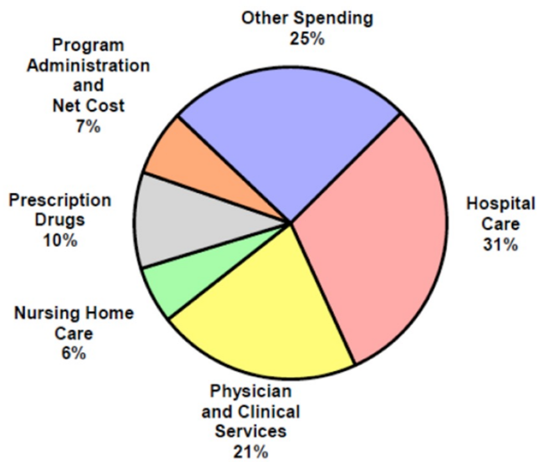
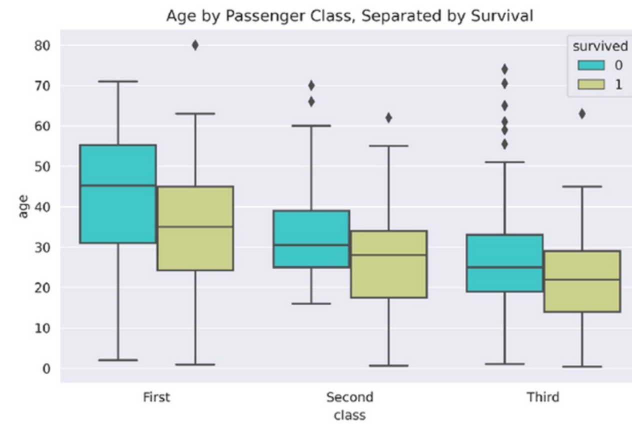
CSC380: Principles of Data Science

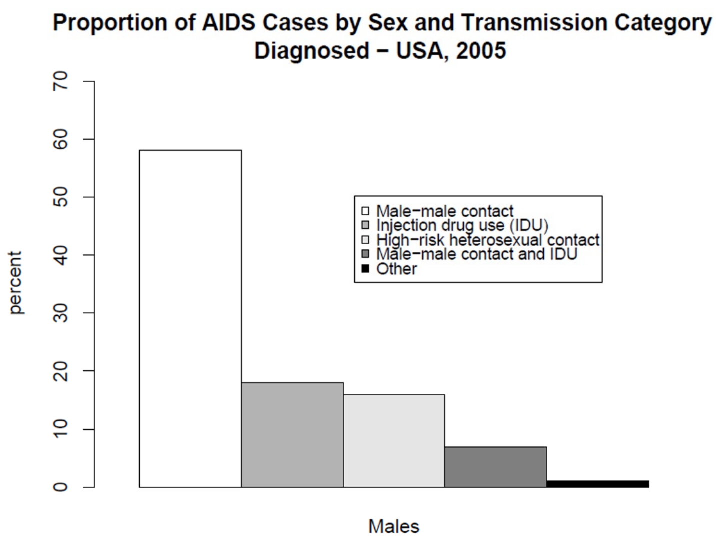
Data Analysis, Collection, and Visualization 2

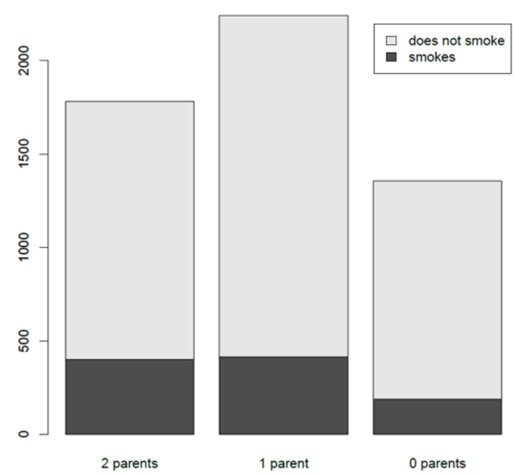
Visualizing Categorical Variables



	student smokes	student does not smoke	total
2 parents smoke	400	1380	1780
1 parent smokes	416	1823	2239
0 parents smoke	188	1168	1356
total	1004	4371	5375







Two-Way Table

Also called contingency table or cross tabulation table…

Frequency

	student smokes	student does not smoke	total
2 parents smoke	400	1380	1780
1 parent smokes	416	1823	2239
0 parents smoke	188	1168	1356
total	1004	4371	5375

Two-Way Table

Also called <u>contingency table</u> or <u>cross tabulation table</u>…

Relative Frequency

	student smokes	student does not smoke	total
2 parents smoke	7.4%	25.7%	33.1%
1 parent smokes	7.7%	33.9%	41.7%
0 parents smoke	3.5%	21.8%	25.2%
total	18.7%	81.3%	100%

Column Variable

Marginal Distribution Of Row Variable

Row Variable

Marginal Distribution Of Column Variable

Joint Distribution

Q: how do you compute the conditional probability P(student smokes | 2 parents smoke)?

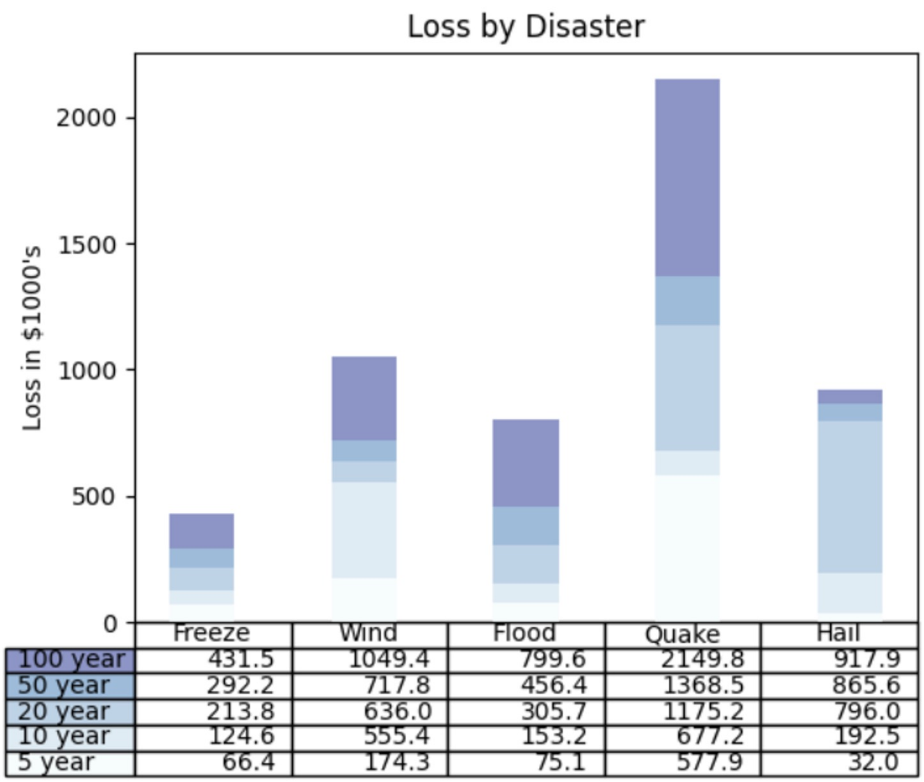
Two-Way Table


```
data = [[ 66386, 174296,  75131, 577908,  32015],
        [ 58230, 381139,  78045,  99308, 160454],
        [ 89135,  80552, 152558, 497981, 603535],
        [ 78415,  81858, 150656, 193263,  69638],
        [139361, 331509, 343164, 781380,  52269]]

columns = ('Freeze', 'Wind', 'Flood', 'Quake', 'Hail')
rows = ['%d year' % x for x in (100, 50, 20, 10, 5)]
```

```
colors = plt.cm.BuPu(np.linspace(0, 0.5, len(rows)))
```

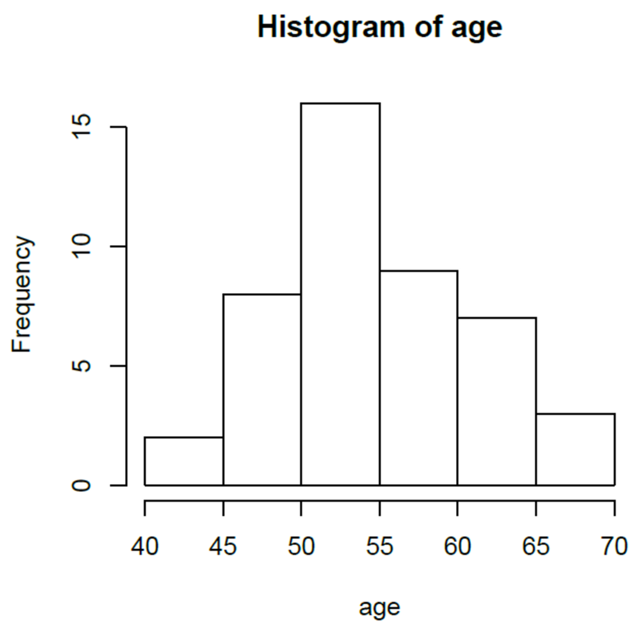
```
the_table = plt.table(cellText=cell_text,
                      rowLabels=rows,
                      rowColours=colors,
                      colLabels=columns,
                      loc='bottom')
```

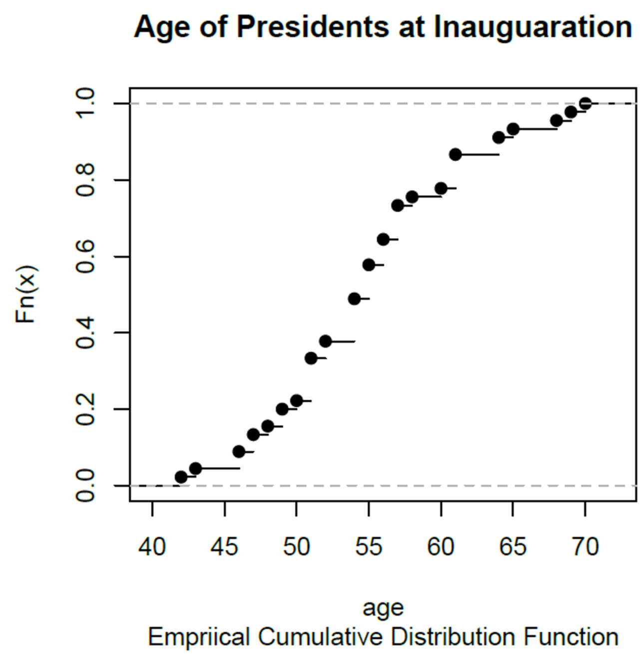
Adding stacked bars requires more steps, full code here: https://matplotlib.org/stable/gallery/misc/table_demo.html



Histogram

Empirical approximation of (quantitative) data generating distribution



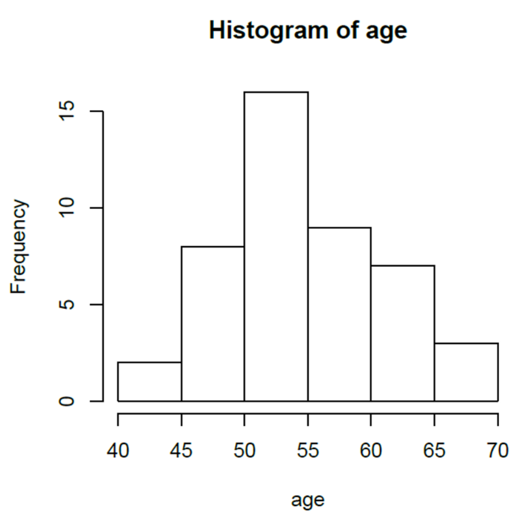


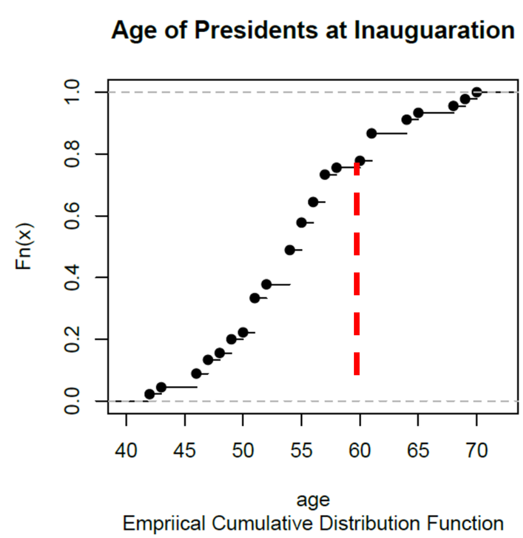
Empirical CDF for each x gives P(X<x),

$$F_n(x) = \frac{1}{n}\#(\text{observations less than or equal to x})$$

Quantile / Percentile

Question Is 60yrs old for a US president? Why or why not?





Empirical CDF for each x gives P(X<x),

$$F_n(x) = \frac{1}{n}\#(\text{observations less than or equal to x})$$

Compute probability of being <60,

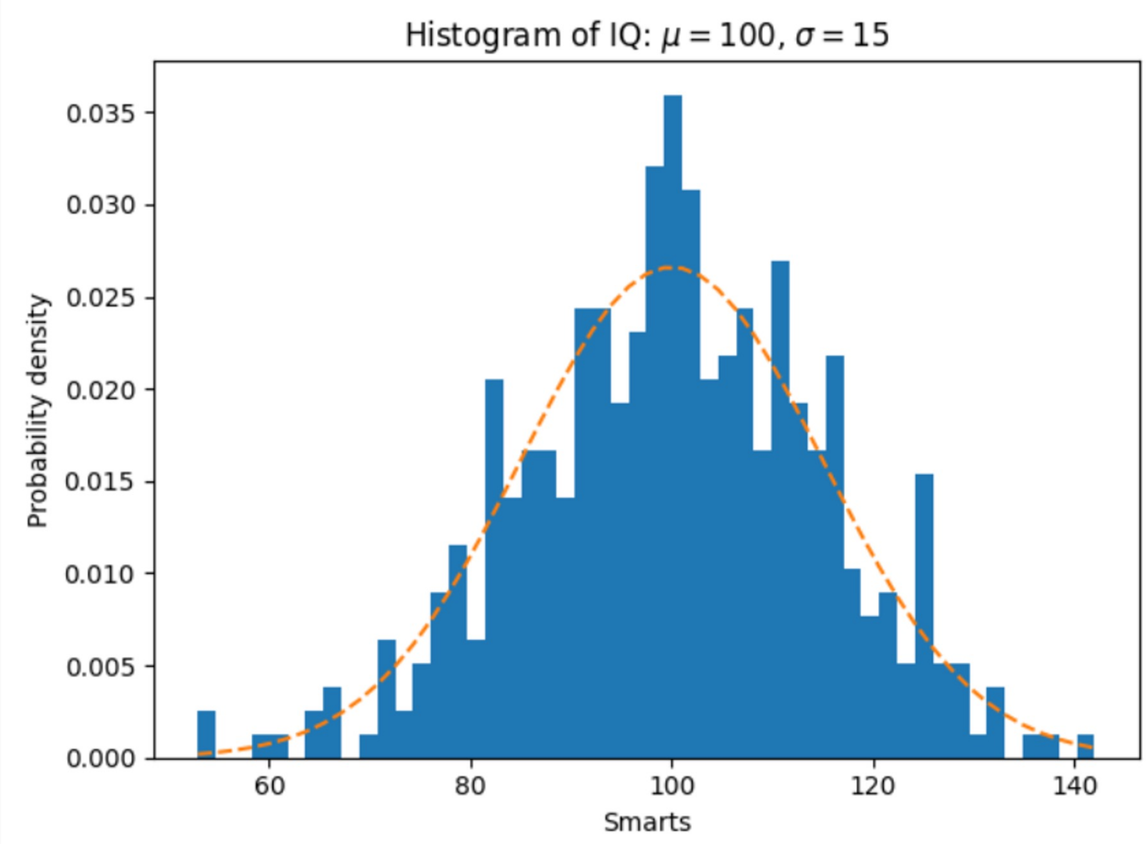
$$F_n(60) \approx 0.8$$

0.8 Quantile or 80th Percentile → About 80% of presidents younger than 60

Histogram

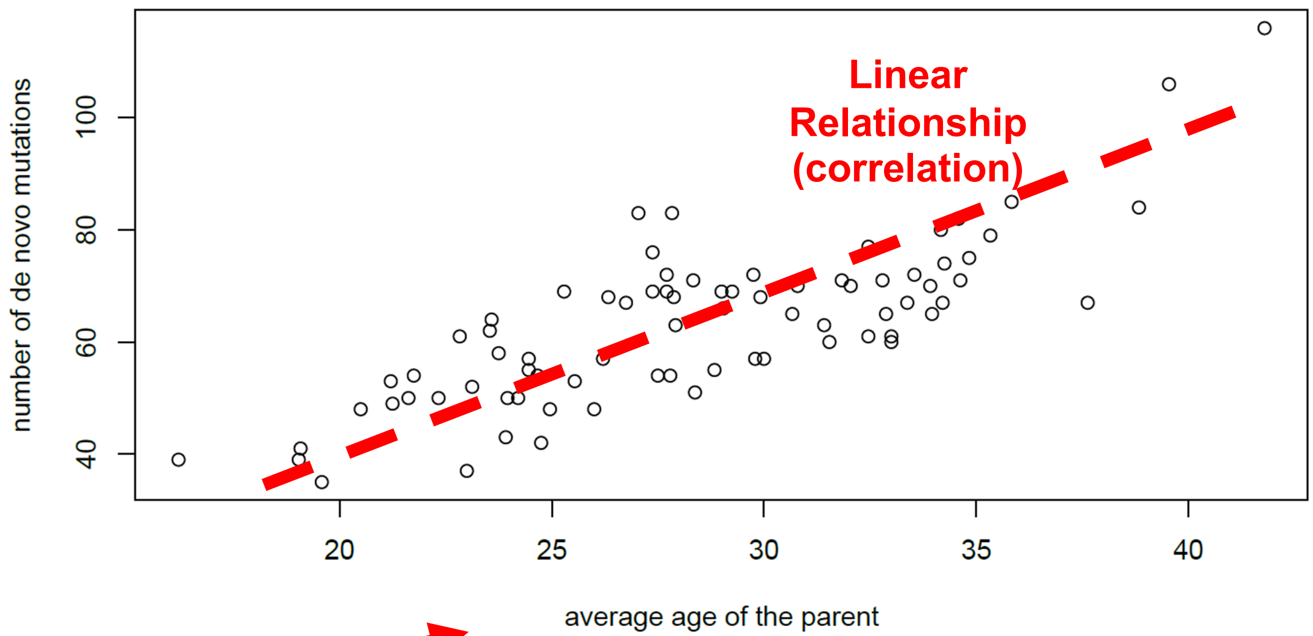

```
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(19680801)

# example data
mu = 100  # mean of distribution
sigma = 15  # standard deviation of distribution
x = mu + sigma * np.random.randn(437)

num_bins = 50

fig, ax = plt.subplots()

# the histogram of the data
n, bins, patches = ax.hist(x, num_bins, density=True)

# add a 'best fit' line
y = ((1 / (np.sqrt(2 * np.pi) * sigma)) *
     np.exp(-0.5 * (1 / sigma * (bins - mu))**2))
ax.plot(bins, y, '--')
ax.set_xlabel('Smarts')
ax.set_ylabel('Probability density')
ax.set_title(r'Histogram of IQ: $\mu=100$, $\sigma=15$')

# Tweak spacing to prevent clipping of ylabel
fig.tight_layout()
plt.show()
```

Standard normal dist



Scatterplot

Compares relationship between two quantitative variables…

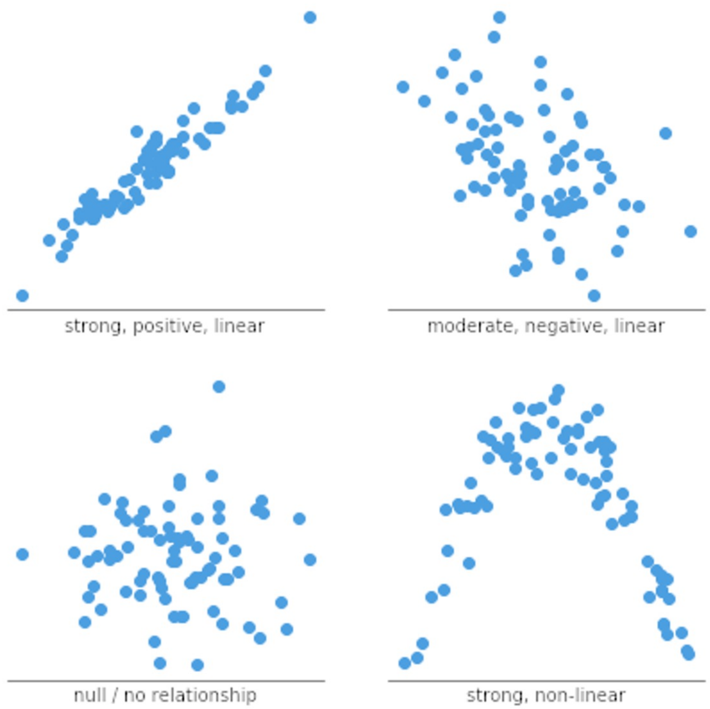


Target (or response) Variable

Explanatory Variable
Also: predictor, descriptor, input

Scatterplot

Compares relationship between two quantitative variables…

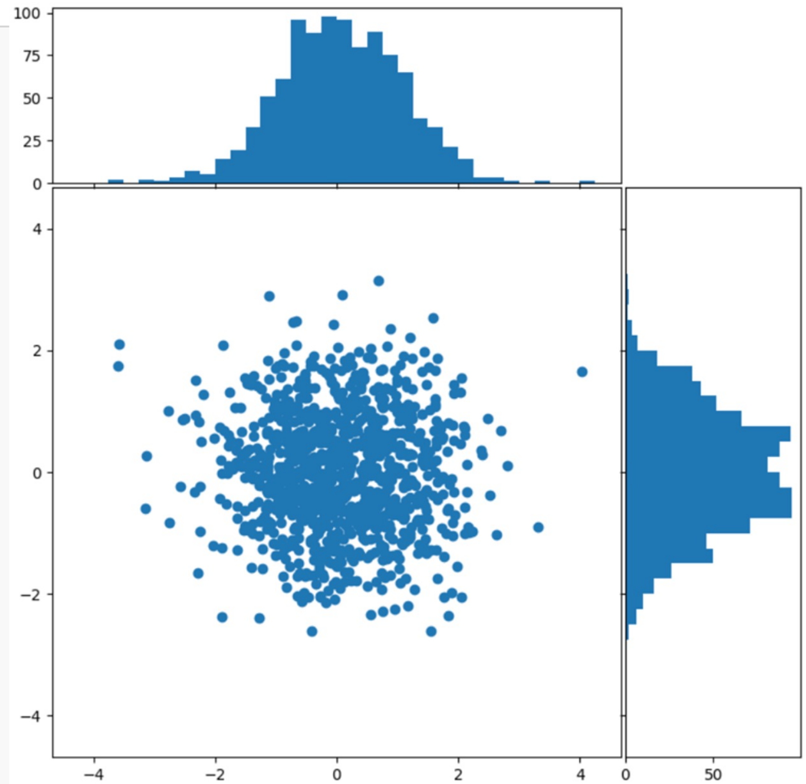


Relationship can also be:

- Nonlinear (e.g. “curvy”)
- Clustered or grouped

Scatterplot + Histogram

```python
import numpy as np
import matplotlib.pyplot as plt

# Fixing random state for reproducibility
np.random.seed(19680801)

# some random data
x = np.random.randn(1000)
y = np.random.randn(1000)


def scatter_hist(x, y, ax, ax_histx, ax_histy):
    # no labels
    ax_histx.tick_params(axis="x", labelbottom=False)
    ax_histy.tick_params(axis="y", labelleft=False)

    # the scatter plot:
    ax.scatter(x, y)

    # now determine nice limits by hand:
    binwidth = 0.25
    xymax = max(np.max(np.abs(x)), np.max(np.abs(y)))
    lim = (int(xymax/binwidth) + 1) * binwidth

    bins = np.arange(-lim, lim + binwidth, binwidth)
    ax_histx.hist(x, bins=bins)
    ax_histy.hist(y, bins=bins, orientation='horizontal')
```



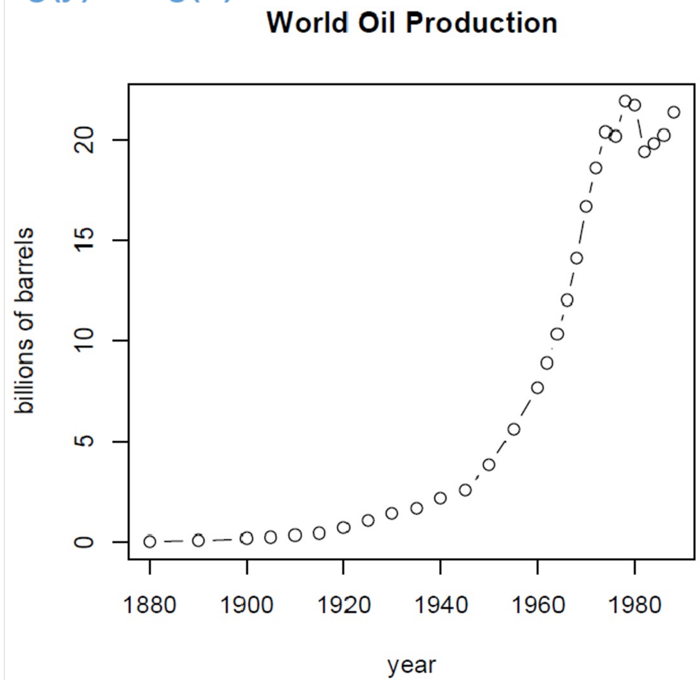
Full Code:
https://matplotlib.org/stable/gallery/lines_bars_and_markers/scatter_hist.html

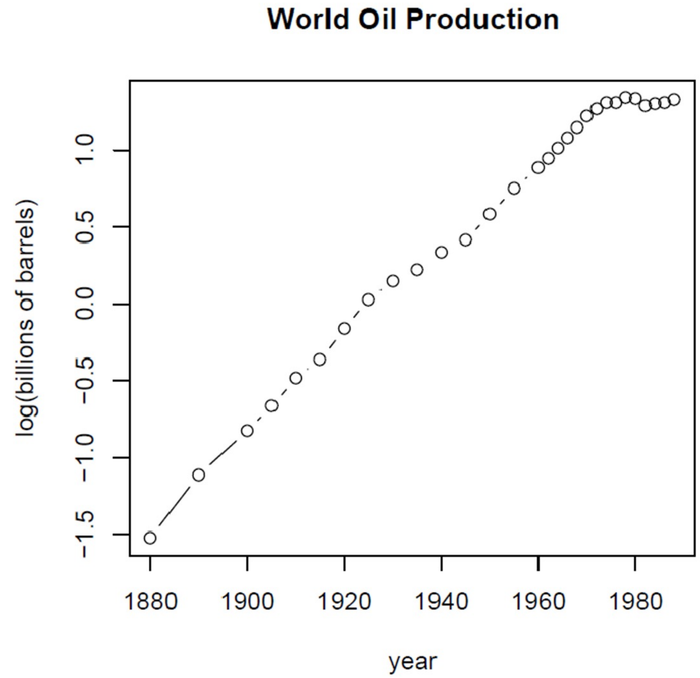
Logarithm Scale

Changing limits and base of y-scale highlights different aspects…

if y = e^x, then log(y) = x
if y = b^x, then log(y) = log(b)*x

=> becomes linear in x

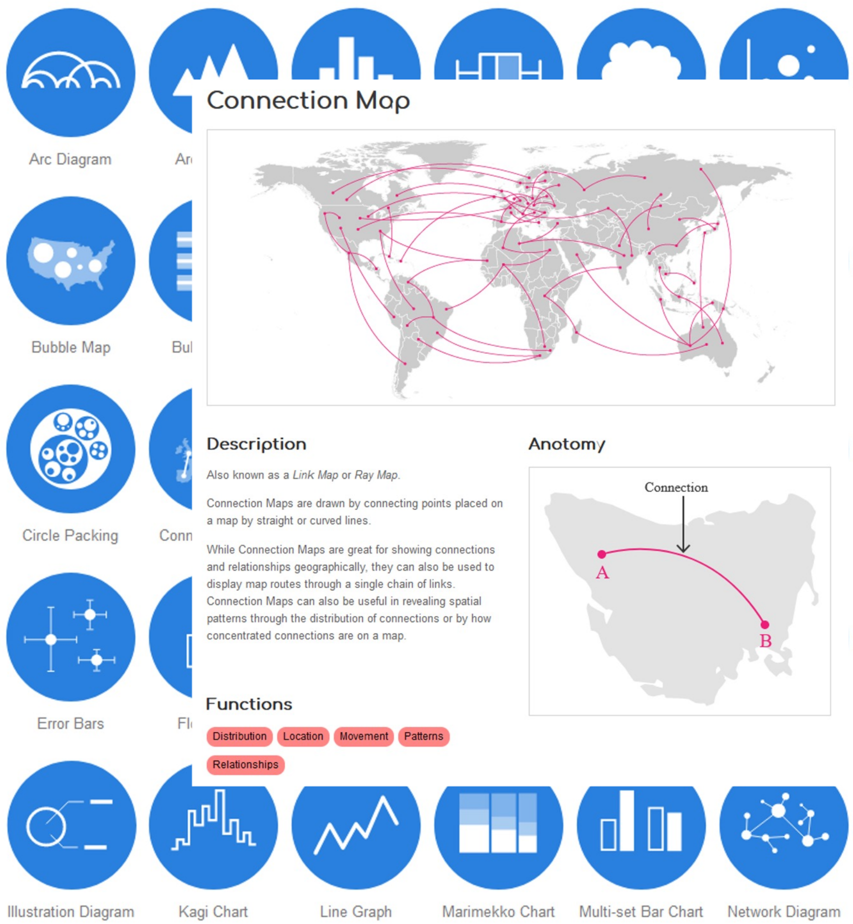




…log-scale emphasizes relative changes in smaller quantities

More Visualization Resources

datavizcatalogue.com



matplotlib.org

scikit-learn.org

Outline

- Data Visualization
- Data Summarization
- Data Collection and Sampling

Data Summarization

- Raw data are hard to interpret
- Visualizations summarize important aspects of the data
- The *empirical distribution* estimates the distribution on data, but can be hard to interpret
- **Summary statistics** characterize aspects of the data distribution like:
  - Location / center
  - Scale / spread

Measuring Location

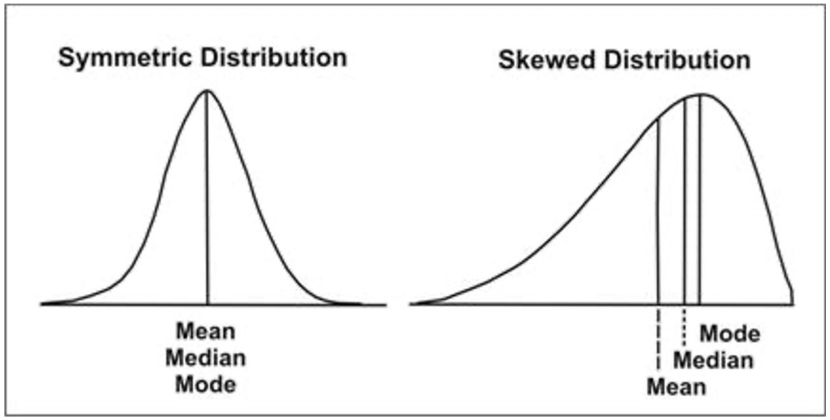
Three common measures of the distribution location…

Mean Average (expected value) of the data distribution

Median Midpoint – 50% of the probability is below and 50% above

Mode Value of highest probability (mass or density)

E.g., [1,2,3] vs [0,10,11]
compute mean and median



…align with symmetric distributions, but diverge with asymmetry

Median

For data $x_1, x_2, \ldots, x_N$ sort the data,

$$x_{(1)}, x_{(2)}, \ldots, x_{(n)}$$

- Notation $x_{(i)}$ means the i-th *lowest* value, e.g. $x_{(i-1)} \leq x_{(i)} \leq x_{(i+1)}$
- $x_{(1)}, x_{(2)}, \ldots, x_{(n)}$ are called *order statistics*

not summary info, but rather a transformation

If n is **odd** then find the middle datapoint,

$$\text{median}(x_1, \ldots, x_n) = x_{((n+1)/2)}$$

If n is **even** then average between both middle datapoints,

$$\text{median}(x_1, \ldots, x_n) = \frac{1}{2}\left(x_{(n/2)} + x_{(n/2+1)}\right)$$

Median

What is the median of the following data?

1, 2, 3, 4, 5, 6, 8, 9 **4.5**

What is the median of the following data?

1, 2, 3, 4, 5, 6, 8, 100 **4.5**

Median is *robust* to outliers

Sample Mean

Empirical estimate of the true mean of the data distribution,

$$\bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

Alternative definition: if the value x occurs n(x) times in the data then,

$$\bar{x} = \frac{1}{N} \underbrace{\sum_{x}}_{\text{for the unique values of } \{x_1, \ldots, x_N\}} x n(x) = \sum_{x} x p(x) \quad \text{where} \quad \underbrace{p(x) = \frac{n(x)}{N}}_{\text{Empirical Distribution}}$$

Recall

- Law of Large Numbers says $\bar{x}$ goes to mean E[X]
- Central Limit Theorem says $\bar{x}$ has Normal distribution, asymptotically.

Sample Mean

Example 2.1. *For the data set* $\{1, 2, 2, 2, 3, 3, 4, 4, 4, 5\}$, *we have* $n = 10$ *and the sum*

$$\begin{aligned} 1+2+2+2+3+3+4+4+4+5 &= 1n(1) + 2n(2) + 3n(3) + 4n(4) + 5n(5) \\ &= 1(1) + 2(3) + 3(2) + 4(3) + 5(1) = 30 \end{aligned}$$

Thus, $\bar{x} = 30/10 = 3.$

Sample Mean

↓
(bacterium)

Example 2.2. *For the data on the length in microns of wild type* Bacillus subtilis *data, we have*

length x	frequency $n(x)$	proportion $p(x)$	product $xp(x)$
1.5	18	0.090	0.135
2.0	71	0.355	0.710
2.5	48	0.240	0.600
3.0	37	0.185	0.555
3.5	16	0.080	0.280
4.0	6	0.030	0.120
4.5	4	0.020	0.090
sum	200	1	2.490

So the sample mean $\bar{x} = 2.49$.

Sample Mean

For any real-valued function h(x) we can compute the mean as,

$$\overline{h(x)} = \frac{1}{N} \sum_{i=1}^{N} h(x_i)$$

Note $\overline{h(x)} \neq h(\bar{x})$ in general.

Example Compute the average of the square of values,

{ 1, 2, 3, 4, 5, 5, 6 }

$$\overline{x^2} = \frac{1}{7}(1 + 2^2 + 3^3 + 4^2 + 2(5^2) + 6^2) \approx 16.57$$

$(\bar{x})^2 \approx 13.80$

Weighted Mean

In some cases we may weigh data differently,

$$\sum_{i=1}^{N} w_i x_i \quad \text{where} \quad \sum_{i=1}^{N} w_i = 1 \qquad 0 \leq w_i \text{ for } i = 1, \ldots, N$$

For example, grades in a class:

$$\text{Grade} = 0.2 \cdot x_{\text{midterm}} + 0.2 \cdot x_{\text{final}} + 0.6 \cdot x_{\text{homework}}$$

Grading Breakdown (example)

- Homework: 60%
- Midterm: 20%
- Final: 20%

Measuring Spread

We have seen estimates of spread via the sample variance,

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

Biased

$$s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

Unbiased

But you might be interested in more detailed information about the spread.

For example, fraction of people with heights <= 5 feet

Measuring Spread

Quartile divide data into 4 equally-sized bins,

- **1st Quartile** : Lowest 25% of data
- **2nd Quartile** : Median (lowest 50% of data)
- **3rd Quartile** : 75% of data is below 3rd quartile
- **4th Quartile** : All the data… not useful

Compute using `np.quantile()` :

```
x = np.random.rand(10) * 100
q = np.quantile(x, (0.25, 0.5, 0.75))
np.set_printoptions(precision=1)
print( "X: " , x )
print( "Q: " , q )
```

```
X:  [90.7 73.9 31.7  2.8 56.3 95.7 15.6 75.8  4.1 19.5]
Q:  [16.6 44.  75.3]
```

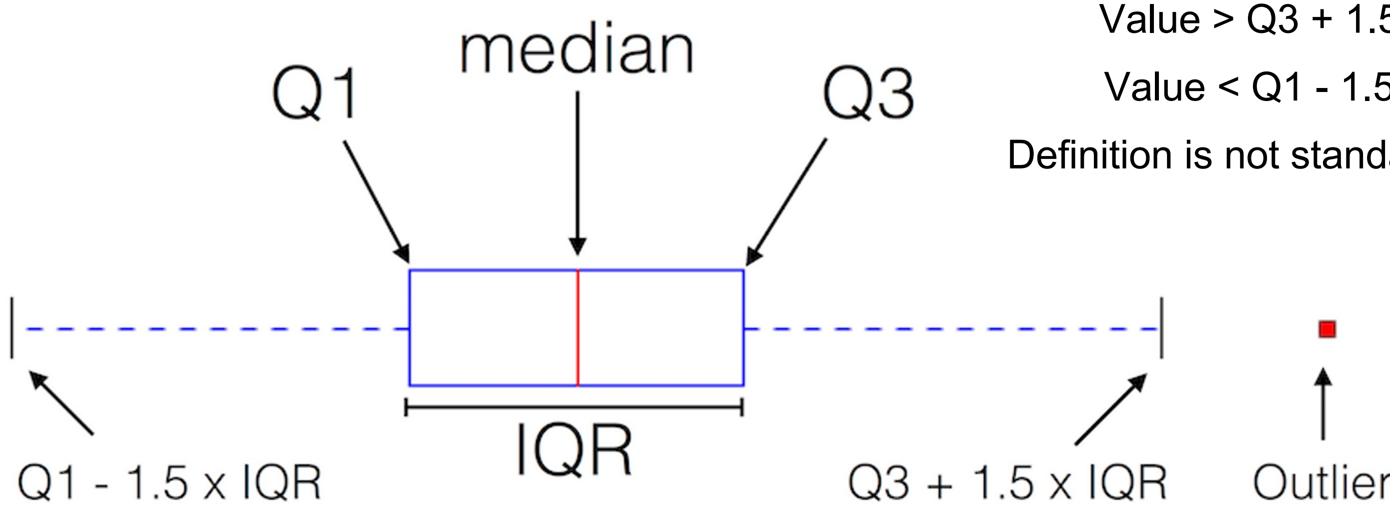
Box Plot

Outliers Are atypical data,

Value > Q3 + 1.5 x IQR

Value < Q1 - 1.5 x IQR

Definition is not standard.



Interquartile-Range (IQR) Measures interval containing 50% of data

$$IQR = Q3 - Q1$$

Region of *typical* data

Box Plot

48 52 57 61 64 72 76 77 81 85 88

Median

48 52 57 61 64 72 76 77 81 85 88

$\underbrace{48 \quad 52 \quad 57 \quad 61 \quad 64 \quad 72}_{\text{First half}} \quad \underbrace{72 \quad 76 \quad 77 \quad 81 \quad 85 \quad 88}_{\text{Second half}}$

Q1 Q3

$\underbrace{48 \quad 52 \quad 57 \quad 61 \quad 64 \quad 72}_{\text{First half}} \quad \underbrace{72 \quad 76 \quad 77 \quad 81 \quad 85 \quad 88}_{\text{Second half}}$

$Q1 = \frac{57 + 61}{2} = 59$

$Q3 = \frac{77 + 81}{2} = 79$

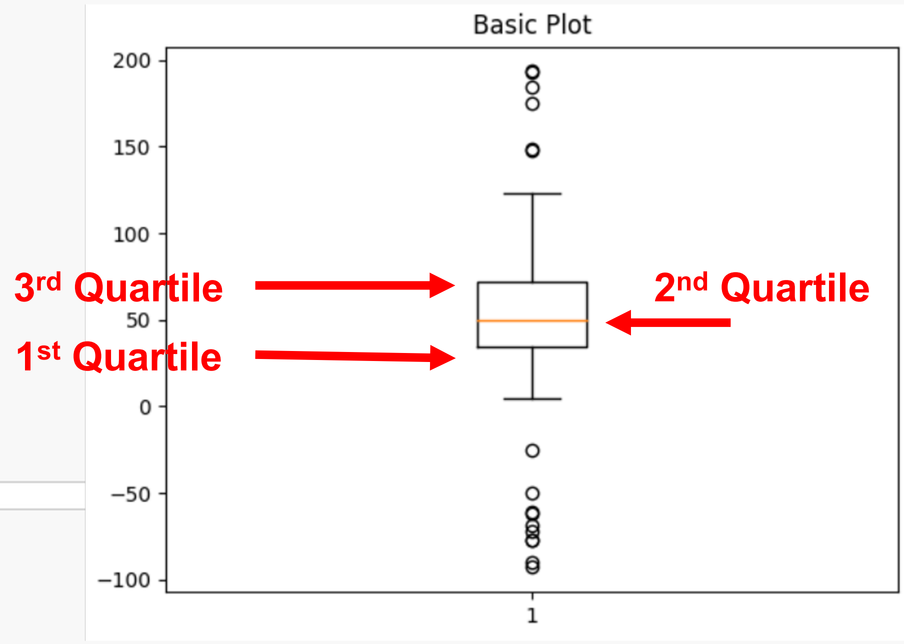
$IQR = Q3 - Q1$

$IQR = 79 - 59 = 20$

Box Plot

```python
import numpy as np
import matplotlib.pyplot as plt

# Fixing random state for reproducibility
np.random.seed(19680801)

# fake up some data
spread = np.random.rand(50) * 100
center = np.ones(25) * 50
flier_high = np.random.rand(10) * 100 + 100
flier_low = np.random.rand(10) * -100
data = np.concatenate((spread, center, flier_high, flier_low))
```

```python
fig1, ax1 = plt.subplots()
ax1.set_title('Basic Plot')
ax1.boxplot(data)
```



SciPy

Python-based ecosystem for math, science and engineering.

As usual, install with Anaconda:

```
> conda install scipy
```

Or with PyPI:

```
> pip install scipy
```

SciPy includes some libraries that directly works with:

SciPy Statistics

SciPy

To compute summary stats (e.g., **mode**):

```
>>> a = np.array([[6, 8, 3, 0],
...               [3, 2, 1, 7],
...               [8, 1, 8, 4],
...               [5, 3, 0, 5],
...               [4, 7, 5, 9]])
>>> from scipy import stats
>>> stats.mode(a)
ModeResult(mode=array([[3, 1, 0, 0]]), count=array([[1, 1, 1, 1]]))
```

numpy has mean, but not mode.

- **numpy** provides popular numerical functions.
- **scipy** provides more serious & specialized functions.

kind of stupid example; tie breaking leads to choose the smallest value

Compute the mode of the whole array set `axis=None`:

```
>>> stats.mode(a, axis=None)
ModeResult(mode=array([3]), count=array([3]))
```

SciPy Statistics

SciPy is a large library, so we import it in bits and pieces…

```
>>> from scipy import stats
```

Access the object norm and call its function mean(): stats.norm.mean()

In some cases, you will import only the functions that you need:

```
>>> from scipy.stats import norm
```

SciPy Statistics

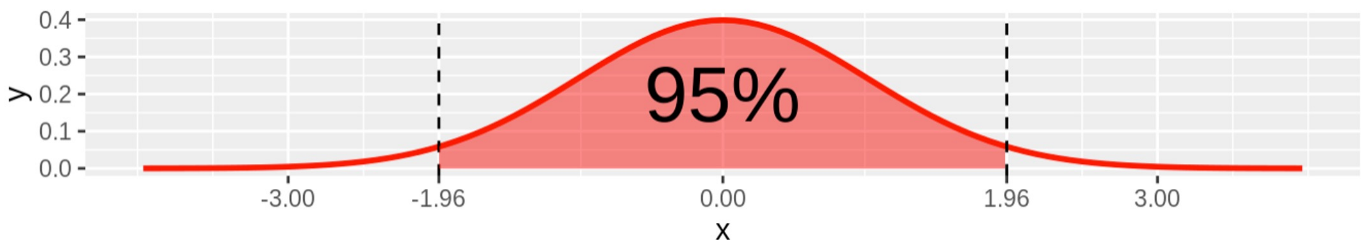
In some cases, you will import only the functions that you need:

```
>>> from scipy.stats import norm
```

contains information about the standard normal distribution

```
>>> norm.mean(), norm.std(), norm.var()
(0.0, 1.0, 1.0)
>>> norm.stats(moments="mv")
(array(0.0), array(1.0))
```

norm.ppf(0.975) returns 0.975-quantile, which is ≈ 1.96



(Fact 2) If $Z \sim \mathcal{N}(0,1)$,

$$P(Z \in [-z, z]) = 1 - 2(1 - \Phi(z))$$

where $\Phi(z) := \mathrm{P}(Z \leq z)$ is the CDF of Z.

z = 1.96: RHS $\approx .95$, 95% confident

z = 2.58: RHS $\approx .99$,

SciPy Statistics

SciPy

Other useful summary statistics:

`moment(a[, moment, axis, nan_policy])`	Calculate the nth moment about the mean for a sample.
`trim_mean(a, proportiontocut[, axis])`	Return mean of array after trimming distribution from both tails.
`iqr(x[, axis, rng, scale, nan_policy, …])`	Compute the interquartile range of the data along the specified axis.
`bootstrap(data, statistic, *[, vectorized, …])`	Compute a two-sided bootstrap confidence interval of a statistic.

do not use this for your homework

...

Anscomb’s Quartet : The Data

We’ll see the risk of looking at the **statistics** only, not the **actual data**.

Four distinct datasets of X and Y…

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

[Source: https://www.geeksforgeeks.org/anscombes-quartet/]

Anscomb’s Quartet : Summary Statistics


```
# Import the csv file
df = pd.read_csv("anscombe.csv")

# Convert pandas dataframe into pandas series
list1 = df['x1']
list2 = df['y1']

# Calculating mean for x1
print('%.1f' % statistics.mean(list1))

# Calculating standard deviation for x1
print('%.2f' % statistics.stdev(list1))

# Calculating mean for y1
print('%.1f' % statistics.mean(list2))

# Calculating standard deviation for y1
print('%.2f' % statistics.stdev(list2))

# Calculating pearson correlation
corr, _ = pearsonr(list1, list2)
print('%.3f' % corr)

# Similarly calculate for the other 3 samples

# This code is contributed by Amiya Rout
```

Summary statistics, e.g. Dataset 1:

Mean X1: 9.0

STDEV X1: 3.32

Mean Y1: 7.5

STDEV Y1: 2.03

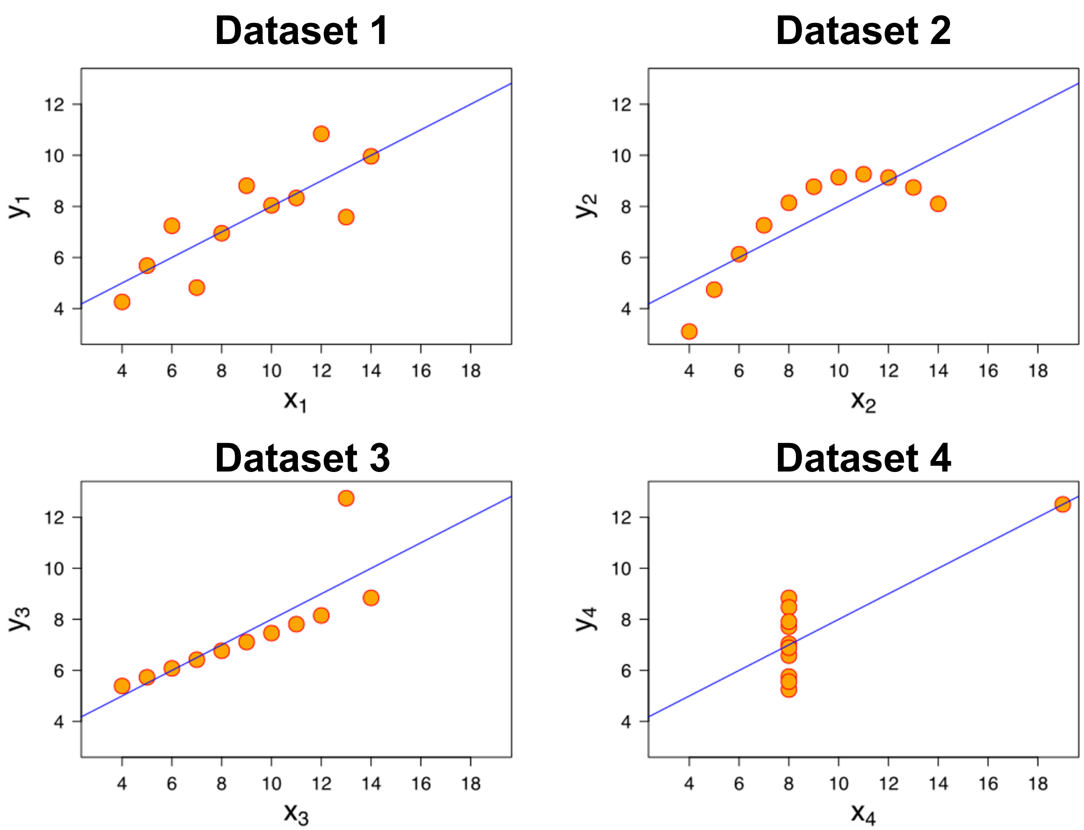
Correlation: 0.816

Actually, **all datasets have the same statistics…**

Question What can we conclude about these data? Are they the same?

[Source: https://www.geeksforgeeks.org/anscombes-quartet/]

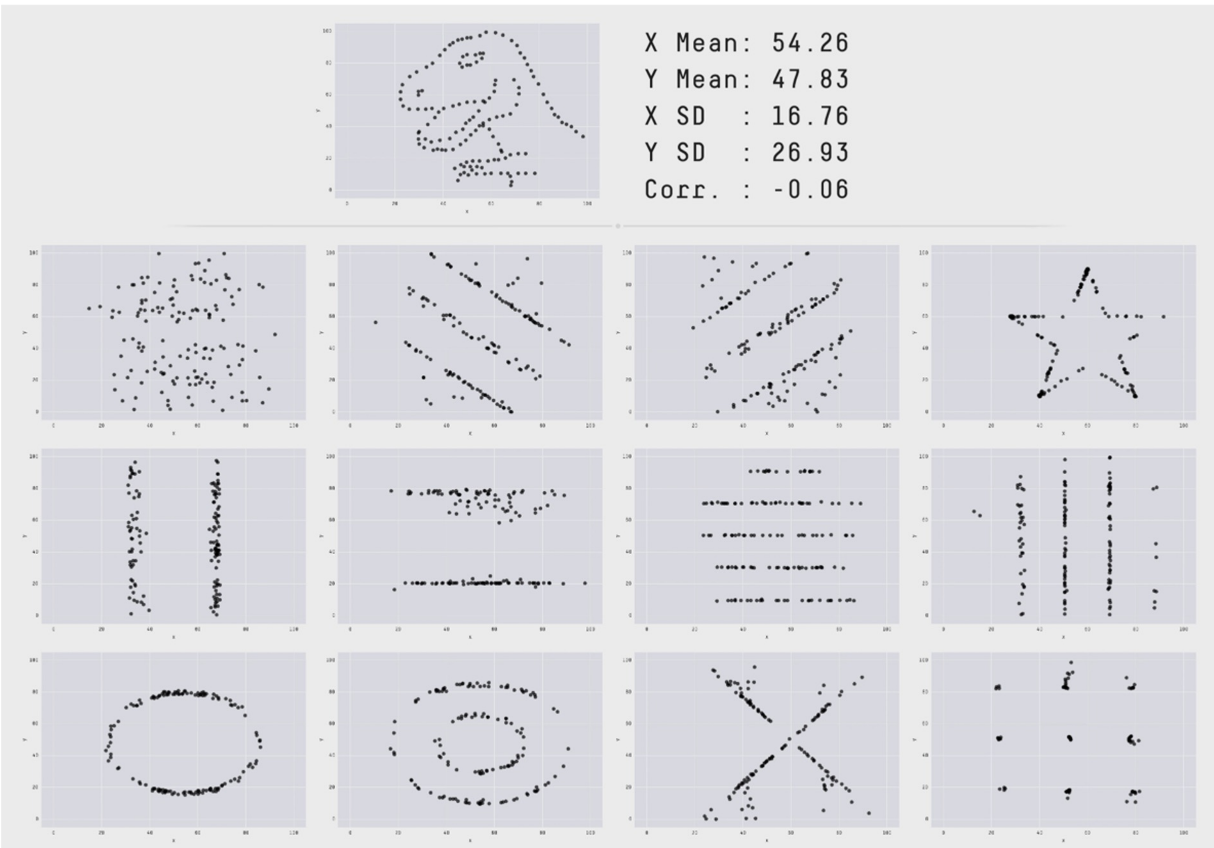
Anscomb’s Quartet : Visualization



Visualizing data clearly indicates that these are *very different* datasets…

…this highlights the **importance of visualizing data**

Datasaurus



13 datasets that all have the same summary statistics, but look very different in simple visualizations

Can be very difficult to see differences in high dimensions, however

[Source: Alberto Cairo]