# Ансамблирование моделей

# Егор Шишковец

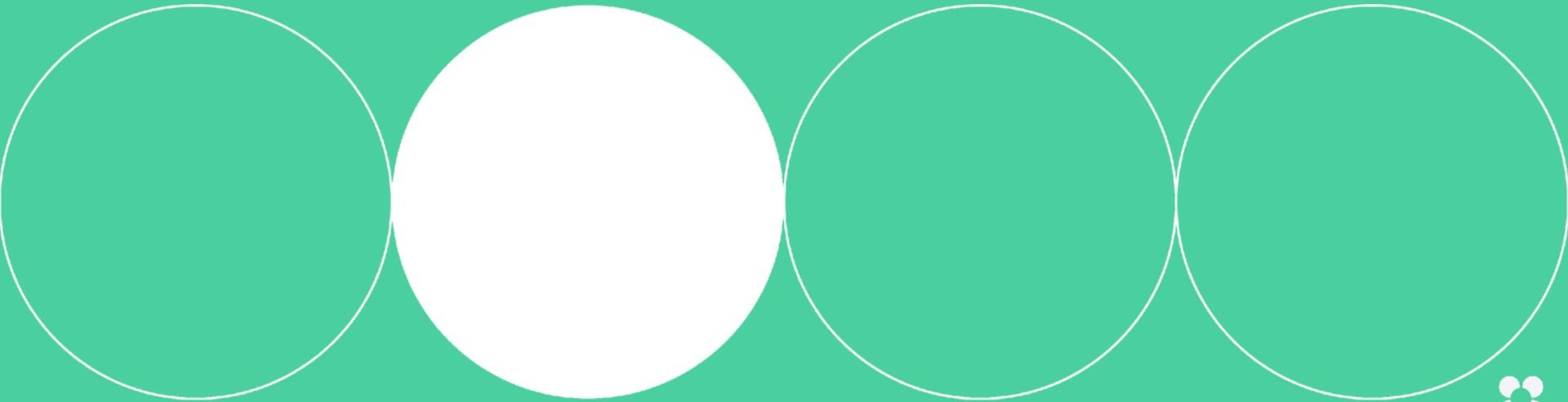
О спикере:

- Team Lead DS @ Честный Знак
- Большой опыт разработки предиктивных моделей в сфере клиентской аналитики
- Автор и разработчик open-source фреймворка АБ-тестирования ABacus

# План занятия

1. Введение
2. Ансамбли
3. Используемые концепции построения
4. Бэггинги, случайные леса, бустинги, стэкинги

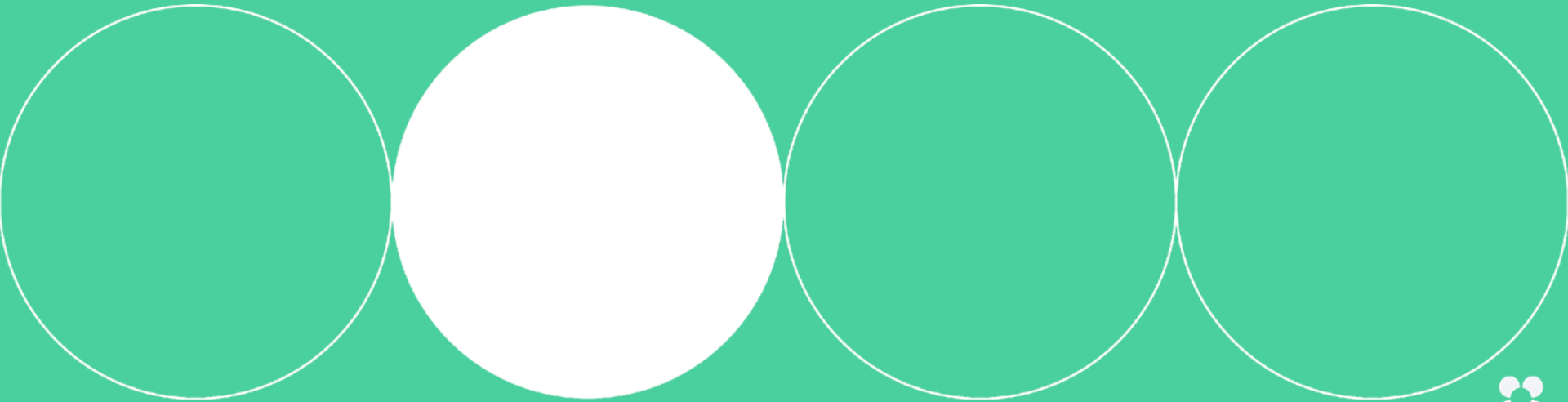
# Введение



Фрэнсис Гальтон “Мудрость толпы”

Фрэнсис Гальтон в 1906 году посетил рынок, где проводилась лотерея для крестьян.
Их собралось около 800 человек и они пытались угадать вес быка, который стоял перед ними. **Бык весил 1198 фунтов**. Ни один крестьянин не угадал точный вес быка, но если посчитать среднее от их предсказаний, то получим **1197 фунтов**.
Эту идею уменьшения ошибки применили и в машинном обучении

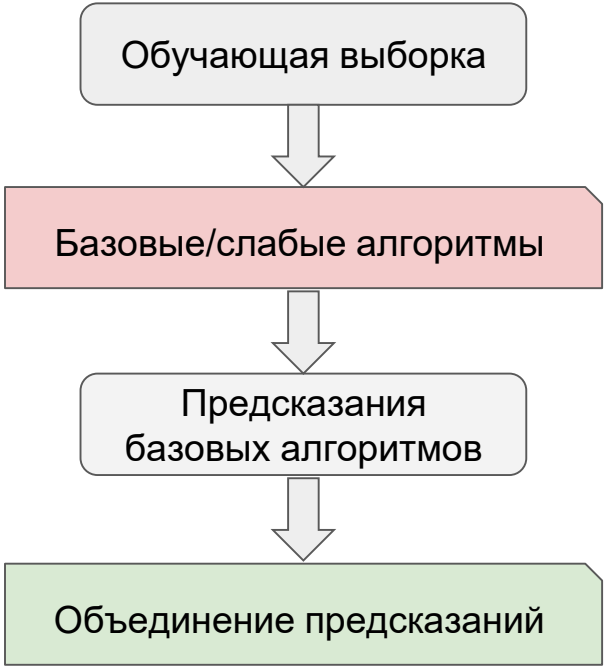
# Ансамбли



# Ансамбли алогритмов (стекинг)

Основная идея:

1. Использовать множество алгоритмов (называют их базовыми/слабыми)
2. Объединить их ответы мета-моделью или более простым методом



# Концепция разброса - смещения

**Ошибка на новых данных** = Шум + Смещение + Разброс, где

ШУМ - ошибка лучшей модели a(x)

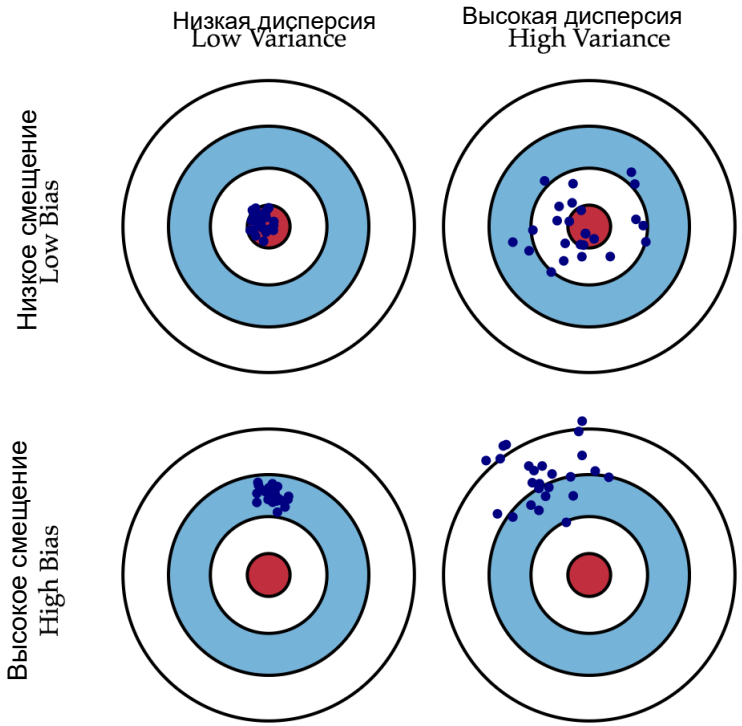
СМЕЩЕНИЕ (отклонение, bias) - отклонение усредненных ответов наших моделей от ответов лучшей модели a(x)

РАЗБРОС (дисперсия, variance) - дисперсия ответов наших моделей

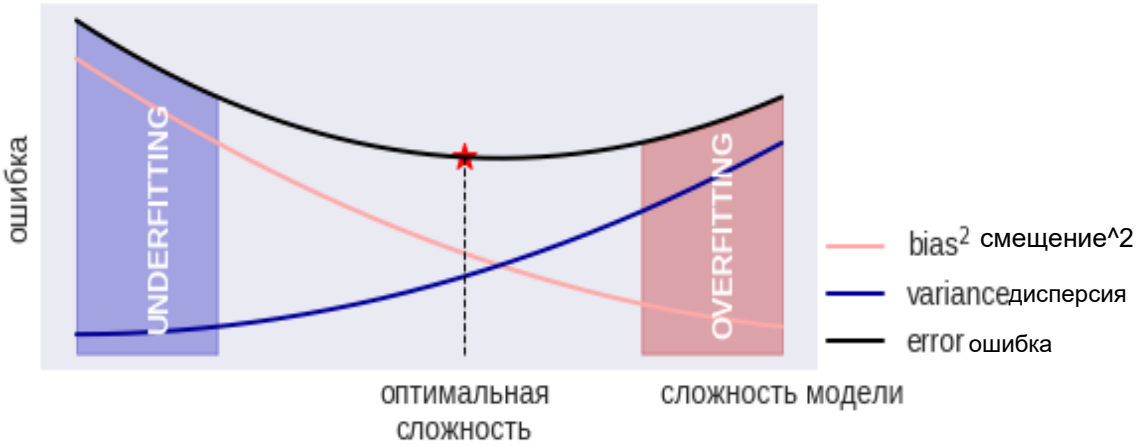
# Концепция разброса - смещения

**Смещением (bias)** – матожидание разности между истинным ответом и выданным алгоритмом, характеризует способность модели алгоритмов настраиваться на целевую зависимость.

**Разброс (variance)** - дисперсию ответов алгоритмов характеризует разнообразие алгоритмов (из-за случайности обучающей выборки, в том числе шума, и стохастической природы настройки)



# Концепция недообучения - переобучения

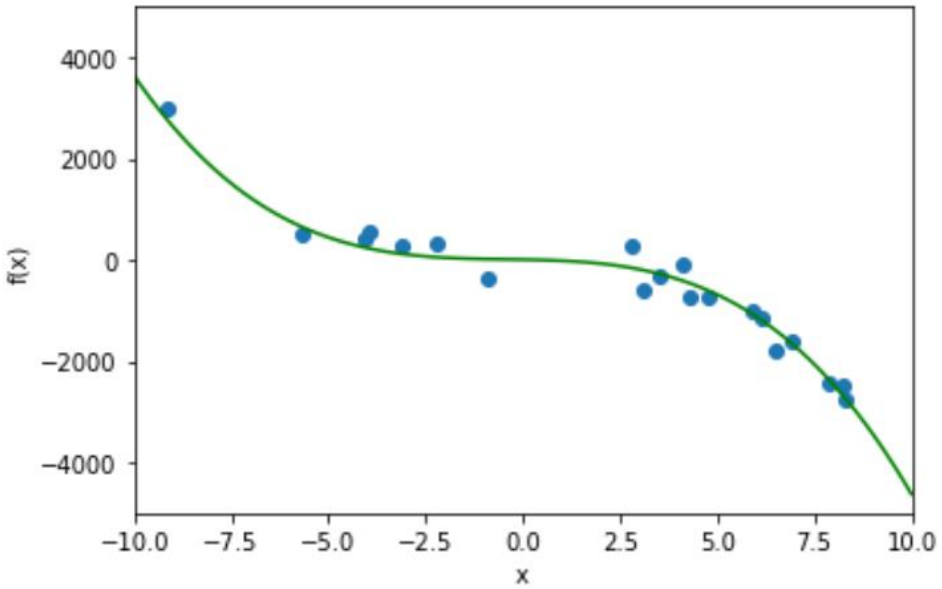



- Недообученые модели, в силу простоты, не могут описать целевую зависимость и имеют большое смещение (bias)
- Переобученные модели слишком хорошо запомнили обучающую выборку и имеют большой разброс (variance)

# Концепция разброса - смещения

Из одного распределения нагенерируем 10 разных выборок

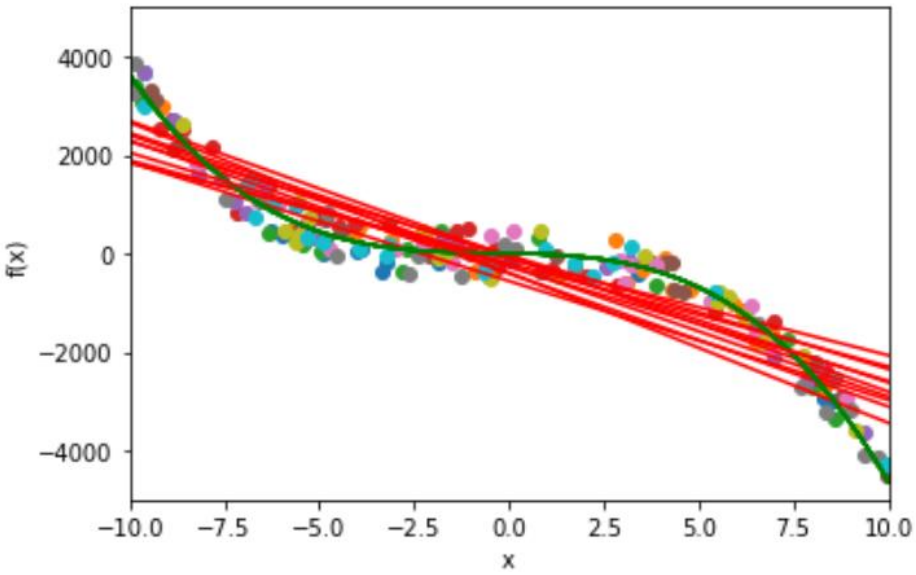
(На изображении представлена только одна)



# Концепция разброса - смещения

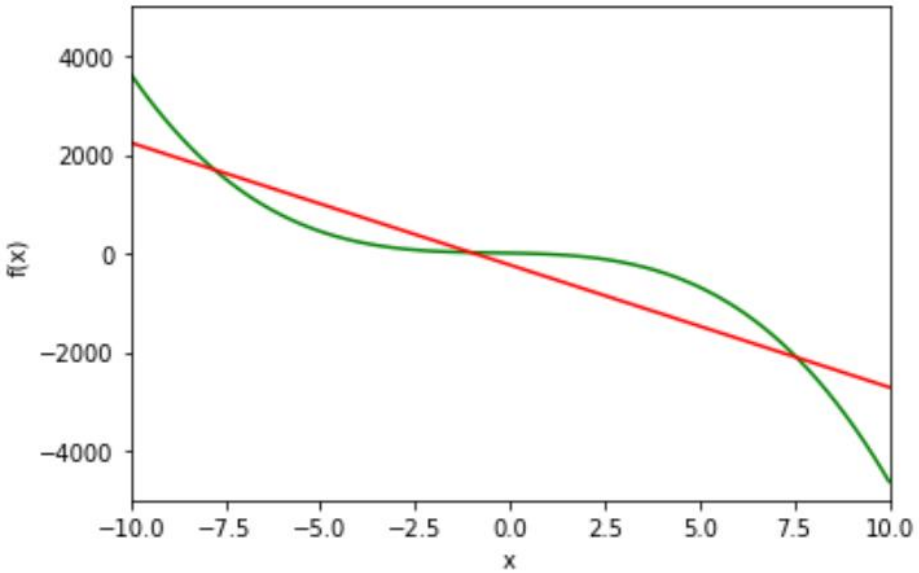
Обучим **линейную регрессию** на 10 выборках

Получаем **низкий разброс**, каждая модель по одиночке не способна запомнить обучающую выборку - **высокое смещение**



# Концепция разброса - смещения

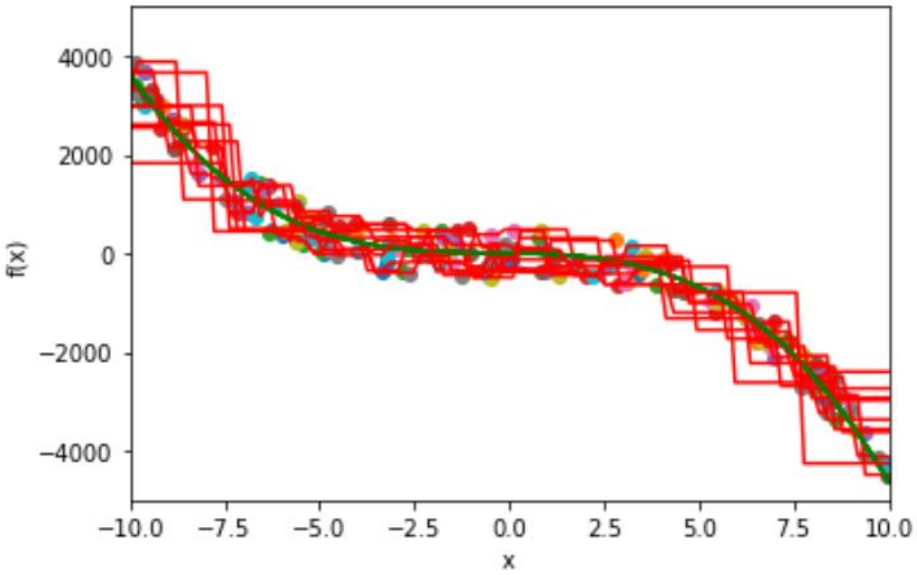
Если усредняем предсказания **линейной регрессии**, то получаем очередную линию, которая не очень похожа на истинную



# Концепция разброса - смещения

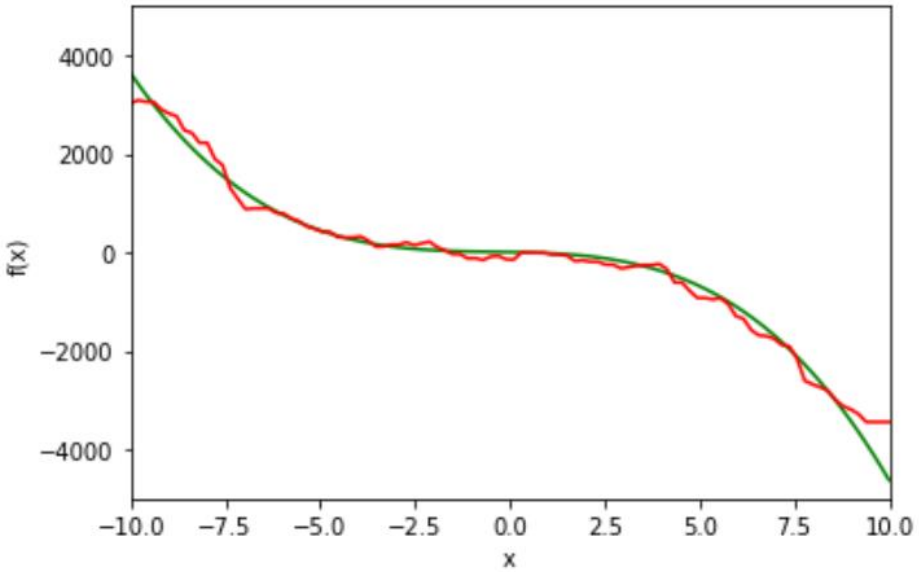
Обучим **деревья решений** на 10 выборках

Получаем **высокий разброс**, но каждая модель способна запомнить обучающую выборку - **низкое смещение**



# Концепция разброса - смещения

Если усредняем предсказания **деревьев решений**, то получаем функцию похожую на истинную

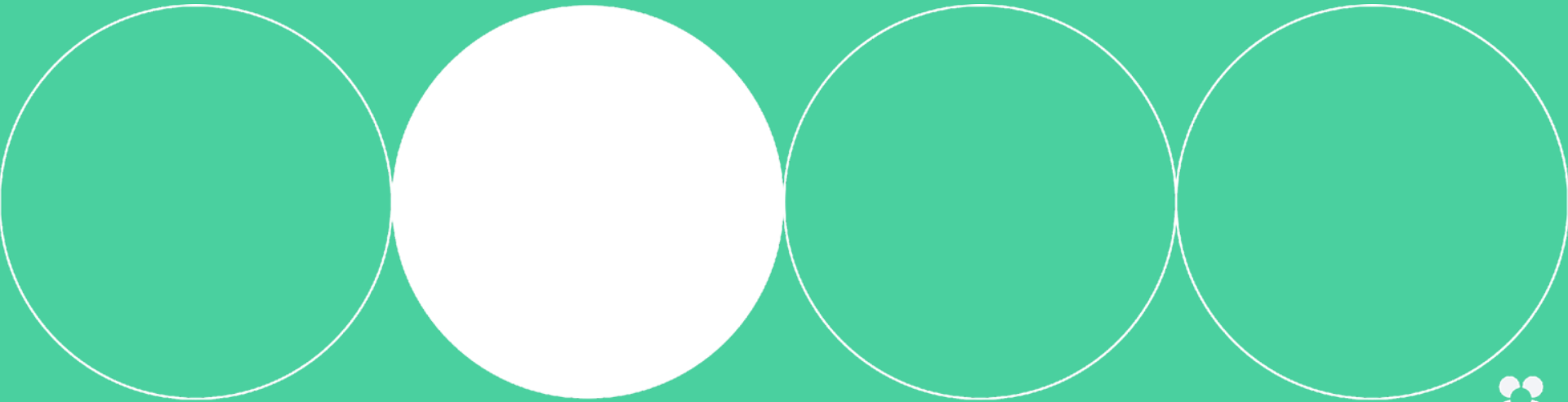


# Концепция разброса - смещения

**Усреднение алгоритмов:**

- Не меняется смещение
- Разброс = $\frac{\text{разброс базового алгоритма}}{N}$ + корреляция между базовыми алгоритмами

# Используемые концепции построения



# Методы ансамблирования

- **Простое голосование** – набор различных моделей, в котором голосует каждая и решает большинство
- **Взвешенное голосование** – набор различных моделей, в котором голосует каждая с заранее определенным весом и решает большинство
- **Смесь экспертов** – набор различных моделей, в котором голосует каждая с весом, зависящем от данных по которым требуется сделать предсказание

Источник:
http://www.machinelearning.ru/wiki/images/5/56/Guschin2015Stacking.pdf
https://hackernoon.com/how-to-develop-a-robust-algorithm-c38e08f32201

# Усреднение

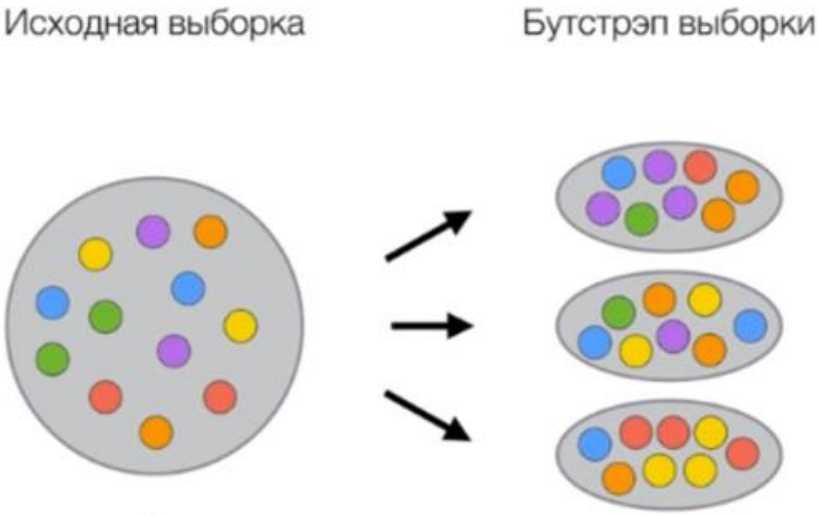
Для регрессии:

- Усредненние
- Медиана
- Усреднение с весами

Для классификации:

- Голосование
- Голосование с весами

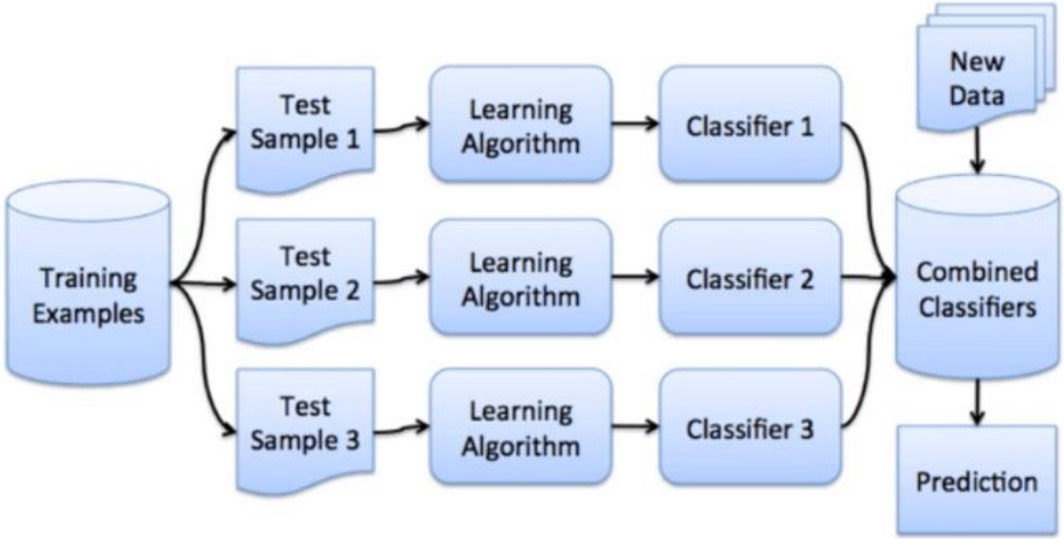
# Бутстрэп



# Бэггинг (Bagging)

## Bagging - bootstrap aggregating

Используют базовые алгоритмы и обучают параллельно на случайном подмножестве обучающей выборки



# Бэггинг

From sklearn.ensemble import BaggingRegressor

**base estimator** object or None - модель регрессии из sklearn (по умолчанию деревья решений)
**n_estimators** - количество моделей
**max_samples** int or float, optional (default=1.0) - количество сэмплов для обучения
**max features** int or float, optional (default=1.0) - количество признаков для обучения

# Случайные подпространства

Предположим, что в наборе данных есть один очень сильный признак.

При использовании беггинга большая часть деревьев будет использовать этот признак в качестве первого, по которому производится деление, в результате чего образуется ансамбль деревьев, которые сильно коррелированы.

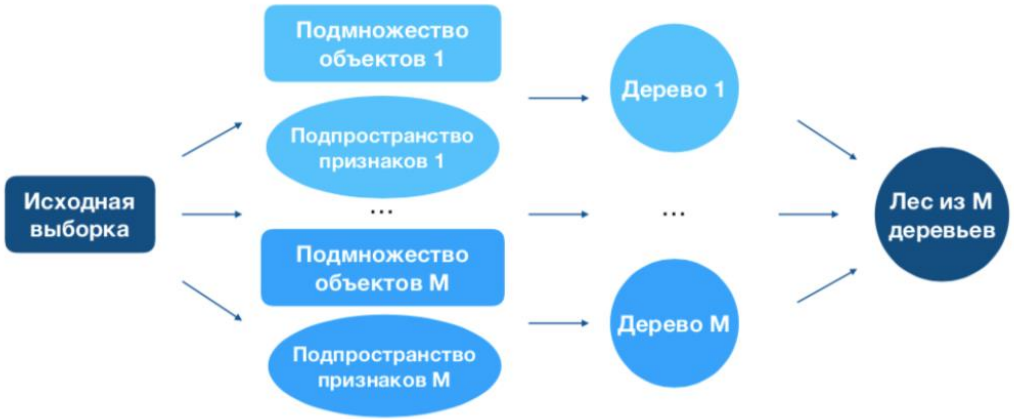
Усреднение высококоррелированных  величин не приводит к значительному уменьшению дисперсии (что является целью беггинга).

# Random Forest (Случайный лес)

Бэггинг + случайные подпространства = случайный лес

Случайный лес - вариация бэггинга над деревьями. Но при построении каждого дерева каждый раз, когда выбирается вопрос, признак выбирается из случайной выборки размера ***m*** из всех признаков.

Для классификации ***m*** обычно выбирается как квадратный корень из p. Для регрессии m обычно выбирается где-то между ***p/3*** и ***p***.



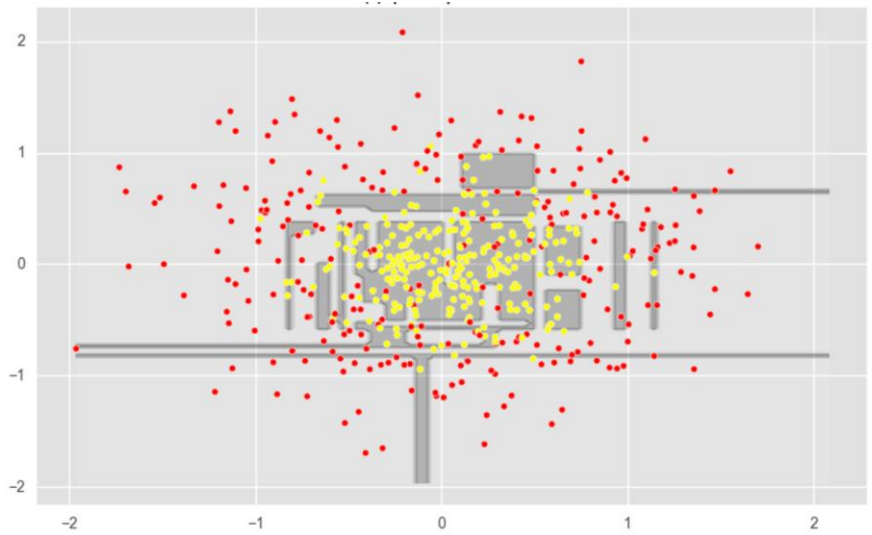
# Random Forest (Случайный лес)

From sklearn.ensemble import RandomForestRegressor

**criterion** - метод оценки ошибки (MSE по умолчанию)
**n_estimators** - количество моделей
**max depth** - максимальная глубина деревьев
**max_samples** int or float, optional (default=1.0) - количество сэмплов для обучения
**max features** int or float, optional (default=1.0) - количество признаков для обучения

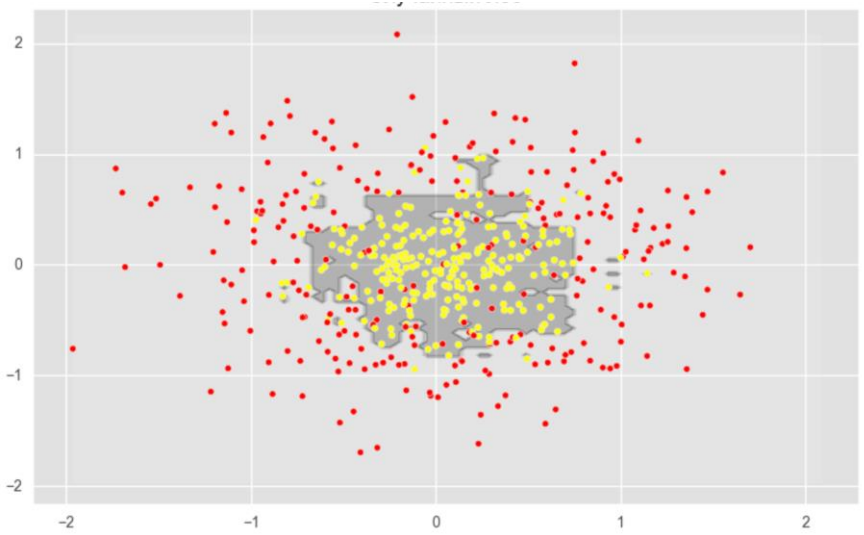
# Ансамбли строят лучшую разделяющую границу

Дерево решений




Граница – прямоугольники
Точность: 80%

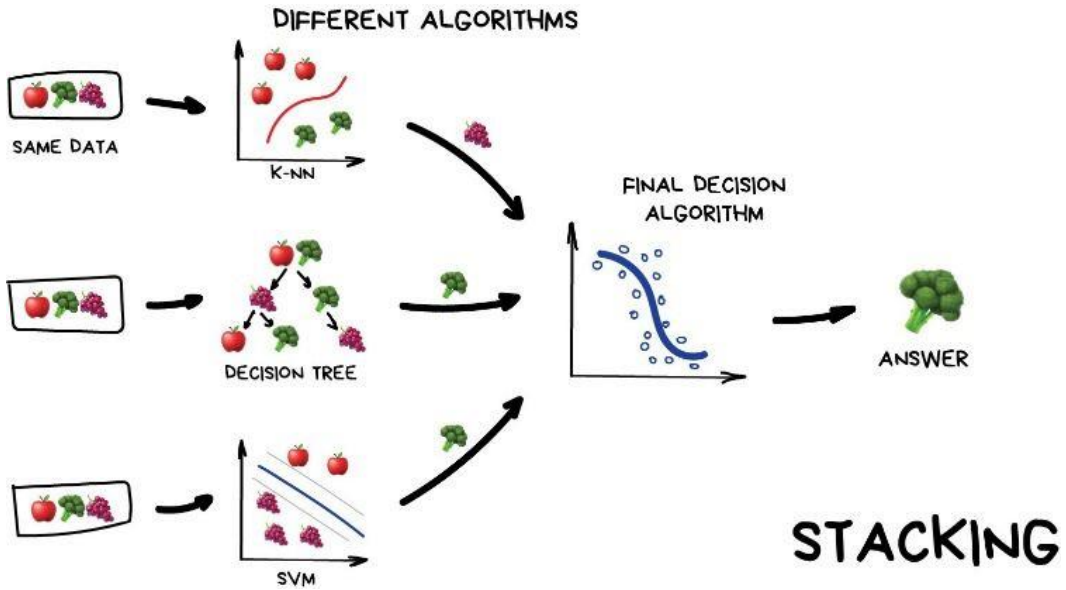
Случайный лес




Граница – гладкая кривая
Точность: 92%

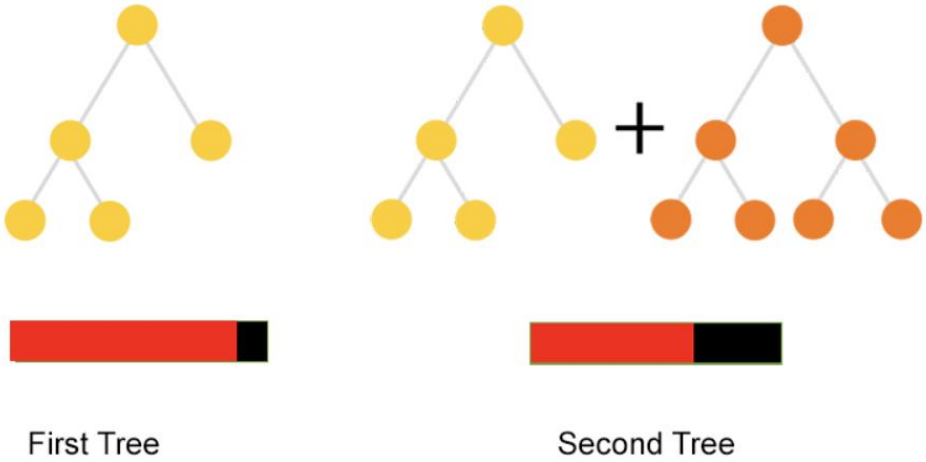
# Стекинг

Используют базовые алгоритмы, обучают их параллельно.
Для объединения результатов используется так называемая **метамодель** (или модель второго уровня) для предсказания конечного результата.



# Бустинг

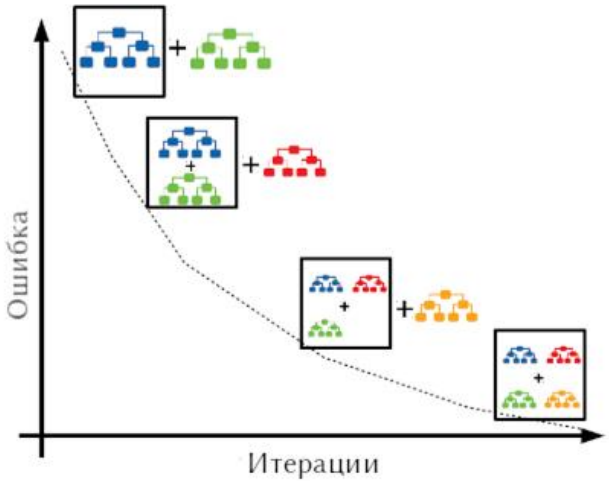
Бустинг (англ. boosting — улучшение) — это процедура последовательного построения композиции однородных базовых алгоритмов, когда каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов



# Бустинг

Градиентный бустинг

Сводит задачу к градиентному спуску и на каждой итерации подгоняет очередной базовый алгоритм в соответствии с антиградиентом ошибки текущей модели ансамбля



# Итоги

# Итоги

1. Введение
2. Ансамбли
3. Используемые концепции построения
4. Бэггинги, случайные леса, бустинги, стэкинги

# Рекомендации для ознакомления

1. Ансамбли в машинном обучении
2. Ансамблевые методы: бэггинг, бустинг и стекинг
3. Бэггинг и бутстрап + композиции в целом
4. Бэггинг и случайный лес
5. Бустинг

# Ансамблирование моделей