Python for Data processing

Lecture 4:
Pandas - Part I

Kosta Rozen

What we already know

A lot about NumPy arrays and PyTorch tensors:

- basic operations
- best practices
- optimization
- linear algebra
- very basics of how machine learning works

Structured and unstructured data

Unstructured data

- images
- signals (including time series)
- text

Each data element (pixel, datapoint, letter) is usually atomic and **is equal** to any other data element. You need to **perform analysis** to get the structure

Structured data

- tabular data
- JSON
- XML

Each data element (row, DB record, XML file) has **internal structure** or **schema**

```
[{‘name’: ‘Anny Smith’, ‘age’: 35, ‘sex’: ‘female’},
 {‘name’: ‘John Black‘, ‘age’: 62, ‘sex’: ‘male’}, ...]
```

Dataframe

Tabular representation of structured data

- well known in R world for years
- **indexed** rows and columns
- SQL-like operations$^{\text{(joins, filtering)}}$, aggregations, alignment and more

Pandas

One of the most respected Python packages for data science

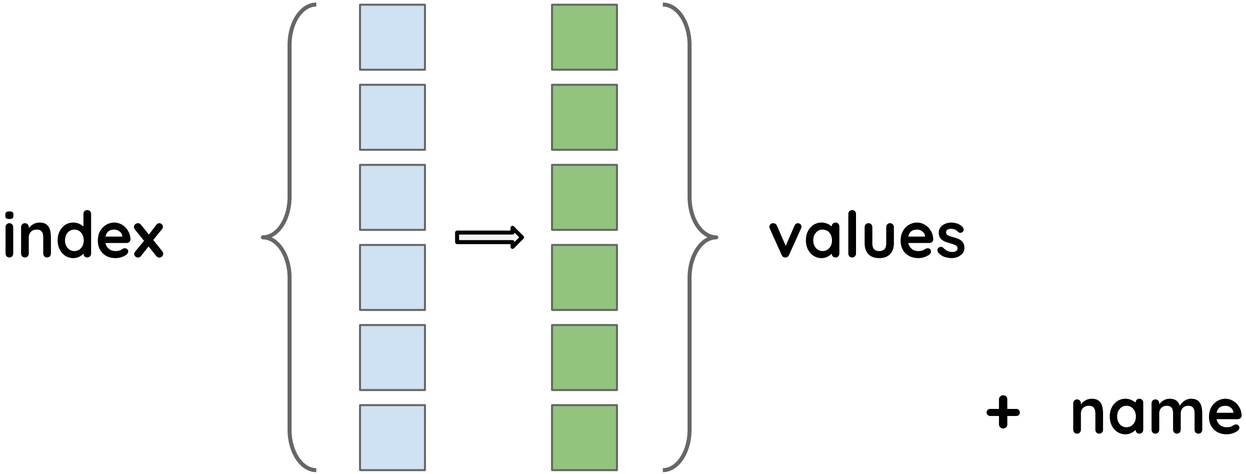
- started in **2008**
- **very fast** (a lot of Cython inside)
- supports **tons of operations and formats**
- extremely **flexible** and **powerful**
- It’s **crazy** sometimes, but you’ll **love** it

Pandas series and df’s

Pandas has two main data structures:

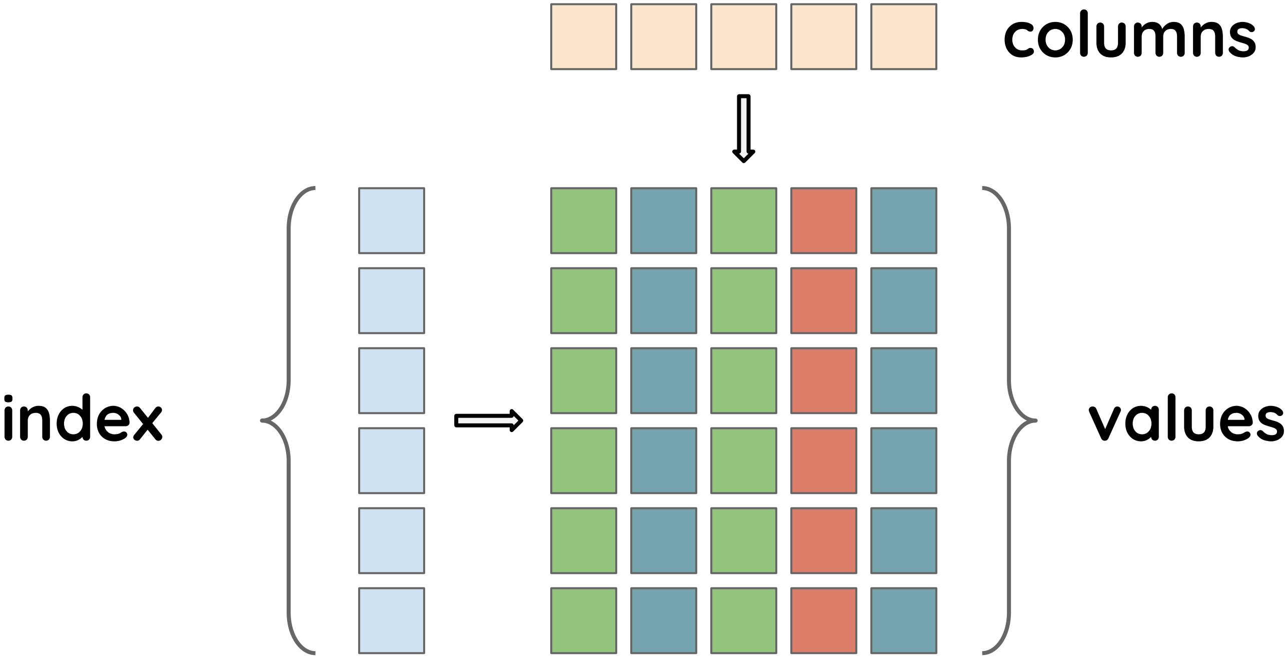
- **pd.Series** for indexed 1D data
- **pd.DataFrame** for indexed tabular data

Pandas series

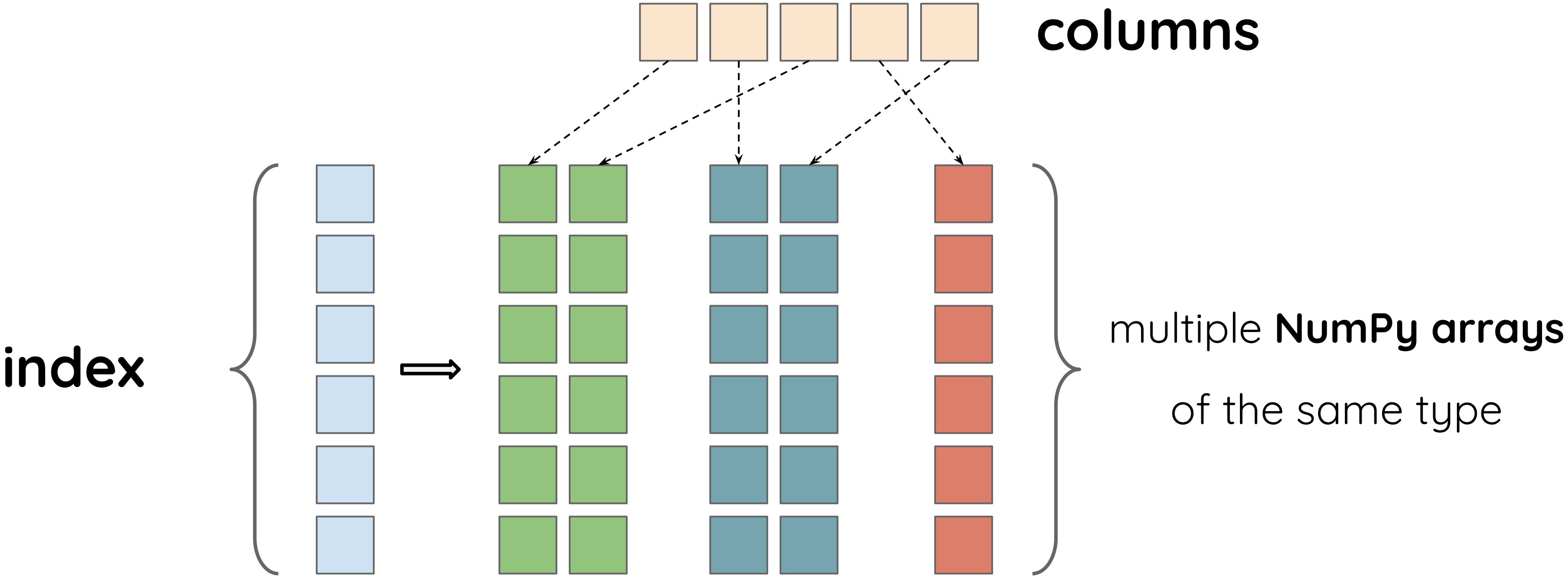


→let's try it out!

Pandas dataframe



Pandas dataframe



Indexing series and dataframes

Reading csv files

The best tool to read CSV and other text files in Python:

```
pd.read_csv(...)
```

→let’s try it out!

Indexing series and df’s

- [] indexing
- **.loc** - label based indexing
- **.iloc** - position based indexing

Boolean indexing is possible and is heavily used

→let’s try it out!

Indexing df: SettingWithCopyWarning

Pandas is not like **numpy**:

- **It’s unknown whether you get view or copy**

Why?

- **It’s hard to give guarantees**
  (but there are rules inside) (but you should not even try to understand them)

→let’s try it out!

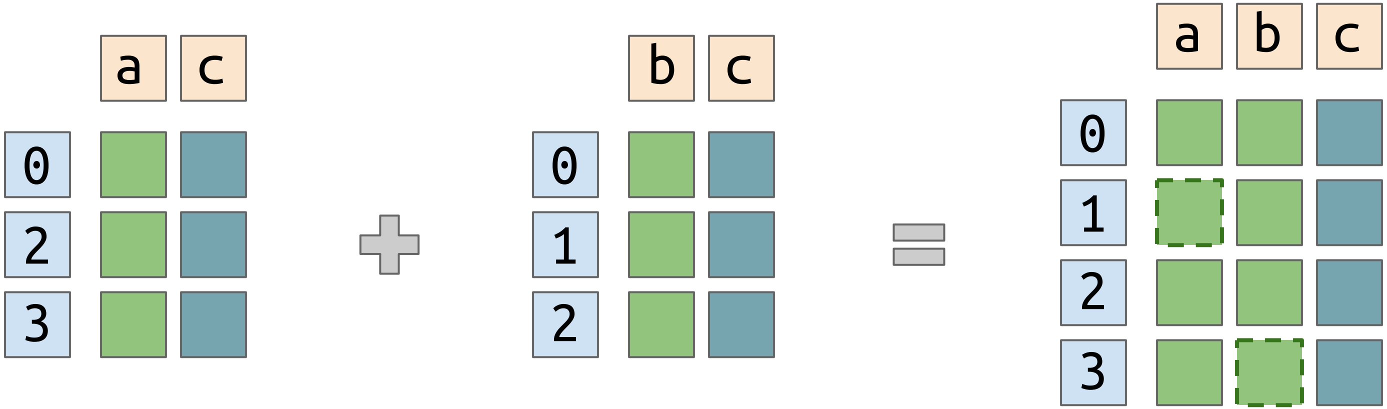
Operations on dataframes

Arithmetic: not your usual NumPy

Pandas **aligns** dataframes for you before performing operations by creating a **union** of row and column indexes

→let’s try it out!

Arithmetic



→let's try it out!

Applying functions to df’s

Pandas allows you to apply custom functions across rows or columns, and elementwise.

And combines results for you appropriately.

It’s usually **fast**.

→let’s try it out!

Dataframe summaries

It’s easy to get general information about dataframe:

df.info()

df.describe()

df.head(), df.tail()

→let’s try it out!

Counts and statistics

To get counts or statistics about column or row:

df[col].unique()

df[col].value_counts()

df.sum(axis=...), df.mean(axis=...), df.std(axis=...)

Powerful in combination with smart indexing.

→let’s try it out!

Replacing and renaming

df.replace allows for flexible replacement of values in dataframes:

- by value, per column

df.rename allows you to easily rename any label, be it column name or index label

→let’s try it out!

Missing data

Pandas is great at handling missing data:

- infers it for you
- backward fill, forward fill and more

→let’s try it out!

Categorical data

Pandas easily calculates one-hot encoded values for any column, adding properly named columns

→let’s try it out!

Special datatypes

Pandas has very good support for:

- **strings** - great for text columns
  (split, replace and other usual string operations, vectorized)
- **datetime** - flexible indexing, handling timezones and extravagant parsing
  (great for anything time series related)

→let's try it out!

What we’ve learned

Basics of Pandas:

- creating, indexing
- operations on dataframes

Assignment

- Explore Pandas
- play with Titanic dataset
- 2 weeks!

questions?