Деревья решений
Ансамбли

Tinkoff Generation

Лиза Корнеева

Деревья решений

Деревья решений вокруг нас

Как построить оптимальное дерево?

Рассмотрим игру «20 вопросов»: один человек загадывает знаменитость, а второй отгадывает, задавая только закрытые вопросы («Да» или «Нет»)

- Вопрос «Это Анджелина Джоли?» в случае ответа «Нет» оставит более 7 миллиардов вариантов для дальнейшего перебора.
- А вот вопрос «Это женщина?» отсечет уже около половины знаменитостей.

Признак «пол» намного лучше разделяет выборку людей, чем «это Анджелина Джоли»

Это интуитивно соответствует понятию **прироста информации**, основанного на **энтропии**.

Как построить оптимальное дерево?

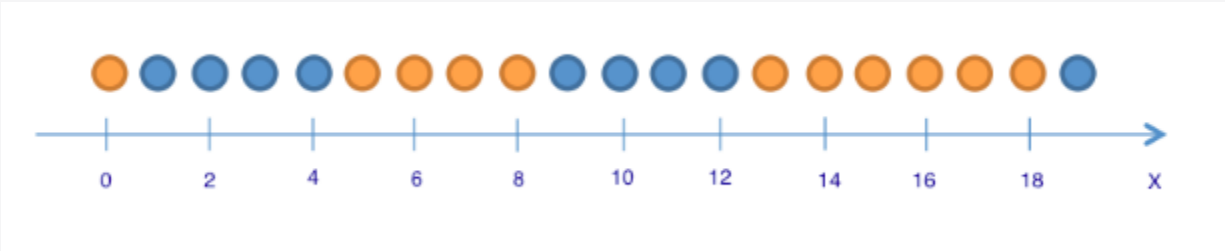
Интуитивно, энтропия соответствует степени хаоса в системе.
Чем выше энтропия, тем менее упорядочена система и наоборот.

$$S = -\sum_{i=1}^{N} p_i \log_2 p_i,$$

где p_i – вероятности нахождения системы в i-ом состоянии.

Пример построения дерева

Будем предсказывать цвет шарика по его координате

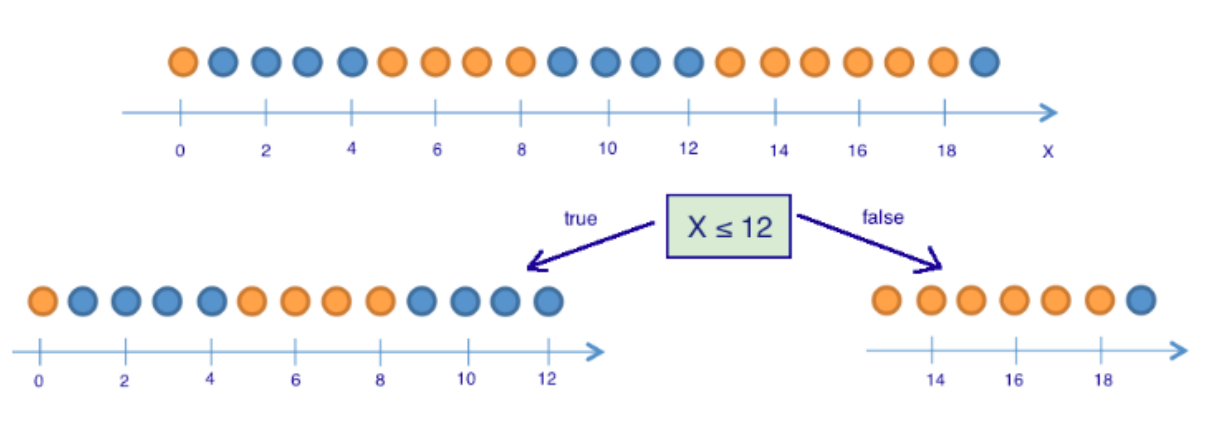


Здесь 9 синих шариков и 11 желтых

Если мы наудачу вытащили шарик, то он

- с вероятностью p_1 = 9/20 будет синим
- с вероятностью p_2 = 11/20 – желтым

Энтропия состояния S0 = −(9/20) * log2(9/20) − (11/20)*log2(11/20) ≈ 1



В левой группе оказалось 13 шаров, из которых 8 синих и 5 желтых.

$S1 = -(5/13) * \log2(5/13) - (8/13) * \log2(8/13) \approx 0.96$

В правой группе оказалось 7 шаров, из которых 1 синий и 6 желтых.

$S2 = -(1/7) * \log2(1/7) - (6/7) * \log2(6/7) \approx 0.6$

Как видим, энтропия уменьшилась в обеих группах по сравнению с начальным состоянием, хоть в левой и не сильно.

Прирост информации (information gain)

Поскольку энтропия – по сути степень хаоса в системе, уменьшение энтропии называют приростом информации.

Формально прирост информации при разбиении выборки по признаку Q (в нашем примере это признак «x ≤ 12») определяется как

$$IG(Q) = S_O - \sum_{i=1}^{q} \frac{N_i}{N} S_i,$$

где q – число групп после разбиения, N_i – число элементов выборки, у которых признак Q имеет i-ое значение.

Прирост информации в нашем примере

В нашем случае после разделения получилось

- две группы (q = 2)
- одна из 13 элементов (N_1 = 13)
- вторая – из 7 (N_2 = 7)

Прирост информации получился

$$IG(x \leq 12) = S_0 - \frac{13}{20} S_1 - \frac{7}{20} S_2 \approx 0.16.$$

Получается, разделив шарики на две группы по признаку «координата меньше либо равна 12», мы уже получили более упорядоченную систему, чем в начале.

Общий принцип построения дерева

В популярных алгоритмах (ID3 и C4.5) используется принцип жадной максимизации прироста информации – на каждом шаге выбирается тот признак, при разделении по которому прирост информации оказывается наибольшим.

Дальше процедура повторяется рекурсивно, пока энтропия не окажется равной нулю или какой-то малой величине.

В разных алгоритмах применяются разные эвристики для "ранней остановки" или "отсечения", чтобы избежать построения переобученного дерева.

Другие критерии качества разбиения

- Неопределенность Джини (Gini impurity)

$$G = 1 - \sum_k (p_k)^2$$

Максимизацию этого критерия можно интерпретировать как максимизацию числа пар объектов одного класса, оказавшихся в одном поддереве.

- Ошибка классификации (misclassification error)

$$E = 1 - \max_k p_k$$

Дерево решений для регрессии

При прогнозировании количественного признака идея построения дерева остается та же, но меняется критерий качества:

$$D = \frac{1}{\ell}\sum_{i=1}^{\ell}(y_i - \frac{1}{\ell}\sum_{i=1}^{\ell} y_i)^2$$

где ℓ – число объектов в листе, y_i – значения целевого признака.

Минимизируя дисперсию вокруг среднего, мы ищем признаки, разбивающие выборку таким образом, что значения целевого признака в каждом листе примерно равны.


Decision tree regressor, MSE = 17.49
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−4
−2
0
2
4

Когда можно остановиться?

- Ограничение максимальной глубины дерева
- Ограничение минимального числа объектов в листе
- Ограничение максимального количества листьев в дереве
- Остановка в случае, если все объекты в листе относятся к одному классу
- Требование, что функционал качества при дроблении улучшался как минимум на x процентов

Плюсы и минусы деревьев

Плюсы

- Легко визуализировать
- Интерпретируемые
- Быстро учатся и прогнозируют
- Не надо много данных

Минусы

- Чувствительны к шумам
- Разделяющая граница имеет ограничения
- Проблема поиска оптимального дерева решений NP-полна
- Переобучаются
- Модель умеет только интерполировать, но не экстраполировать

Ансамбли

Коллективный разум

Группа людей дает ответ точнее, нежели эксперт

Эксперимент

1. Девушка набила гигантскую банку m&ms
2. Спрашивала, сколько в банке конфет
3. В эксперименте участвовало 160 человек
4. Разные ответы от 400 до 50 000
5. Затем посчитали среднее
6. Среднее оказалось равно 4515, что всего на 5 больше, чем реальное число конфет

Коллективный разум

https://cindicator.com/

We create new questions every day

Cindicator's in-house team of professional financial analysts monitors crypto and traditional assets to create new questions about the most interesting market opportunities.

You make forecasts and score points

You can answer all of the questions or just the ones you like. Every question has a deadline. Once the question is closed, you win points if your forecast was correct and lose points if you were wrong.

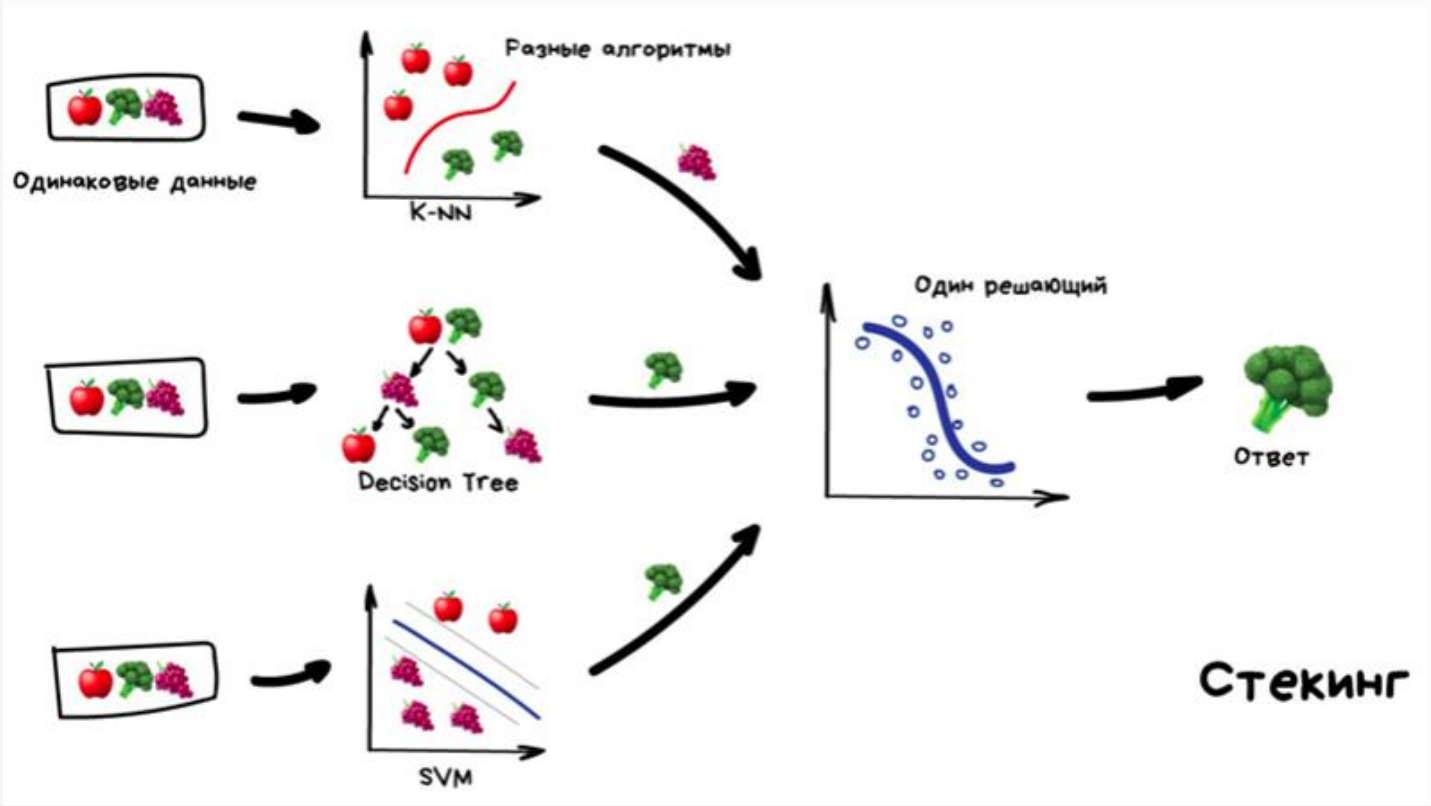
Hybrid Intelligence generates indicators

Several layers of machine learning models process all the answers to create valuable indicators for traders who hold CND, Cindicator's ERC-20 token. Your forecasts directly contribute to making Cindicator's Hybrid Intelligence more valuable for all ecosystem participants, including yourself.

You get rewards every month

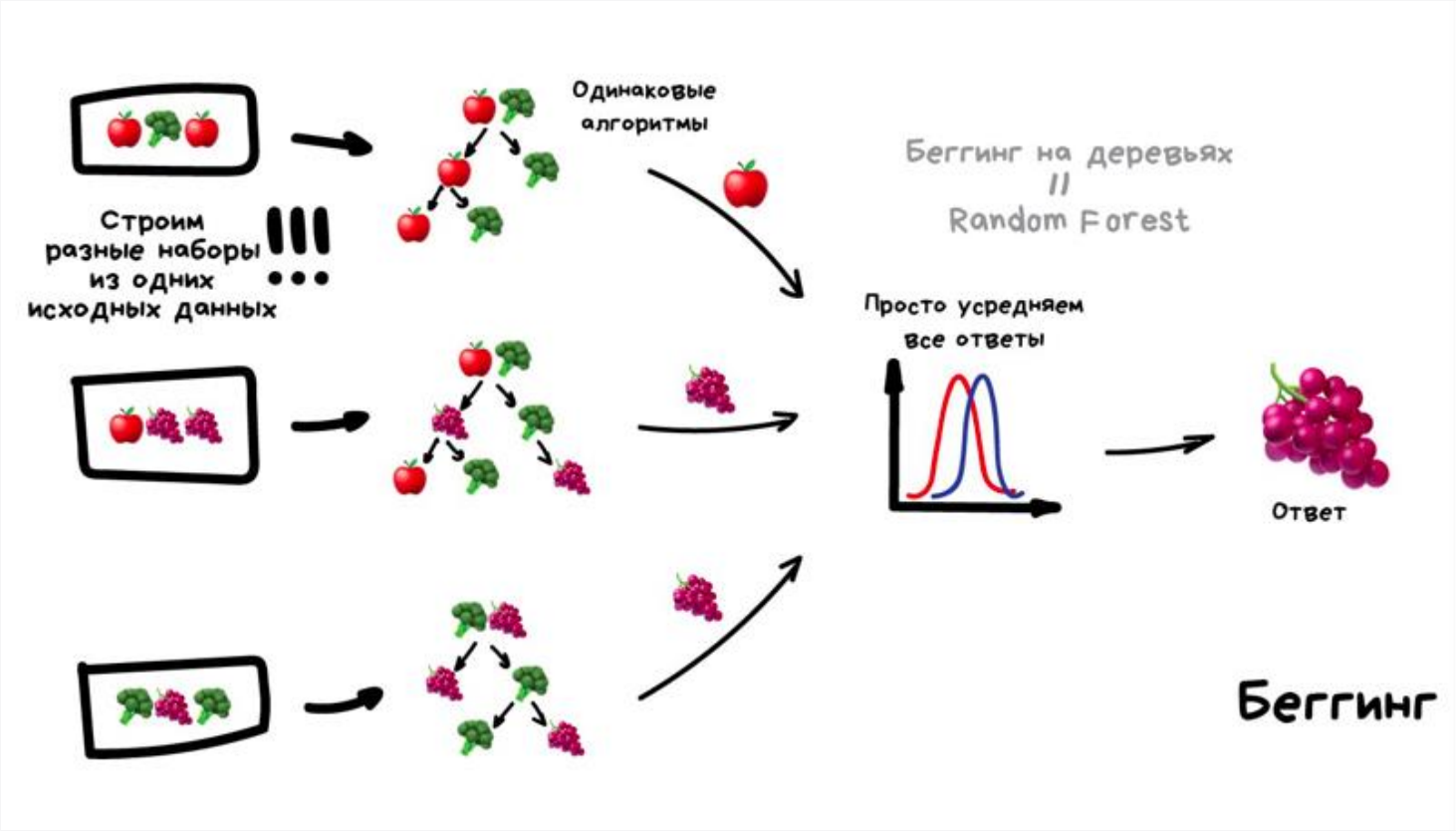
At the end of the month, you win a share of the Cindicator ecosystem's motivational pool (in CND or ETH) as long as your rating is positive. The more points you have and the higher your rating, the greater the reward you win! And the higher your accuracy, the more valuable the whole Hybrid Intelligence ecosystem becomes.

Стекинг


Одинаковые данные
Разные алгоритмы
K-NN
Decision Tree
SVM
Один решающий
Ответ
Стекинг

Случайный лес

Бэггинг


Одинаковые алгоритмы
Строим разные наборы из одних исходных данных
!!!
Беггинг на деревьях
//
Random Forest
Просто усредняем все ответы
Ответ
Беггинг

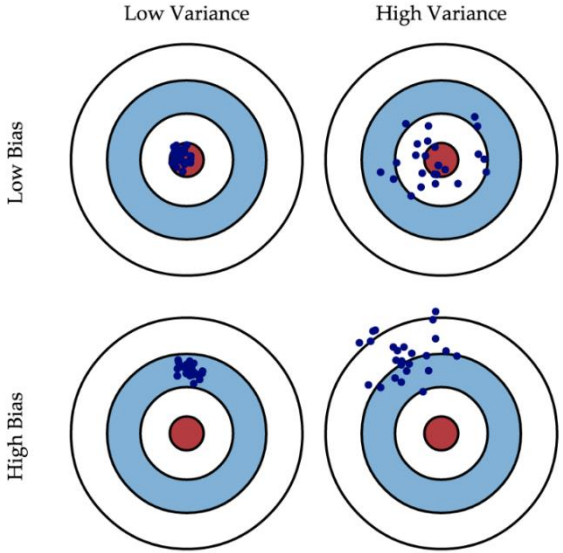
Случайный лес

Бэггинг позволяет объединить несмещенные, но чувствительные к обучающей выборке алгоритмы в несмещенную композицию с низкой дисперсией

1. Деревья могут достигать нулевую ошибку на любой выборке (низкое смещение)
2. Деревья легко переобучаются

Делаем рандомизацию по двум направлениям

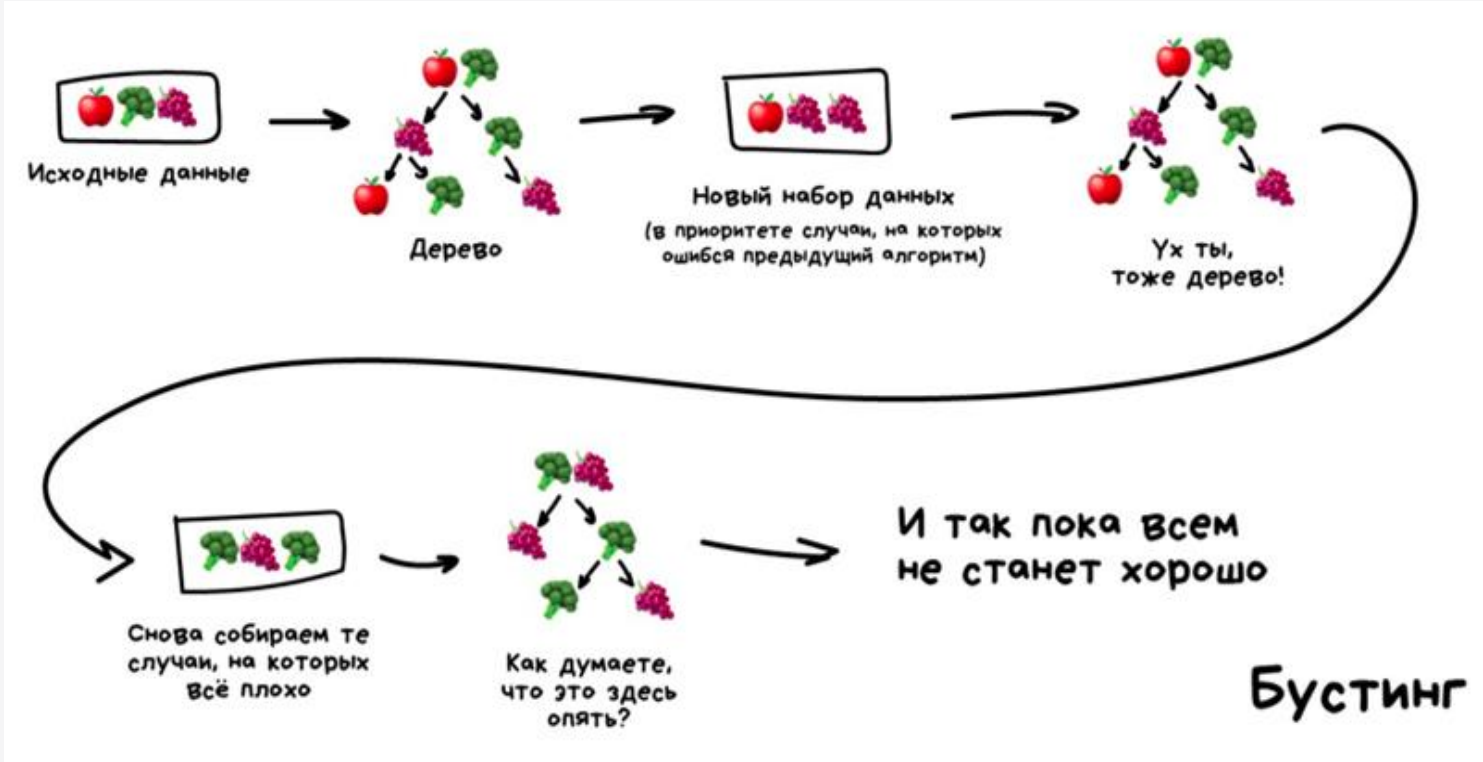
1. Подвыборка объектов
2. В каждой вершине разбиение ищется по подмножеству признаков



https://github.com/esokolov/ml-course-hse/blob/master/2018-fall/lecture-notes/lecture08-ensembles.pdf

Градиентный бустинг

Бустинг


Исходные данные
Дерево
Новый набор данных
(в приоритете случаи, на которых ошибся предыдущий алгоритм)
Ух ты, тоже дерево!
Снова собираем те случаи, на которых всё плохо
Как думаете, что это здесь опять?
И так пока всем не станет хорошо
Бустинг

Градиентный бустинг

Рассмотрим регрессию – хотим построить итоговый алгоритм как сумму базовых

1. Строим первый базовый алгоритм
2. Считаем остатки – расстояния от нашего ответа до реального
3. Если прибавить эти остатки к ответам построенного алгоритма, то он не будет допускать ошибок на обучающей выборке
4. Строим следующий алгоритм так, чтобы ответы были близки к остаткам

Для сложных моделей изменения ответов делают учитывая **градиент функции потерь**

Известные алгоритмы: lightGBM, xgboost, catboost

https://github.com/esokolov/ml-course-hse/blob/master/2018-fall/lecture-notes/lecture09-ensembles.pdf
https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db

Вопросы?