Data 100

Spring 2018

Lecture 3

Data Tables, Indexes, pandas

Slides created by Sam Lau (samlau95@berkeley.edu), Sp2018 updates by Fernando Perez

Announcements

HW1 out

Where we are

Data Science Lifecycle

- Ask question(s)

- Obtain data

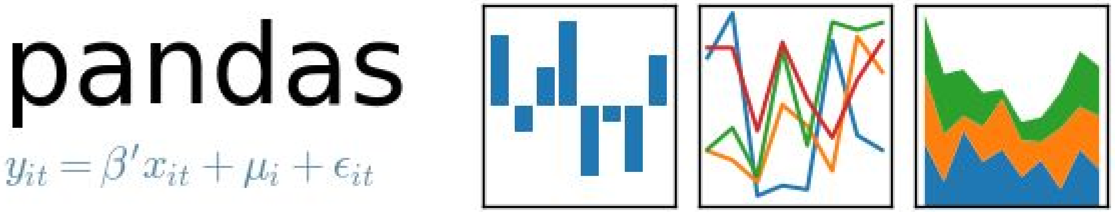
- Understand the data

- Understand the world

Data Science Lifecycle

- **Ask question(s)**	- **Your brain**
- **Obtain data**	- **The Internet**
- **Understand the data**	- **pandas and EDA**
- Understand the world	- Inference and prediction

Today: pandas



http://pandas.pydata.org/

How this lecture will work

- Using the dataset of baby names, we will...
- Ask questions
- Break down each question into steps
- Learn the pandas knowledge needed for each step

What you will learn

- Data manipulation in pandas
  - Sorting, filtering, grouping, pivot tables

- Data visualization in pandas and seaborn
  - Bar charts, histograms, scatter plots

- Prior knowledge of all concepts assumed!
  - ~3 weeks of Data 8 in 1.5 hours
  - Practical, not conceptual

You won’t remember everything, but...

Getting the data

(Demo)

Question 1:
What was the most popular name in CA last year?

(2-min discussion)

Always have high-level steps

1. Read in the data for CA	1. `Table.read_table`
2. Keep only year 2016	2. `Table.where`
3. Sort rows by count	3. `Table.sort`

In pandas

1.	Read in the data for CA	1.	`pd.read_csv`
2.	Keep only year 2016	2.	Slicing
3.	Sort rows by count	3.	`df.sort_values`

(Demo)

Recap

- `pd.read_csv(...)` => DataFrame
  - DataFrame is like the Data 8 Table
  - Series is like a NumPy array

- Slice DFs by label or by position
  - `df.loc` and `df.iloc`
  - DF index is a label for each row, used for slicing

- `df.sort_values(...)` like `Table.sort`

Question 2:
What were the most popular names in each state for each year?

(2-min discussion)

Break it down

1. Put all DFs together	1. `pd.concat`
2. Group by state and year	2. `df.groupby`

(Demo)

Recap

- `zipfile`
  - Work with compressed archives efficiently in-memory
- `df.groupby(...).agg(...)`
  - Groups one or more columns, applying aggregate function on each group
- `df.groupby(...).sum() # or .max(), etc.`
  - Shorthand for `df.groupby(...).agg(np.sum)`

When do I need to group?

- Do I need to count the times each value appears?

- Do I need to aggregate values together?

- Am I looping through a column’s unique values?

Question 3:
Can I deduce gender from the last letter of a person’s name?

Survey Question

Which last letter is most indicative of a person’s birth sex?

bit.ly/ds100-sp18-c7a

1. g
2. m
3. t
4. z
5. e
6. This is a trick question!

Break it down

1. Compute last letter of each name
2. Group by last letter
3. Visualize distribution

1. `series.str`
2. `df.groupby`
3. `df.plot`

(Demo)

Recap

- `series.str`
  - To use string methods
  - Use `series.apply` when you need flexibility

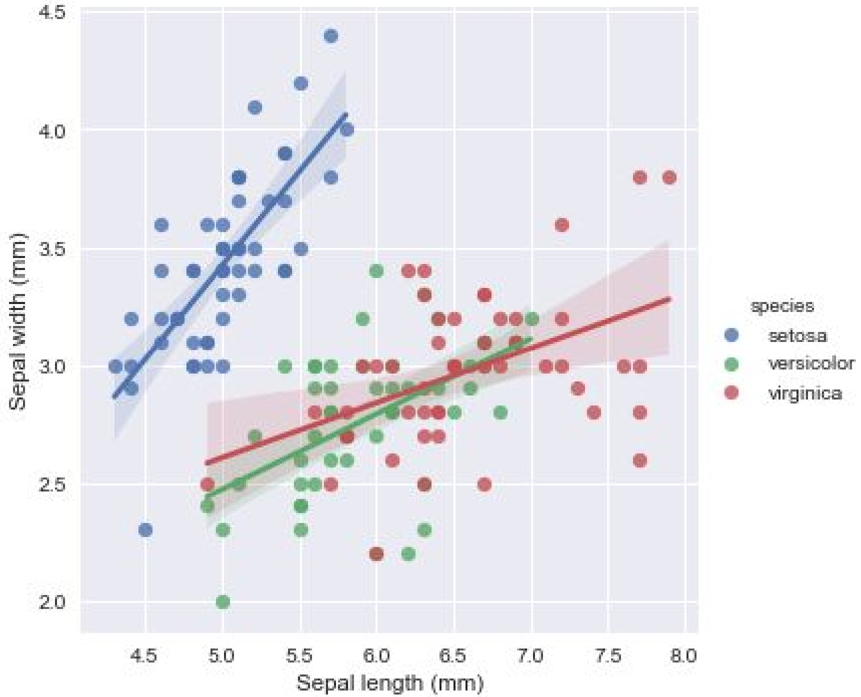
- `df.pivot_table(...)`
  - Computes a pivot table

- `df.plot`
  - To use plotting methods

When do I need to pivot?

- Am I grouping by two columns...
- And do I want the resulting table to be easier to read?
- Or, am I using pandas plotting on the groups?

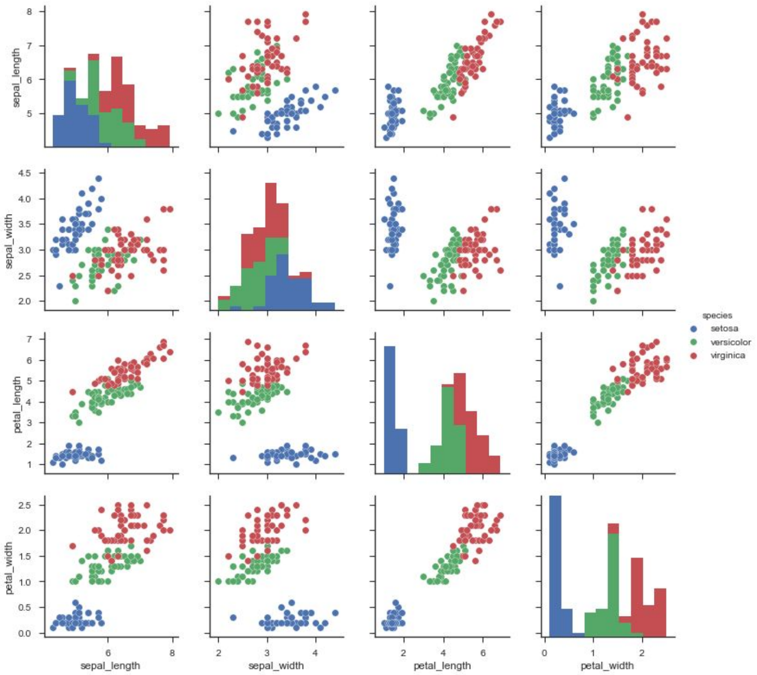
Seaborn



http://seaborn.pydata.org/index.html

Seaborn

- Statistical data visualization
- Has common plots with some bonus features
  - And some fancier plots too
- Works well with pandas DataFrames



```
sns.pairplot(df, hue="species")
```

How to Seaborn

- DataFrame should ideally be in long-form (not grouped)

- Most Seaborn methods work like this:

```
sns.barplot(x=..., y=..., hue=..., data=df)
```

(Demo)

Recap

- Pandas for tabular data manipulation
  - Slicing for row/column selection
  - Group with `df.groupby`
  - Pivot with `df.pivot_table`
  - Join with `pd.merge` (covered in lab next week)
  - `df.plot` for basic plots

- Seaborn for statistical plots
  - Reference the docs for available methods

Use the docs!

And Google.