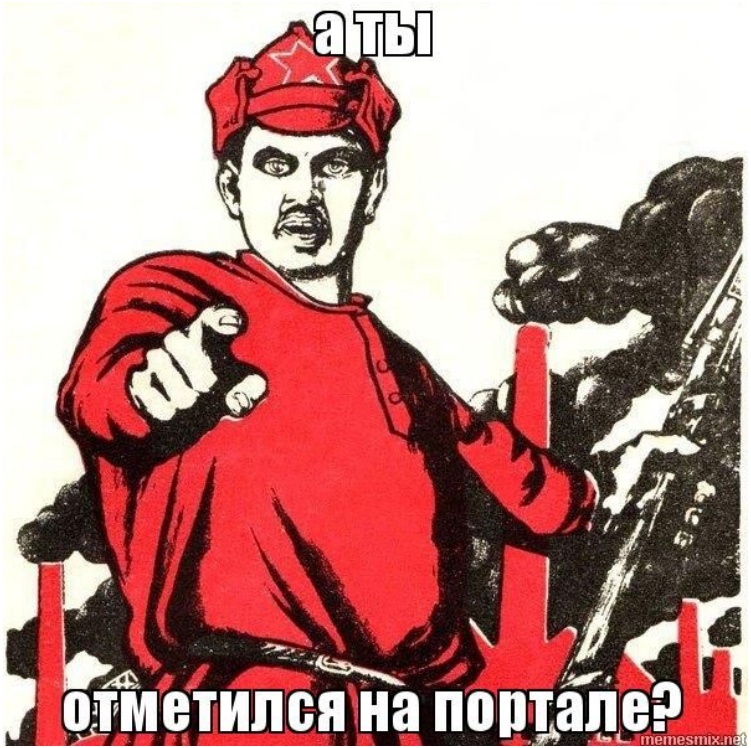
Python

Лекция 3: Jupyter и все, все, все...

Отметься на портале!



План занятия

1. Немного о jupyter
2. Numpy
3. Pandas
4. Flask
5. Введение в ML
6. Домашнее задание
7. Collections
8. Functools
9. Intertools

Кахуууут! Вспоминаем! Садимся ближе!

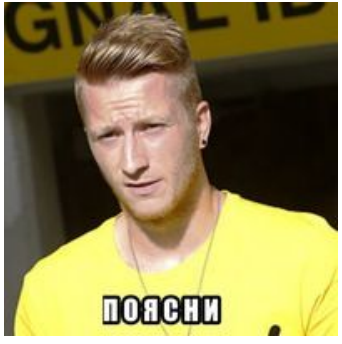
kahoot.it

Подготовимся к лекции

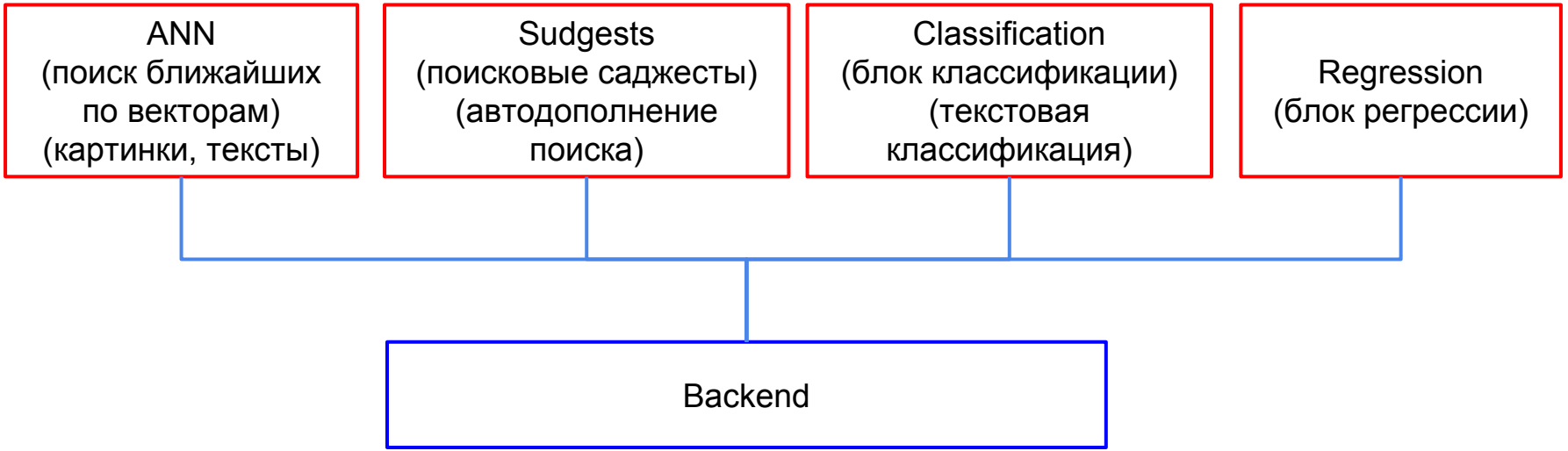
1. Переходим в директорию вашего форка
2. Открываем в ней терминал
3. git checkout master
4. git pull upstream master
5. git push -u origin master

Про проекты

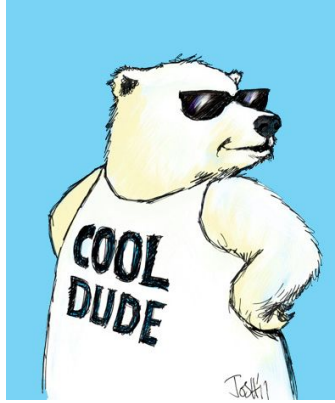
1. Объединиться в команды - кто знает с кем пишите нам, кто нет, пишите тоже, объединим
2. Каждой команде ставится ментор (один из нас)
3. Команде нужно определить тему проекта (говорим далее), согласовать с ментором. Изменение темы возможно, но не должно пересекаться с другими командами (реализация на вашей совести)
4. Со след занятия дз привязаны к проектам (НЕ так, что один пишет, остальные отдыхают - поясним перед дз)



Архитектура проекта (примерная)



Примеры проектов



Примеры проектов

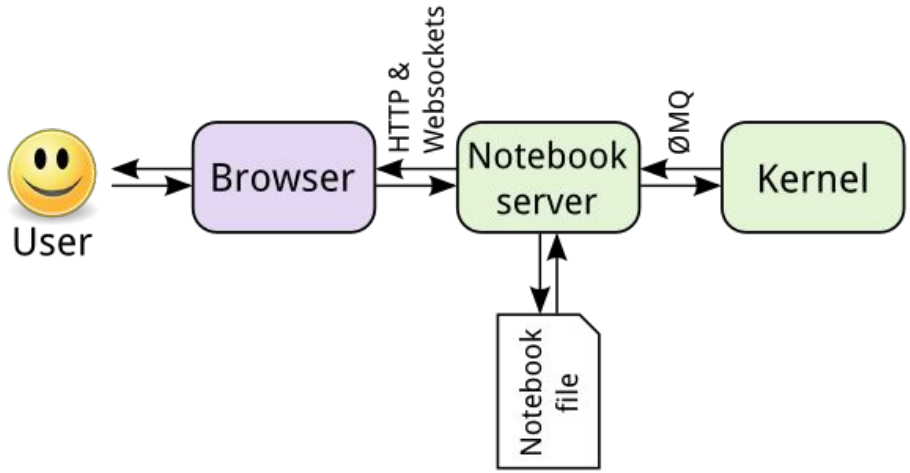
- Поваренная книга:
  - классификация по кухням мира
  - саджесты поиска блюда
  - регрессия рецепта на калорийность
  - классификация по типам блюда (завтрак, обед, ужин)
  - поиск похожего блюда по картинке, по текстовому описанию.
- Аналог маркета (подзадачи аналогичны поваренной книге).
- Система авторизации по фотографии (+\- нужно знание нейронок.)
- Сервис сводки новостей о заданных компаниях и их котировках (парсинг Ленты, классификация новостей, выделение интентов, построение регрессионной модели)
- Сервис гид-по-кино (парсинг кинопоиска и тд. классификация отзывов, простые рекомендации и тд.)
- Сервис гид-по-москве (парсинг куда-го как один из вариантов)
- сервис чат-бот с подсказками по программированию (парсинг stackoverflow, классификация запросов, ANN-поиск по LSH - как один из вариантов)
- Парсинг cian и headhunter (и отсюда какое нибудь аналитическое решение типа если вы столько то зарабатываете и тут работаете, то вам можно найти вот эти квартиры, тут можно quad-tree уже строить и модифицированный ANN)
- *Оптимизация логистики (хардово)
- В конце - публичная защита с рассказом кто, что и как сделал!

Jupyter notebook

Писать код в IDE или терминале

Писать код и выполнять его прямо в браузере

Jupyter notebook. Основы




```
{
 "cells": [
  {
   "cell_type": "code",
   "execution_count": null,
   "metadata": {},
   "outputs": [],
   "source": []
  }
 ],
 "metadata": {
  "kernelspec": {
   "display_name": "Python 3",
   "language": "python",
   "name": "python3"
  },
  "language_info": {
   "codemirror_mode": {
    "name": "ipython",
    "version": 3
   },
   "file_extension": ".py",
   "mimetype": "text/x-python",
   "name": "python",
   "nbconvert_exporter": "python",
   "pygments_lexer": "ipython3",
   "version": "3.6.5"
  }
 },
 "nbformat": 4,
 "nbformat_minor": 2
}
```

Jupyter notebook. Magics, shortcuts

Keyboard shortcuts

The Jupyter Notebook has two different keyboard input modes. **Edit mode** allows you to type code or text into a cell and is indicated by a green cell border. **Command mode** binds the keyboard to notebook level commands and is indicated by a grey cell border with a blue left margin.

Command Mode (press `Esc` to enable)

Edit Shortcuts

- `F`: find and replace
- `Ctrl-Shift-F`: open the command palette
- `Ctrl-Shift-P`: open the command palette
- `Enter`: enter edit mode
- `P`: open the command palette
- `Shift-Enter`: run cell, select below
- `Ctrl-Enter`: run selected cells
- `Alt-Enter`: run cell and insert below
- `Y`: change cell to code
- `M`: change cell to markdown
- `R`: change cell to raw
- `1`: change cell to heading 1
- `2`: change cell to heading 2
- `3`: change cell to heading 3
- `4`: change cell to heading 4
- `5`: change cell to heading 5
- `6`: change cell to heading 6
- `K`: select cell above
- `Up`: select cell above
- `Down`: select cell below
- `J`: select cell below
- `Shift-K`: extend selected cells above
- `Shift-Up`: extend selected cells above
- `Shift-Down`: extend selected cells below
- `Shift-J`: extend selected cells below
- `A`: insert cell above
- `B`: insert cell below
- `X`: cut selected cells
- `C`: copy selected cells
- `Shift-V`: paste cells above
- `V`: paste cells below
- `Z`: undo cell deletion
- `D`, `D`: delete selected cells
- `Shift-M`: merge selected cells, or current cell with cell below if only one cell is selected
- `Ctrl-S`: Save and Checkpoint
- `S`: Save and Checkpoint
- `L`: toggle line numbers
- `O`: toggle output of selected cells
- `Shift-O`: toggle output scrolling of selected cells
- `H`: show keyboard shortcuts
- `I`, `I`: interrupt the kernel
- `0`, `0`: restart the kernel (with dialog)
- `Esc`: close the pager
- `Q`: close the pager
- `Shift-L`: toggles line numbers in all

[Magics]

В ноутбук

NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

!Строгая типизация (int, float, unicode, ...)

```
np.array(["строка", 5])
```

```
array(['строка', '5'], dtype='<U6')
```

```
two_dimarray.tolist()
```

```
[[5, 4], [4, 6]]
```

```
two_dimarray = np.array([[5, 4], [4, 6]])
two_dimarray.shape
```

```
(2, 2)
```

```
two_dimarray = np.array([[5, 4], [4, 6]])
len(two_dimarray)
```

```
2
```

В ноутбук

NumPy. Базовые операции.

1. Преобразование формы и размера

```
example = np.array([5, 4, 7, 8])
print(example)
print(example.reshape(2, 2))
print(example.reshape(2, 2).flatten())
```

```
[5 4 7 8]
[[5 4]
 [7 8]]
[5 4 7 8]
```

Как сделать flatten с помощь reshape? Напишите тест

2. Объединение массивов

```
example = np.array([5., 6.])
example = np.append(example, example)
print(example)
example = np.array([[5.], [6.]])
print(np.concatenate((example, example), axis=0))
print(np.concatenate((example, example), axis=1))
print(np.vstack((example, example)))
print(np.hstack((example, example)))
```

```
[5. 6. 5. 6.]
[[5.]
 [6.]
 [5.]
 [6.]]
[[5. 5.]
 [6. 6.]]
[[5.]
 [6.]
 [5.]
 [6.]]
[[5. 5.]
 [6. 6.]]
```

NumPy. Математические операции.

1. Деление, умножение, сложение, вычитание применяемое ко всем элементам (векторная операция)
2. Транспонирование
3. Операции над векторами (над векторами одной размерности)
4. min, max, mean, argmax, argmin, std, var …
5. Сравнение и выбор по значению
6. Векторное умножение

Все просто, все в ноутбуке

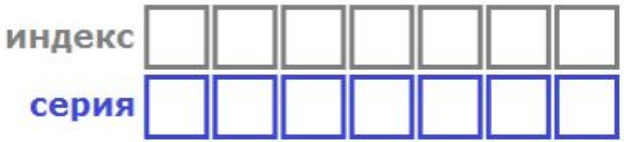
Вопросы по numpy?

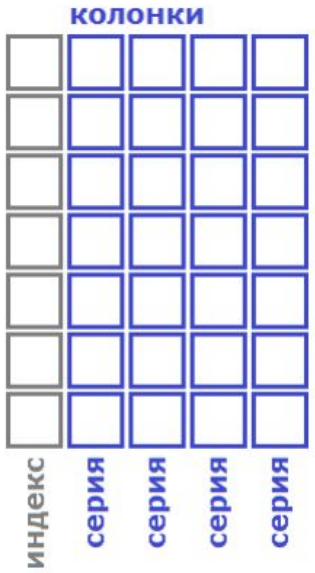
Pandas

Pandas — программная библиотека на языке Python для обработки и анализа данных. Работа pandas с данными строится поверх библиотеки NumPy.

Основные структуры:

- Series pd.Series(....)
- DataFrame pd.DataFrame(...)
- … есть более сложные представления (pd.panel…)





Pandas. Создание датафрейма

DataFrame

	date	number_of_game	day_of_week	v_name	v_league	v_game_number	h_name	h_league	h_game_number	v_score	h_score	length_outs
0	01871054	0	Thu	CL1	na	1	FW1	na	1	0	2	54.0
1	18710505	0	Fri	BS1	na	1	WS3	na	1	20	18	54.0
2	18710506	0	Sat	CL1	na	2	RC1	na	1	12	4	54.0

IntBlock

	0	1	2	3	4	5
0	01871054	0	1	1	0	2
1	18710505	0	1	1	20	18
2	18710506	0	2	1	12	4

ObjectBlock

	0	1	2	3	4
0	Thu	CL1	na	FW1	na
1	Fri	BS1	na	WS3	na
2	Sat	CL1	na	RC1	na

FloatBlock

	0
0	54.0
1	54.0
2	54.0

1. Зачитать откуда угодно

```
data = pd.read_csv("sample.tsv", sep="\t", index=None)
data = pd.read_excel("sample.xls")
data = pd.read_sql("SELECT * FROM table;", conn_string)
data = pd.read_hdf('sample.h5', 'dataframe')
data = pd.read_csv('https://raw.githubusercontent.com/rmcelreath/rethinking/master/data/Howell1.csv', sep=';')
data = pd.read_html("some.html")
```

2. Составить из массива/np.array/dict

```
simple_array = pd.DataFrame(np.array([[4, 5, 6], [1, 2, 3]]), columns=['col1', 'col2', 'col3'])
simple_array = pd.DataFrame([['колонка', 233, True], [15]], columns=['col1', 'col2', 'col3'])
simple_dict = pd.DataFrame(dict([('A',[1., np.nan, 2., 1.]), ('B',[2.2, np.nan, np.nan, 0.0])]))
```

3. Составить из различных структур

```
data = pd.DataFrame({ 'A' : [1., 4., 2., 1.],
'B' : pd.Timestamp('20130102'),
'C' : pd.Series(1,index=list(range(4)),dtype='float32'),
'D' : np.array([3] * 4,dtype='int32'),
'E' : pd.Categorical(["test","train","test","train"]),
'F' : 'foo' }, index=pd.period_range('Jan-2000', periods=4,
freq='M'))
```

Pandas. Осмотр датафрейма. Базовые операции

Посмотреть на записи в датафрейме

```
sample_data.head(34)
sample_data.tail(43)
sample_data.sample(1)
```

Посмотреть на свойства датафрейма

```
titanik_excel.describe()
titanik_excel.info()
titanik_excel.columns
titanik_excel.shape
```

Сумма, среднее, подсчет уникальных элементов

```
sample_data['ColName'].sum()
sample_data.ColName.mean()
sample_data['Colname'].unique()
sample_data['Colname'].value_counts()
```

Ваша команда - ваша парта. Найти:

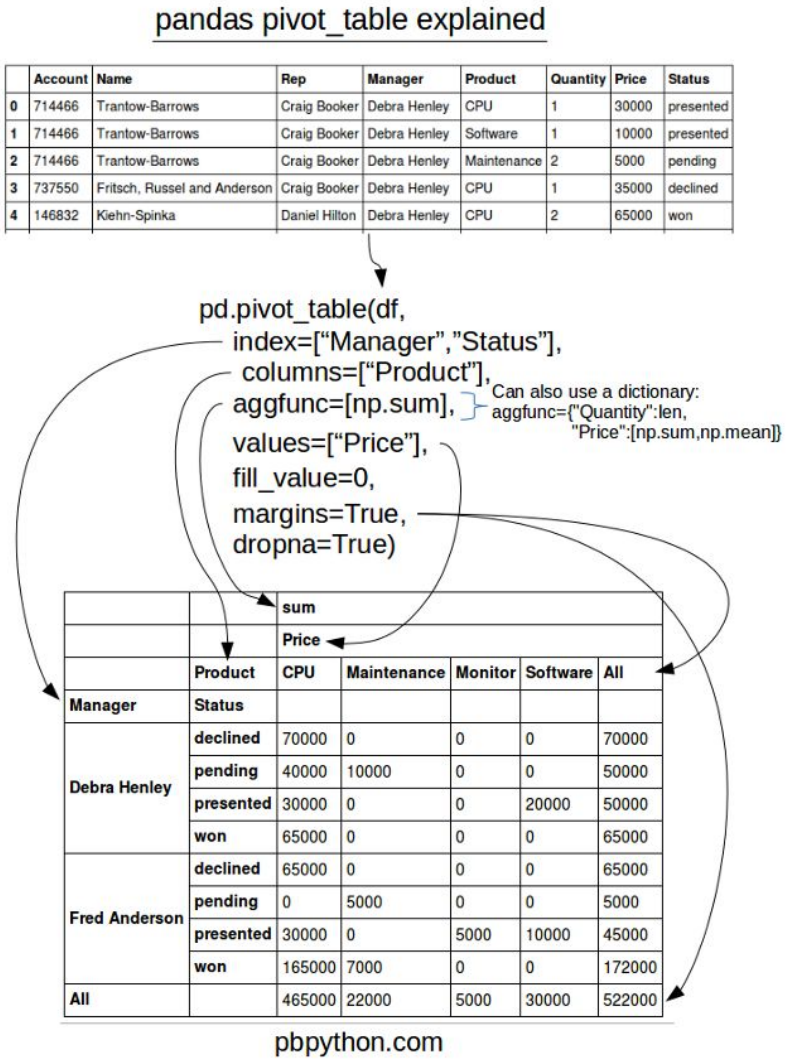
- метод, который сортирует датафрейм
- метод, который позволяет преобразовать тип данных в колонке
- метод, который позволяет получить n-ю строку

2 минуты

Pandas. Группировка

Различные способы группировки:

- groupby()
- pivot_table()
- crosstab()

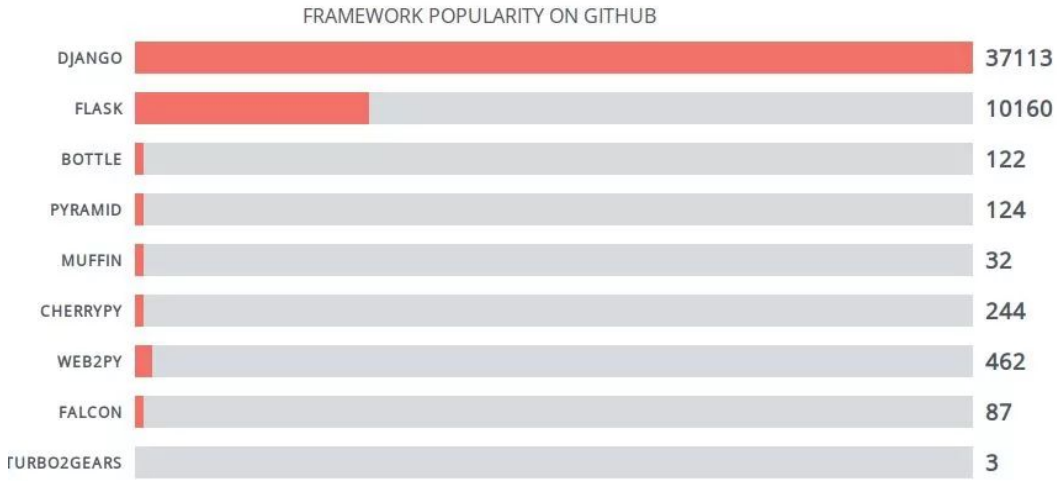


	Account	Name	Rep	Manager	Product	Quantity	Price	Status
0	714466	Trantow-Barrows	Craig Booker	Debra Henley	CPU	1	30000	presented
1	714466	Trantow-Barrows	Craig Booker	Debra Henley	Software	1	10000	presented
2	714466	Trantow-Barrows	Craig Booker	Debra Henley	Maintenance	2	5000	pending
3	737550	Fritsch, Russel and Anderson	Craig Booker	Debra Henley	CPU	1	35000	declined
4	146832	Kiehn-Spinka	Daniel Hilton	Debra Henley	CPU	2	65000	won

		sum				
		Price				
	Product	CPU	Maintenance	Monitor	Software	All
Manager	Status					
Debra Henley	declined	70000	0	0	0	70000
	pending	40000	10000	0	0	50000
	presented	30000	0	0	20000	50000
	won	65000	0	0	0	65000
Fred Anderson	declined	65000	0	0	0	65000
	pending	0	5000	0	0	5000
	presented	30000	0	5000	10000	45000
	won	165000	7000	0	0	172000
All		465000	22000	5000	30000	522000

Flask

Микро фреймворк для создания веб приложений



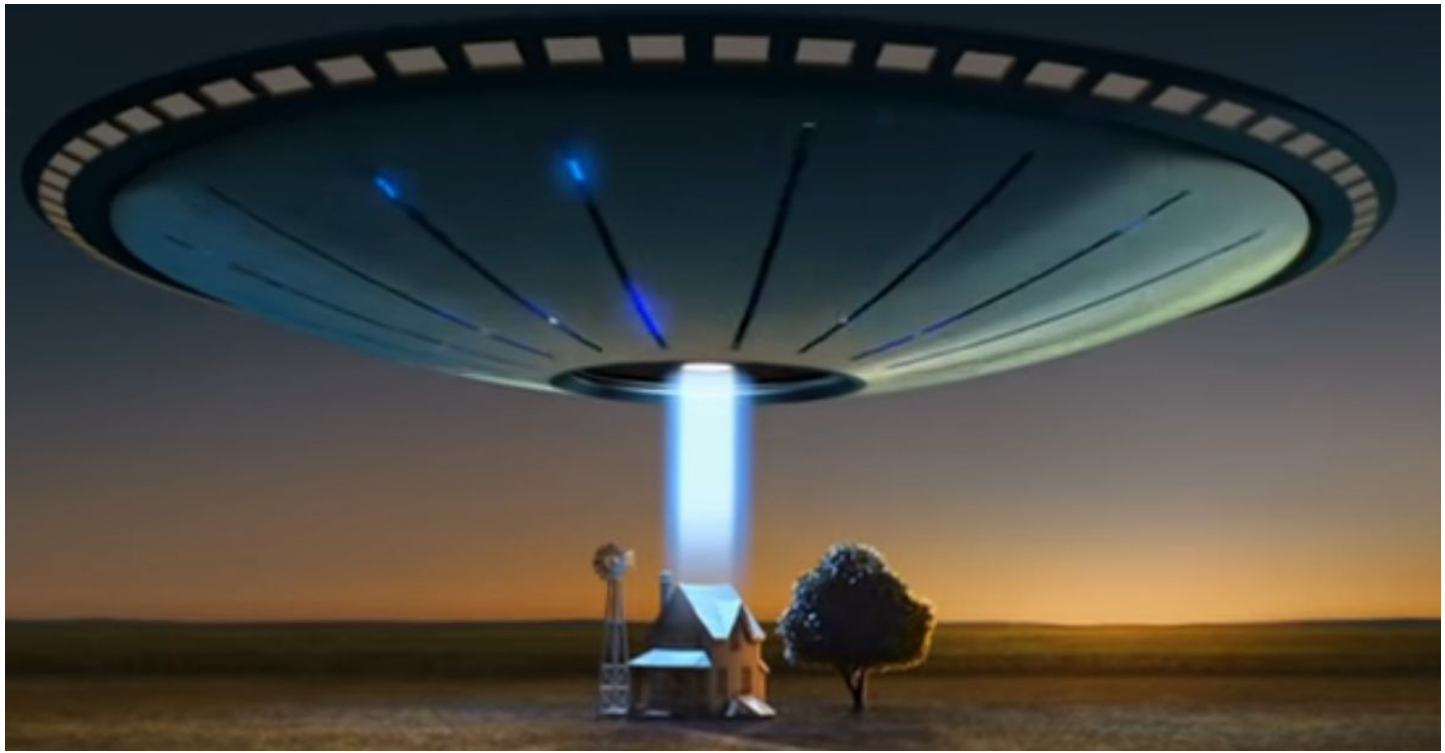
http://flask.pocoo.org/docs/1.0/quickstart/#quickstart

http://flask.pocoo.org/docs/1.0/tutorial/

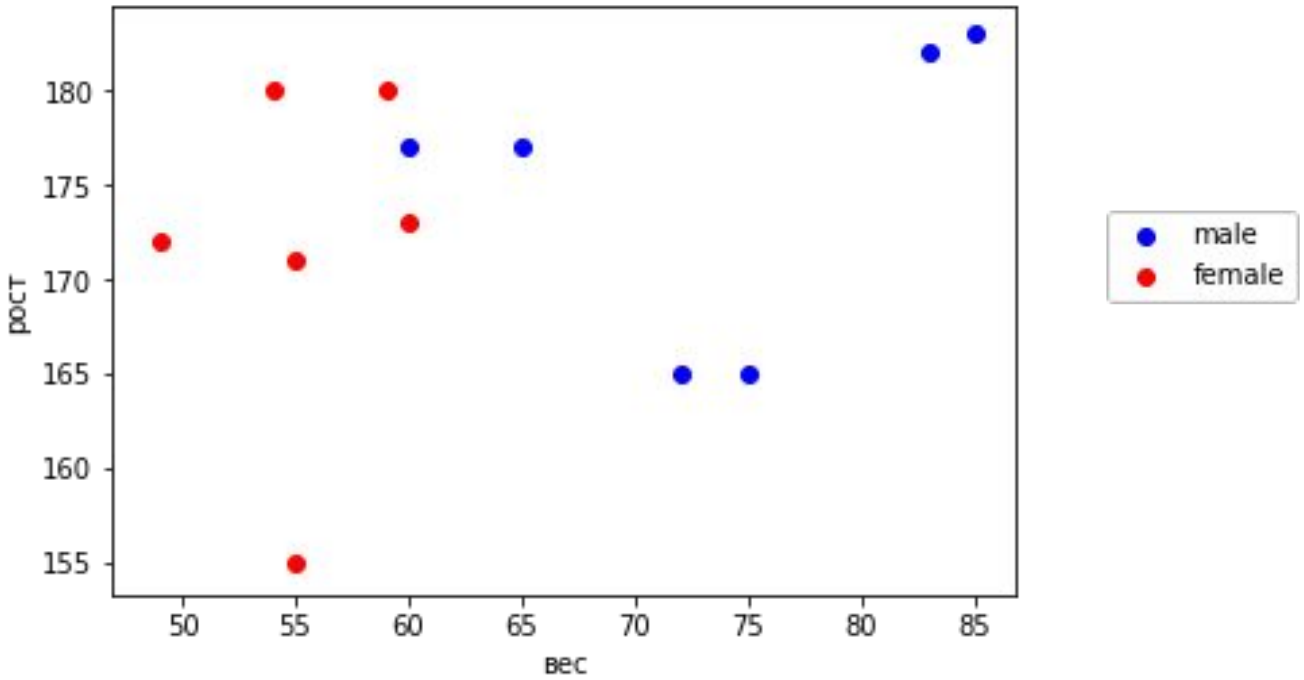
Перерыв

Break

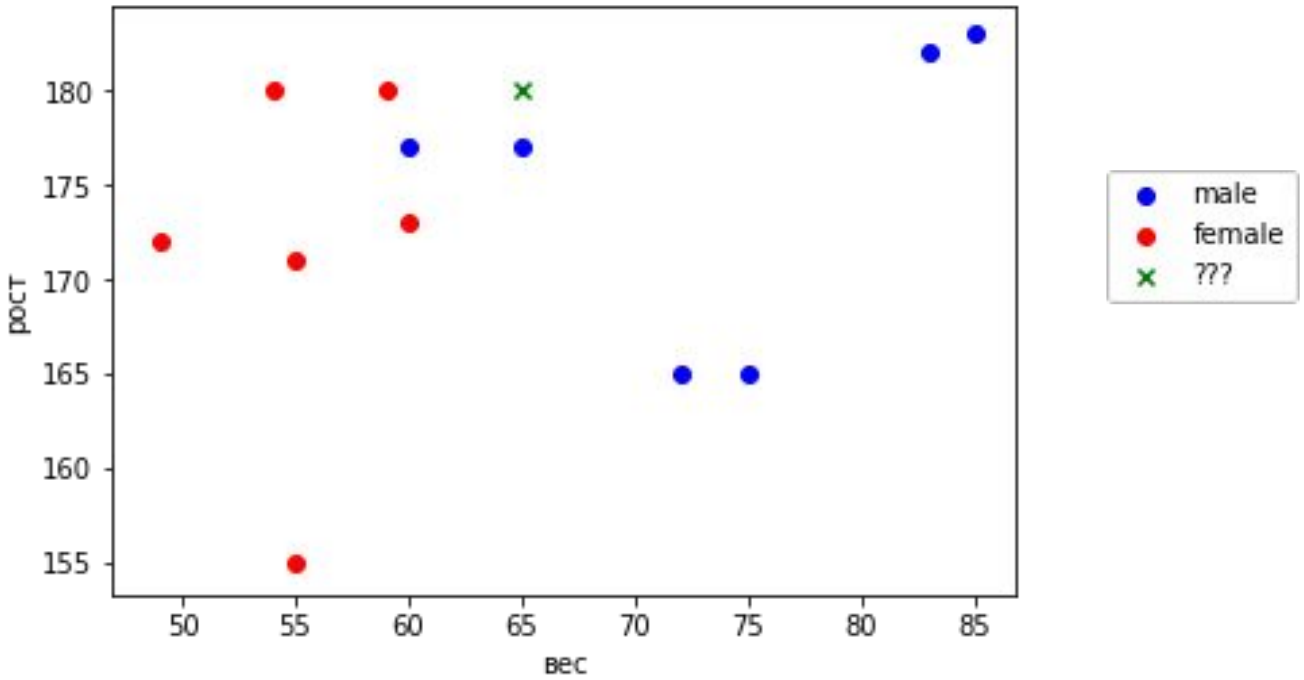
Machine Learning



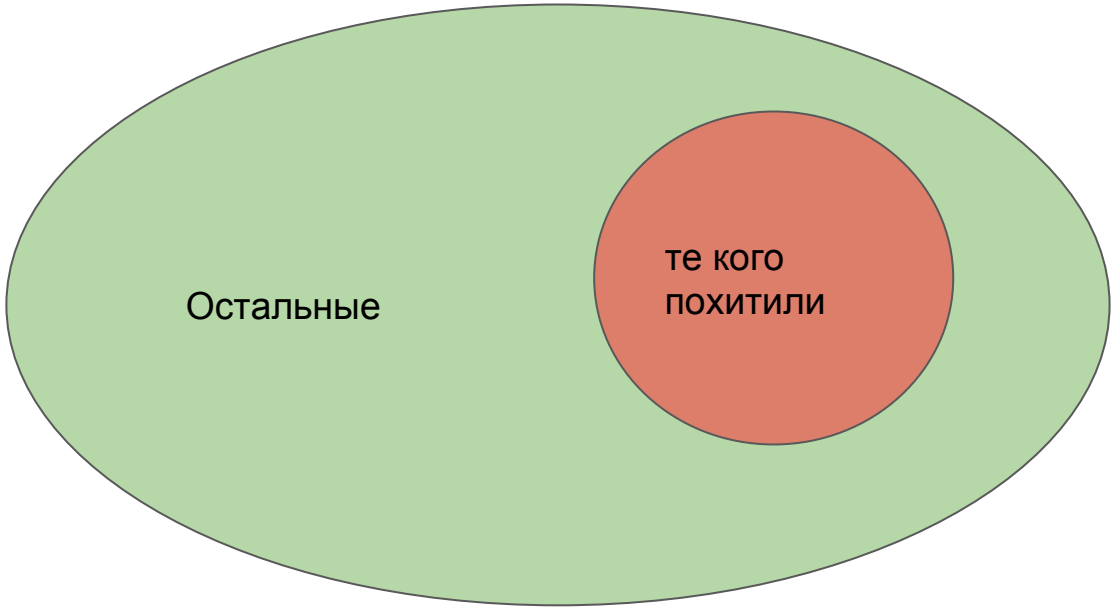
Как узнать пол человека по его двум измерениям?



Как узнать пол человека по его двум измерениям?

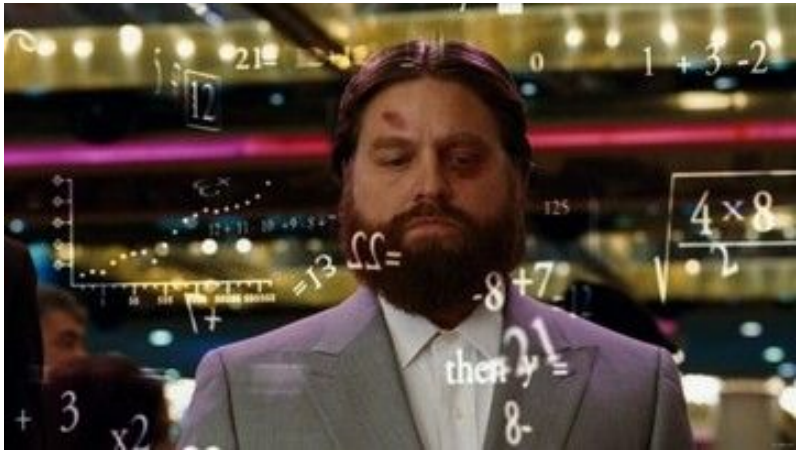


Как понять какой из алгоритмов лучше?



Что именно и как будем оценивать?

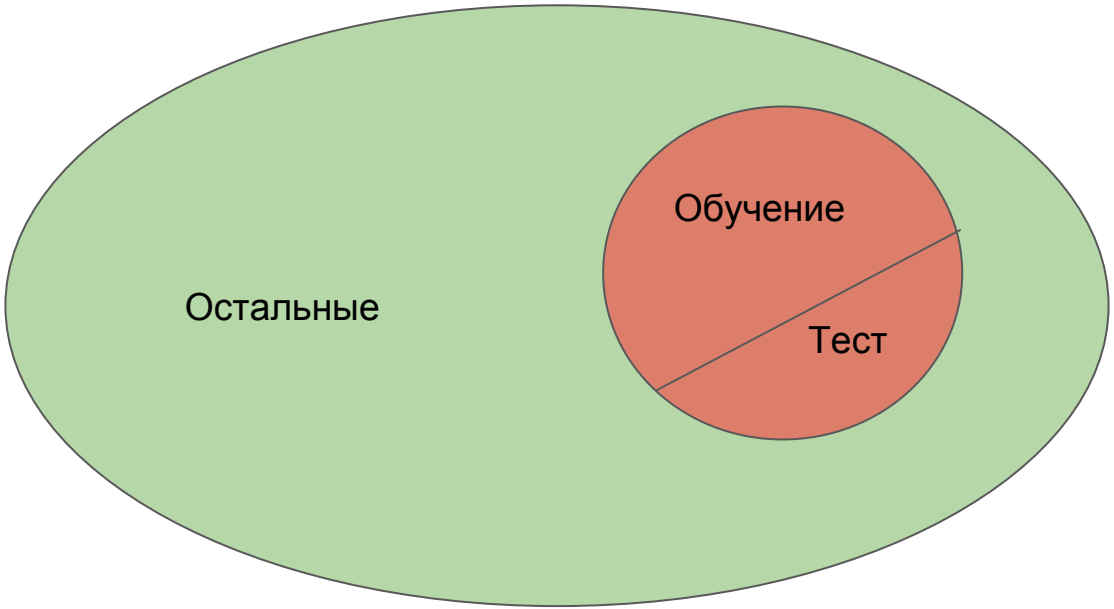
Нужно как-то выразить количественно какое решение лучше. Лучше числом.
Мы ведь понимаем как сравнивать числа?



Ответим на вопросы:

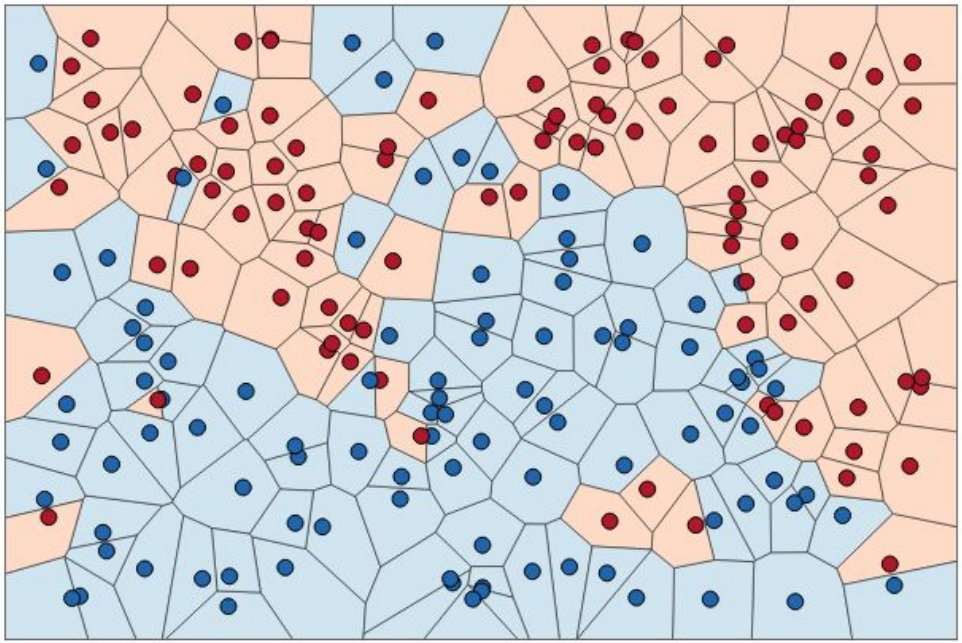
- что меньше 0.5 или 0.3
- что больше 0.682 или 0.621
- что лучше 0.72 или 0.56

Как понять какой из алгоритмов лучше?



Q: Что зависит от нас, а что от данных?

Переобучение



Алгоритм сильно подстроился под тренировочную выборку

credits: http://scott.fortmann-roe.com/docs/BiasVariance.html

Привет соседи

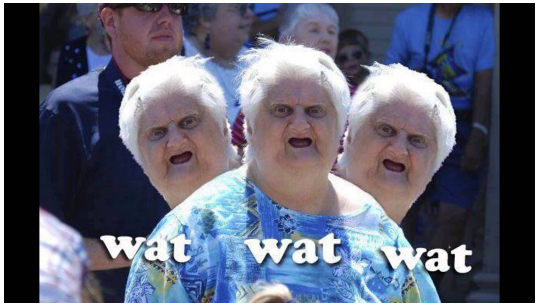
Попробуем пройти пайплайн машинного обучения ?

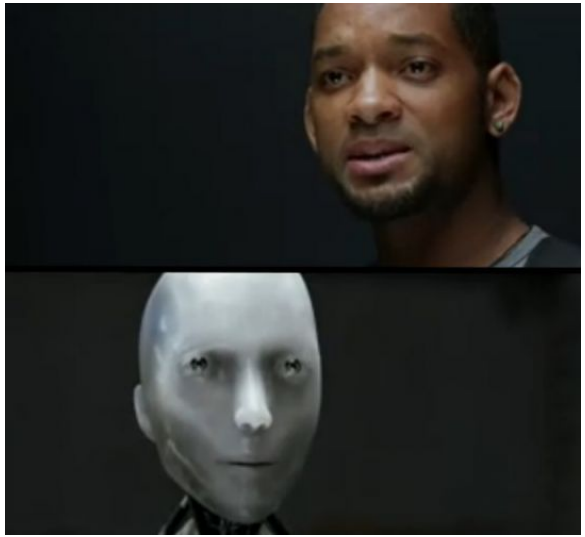
Тогда в ноутбук!

ML

Machine Learning (машинное обучение) – подраздел искусственного интеллекта, изучающий методы построения алгоритмов, способных обучаться.

Обучаться?





ML

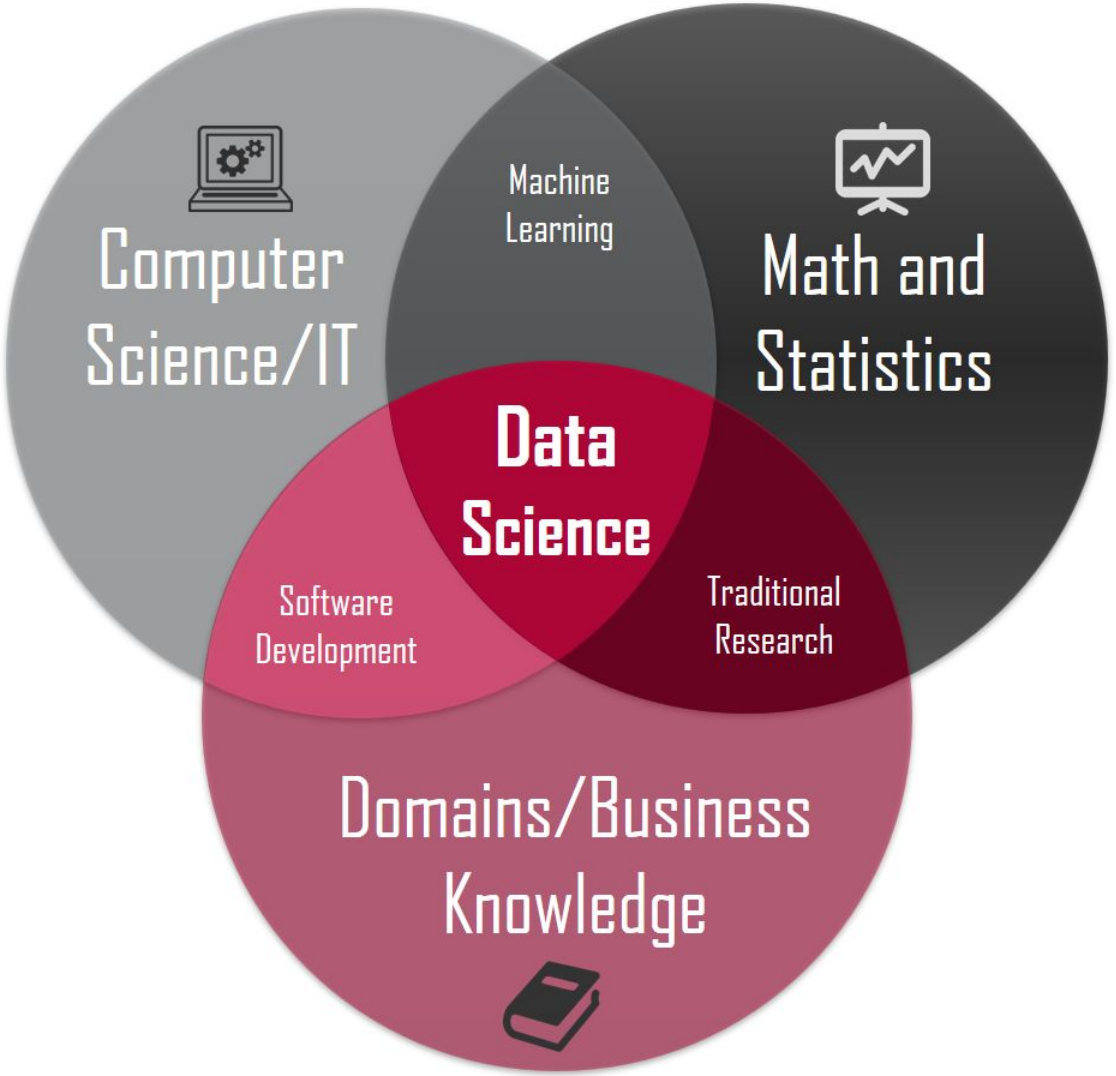
Томас Митчел: “Компьютерная программа обучается на основе опыта E относительно некоторого класса задач T и меры качества P, если качество решения этих задач (относительно P) улучшается с приобретением опыта E”.

T - задача: Обучение с учителем, Обучение без учителя, Обучение с подкреплением, и тд.

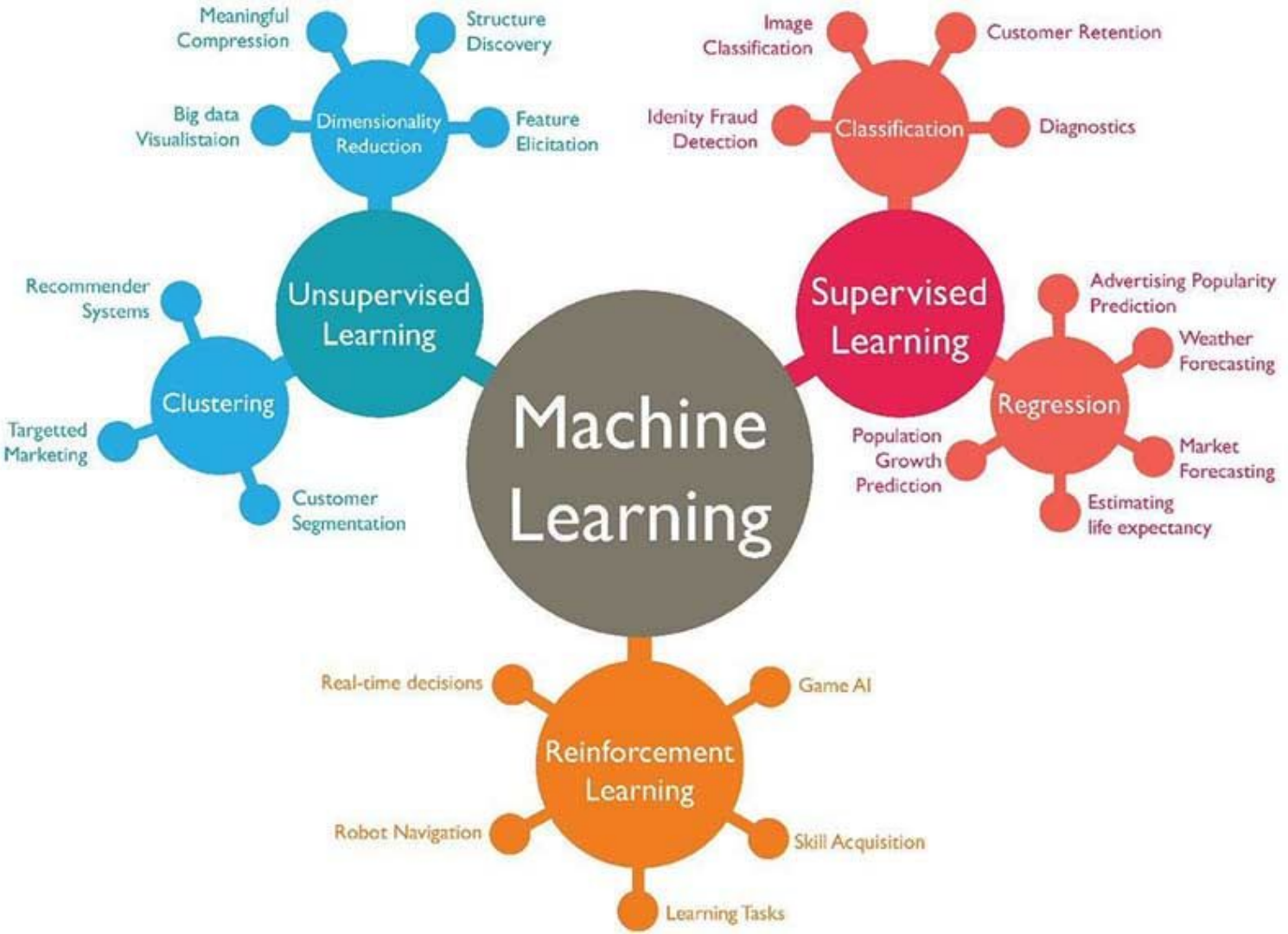
P - метрика: Численное значение показывающее насколько хорошо алгоритм решает задачу.

E - опыт: Для алгоритмов это данные (с метками и без).

ML


Computer Science/IT
Machine Learning
Math and Statistics
Data Science
Software Development
Traditional Research
Domains/Business Knowledge

ML


Machine Learning
Unsupervised Learning
Dimensionality Reduction
Meaningful Compression
Structure Discovery
Big data Visualistaion
Feature Elicitation
Clustering
Recommender Systems
Targetted Marketing
Customer Segmentation
Supervised Learning
Classification
Image Classification
Customer Retention
Idenity Fraud Detection
Diagnostics
Regression
Advertising Popularity Prediction
Weather Forecasting
Population Growth Prediction
Market Forecasting
Estimating life expectancy
Reinforcement Learning
Real-time decisions
Game AI
Robot Navigation
Skill Acquisition
Learning Tasks

Задачи в ML

- Обучение с учителем (supervised learning)
  - регрессия
  - ранжирование
  - классификация
- Обучение без учителя (unsupervised learning)
  - кластеризация
  - снижение размерности
  - детектирование аномалий
- Обучение с подкреплением
- Обучение с частичным привлечением учителя

Обучение с учителем

Пусть X - множество описаний объектов, Y - множество допустимых ответов.

Существует неизвестная целевая зависимость $f : X \to Y$

значения которой известны только на объектах обучающей выборки

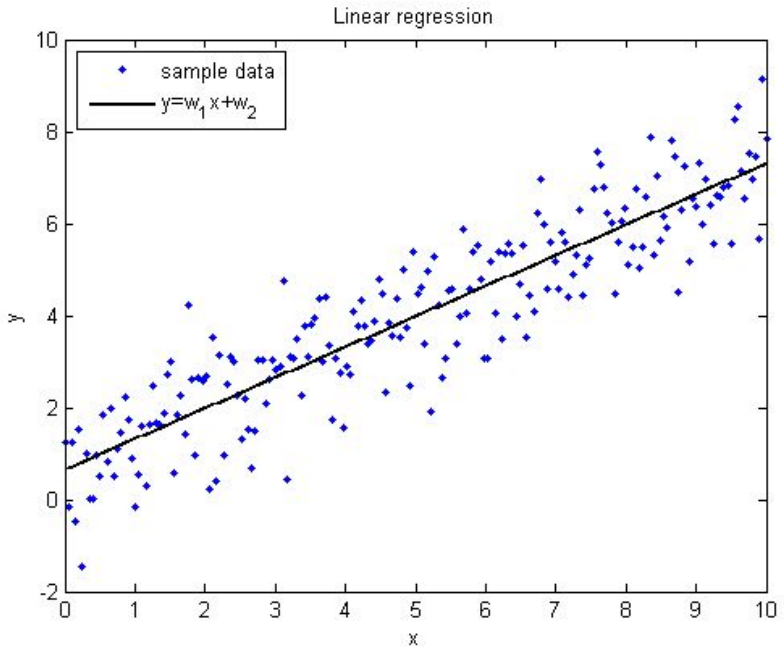
$$X^m = \{(x_1, y_1), \ldots, (x_m, y_m)\}$$

Требуется построить алгоритм $a : X \to Y$, который приближал бы неизвестную

целевую зависимость как на элементах выборки, так и на всём множестве X.

Задачи

- Обучение с учителем
  - регрессия
  - ранжирование
  - классификация
- Обучение без учителя
  - кластеризация
  - снижение размерности
  - детектирование аномалий

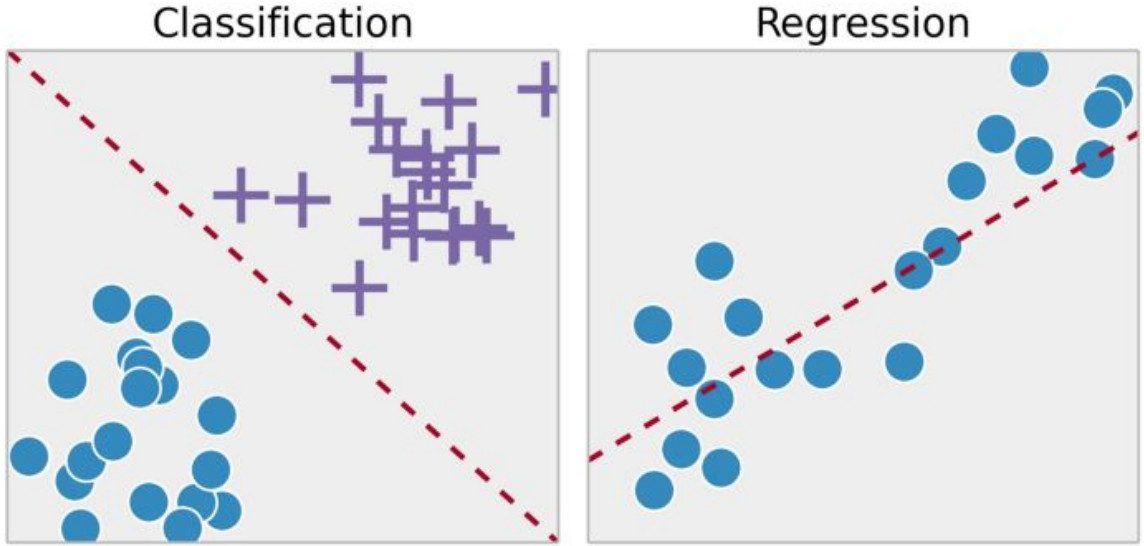


Регрессия: Непрерывные значения целевой переменной (предсказание объема продаж, цены на жилье, температуры и тд)

Какие метрики качества?

Задачи

- Обучение с учителем
  - регрессия
  - классификация
  - ранжирование
- Обучение без учителя
  - кластеризация
  - снижение размерности
  - детектирование аномалий

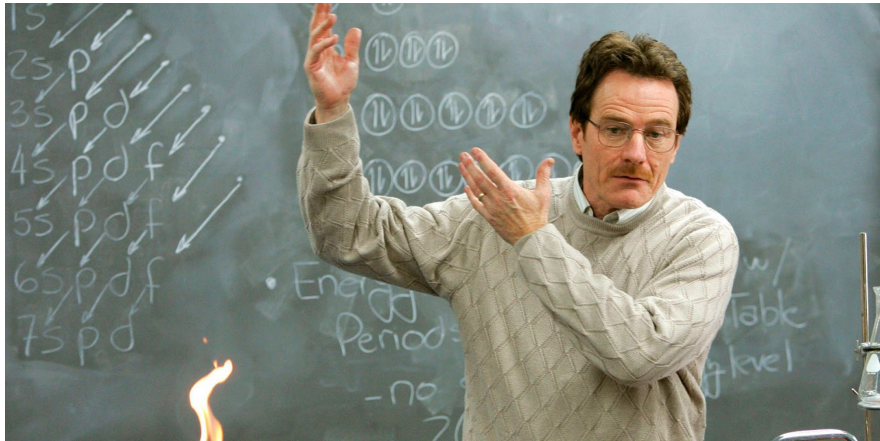


Классификация: Дискретные (конечное число) значения целевой переменной (Предсказание кто изображен на картинке, просрочки кредита, клика на рекламу, спам/не спам)

Какие метрики качества?

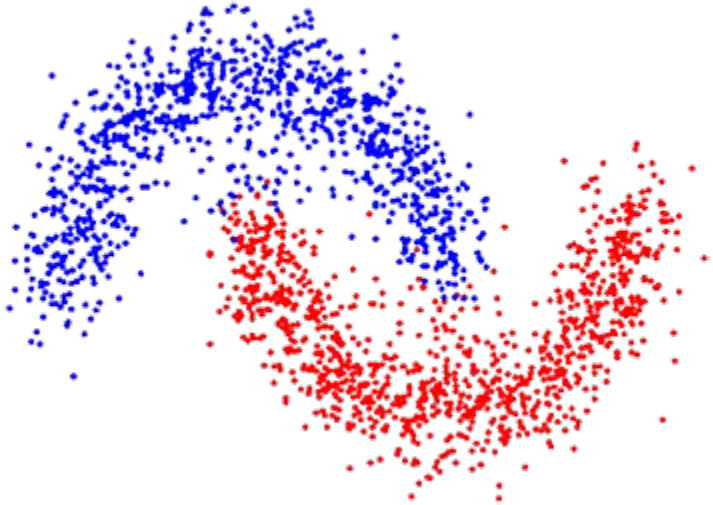
Обучение без учителя

Обучение без учителя - класс задач в которых известны только описания множества объектов (обучающей выборки), и требуется обнаружить внутренние взаимосвязи, зависимости, закономерности, существующие между объектами.



Задачи

- Обучение с учителем
  - регрессия
  - классификация
  - ранжирование
- Обучение без учителя
  - кластеризация
  - снижение размерности
  - детектирование аномалий

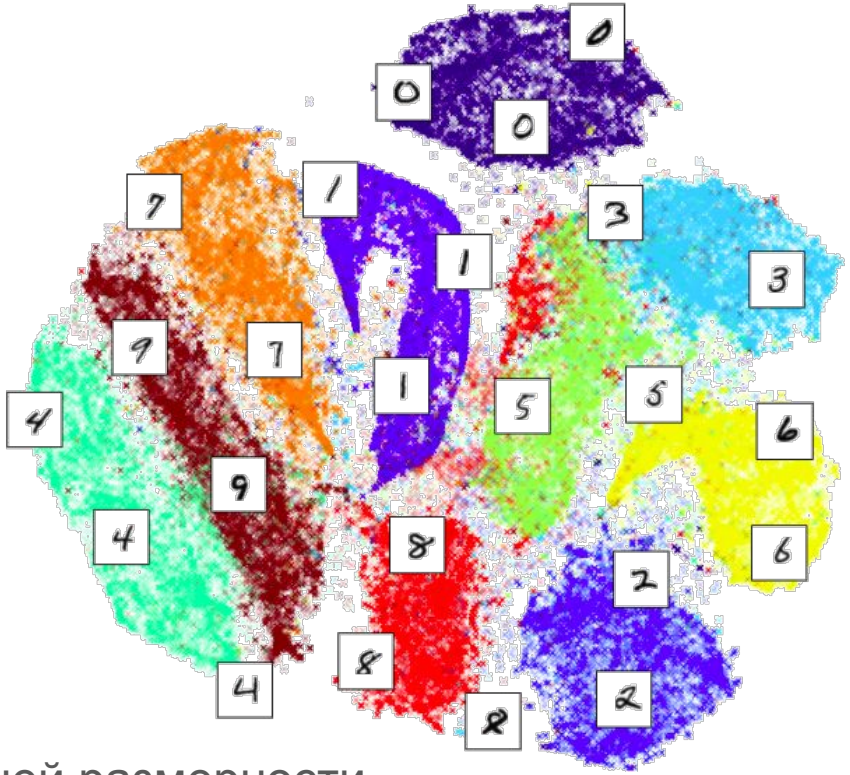


Разбиение выборки объектов на непересекающиеся группы;
объекты в одном кластере - должны быть похожи, в разных - различаться.
(Сегментация клиентов по поведению, выделение сообществ в соцсетях)

Какие метрики качества?

Задачи

- Обучение с учителем
  - регрессия
  - классификация
  - ранжирование
- Обучение без учителя
  - кластеризация
  - снижение размерности
  - детектирование аномалий

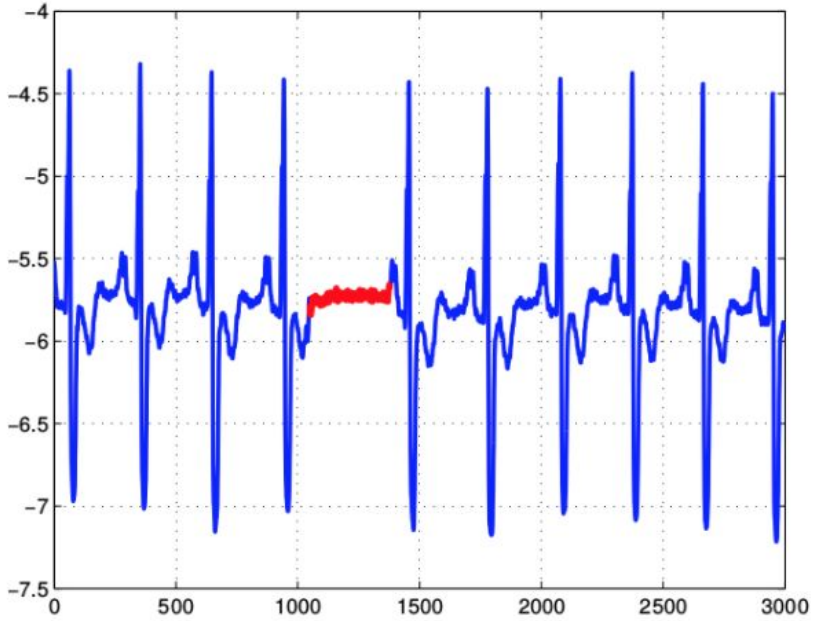


Представить данные в пространстве меньшей размерности
минимизировав потери информации
(Меньше лишнего, возможность визуализировать)

Какие метрики качества?

Задачи

- Обучение с учителем
  - регрессия
  - классификация
  - ранжирование
- Обучение без учителя
  - кластеризация
  - снижение размерности
  - детектирование аномалий



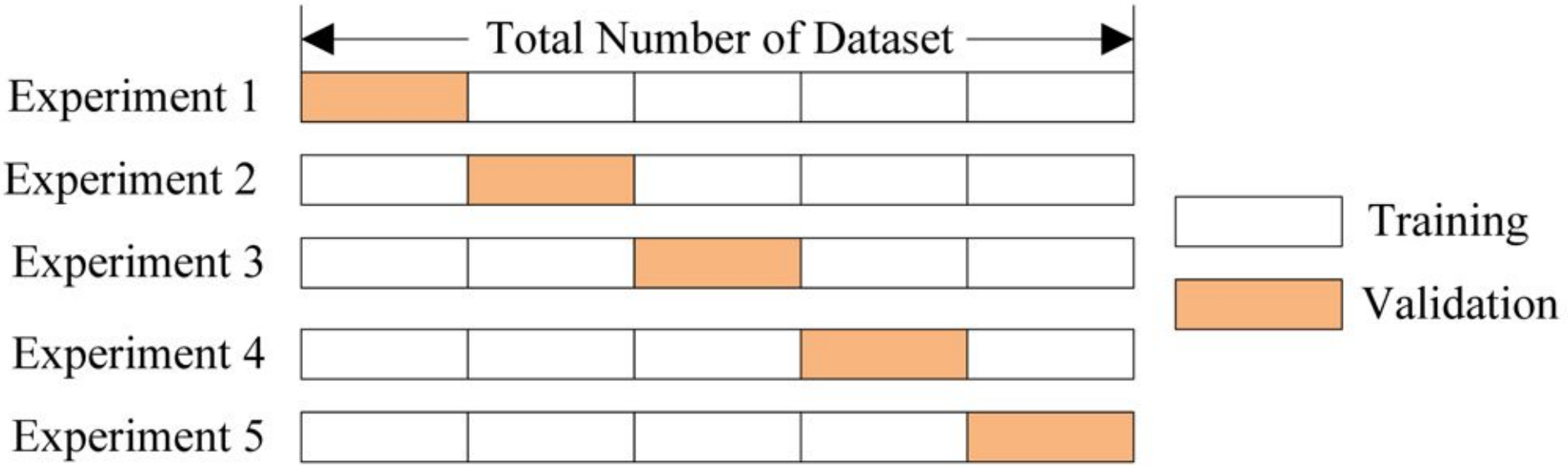
Выделение в данных нетипичных представителей
(Обнаружение мошеннических переводов, вмешательств, поломок)

Какие метрики качества?

Взглянем заново на обучение с учителем

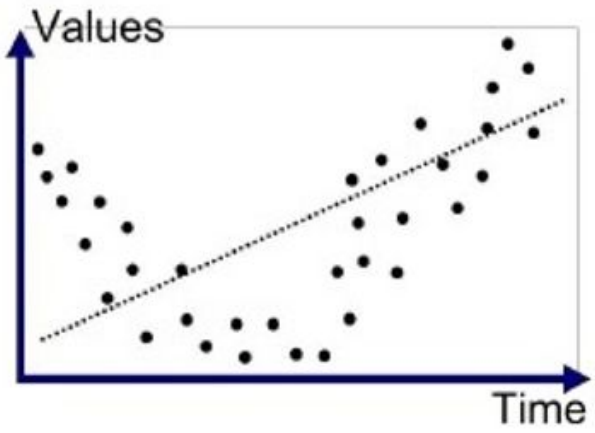
- Задача
- Данные (признаковое описание объектов и целевые метки)
- Метрика качества
- Правильная валидация (алгоритм должен быть обобщаемым)
- Выбор модели (класса моделей)
- Подготовка признаков (в соответствии с моделью)
- Обучение (подбор параметров, оптимизируя метрику)
- up to you - оценка адекватности, исследование примеров где модель ошибается

Перекрестная проверка (cross validation)

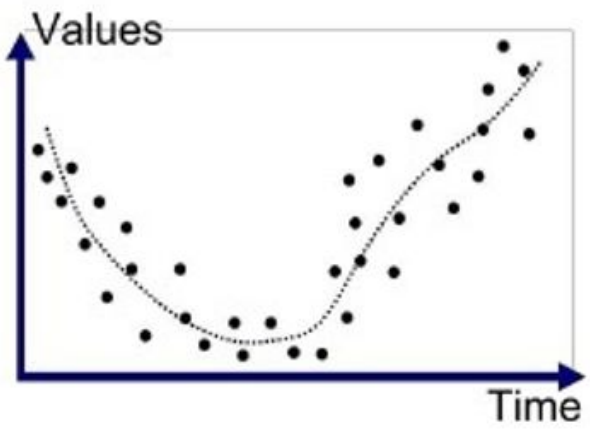


https://en.wikipedia.org/wiki/Cross-validation_(statistics)
Перекрестная проверка

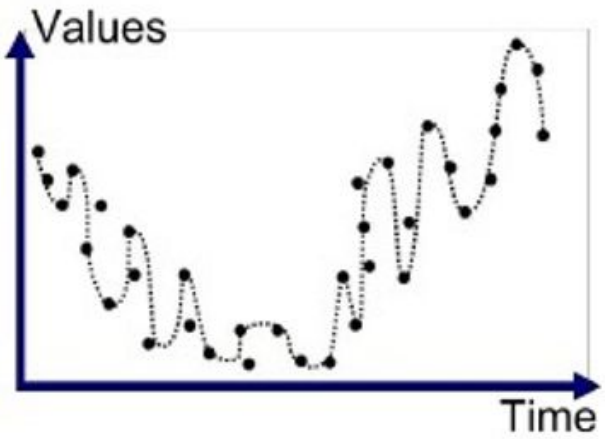
Недообучение - Переобучение



Underfitted



Good Fit/Robust



Overfitted

Что с классификацией?

Переобучение, Bias_variance

Data leakage

Переобучение может быть не только из-за сложной модели. Может быть из-за утечки в данных.

Простой пример - стандартизация признаков, хотя в идеальном мире - не страшно.

На чем считать среднее и дисперсию?

- трейн
- тест
- трейн + тест

Как избежать факапа?

Резюме по ML

ML - аппроксимация алгоритмов (читай зависимостей) которые должны обобщаться на новые данные. Аппроксимация происходит за счет данных для :

- ускорения алгоритмов
  (игры - шахматы, го)
- решения сложных проблем
  (сложные или неизвестные распределения: картинки, физика, химия, поведение людей)

В отличие от статистики:

- не заботимся о природе распределения, заботимся о решении.

В отличие от CS:

- алгоритмы (паттерны) берем из настоящих данных, а не хардкодим
- улучшения алгоритма в основном зависят от хорошей настройки алгоритма к самим данным и наличия хорошего “сигнала” в данных

Резюме по ML

ML = Модель + оптимизация + предположения о данных

Оптимизация:

- перебор
- случайное угадывание
- жадные методы
- оптимизация (градиентная, генетическая, дискретная)

Предположения о данных:

регрессии: взаимодействие между признаками
картинок: рядом находящиеся пиксели - скоррелированы,
позиция объекта на картинке - не сильно важно,объекты - связные регионы.
рекомендации: людям с одинаковыми интересами нравятся одинаковые товары
обработка языка: много синонимов, тексты одного автора имеют одинаковые статистические свойства.

Резюме по ML

Теорема.

No Free Lunch: универсального алгоритма нет.
Чтобы успешно решить один тип проблем - нужно пожертвовать способностью к обобщению на остальных задачах.

Резюме по ML

ML - в задаче обучения с учителем - это про модели которые обобщаются на новые данные.

Нужно правильно подготавливать модель и честно оценивать ее обобщающую способность на hold-out датасете (Тесте)

Не допускать такого чтобы информация про тест была известна в процессе обучения

Домашнее задание

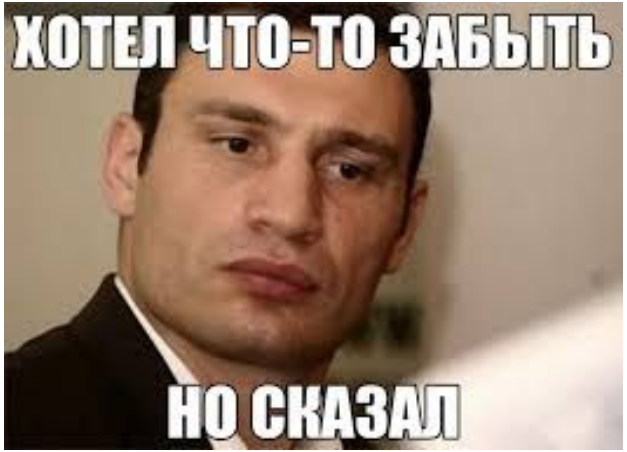
1. Ветка homework_03
2. Недельник - набор ноутбуков. Сдача к 00:00 17 октября. Желательно сдавать подневно, но никаких штрафов нет.
3. Flask-сервер, простой. Дедлайн 00:00 24 октября.
4. ДЗ по машинному обучению, ветке. Дедлайн 00:00 24 октября
5. Как всегда pull-request

Проверка первого дз - проверим на неделе. Второе - присылайте.

Источники

Оптимизация
Визуализация в Python
Flask мануал
Пандас от Дьяконова
О работе ipython, иногда полезно
Обзор numpy
Просто о numpy
Хабростатья про numpy
10 минут и чемпион pandas
Мануал для новичков пандаса
Крутые туторы по numpy + scipy
Объемный обзор фласка
Хаброучебник фласка

Оставьте обратную связь!!!



Appendix

SciPy

Обертка над NumPy

Пакет с набором функций
для научных вычислений где вы можете:

- интегрировать
- решать линейные уравнения
- находить собственные вектора
- находить экстремумы функций
- оптимально работать с разреженными данными
- + спец. пакеты для фурье/обраб.сигналов/изображений и тд

См. подробности

`scipy.cluster`	Vector quantization / Kmeans
`scipy.constants`	Physical and mathematical constants
`scipy.fftpack`	Fourier transform
`scipy.integrate`	Integration routines
`scipy.interpolate`	Interpolation
`scipy.io`	Data input and output
`scipy.linalg`	Linear algebra routines
`scipy.ndimage`	n-dimensional image package
`scipy.odr`	Orthogonal distance regression
`scipy.optimize`	Optimization
`scipy.signal`	Signal processing
`scipy.sparse`	Sparse matrices
`scipy.spatial`	Spatial data structures and algorithms
`scipy.special`	Any special mathematical functions
`scipy.stats`	Statistics

Collections

Модуль с реализацией некоторых типов данных основанных на dict, set, list.

https://docs.python.org/3/library/collections.html

Functools

Модуль предоставляющий реализацию функций высокого уровня - функции которые возвращают другие функции.

https://docs.python.org/3/library/functools.html

Itertools

Модуль itertools может помочь создать свои итераторы.

https://docs.python.org/3.7/library/itertools.html