



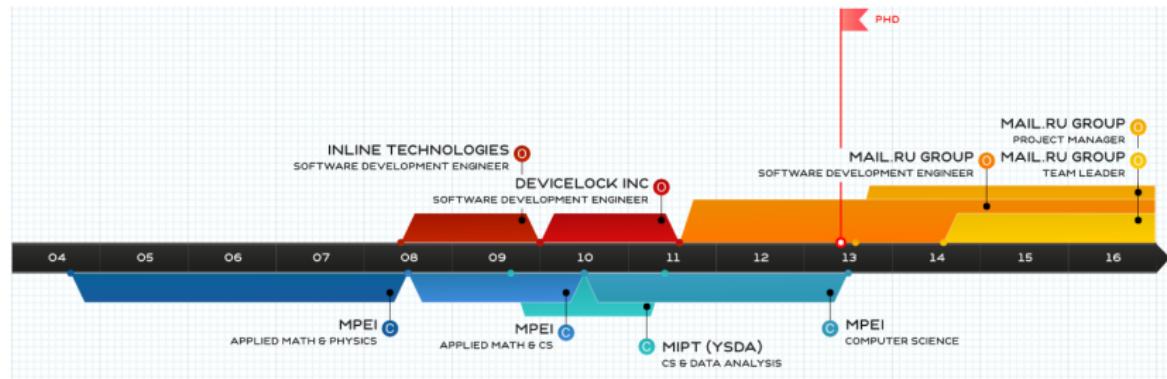
ТЕХНОСФЕРА

Лекция 1 Алгоритмические композиции Начало

Владимир Гулин

10 февраля 2018 г.

Владимир Гулин



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

тел.: +7 (915) 416-95-75

Структура курса

1. Ансамбли. Стохастические модели
2. Ансамбли. Бустинг^H
3. Ансамбли. Кульминация и развязка
4. Рубежный контроль
5. Вопросы построения обучающих выборок ^H
6. Частичное обучение ^H
7. Рубежный контроль
8. Рекомендательные системы 1 ^H
9. Рекомендательные системы 2 ^H
10. Контентные рекомендательные системы ^H
11. Рубежный контроль
12. Машинное обучение в online рекламе 1.
13. Машинное обучение в online рекламе 2. ^H
14. Машинное обучение на больших данных. ^H
15. Рубежный контроль
16. Защита курсового проекта

Контроль знаний

Правила игры

6 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!!!

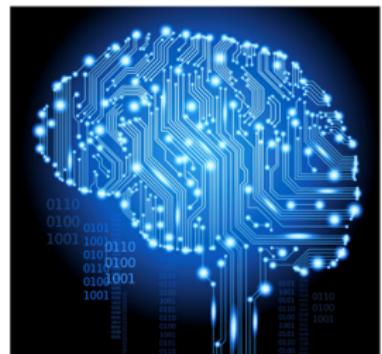
Немного о структуре курса

Что такое система машинного обучения?

Данные



Алгоритм обучения

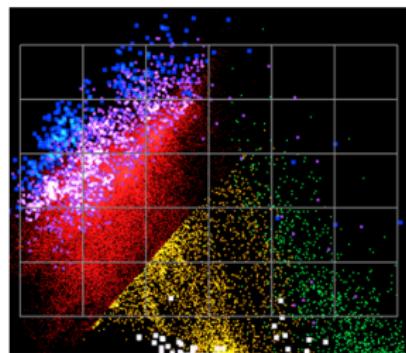
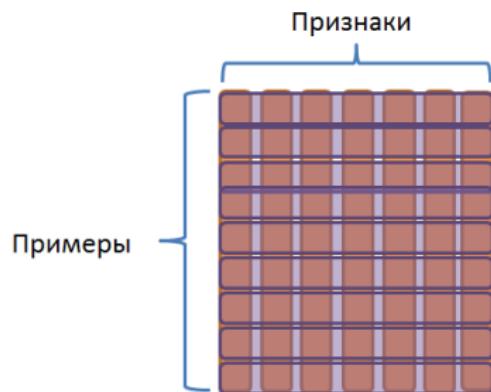


В чем сложность?

#	△pub	Team Name	Kernel	Team Members	Score	Entries	Last
1	—	Роман Васильев			0.31691	34	10mo
2	▲ 1	Vladimir Bugaevsky			0.30655	9	10mo
3	▼ 1	Timur Mubarakshin			0.30516	13	10mo
4	—	Михаил Постников			0.30497	34	10mo
5	▲ 1	ralina			0.30466	21	10mo
Baseline					0.30175		
6	▲ 2	PashaAdamenko			0.29895	34	10mo
7	▼ 2	Alexey Katsman [ITMO]			0.29232	22	9mo
8	▲ 1	Dana Zlochevskaya			0.29154	15	10mo
9	▼ 2	Adel Lesuchevskiy [ITMO]			0.28924	7	9mo
10	—	sergei.grabalin			0.28290	42	10mo
Simple baseline					0.24552		
11	—	(ITMO)Амир Муратов			0.14487	1	9mo
12	—	Admiralskiy Yury			0.13538	1	10mo

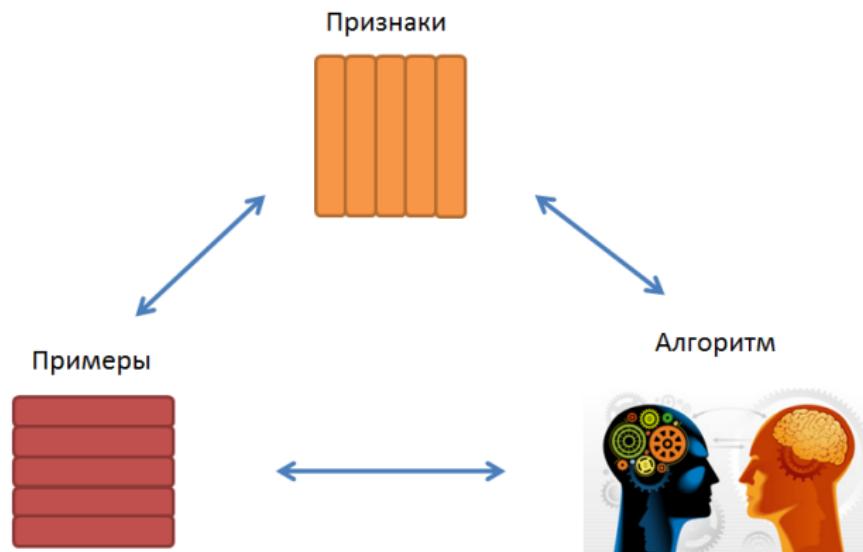
Немного о структуре курса

Что такое данные?



Немного о структуре курса

Какая связь?



А что важнее?

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

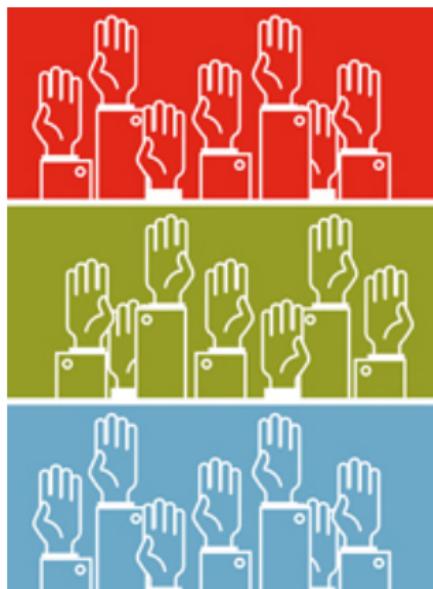
$$h(\mathbf{x}) = \frac{1}{T} \sum_{i=1}^T b_i a_i(\mathbf{x}), \quad b_i \in \mathcal{R}$$



Смесь экспертов

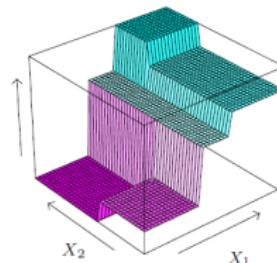
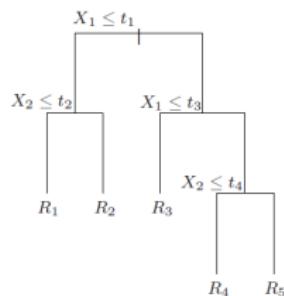
Mixture of Experts

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



Топологические композиции

Деревья решений

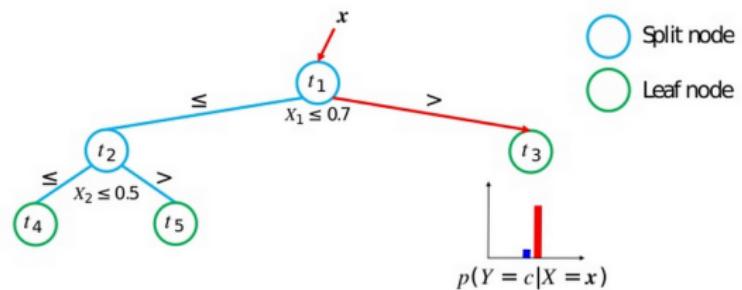
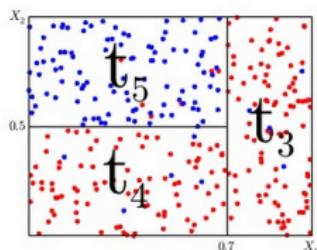


Нужно понимать:

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

Топологические композиции

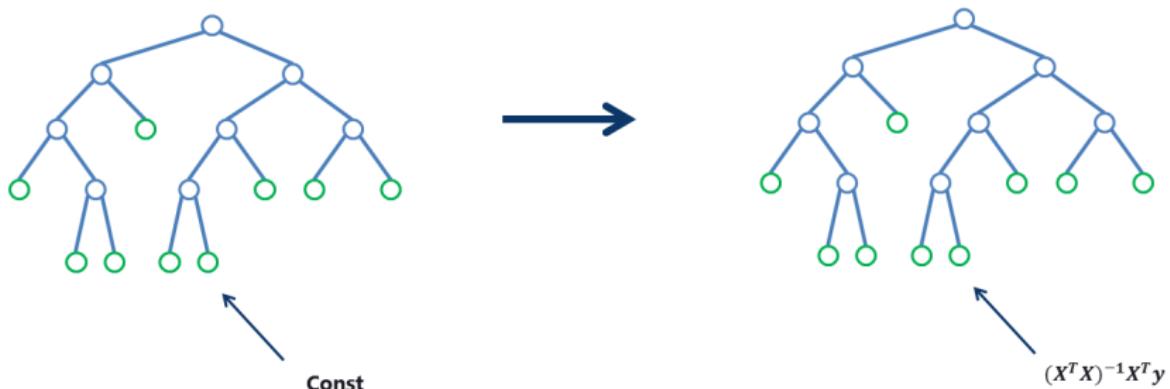
Деревья решений (принцип работы)



Топологические композиции

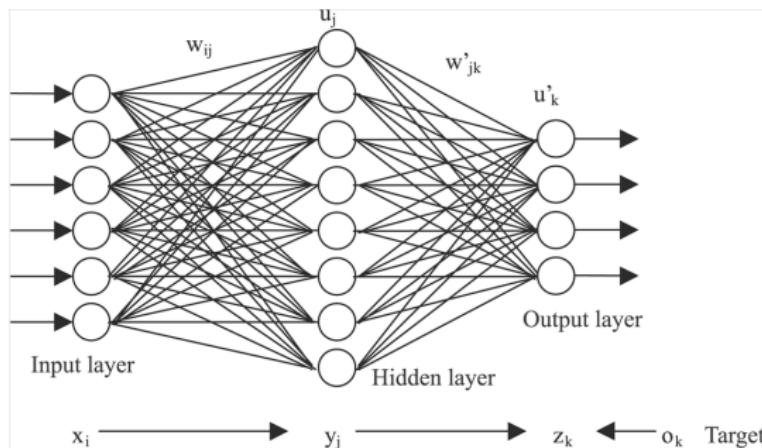
Модельные деревья решений (Model decision trees)

- ▶ 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



Текущее положение дел

How to win a Kaggle competition?



Anthony Goldbloom

“According to Anthony, in the history of Kaggle competitions, there are only two Machine Learning approaches that win competitions:
Handcrafted & Neural Networks.”

Neural nets

Где нейронные сети побеждают?

- ▶ Computer Vision (ImageNet)
- ▶ Speech recognition



Decision trees compositions (Handcrafted feature engineering)

Где побеждают ансамбли деревьев решений?

- ▶ Recomendation systems (Netflix Prize 2009)
- ▶ Learning to rank (Yahoo Learning to rank challenge 2010)
- ▶ Crowdflower Search Results Relevance (2015)
- ▶ Avito Context Ad Clicks (2015)
- ▶ Везде :)



“As long as Kaggle has been around, Anthony says, it has **almost always** been **ensembles of decision trees that have won competitions.**”

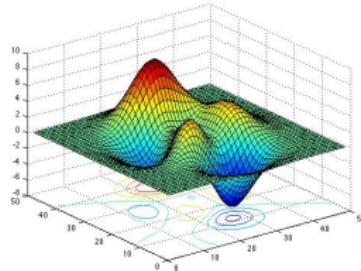
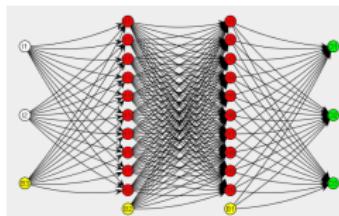
Кто использует ансамбли деревьев?

- ▶ Google
- ▶ Bing (Microsoft)
- ▶ Amazon
- ▶ Mail.Ru
- ▶ Yandex
- ▶ CERN
- ▶ Да кто только не использует :)

Взгляд с точки зрения функционального анализа

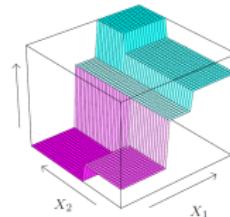
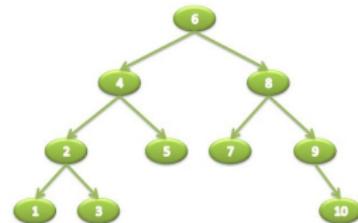
Гладкие функции

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



Кусочно-постоянные функции

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



Однако, и то и другое это ансамбли!

Методы построения алгоритмических композиций

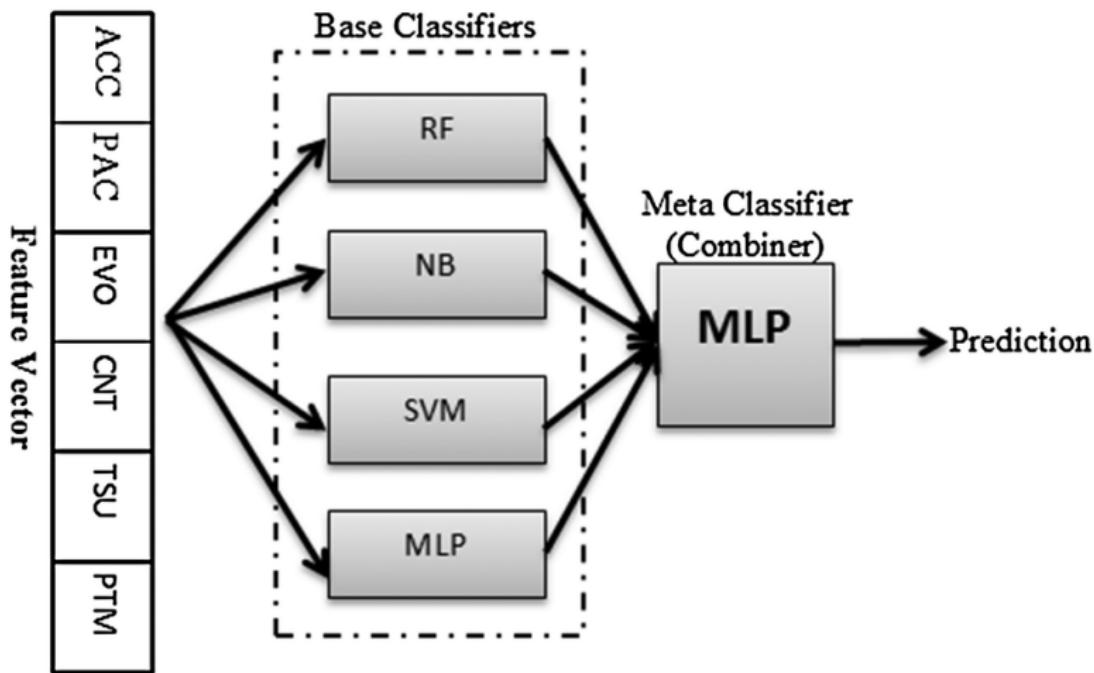
Стохастические методы

Бустинг

Мета алгоритмы. Stacking & Blending

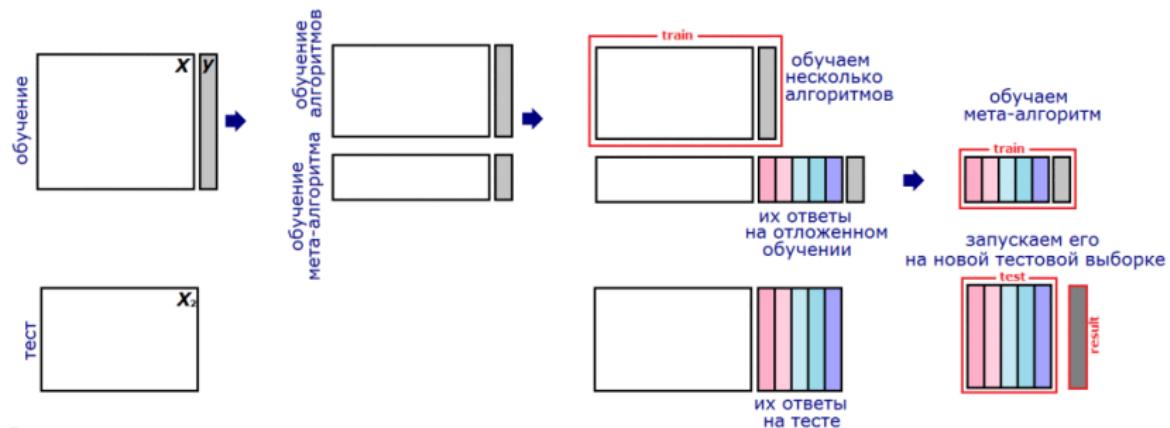
Stacking

Мета-алгоритм

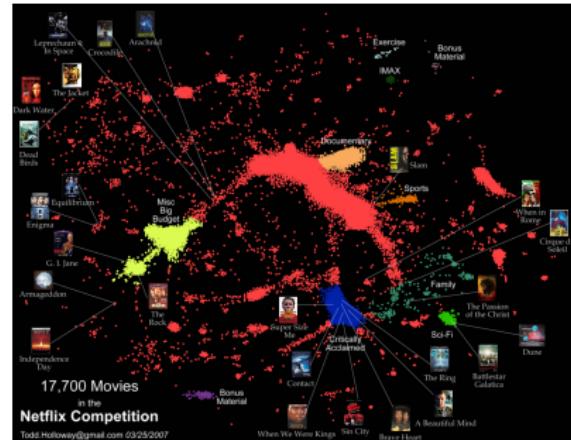


Blending

Классическая схема



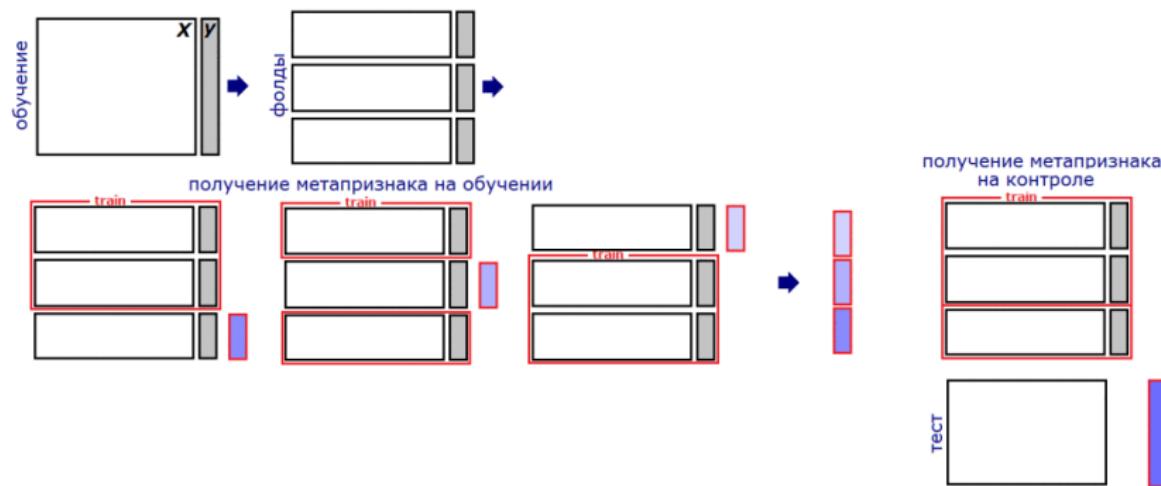
Netflix Challenge



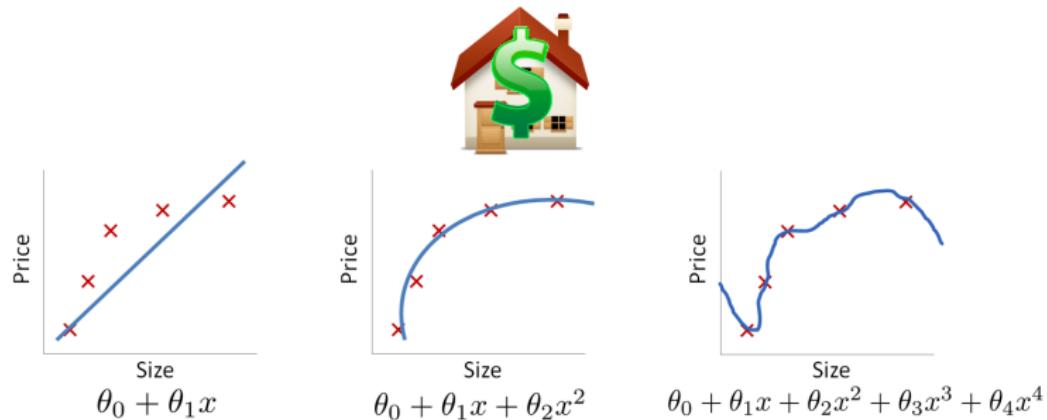
- ▶ Задача предсказания оценки фильму
 - ▶ ПФ - 1000000\$
 - ▶ Победил Stacking на “зоопарке” алгоритмов

Stacking

Классическая схема



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

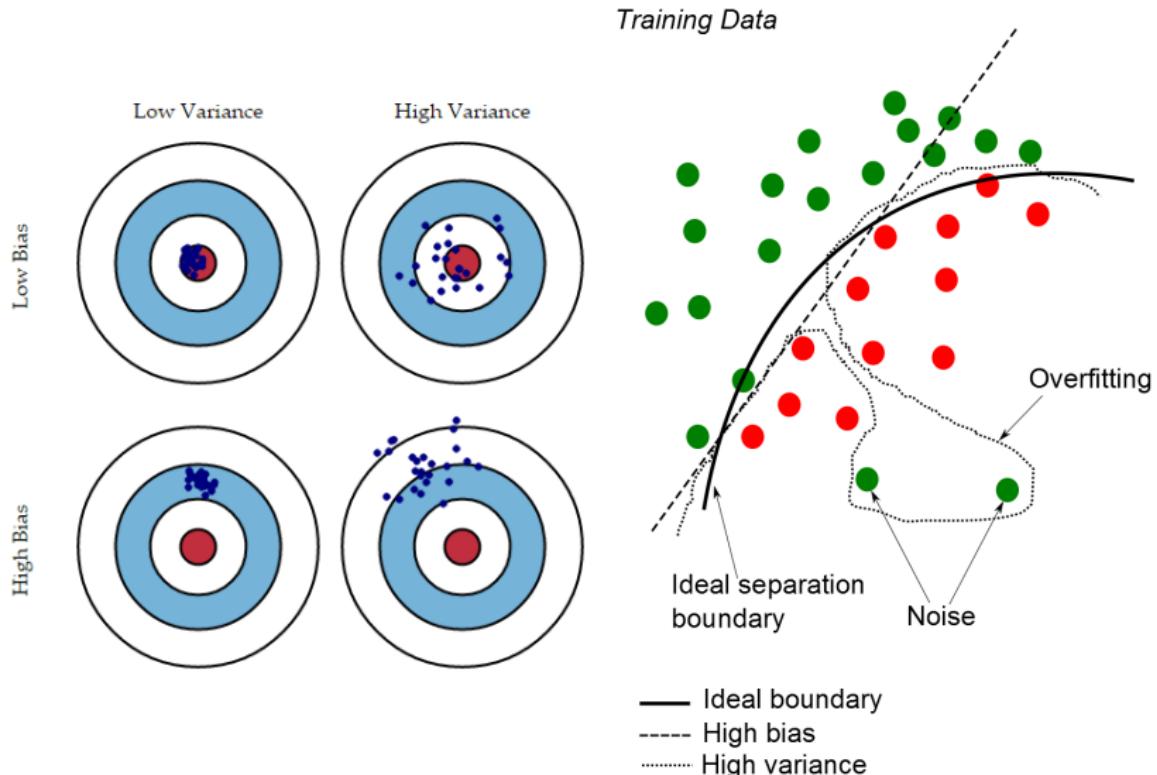
Это можно переписать в виде

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

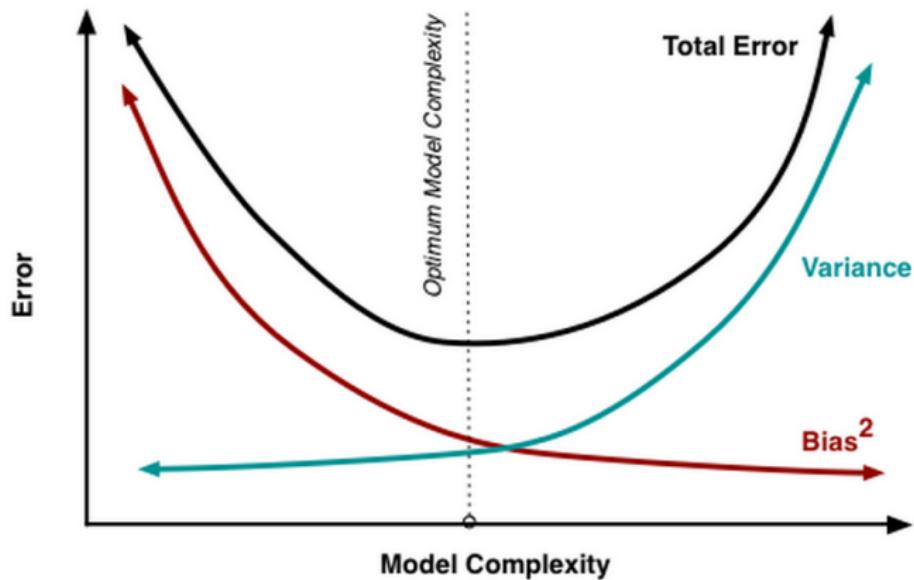
т.к. f -детерминирована, а ϵ и $h(\mathbf{x}_0)$ - независимы

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

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.

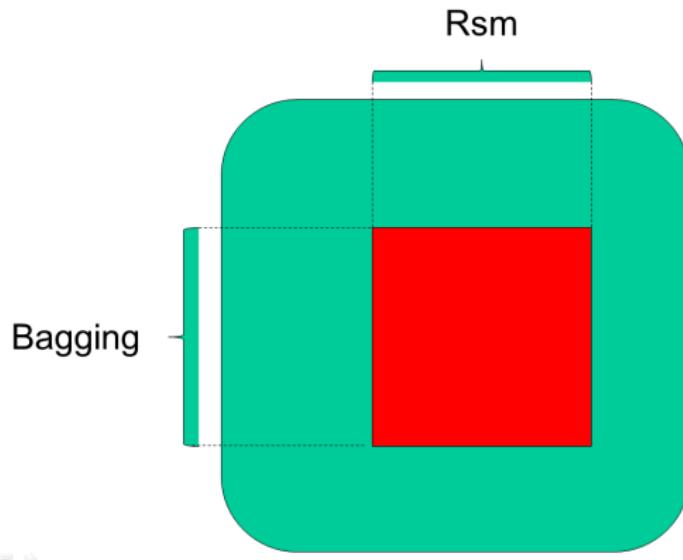
Благодаря описанным стратегиям добиваемся максимального различия между базовыми алгоритмами.

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.

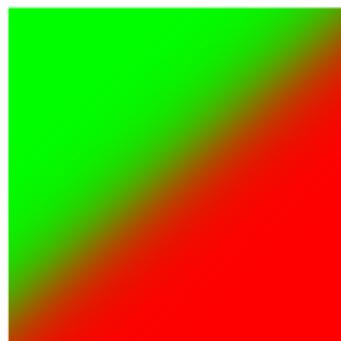
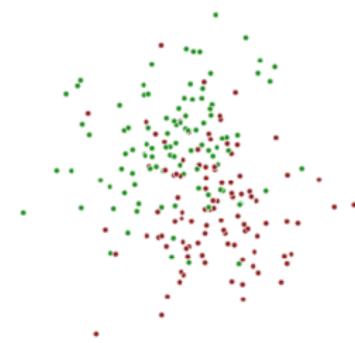


“Неустойчивость” деревьев решений

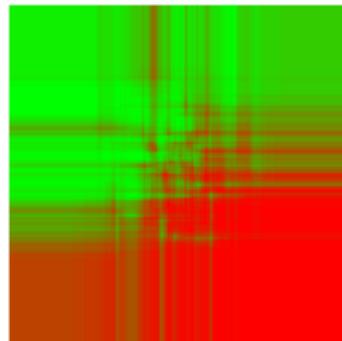
- ▶ Незначительные изменения в данных приводят к значительным изменениям в топологии дерева



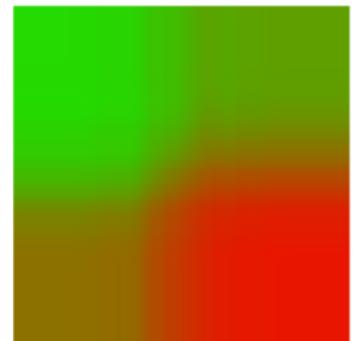
Random Forest. Пример



(a) Original data

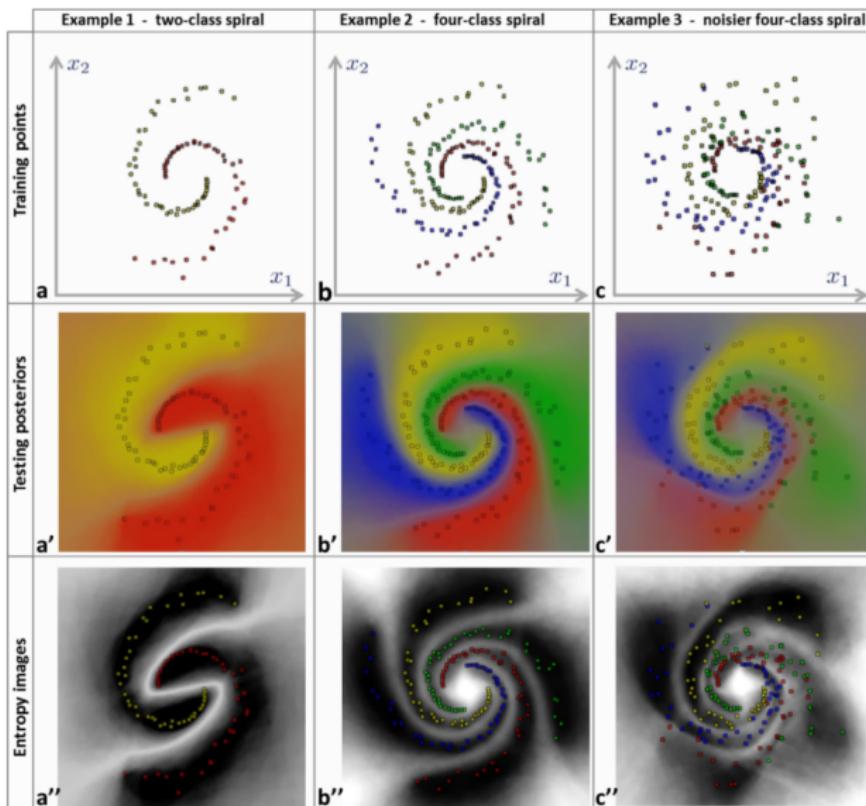


(b) RF (50 Trees)

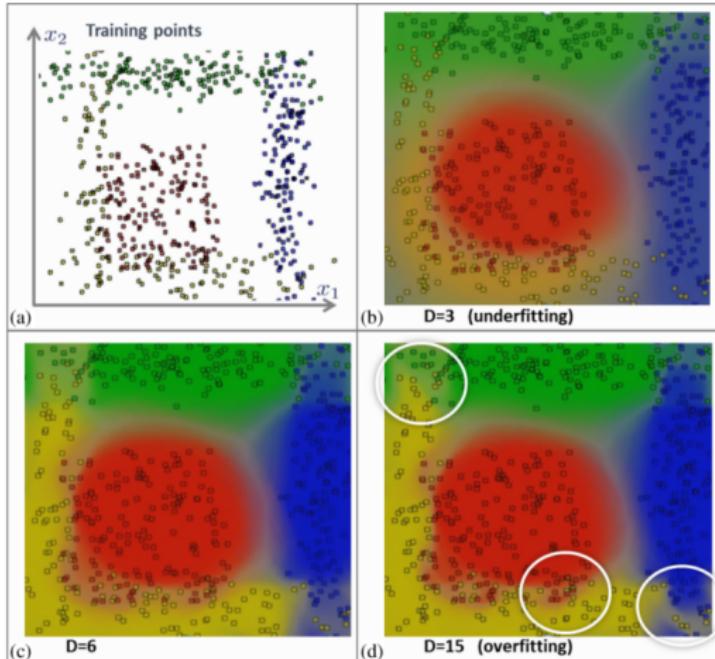


(c) RF (2000 Trees)

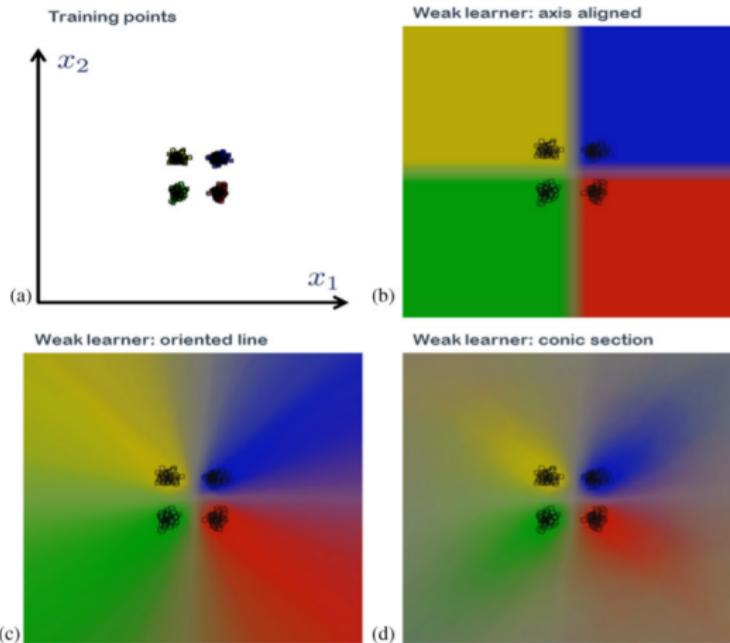
Multiple Classes and Training Noise



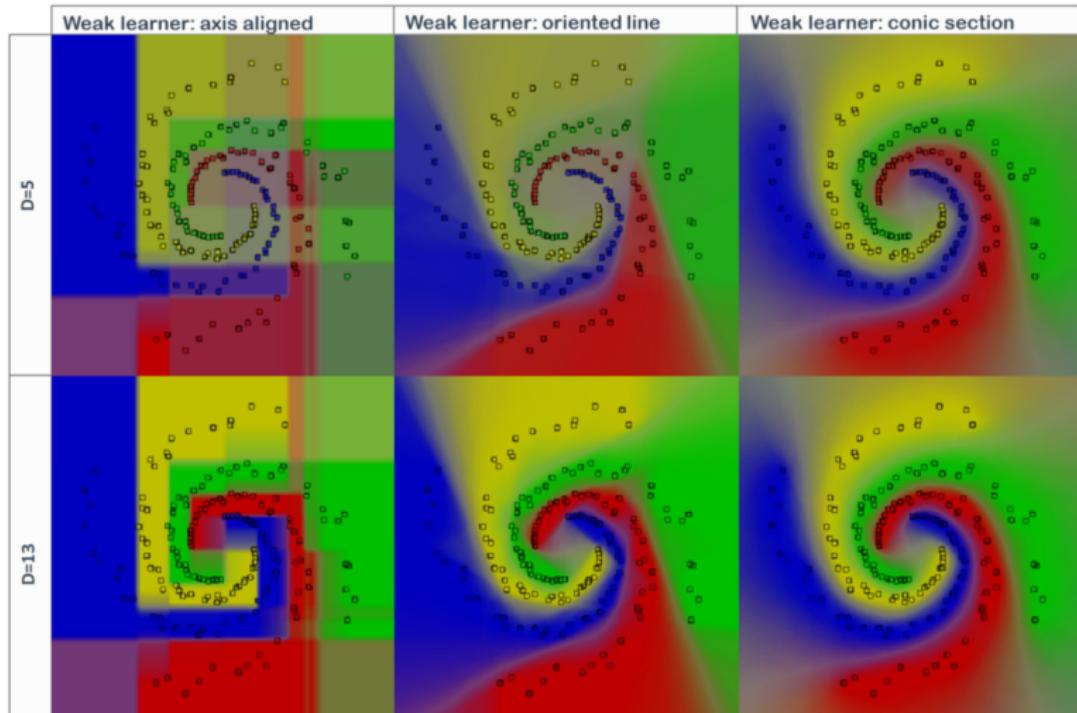
Effect of tree depth



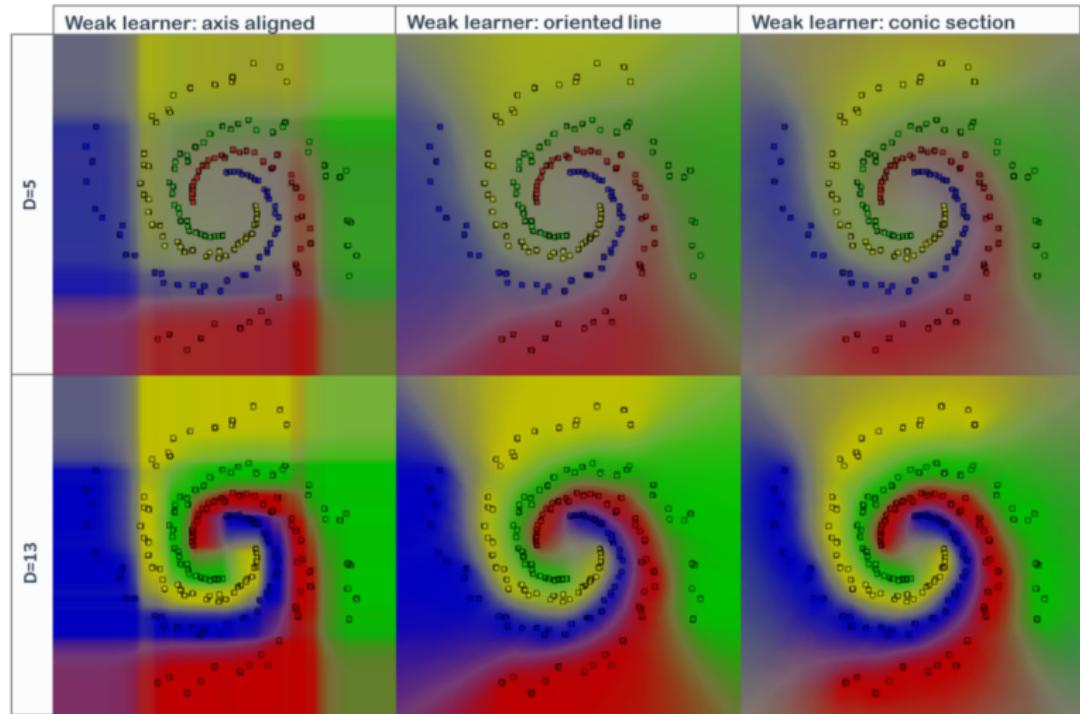
Effect of week learner



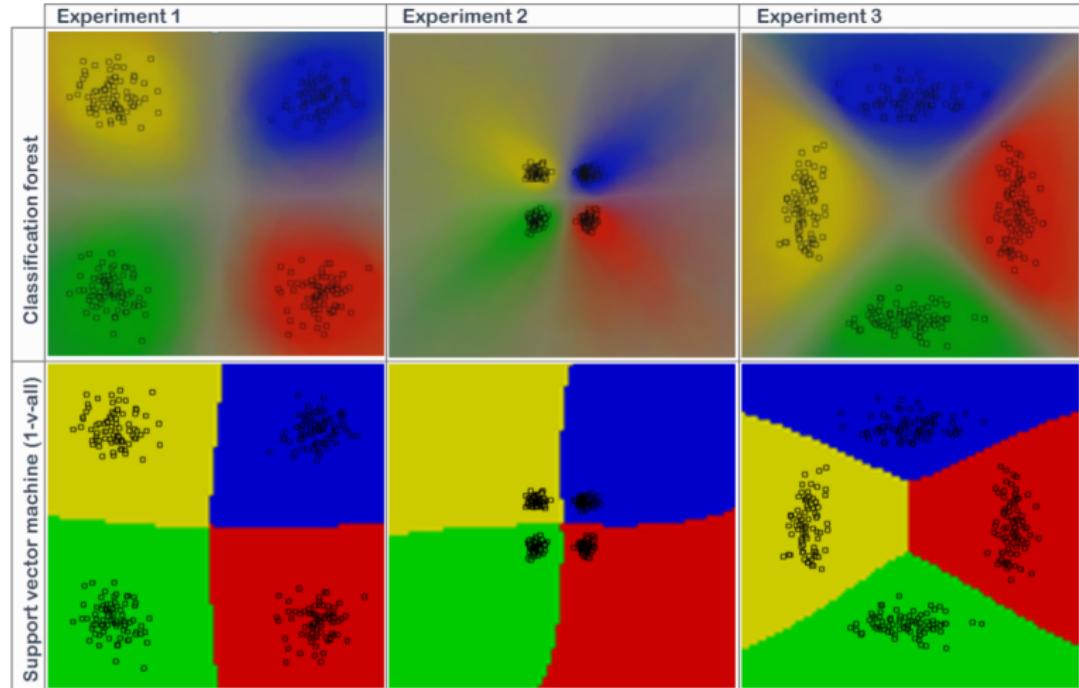
Effect of week learner



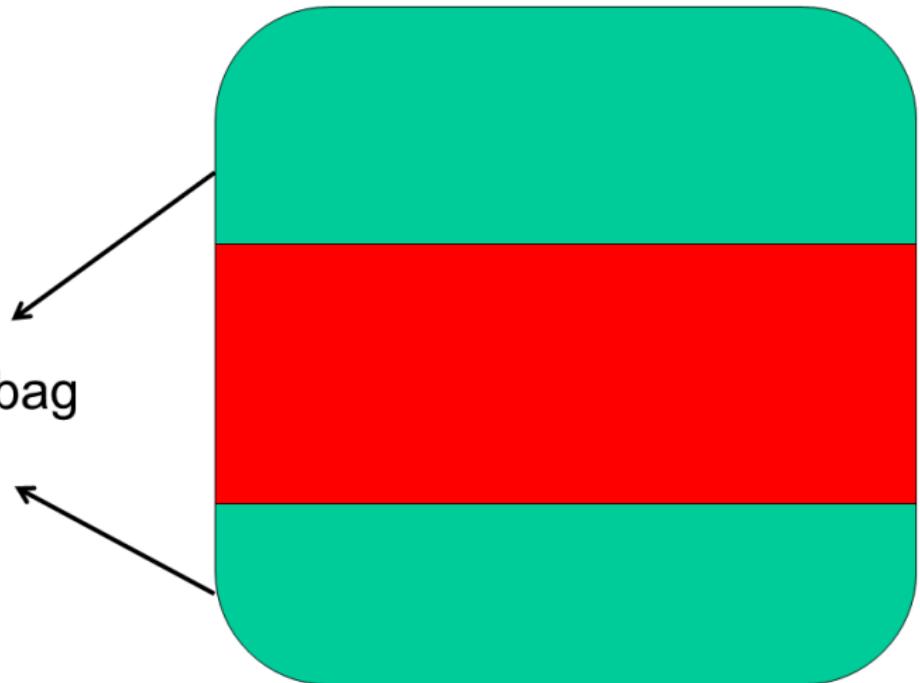
Effect of Randomness



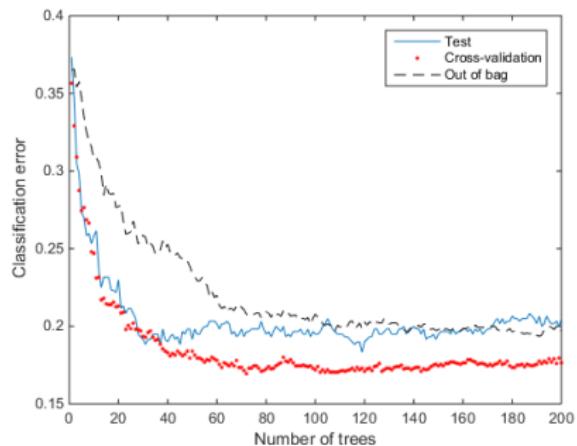
Сравнение с SVM



Out of bag samples

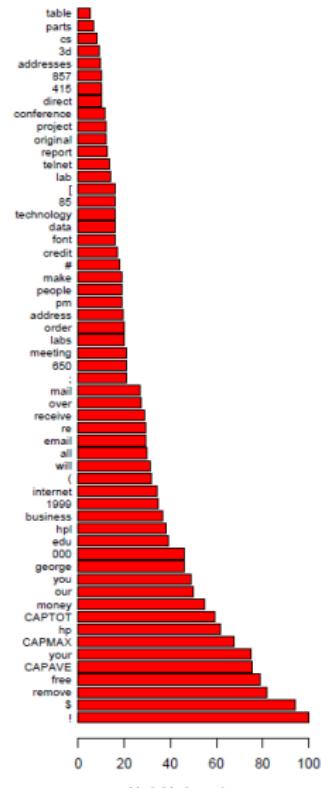


Random Forest & overfitting

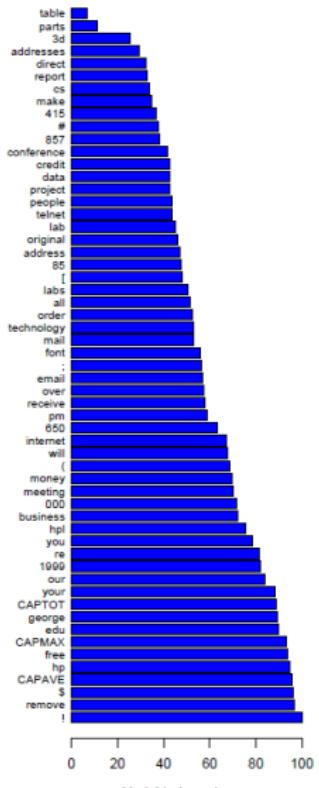


- ▶ Не переобучается с ростом числа алгоритмов

Feature importance

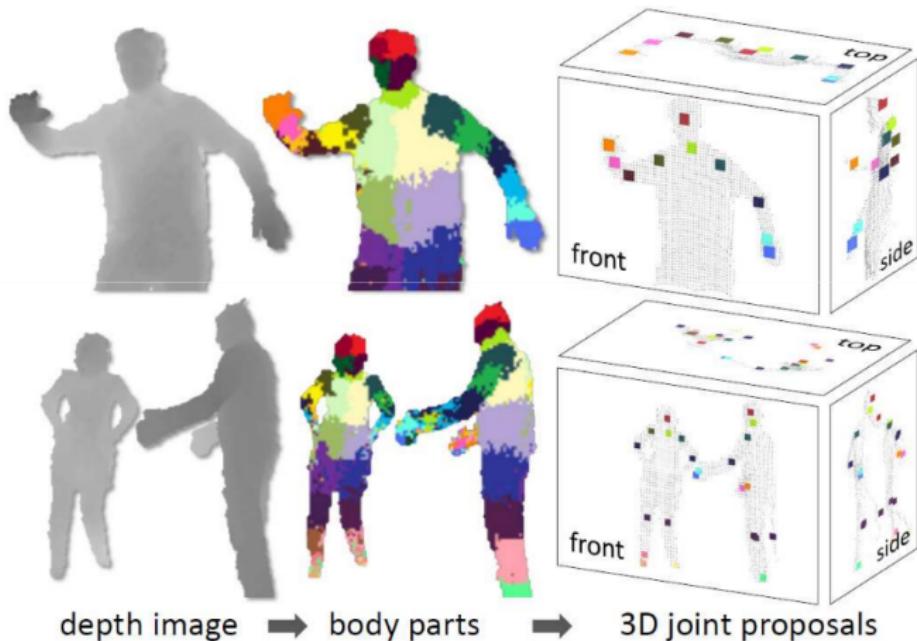


Variable Importance



Variable Importance

Random Forest and the Kinect



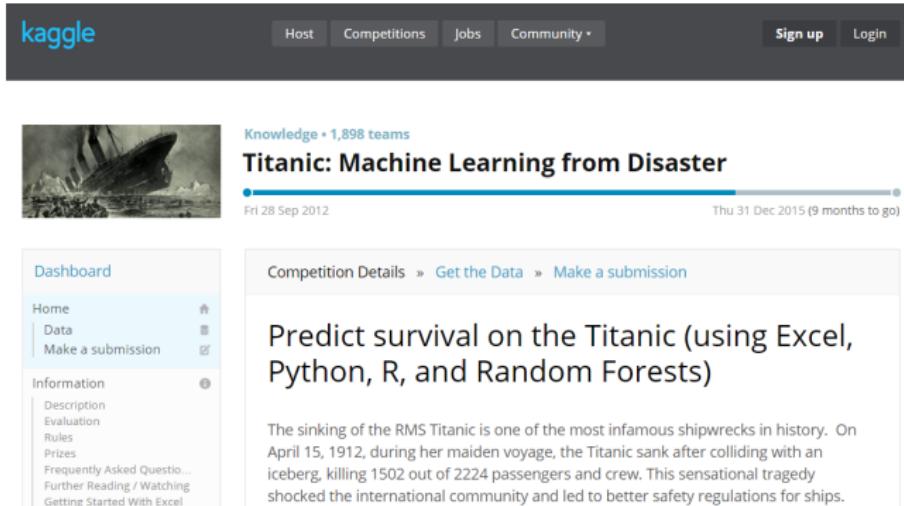
Определение части тела для каждого пикселя по карте глубины, которую возвращают датчики kinect.

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 navigation bar with the 'kaggle' logo, 'Host', 'Competitions', 'Jobs', 'Community', 'Sign up', and 'Login' buttons. Below the navigation bar, there is a banner for the 'Titanic: Machine Learning from Disaster' competition. The banner features a black and white illustration of the Titanic sinking, the text 'Knowledge • 1,898 teams', the competition name, and a timeline from 'Fri 28 Sep 2012' to 'Thu 31 Dec 2015 (9 months to go)'. The main content area is divided into two sections: a sidebar on the left and a main content area on the right. The sidebar contains a 'Dashboard' section with links for 'Home', 'Data', 'Make a submission', 'Information' (with links for 'Description', 'Evaluation', 'Rules', 'Prizes', 'FAQ', 'Further Reading / Watching', and 'Getting Started With Excel'), and a 'Competition Details' section with links for 'Competition Details', 'Get the Data', and 'Make a submission'. The main content area displays the competition title 'Titanic: Machine Learning from Disaster' and a large call-to-action button that says 'Predict survival on the Titanic (using Excel, Python, R, and Random Forests)'. Below this button, there is a descriptive text about the sinking of the Titanic and its historical significance.

Titanic: Machine Learning from Disaster

Knowledge • 1,898 teams

Fri 28 Sep 2012 — Thu 31 Dec 2015 (9 months to go)

Dashboard

- Home
- Data
- Make a submission

Information

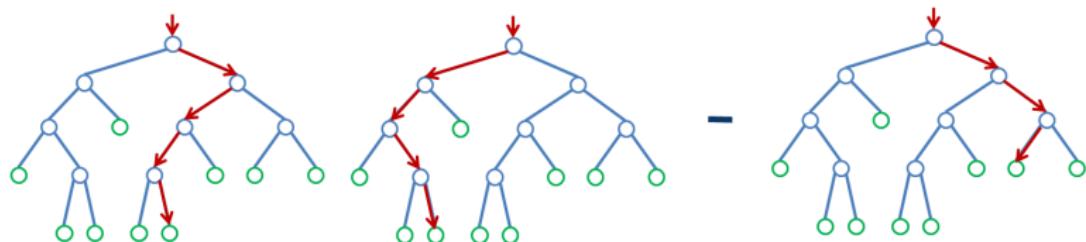
- Description
- Evaluation
- Rules
- Prizes
- Frequently Asked Questions
- Further Reading / Watching
- Getting Started With Excel

Competition Details » Get the Data » Make a submission

Predict survival on the Titanic (using Excel, Python, R, and Random Forests)

The sinking of the RMS Titanic is one of the most infamous shipwrecks in history. On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew. This sensational tragedy shocked the international community and led to better safety regulations for ships.

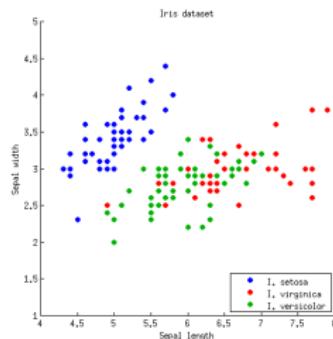
Extremely Randomized Trees



- ▶ 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

1. 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
```

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

Вопросы

