



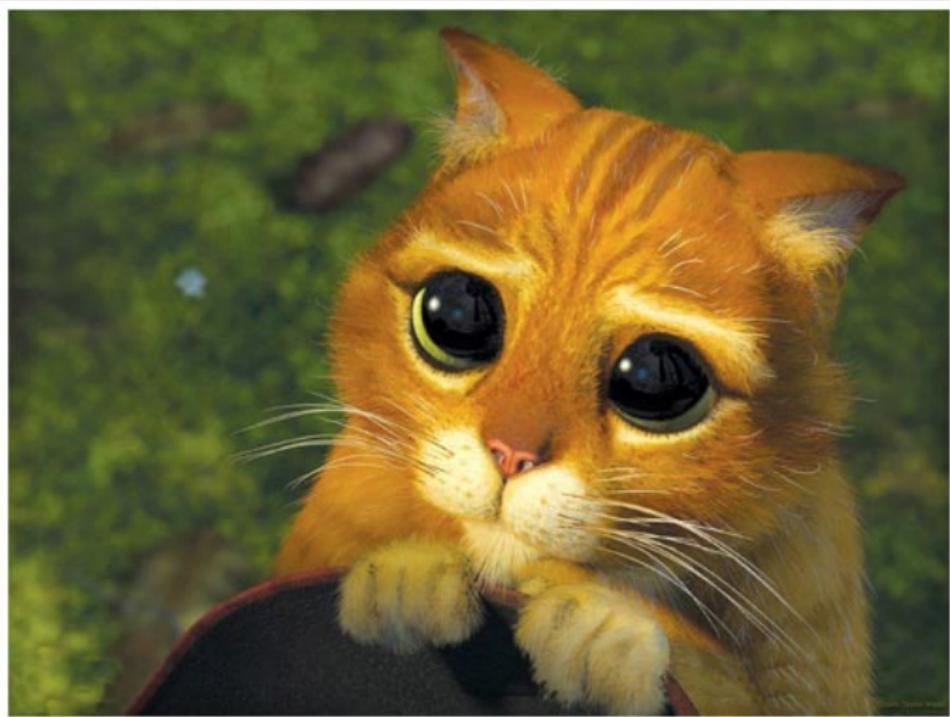
ТЕХНОСФЕРА

Лекция 1 Алгоритмические композиции Начало

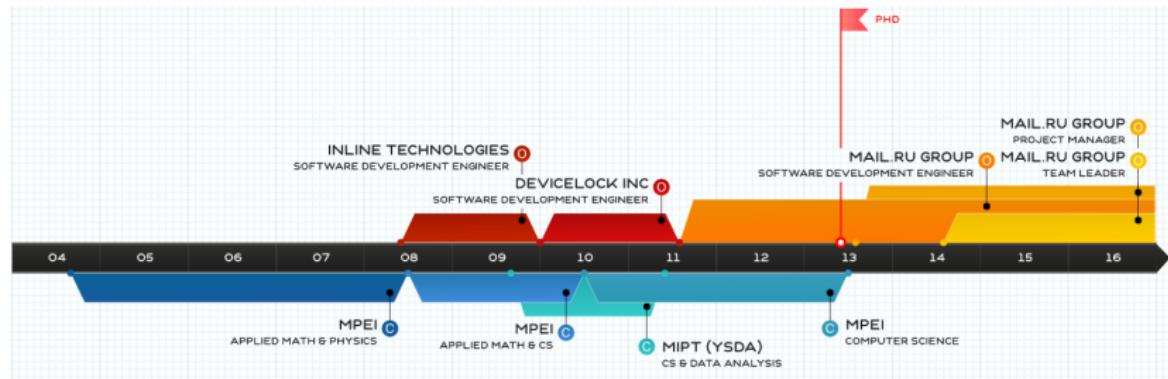
Владимир Гулин

12 февраля 2016 г.

Отметиться



Владимир Гулин



e-mail: gulin.vladimir.sfera@mail.ru

тел.: +7 (915) 416-95-75

Структура курса

Модуль 1 (Владимир Гулин)

1. Алгоритмические композиции 1
2. Алгоритмические композиции 2^H
3. Снижение размерности. Отбор признаков
4. Снижение размерности. Выделение признаков ^H
5. Активное обучение ^H
6. Частичное обучение
7. Рекомендательные системы. Коллаборативная фильтрация ^H

Модуль 2 (Павел Нестеров)

8. Многослойный персепtron. Алгоритм обратного распространения ошибки ^H
9. Улучшение алгоритма обратного распространения ошибки
10. Стохастические нейронные сети ^H
11. Предобучение многослойного персептрана ^H
12. Глубокие сети
13. Сверточные сети
14. Перспективы глубоких нейронных сетей

Контроль знаний

Правила игры

7 домашних задания на 1-8 часов самостоятельной работы каждое (в сумме 100 баллов) + курсовой проект (40 баллов)

На выполнение каждого дз дается 2 недели, + еще две недели, чтобы досдать и получить оценку.

Через 4 недели после получения домашки оценку получить за данную домашку будет невозможно!!!

Немного о структуре курса

Что такое система машинного обучения?

Данные

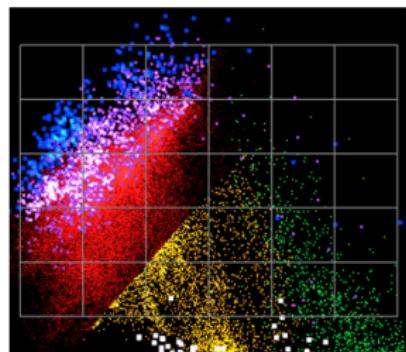
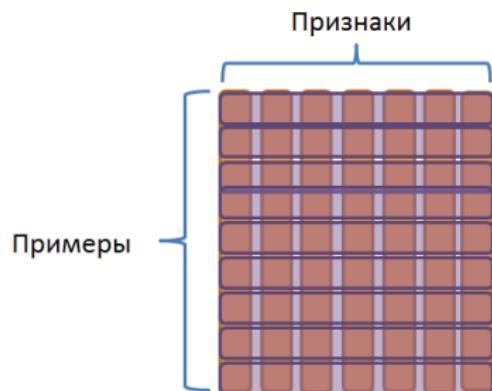


Алгоритм обучения



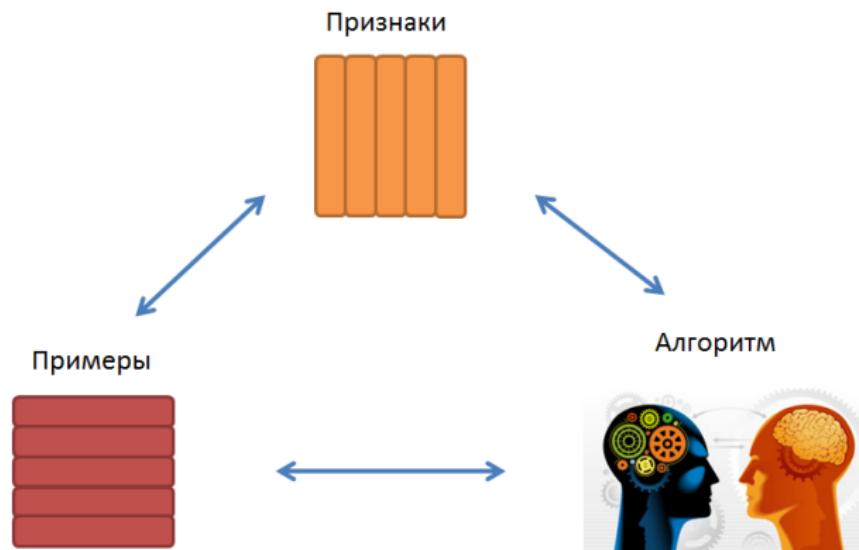
Немного о структуре курса

Что такое данные?



Немного о структуре курса

Какая связь?



А что важнее?

Немного о структуре курса

Asirra

Please click on all the images that show cats:

?

?

adopt me	adopt me	adopt me	adopt me
adopt me	adopt me	adopt me	adopt me
adopt me	adopt me	adopt me	adopt me

Точность распознавания в 2006 (60%)
 $0.6^{12} \approx 0.002$

Немного о структуре курса

ASIRRA



After 8 years of operation, Asirra is shutting down effective October 1, 2014. Thank you to all of our users!

Соревновашка на kaggle.com (Cats vs Dogs)
Точность распознавания в 2014 (98.9%)
 $0.989^{12} \approx 0.875$

План лекции

Мотивация

Обзор методов построения алгоритмических композиций

Стохастические методы построения алгоритмических композиций

Random Forest

Выбираем ноутбук

Параметры ноутбука

- ▶ Камень
- ▶ Память
- ▶ Видеокарта
- ▶ Диагональ монитора
- ▶ и т.д



Воспользуемся экспертым мнением

- ▶ Консультант в магазине
- ▶ Сосед “программист”
- ▶ Друзья
- ▶ Форумы

Задача обучения с учителем

Постановка задачи

Пусть дан набор объектов $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$, $\mathbf{x}_i \in \mathcal{X}$, $y_i \in \mathcal{Y}$, $i \in 1, \dots, N$, полученный из неизвестной закономерности $y = f(\mathbf{x})$. Необходимо построить такую $h(\mathbf{x})$, которая наиболее точно аппроксимирует $f(\mathbf{x})$.

Будем искать неизвестную

$$h(\mathbf{x}) = C(a_1(\mathbf{x}), \dots, a_T(\mathbf{x}))$$

$a_i(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{R}$, $\forall i \in \{1, \dots, T\}$ - базовые модели
 $C : \mathcal{R} \rightarrow \mathcal{Y}$ - решающее правило

Простое голосование

Simple Voting

$$h(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T a_i(\mathbf{x})$$



Взвешенное голосование

Weighted Voting

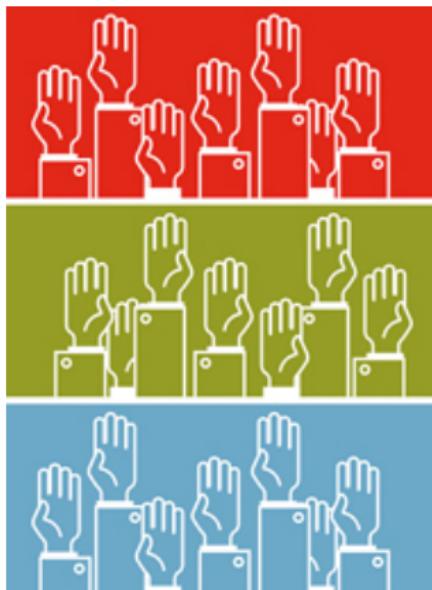
$$h(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T b_i a_i(\mathbf{x}), \quad b_i \in \mathcal{R}$$



Смесь экспертов

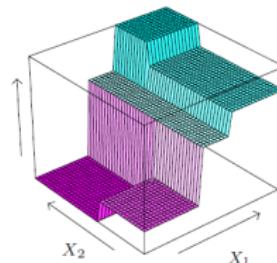
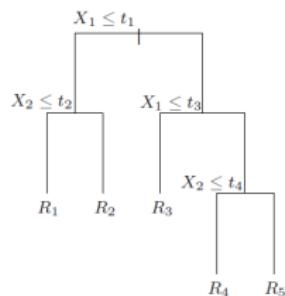
Mixture of Experts

$$h(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T b_i(\mathbf{x}) a_i(\mathbf{x}), \quad b_i(\mathbf{x}) : \mathcal{X} \rightarrow \mathcal{R}$$



Топологические композиции

Деревья решений

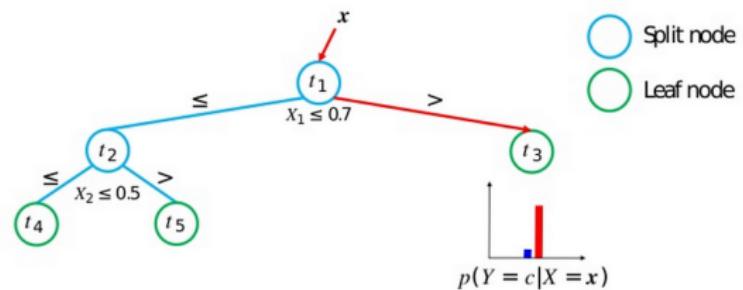
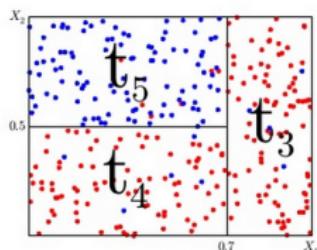


Нужно понимать:

- ▶ Любое недвоичное дерево решений можно представить двоичным
- ▶ Любая линейная комбинация деревьев решений есть дерево решений

Топологические композиции

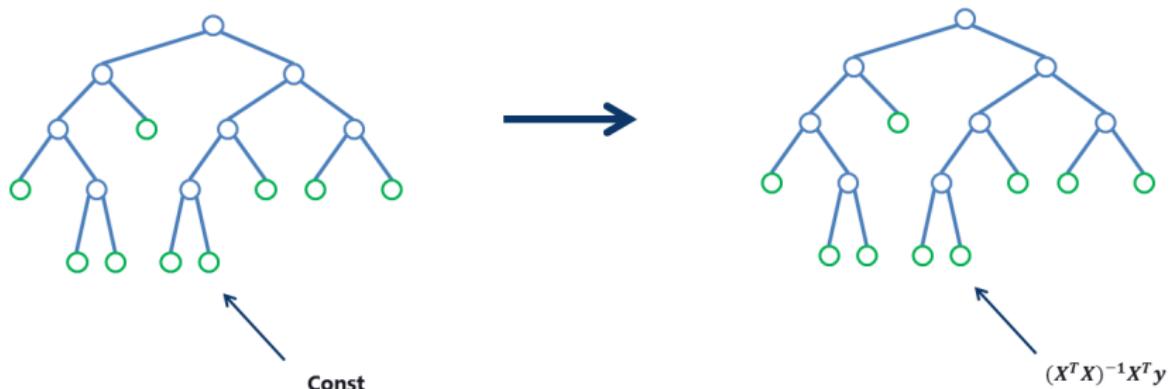
Деревья решений (принцип работы)



Топологические композиции

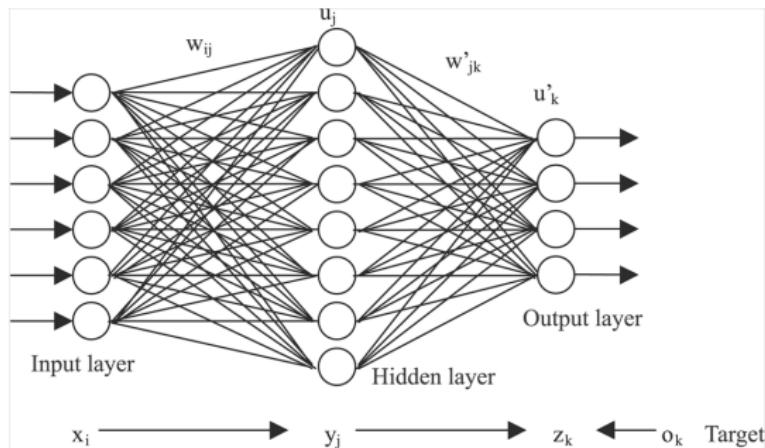
Модельные деревья решений (Model decision trees)

- ▶ Поместим в листья деревьев какие-нибудь алгоритмы вместо констант



Нейронные сети

Neural networks



$$h(\mathbf{x}) = g(\mathbf{w}^T \mathbf{x}) - \text{композиция линейных моделей}$$

Текущее положение дел

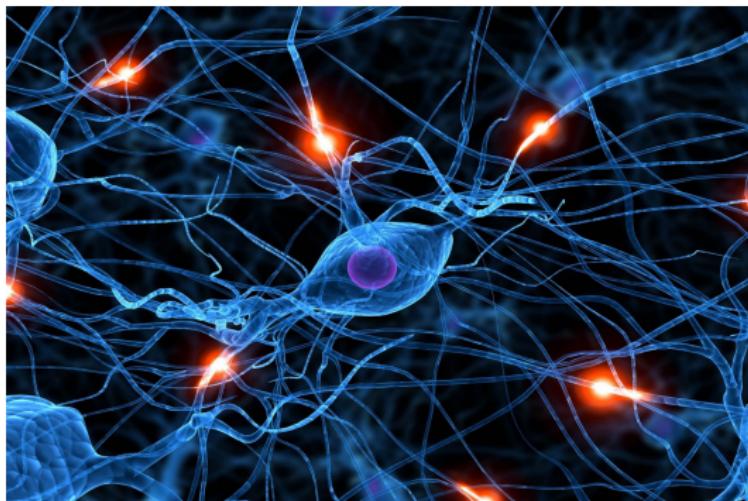
Decision Trees Compositions vs Neural Nets



Neural nets

Где нейронные сети побеждают?

- ▶ Computer Vision (ImageNet)
- ▶ Speech recognition



Decision trees compositions

Где побеждают ансамбли деревьев решений?

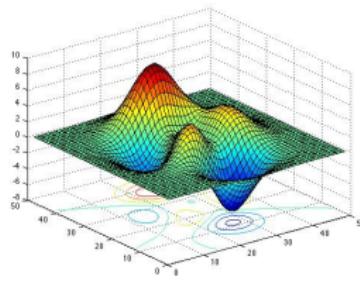
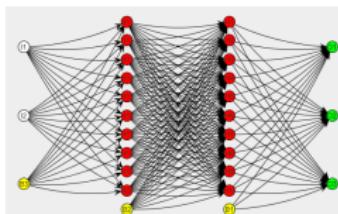
- ▶ Learning to rank (Yahoo Learning to rank challenge 2010)
- ▶ Text categorization (NIPS 2006)
- ▶ Везде



Взгляд с точки зрения функционального анализа

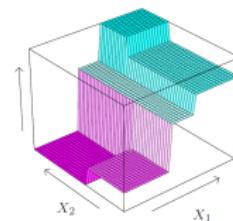
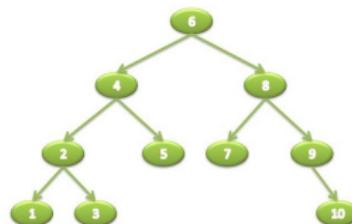
Гладкие функции

$$h(\mathbf{x}) = \sum \sigma(\dots \sum \sigma(\mathbf{w}^T \mathbf{x}))$$



Кусочно-постоянные функции

$$h(\mathbf{x}) = \sum_d c_d I\{\mathbf{x} \in R_d\}$$



Методы построения алгоритмических композиций

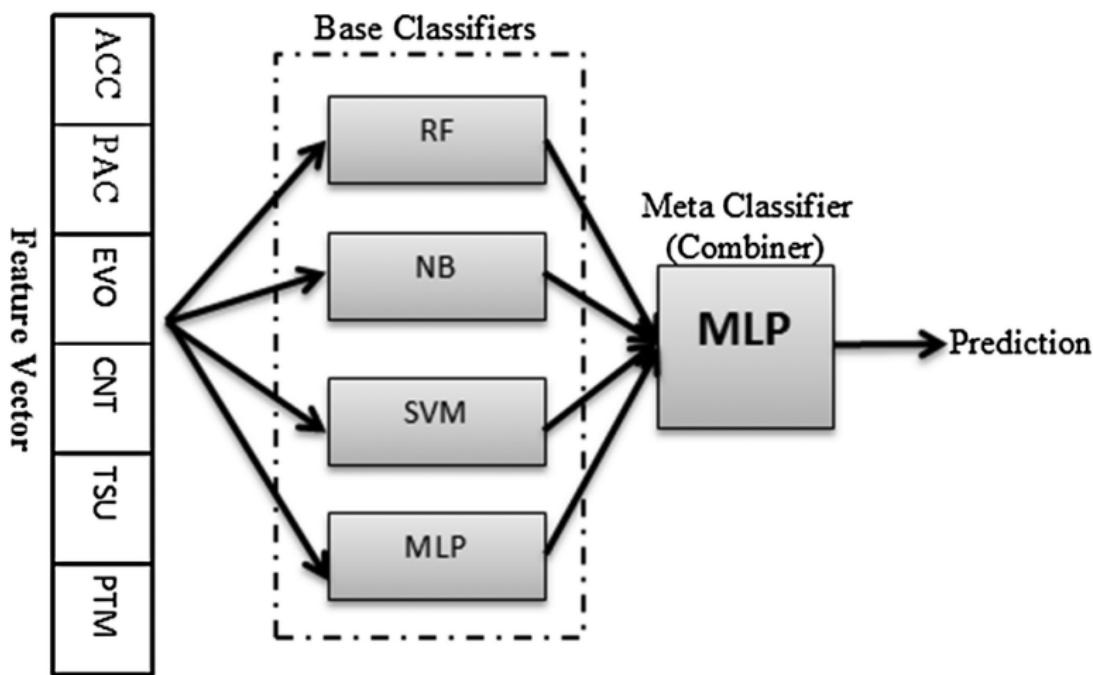
Стохастические методы

Бустинг

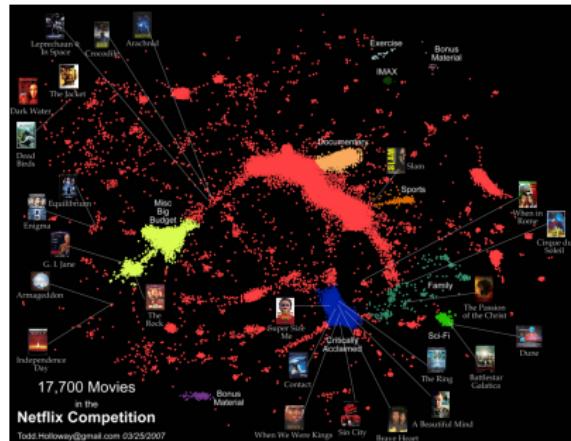
Мета алгоритмы. Stacking

Stacking

Мета-алгоритм

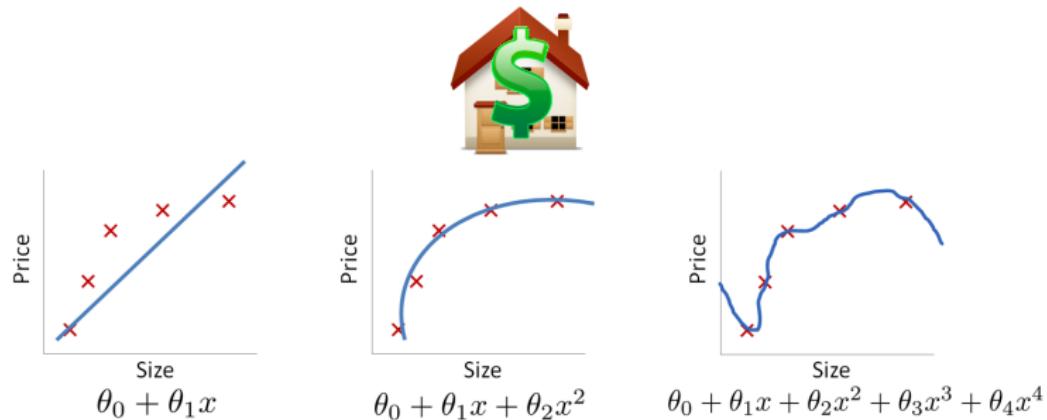


Netflix Challenge



- ▶ Задача предсказания оценки фильму
 - ▶ ПФ - 1000000\$
 - ▶ Победил Stacking на “зоопарке” алгоритмов

Немного теории



Недообучение - сложность модели недостаточна. Данные имеют более сложную природу.

Переобучение - сложность модели избыточна. Модель слишком настроена на выборку. Обобщающая способность алгоритма низка.

The bias-variance Decomposition

- ▶ Задача регрессии
- ▶ $y = f(x) + \epsilon$
- ▶ $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon)$
- ▶ Ищем $h(\mathbf{x})$ аппроксимирующую $f(\mathbf{x})$ Таким образом, мы можем оценить ожидание среднеквадратичной ошибки для некоторой точки \mathbf{x}_0

$$Err(\mathbf{x}_0) = E[(y - h(\mathbf{x}_0))^2]$$

The bias-variance Decomposition

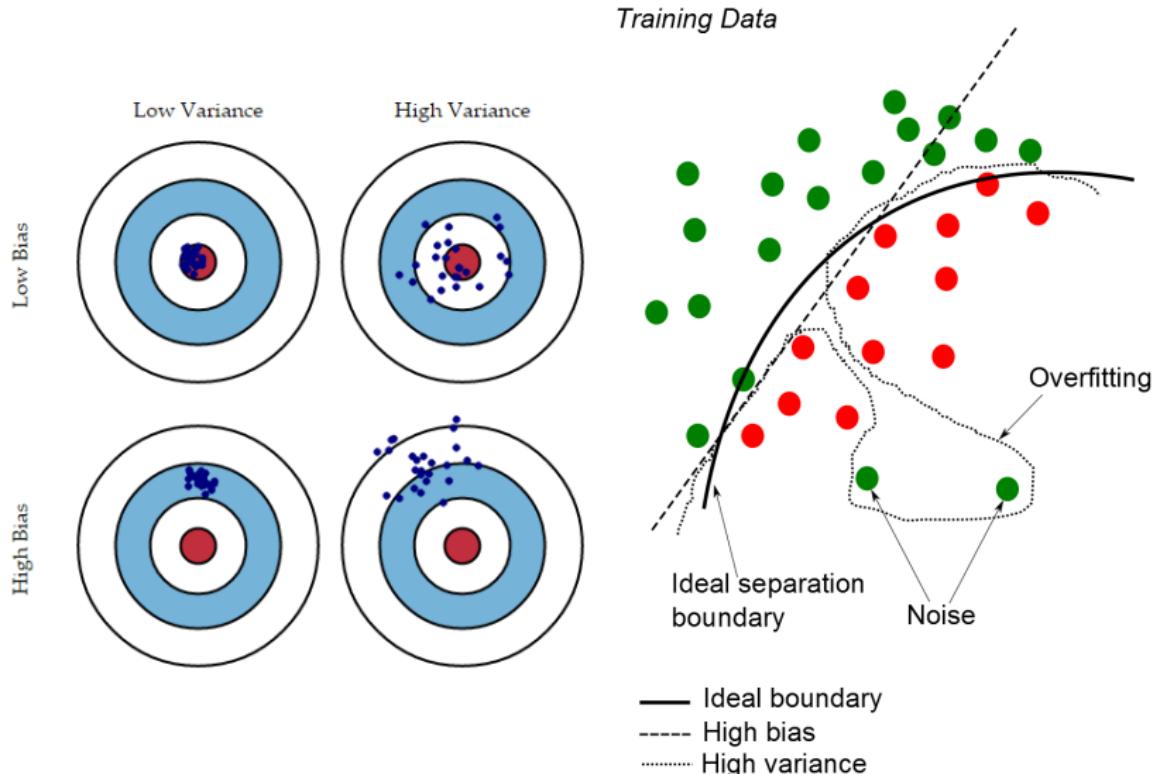
Это можно переписать в виде

$$\begin{aligned} Err(\mathbf{x}_0) &= E[(y - h(\mathbf{x}_0))^2] = \\ &= E[(y^2 + h(\mathbf{x}_0)^2 - 2yh(\mathbf{x}_0))] = E[y^2] + E[h(\mathbf{x}_0)^2] - 2E[yh(\mathbf{x}_0)] \\ E[y^2] &= Var[y] + E[y]^2 \\ Var[y] &= E[(y - E(y))^2] = E[(y - f(\mathbf{x}_0))^2] = E[(f(\mathbf{x}_0) + \epsilon - f(\mathbf{x}_0))^2] = \\ &= E[\epsilon^2] = Var[\epsilon] + E[\epsilon]^2 = \sigma_\epsilon^2 \\ E[h(\mathbf{x}_0)^2] &= Var[h(\mathbf{x}_0)] + E[h(\mathbf{x}_0)]^2 = \\ E[yh(\mathbf{x}_0)] &= E[(f + \epsilon)h(\mathbf{x}_0)] = fE[h(\mathbf{x}_0)] \end{aligned}$$

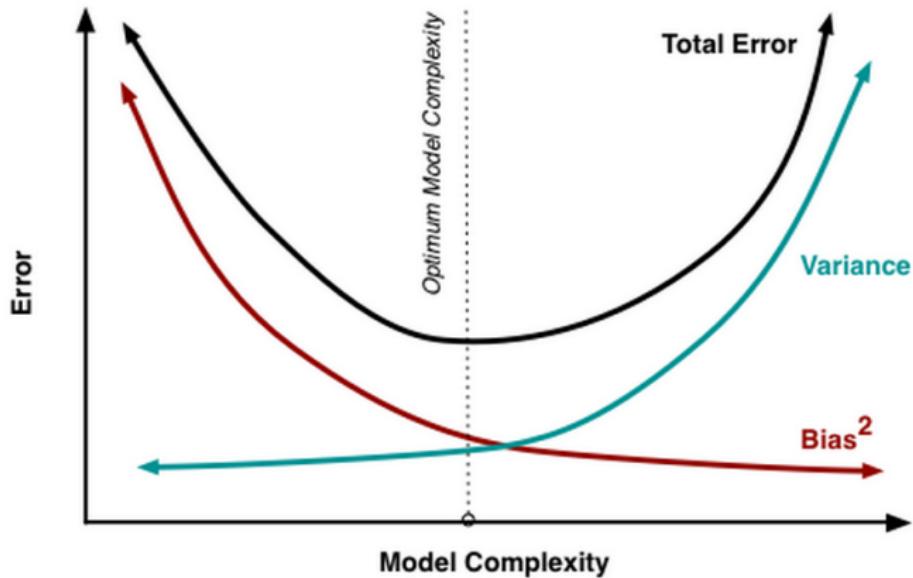
т.к. f -детерминирована, а ϵ и $h(\mathbf{x}_0)$ - независимы

$$\begin{aligned} Err(\mathbf{x}_0) &= Var[y] + E[y]^2 + Var[h(\mathbf{x}_0)] + E[h(\mathbf{x}_0)]^2 - 2fE[h(\mathbf{x}_0)] = \\ &= Var[y] + Var[h(\mathbf{x}_0)] + (f - E[h(\mathbf{x}_0)])^2 \\ Err(\mathbf{x}_0) &= (E[h(\mathbf{x}_0)] - f(\mathbf{x}_0))^2 + E[h(\mathbf{x}_0) - E(h(\mathbf{x}_0))]^2 + \sigma_\epsilon^2 \\ Err(\mathbf{x}_0) &= Bias^2 + Variance + Noise(Irreducible\ Error) \end{aligned}$$

Bias and variance tradeoff



Bias and variance tradeoff



Стохастические методы

Стратегии

- ▶ **Bagging = Bootstrap aggregation.** Обучаем алгоритмы по случайным подвыборкам размера N , полученным с помощью выбора с возвращением.
- ▶ **RSM = Random Subspace Method.** Метод случайных подпространств. Обучаем алгоритмы по случайным подмножествам признаков.

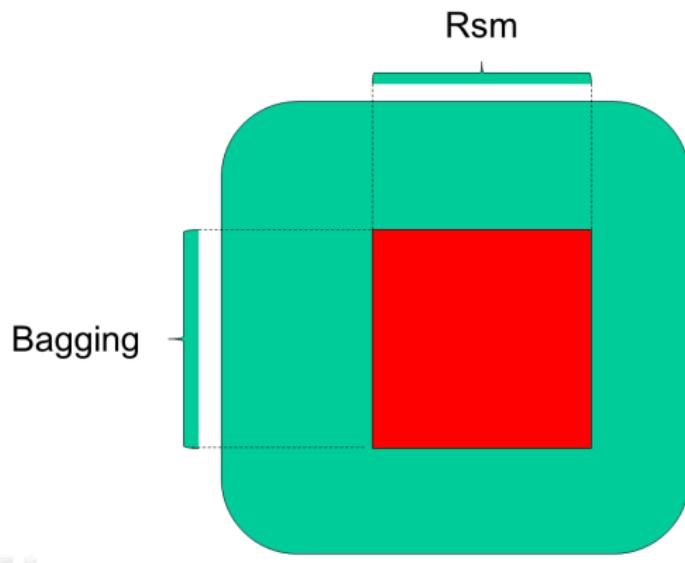
Благодаря описанным стратегиям добиваемся максимального различия между базовыми алгоритмами.

Bagging & RSM

```
1 function BaggingRSM(X, F, poi, pof, T):
2     # X - point from dataset
3     # F - original set of features
4     # poi - percent of items
5     # pof - percent of features
6     # T - number of algorithms
7     for j in 1..T:
8         fsubset = sample_features(F, pof)
9         xsubset = sample_items(X, poi)
10        a_j = build_base_model(xsubset, fsubset)
11
12    return a_1 ... a_T
```

Bagging & RSM

Геометрическая интерпретация



Bias-Variance for Bagging & RSM

Модель простого голосования

$$h(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T a_i(\mathbf{x})$$

Смещение и разброс

$$\begin{aligned} Bias^2 &= (E[h(\mathbf{x}_0)] - f(\mathbf{x}_0))^2 = (E[a_i(\mathbf{x}_0)] - f(\mathbf{x}_0))^2 \\ Variance &= E[h(\mathbf{x}_0) - E(h(\mathbf{x}_0))]^2 = \\ &= \frac{1}{T^2} \sum_{i=1}^T \sum_{j=1}^T E[(a_i(\mathbf{x}_0) - E[a_i(\mathbf{x}_0)])(a_j(\mathbf{x}_0) - E[a_j(\mathbf{x}_0)])] = \\ &= \frac{1}{T^2} \sum_{i=1}^T \left(\sigma^2 + \sum_{i \neq j} cov(a_i(\mathbf{x}_0), a_j(\mathbf{x}_0)) \right) = \\ &= \frac{1}{T^2} \sum_{i=1}^T (\sigma^2 + (T-1)\sigma^2 \cdot \rho) = \\ &= \frac{\sigma^2}{T} + \frac{(T-1)\sigma^2 \cdot \rho}{T} = \rho\sigma^2 - \frac{\rho\sigma^2}{T} + \frac{\sigma^2}{T} = \rho\sigma^2 + \sigma^2 \frac{1-\rho}{T} \end{aligned}$$

Leo Breiman



- ▶ Один из авторов деревьев решений
- ▶ Автор bagging
- ▶ Сделал очень много для развития машинного обучения

Random Forest

Случайный лес (random forest) - bagging & rsm над решающими деревьями.

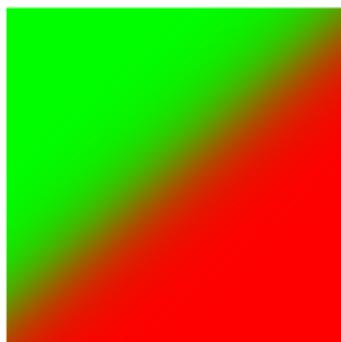
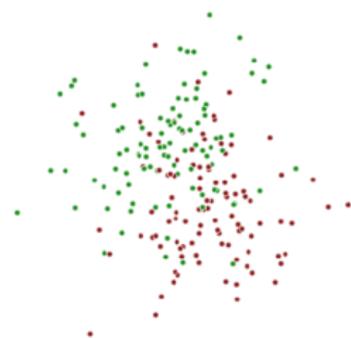


“Неустойчивость” деревьев решений

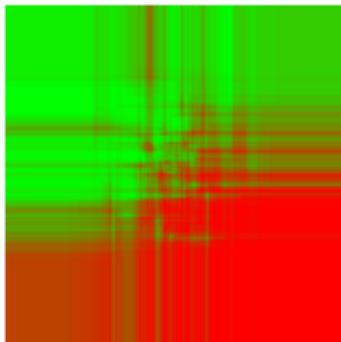
- ▶ Незначительные изменения в данных приводят к значительным изменениям в топологии дерева



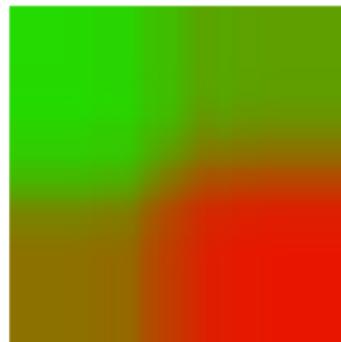
Random Forest. Пример



(a) Original data

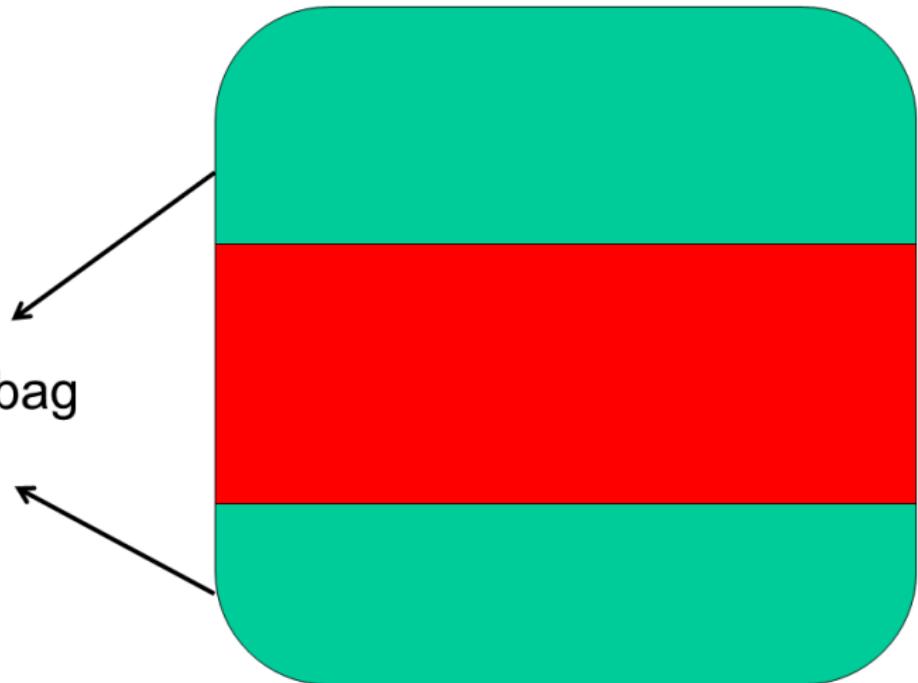


(b) RF (50 Trees)

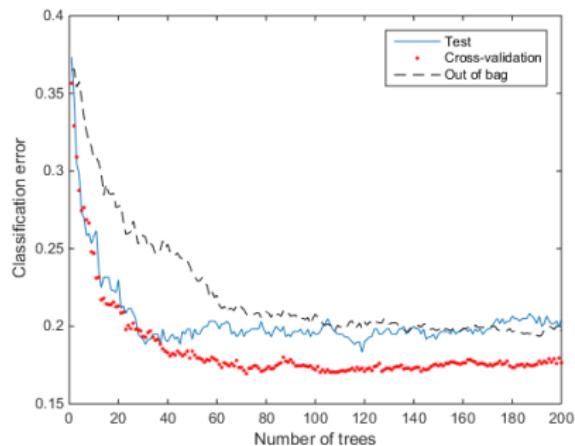


(c) RF (2000 Trees)

Out of bag samples



Random Forest & overfitting



- ▶ Не переобучается с ростом числа алгоритмов

Random Forest. Итоги

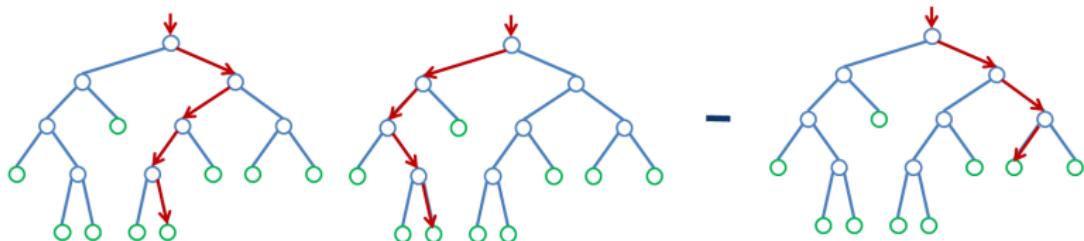
- ✓ Алгоритм прост
- ✓ Не переобучается
- ✓ Хорошо параллелируется
- ✓ Не требует сложной настройки параметров
- ✓ Не требует нормализации данных (деревья решений)
- ✗ Модели получаются большие и неинтерпретируемые
- ✗ Плохо работает с полиномиальными зависимостями.
- ✗ “Медленно” работает для большого объема данных.

Random Forest - не “серебренная” пуля

Предсказываем кто выживет на лодке

The screenshot shows the Kaggle website interface. At the top, there is a dark header bar with the 'kaggle' logo on the left, followed by navigation links: Host, Competitions, Jobs, Community, Sign up, and Login. Below the header, there is a large image of the RMS Titanic sinking. To the right of the image, the text 'Knowledge • 1,898 teams' is displayed. The main title of the competition is 'Titanic: Machine Learning from Disaster'. Below the title, the start date is listed as 'Fri 28 Sep 2012' and the end date as 'Thu 31 Dec 2015 (9 months to go)'. On the left side, there is a sidebar with a 'Dashboard' section containing links for Home, Data, and Make a submission. Another section titled 'Information' contains links for Description, Evaluation, Rules, Prizes, Frequently Asked Questions, Further Reading / Watching, and Getting Started With Excel. The main content area on the right is titled 'Competition Details' and includes links for 'Get the Data' and 'Make a submission'. The main description text reads: 'Predict survival on the Titanic (using Excel, Python, R, and Random Forests)'. It describes the tragic sinking of the RMS Titanic on April 15, 1912, where 1502 passengers and crew lost their lives, leading to better safety regulations for ships.

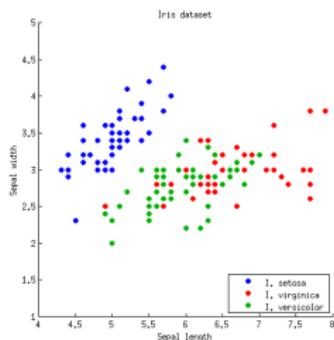
Extremely Randomized Trees



- ▶ Проверяем только один предикат на фичу
- ▶ Используем все фичи для каждого дерева
- ▶ Работает в разы быстрее

Perfect random trees ensembles

- ▶ Быстрое построение деревьев
- ▶ Качество сопоставимо с RF
- ▶ Работает только для задачи классификации



Kaggle.com about Random Forest

This ease of use also makes Random Forests an ideal tool for people without a background in statistics, allowing lay people to produce fairly strong predictions free from many common mistakes, with only a small amount of research and programming.

- ▶ <https://www.kaggle.com/wiki/RandomForests>

Задача

Дано: Имеется набор данных из системы поискового антиспама.

Требуется: Требуется построить классификатор для представленных данных на основе алгоритма Random Forest.

Провести сравнение данного метода с известными алгоритмами классификации. Ответить на вопросы.

Пошаговая инструкция

- Скачать данные и запустить шаблон кода на python
<https://goo.gl/NJjR9D>

```
$ python rf.py -h
$ python rf.py -tr spam.train.txt -te spam.test.txt
```

- Сравнить RF с другими известными алгоритмами классификации.
- Написать функцию подбирающую параметры числа деревьев и процента признаков в деревьях. Построить график.
- Ответить на вопрос: Почему качество классификации для класса spam выше чем для класса notspam?



Вопросы

