Lecture Notes for

Machine Learning in Python

⟦👨‍🏫, 👨‍💻, 🐍, 👨‍🔬⟧

Professor Eric Larson

Preprocessing and Visualization

Class Logistics and Agenda

- Participation (Quizzes) / Teams
- Be sure you look at **Lab One**!
- Dataset Selection Now Complete! Probably! … maybe?
- Office Hours conflict with faculty meeting
- Agenda
  - Finish Pandas Demo with Imputation, *if needed*
  - Data Exploration
  - Data Preprocessing
  - Data Visualization

Class Overview, by topic

Table Data Visualization

Numpy, Pandas, Seaborn
Overviews with some in-depth discussion

Dimension Reduction and Image Processing

Scikit-learn, Scikit Image,
Intuition only, Some mathematics

Linear and Logistic Regression

Numpy, Recreate API for Scikit-learn
Detailed mathematics for simple optimization
intuition for advanced optimization

Neural Networks and Back Prop.

Numpy
Detailed mathematics for NN operations

Wide and Deep Networks

Convolutional Networks

Recurrent Networks

Keras, Tensorflow
Intuition, Detailed implement.

Ethics in Language Models

ConceptNet
Case studies

Last Time

- Datatypes
- Imputation
- *Some Document Features*

K-Nearest Neighbors Imputation

TID	Pregnant	BMI	Age	Diabetes
1	Y	33.6	31-40	positive
2	N	26.6	31-40	negative
3	Y	23.3	?	positive
4	?	28.1	21-30	negative
5	N	43.1	31-40	positive
6	Y	25.6	21-30	negative
7	Y	31.0	21-30	positive
8	Y	35.3	?	negative
9	N	30.5	51-60	positive
10	Y	37.6	21-30	positive

For $k = 3$, find 3 closest neighbors

TID	Preg.	BMI	Age	Diabetes	Distance d_k
3	Y	23.3	?	positive	0
6	Y	25.6	21-30	negative	(0 + 2.3 + 1)/3
2	N	26.6	31-40	negative	(1 + 3.3 + 1)/3
4	?	28.1	21-30	negative	(4.8 + 1)/2

… repeat for all rows, select 3 closest …

Imputed Age: 21-30

$$= \frac{1}{|F_{valid}|} \sum_{i \in F_{valid}} \| f_i - f_i^{(k)} \|$$

i^{th} feature, f_i in row

Distance can be calculated differently:

- Difference for valid features only
- May need to normalize ranges
- Weight neighbors differently?
- Have min # of valid features?
- Type: Euclidean, city-block, etc.

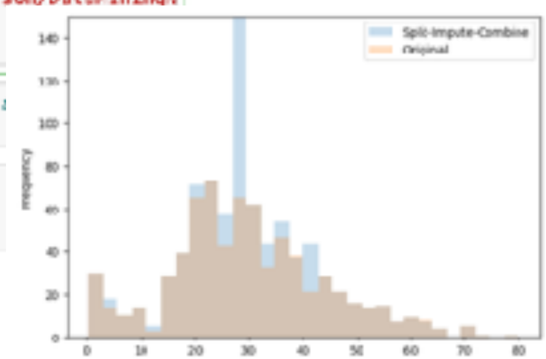
44

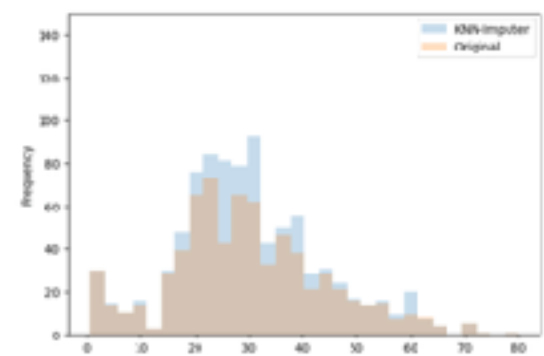
Loading the Titanic Data for Example Visualizations

```
# load the Titanic dataset
import pandas as pd
import numpy as np

print('Pandas:', pd.__version__)
print('Numpy:',np.__version__)

df = pd.read_csv('https://raw.githubusercontent.com/eclarson/DataMiningN

df.head()
```

```
# note that the describe function defaults to using only
df.describe()
```

```
print(df.dtypes)
print('===========')
print(df.info())
```





Demo

"if needed"
Pandas demo

- DataFrames
- Loading
- Indexing
- Imputing

`03.Data Visualization.ipynb`

Data Exploration

Somebody forgot to impute, 🫣

What is data exploration?

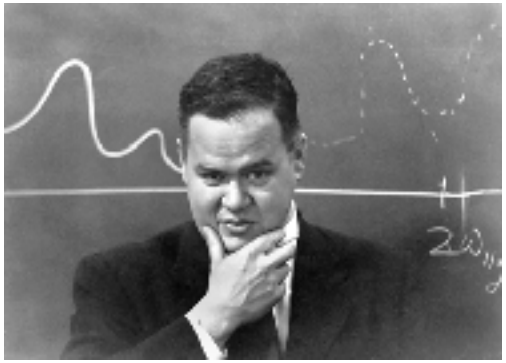
Must know the **Business/Policy Understanding** before Exploring!

Data Exploration: Generic methods for understanding the data distributions and trends

- Helps to guide preprocessing and analysis

- Exploratory Data Analysis (EDA) by Dr. John Tukey:
  - Tukey's take: Visualizing, Clustering and Anomaly detection

- Larson's take:
  - Feature statistics, aggregations
  - Visualizations without research questions
  - Examples:
    - Will we impute? Any obvious outliers?
    - How is target distributed?



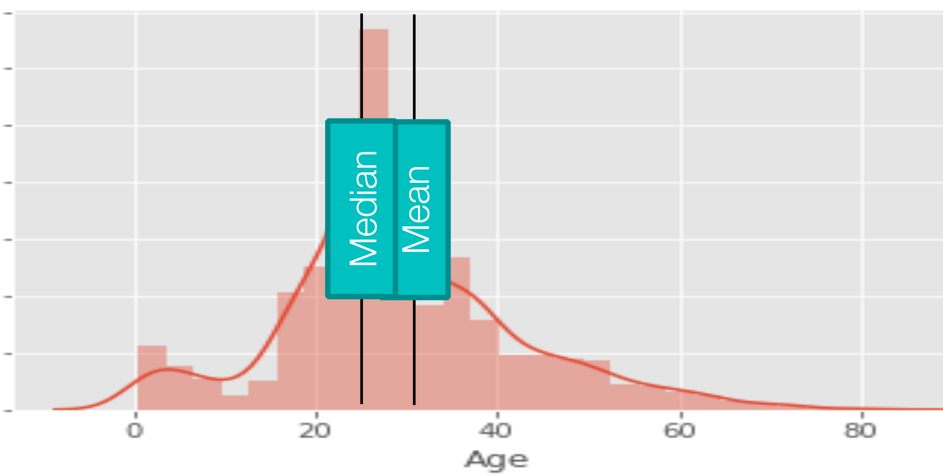
Summary Statistics of Features

- frequency, location, and spread
  - Examples: location by **mean or percentile** (numeric)
    spread by **standard deviation** (numeric)
    frequency by **mode** (categorical)

- Most summary statistics can be calculated in a single pass through the data

$$\text{sample} \quad \mathrm{mean}(x) = \frac{1}{N}\sum_{i=1}^{N} x_i$$

$$\text{sample} \quad \mathrm{median}(x) = x_{50\%}$$



Measures of Spread

- **Dynamic Range** (max - min), *e.g., 0-65 years*
- The **variance** or standard deviation is the most common measure of the spread of a set of points.

$$\text{sample} \quad \text{var}(x) = \sigma_x^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

- σ^2 can be sensitive to outliers, so other measures are also popular:

Average Absolute Difference

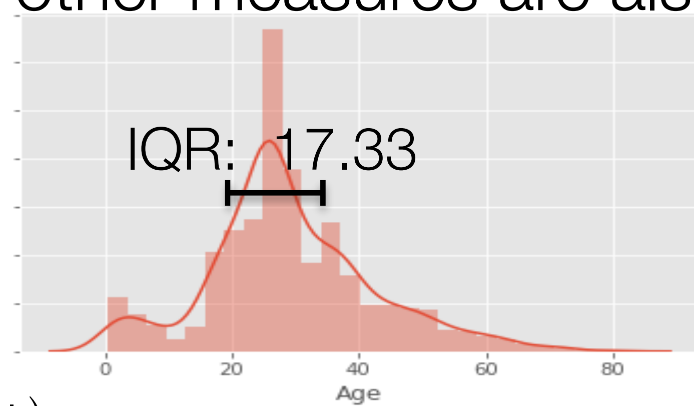
$$\text{AAD}(x) = \frac{1}{N}\sum_{i=1}^{N}\left|x_i - \bar{x}\right|$$

Median Absolute Difference

$$\text{MAD}(x) = \text{median}\left(|x_1 - \bar{x}|, \ldots, |x_i - \bar{x}|, \ldots, |x_N - \bar{x}|\right)$$

Interquartile Range

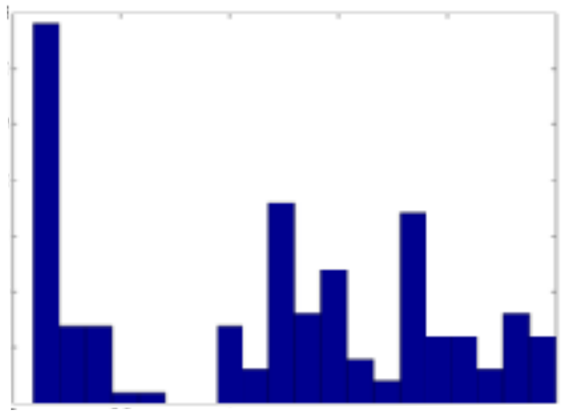
$$\text{IQR}(x) = |x_{75\%} - x_{25\%}|$$



STD: 13.89
AAD: 10.67
MAD: 8.29

Self Test 2a.1

What measure of **spread** is **most appropriate** for the data in the histogram below?

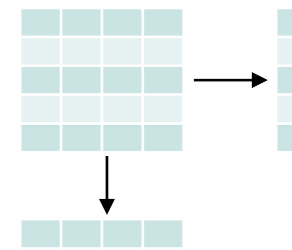


A) Standard Deviation
B) Interquartile Range
C) Median Absolute Difference
D) None of these

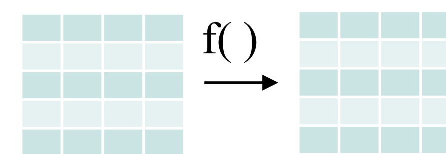
Data Preprocessing

Common Preprocessing Techniques

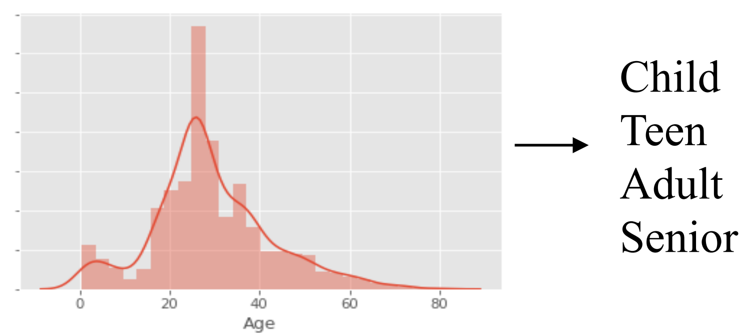
- **Aggregation: Combine features/samples**
  - Reduce the number of attributes or objects
  - Aggregated data tends to be more stable



- **Transformation: Change of scale, range**
  - Normalize dynamic ranges
  - More numerically stable when combining



- **Quantization: Make discrete**
  - More stable
  - More semantically meaningful

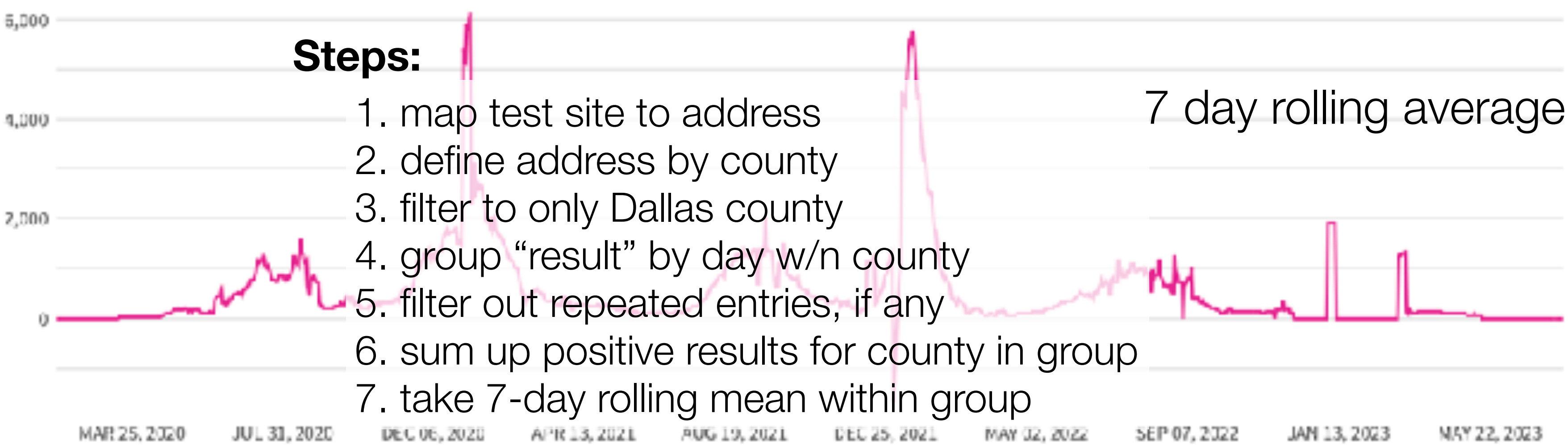


Preprocessing: Aggregation

COVID Cases in Dallas County
696,581

data/image from [cdc.gov](cdc.gov)

Number of daily cases



7 day rolling average

Steps:

1. map test site to address
2. define address by county
3. filter to only Dallas county
4. group "result" by day w/n county
5. filter out repeated entries, if any
6. sum up positive results for county in group
7. take 7-day rolling mean within group

Mar. 2020

May 2023

How has aggregation has been used to create these plots?

TID	*Location*	*time*	*test*	*Probable?*
1	*test site name*	*day and hour*	*test result*	*yes/no*

Preprocessing: Transformation

- Monotonically map one set of values to a set of replacement values
- **Standardization and Normalization**
  - z-scores
    ```
    df_normalized = (df-df.mean())/(df.std())
    ```
  - min/max
    ```
    df_normalized = (df-df.min())/(df.max()-df.min())
    ```

Normalization options in scikit-learn:

`preprocessing.maxabs_scale`(X, *[, axis, copy])	Scale each feature to the [-1, 1] range without breaking the sparsity.
`preprocessing.minmax_scale`(X[, ...])	Transform features by scaling each feature to a given range.
`preprocessing.normalize`(X[, norm, axis, ...])	Scale input vectors individually to unit norm (vector length).
`preprocessing.quantile_transform`(X, *[, ...])	Transform features using quantiles information.
`preprocessing.robust_scale`(X, *[, axis, ...])	Standardize a dataset along any axis
`preprocessing.scale`(X, *[, axis, with_mean, ...])	Standardize a dataset along any axis.
`preprocessing.power_transform`(X[, method, ...])	Power transforms are a family of parametric, monotonic transformations that are applied to make data more Gaussian-like.

Attribute Transformation in Python

```
>>> from sklearn import preprocessing
>>> import numpy as np
>>> X = np.array([[ 1., -1.,  2.],
...               [ 2.,  0.,  0.],
...               [ 0.,  1., -1.]])
>>> X_scaled = preprocessing.scale(X)
>>> X_scaled
array([[ 0.  ..., -1.22...,  1.33...],
       [ 1.22...,  0.  ..., -0.26...],
       [-1.22...,  1.22..., -1.06...]])
```

using direct functions

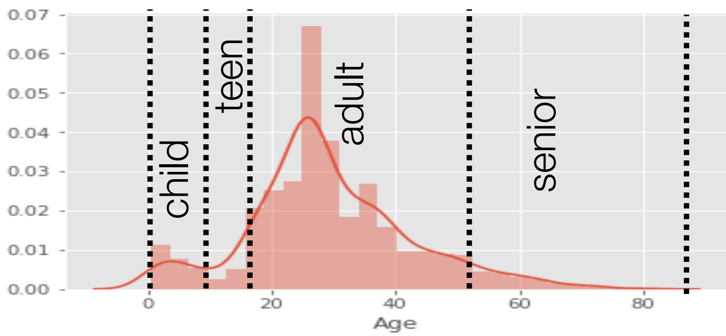
```
>>> scaler = preprocessing.StandardScaler().fit(X)
>>> scaler
StandardScaler(copy=True, with_mean=True, with_std=True)

>>> scaler.mean_
array([ 1. ...,  0. ...,  0.33...])

>>> scaler.std_
array([ 0.81...,  0.81...,  1.24...])

>>> scaler.transform(X)
array([[ 0.  ..., -1.22...,  1.33...],
       [ 1.22...,  0.  ..., -0.26...],
       [-1.22...,  1.22..., -1.06...]])
```

using object oriented approach
Preferred!!

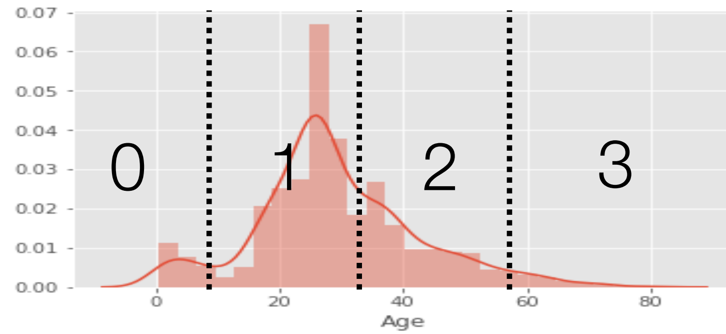
Preprocessing: Quantization

Expert selected

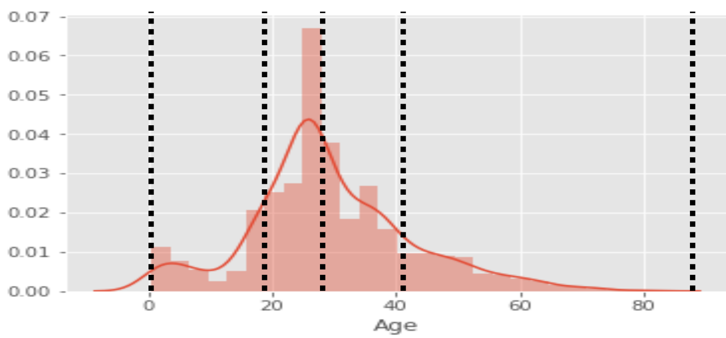
```
pandas.cut(dataframe.var, [5,10,15])
```



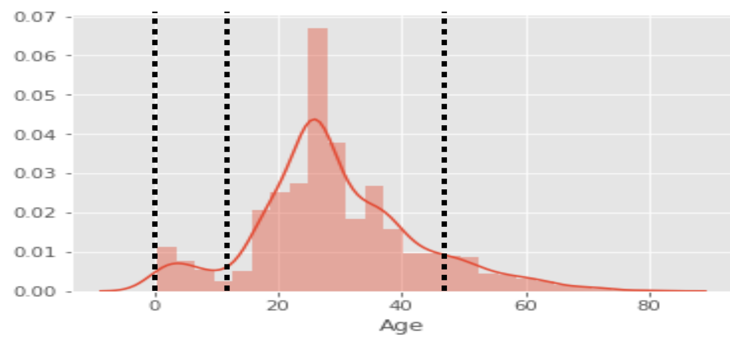
Data



Equal interval width



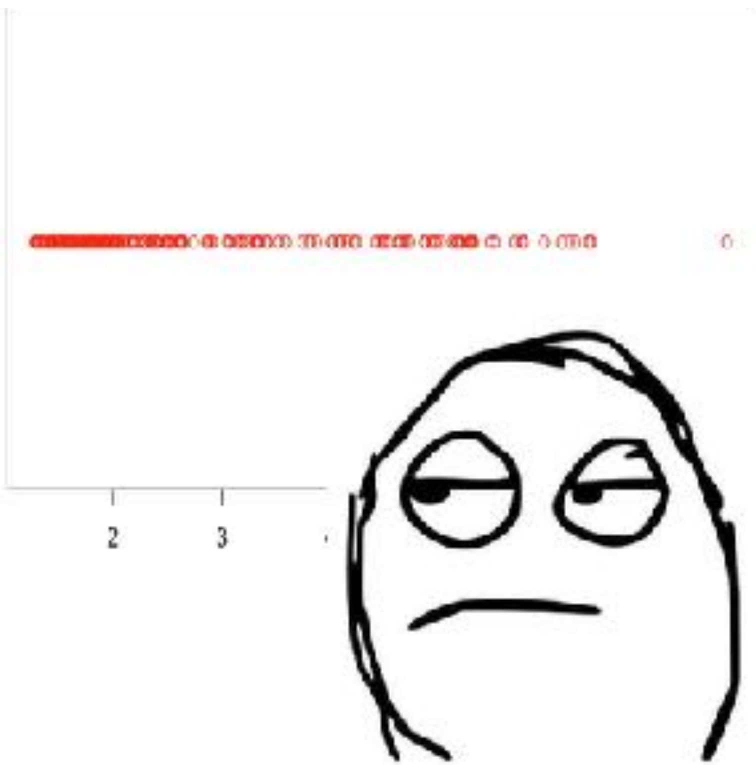
Equal frequency

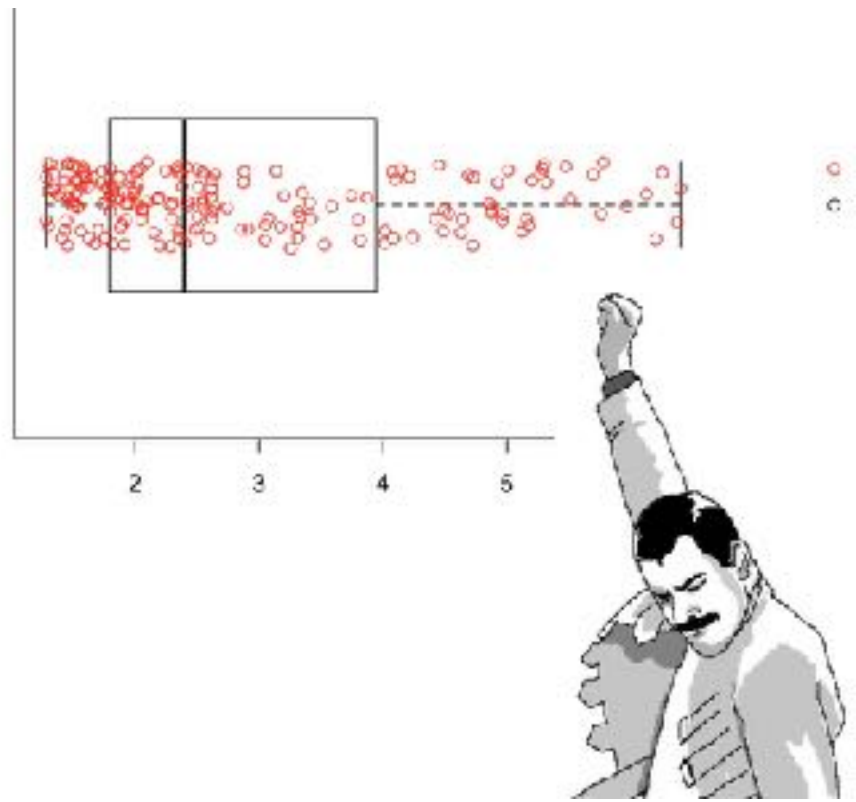


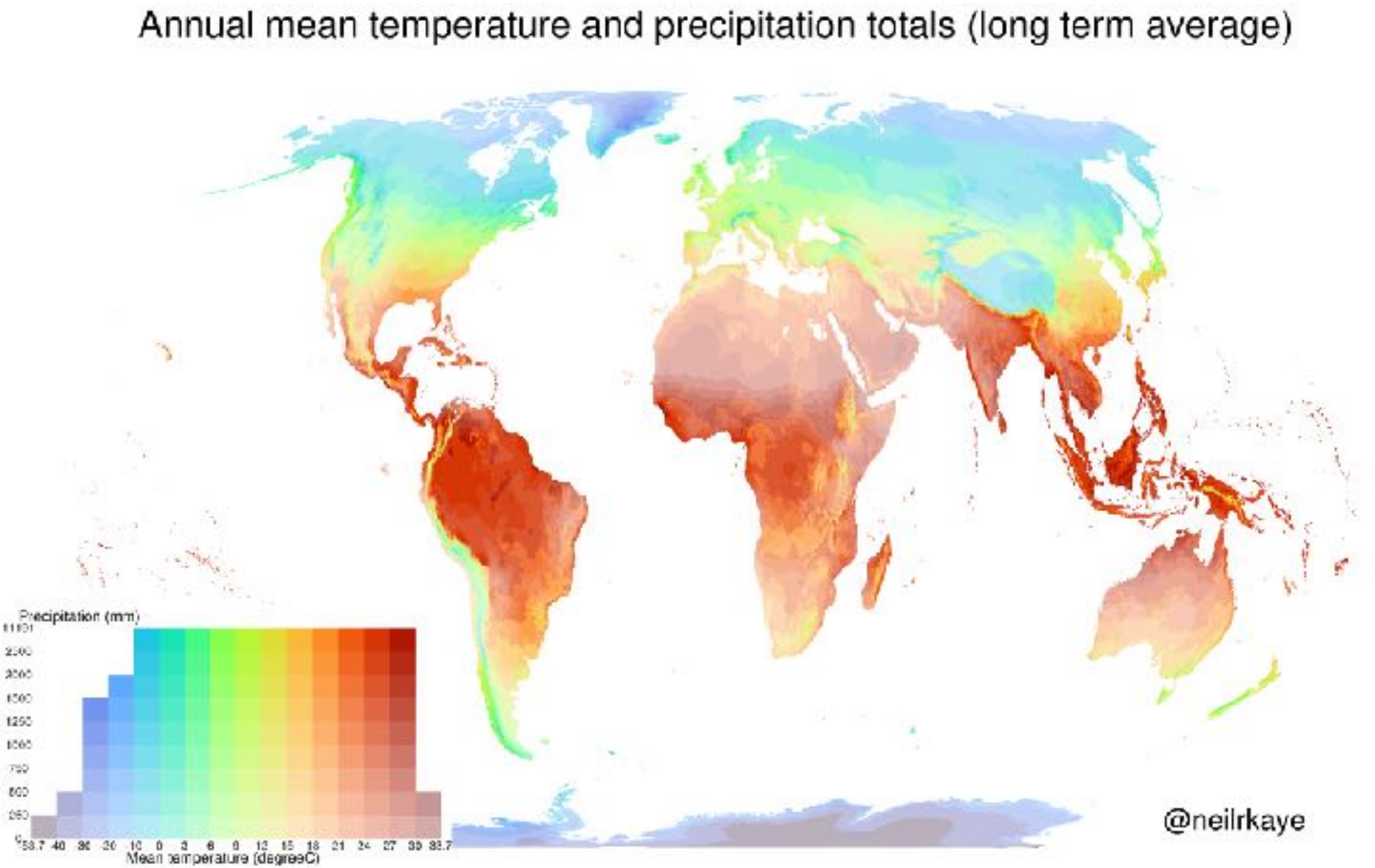
clustering: *e.g.*, K-means

```
num_quantiles = 4
pandas.qcut(dataframe.var, num_quantiles)
```

Data Visualization






Annual mean temperature and precipitation totals (long term average)
Precipitation (mm)
Mean temperature (degreeC)
@neilrkaye

Choosing How/What to Visualize?

- Perform EDA, get the basics out of the way
- Look at business/policy for data and ask an **interesting** question
- Think about the **best plot** to answer the question
    - Do you have the **right data** for visualizing?
    - Do you need to **worry** about the **amount** of data in the plot (aliasing, low samples, etc.)?
    - Can your question be answered **reliably**?
- **Interpret** the visualization: Did it answer the question?
    - **No**: Think of another visual
    - **Kinda**: Ask a follow up question
    - **Yes**: No it didn't, think more critically

Matplotlib

- Python plotting utility
  - Has **low level plotting** functionality
  - Highly **similar to Matlab and R** for plotting
- Extended to be visually more beautiful by
  - **seaborn**: stanford data visualization group

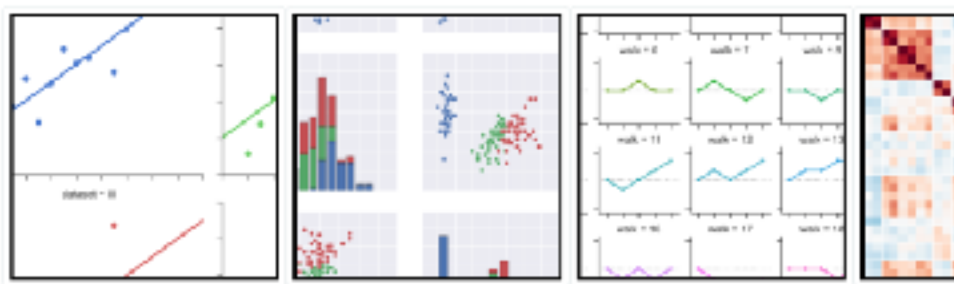
John Hunter (1968-2012)

On August 28 2012, John D. Hunter, the creator of matplotlib, died from complications arising from cancer treatment, after a brief but intense battle with this terrible illness. John is survived by his wife Miriam, his three daughters Rahel, Ava and Clara, his sisters Layne and Mary, and his mother Sarah.

If you have benefited from John's many contributions, please say thanks in the way that would matter most to him. Please consider making a donation to the John Hunter Memorial Fund.

matplotlib

Seaborn: statistical data visualization



Let’s look at some graphs

Demo

- You tell me what conclusions we are getting from these graphs
  - Histogram
  - KDE
  - HeatMaps and Correlation
  - Scatter and Scatter Matrix
  - Box / Violin / Swarm

`03.Data Visualization.ipynb`

Matplotlib
Seaborn
Plotly

For Next Lecture

- Next Time:
    - Finish Visualization Demo
    - First Town Hall Meeting