



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ



ФАКУЛЬТЕТ ГЕОГРАФИИ
И ГЕОИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

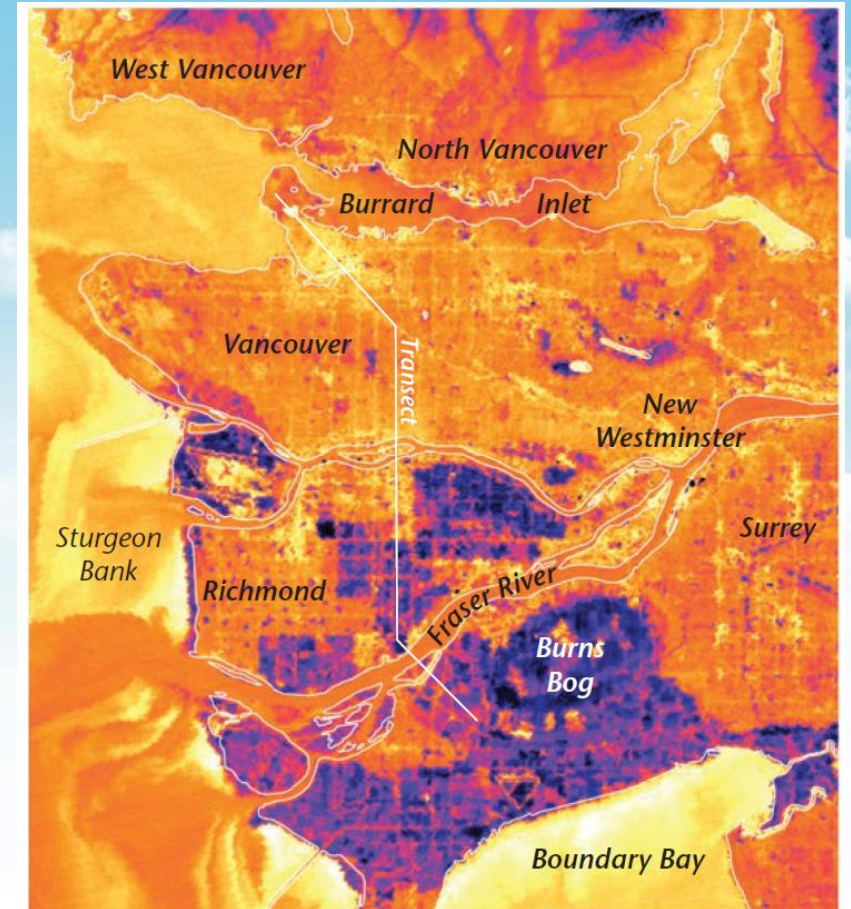
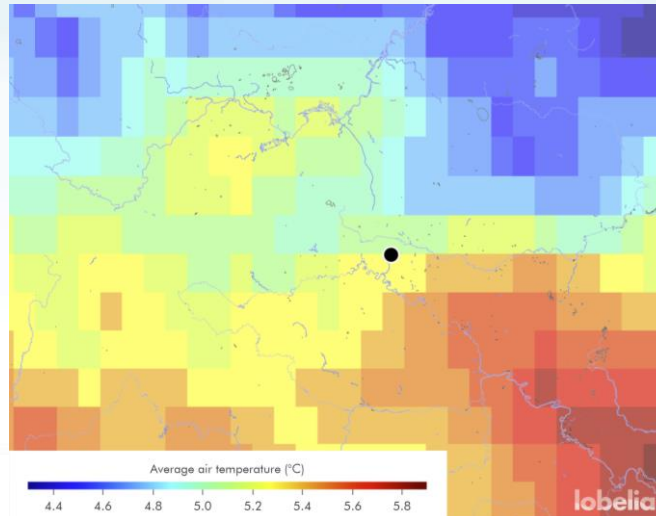
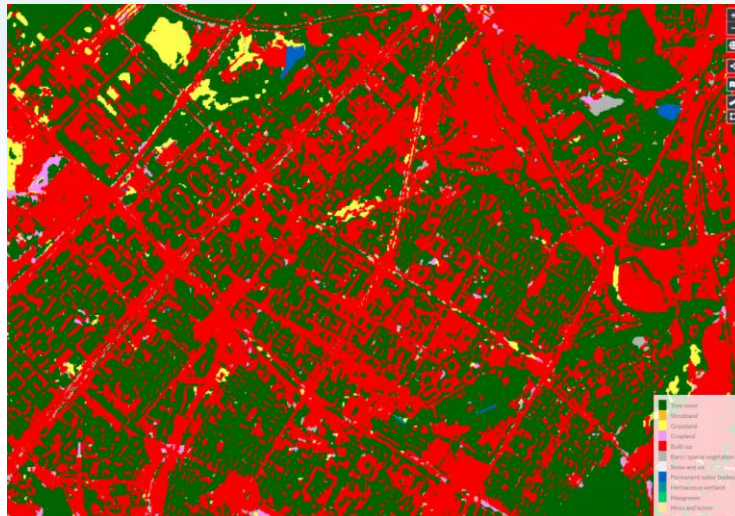
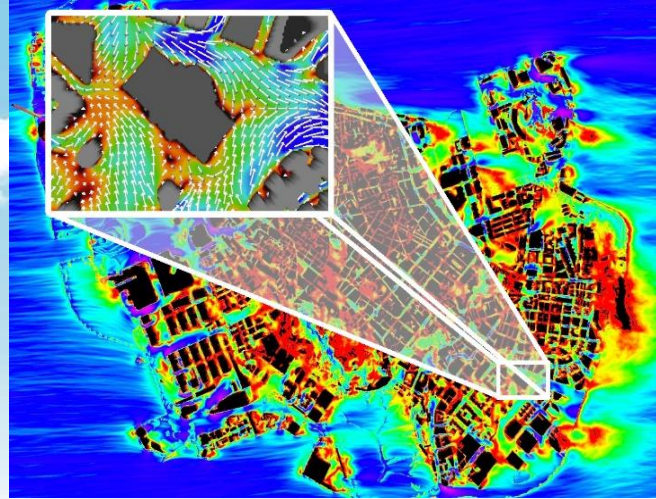
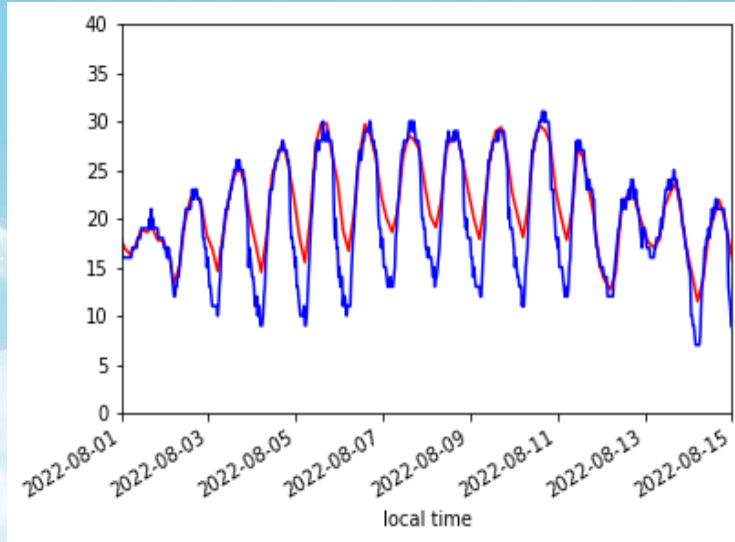
Курс «Моделировании климата городов» 2025, лекция №3

Технические средства анализа данных

Михаил Иванович Варенцов

mvarentsov@hse.ru

В предыдущих сериях...



В предыдущих сериях...



Advances in Sensors

- Advances in wireless data transmission
- Low power design (reduced static/dynamic consumption)
- Reduced size
- Location awareness
- Higher resolution

- Internet of Things (IoT), Web of things (WoT), Internet of Everything (IoE)
- Secure data transmission and system protection
- Real-time data integration in dashboards and digital twins

Advances in Digital Infrastructure

- Cloud computing
- Edge computing
- Increased computational power and efficiency
- Increased storage capacity
- Improved communication networks

Increased Accessibility

UCI Applications

- Humans as sensors
- Novel non-obtrusive smart devices and applications
- New sensing methods (e.g., mobile, body-worn, garments, drones, CubeSats, LIDAR, MLS)
- Methods across scales

- Climate-sensitive urban design and planning
- Development of adaptation and mitigation strategies for urban climate challenges (such as heat and air quality)
- More comprehensive vulnerability and inequity analyses
- Improving human health and wellbeing through human-centric approaches

- On-demand cloud computing and APIs (e.g., AWS, Azure)
- Real-time data analytics
- Public/scientific cloud computing, visualization, and analytics platforms (e.g., Google Earth Engine)
- Cloud-based climate modeling
- Digital twins of Earth/urban atmosphere

Novel Data Sources

- Government or commercial urban data (3D building models)
- Community generated and curated data
- Incidental data (social media, consumer data)
- Public domain data (web scraping)

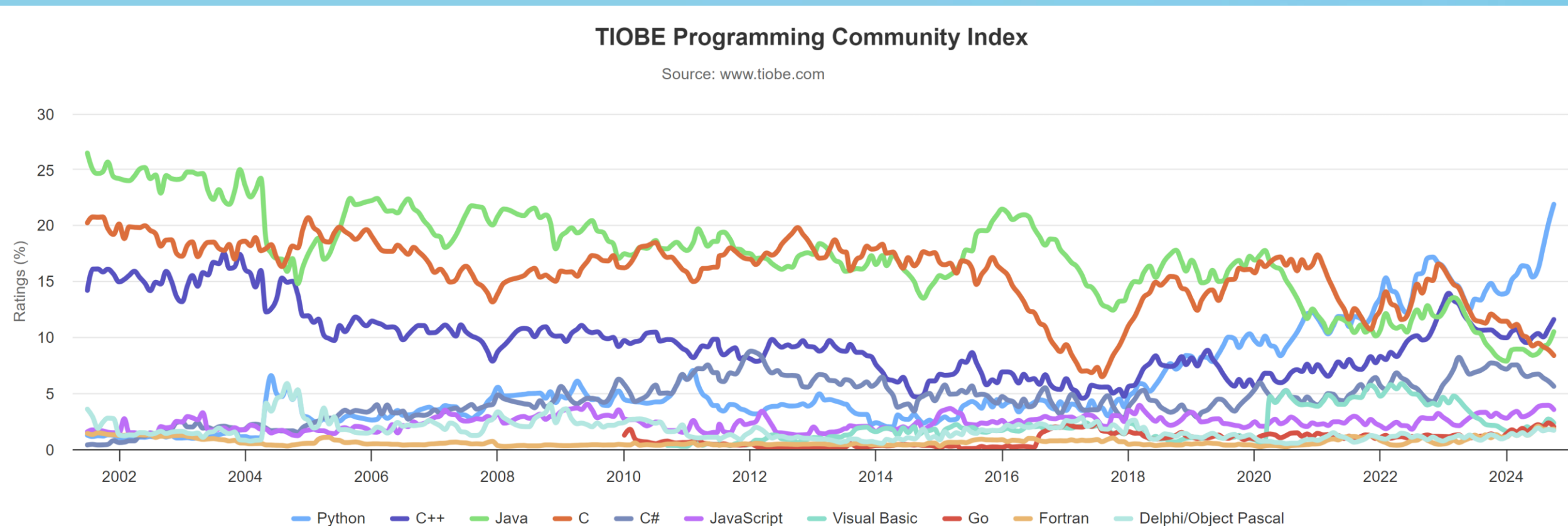
- Proxy calculation and modeling using novel data sources (e.g., Google Street View)
- Pattern recognition of environmental impacts based on open data sources (e.g., Facebook/Twitter)
- Novel open-source data formats and standards (e.g. CityGML)

Advances in Analytical Algorithms & Platforms

- Artificial intelligence (including machine and deep learning)
- Augmented data management
- Procedural, predictive, and agent-based modeling
- Image processing

Python для анализа данных

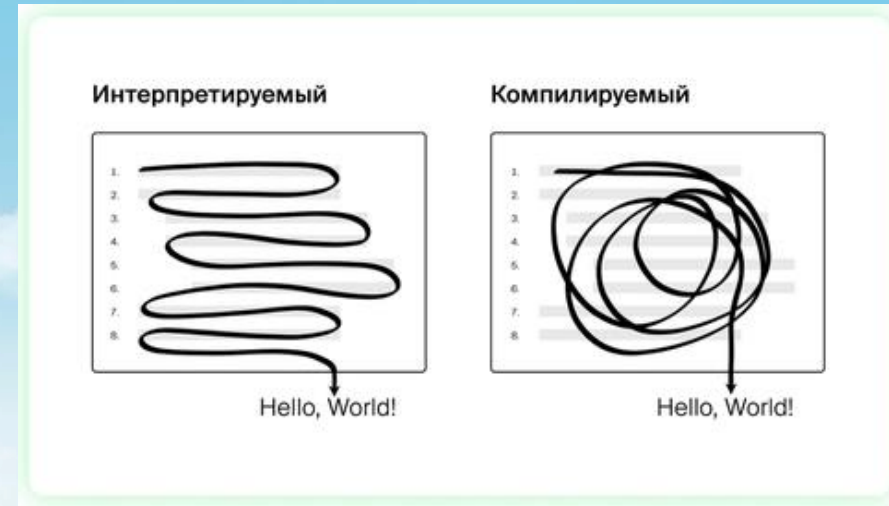
Почему Python?



Рейтинг TIOBE (TIOBE Index) — это ежемесячный индекс, который отражает популярность языков программирования. Он основан на количестве поисковых запросов в различных поисковых системах, таких как Google, Bing, Yahoo!, Wikipedia и других.

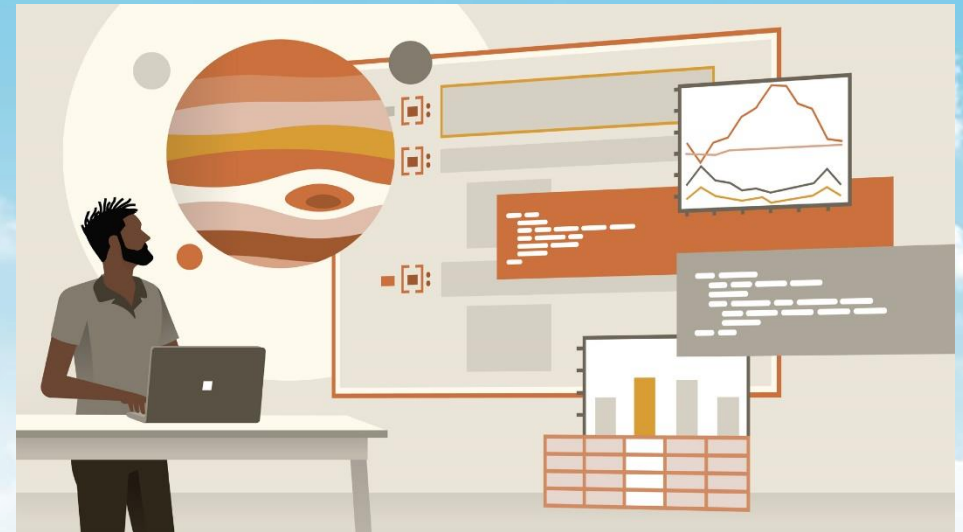
Экосистема Python

- ❑ Python – один из интерпретируемых языков программирования (наряду с R, Matlab, Julia и др.).
 - Исполняется ровно точно то, что написано
 - Исполняется построчно
 - Ошибки идентифицируются только в момент исполнения
- ❑ Ключевые элементы экосистемы:
 - **Дистрибутив Python** (например Anaconda)
 - **Среда выполнения:** python (встроенная), ipython, Jupiter notebook, Google Colab и пр.
 - **Среды разработки (IDE):** VS Code, PyCharm, Spyder, Jupiter Lab и пр.
 - **Пакетный менеджер** (pip, conda)
 - **Окружение (environment)**



Jupyter Notebook

- ❑ Поддержка разных языков программирования (Ju-Py-teR - Julia, Python & R), bash и др.
- ❑ Нелинейная последовательность исполнения (важно об этом помнить)
- ❑ Возможность дополнить код форматированными и иллюстрированными пояснениями в формате markdown
- ❑ Интерактивные графики и элементы
- ❑ Платформонезависимость
- ❑ Возможности работы в других средах разработки (VS Code)



nature

[Explore content](#) ▾ [About the journal](#) ▾ [Publish with us](#) ▾ [Subscribe](#)

[nature](#) > [toolbox](#) > article

TOOLBOX | 30 October 2018

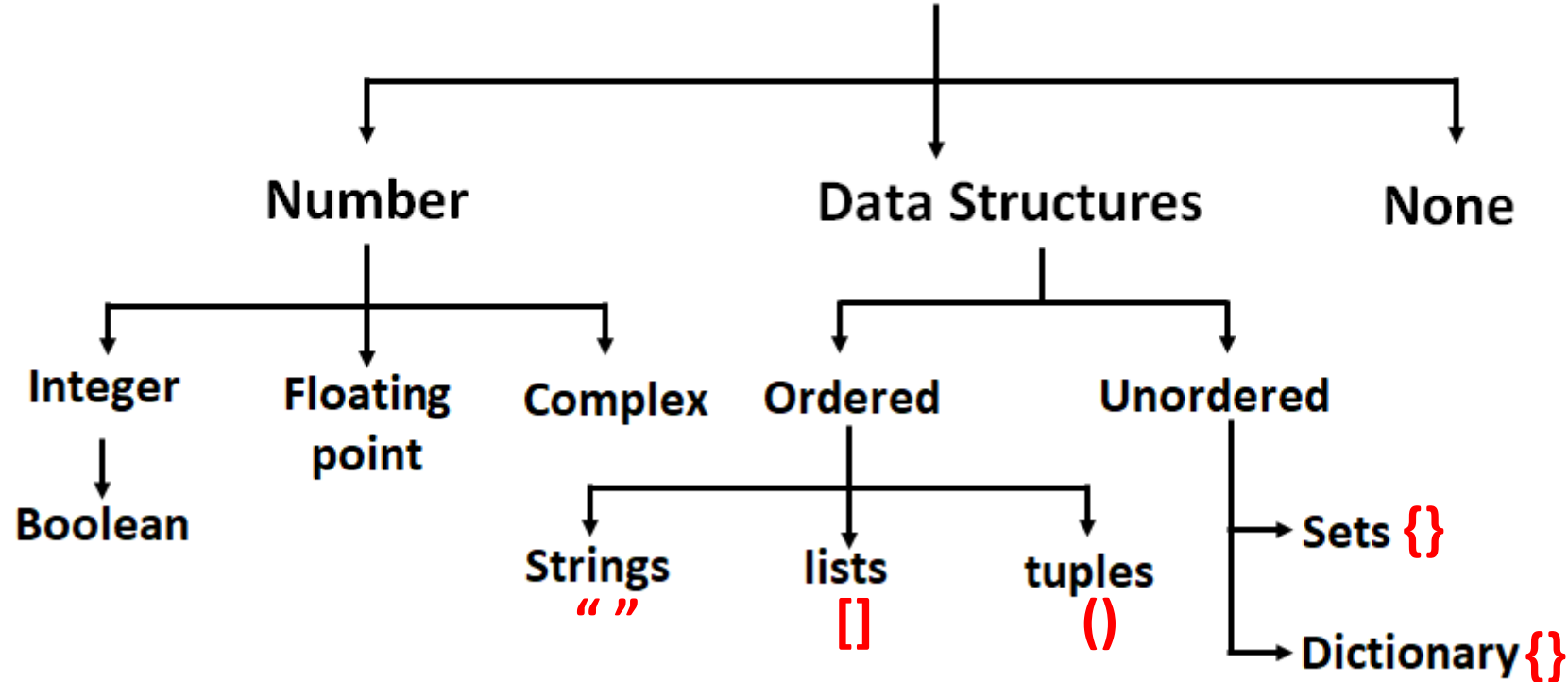
Why Jupyter is data scientists' computational notebook of choice

An improved architecture and enthusiastic user base are driving uptake of the open-source web tool.

Базовые типы данных в Python

teachoo

Python Data Types



Функции и классы в Python



```
1 def double(x):
2     return x * 2
3
4 def add_one(x):
5     return x + 1
6
7 print(double(add_one(1)))
8
9 # result:
10 # 4
11
```



```
class Dog:
    def __init__(self, weight, height, breed, name):
        self.weight = weight
        self.height = height
        self.breed = breed
        self.name = name

    def info(self):
        return "Hello my name is {}".format(self.name)

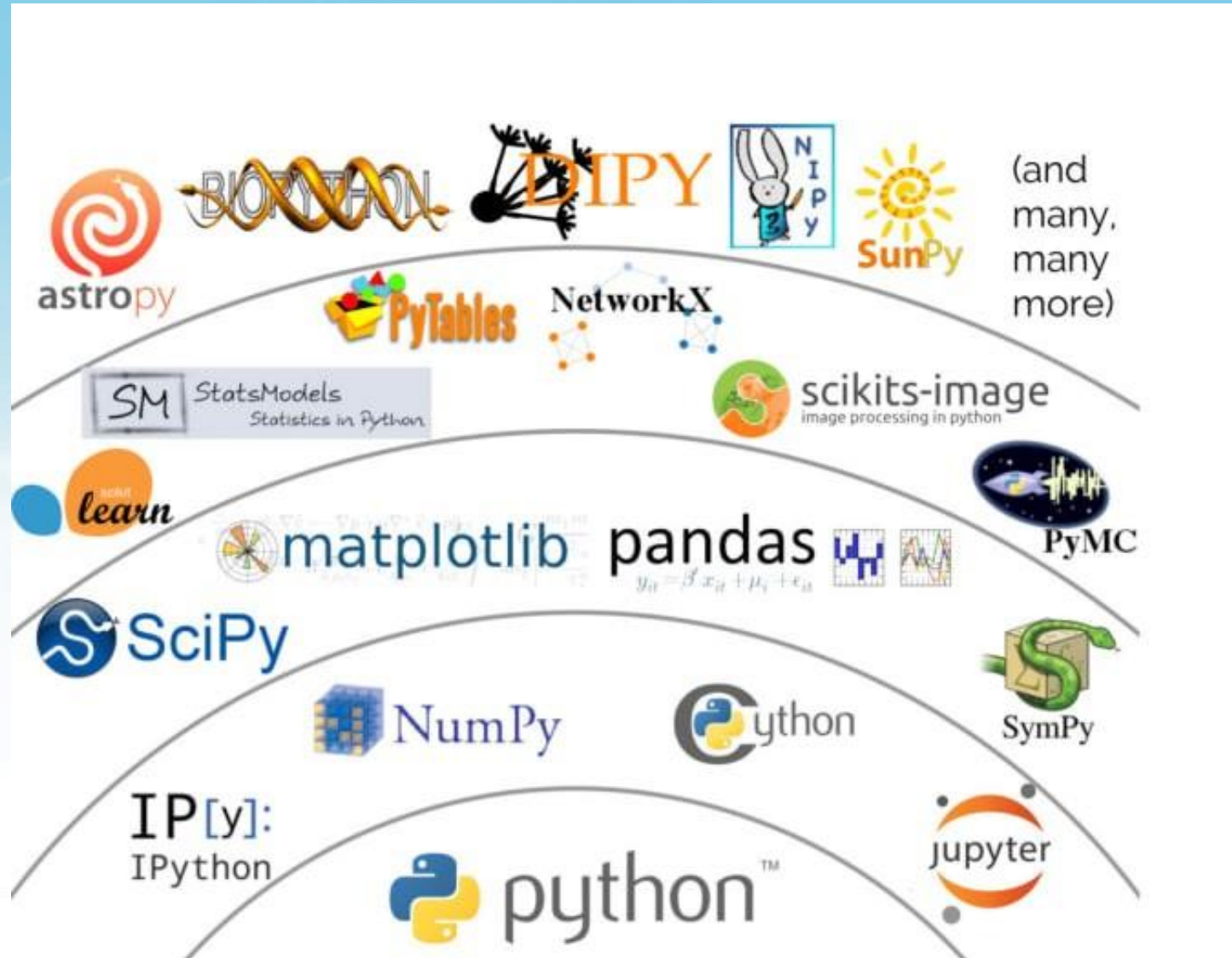
    def bark(self):
        return "The {} is barking".format(self.breed)

    def bmi(self):
        return int(self.weight / self.height)

Dog1 = Dog(30, 50, "golden retriever", "Terry the Legend")
Dog2 = Dog(900, 4, "poodle", "Fluffy the Amazing")

print(Dog1.info())
print(Dog1.bark())
print(Dog2.bmi())
```

Python для анализа данных



Python для анализа данных

Наиболее важные для нашего курса:

- ❑ [NumPy](#) – работа с многомерными массивами, матрицами, математические операции
- ❑ [Pandas](#) – работа табличными данными
- ❑ [Matplotlib](#) – графики на все случаи жизни с ручной настройкой
- ❑ [Scikit-learn](#) – базовый уровень машинного обучения
- ❑ [Scipy](#) – статистический анализ

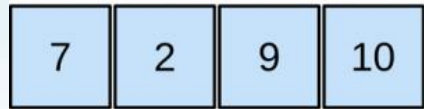
Также могут пригодиться:

- ❑ [Xarray](#) – работа с многомерными массивами данных, имеющих пространственно-временную привязку (netcdf, grib, hdf)
- ❑ [Seaborn](#) – продвинутое графическое представление для статистического анализа
- ❑ [Rasterio](#) – работа с пространственными растровыми данными (geotiff)
- ❑ [Shapely](#), [geopandas](#) – работа с векторными пространственными данными



Многомерные массивы: NumPy

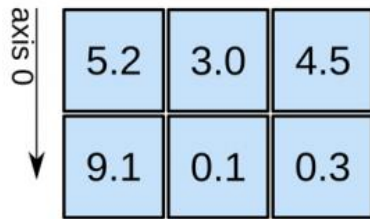
1D array



axis 0 →

shape: (4,)

2D array

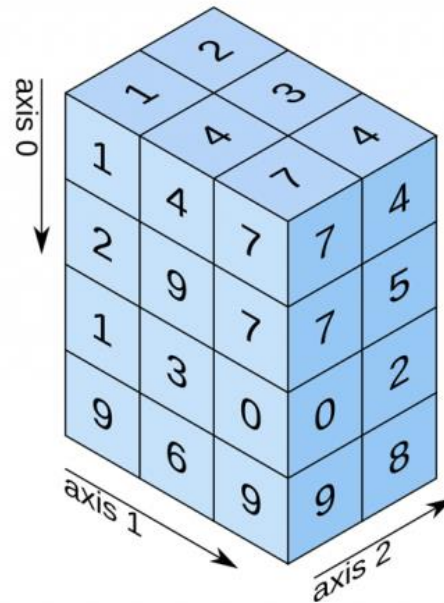


axis 0 ↓

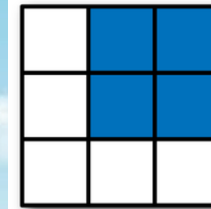
axis 1 →

shape: (2, 3)

3D array



shape: (4, 3, 2)

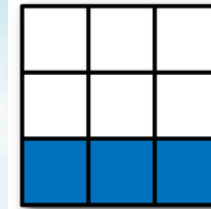


Expression

arr[:2, 1:]

Shape

(2, 2)



arr[2]

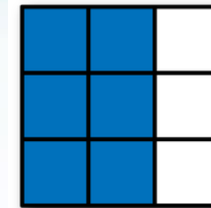
(3,)

arr[2, :]

(3,)

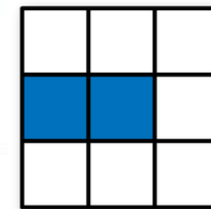
arr[2:, :]

(1, 3)



arr[:, :2]

(3, 2)



arr[1, :2]

(2,)

arr[1:2, :2]

(1, 2)

Прямоугольные данные: Pandas

Ключевые термины для прямоугольных данных

Кадр данных (data frame)

Прямоугольные данные (подобно электронной таблице) — это базовая структура данных для статистических и автоматически обучающихся моделей.

Признак (feature)

Столбец в таблице обычно называется признаком.

Синонимы: атрибут, вход, предсказатель, предиктор, переменная.

Исход (outcome)

Многие проекты науки о данных предусматривают с предсказание исхода — нередко в формате да/нет (например, в табл. 1.1 это ответ на вопрос "Были ли торги состязательными или нет?"). Признаки иногда используются для предсказания исхода в эксперименте или статистическом исследовании.

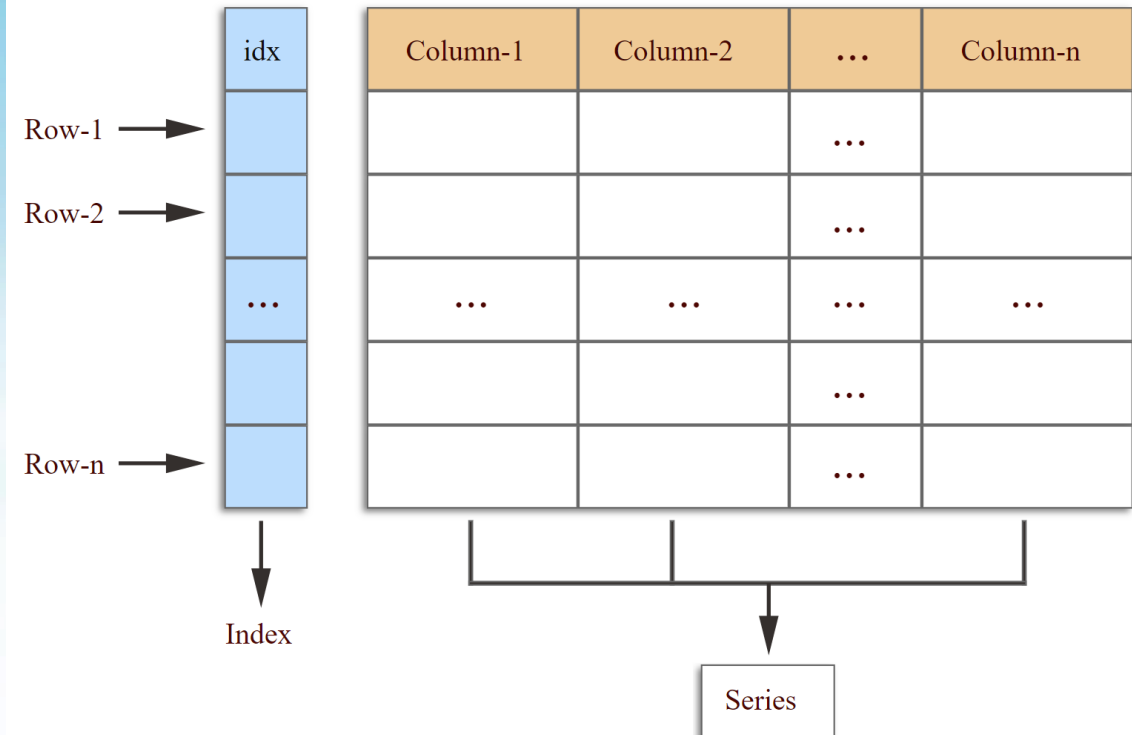
Синонимы: результат, зависимая переменная, отклик, цель, выход.

Записи (records)

Строка в таблице обычно называется записью.

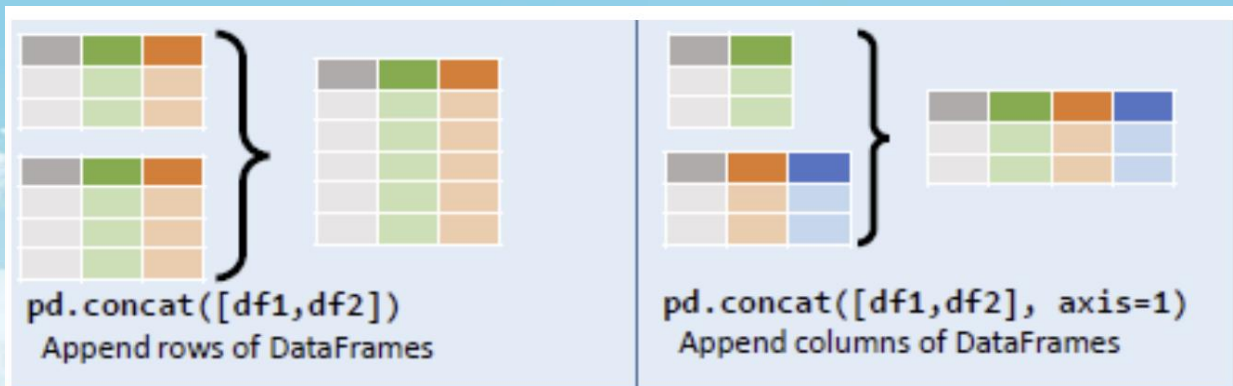
Синонимы: случай, пример, прецедент, экземпляр, наблюдение, шаблон, паттерн, образец.

Pandas Data structure

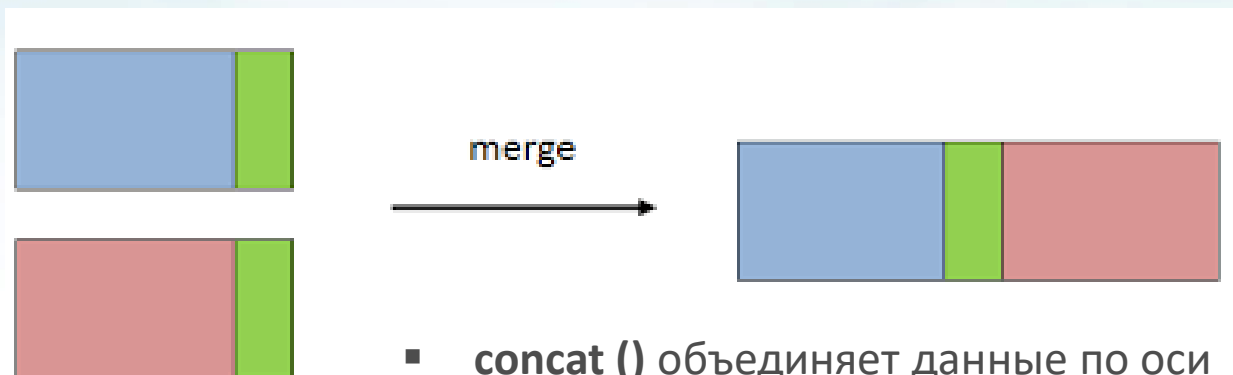


Конкатенация данных в Pandas

pd.concat

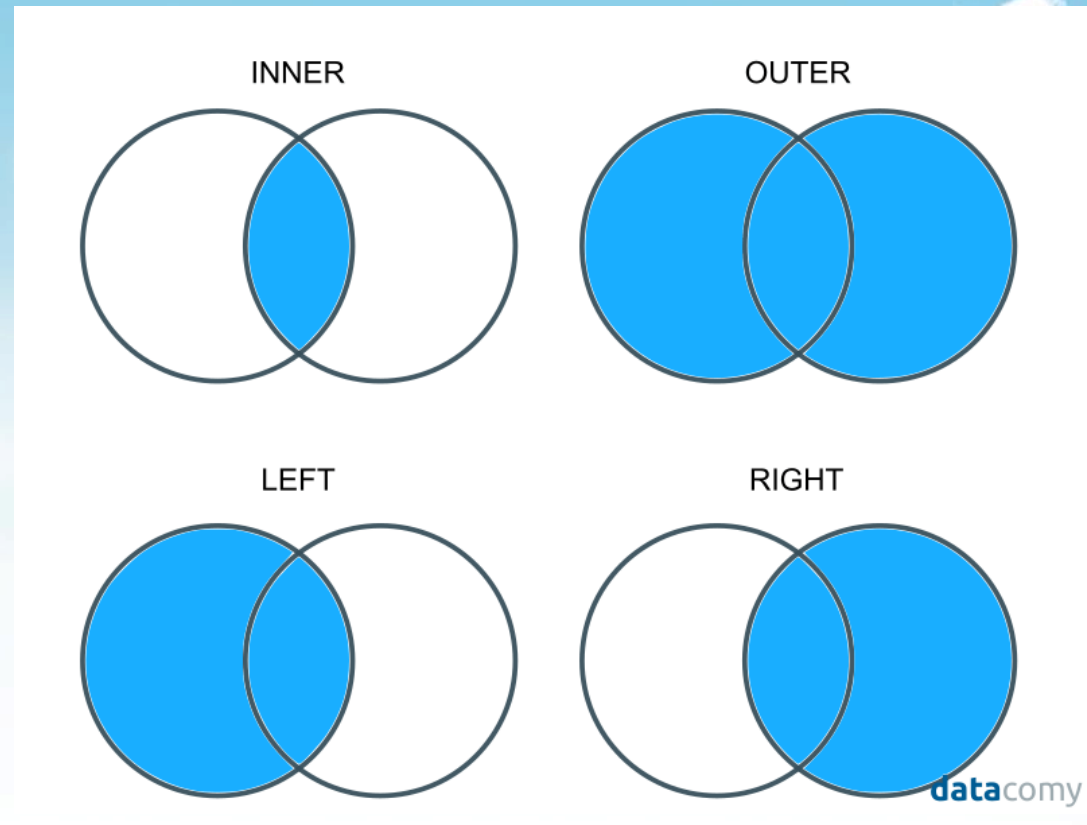


pd.merge

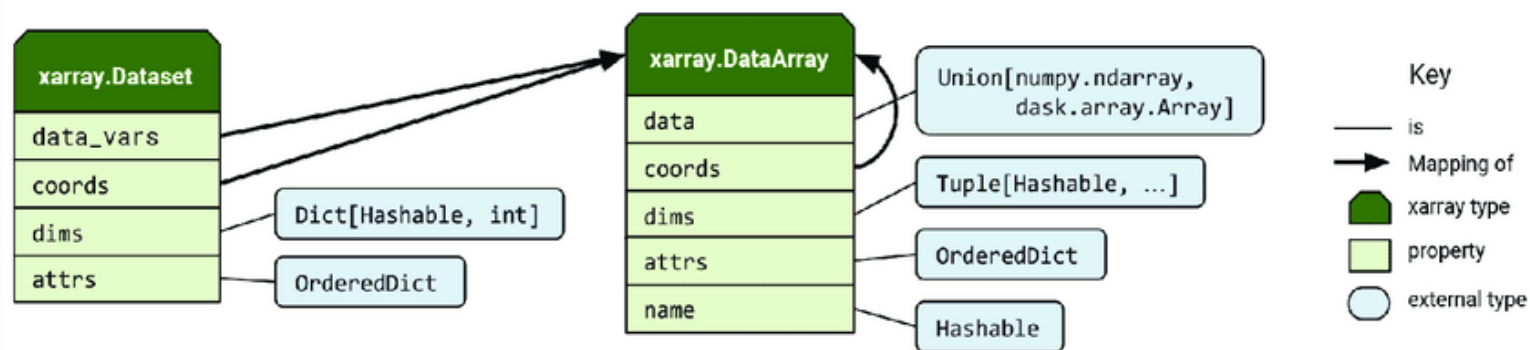
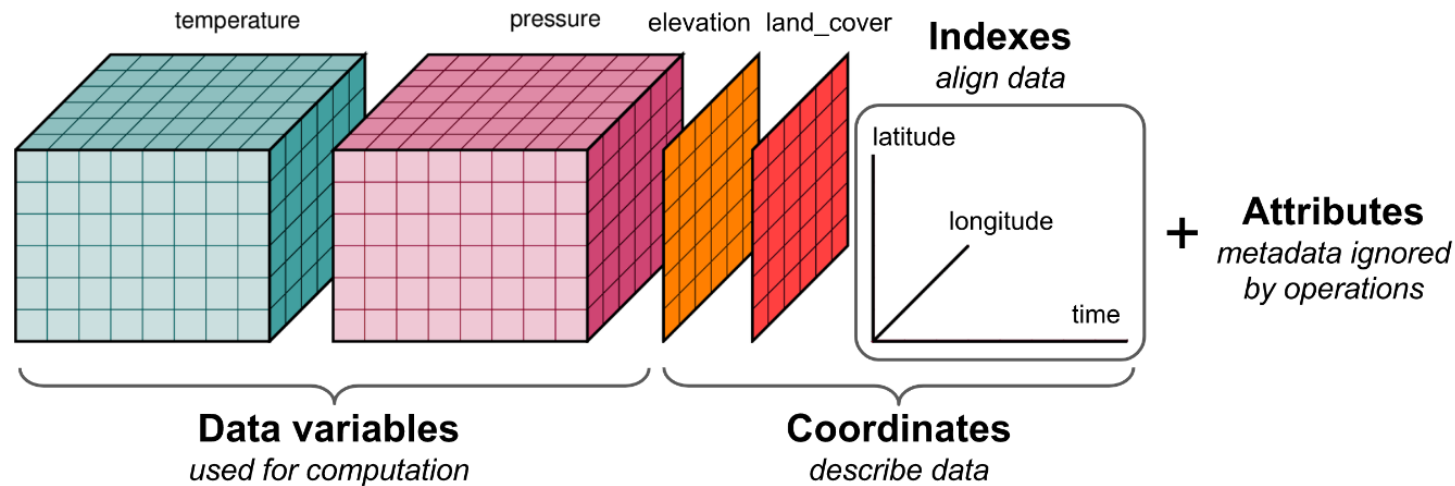


- `concat()` объединяет данные по оси
- `join()` объединяет два кадра данных по индексу.
- `merge()` объединяет два кадра данных по любому указанному столбцу.

DataFrame.join



Многомерные пространственные данные: xarray



Библиотека Xarray

<https://docs.xarray.dev/en/stable/>



Растровые геоданные: rasterio

Библиотека rasterio

File connection

```
src = rasterio.open('data/srtm.tif')
src

<open DatasetReader name='data/srtm.tif' mode='r'>
```

dict
Metadata

```
src.meta

{'driver': 'GTiff',
 'dtype': 'uint16',
 'nodata': 65535.0,
 'width': 465,
 'height': 457,
 'count': 1,
 'crs': CRS.from_epsg(4326),
 'transform': Affine(0.0008333333332777796, 0.0, -113.23958321278403,
 0.0, -0.0008333333332777843, 37.512916763165805)}
```

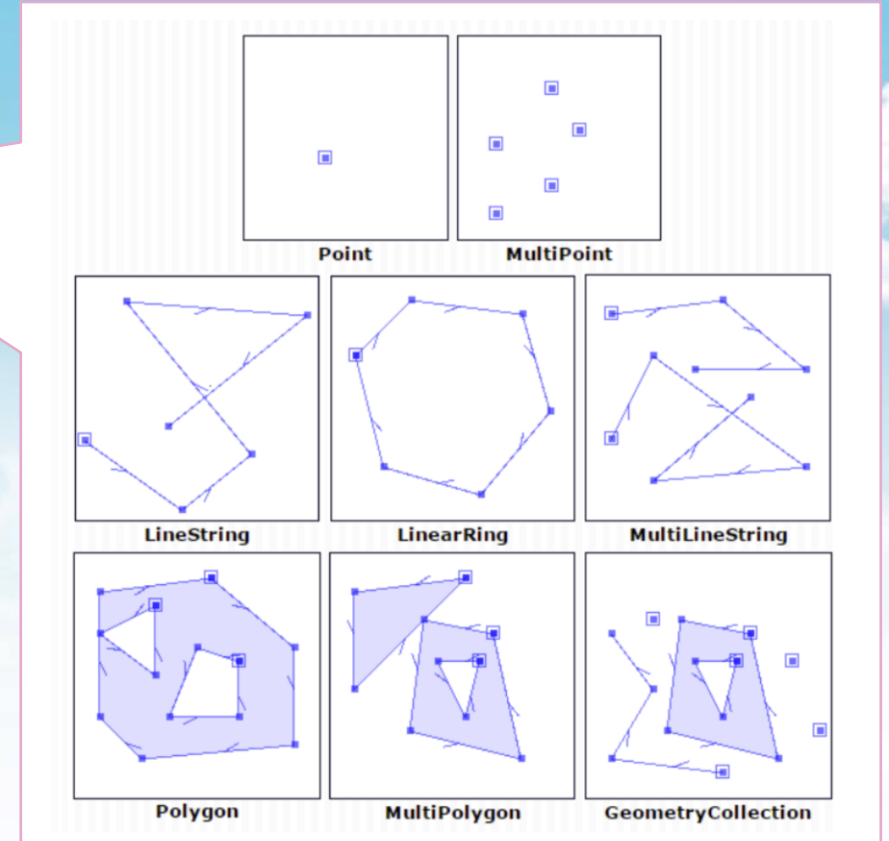
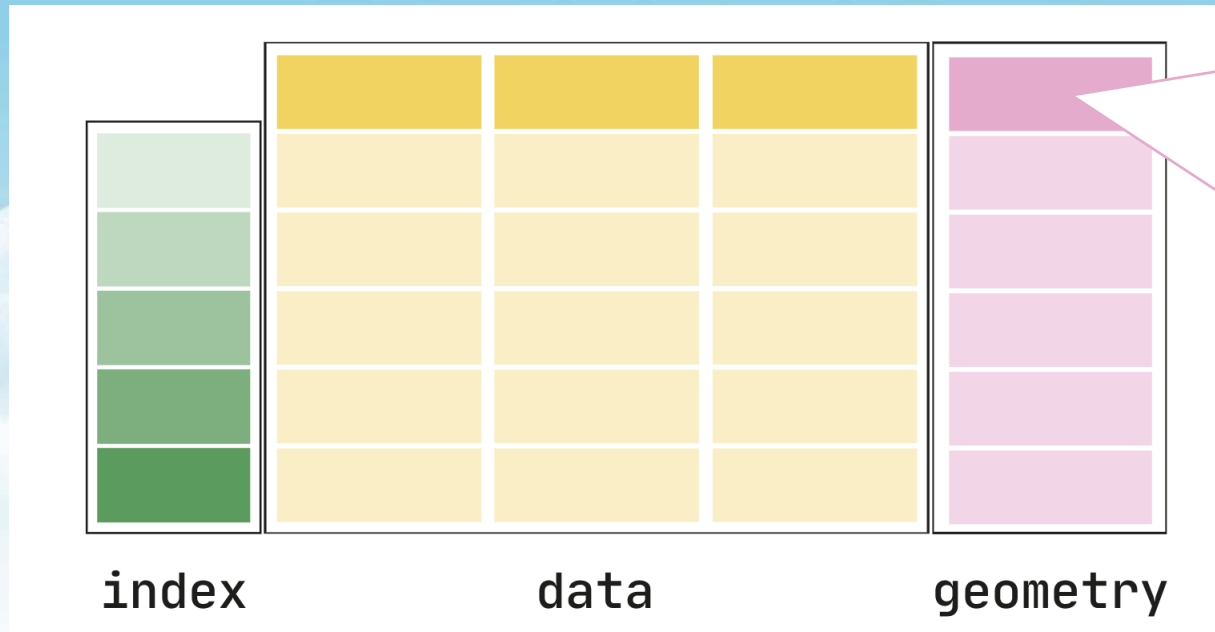
ndarray
Values

```
src.read(1)

array([[1728, 1718, 1715, ..., 2654, 2674, 2685],
       [1737, 1727, 1717, ..., 2649, 2677, 2693],
       [1739, 1734, 1727, ..., 2644, 2672, 2695],
       ...,
       [1326, 1328, 1329, ..., 1777, 1778, 1775],
       [1320, 1323, 1326, ..., 1771, 1770, 1772],
       [1319, 1319, 1322, ..., 1768, 1770, 1772]], dtype=uint16)
```

<https://rasterio.readthedocs.io/en/stable/#>


Векторные геоданные








 Shapely +  pandas =  GeoPandas




<https://geopandas.org/en/stable/>


Репозиторий курса


**UClim4HSE-2025** Public







 Pin  Unwatch 1




 main  1 Branch  0 Tags

 Go to file  Add file  Code

**mvarentsov** Update HW2.md

4a26ae3 · 4 days ago  18 Commits

 homeworks	Update HW2.md	4 days ago
 presentations	L2 presentation	last week
 scripts	HW2 description	4 days ago
 .gitattributes	Initial commit	2 weeks ago
 .gitignore	bug fix	last week
 README.md	Update README.md	4 days ago

 **README**  

UClim4HSE-2025

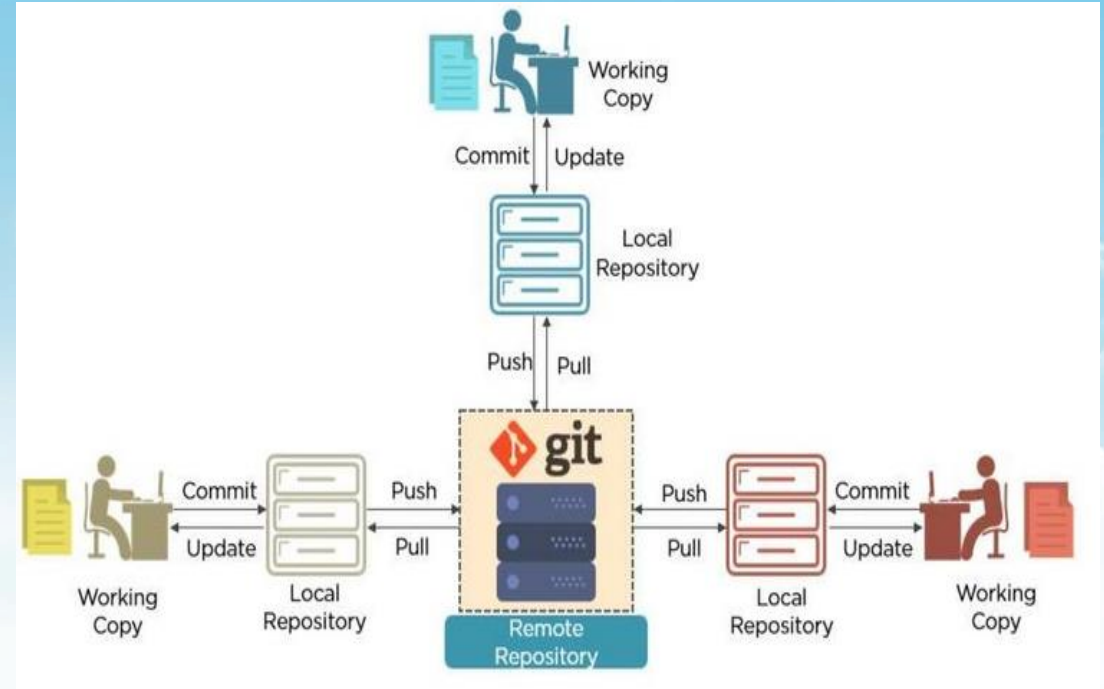
Курс "Моделирование климата городов" для факультета географии и геоинформационных технологий Высшей школы экономики.

Читает [Михаил Варенцов](#) ([Истина МГУ](#) | [ResearchGate](#)). <https://github.com/mvarentsov/UCLIM4HSE-2025/>

[Группа в Telegram](#)

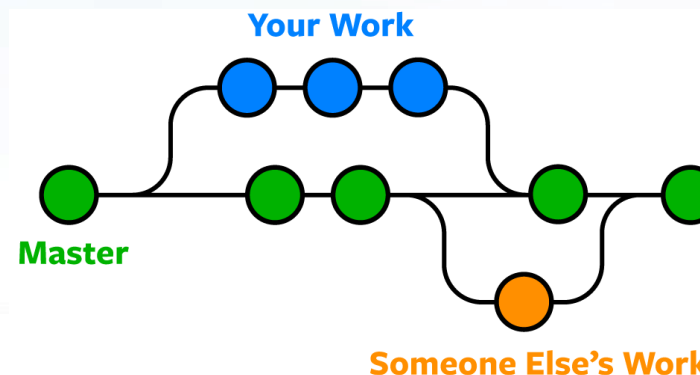
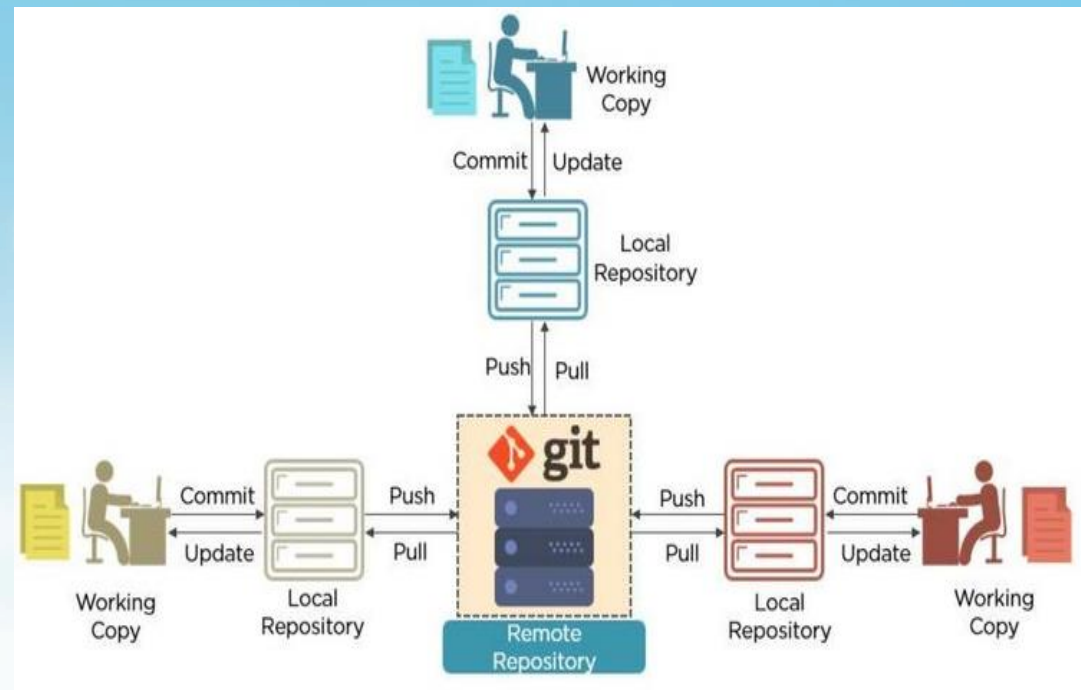
Репозитории Git – что это и зачем?

- ❑ Git - система управления версиями
- ❑ Придумал Линус Торвалдс при разработке ядра Линукс
- ❑ Современный стандарт для совместной разработки, в том числе в научной сфере и Data Science
- ❑ Задачи Git:
 - Синхронизация
 - Резервное копирование
 - Отслеживание и отмена изменений
 - Командная работа



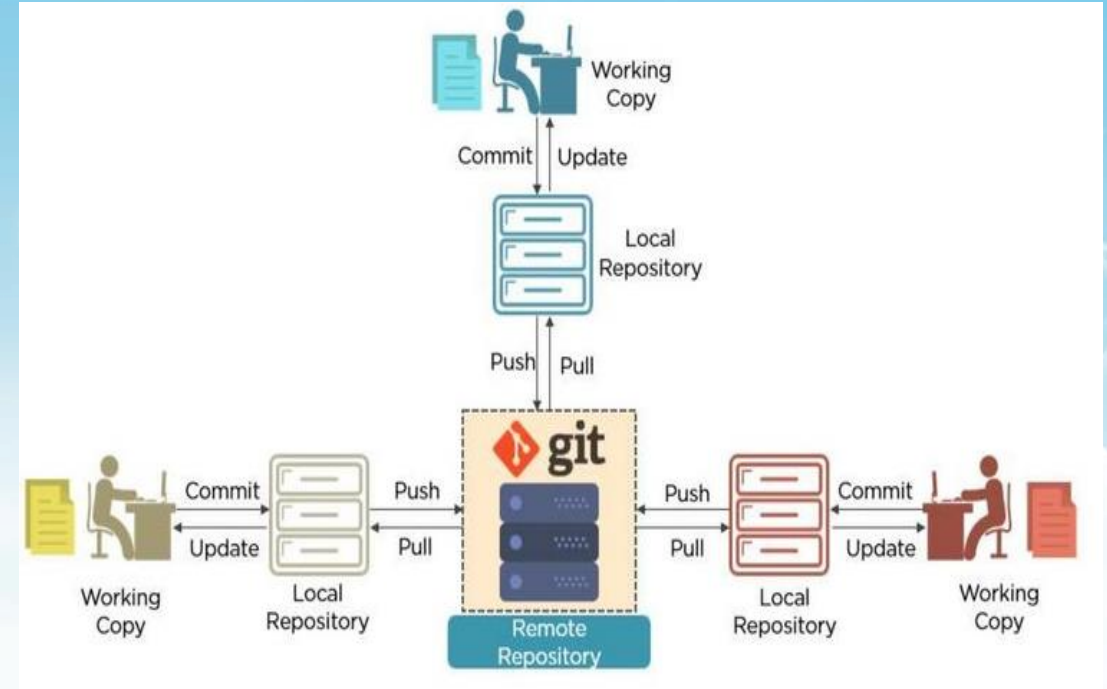
Основные понятия Git

- ❑ **Репозиторий** — место, где хранится код (или данные)
 - Локальный
 - Удаленный (remote)
- ❑ **Коммит (commit)** — зафиксированное и неизменяемое состояние репозитория. Чаще всего их создают, когда:
 - Создан новый функционал
 - Добавлен новый блок на верстке
 - Исправлены ошибки по коду
 - Завершен рабочий день
- ❑ **Ветка (branch)** - независимая последовательность коммитов в хронологическом порядке



Основные команды Git

- ❑ **git clone** – получения локальной копии существующего Git-репозитория
`git clone https://github.com/mvarentsov/Urban-climate-modelling4HSE.git`
- ❑ **git fetch** – загрузка содержимого из удаленного репозитория (без изменения локального репозитория)
- ❑ **git pull = Git fetch + Git merge** – загрузка содержимого из удаленного репозитория и обновление локального репозитория
- ❑ **git add** – добавить файл в список тех, которые отслеживаются для текущего коммита
- ❑ **git commit** – зафиксировать текущие изменения
- ❑ **git push** – выгрузка содержимого локального репозитория в удаленный репозиторий.



Графические интерфейсы для Git

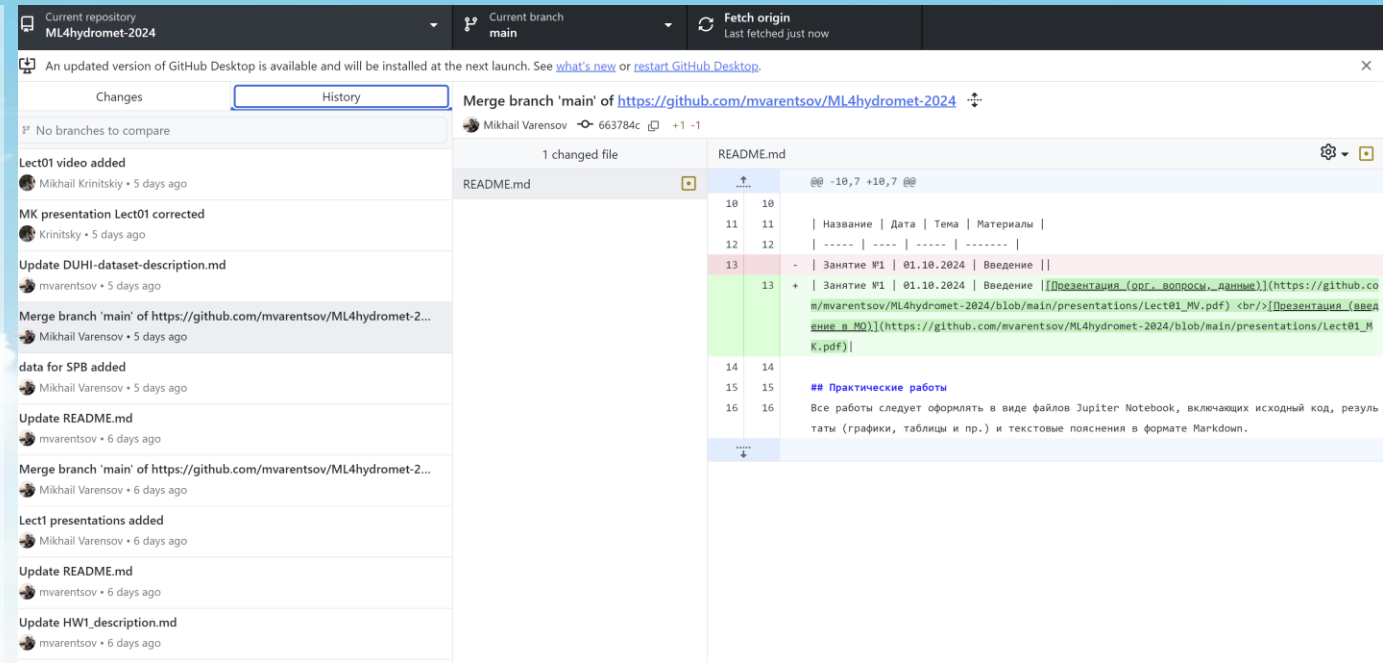
❑ GitHub – облачный Git-сервер

- Альтернатива – создать собственный сервер, например на базе GitLab

❑ GitHub Desktop – графический клиент для GitHub.

❑ Есть и альтернативы

- Sourcetree (Windows, macOS и Linux)
- GitKraken (Windows, macOS)
- ...





Перейдем к практике