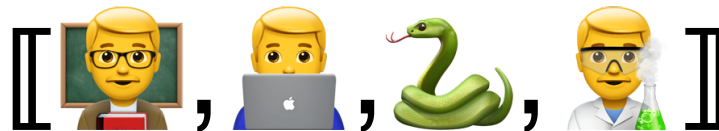


# Lecture Notes for **Machine Learning in Python**



## Professor Eric Larson **Preprocessing and Visualization**

# Class Logistics and Agenda

- Participation/Teams
- Be sure you look at **Lab One!**
- Dataset Selection Now Complete! Probably! ... maybe?
- 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
- Document Features

## Feature Type Representation Review

	Attribute	Representation Transformation	Comments
Discrete	Nominal	Any permutation of values <b>one hot encoding</b>	If all employee ID numbers were reassigned, would it make any difference?
	Ordinal	An order-preserving change of values, i.e., $\text{new\_value} = f(\text{old\_value})$ where $f$ is a monotonic function. <b>integer</b>	An attribute encompassing the notion of good, better, best can be represented equally well by the values {1, 2, 3} or by {0.5, 1, 10}.
Continuous	Interval	$\text{new\_value} = a * \text{old\_value} + b$ where $a$ and $b$ are constants. <b>float</b>	Thus, the Fahrenheit and Celsius temperature scales differ in terms of where their zero value is and the size of a unit (degree).
	Ratio	$\text{new\_value} = a * \text{old\_value}$ <b>float</b>	Length can be measured in meters or feet.

## K-Nearest Neighbors Imputation

ID	Pregnant	SMV	Age	Diabetes
1	Y	23.0	21-30	positive
2	N	90.0	31-40	negative
3	Y	33.3	?	positive
4	?	38.1	21-30	negative
5	N	23.1	31-40	positive
6	Y	90.0	21-30	negative
7	Y	0.0	21-30	positive
8	Y	35.3	?	negative
9	N	30.5	31-40	positive
10	Y	87.5	31-40	positive

For K=3, find 3 closest neighbors

ID	Pregnant	SMV	Age	Diabetes	Dist
3	Y	23.3	?	positive	C
6	Y	26.6	21-30	negative	(1-2+3)/3
9	N	90.0	31-40	negative	(1+3+4)/3
4	?	38.1	21-30	negative	(4+5+7)/3

How to calculate distance?

- Difference for valid features only
- May need to normalize ranges
- Or weight neighbors differently
- Or have min # of valid features
- Euclidean, city-block, etc.

## Demo

Start  
Pandas demo  
DataFrames  
Loading  
Indexing  
Imputing



## “if needed” Pandas demo

DataFrames

Loading

Indexing

Imputing



03.Data Visualization.ipynb

# Data Exploration



**Lottery.com**

11 33 44 59 67 8  
POWER PLAY: 3X

SEE ALL RESULTS

PLAY THE REAL U.S. **POWERBALL**

**\$NaN MILLION**

19:42:51 UNTIL PLAY CLOSES

TEST YOUR LUCK!

**GET TICKETS**

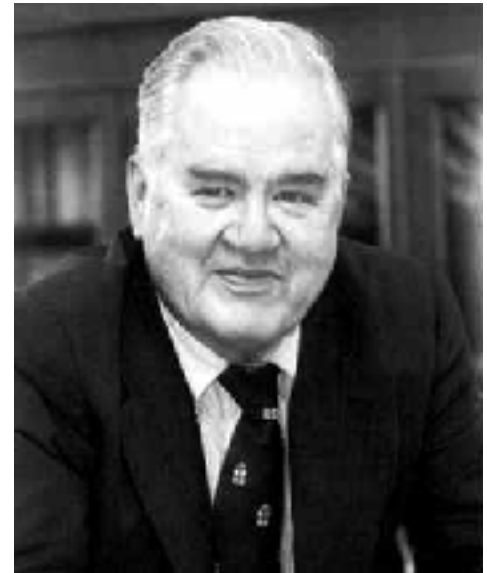
YOU WON THE JACKPOT.  
WINNING NUMBERS: 1 23 32 45 54

The image is a digital advertisement for the U.S. Powerball lottery. At the top, the 'Lottery.com' logo is on the left, and the winning numbers '11 33 44 59 67 8' are displayed in white circles on a dark background, with the '8' being a red Powerball. Below the numbers, it says 'POWER PLAY: 3X'. In the center, there's a button that says 'SEE ALL RESULTS' next to a small red icon. Below that, the text reads 'PLAY THE REAL U.S. POWERBALL' followed by '\$NaN MILLION' in large red letters, indicating a broken link or placeholder for the jackpot amount. Underneath, it says '19:42:51 UNTIL PLAY CLOSES'. At the bottom left, there's an image of a smartphone displaying a 'YOU WON THE JACKPOT' message and 'WINNING NUMBERS: 1 23 32 45 54'. At the bottom right, there's a green button that says 'GET TICKETS' and the text 'TEST YOUR LUCK!' above it.

# What is data exploration?

A preliminary exploration of the data to better understand its characteristics.

- Help **select** the **right tool** for preprocessing or analysis
- Exploratory Data Analysis (EDA) by Dr. John Tukey:
  - The focus was visualization
  - Clustering and anomaly detection were viewed as exploratory techniques
- In our discussion,
  - Summary statistics, aggregations
  - Visualizing summaries



# Summary Statistics

- frequency, location, and spread
  - Examples: location by **mean**  
spread by **standard deviation**
- Most summary statistics can be calculated in a single pass through the data

$$\text{sample mean}(x) = \bar{x} = \frac{1}{m} \sum_{i=1}^m x_i$$

$$\text{sample median}(x) = \begin{cases} x_{(r+1)} & \text{if } m \text{ is odd, i.e., } m = 2r + 1 \\ \frac{1}{2}(x_{(r)} + x_{(r+1)}) & \text{if } m \text{ is even, i.e., } m = 2r \end{cases}$$

- For nominal data, mode or frequency is most common



# Measures of Spread

- **Range** is the difference between the max and min
- The **variance** or standard deviation is the most common measure of the spread of a set of points.

$$\text{sample variance}(x) = s_x^2 = \frac{1}{m-1} \sum_{i=1}^m (x_i - \bar{x})^2$$

- However, this is also sensitive to outliers, so that other measures are often used.

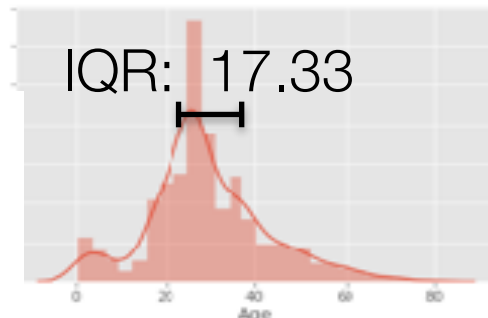
Average Absolute Difference

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

Median Absolute Difference

$$\text{MAD}(x) = \text{median}\left(\{|x_1 - \bar{x}|, \dots, |x_m - \bar{x}|\}\right)$$

$$\text{interquartile range}(x) = x_{75\%} - x_{25\%}$$



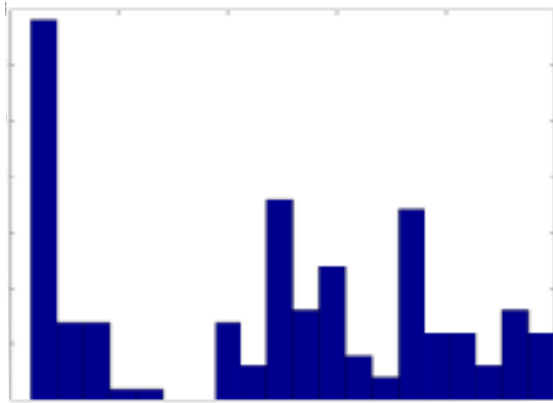
STD: 13.89

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



# 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**
    - ◆ Normalize dynamic ranges
    - ◆ More numerically stable when combining
  - **Quantization: Make discrete**
    - ◆ More stable
    - ◆ More semantically meaningful

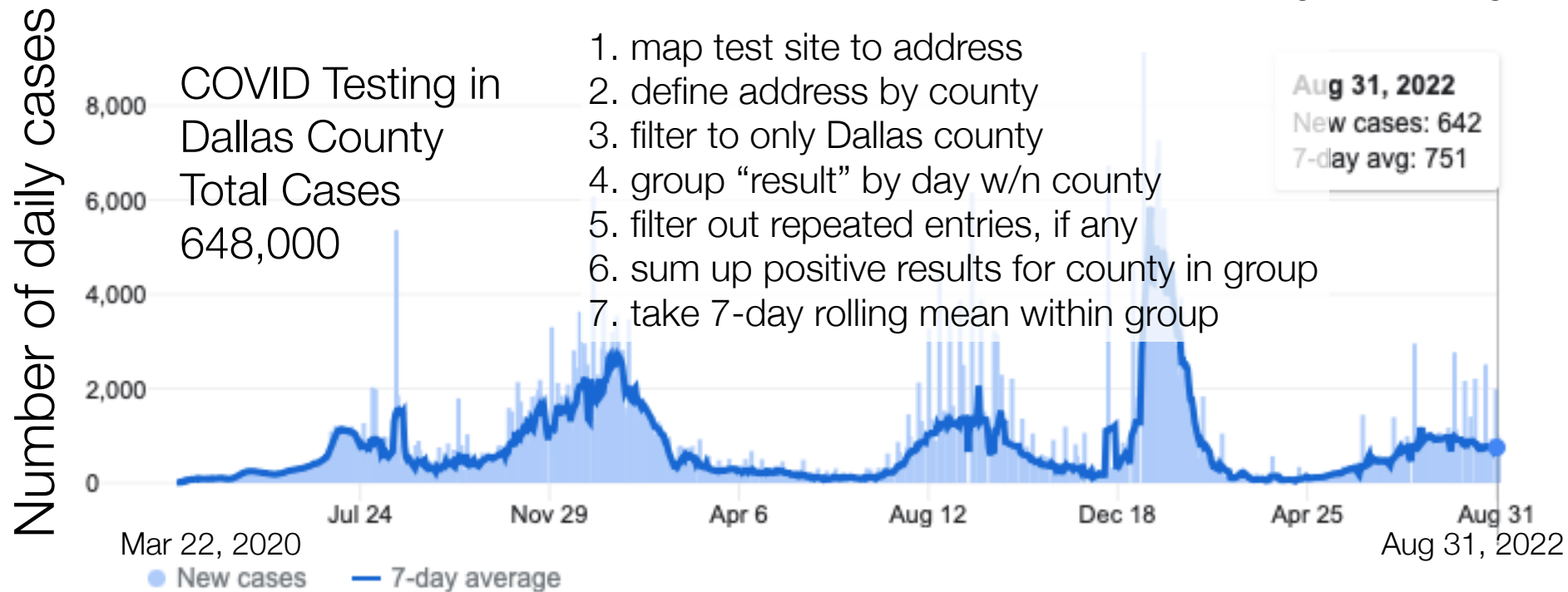
# Preprocessing: Aggregation

data from [cdc.gov](https://www.cdc.gov)  
image from Google

## 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

COVID Testing in  
Dallas County  
Total Cases  
648,000



How has aggregation has been used to create these plots?

<i><b>TID</b></i>	<i><b>Location</b></i>	<i><b>time</b></i>	<i><b>test</b></i>	<i><b>Probable?</b></i>
<b>1</b>	<i>test site name</i>	<i>day and hour</i>	<i>test result</i>	<i>yes/no</i>

# 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:

<code>preprocessing.maxabs_scale(X, *[, axis, copy])</code>	Scale each feature to the $[-1, 1]$ range without breaking the sparsity.
<code>preprocessing.minmax_scale(X[, ...])</code>	Transform features by scaling each feature to a given range.
<code>preprocessing.normalize(X[, norm, axis, ...])</code>	Scale input vectors individually to unit norm (vector length).
<code>preprocessing.quantile_transform(X, *[, ...])</code>	Transform features using quantiles information.
<code>preprocessing.robust_scale(X, *[, axis, ...])</code>	Standardize a dataset along any axis
<code>preprocessing.scale(X, *[, axis, with_mean, ...])</code>	Standardize a dataset along any axis.
<code>preprocessing.power_transform(X[, method, ...])</code>	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...])
```

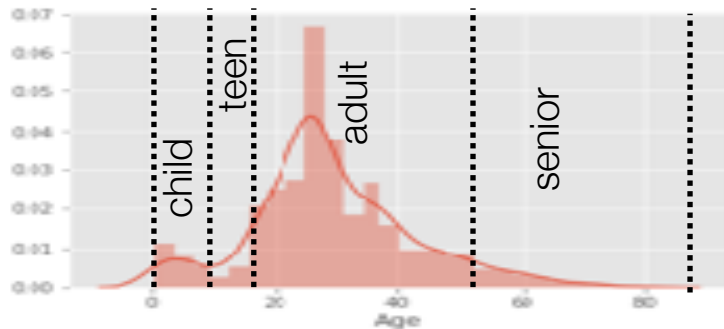
```
>>> scaler.transform(X)
array([[ 0. ...., -1.22...,  1.33...],
       [ 1.22...,  0. ...., -0.26...],
       [-1.22...,  1.22..., -1.06...]])
```

using object oriented approach  
**Preferred!!**

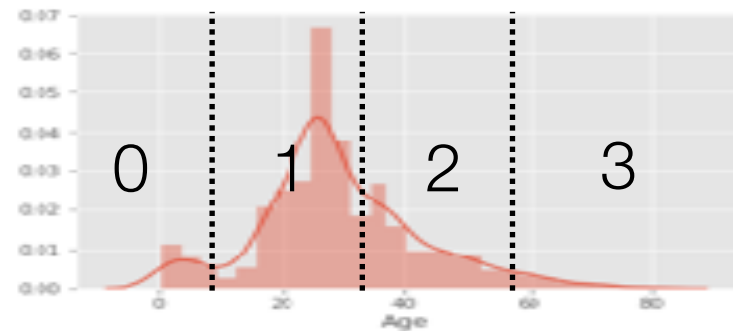
# Preprocessing: Quantization

**Expert selected**

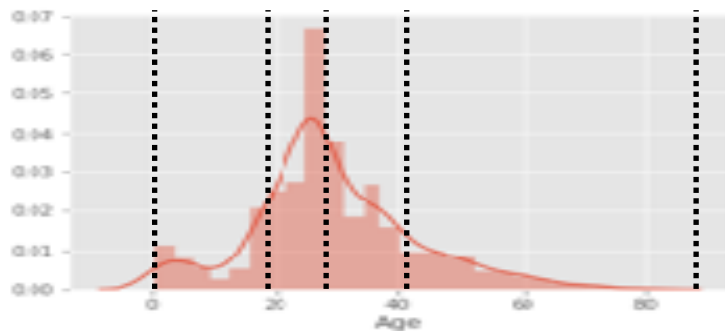
`pandas.cut(dataframe.var, [5,10,15])`



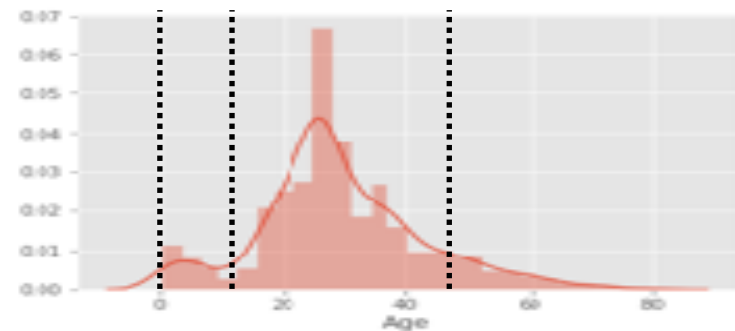
Data



Equal interval width



Equal frequency



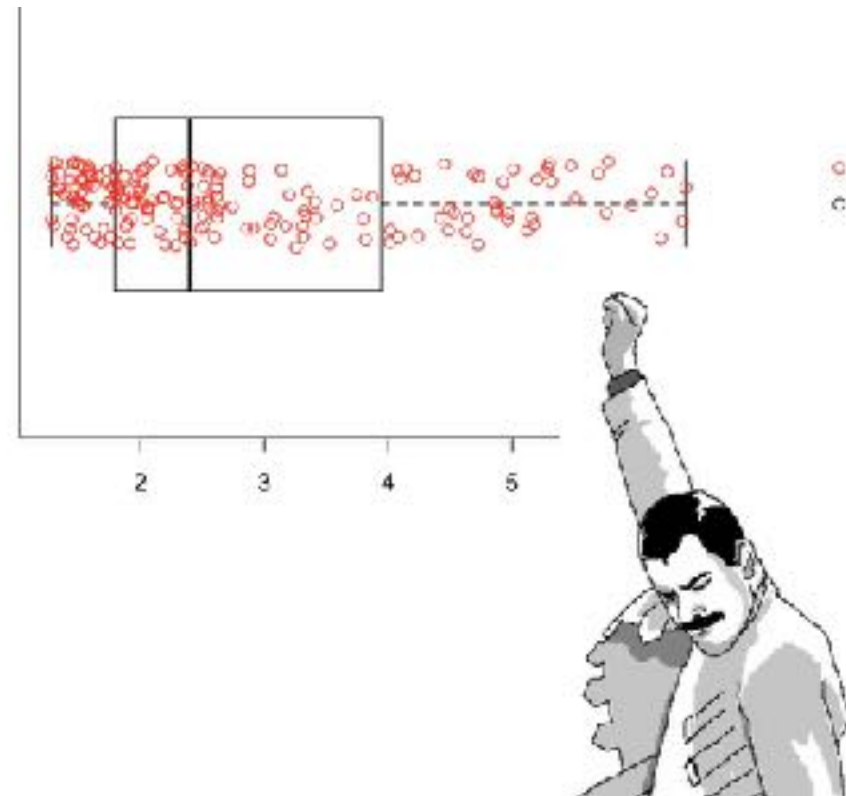
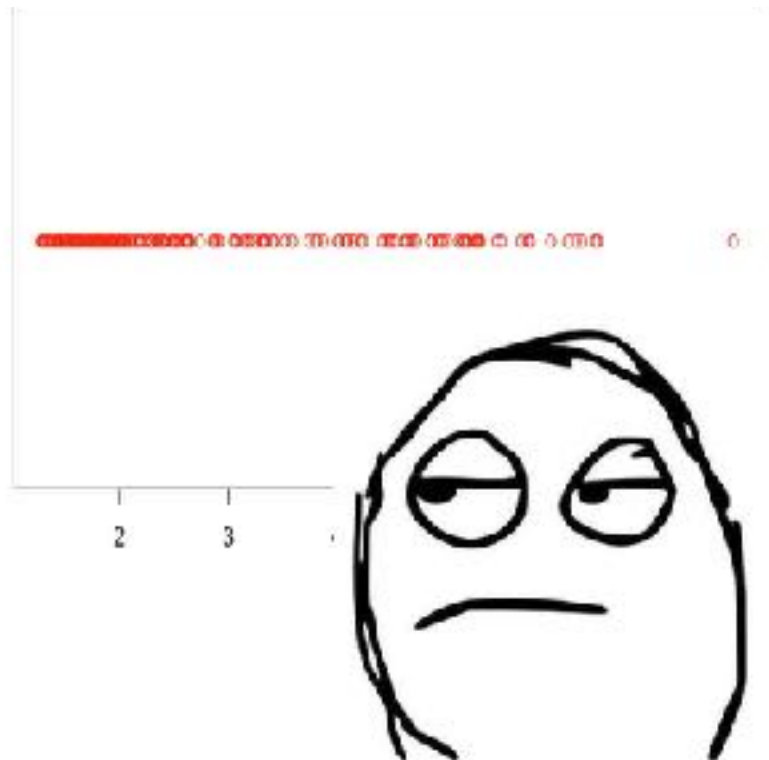
clustering: e.g., K-means

`num_quantiles = 4`

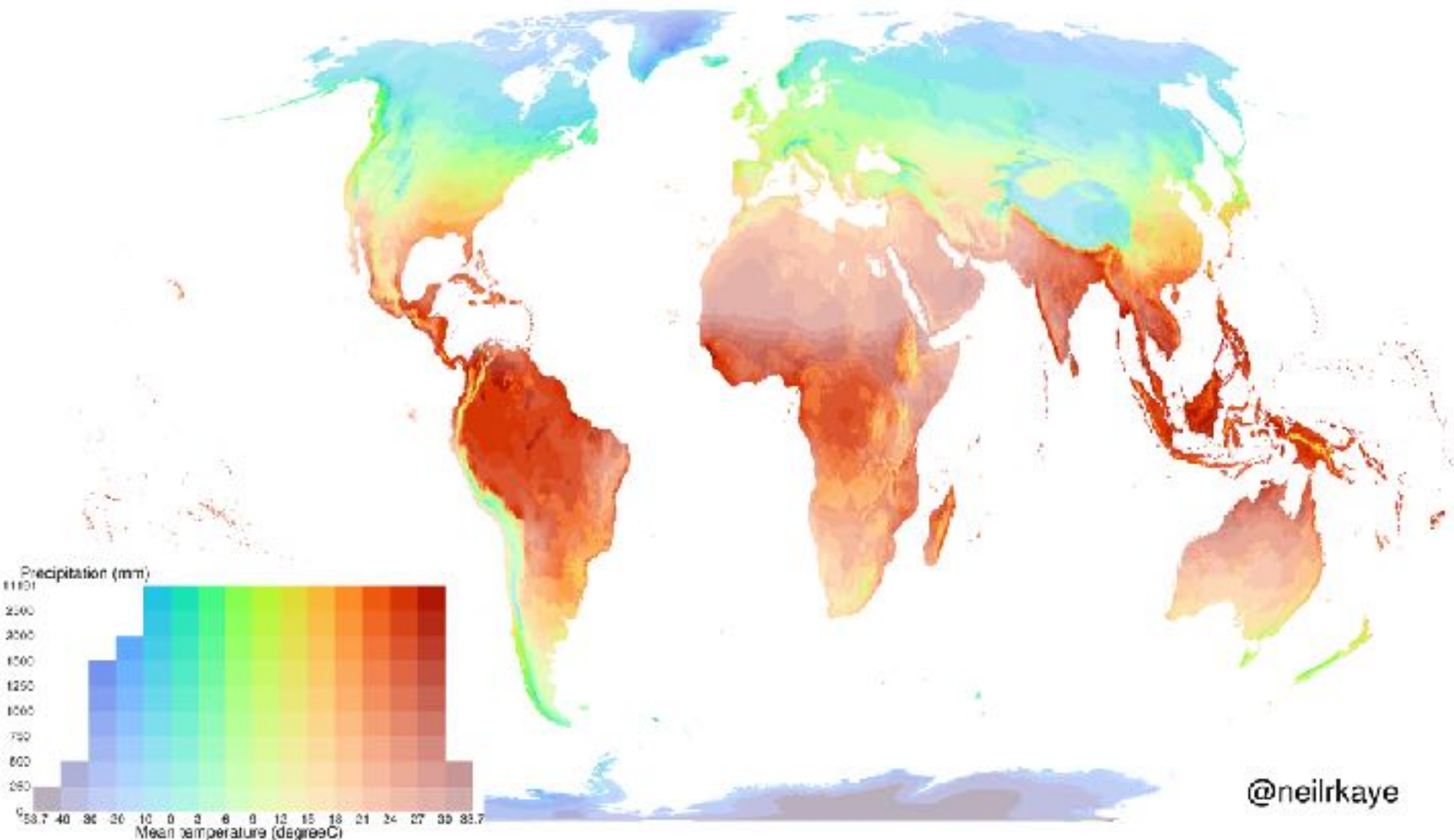
`pandas.qcut(dataframe.var, num_quantiles)`



# Data Visualization



## Annual mean temperature and precipitation totals (long term average)



# Choosing How/What to Visualize?

- Start with a question you want to understand
- 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)

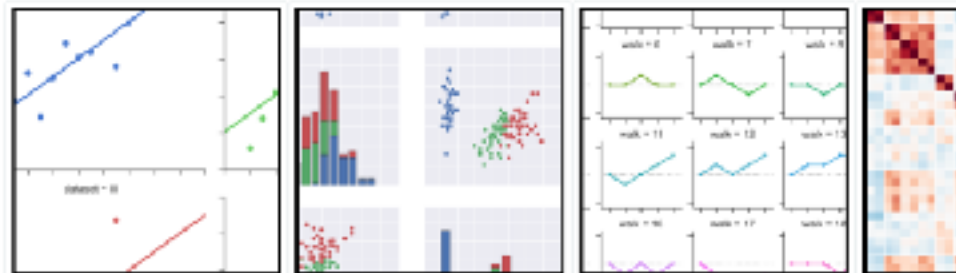


On August 28 2012, John D. Hunter, the creator of matplotlib, died from complications arising from cancer treatment, after a brief but intense battle with this terrible illness. John is survived by his wife Miriam, his three daughters Rahel, Ava and Clara, his sisters Layne and Mary, and his mother Sarah.

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](#).



## Seaborn: statistical data visualization



- 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

03.Data Visualization.ipynb



## Other Tutorials:

<https://t.co/zNzD8Q8w5E>

<http://matplotlib.org/examples/index.html>

<http://stanford.edu/~mwaskom/software/seaborn/index.html>

<http://pandas.pydata.org/pandas-docs/stable/visualization.html>

[http://nbviewer.ipython.org/github/mwaskom/seaborn/blob/master/examples/plotting\\_distributions.ipynb](http://nbviewer.ipython.org/github/mwaskom/seaborn/blob/master/examples/plotting_distributions.ipynb)

# For Next Lecture

- Next Time:
  - Finish Visualization Demo
  - First Town Hall Meeting
- Look at chapter 5 of Python Machine Learning