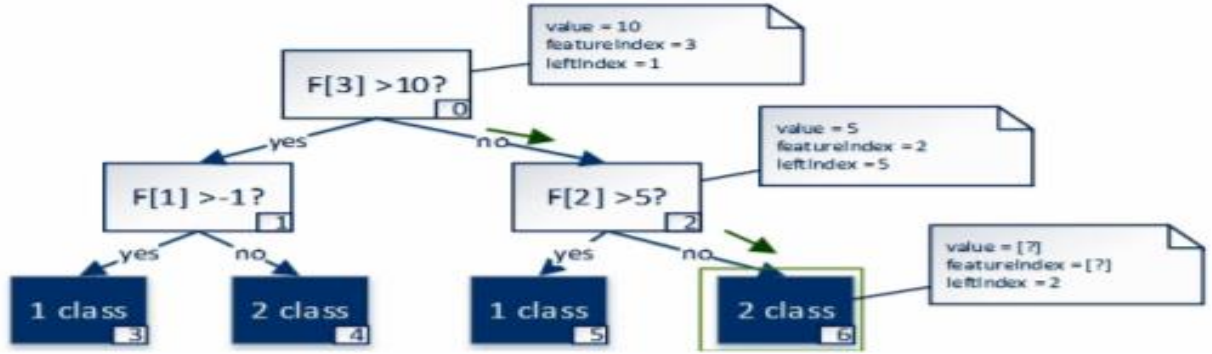
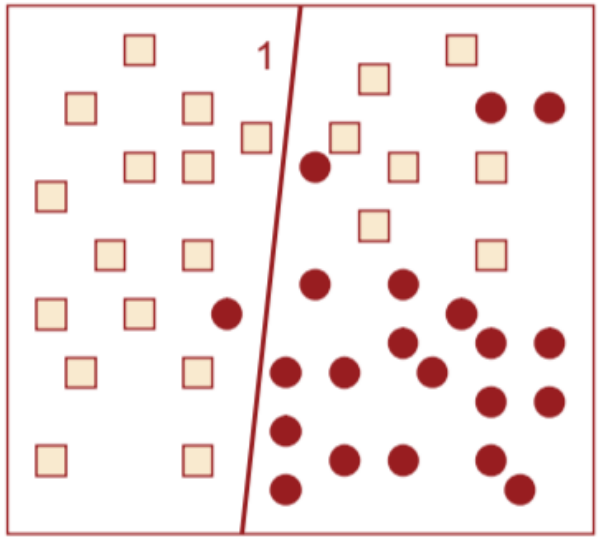


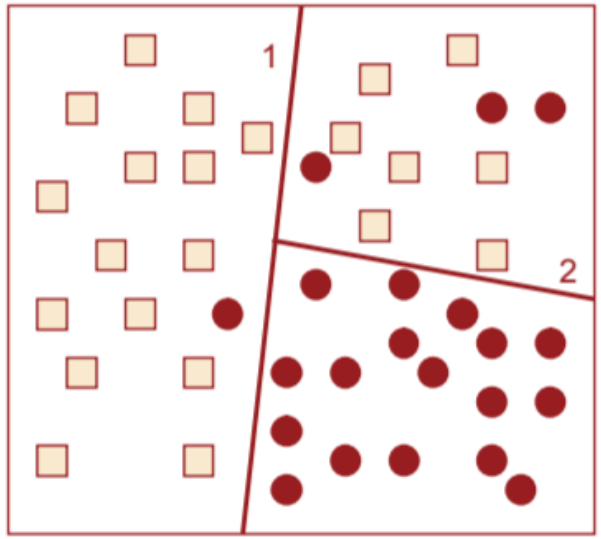


Основные понятия и определения бустинга

ФИНАНСОВЫЙ УНИВЕРСИТЕТ
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ

Идея бустинга

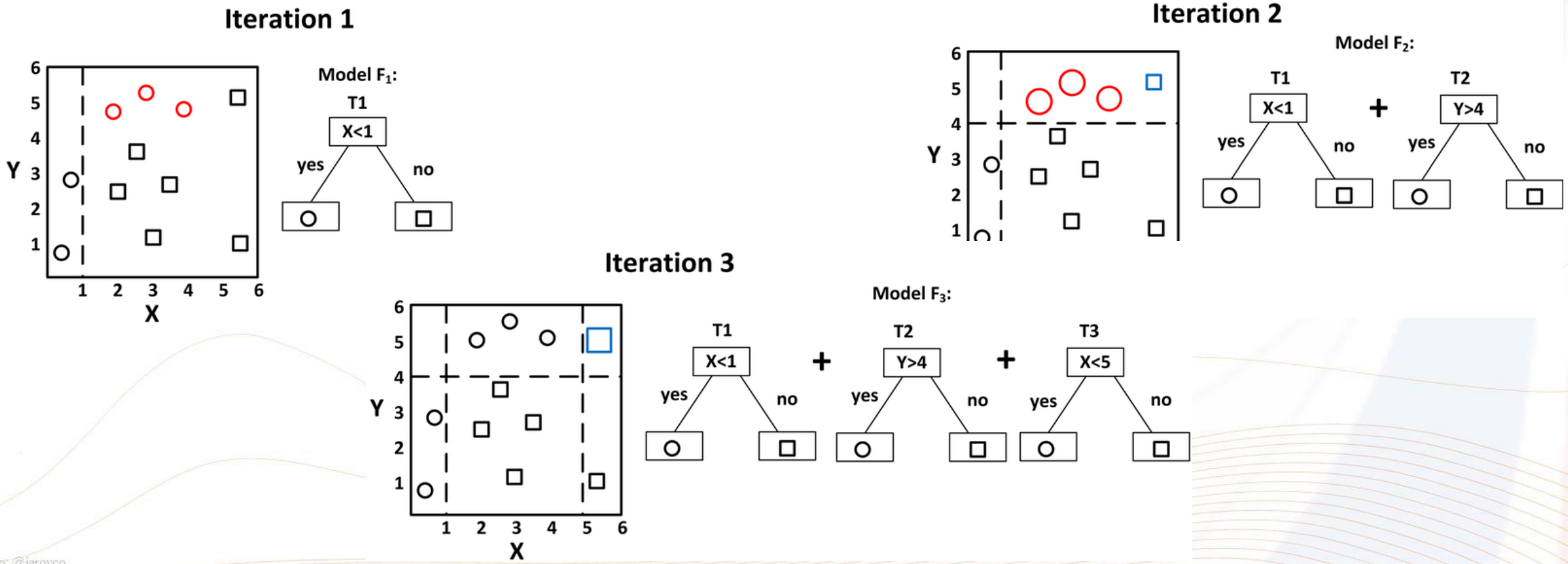




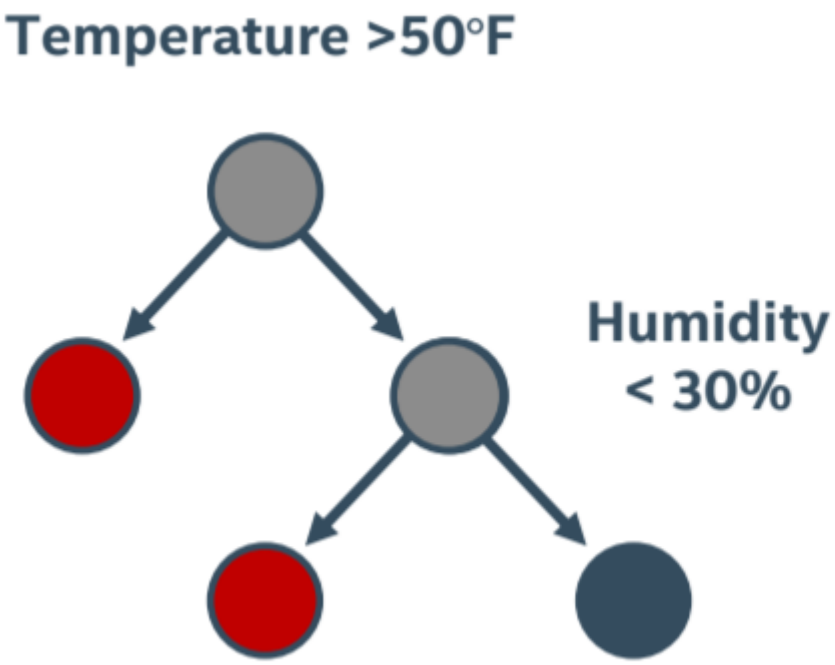
На рисунке изображено множество примеров, содержащее два класса — круги и квадраты. В результате обучения построена модель, разделившая классы по линии 1. Поскольку классы пересекаются, они являются трудноразделимыми, из-за чего значительное количество квадратов оказалось распознано как круги. Ожидать от такой модели хорошей работы с новыми данными не следует, при этом ее возможности уже исчерпаны: как бы мы ни провели линию 1, всегда будет присутствовать значительная ошибка классификации. Неизбежно возникает вопрос: как ***усилить «слабую» модель***, что сделать для ***повышения*** эффективности классификации? Вполне логичным выходом из ситуации является попытка применить к неудачным результатам работы первой модели еще одну модель, задача которой — классифицировать те примеры, что остались нераспознанными. Результаты работы второй модели представлены на втором рисунке. Как видно, ей удалось почти полностью разделить круги и квадраты. Если и после этого результаты неудовлетворительны, можно применить третью модель и т. д. до тех пор, пока не будет получено достаточно точное решение. Таким образом, для решения одной задачи классификации или регрессии мы применили несколько моделей, при этом нас интересует не результат работы каждой отдельной модели, а результат, который дает весь набор моделей. Такие совокупности моделей называются ансамблями моделей

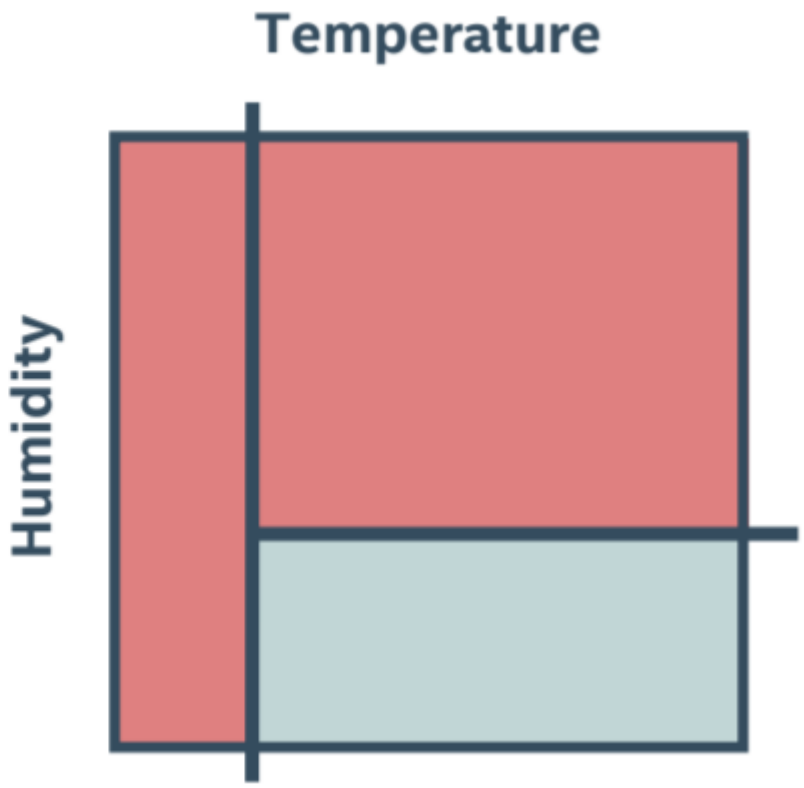
Идея бустинга

Таким образом, параметрами, настраиваемыми на каждой итерации, являются веса примеров. При этом чем больше раз пример был неправильно распознан предыдущими моделями, тем выше его вес. Веса примеров увеличиваются или уменьшаются в зависимости от выхода каждого нового классификатора. В результате некоторые трудные примеры могут стать еще труднее, а легкие — еще легче, и наоборот. После каждой итерации веса отражают, насколько часто тот или иной пример классифицировался неправильно.



Базовый ученик



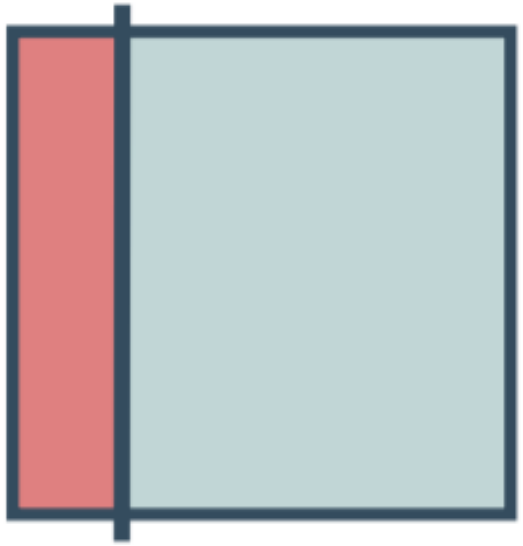


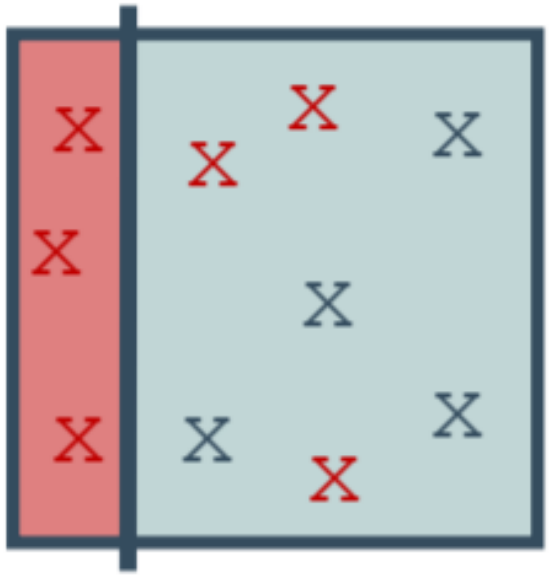
Обзор бустинга

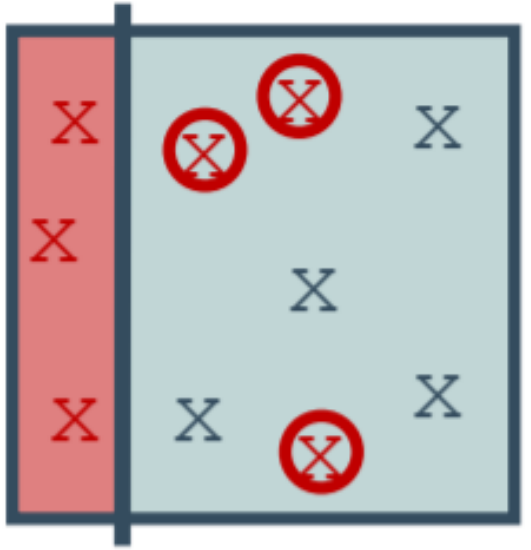
1 Создать начальное одноуровневое дерево решений высотой 1 (decision stump)

2 Обучить на данных и рассчитать ошибки (остатки)

3 Изменить веса примеров

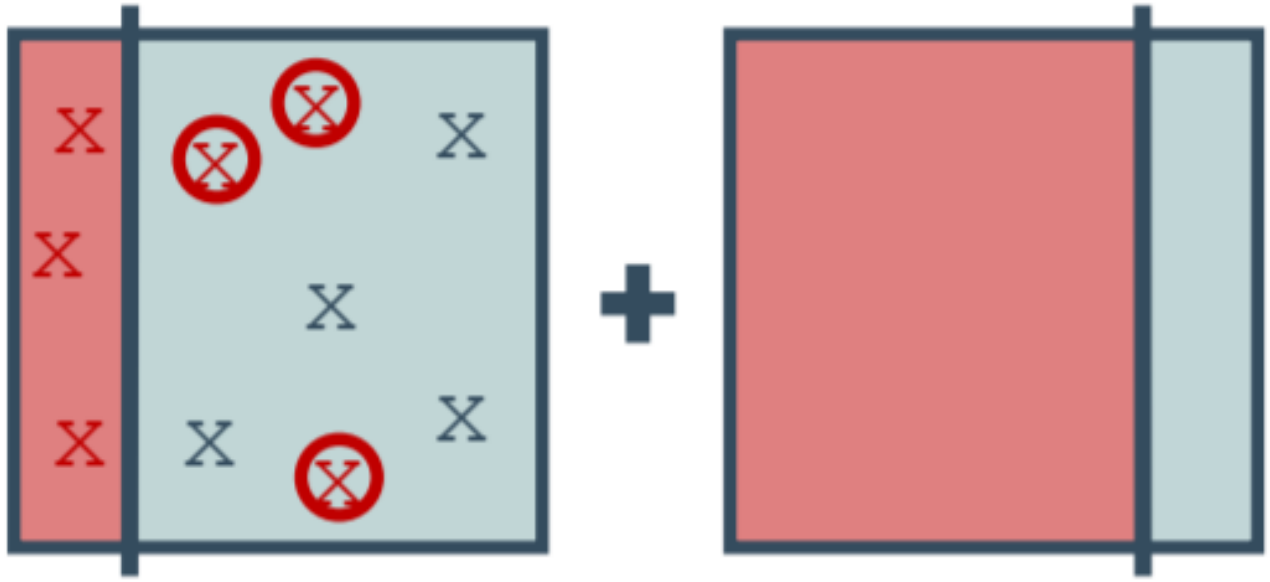






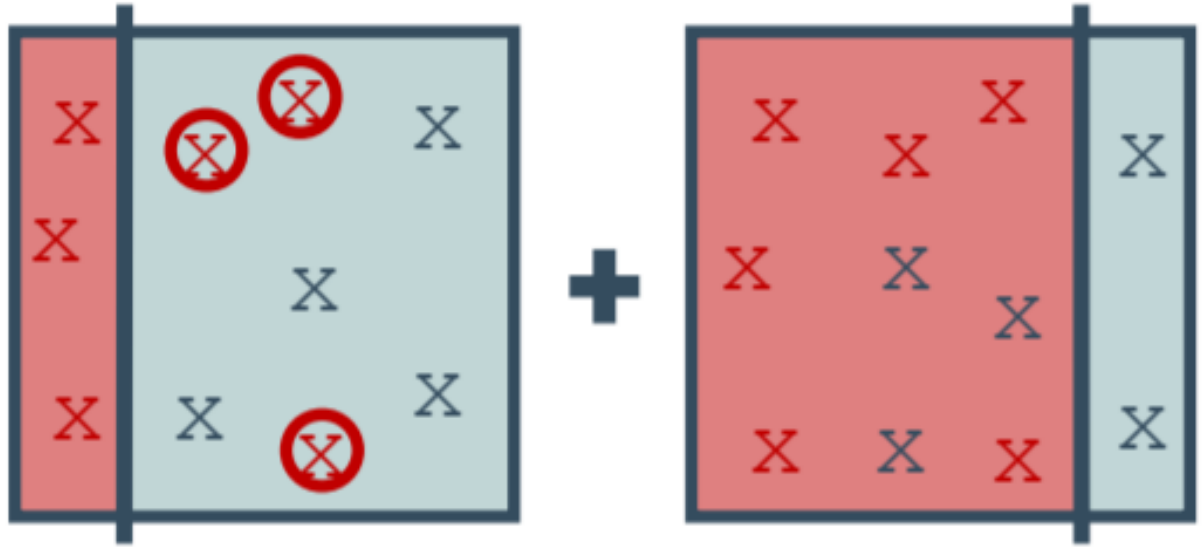
Обзор бустинга

Найти новое одноуровневое дерево решений (decision stump) , чтобы соответствовать взвешенным остаткам



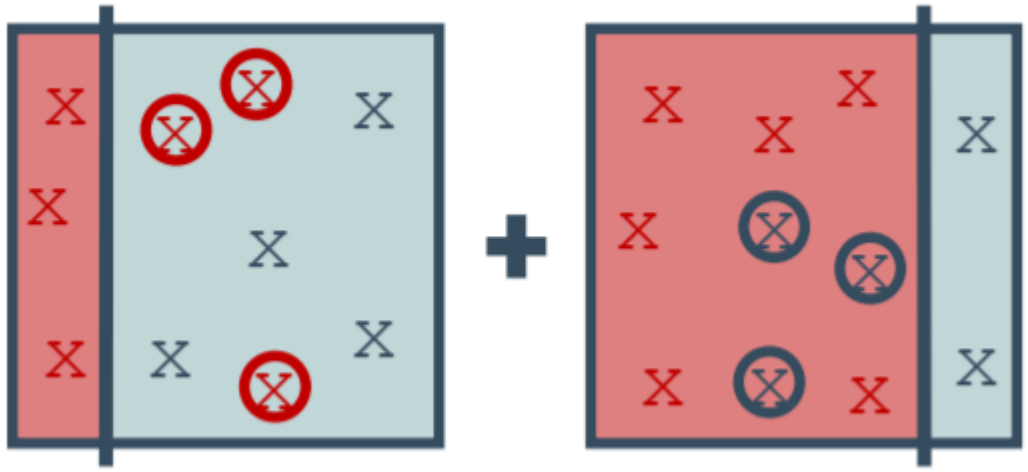
Обзор бустинга

Подгоните (обучите) новое одноуровневое дерево решений (decision stump) к текущим остаткам



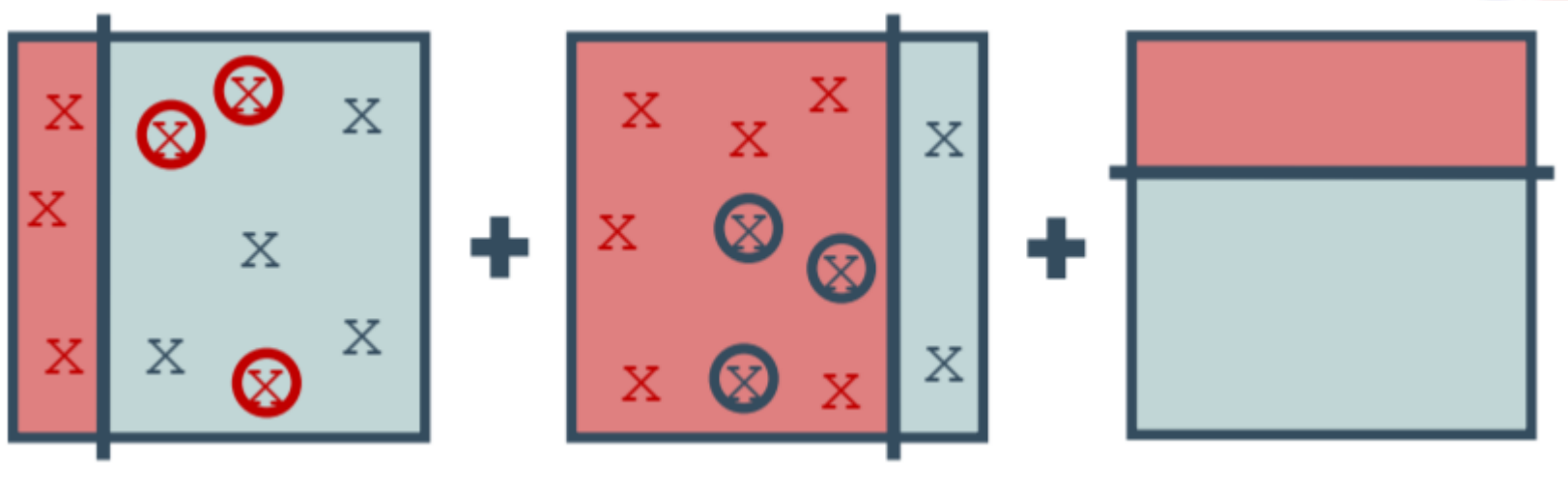
Обзор бустинга

Расчет ошибок и весов примеров



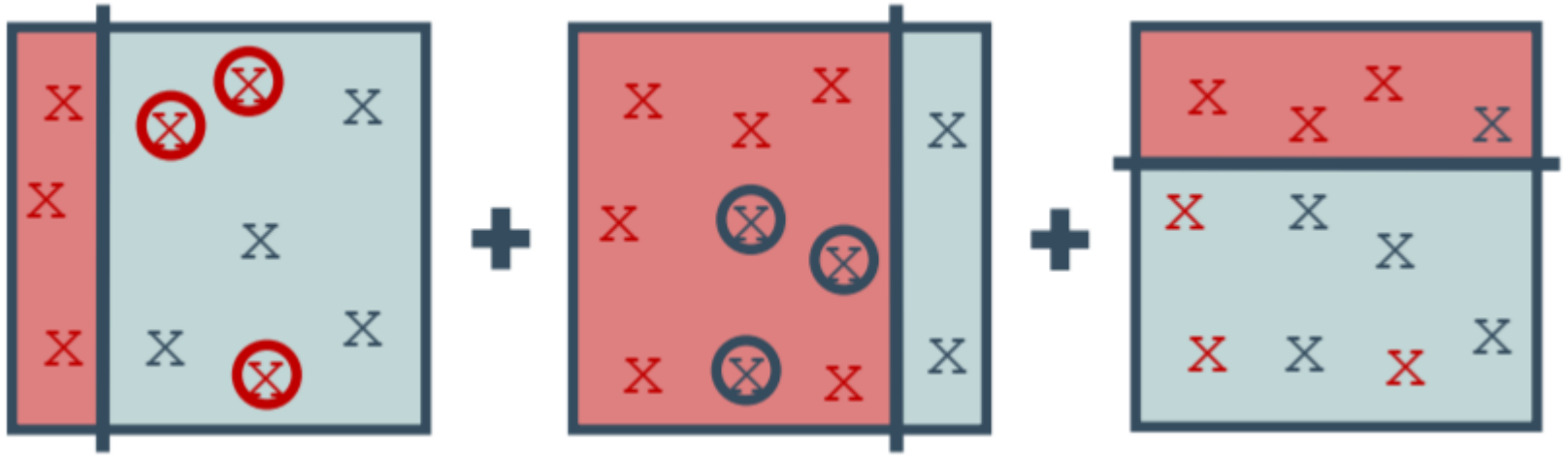
Обзор бустинга

Найти новое одноуровневое дерево, чтобы соответствовать взвешенным остаткам

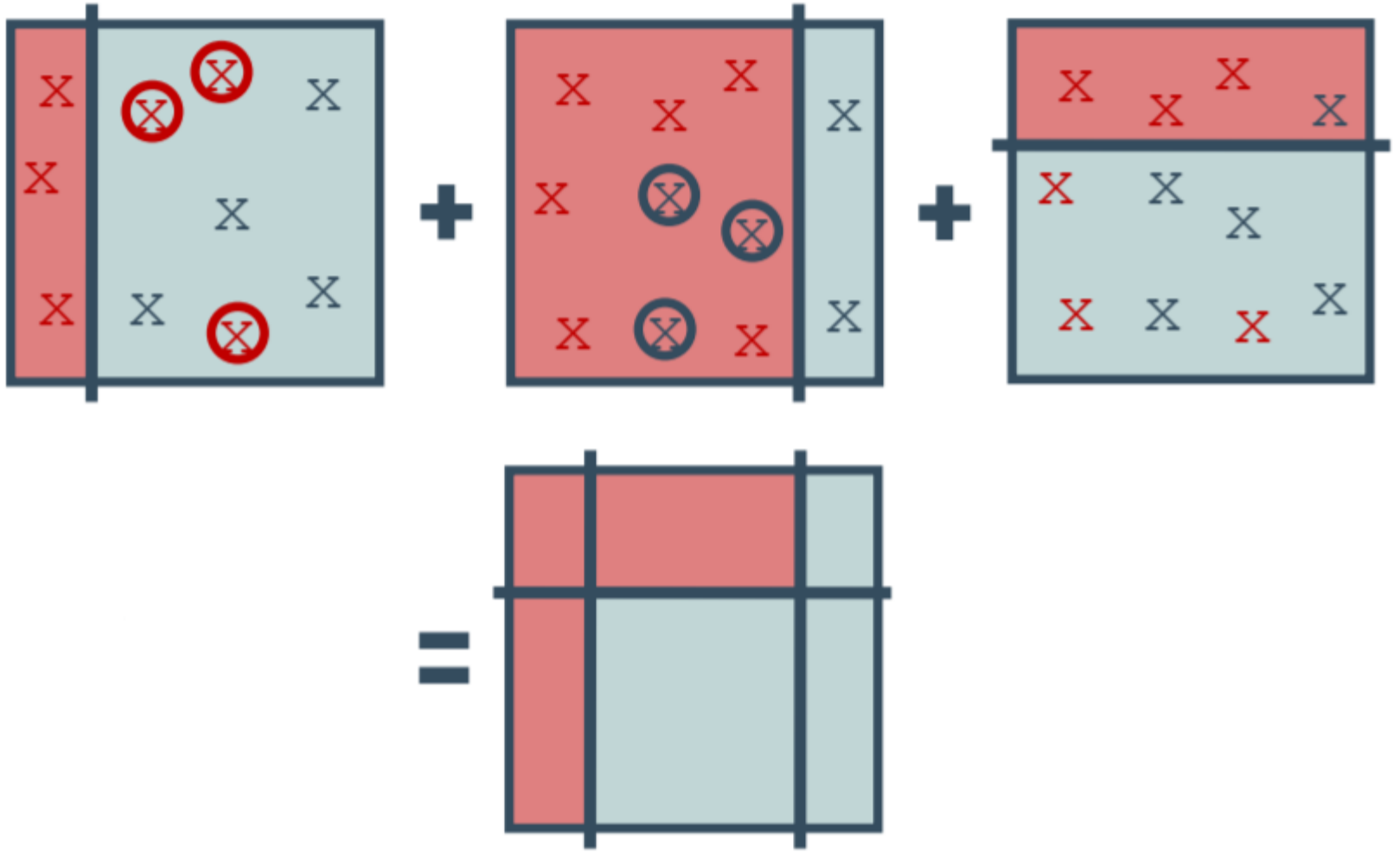


Обзор бустинга

Подгоните (обучите) новое одноуровневое дерево решений (decision stump) к текущим остаткам



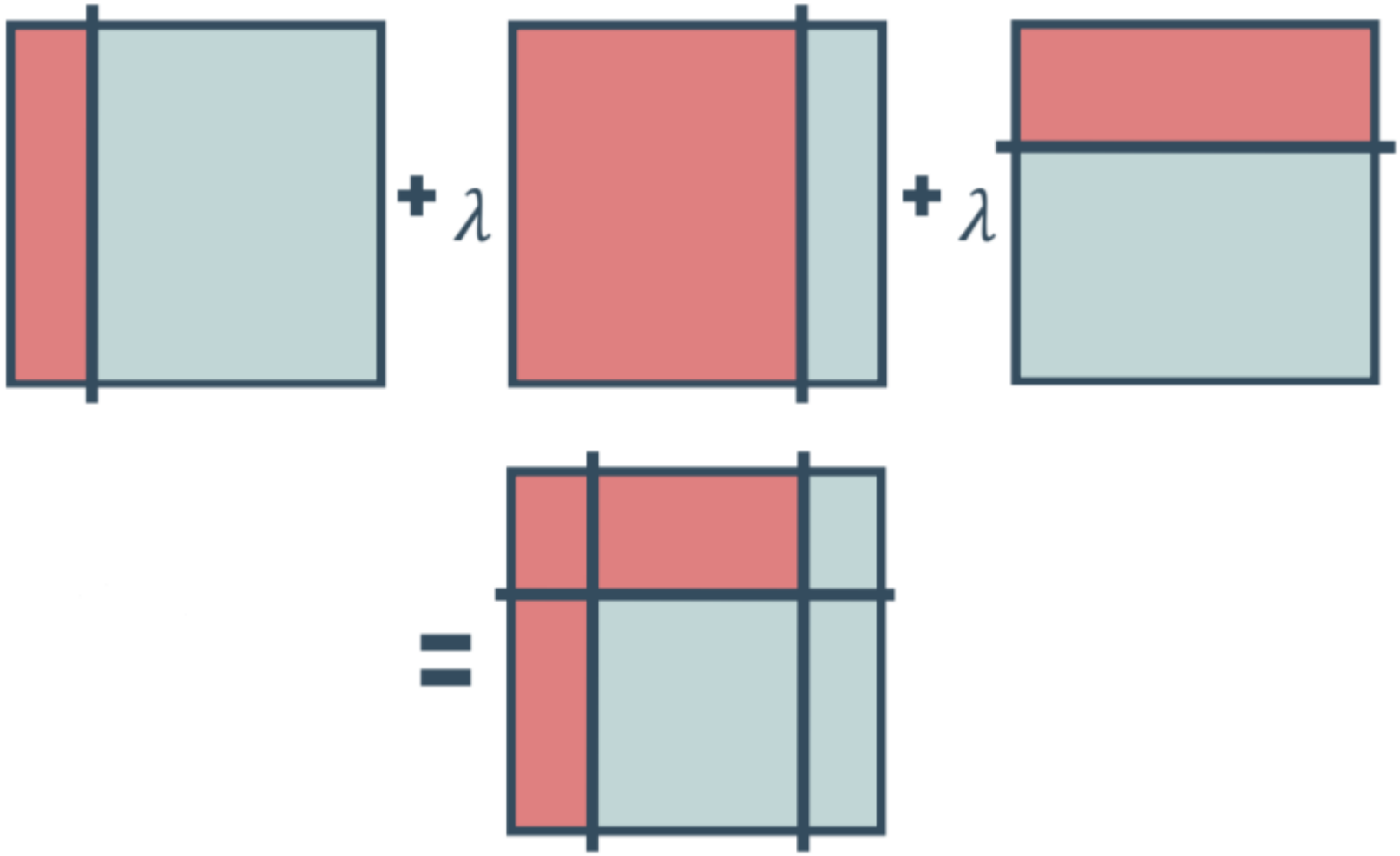
Обзор бустинга



Объединить в единый классификатор

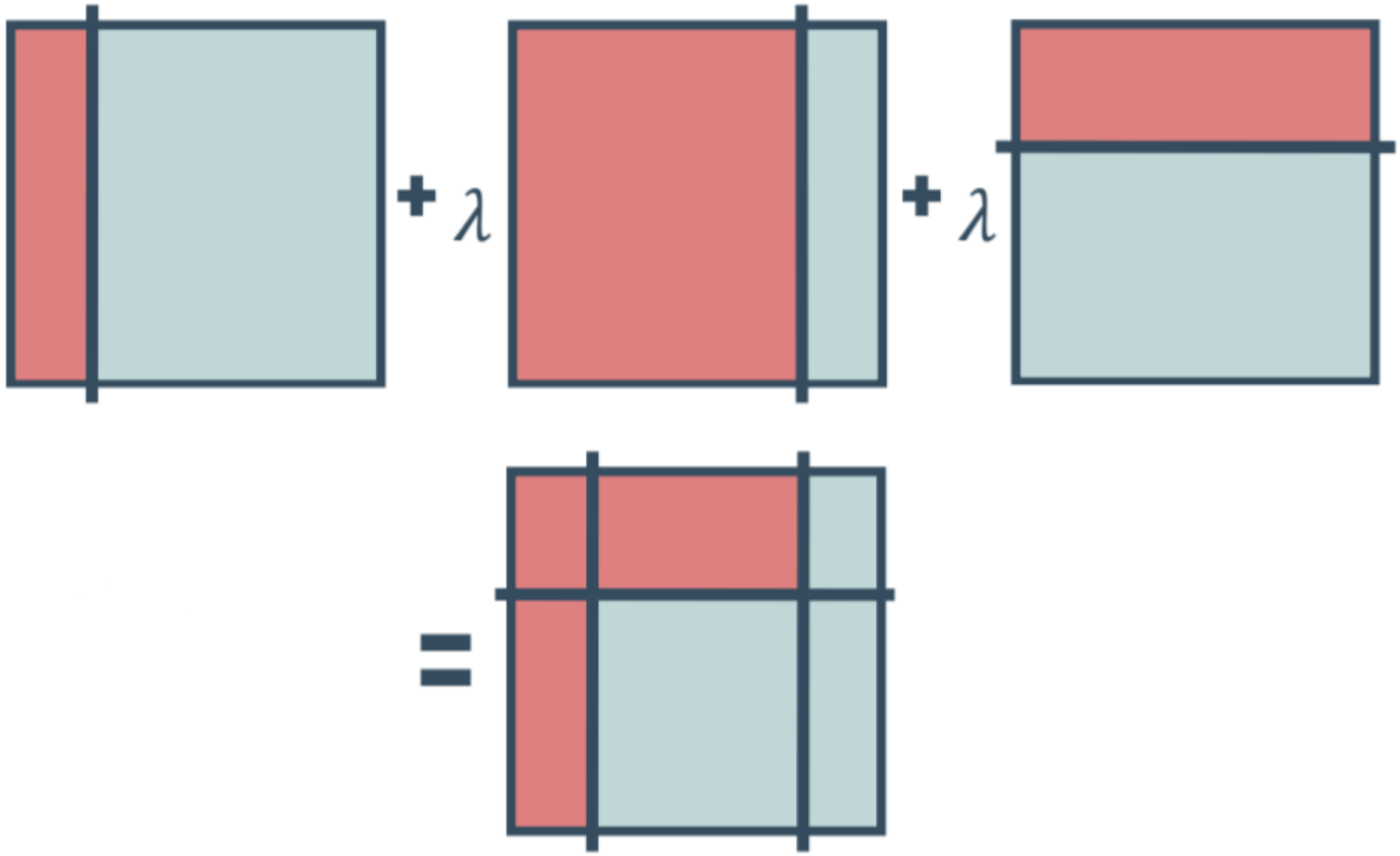
Обзор бустинга

Результатом является взвешенная сумма всех классификаторов

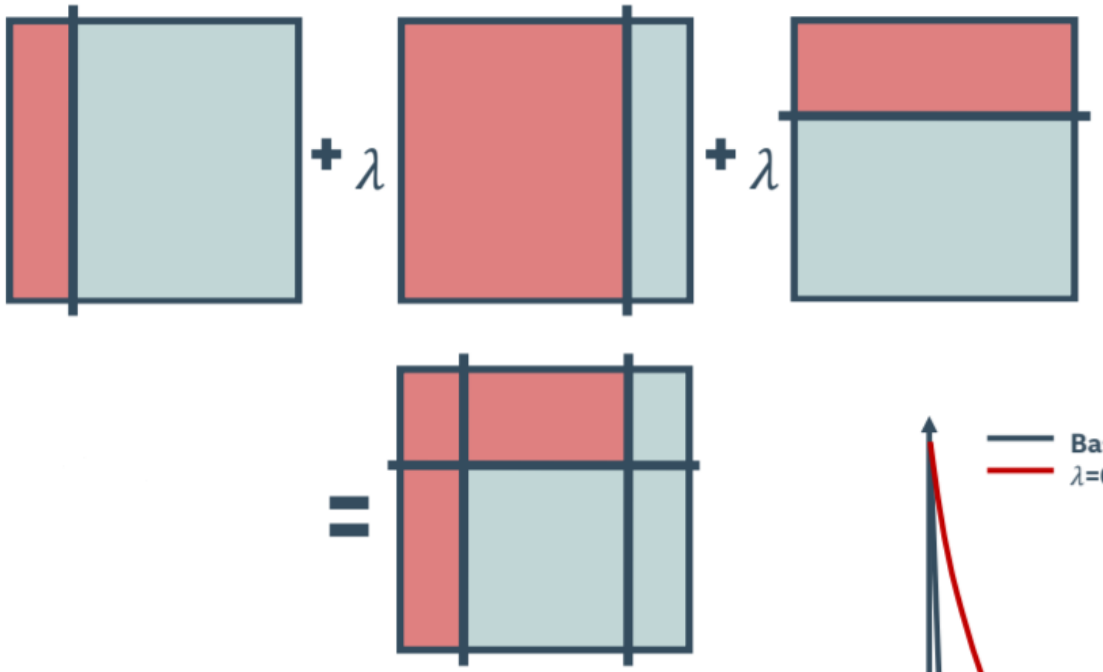


Обзор бустинга

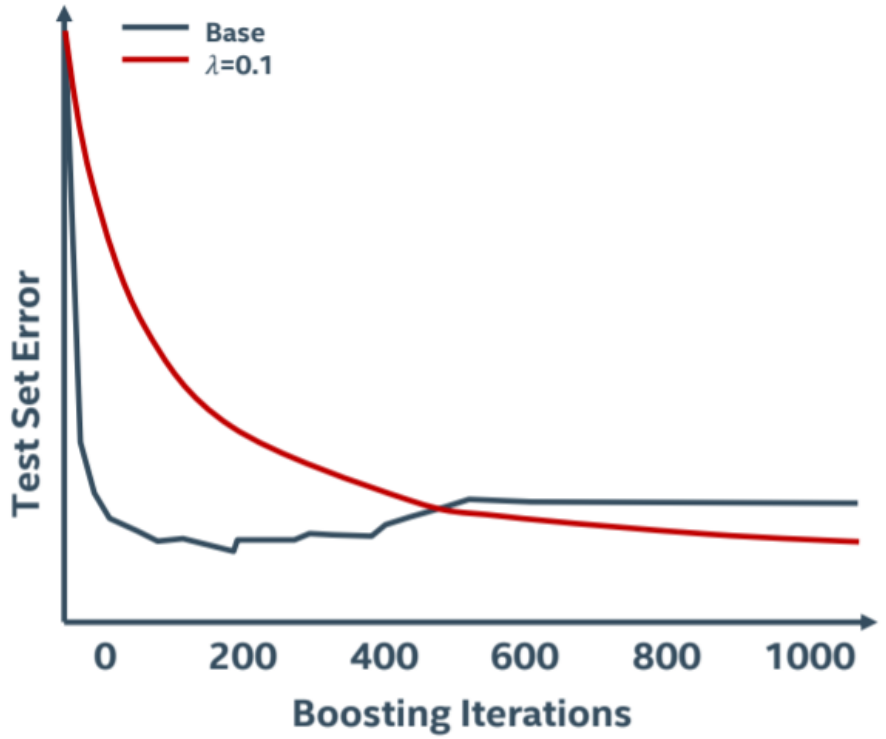
Последовательные классификаторы взвешиваются с коэффициентами скорости обучения (λ)



Обзор бустинга



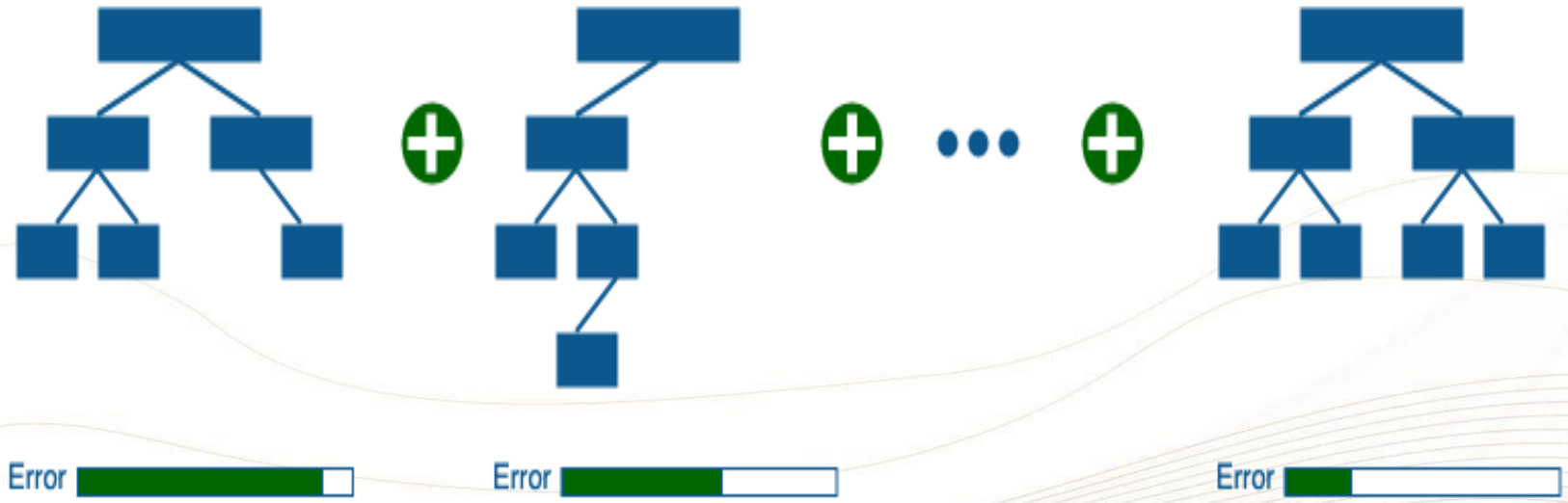
Использование скорости обучения λ <1 помогает предотвратить переобучение модели (регуляризация)



Идея градиентного бустинга

Градиентный бустинг — это техника машинного обучения для задач классификации и регрессии, которая строит модель предсказания в форме ансамбля слабых предсказывающих моделей, обычно деревьев решений. Обучение ансамбля проводится последовательно в отличие, например от бэггинга.

Общая идея метода - аддитивное обучение . На каждой итерации новое дерево обучается на значениях градиентов остатков между целевыми значениями и текущими прогнозируемыми значениями, а затем алгоритм проводит градиентный спуск на основе изученных градиентов.



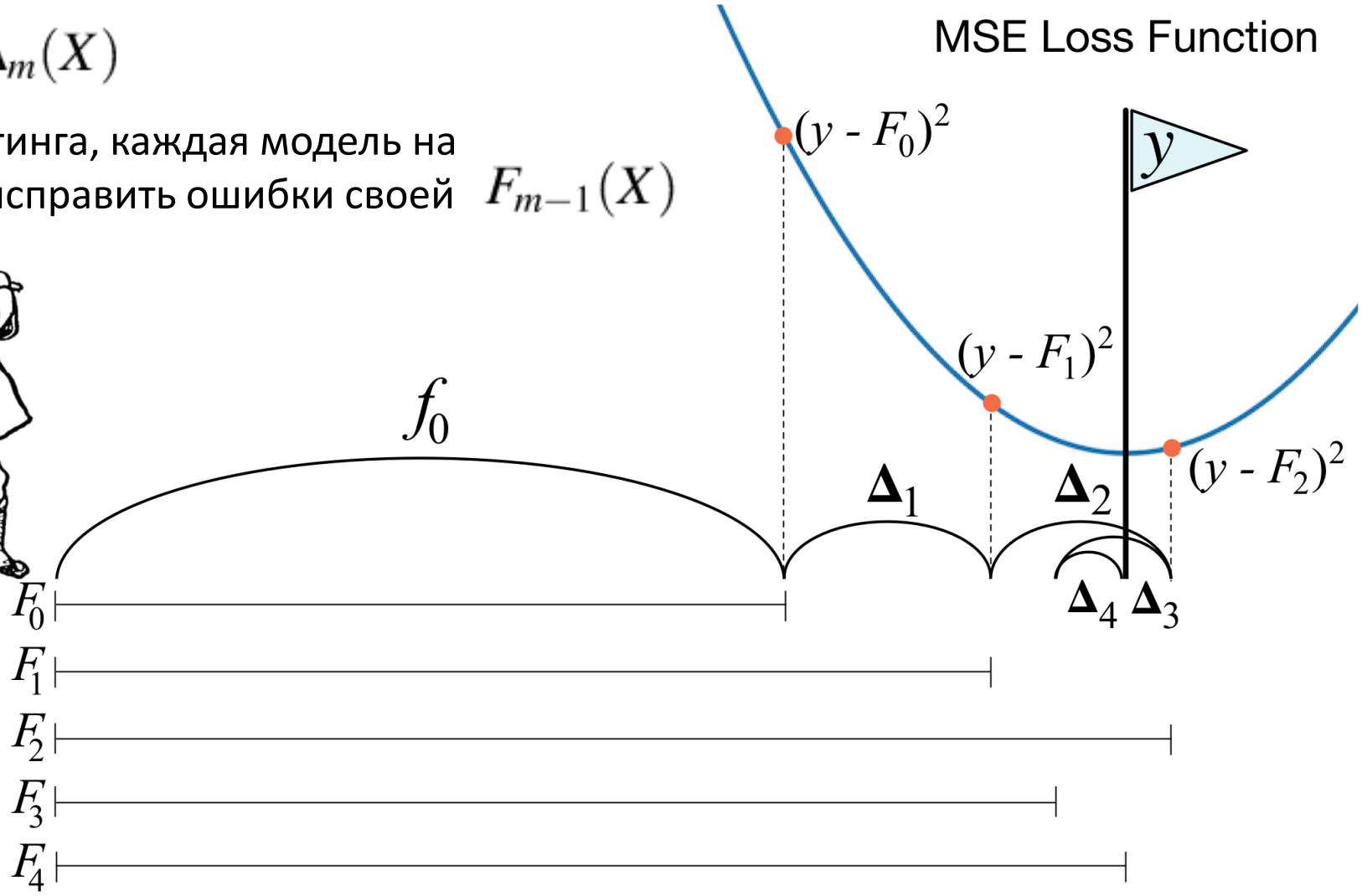
- Градиентный бустинг является одним из наиболее эффективных способов построения ансамблевых моделей. Комбинация градиентного бустинга с деревьями решений обеспечивает лучшие результаты во многих приложениях со структурированными данными
- Градиентный бустинг является техникой машинного обучения , которая работает путем постепенного обучения более сложным моделям для максимизации точности прогнозов. Градиентный бустинг особенно полезен для прогностических моделей, которые анализируют упорядоченные (непрерывные) данные и категориальные данные. Прогноз кредитного рейтинга, который содержит числовые характеристики (возраст и зарплата) и категориальные характеристики (род занятий), является одним из таких примеров
- Градиентный бустинг выигрывает при обучении на огромных наборах данных. Кроме того, техника эффективно ускоряется с помощью графических процессоров

Идея градиентного бустинга

Градиентный бустинг выполняет добавление слабых моделей Δ_m к нашей аддитивной модели:

$$F_m(X) = F_{m-1}(X) + \eta \Delta_m(X)$$

Как и в других вариантах бустинга, каждая модель на шаге *m* $F_m(X)$ пытается исправить ошибки своей предшественницы $F_{m-1}(X)$



Идея градиентного бустинга

Ключевое понимание

Остаток $y - y^{/}$ ($\mathbf{y} - F_{m-1}(X)$) - это не просто величина разницы, это вектор направления. Более того, вектор указывает направление наилучшего приближения и, следовательно, меньших потерь (ошибок). Это говорит о том, что вектор направления также является отрицательным градиентом функции потерь.

Приближение к вектору остатка в GBM выполняет отрицательный градиент функции ошибок с помощью градиентного спуска.

Когда L является среднеквадратичной функцией потерь (MSE), градиент L является остаточным вектором, и оптимизатор градиентного спуска должен находить этот остаток, что в точности и делает градиентный бустинг.

Добавляемая модель обучается и является приближением к вектору остатка $\mathbf{y} - F_{m-1}(X)$.

Таким образом, добавление еще одной слабой модели Δ_m в GBM фактически добавляет отрицательный градиент функции потерь, чтобы получить следующее приближение:

Градиентный спуск **Градиентный бустинг**

$$\mathbf{x}_t = \mathbf{x}_{t-1} - \eta \nabla f(\mathbf{x}_{t-1}) \qquad \hat{\mathbf{y}}_m = \hat{\mathbf{y}}_{m-1} + \eta(-\nabla L(\mathbf{y}, \hat{\mathbf{y}}_{m-1}))$$

Алгоритм градиентного бустинга

1. Создайте дерево решений на обучающей выборке $\{X, y\}$ и сделайте прогнозы $\hat{y_1}$.
2. Рассчитайте остатки $e_1 = y - \hat{y_1}$.
3. Добавьте новое дерево решений и обучите его на остатках e_1 в качестве целевой переменной. Обучающая выборка $\{X, e_1\}$. Сделайте прогнозы $\hat{e_1}$.
4. Исправьте предыдущие прогнозы, добавив предсказанные остатки $\hat{e_1}$, то есть $\hat{y_2} = \hat{e_1} + \hat{y_1}$.
5. Подогнать (обучить) следующее дерево решений по остаткам $e_2 = y - \hat{y_2}$.
6. Затем повторите шаги (2) - (5) и используйте критерии остановки (оценки по тестовым данным), когда модель начинает переобучаться или останавливаться, если сумма невязок (ошибка) перестанет уменьшать.

Отличия градиентного бустинга и стандартного бустинга

В чем разница между стандартным бустингом и градиентным бустингом?

В последнем случае мы используем градиентный спуск, чтобы минимизировать выбранную функцию потерь, и вместо остатков мы используем более общее определение псевдо-остатков , рассчитываемых как градиент функции потерь $e = -\frac{\partial L}{\partial \hat{y}}$ (фактически для MSE, градиент точно (с точностью до константы) равен $y - \hat{y}$, поэтому для этого случая псевдо-остаток = остаток).

Пример градиентного бустинга

Будем использовать простой пример, чтобы понять алгоритм GBM. Мы должны предсказать возраст группы людей, используя следующие данные:

ID	Married	Gender	Current City	Monthly Income	Age (target)
1	Y	M	A	51,000	35
2	N	F	B	25,000	24
3	Y	M	A	74,000	38
4	N	F	A	29,000	30
5	N	F	B	37,000	33

Пример градиентного бустинга

1. Предполагается, что средний возраст является прогнозируемым значением для всех наблюдений в наборе данных.
2. Ошибки $y - y^{/}$ рассчитываются с использованием просто среднего прогноза и фактических значений возраста.

ID	Married	Gender	Current City	Monthly Income	Age (target)	Mean Age (prediction 1)	Residual 1
1	Y	M	A	51,000	35	32	3
2	N	F	B	25,000	24	32	-8
3	Y	M	A	74,000	38	32	6
4	N	F	A	29,000	30	32	-2
5	N	F	B	37,000	33	32	1

Пример градиентного бустинга

3. **Обучаем добавляемое дерево решений с использованием вычисленных выше ошибок в качестве целевой переменной, представляющих собой значения градиента функции потерь в точках обучающей выборки.**
4. Прогнозы этой модели сочетаются с прогнозами 1.

ID	Age (target)	Mean Age (prediction 1)	Residual 1 (new target)	Prediction 2	Combine (mean+pred2)
1	35	32	3	3	35
2	24	32	-8	-5	27
3	38	32	6	3	35
4	30	32	-2	-5	27
5	33	32	1	3	35

Пример градиентного бустинга

5. Это значение, рассчитанное выше, является новым прогнозом.
6. Новые ошибки рассчитываются с использованием этого прогнозируемого значения и фактического значения.

ID	Age (target)	Mean Age (prediction 1)	Residual 1 (new target)	Prediction 2	Combine (mean+pred2)	Residual 2 (latest target)
1	35	32	3	3	35	0
2	24	32	-8	-5	27	-3
3	38	32	6	3	35	-3
4	30	32	-2	-5	27	3
5	33	32	1	3	35	-2

7. Шаги 2-6 повторяются до тех пор, пока не будет достигнуто максимальное количество итераций (или функция ошибок не перестанет изменяться).

AdaBoostClassifier: синтаксис

Импортируем класс, содержащий метод классификации

```
from sklearn.ensemble import AdaBoostClassifier
```

Создадим экземпляр класса

```
ABC = AdaBoostClassifier(base_estimator=DecisionTreeClassifier(),
          learning_rate=0.1, n_estimators=200)
```

Обучим модель на обучающем выборке, а затем прогнозируем ожидаемое значение на тестовой выборке

```
   ABC = ABC.fit (X_train, y_train)
y_predict = ABC.predict(X_test)
```

Используйте AdaBoostRegressor для регрессии.
Настройте параметры перекрестной проверки

GradientBoostingClassifier: синтаксис

Импортируем класс, содержащий метод классификации

```
from sklearn.ensemble import GradientBoostingClassifier
```

Создадим экземпляр класса

```
GBC = GradientBoostingClassifier(learning_rate=0.1,
        max_features=1, subsample=0.5,
n_estimators=200)
```

Обучим модель на обучающем выборке, а затем прогнозируем ожидаемое значение на тестовой выборке

```
    GBC = GBC.fit (X_train, y_train)
y_predict = GBC.predict(X_test)
```

Используйте GradientBoostingRegressor для регрессии.
Настройте параметры перекрестной проверки

Реализации градиентного бустинга

Существует несколько реализаций градиентного бустинга в бесплатных библиотеках машинного обучения, включая:

- scikit-Learn GradientBoosting
- XGBoost
- LightGBM
- CatBoost
- h2o.ai
- TensorFlow 2.0

Сильные и слабые стороны градиентного бустинга

Сильные стороны:

- на практике быстрее, чем случайный лес
- такие реализации, как LightGBM, могут обрабатывать большие объемы данных и требуют меньшего объема памяти
- приобрел чрезвычайную популярность (LightGBM, XGBoost, CatBoost, h2o.ai) благодаря повышенной точности решения задач классификации и регрессии
- Подобно случайному лесу, можно повысить интерпретируемость модели, визуализируя важность функций и отдельные деревья решений

Слабые стороны:

- чувствителен к переобучению, поэтому очень важно правильно настроить гиперпараметры