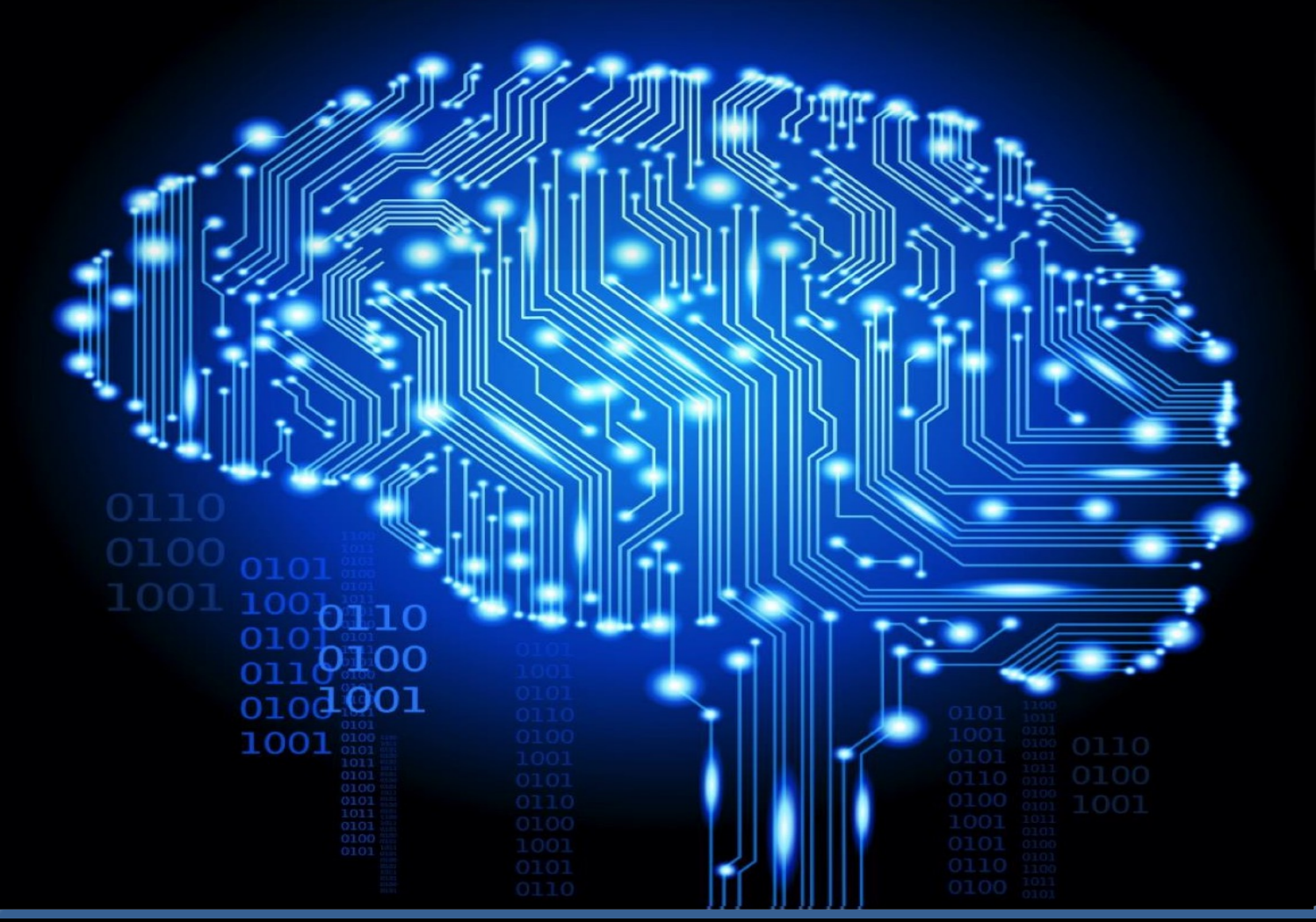
Statistical Learning and Computational Finance Lab.
Department of Industrial Engineering
http://slcf.snu.ac.kr



Lab Session 2:Data Visualization and Clustering

Reference

- Official docs of Matplotlib, seaborn, plotly, and scikit-learn
- Top 6 Python Libraries for Visualization: Which one to Use?
  - https://towardsdatascience.com/top-6-python-libraries-for-visualization-which-one-to-use-fe43381cd658
- 7 Key Principles of Effective Data Visualization
  - https://medium.com/gobeyond-ai/7-key-principles-of-effective-data-visualization-b854b0b81946

Agenda

- Preview for Data Visualization and Clustering Exercise
- Python Data Visualization Libraries
- Visualization examples with Titanic Dataset
- Clustering Examples with Iris Dataset
- Exercise with Kaggle Dataset

Preview for Data Visualization and Clustering Exercise

Image Source: https://www.searchenginejournal.com/data-visualizations-seo-reporting/434909/#close

What is Effective Data Visualization?

- By using graphical or pictorial representations of data, businesses find it easier to analyze information, draw conclusions, and address issues in a timely manner.
- 7 Key principles of effective data visualization

1. Determine the best visual
2. Balance the design
3. Focus on the key areas
4. Keep in simple
5. Incorporate interactivity
6. Use patterns
7. Compare aspects

Types of Variables

- Categorical
  - ex) Sex, Blood Type, MBTI
- Numerical
  - Discrete
    - Integer values (Not always)
    - ex) Year, Age
  - Continuous
    - Float values
    - ex) Height, Weight
- String(Text)
  - ex) Name, Comments
- Chronological
  - ex) Date, Time
- Image
- URLs

and so on…

Why should we consider variable types?

- Way to convey the message and extract insights from data should be different by variable types
  - Can you think of examples?
- Along with data pre-processing, you should be able to deal with different data types

What are some other things to consider?

- Color Coding
  - To make your work more interpretable and ‘visible’, you must be careful when choosing color combinations
    - Python libraries offer some color palettes, so just use them!
    - Tip. Please keep in mind that you always have to make colorblind friendly palettes!
      - https://davidmathlogic.com/colorblind/#%23D81B60-%231E88E5-%23FFC107-%23004D40

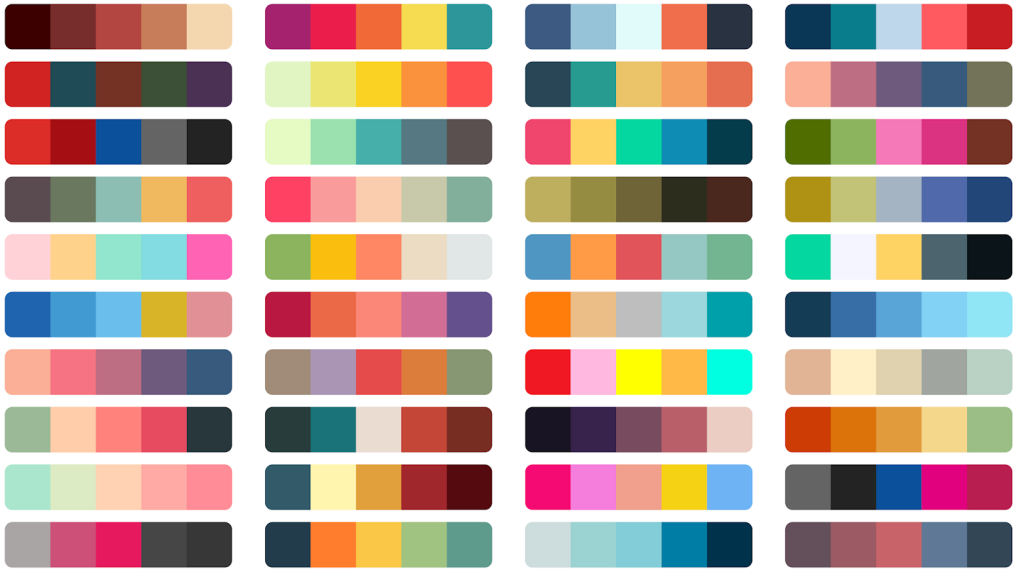


Image Source: https://www.flerlagetwins.com/2021/06/datafam-colors-color-palette.html

Overall,

- Data visualization itself is a huge research area
- If you are interested, there are many articles and blog posts about this topic so please check them out!
  - Keywords
    - Data Visualization
    - Information Visualization
    - Business Intelligence
      - Tableau, Power BI

Python Data Visualization Libraries

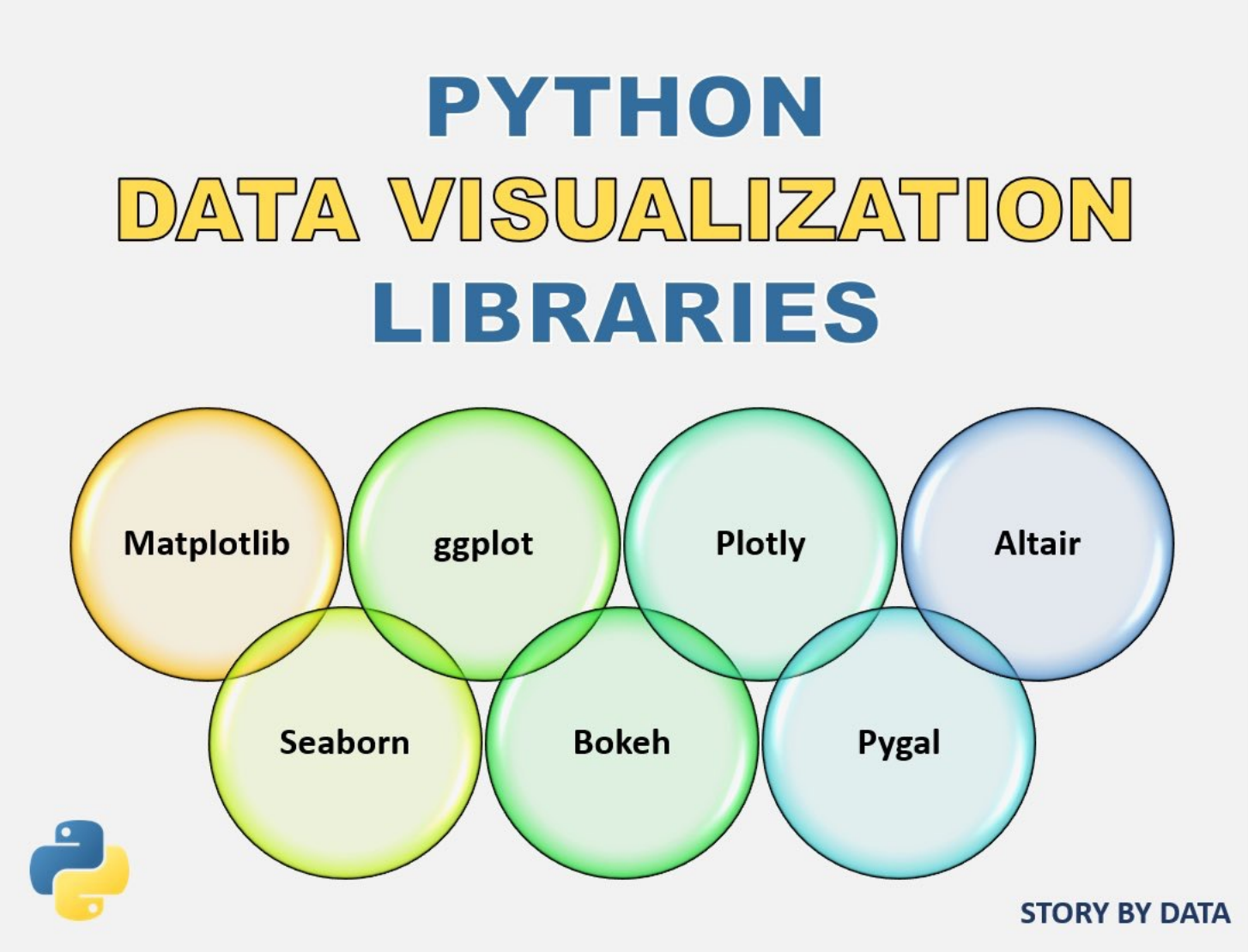


Image Source: https://twitter.com/datacated_/status/1166337648341991424

Famous Libraries for Data Visualization

- Matplotlib
- Seaborn
- Plotly
- Bokeh
- Pygal
- Altair
- and so on…

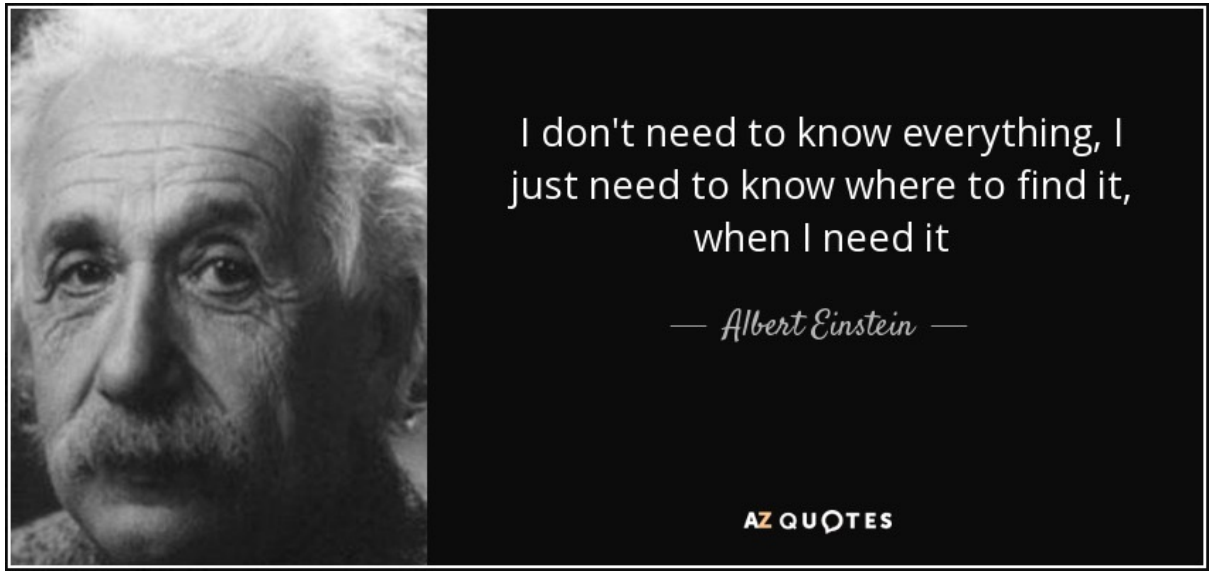


Image Source: https://www.azquotes.com/quote/457953

Matplotlib

- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
- It's designed to resemble figures made with MATLAB
- Go-to library for visualization

Seaborn

- Seaborn is a Python data visualization library based on Matplotlib.
- It provides a high-level interface for drawing attractive and informative statistical graphics.
- Users can set themes, apply pre-set color combinations, etc…
  - It helps us to generate fancier plots compared to Matplotlib

Plotly

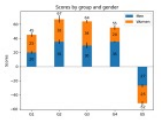
- Plotly's Python graphing library makes interactive, publication-quality graphs.
- It easily generates interactive plots with just a few lines of code.
- Highly compatible with Dash(framework for building data apps)

Pros and Cons

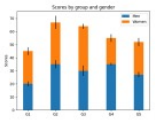
- Matplotlib
  - Pros: Easy to see the property of data, Highly customizable
  - Cons: Extremely low-level interface, Doesn't look that nice
- Seaborn
  - Pros: Less code required, Make common-used plots prettier
  - Cons: Constrained in uses (less collection than Matplotlib)
- Plotly
  - Pros: Easy to create interactive and complex plots
  - Cons: A bit heavy

Example Gallery

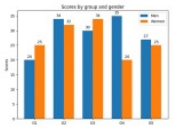
Matplotlib



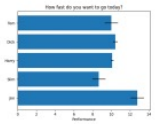
Bar Label Demo



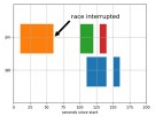
Stacked bar chart



Grouped bar chart with labels

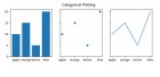


Horizontal bar chart

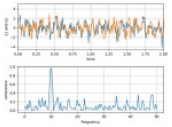


Broken Barh

CapStyle

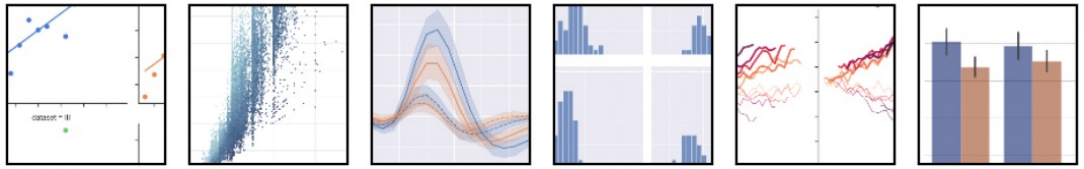


Plotting categorical variables



Plotting the coherence of two signals

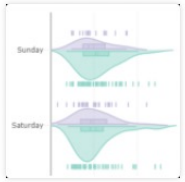
Seaborn



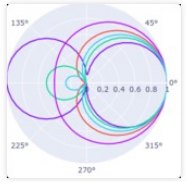
Plotly

Fundamentals

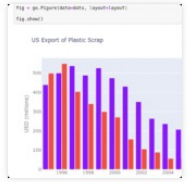
More Fundamentals »



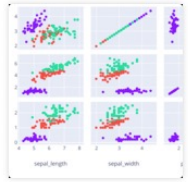
The Figure Data Structure



Creating and Updating Figures



Displaying Figures

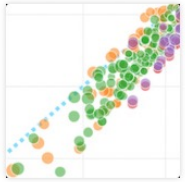


Plotly Express

Analytical Apps with Dash

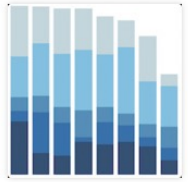
Basic Charts

More Basic Charts »

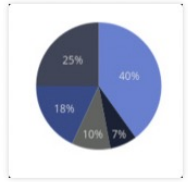


Scatter Plots

Line Charts



Bar Charts



Pie Charts

Bubble Charts

Tips from TA

- Literally, no one remembers every single method in python libraries
  - Make your own cheatsheet
  - Practice googling skills (THE MOST IMPORTANT SKILL)
  - Run tutorial codes from official websites
  - Search for awesome projects in github
    - https://github.com/javierluraschi/awesome-dataviz
    - https://visme.co/blog/best-data-visualizations/
    - etc…

Visualization examples with Titanic Dataset

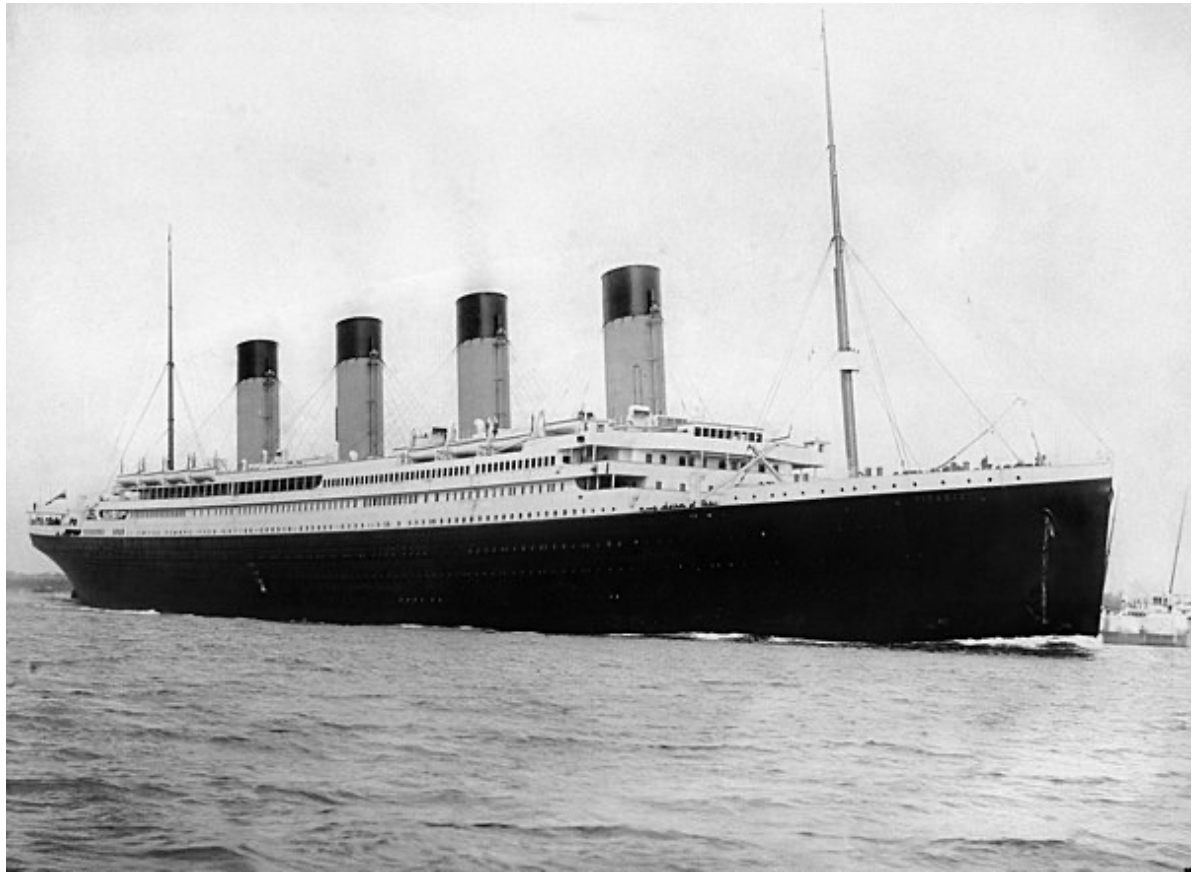


Image Source: https://en.wikipedia.org/wiki/Titanic

Data Dictionary

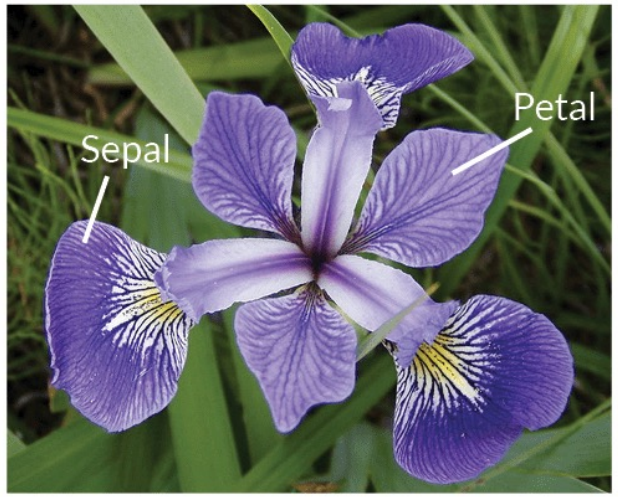
Column info:

- survived: Survival(0 = No, 1 = Yes)
- pclass: Ticket Class (1 = 'First', 2 = 'Second', 3 = 'Third')
- sex: Sex
- age: Age in years
- sibsp: # of siblings / spouses aboard the Titanic
- parch: # of parents / children aboard the Titanic
- fare: Passenger fare
- embarked: Port of embarkation (C = 'Cherbourg', Q = 'Queenstown', S = 'Southampton')
- class: Same info with 'pclass' but in Categorical form
- who: Age + Sex
- adult_male: Same info with 'sex' but in Boolean form
- deck: Passenger deck
- embark_town: Same info with 'embarked' but each city in its full name
- alive: Same info with 'survived' but in text form
- alone: Whether the passenger was aboard alone

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 15 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   survived     891 non-null    int64
 1   pclass       891 non-null    int64
 2   sex          891 non-null    object
 3   age          714 non-null    float64
 4   sibsp        891 non-null    int64
 5   parch        891 non-null    int64
 6   fare         891 non-null    float64
 7   embarked     889 non-null    object
 8   class        891 non-null    category
 9   who          891 non-null    object
 10  adult_male   891 non-null    bool
 11  deck         203 non-null    category
 12  embark_town  889 non-null    object
 13  alive        891 non-null    object
 14  alone        891 non-null    bool
dtypes: bool(2), category(2), float64(2), int64(4), object(5)
memory usage: 80.7+ KB
```

Time to code~

Clustering with Iris Dataset



Iris Versicolor

Iris Setosa

Iris Virginica

Image Source: http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html

Recall: Clustering

- According to Wikipedia,
  - **Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar (in some sense) to each other than to those in other groups (clusters).
  - It is a main task of exploratory data analysis, and a common technique for statistical data analysis, used in many fields, including pattern recognition, image analysis, information retrieval, bioinformatics, data compression, computer graphics and machine learning.
  - It can be achieved by various algorithms that differ significantly in their understanding of what constitutes a cluster and how to efficiently find them.
    - Examples of algorithms: K-Means, Affinity Propagation, Mean-shift, DBSCAN, Gaussian Mixtures

K-Means Clustering

- The K-Means algorithm clusters data by trying to separate samples in n groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares
  - requires the number of clusters to be specified
  - scales well to large number of samples
- It aims to choose centroids that minimize the inertia

$$\sum_{i=0}^{n} \min_{\{\mu_j \in C\}} \left(\left| \left| x_i - \mu_j \right| \right|^2 \right)$$

How K-Means Clustering works

- Step 1: Choose the initial centroids
  - Randomly choose initial centorids with k data points (without replacement)
- Step 2: Looping between two other steps
  - Step 2-1. Assign each sample to its nearest centorid
  - Step 2-2. Create new centroids by taking the mean value of all samples assigned to each previous centroid
  - Repeats step 2 until the centroids do not move significantly

K-Means Clustering Examples

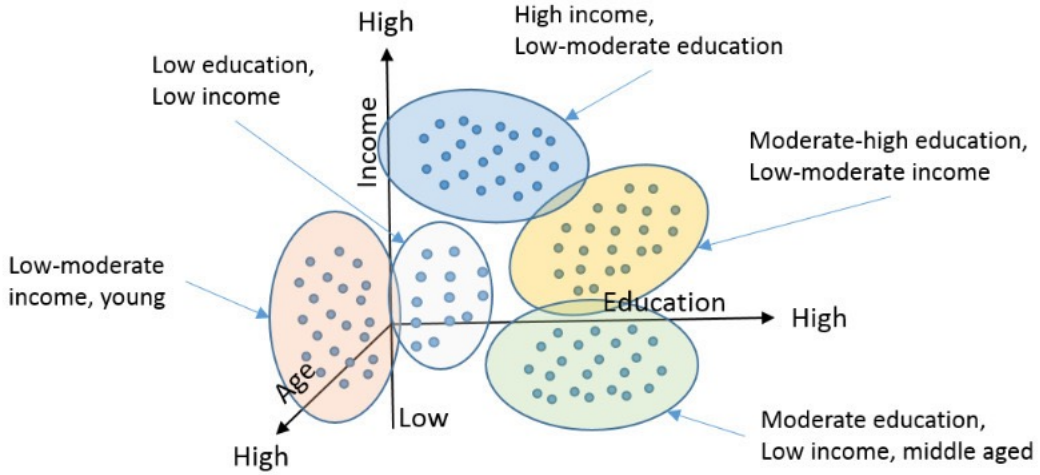


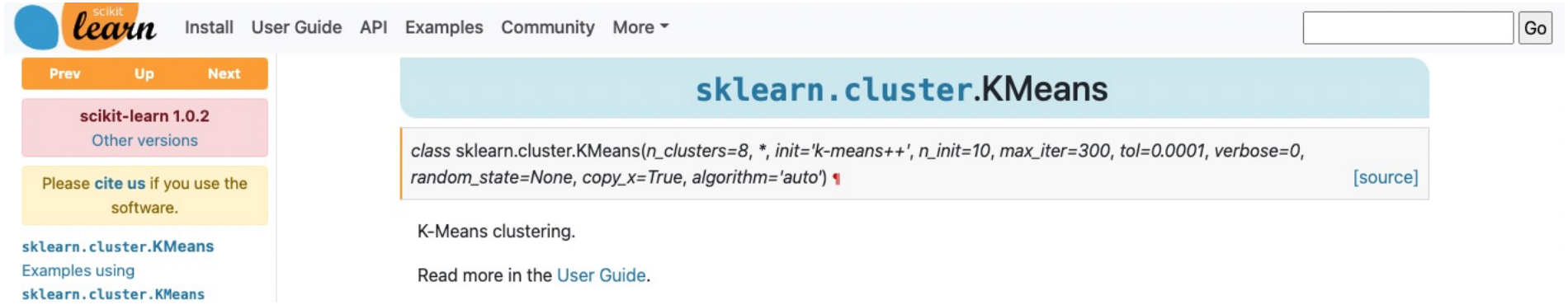
Image Source:
- (Left) https://www.kdnuggets.com/2019/08/introduction-image-segmentation-k-means-clustering.html
- (Right) https://www.researchgate.net/figure/Example-3-of-K-means-clustering-using-R_fig1_313066371

Gaussian Mixture

- A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.
  - One can think of mixture models as generalizing k-means clustering to incorporate information about the covariance structure of the data as well as the centers of the latent Gaussians.

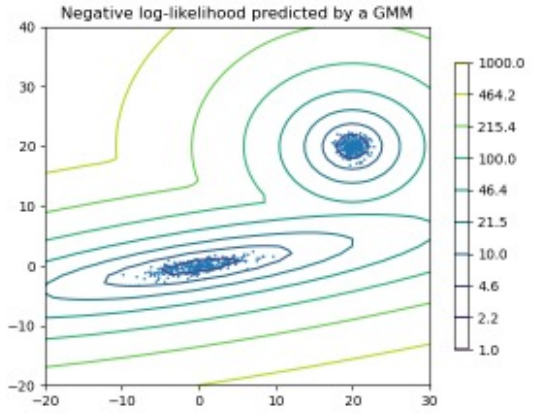
Clustering with Python

- Of course, you can implement K-Means algorithm by yourself
  - But who does that?
- Let's use KMeans module in scikit-learn
  - User Guide: https://scikit-learn.org/stable/modules/clustering.html#k-means



Clustering with Python

- If you want to use Gaussian Mixture models, use sklearn.mixture package
  - sklearn.mixture is a package which enables one to learn Gaussian Mixture Models (diagonal, spherical, tied and full covariance matrices supported), sample them, and estimate them from data. Facilities to help determine the appropriate number of components are also provided.



Two-component Gaussian mixture model: *data points, and equi-probability surfaces of the model.*

Let’s practice with Iris dataset and randomly generated data points!

Exercise with Kaggle Dataset

Image Source
- Top: https://www.introducingnewyork.com/taxis
- Bottom: https://www.daysoftheyear.com/days/penguin-awareness-day/