ТЕХНОСФЕРА

Занятие № 5

Ансамбли моделей

Евгений Некрасов

План занятия

1. Ансамбли моделей
2. Стэкинг
3. Бэггинг и бустинг
4. Random Forest
5. Gradient Boosting
6. Автоматический подбор параметров модели

О значимости ансамблей

Лучшие алгоритмы машинного обучения по точности:

- Градиентный бустинг для классических задач
- Искусственные нейронные сети для изображений, видео, звука

В соревнованиях kaggle всегда* побеждают ансамбли

Ансамбли моделей

Коллективное принятие решений как правило превосходит по качеству индивидуальное принятие решений

Простое голосование

Классификация: класс определяется большинством голосов или усреднением скоров

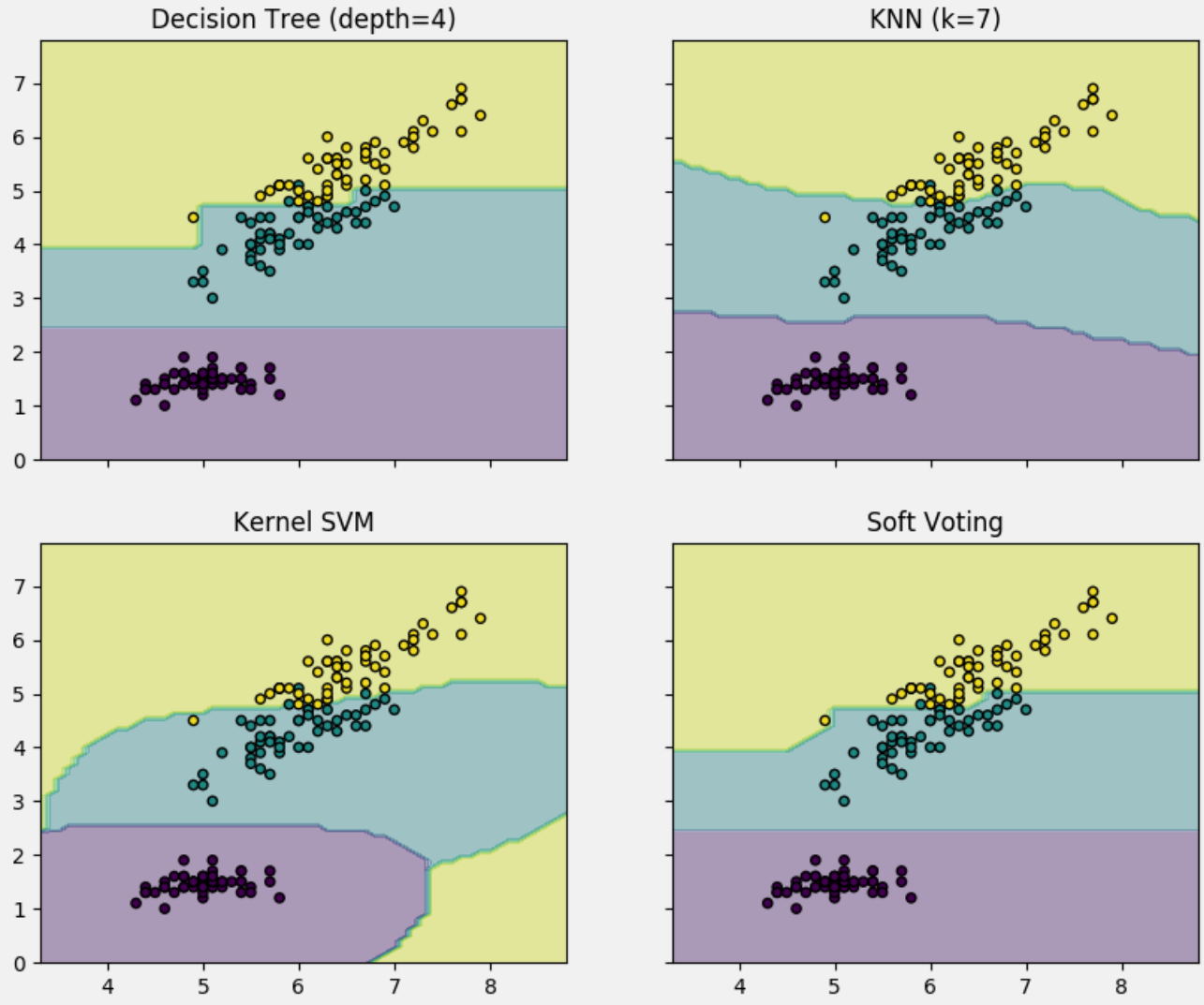
Регрессия: среднее значение

Взвешенное голосование

Классификация: класс определяется большинством голосов с учетом веса, или усреднением скоров с учетом веса

Регрессия: среднее взвешенное значение

Пример голосования



Недостатки голосования

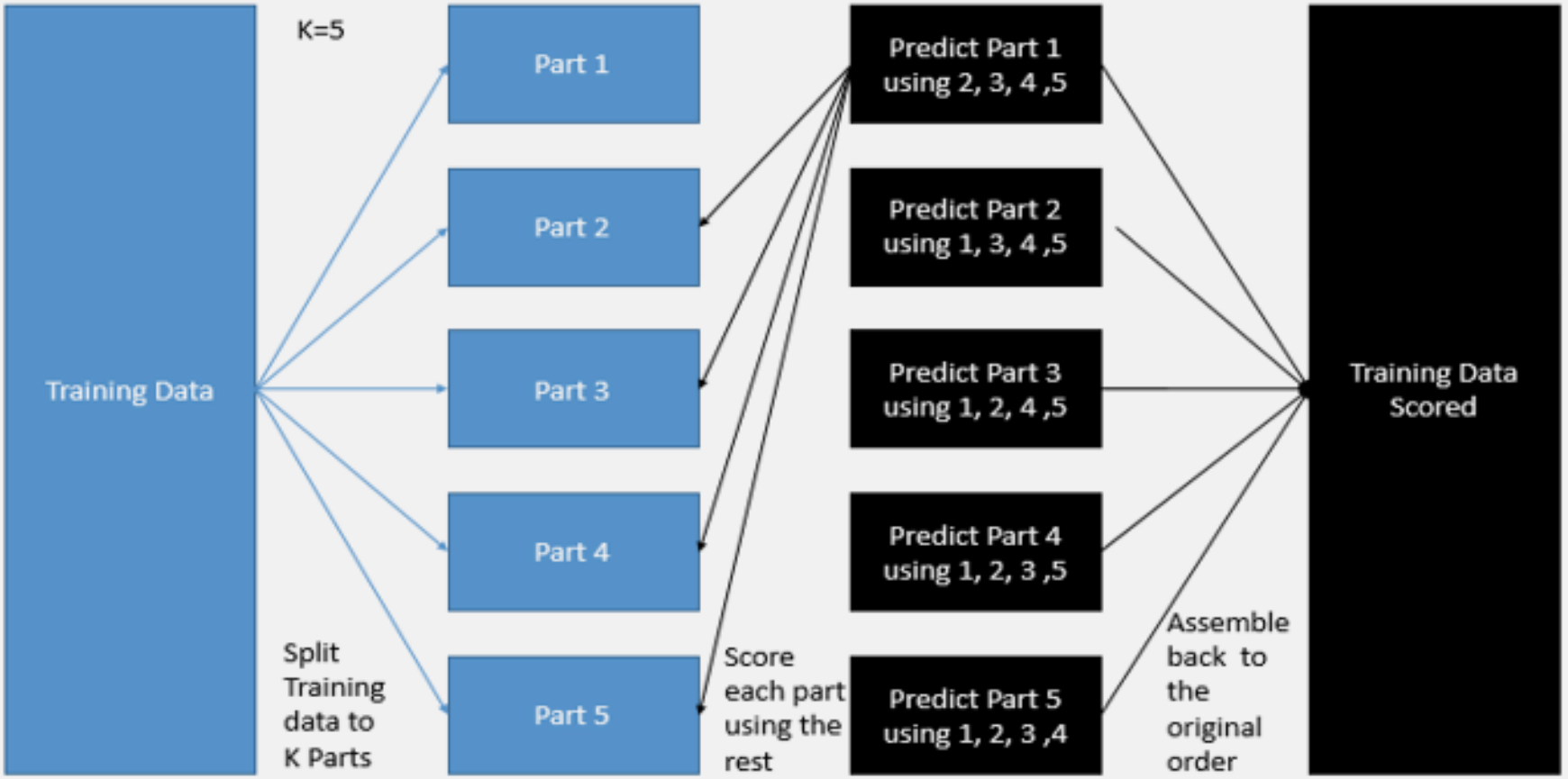
1. Голосование не учитывает особенностей сампла
2. Голосование не учитывает особенностей поведения отдельных моделей
3. Голосование по сути является простой моделью

Стэкинг

Идея:

Построим модель, которая будет предсказывать правильный ответ на основе предсказаний других моделей

Стэкинг - преобразование Train



Стэкинг - преобразование Test

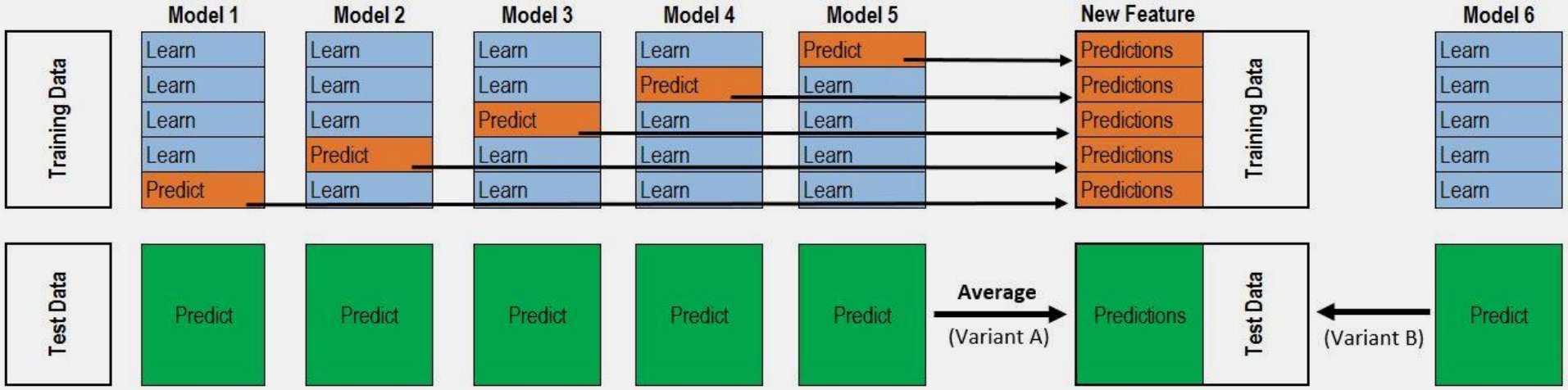
Вариант A:

1. Выполняем предсказание на Test каждой из K моделей кросс-валидационного предсказания
2. Усредняем K предсказаний

Вариант B:

1. Обучаем модель на полном датасете
2. Выполняем предсказание на Test

Стэкинг


Model 1
Model 2
Model 3
Model 4
Model 5
New Feature
Model 6
Training Data
Learn
Predict
Predictions
Training Data
Test Data
Average
(Variant A)
(Variant B)

Стэкинг

Кросс-валидационное предсказание будем называть **метапризнаком**.

Стэкинг можно делать как на наборе метапризнаков, так и на наборе метапризнаков + набор исходных признаков.

Стэкинг может быть многоуровневым.

Стэкинг реальный пример

Соревнование:

Homesite Quote Conversion

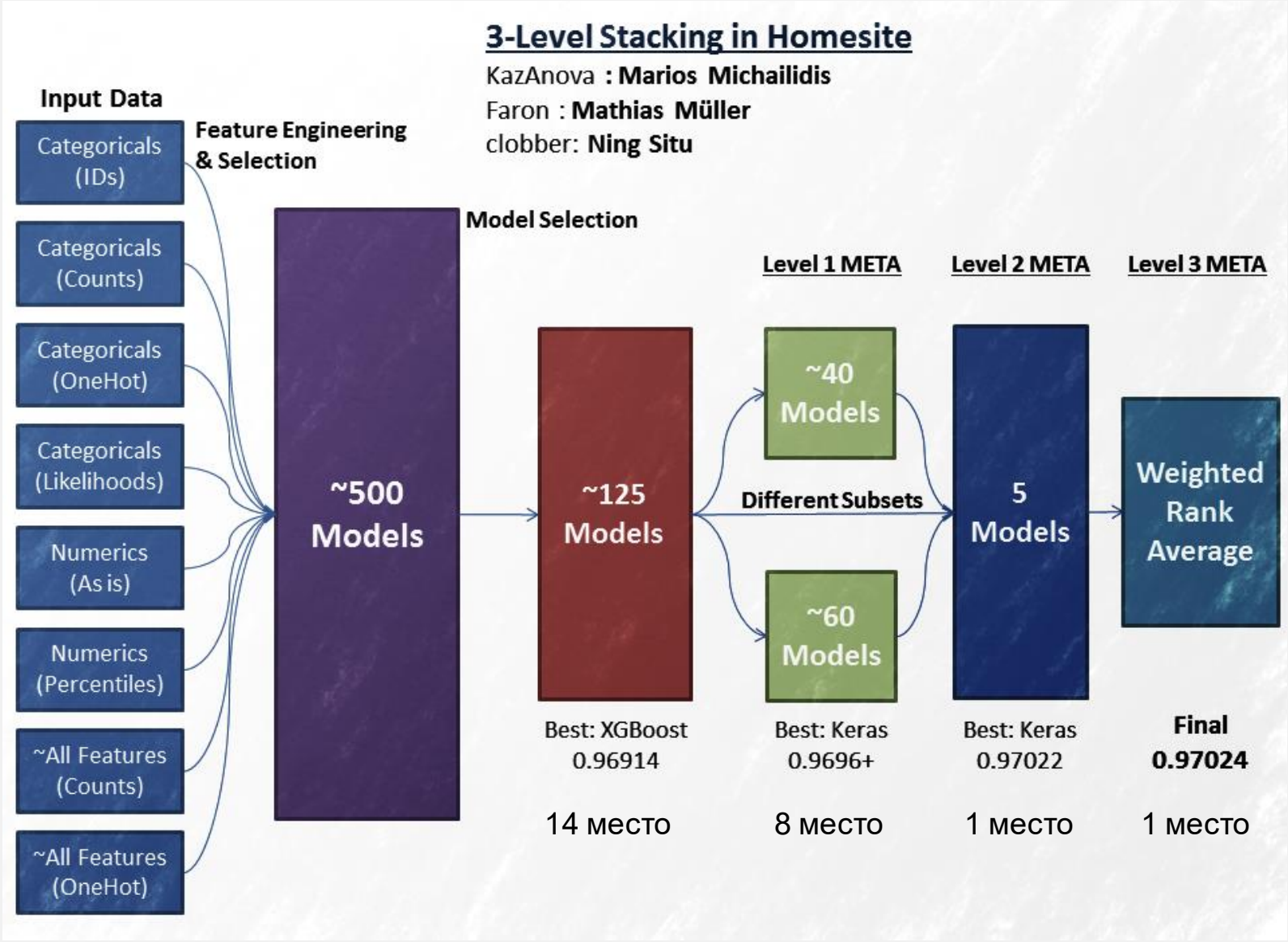
Задача:

Предсказать какие клиенты приобретут указанный страховой план на недвижимость

Пример:

Решение команды, занявшей 1 место

Стэкинг реальный пример



Бэггинг и Бустинг

Идея бэггинга:

1. Построим много слегка различающихся моделей
2. Методом усреднения выберем итоговый ответ

Идея бустинга:

Каждая следующая модель в ансамбле пытается предсказать ошибку всех предыдущих моделей ансамбля

Бутстрэп

Бутстрэп (bootstrap) – метод исследования распределения статистик вероятностных распределений, основанный на многократной генерации псевдовыборок на базе имеющейся выборки.

1. Из исходной выборки генерим псевдовыборки методом случайного выбора с возвращением.
2. На псевдовыборках считаем целевую статистику.
3. Анализируем распределение целевой статистики на псевдовыборках.

Бэггинг

Bagging – Bootstrap aggregating (Leo Breiman, 1994)

- Из Train генерим методом случайного выбора самплов с возвращением $Train'_1$ … $Train'_N$
- На каждом Train' строим модель
- Итоговое предсказание получаем усреднением предсказаний всех моделей или простым голосованием

RSM

RSM – Random Subspace Method или feature bagging

- Из Train генерим методом случайного выбора признаков без возвращения Train'$_1$ … Train'$_N$
- На каждом Train' строим модель
- Итоговое предсказание получаем усреднением предсказаний всех моделей

Random Forest

Алгоритм:

1. Выполняем N раз:

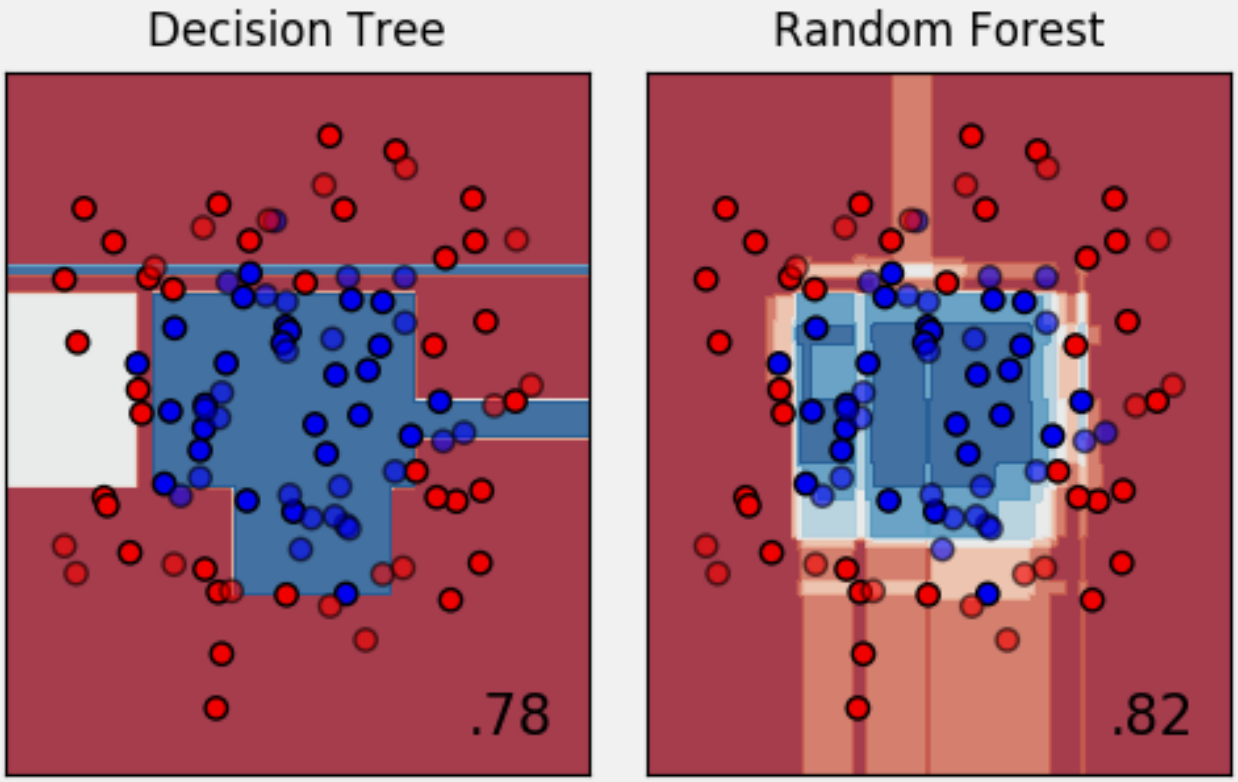
1.1. Бутстрэп самплов

1.2. Случайное подпространство признаков

1.3. Построение дерева решений

2. Выбираем ответ модели методом усреднения предсказаний или простого голосования

Random Forest



Random Forest выдает качество лучше, чем единичное решающее дерево

Random Forest

Плюсы:

1. Алгоритм прост
2. Не переобучается
3. Хорошо параллелится
4. Не требует сложной настройки параметров
5. Не требует нормализации данных

Минусы:

1. Модели не интерпретируемые*
2. Плохо работает с полиномиальными зависимостями

Параметры Random Forest

RandomForestRegressor(n_estimators, criterion, max_depth, min_samples_split, min_samples_leaf, min_weight_fraction_leaf, max_features, max_leaf_nodes, min_impurity_decrease, min_impurity_split, bootstrap, oob_score, n_jobs, random_state, verbose, warm_start)

Параметры функции потерь

Параметры ансамбля

Параметры дерева

Параметры технические

Градиентный бустинг

Идея:

1. Представляем итоговую модель f(x) как сумму слабых моделей h(x) (обычно решающие деревья малой глубины).
2. Пусть задана дифференцируемая функция потерь L(y, f(x))
3. На каждом шаге мы ищем модель h(x), которая бы апроксимировала вектор антиградиента L

Градиентный бустинг

1. Инициализировать GBM константным значением $\hat{f}(x) = \hat{f}_0, \hat{f}_0 = \gamma, \gamma \in \mathbb{R}$

$$\hat{f}_0 = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$$

2. Для каждой итерации $t = 1, \ldots, M$ повторять:

   1. Посчитать псевдо-остатки r_t

$$r_{it} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}(x)}, \quad \text{for } i = 1, \ldots, n$$

   2. Построить новый базовый алгоритм $h_t(x)$ как регрессию на псевдо-остатках $\{(x_i, r_{it})\}_{i=1,\ldots,n}$

   3. Найти оптимальный коэффициент ρ_t при $h_t(x)$ относительно исходной функции потерь

$$\rho_t = \arg\min_{\rho} \sum_{i=1}^{n} L(y_i, \hat{f}(x_i) + \rho \cdot h(x_i, \theta))$$

   4. Сохранить $\hat{f}_t(x) = \rho_t \cdot h_t(x)$

   5. Обновить текущее приближение $\hat{f}(x)$

$$\hat{f}(x) \leftarrow \hat{f}(x) + \hat{f}_t(x) = \sum_{i=0}^{t} \hat{f}_i(x)$$

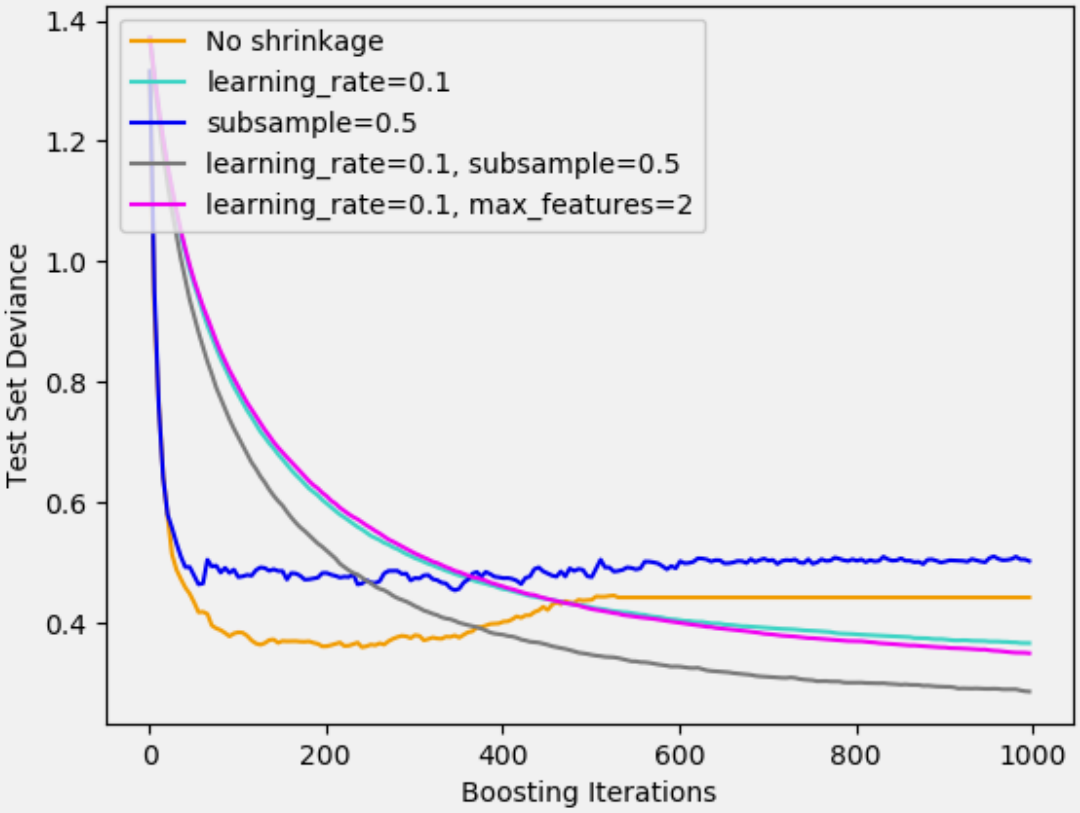
3. Скомпоновать итоговую GBM модель $\hat{f}(x)$

$$\hat{f}(x) = \sum_{i=0}^{M} \hat{f}_i(x)$$

Визуализация алгоритма

http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html

Регуляризация градиентного бустига



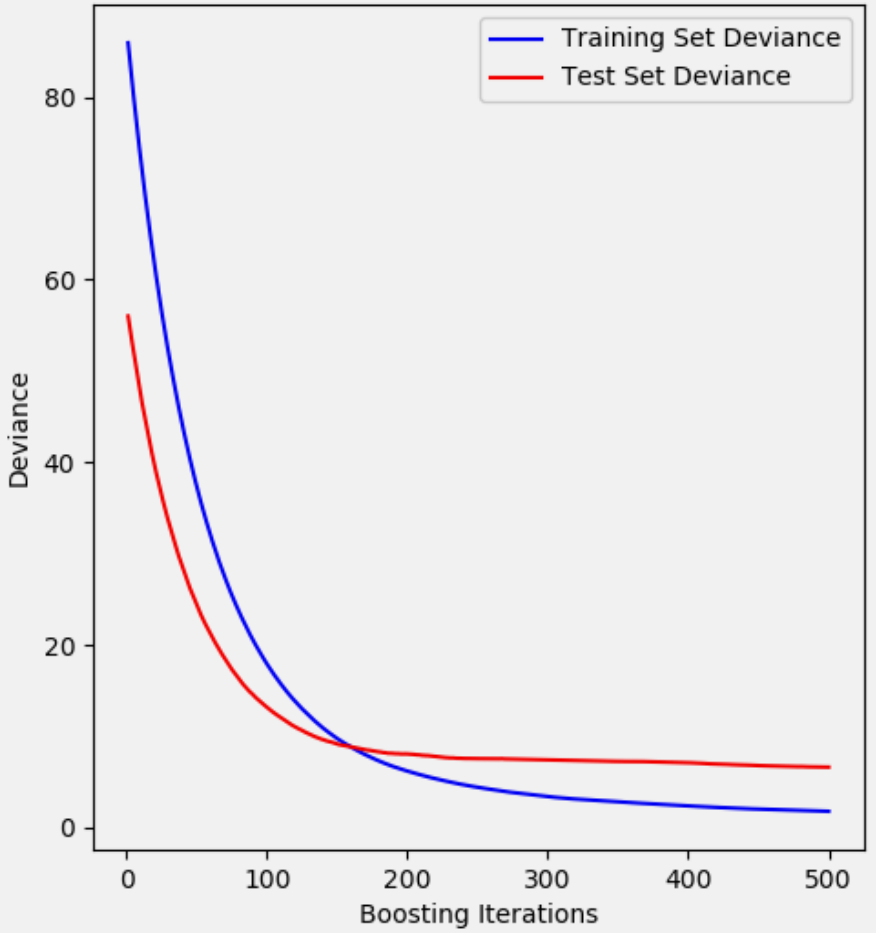
В алгоритме градиентного бустинга также применимы bagging и RSM

Переобучение градиентного бустига и early stopping

Градиентный бустинг

почти не переобучается

Early stopping - техника подбора оптимального числа итераций с помощью оценки качества ансабля на валидационном датасете на каждой итерации



Открытые реализации градиентного бустинга

dmlc XGBoost

Yandex CatBoost

Параметры градиентного бустинга

GradientBoostingClassifier(loss, learning_rate, n_estimators, subsample, criterion, min_samples_split, min_samples_leaf, min_weight_fraction_leaf, max_depth, min_impurity_decrease, min_impurity_split, init, random_state, max_features, verbose, max_leaf_nodes, warm_start, presort, validation_fraction, n_iter_no_change, tol)

Параметры функции потерь

Параметры ансамбля

Параметры дерева

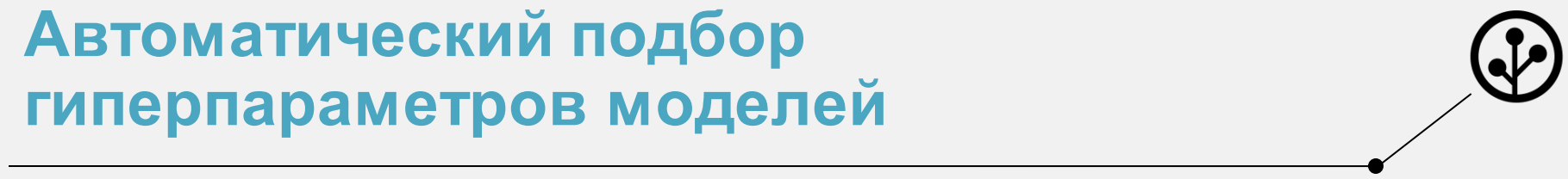
Параметры технические

Интерактивный пример

Как параметры модели влияют на результат:

http://arogozhnikov.github.io/2016/07/05/gradient_boosting_playground.html

Автоматический подбор гиперпараметров моделей

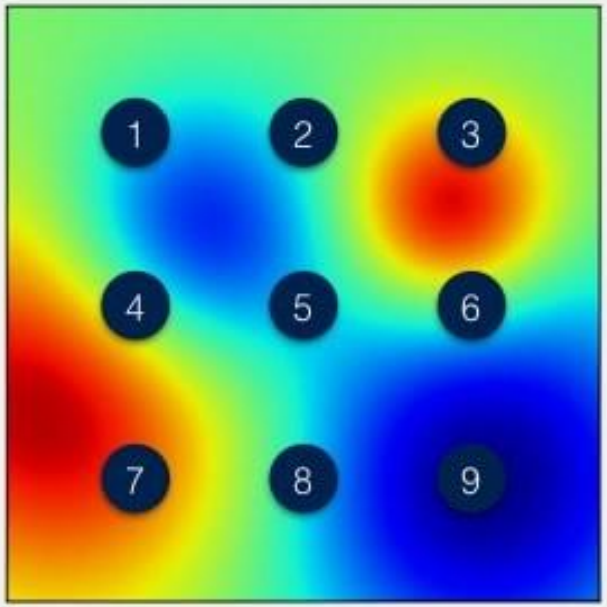


Модель(параметры) -> качество

Задача - максимизировать качество

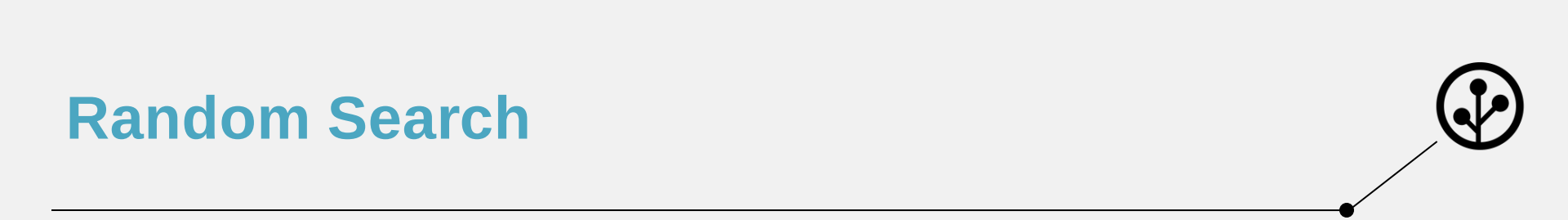
Grid Search

1. Перебираем параметры модели по решетке
2. Выбираем параметры, которые дают самое высокое качество

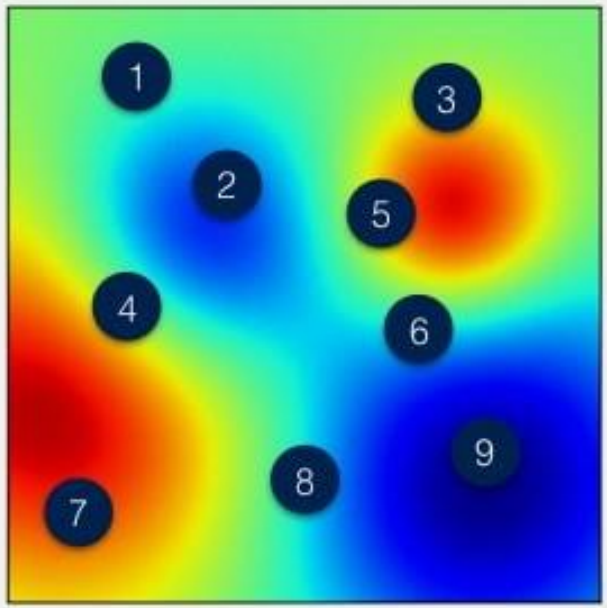


Grid Search

Random Search



1. Сэмплируем параметры модели
2. Выбираем параметры, которые дают самое высокое качество



Random Search

Практические Рекомендации

1. С использованием "производительного" разбиения датасета находим где примерно находится максимум качества в пространстве гиперпараметров

2. С использованием "точного" разбиения подбираем оптимальные параметры в окрестности максимума качества

Продвинутые методы оптимизации

Идея:

На основании уже совершенных проб пытаемся предсказать где находится глобальный максимум качества

Плюсы:

Находит максимум за меньшее число проб

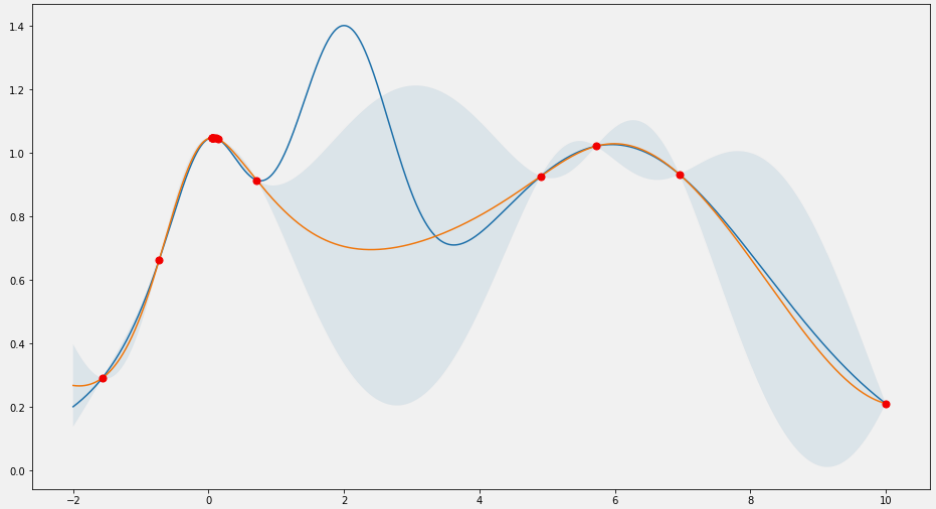
Минусы:

Сложен в настройке

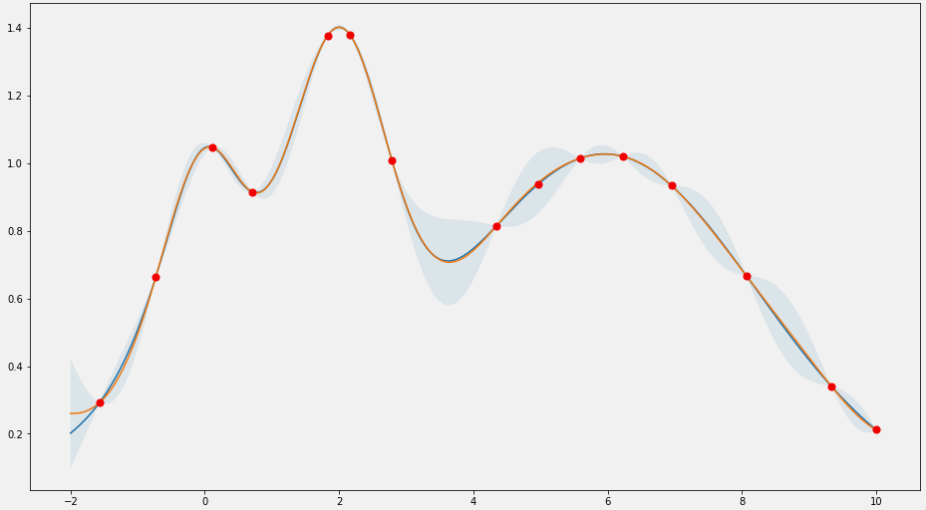
Может упустить глобальный максимум

Exploitation vs Exploration

Exploitation



Exploration



Инструменты для оптимизации гиперпараметров

- sklearn
- Hyperopt
- BayesianOptimization
- Hyppopy
- Optunity
- Optuna

ТЕХНОСФЕРА

Семинар

Соревнование "Property prices"

https://www.kaggle.com/c/introml2019-2

Домашнее задание №#5

- Сделать сабмит решения
- Выложить решение на github.com
- Прислать ссылку на код решения, свой профиль kaggle

Срок сдачи

10 ноября 2019

ТЕХНОСФЕРА

Спасибо за внимание!

Евгений Некрасов

e.nekrasov@corp.mail.ru