

# Welcome to Data Analysis v9.0!

DAC's  
~~DKE~~ Data Analysis Students



What our parents think we do



What our friends think we do



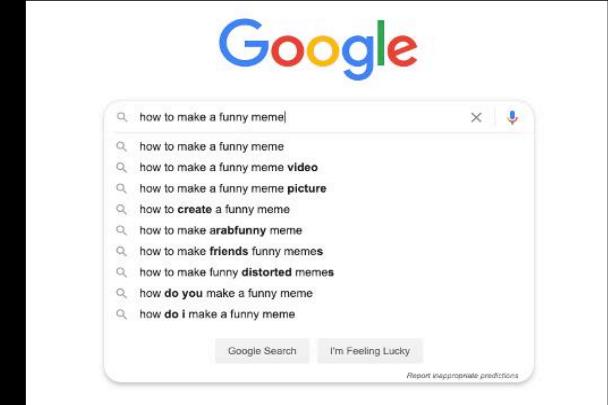
What society thinks we do



What we think we do



What Jerry thinks we do

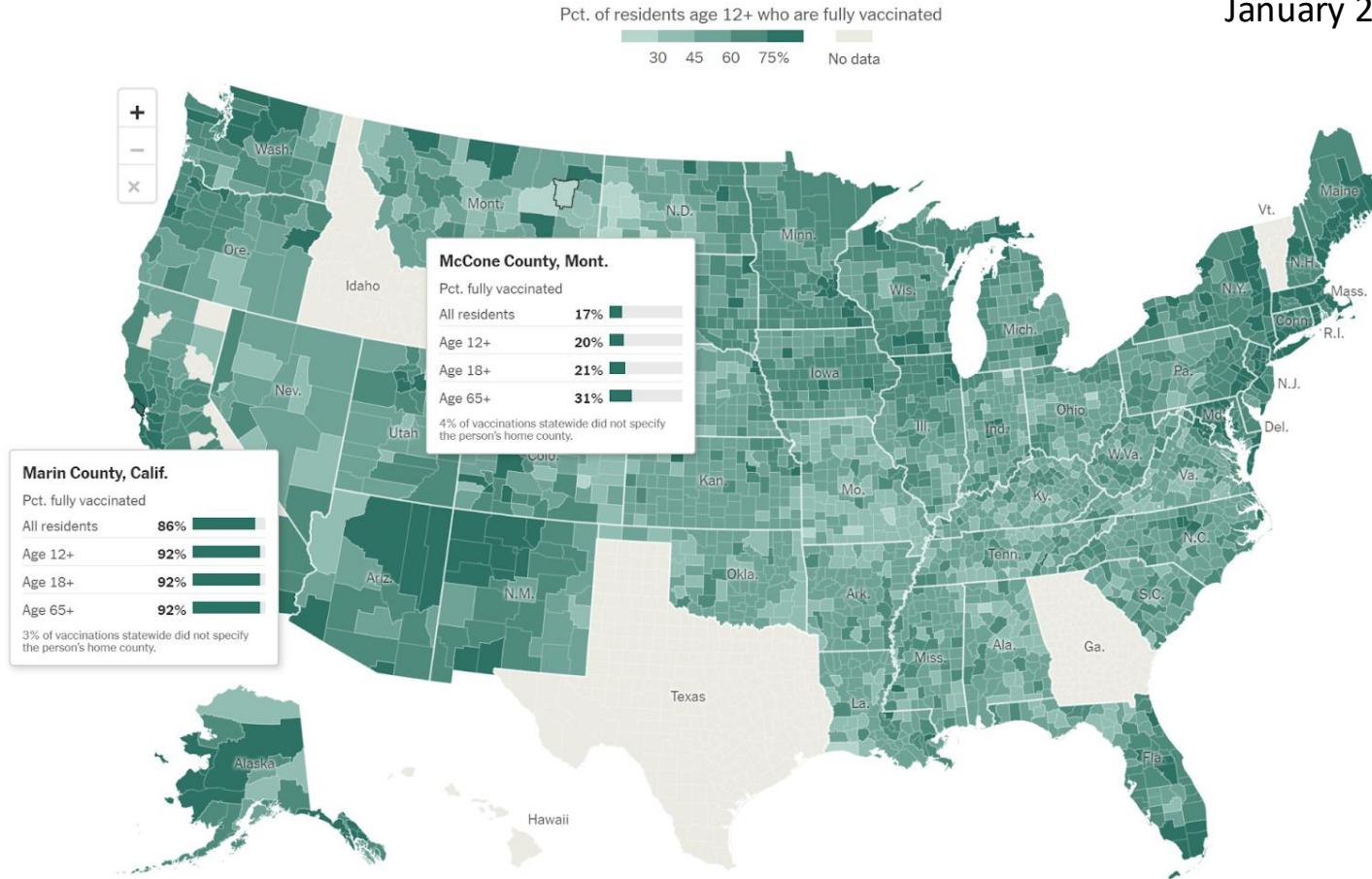


What we actually do

# Why?

[NYTimes](#)

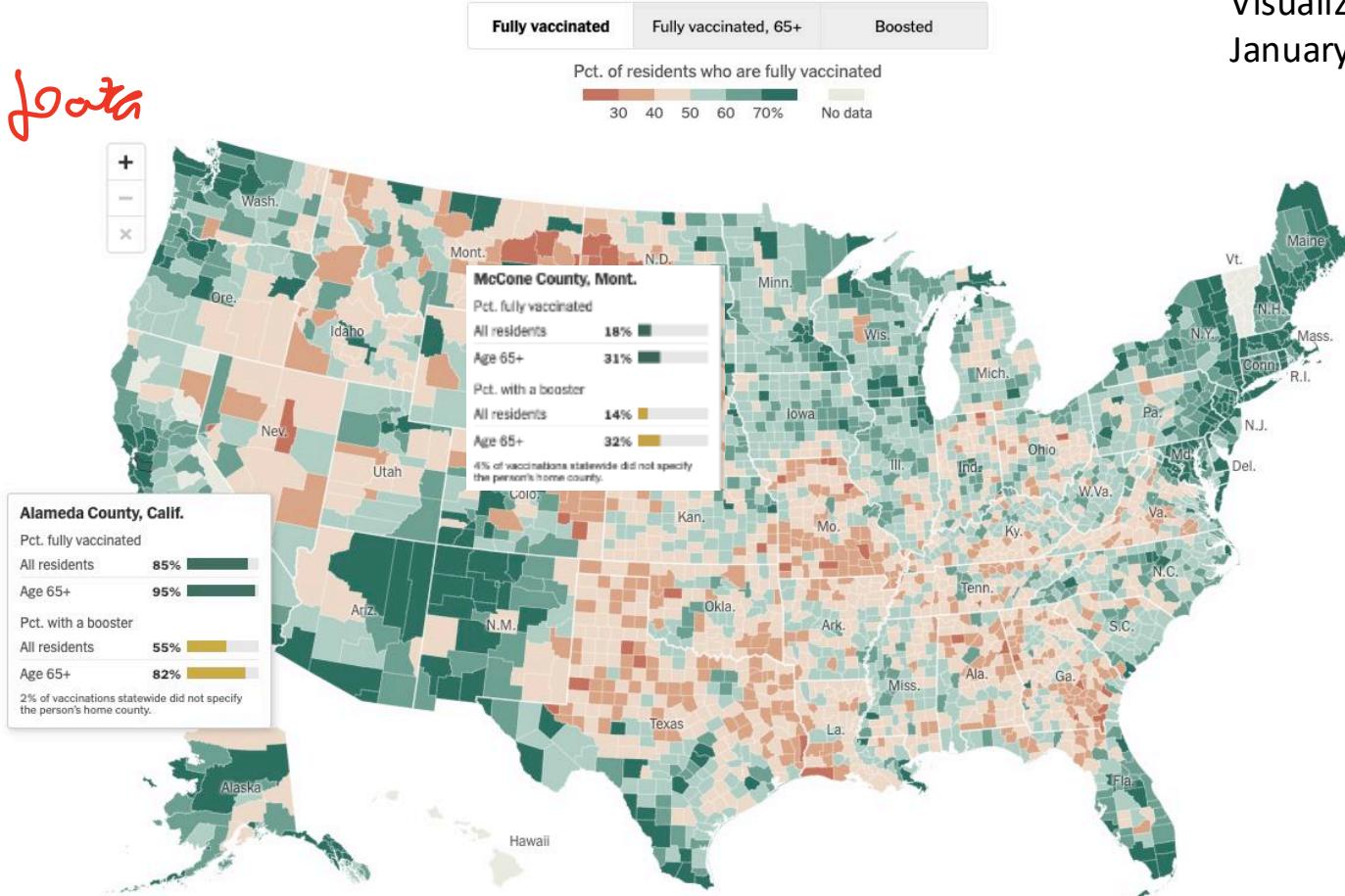
Visualization:  
January 2022



# Why?

Belief is  
social

Save Data



[NYTimes](#)

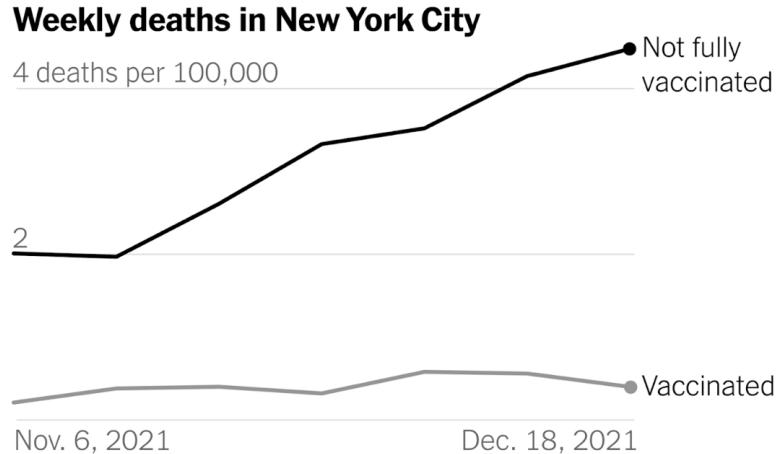
Visualization:  
January 2023

# Why?

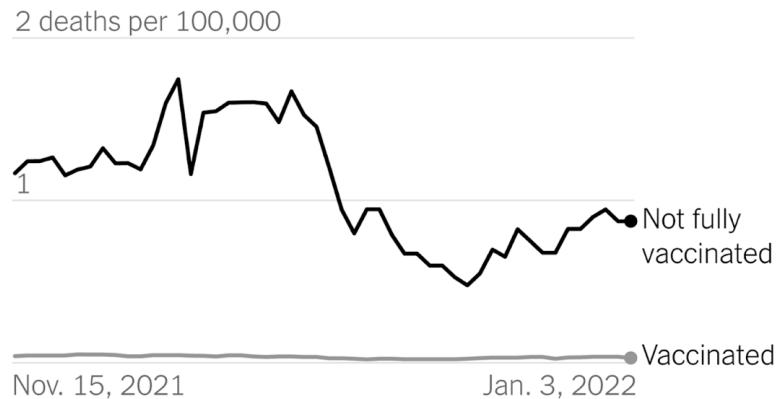
Data is a tool to  
navigate the  
complexity of our  
reality

[Link](#)

Reality is too  
Complex



**Daily average deaths in the Seattle area**



# Why?

Accident or Purpose

## But data is easy to abuse and/or misinterpret



William Makis MD  
@MakisMD

...

Stunning numbers from Denmark:

56% double vaccinated are catching 65% of "other variants" & 79% of Omicron

Most vulnerable group to Omicron BY FAR

25% boosted still catch 10% of Omicron cases, while unvaccinated catch 8.5%

This is worse than vaccine failure. This is damage.

1:08 AM · Dec 20, 2021 · Twitter Web App

805 Retweets 136 Quote Tweets

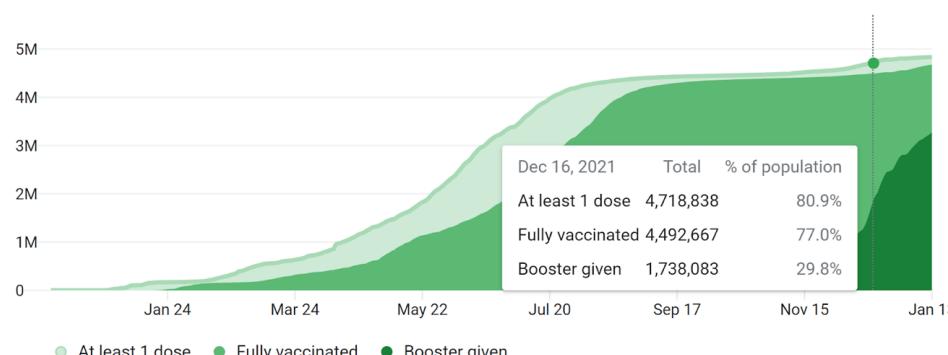
1,146 Likes

Table 4. Vaccination status for individuals ≥12 years infected with Omicron compared to other variants, data included in the table are from 22 November to 16 December 2021

Tabel 4. Vaccinationsstatus for personer ≥12 år med omikron-infektion sammenlignet med andre varianter i perioden fra og med 22. november 2021 til og med 16. december 2021

| Vaccination status<br>(12+ year olds) | Other variants<br>(No. of cases) | Other variants<br>(%) | Omicron (No.<br>of cases) | Omicron<br>(%) |
|---------------------------------------|----------------------------------|-----------------------|---------------------------|----------------|
| Booster vaccinated                    | 8,866                            | 8.6                   | 1,851                     | 10.8           |
| Fully vaccinated                      | 67,034                           | 65.3                  | 13,546                    | 79.0           |
| Not vaccinated                        | 23,492                           | 22.9                  | 1,454                     | 8.5            |
| Received first dose                   | 3,216                            | 3.1                   | 304                       | 1.8            |
| Total                                 | 102,608                          | 99.9                  | 17,155                    | 100.0          |

Individuals aged 5-11 years have recently been invited for COVID-19 vaccination, hence the vaccination coverage is relatively low in this age group and not included in Table 4.



Sure they Realy  
Of Vaccinate  
People

# Why?

Even important entities communicate poorly

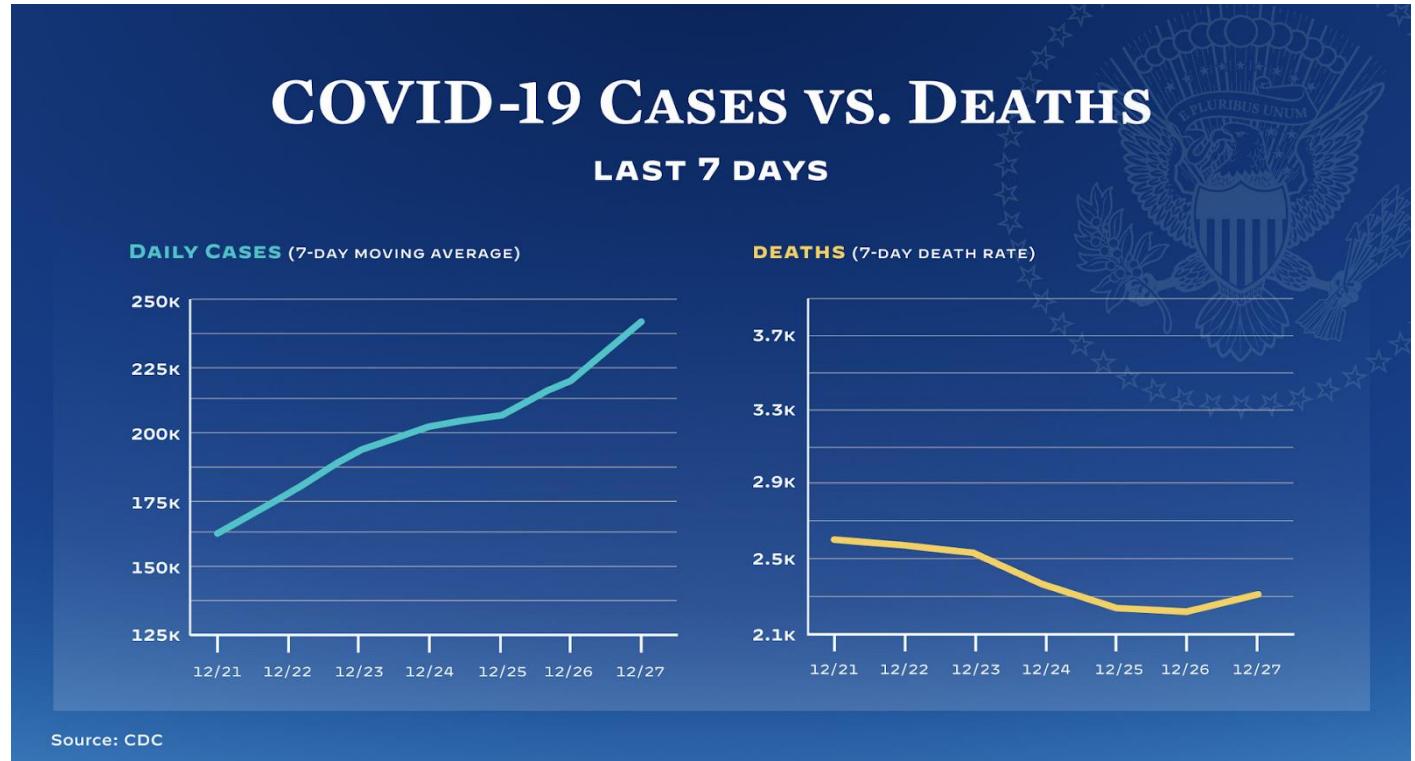


The White House   
@WhiteHouse

Omicron cases are on the rise, but it's clear that vaccines and boosters are making a difference. Vaccines and boosters help prevent severe illness and death — if you haven't already, go get your vaccine and booster.

4:56 PM · Dec 29, 2021 · The White House

Actual Data: [Link](#)



## Why?

Data Driven

Data science involves **data-focused**, ***computational*** and ***inferential*** approaches to:

- Explore and understand the world (aka science)
- Address and solve practical challenges (aka engineering)

Data Science

After this course, you should be able to take data and produce useful insights on the world's most challenging and ambiguous problems.

# What?

- Convey basic skills about all steps in data analysis (DA)

## Theory

- Key Statistical/ML concepts
- Important metrics for evaluation *of the node*
- Extracting insights from analysis of the models

## Practice

- Implement algorithms using python libraries
- Use online tools and DA resources
- Handle different kinds of data

## Impact

- Solve real-life problems using DA
- Evaluate the social impact of DA
- Communicate & visualize outcomes in a meaningful way

We can teach by  
I →

# KEN3450

# Data Analysis

**Chapter I:**  
Introduction,  
Exploratory Data Analysis &  
Effective Visualizations

Gerasimos (Jerry) Spanakis, PhD  
<http://jerryspan.github.io>

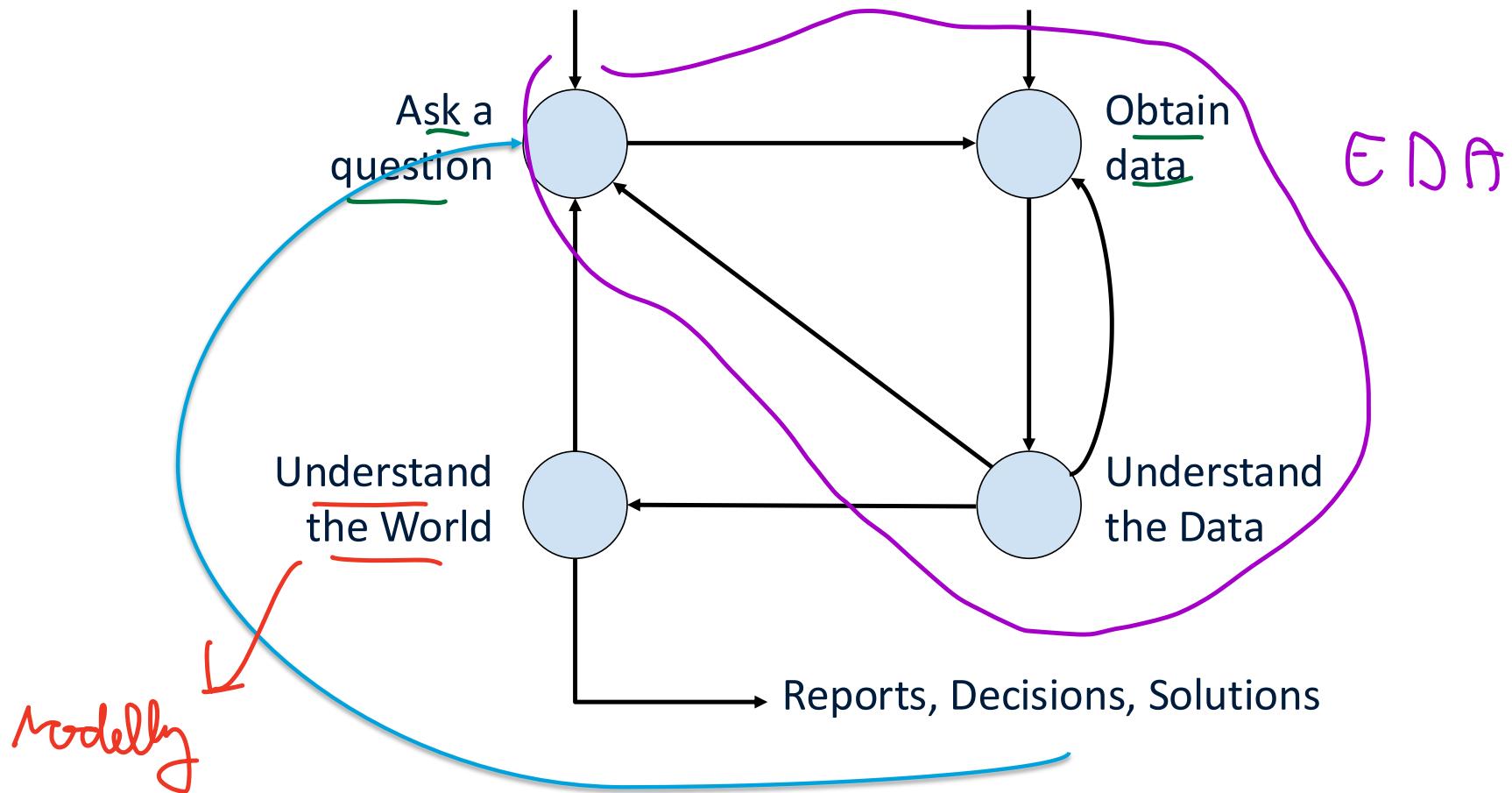


## Learning goals

- Question the source of a dataset
- Practice simple pre-processing techniques
- Use EDA as a tool to further expand the analysis *as seen in the Bacha Card*
- Understand why visualization/plotting is important
- Learn aspects that tend to make visualizes effective and ineffective
- Feel comfortable designing plots that best convey your message *the person should look and get the message*
- Gain experience in producing plots with Python

# Data science lifecycle

"Feedback loops"



# This week: EDA & Visualizations

---

Second better 1st Step



Before trying of  
Running an ML model

# EDA key properties

More on the engineering side:

- Data sources (also relates to science)
- Data formats
- Data models (databases: not our focus)
- Data storage engines & processing (MLops: not our focus)

More on the science side:

- Granularity
- Scope
- Temporality
- Faithfulness

## Data sources

- User generated *Copy User Inputs (yesterday), click*
- Systems generated
- Internal databases:  
users, inventory, customer relationships
- Third-party data *Creepy found us*

# Data sources

↓ *Behavioral Data*

| Users generated data                 | Systems generated data<br><i>System Generated</i>  |
|--------------------------------------|--|
| User inputs<br><i>Can't be Clean</i> | Logs, metadata, predictions  |
| Easily mal-formatted                 | Easier to standardize  |
| Need to be processed <b>ASAP</b>     | OK to process periodically<br>(unless to detect problems ASAP)<br><i>We can Process until<br/>See them</i> |

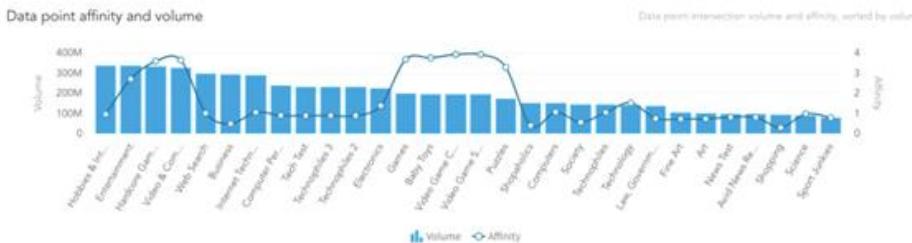
Users' behavioral data (clicks, time spent, etc.) is often system-generated but is considered **user data**

# Third-party data: creepy but fascinating

- Types of data
  - social media, income, job
- Demographic group
  - men, age 25-34, work in tech
- More available with Mobile Advertiser ID
- Useful for learning features
  - people who like A also like B

## Top interests

They love computing and electronic entertainment. If you want to reach players, try targeting at their top interests.



## Remote working

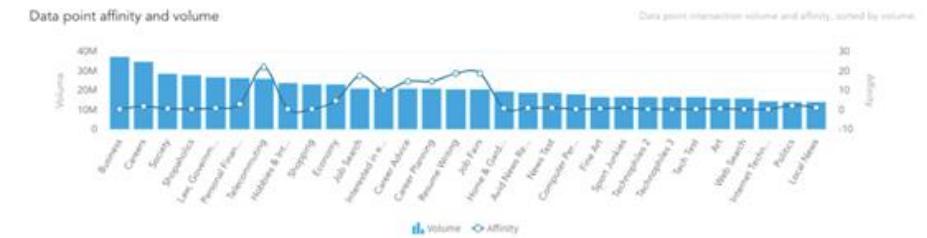
Millions of people decided to #stayhome and work remotely to limit the spread of coronavirus. Use our Remote working segment to easily reach them and show software or products that will help them stay effective.

## How did we build the segment?

Our segment includes profiles of users who recently read articles, watched videos or used mobile apps which refers to:

- remote working
- effective ways of working from home
- tools for remote workers
- homeschooling and e-learning

If you want to reach remote workers, try to extend your target group by selecting the top interests, which include Telecommuting, Career Planning or Personal Finance.



## The end of tracking IDs ...

EDITORS' PICK | Jun 24, 2020, 12:38am EDT | 685,400 views

# Apple Just Crippled IDFA, Sending An \$80 Billion Industry Into Upheaval

In response, the state-backed China Advertising Association, which has 2,000 members, has launched a new way to track and identify iPhone users called CAID, which is being widely tested by tech companies and advertisers in the country.

would like  
permission to track you  
across apps and websites  
owned by other companies.

Your data will be used to deliver  
personalized ads to you.

[Allow Tracking](#)

[Ask App Not to Track](#)

# Data formats

Most Popular

Row-major

Column-major

Single Format

| Format   | Binary/Text | Human-readable | Example use cases             |
|----------|-------------|----------------|-------------------------------|
| JSON     | Text        | Yes            | Everywhere                    |
| CSV      | Text        | Yes            | Everywhere                    |
| Parquet  | Binary      | No             | Hadoop, Amazon Redshift       |
| Avro     | Binary      | No             | Hadoop                        |
| Protobuf | Binary      | No             | Google, TensorFlow (TFRecord) |
| Pickle   | Binary      | No             | Python, PyTorch serialization |

How to store multimodal data?

{'image': [[200,155,0], [255,255,255], ...], 'label': 'car', 'id': 1} *JSON NoSQL DBs*

What are the access patterns

- How frequently the data will be accessed?
- Which hardware will we use (e.g. complex ML models run on CPU/GPU/TPU)

*Access patterns affect how fast I can access*

*The access patterns determine a lot*

## Row-major vs. column-major

We Write Right

Major / Priority

- Column-major: Count across columns → range operates on  
• stored and retrieved column-by-column Feature Level

PANDAS Column 1 Column 2 Column 3

(Only for loops in Pandas is BDX)

| Sample 1 | ... | ... | ... |
|----------|-----|-----|-----|
| Sample 2 | ... | ... | ... |
| Sample 3 | ... | ... | ... |

Row-major: easier and faster

- stored and retrieved row-to-row

row-by-row by Rows

- good for accessing samples

# Row-major vs. column-major: DataFrame vs. ndarray

## Pandas DataFrame: column-major

- accessing a row much slower than accessing a column and NumPy

*Buy Rows Is Very Slow  
(Why we avoid for loops)*

## NumPy ndarray: row-major by default

- can specify to be column-based

```
# Get the column `date`, 1000 loops
%timeit -n1000 df["Date"]
```

```
# Get the first row, 1000 loops
%timeit -n1000 df.iloc[0]
```

```
1.78 µs ± 167 ns per loop (mean ± std. dev. of 7 runs, 1000 loops each)
145 µs ± 9.41 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```

```
df_np = df.to_numpy()
%timeit -n1000 df_np[0]
%timeit -n1000 df_np[:,0]
```

```
147 ns ± 1.54 ns per loop (mean ± std. dev. of 7 runs, 1000 loops each)
204 ns ± 0.678 ns per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```

# Text vs. binary formats

|                                   | <b>Text files</b>       | <b>Binary files</b>              |
|-----------------------------------|-------------------------|----------------------------------|
| Examples                          | CSV, JSON               | Parquet                          |
| Pros                              | Human readable          | Compact                          |
| Store the number <i>1000000</i> ? | 7 characters -> 7 bytes | If stored as int32, only 4 bytes |

You can unload the result of an Amazon Redshift query to your Amazon S3 data lake in Apache Parquet, an efficient open columnar storage format for analytics. Parquet format is up to 2x faster to unload and consumes up to 6x less storage in Amazon S3, compared with text formats. This enables you to save data transformation and enrichment you have done in



# Data models

- Describe how data is represented
  - Structured vs. unstructured
- For structured data: two main paradigms:
  - Relational model (similar to the SQL model)
  - NoSQL (gives us the .json format)

Data models tells us how the data is

Represented

-Unstruktur  
-Struktur

```
# Document 1: harry_potter.json
{
    "Title": "Harry Potter",
    "Author": "J.K. Rowling",
    "Publisher": "Banana Press",
    "Country": "UK",
    "Sold as": [
        {"Format": "Paperback", "Price": "$20"},
        {"Format": "E-book", "Price": "$10"}
    ]
}

# Document 2: sherlock Holmes.json
{
    "Title": "Sherlock Holmes",
    "Author": "Conan Doyle",
    "Publisher": "Guava Press",
    "Country": "US",
    "Sold as": [
        {"Format": "Paperback", "Price": "$30"},
        {"Format": "E-book", "Price": "$15"}
    ]
}

# Document 3: the_hobbit.json
{
    "Title": "The Hobbit",
    "Author": "J.R.R. Tolkien",
    "Publisher": "Banana Press",
    "Country": "UK",
    "Sold as": [
        {"Format": "Paperback", "Price": "$30"},
    ]
}
```

↑  
Struktur

# Structured vs. unstructured data

| Structured  | Unstructured  |
|---|---|
| Schema clearly defined  | Whatever <i>(quite literally)</i>   |
| Easy to search and analyze<br><i>Because it's no well Structured.</i>               | Fast arrival (e.g. no need to clean up first)<br><i>But comes at a Cost</i> |
| Can only handle data with specific schema<br><i>Has to match the Schema Defined</i> | Can handle data <u>from any source</u><br><i>Easier to Start With</i>       |
| Schema changes will cause a lot of trouble  | <u>No need to worry about schema changes</u><br><i>Easier to Adopt</i>      |
| Data warehouses<br><i>Organized, easy to find stuff</i>                             | Data lakes<br><i>Just Dump Stuff</i><br>↑<br><i>Fit the to Sol</i>          |

# Data Storage Engines & Processing

## Databases optimized for

Transactional processing  
(OLTP)

Row fat

Analytical processing  
(OLAP)

Aggregation  
Columns

- Transactions: tweeting, ordering an Uber, uploading a new model, etc.
- Operations:
  - Insert when generated
  - Occasional update/delete

- How to get aggregated information from a large amount of data?
  - e.g. what's the average ride price last month for riders at Uber?
- Operations:
  - Mostly SELECT

Aggregate WAVC  
Po

# QLTP & OLAP are outdated terms

● OLAP  
Search term

● OLTP  
Search term

+ Add comparison

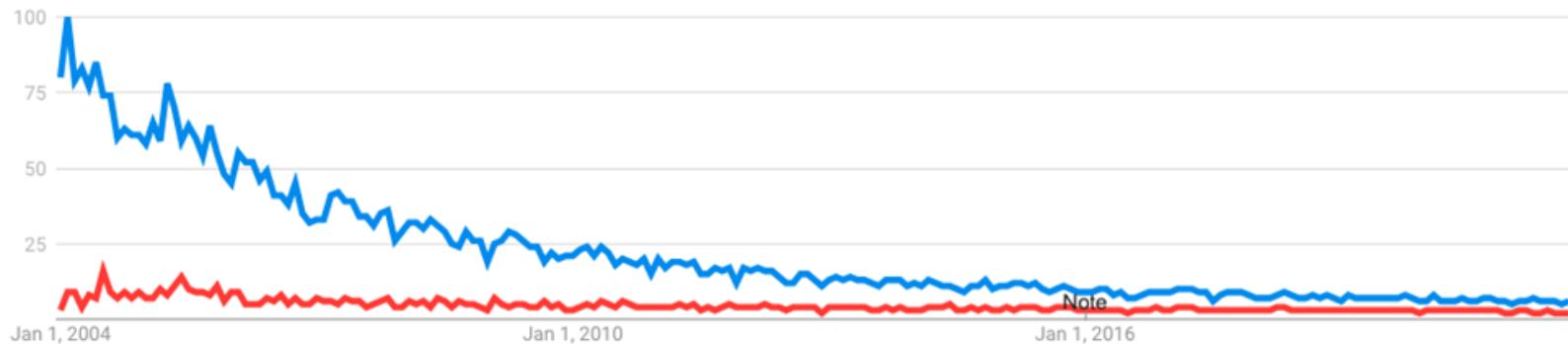
Worldwide ▾

2004 - present ▾

All categories ▾

Web Search ▾

Interest over time ?



# Decoupling storage & processing

- Good for
- Hard for Processing now

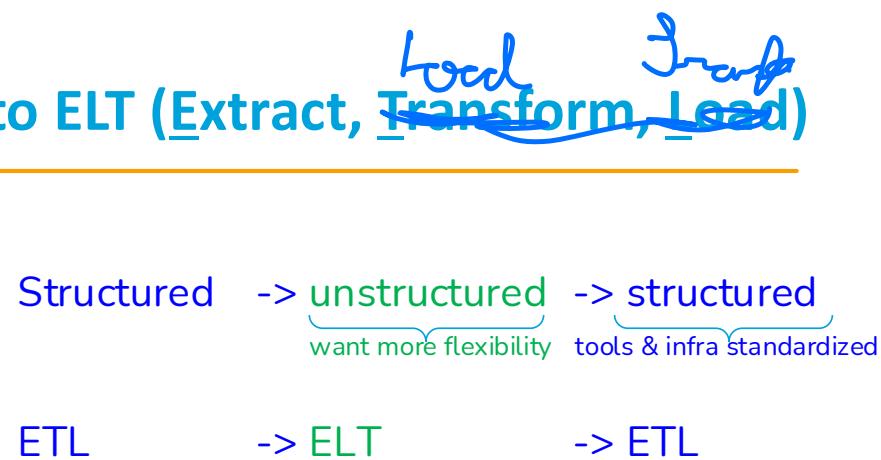
- OLTP & OLAP: how data is stored is also how it's processed
  - Same data being stored in multiple databases
  - Each uses a different processing engine for different query types
- New paradigm: storage is decoupled from processing
  - Data can be stored in the same place
  - A processing layer on top that can be optimized for different query types



teradata.



## From ETL (Extract, Transform, Load) to ELT (Extract, Transform, Load)



**Transform:** the “meaty”/“juicy” part (the “heart” of EDA)

- cleaning, validating, transposing, deriving values, joining from multiple sources, deduplicating, splitting, aggregating, etc.

ChatGPT 4o &gt;



### ETL (Extract, Transform, Load):

1. **Extract:** You take raw data from the source (databases, files, etc.).
  2. **Transform:** You clean, structure, and modify the data *before* storing it in your data warehouse.
  3. **Load:** Once the data is transformed, you store it in the data warehouse.
- **Key idea:** Transformation happens **before** loading.
  - **Limitations:** This approach can be slow for large amounts of raw data because the transformation step requires additional resources and processing before storage.

### ELT (Extract, Load, Transform):

1. **Extract:** You take raw data from the source (same as ETL).
  2. **Load:** You store the raw data in the data warehouse or data lake without making any changes.
  3. **Transform:** The data is transformed **after** being loaded into the data warehouse.
- **Key idea:** Transformation happens **after** loading.
  - **Advantages:**
    - More flexible because raw data is always available for different types of analyses

ChatGPT 4o &gt;



- **Advantages:**

- More flexible because raw data is always available for different types of analyses.
  - Modern systems (like Snowflake or BigQuery) allow for faster, scalable transformations directly in the database.

### Why the Transition to ELT?

1. **Flexibility:**

- With ELT, you store raw data and can transform it differently depending on the analysis you want to do.
  - In ETL, once the data is transformed and loaded, you lose the original raw data unless stored separately.

2. **Scalability:**

- ELT works well with modern cloud-based systems like Snowflake and Google BigQuery, which can handle large amounts of raw data and perform transformations efficiently.

3. **Unstructured Data:**

- ELT is better suited for unstructured data (e.g., text, images) because you can store raw data and process it later as needed.

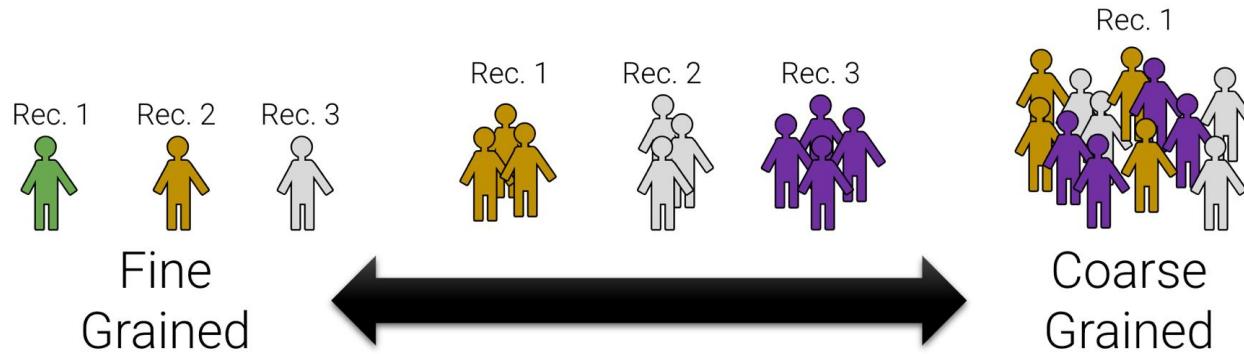
Does this explanation help clarify the transition?

Message



# EDA - Granularity

What is the Row?



- What does each record represent? *now*
- Purchases, persons, trees, ...
- Do all records capture granularity on the same level?
- If the data is coarse, how were records aggregated?

I now right be a Person and another a car

Based on what?

# EDA - Scope

- Does my data cover my area of interest?
  - E.g. I am studying crime data in Limburg but I only have data from Maastricht
- Is my data too broad?
  - E.g. *Sampling* protocol might lead to poor coverage
- Does my data cover the right time frame?
  - E.g. COVID19 test reporting



# EDA - Temporality

- Data changes – when was the data collected?
- What is the meaning of time/date fields?
  - e.g. when the event “happened”
  - e.g. when data was collected vs. entered
- Time depends on “where” VS vs Bangalore
- Are there strange null values?
  - e.g. January 1<sup>st</sup> 1970 or 1900
- What about periodicity?

*when it happened, when it  
was collected..*

*VS vs Bangalore*

*Do these data collected over a scope of time*

# EDA - Faithfulness

- Do I trust this data?
- Does my data contain unrealistic or “incorrect” values
  - Dates in the future for events in the past
  - Locations that don’t exist
  - Negative counts
  - Large outliers
  - ...
- Does my data violate obvious dependences?
  - Age and birthday don’t match
  - ...
- Was the data entered by hand?
  - Expect spelling errors
- Are there any obvious signs of data falsification?
  - Repeated names, fake looking email addresses

Dates in the Future  
Negative Count  
Large Outliers  
...  
Unrealistic Values

## Common feature engineering operations

Common

1. Handling missing values
2. Scaling
3. Discretization
4. Categorical features
5. Feature crossing

## Handling missing values

- Not all missing values are equal
  - Missing not at random (MNAR)
  - Missing at random (MAR)
  - Missing completely at random (MCAR)

There are 3 types of  
Missing Values



3 Types

Applies to missing because of the value

## Handling missing values

Missing not at random – when a value is missing due to the value itself

missing because of the value  
Jhp

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      | (\$350,000?)  |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      | (\$350,000?)  | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

Value could not be found

## Handling missing values

Missing at random – when a value is missing due to another observed variable

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      |               |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      |               | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

Random because we still do not know the reason. Ages may in the

1st Place

But we can  
see that  
all the line  
we have

Gender A,  
the Data  
is noisy

Ages may because of Gender A

## Handling missing values

What we usually think as  
Random

Missing completely at random – there is no pattern to which values are missing

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      |               |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      |               | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

## Handling missing values

How do we handle them?

- Deletion – removing data with missing entries *like Ron or Carl*
- Imputation – filling missing fields with certain values

Many people prefer deletion not because it's better, but it's easier to do

Once we see that there are missing values, then one of the things we can do: Delete      Impute

# Handling missing values

- Deletion
  - Column deletion – remove columns with too many missing entries *But could be a Useful Feature*
    - drawbacks – even if half the values are missing, the remaining data still potentially useful information for predictions *We are Deleting a Feature*
    - e.g. even if over half the column for 'Marital status' is missing, marital status is still highly correlated with house purchasing *In this case, deleting this feature could reduce a lot our predictive power*
  - Row deletion

|                |
|----------------|
| Marital status |
|                |
|                |
|                |
| Married        |
|                |
|                |
| Single         |
|                |
| Single         |
|                |

# Handling missing values

- **Deletion**
  - Column deletion
  - Row deletion



# Handling missing values

- Row deletion
  - Good for: data missing completely at random (MCAR) and few values missing

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  | 39  | A      | 150,000       | Married        | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        | Single         | 0                  | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      | 75,000        | Married        | 2                  | Engineer | Yes  |
| 5  | 35  | B      | 35,000        | Single         | 0                  | Doctor   | Yes  |
| 6  | 32  | A      | 50,000        | Married        | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         | 2                  | Teacher  | No   |
| 8  | 20  | B      | 10,000        | Single         | 1                  | Student  | No   |

# Handling missing values

- Row deletion
  - Bad when many examples have missing fields

We could end up deleting almost whole data

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      |               |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      |               | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

# Handling missing values

We could Probably Fix It  
↑

- Row deletion

- Bad for: missing values are not at random (MNAR)
- Missing information is information itself

For example, here we would find out about the front

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      | \$350,000?    |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      | \$350,000?    | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

# Handling missing values

- Row deletion
  - Bad for: missing data at random (MAR)
  - Can potentially bias data – we've accidentally removed all examples with gender 'A'

In row we delete all with  
With gender A, since all  
have missed marital stat

| ID | Age | Gender | Annual income | Marital status | Number of children | Job      | Buy? |
|----|-----|--------|---------------|----------------|--------------------|----------|------|
| 1  |     | A      | 150,000       |                | 1                  | Engineer | No   |
| 2  | 27  | B      | 50,000        |                |                    | Teacher  | No   |
| 3  |     | A      | 100,000       | Married        | 2                  |          | Yes  |
| 4  | 40  | B      |               |                | 2                  | Engineer | Yes  |
| 5  | 35  | B      |               | Single         | 0                  | Doctor   | Yes  |
| 6  |     | A      | 50,000        |                | 0                  | Teacher  | No   |
| 7  | 33  | B      | 60,000        | Single         |                    | Teacher  | No   |
| 8  | 20  | B      | 10,000        |                |                    | Student  | No   |

We get rid of all the People with Gender A

# Imputation "Fill in the Values"

- Fill missing fields with certain values

- Defaults
  - e.g. 0, or the empty string, etc.
- Statistical measures – mean, median, mode → not Full
  - e.g. if a day in July is missing its temperature value, fill it with the median temperature in July

We can be Smart

Depending on the dataset and the application, there might be a different solution applicable!

We can also try to Predict

# Scaling of Data in Some Context

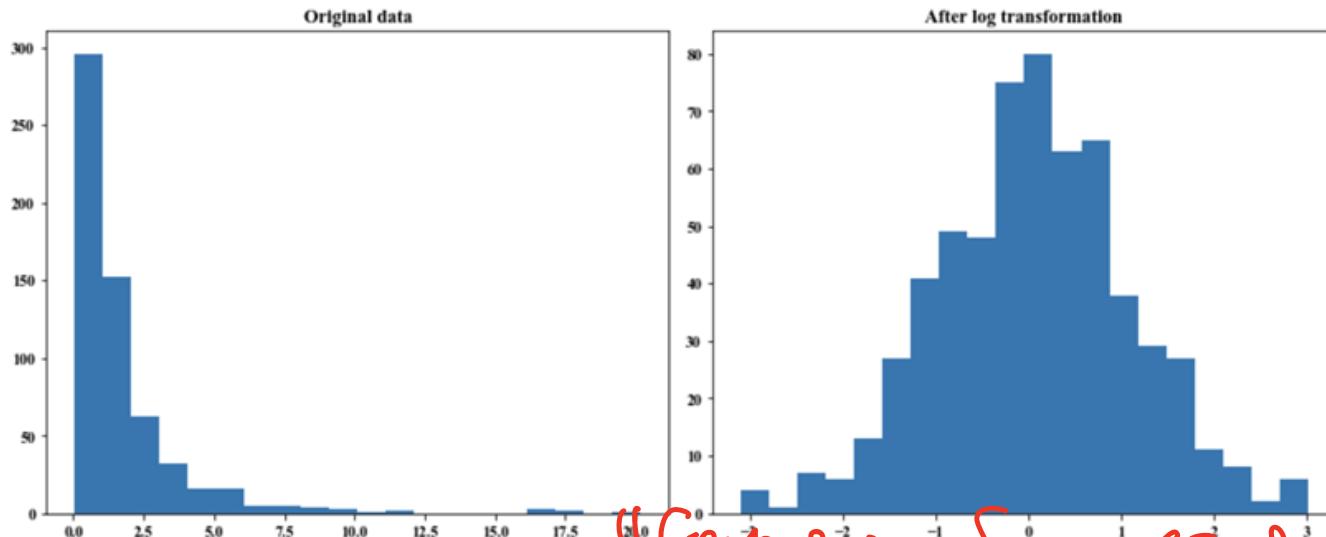
| scaling type  | use case  |
|---|---|
| <u>min/max</u> <del>dataset</del><br><u>normalization</u> | Any -- <u>no assumptions about variables</u><br><u>We can do it without fear with no loss</u> |
| <u>z-score</u><br><u>normalization</u>                    | When variables follow a <u>normal distribution</u><br><u>Assume</u>                           |
| <u>log scaling</u>  | When variables follow an <u>exponential distribution</u><br><u>↳ Sigmoid Data</u>             |

- scaling can be a common source of data leakage (more on this later in course)
- scaling variables requires global statistics

## Log scaling

Helps w/ Skewed Data

- Help with skewed data
- Often gives performance gain



"Compresses Large Values And  
Stretches Smaller Ones"

## Discretization

Continuous Features ↓ Discretize → Bucket

- Turning a continuous feature into a discrete feature (quantization)
- Create buckets for different ranges
  - Incorporate knowledge/expertise about each variable by constructing specific buckets
- Examples
  - Income
    - Lower income:  $x < \$35,000$
    - Middle income:  $\$35,000 \leq x < \$100,000$
    - High income:  $x \geq \$100,000$
  - Age
    - Minors:  $x < 18$
    - College:  $18 \leq x < 22$
    - Young adult:  $22 \leq x < 30$
    - $30 \leq x < 40$
    - $40 \leq x < 65$
    - Seniors:  $x \geq 65$

Probably doing it at Random Would not  
be a Great Idea

Group People

Might Hinge a More Natural Way

Perhaps We do not have  
that many People, So Group  
would work

## Encoding Categorical Features

Categorical Features are  
Interesting

- Example: you want to build a recommendation system for Amazon
  - There are over 2 million brands that we need to recommend

How do we encode the different brands/vendors?

Simpler: One-Hot Encoding

What if New Brand? One-hot Brand?

# Encoding Categorical Features

- one-hot encoding! How do we handle a new brand that wants to join Amazon?  
*By Grouping Diversity*
- → encode unseen brands with "UNKNOWN"  
Problem! "UNKNOWN" was not seen during training, so none of the products in this category are being recommended
- → Group bottom 1% of brands and newcomers into "UNKNOWN" category
- Potential further issues: All newcomers are treated the same as unpopular brands in the platform
  - Solution 1: represent each category with its attributes
  - Solution 2: Hashing (widely used in industry, considered "hacky")

*What if I have a New Brand*

Search

ChatGPT

DALL-E

Arduino Maestro

Explore GPTs

## Projects

Report 3 - KE@Work

Aucos - Semester 3

Bachelor Thesis

## Chats

EDA Questions and Answers

Anmeldung Aachen Termin buchen

GOSDT Machine Learning Project

Search

ChatGPT

DALL-E

Arduino Maestro

Explore GPTs

## Projects

Report 3 - KE@Work

Aucos - Semester 3

Bachelor Thesis

## Chats

EDA Questions and Answers

Anmeldung Aachen Termin buchen

GOSDT Machine Learning Project



João Victor Coutinho Vira... •••

Artlier

ChatGPT 4o &gt;



Sure! Representing a category (e.g., a brand) with **attributes** involves encoding the category using **features derived from its properties** rather than assigning a unique identifier or binary column. Here's how it works in more detail:

### What Does It Mean to Represent a Category with Attributes?

Instead of directly encoding the category (e.g., "Brand A"), you describe it using **meaningful characteristics**. These attributes could include:

- Average product price
- Customer ratings
- Product popularity (e.g., sales volume)
- Category of products sold (e.g., electronics, clothing)
- Region or market where the brand operates

By using these properties, the model can learn general patterns and make predictions even for **unseen categories**.

### Why Is It Better?

#### 1. Handles New Categories:

Message

ChatGPT 4o &gt;

predictions even for unseen categories.

### Why Is It Better?

#### 1. Handles New Categories:

- If a new category (e.g., a new brand) is introduced, its attributes can still provide information to the model.
- For example, a new brand might have an average rating of 4.5 and sell electronics, allowing the model to generalize its predictions.

#### 2. Better Generalization:

- The model focuses on the **characteristics** of a category rather than memorizing specific labels.
- Example: A brand with high ratings and a low price is likely to perform well, even if it's new.

#### 3. Reduces Sparsity:

- One-hot encoding creates sparse matrices (many 0s), especially with many categories.
- Attribute-based representation is more compact and efficient.

### Example

Message



Search

ChatGPT

DALL-E

Arduino Maestro

Explore GPTs

## Projects

Report 3 - KE@Work

Aucos - Semester 3

Bachelor Thesis

## Chats

EDA Questions and Answers

Anmeldung Aachen Termin buchen

GOSDT Machine Learning Project

Search

ChatGPT

DALL-E

Arduino Maestro

Explore GPTs

## Projects

Report 3 - KE@Work

Aucos - Semester 3

Bachelor Thesis

## Chats

EDA Questions and Answers

Anmeldung Aachen Termin buchen

GOSDT Machine Learning Project



João Victor Coutinho Vira...

...

ChatGPT 4o &gt;

## Example

Suppose you have the following brands:

| Brand         | Avg Price | Avg Rating | Sales Volume | Category    |
|---------------|-----------|------------|--------------|-------------|
| Brand A       | 20        | 4.5        | 1000         | Electronics |
| Brand B       | 15        | 4.2        | 800          | Clothing    |
| Brand C (new) | 22        | 4.6        | 500          | Electronics |

Instead of encoding these brands with one-hot encoding (e.g., [1, 0, 0] for Brand A), you use the attributes:

- **Brand A:** [20, 4.5, 1000, Electronics]
- **Brand C (new):** [22, 4.6, 500, Electronics]

Even though Brand C is new, its attributes allow the model to infer its likely behavior.

## When to Use This Approach

1. When you have **rich metadata** about categories.
2. When the number of categories is very large or dynamic (e.g., e-commerce platforms with new brands added frequently).
3. When you want to avoid sparsity and overfitting common in one-hot encoding.



ChatGPT 4o &gt;

you use the attributes.

- **Brand A:** [20, 4.5, 1000, Electronics]
- **Brand C (new):** [22, 4.6, 500, Electronics]

Even though Brand C is new, its attributes allow the model to infer its likely behavior.

## When to Use This Approach

1. When you have **rich metadata** about categories.
2. When the number of categories is very large or dynamic (e.g., e-commerce platforms with new brands added frequently).
3. When you want to avoid sparsity and overfitting common in one-hot encoding.

## Challenges

- Requires **feature engineering**: You need meaningful and relevant attributes for the categories.
- May not work well if the attributes are noisy or insufficient to describe the category.

If you'd like, I can help brainstorm possible attributes for your specific use case!

Message



# Hashing!

## What is Hashing in Encoding?

Hashing maps each category (e.g., brand names) to a fixed-length numeric value or hash using a **hash function**. This avoids creating a separate column for each category, as in one-hot encoding.

## How Does It Work?

### 1. Hash Function:

- A hash function takes an input (e.g., a category like "Brand A") and produces a numeric value (the hash).
- Example:  $\text{hash}(\text{"Brand A"}) \rightarrow 5$

### 2. Fixed-Length Vector:

- Instead of creating a column for every category, the hash value is used to determine the index in a fixed-length vector (e.g., a vector of size 100).
- Example:
  - "Brand A" hashes to index 5  $\rightarrow$  Increment vector[5].
  - "Brand B" hashes to index 20  $\rightarrow$  Increment vector[20].

### 3. Collisions:

- Since the vector size is fixed, multiple categories may hash to the same index (a **collision**). The model learns to work around this by observing overall patterns.

Message

...

ChatGPT 4o &gt;

60%



### 3. Collisions:

- Since the vector size is fixed, multiple categories may hash to the same index (a **collision**). The model learns to work around this by observing overall patterns.

## Advantages of Hashing

### 1. Handles New Categories:

- A new category not seen during training will still hash to a valid index, so the model can handle it without errors.

### 2. Scalability:

- Unlike one-hot encoding, hashing doesn't require creating a column for every unique category.
- For example, even if there are 1 million categories, you can use a vector of size 1000.

### 3. Memory Efficiency:

- Saves memory compared to one-hot encoding, especially for datasets with many unique categories.

### 4. Simplicity:

- Widely used in industry for large-scale data processing (e.g., in recommendation systems).



Message





## Disadvantages of Hashing

### 1. Collisions:

- Different categories can hash to the same value, causing the model to confuse them.
- This can reduce interpretability and slightly affect performance.

### 2. "Hacky" Nature:

- Since the mapping is determined by the hash function, it's harder to interpret what each hashed feature represents.

## Example

Suppose we have three categories: "Brand A", "Brand B", and "Brand C". Using a hash function:

- "Brand A" → Hash value: 12 → Maps to vector[12].
- "Brand B" → Hash value: 5 → Maps to vector[5].
- "Brand C" → Hash value: 12 → **Collision** with "Brand A" → Both map to vector[12].

Final vector:

[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, ...]



## Example

Suppose we have three categories: "Brand A", "Brand B", and "Brand C". Using a hash function:

- "Brand A" → Hash value: 12 → Maps to vector[12].
- "Brand B" → Hash value: 5 → Maps to vector[5].
- "Brand C" → Hash value: 12 → **Collision** with "Brand A" → Both map to vector[12].

Final vector:

[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 2, 0, ...]

Here, index 12 has a value of 2 because both "Brand A" and "Brand C" mapped to it.

## When to Use Hashing

- When there are **too many categories** to use one-hot encoding efficiently.
- When the dataset is **dynamic**, with new categories added frequently (e.g., new brands on an e-commerce platform).

Let me know if you'd like more clarification!

Message



# Feature Crossing

- Combine two or more features to create a new feature

|                     |           |            |           |           |            |
|---------------------|-----------|------------|-----------|-----------|------------|
| Marriage            | Single    | Married    | Single    | Single    | Married    |
| Children            | 0         | 2          | 1         | 0         | 1          |
| Marriage & children | Single, 0 | Married, 2 | Single, 1 | Single, 0 | Married, 1 |

New one

- Helps models learn non-linear relationships between variables
- Warning – feature crossing can blow up your feature space
  - e.g. Feature A and B both have 100 categories → Feature A x B will have 10,000 categories
  - Apart from scalability issues, increases the risk of overfitting

ChatGPT 4o &gt;



## What is Feature Crossing?

- Feature crossing involves **combining two or more features** to create a new feature.
- It is useful for capturing **interactions between features** and learning **non-linear relationships**.

## How Does it Work?

1. Take two features and combine their values into a new feature.
  - Example: "Marriage" (Single/Married) and "Children" (0, 1, 2, etc.).
  - The crossed feature could be represented as:
    - "Single, 0"
    - "Married, 2"
    - "Single, 1"
2. The new feature includes interactions:
  - For instance, "Single with 2 children" might mean something different from "Married with 2 children."

## Why Use Feature Crossing?



Message

ChatGPT 4o &gt;



## Why Use Feature Crossing?

- **Learn Non-Linear Relationships:**
  - Some relationships between variables are not linear, and crossing features helps capture these.
  - For example, the effect of having children might depend on marital status.
- **Improved Model Accuracy:**
  - Especially useful for tree-based models and neural networks, where interaction terms can significantly enhance performance.

## Challenges and Warnings

1. **Feature Space Explosion:**
  - When features have many unique categories, crossing them can result in a massive number of combinations.
    - Example:
      - Feature A has 100 categories, and Feature B has 100 categories.
      - Feature A × B will result in **10,000 categories**.
    - This can lead to memory and computation issues.
2. **Overfitting:**



More features mean more chances for the model to fit noise rather than signal.

Message





## 2. Overfitting:

- More features mean more chances for the model to fit noise rather than meaningful patterns, especially with limited data.

## 3. Scalability Issues:

- For large datasets or real-time systems, handling a large feature space becomes impractical.

---

## When to Use Feature Crossing?

- When domain knowledge suggests that **interactions** between features are meaningful.
  - Example: Combining "Season" and "Product Type" in sales data could reveal specific patterns like "Winter + Jackets."
- In models that can handle sparse features efficiently (e.g., tree-based models, logistic regression with regularization).

---

## Example

If you have two features:

- "Age Group" with categories: [Child, Adult, Senior]
- "Region" with categories: [Urban, Suburban, Rural]



- Example: Combining "Season" and "Product Type" in sales data could reveal specific patterns like "Winter + Jackets."
- In models that can handle sparse features efficiently (e.g., tree-based models, logistic regression with regularization).

---

## Example

If you have two features:

- "Age Group" with categories: [Child, Adult, Senior]
- "Region" with categories: [Urban, Suburban, Rural]

Feature crossing creates combinations:

- "Child-Urban," "Adult-Rural," "Senior-Suburban," etc.

---

## Key Takeaways

- Benefits:** Helps capture complex relationships, boosts accuracy.
- Risks:** Increases computational costs, risk of overfitting, and complexity.
- Use only when there's evidence that feature interactions matter.

Let me know if you need further clarification or examples!

Message



## Data types

then

→ Big Categ: - categorical

- **Nominal:** categorical data, no ordering
  - Example: Pet: {dog, cat, rabbit, ...}
  - Operations:  $=, \neq$  *essentially we can only compare if they are the same or different*
- **Ordinal:** categorical data, with ordering
  - Example: Rating: {1,2,3,4,5}
  - Operations:  $=, \neq, \geq, \leq, >, <$  *No "Distance"*
- **Interval:** numerical data, zero doesn't mean zero "quantity"
  - Example: Temperature Fahrenheit
  - Operations:  $=, \neq, \geq, \leq, >, <, +, -$
- **Ratio:** numerical data, zero has meaning related to zero "quantity"
  - Example: Weight, Temperature Kelvin
  - Operations:  $=, \neq, \geq, \leq, >, <, +, -, /$

NUMERICAL

○ is part of quantity

# Data types: Descriptive statistics (and plots)

- We are going to focus on the **type** and **dimensionality** of the underlying data
  - (Not an exhaustive list)
    - 1D: bar chart, pie chart, histogram, boxplot
    - 2D: scatter plot, line plot, stacked box & whisker plots, heatmap
    - 3D+: scatter matrix, bubble chart
- One of the  
we can only Plot  
in  
up-to 3D*

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

## Interval and ratio data: Descriptives

↑ Because they are Numerical  
Quantitative or Numeric

- Min, max, median, mean (or average)

In [12]:

```
# now use numpy to compute descriptive statistics for ages
print(len(ages))
print(np.min(ages))
print(np.max(ages))
print(np.mean(ages))
print(np.median(ages))
```

```
32561
17
90
38.58164675532078
37.0
```

Makes Sense + Compute Statistics  
Summarizes Numerical Data

|          | Data |
|----------|------|
| Nominal  | x    |
| Ordinal  | x    |
| Interval | ✓    |
| Ratio    | ✓    |

}

only for quantitative

Data →

Count per bin (range of values)

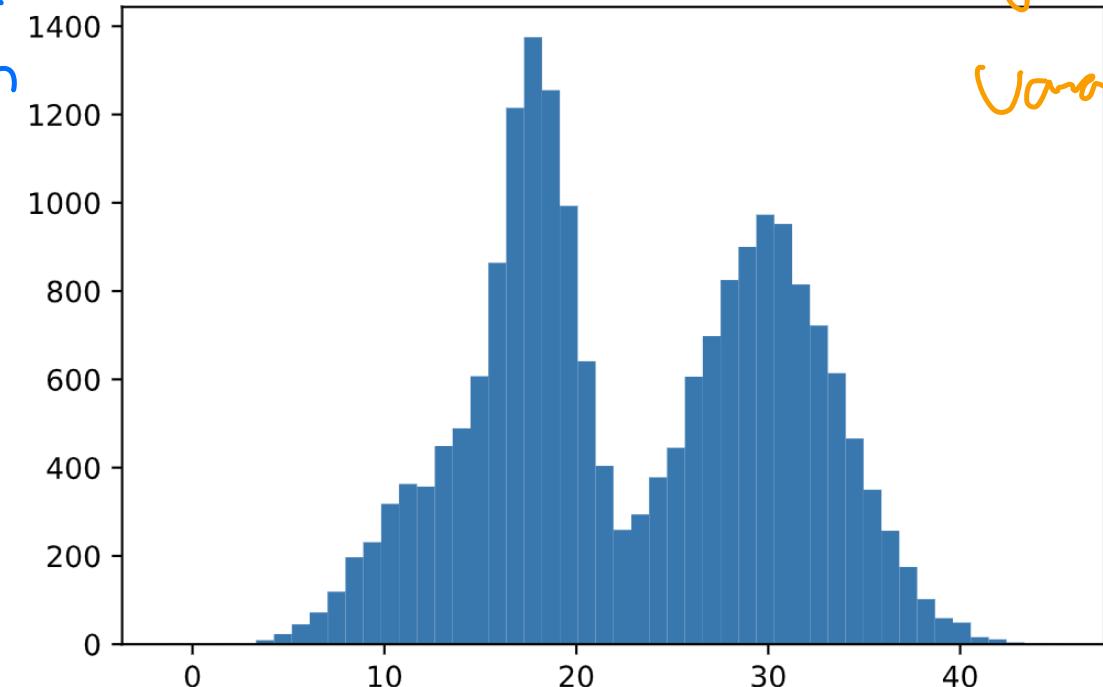
→ Plot with bar chart

Bar chart we get

`plt.hist(x_values, density=True)`

`sns.histplot(data=df, x="x_column", stat="density")`

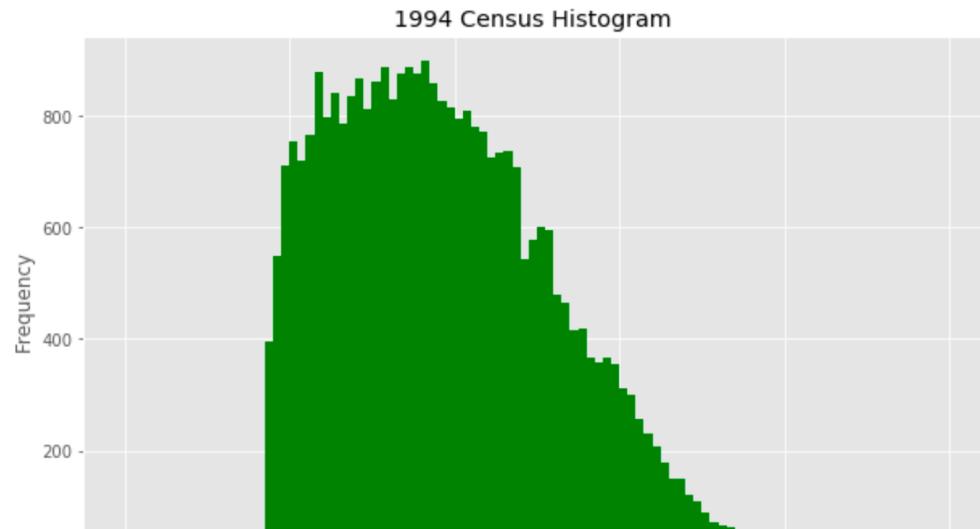
NOT Val, Histogram  
But Cont for  
y Bin



1D graph → We are  
only seeing 1  
variable

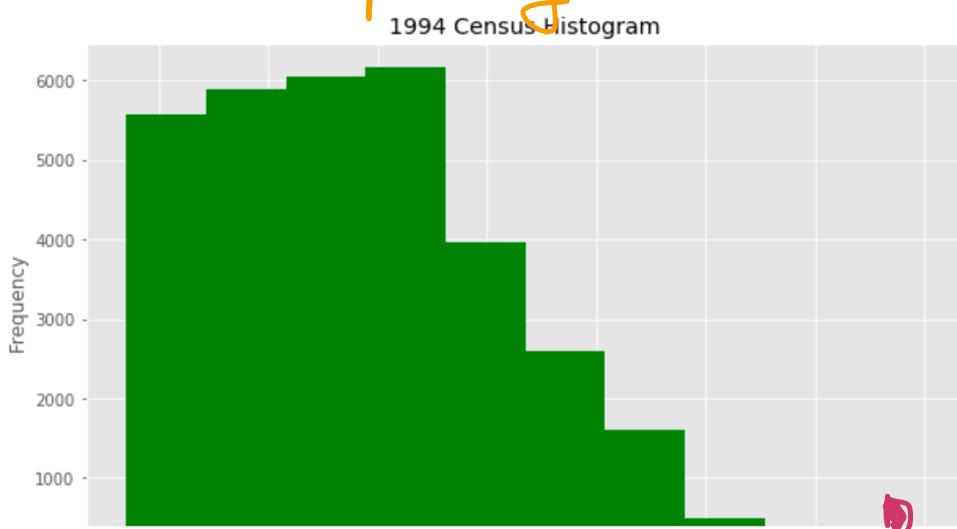
$x \rightarrow \text{Bin}$

## Different bins tell different stories...



More Bins

- Note: We can use histogram to detect (potential) outliers



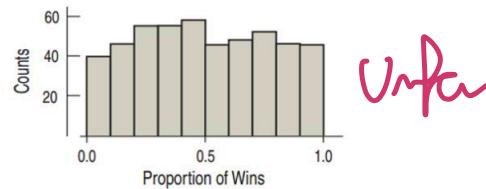
Less Bins

If we see this

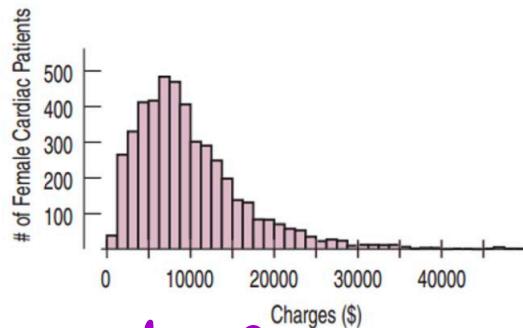
# What do we look for in a histogram?

- Distribution the 1st thing we look at... The Slope  
(Uniform, Skewed Right / Skewed Left)

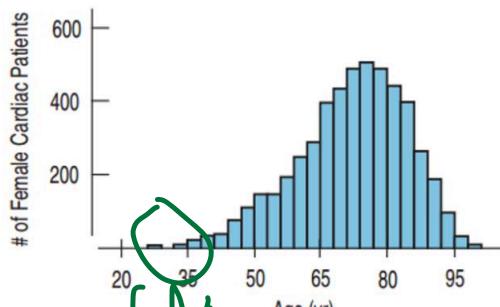
Where the Tail is



Unif



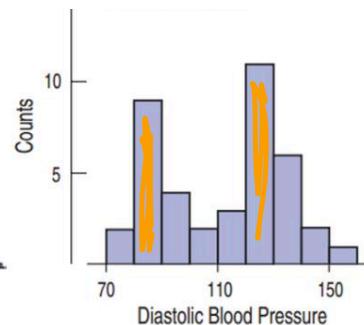
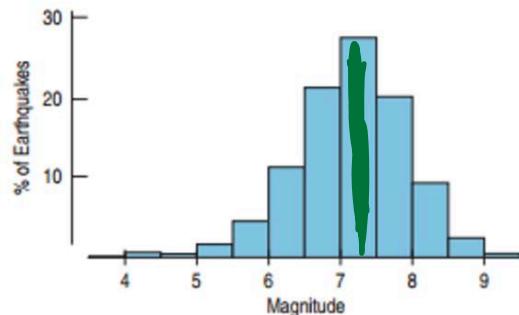
Skew Rgt  
(long tail on Rgt.)



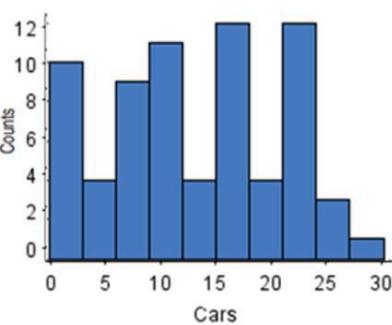
Left Skewed

## What do we look for in a histogram, cont.?

- Mode: A “hump” or a high-frequency bin *not Prob Value*
  - unimodal, bimodal, multimodal
  - approx. by  $mean-mode = 3 \times (mean-median)$  *With & the no Freqs*



*Bi-modal*



*No MOD*



The **mode** is a statistical concept used to identify the most frequently occurring value(s) in a dataset. Let's break down the slide:

## What is the Mode?

- The mode represents the “**hump**” or **high-frequency bin** in a histogram.
- It is the value (or range of values in a histogram bin) that appears the most in the dataset.

## Types of Modes in a Dataset

### 1. Unimodal:

- The dataset has **one hump** or one mode.
- Example: The first histogram on the slide shows one peak around a magnitude of 7.

### 2. Bimodal:

- The dataset has **two distinct humps**, indicating two dominant modes.
- Example: The second histogram (diastolic blood pressure) has peaks around 110 and 150.

### 3. Multimodal:

The dataset has **more than two humps**, indicating multiple frequent values.



...



### 3. Multimodal:

- The dataset has **more than two humps**, indicating multiple frequent values.

### 4. No Mode:

- If all values occur with the same frequency or there's no clear peak, the dataset has **no mode**.
- Example: The third histogram (number of cars) shows no distinct peak.

## Why is Mode Important?

- Helps identify the **central tendency** of data.
- Useful for detecting patterns, clusters, or subgroups in data.

## Approximation Formula

The mode can be approximated using the formula:

$$\text{Mean} - \text{Mode} \approx 3 \times (\text{Mean} - \text{Median})$$

This works for skewed distributions:

- If the data is **right-skewed**, the mode is less than the mean.
- If the data is **left-skewed**, the mode is greater than the mean.



Message





## Approximation Formula

The mode can be approximated using the formula:

$$\text{Mean} - \text{Mode} \approx 3 \times (\text{Mean} - \text{Median})$$

This works for skewed distributions:

- If the data is **right-skewed**, the mode is less than the mean.
- If the data is **left-skewed**, the mode is greater than the mean.

## How to Interpret Histograms?

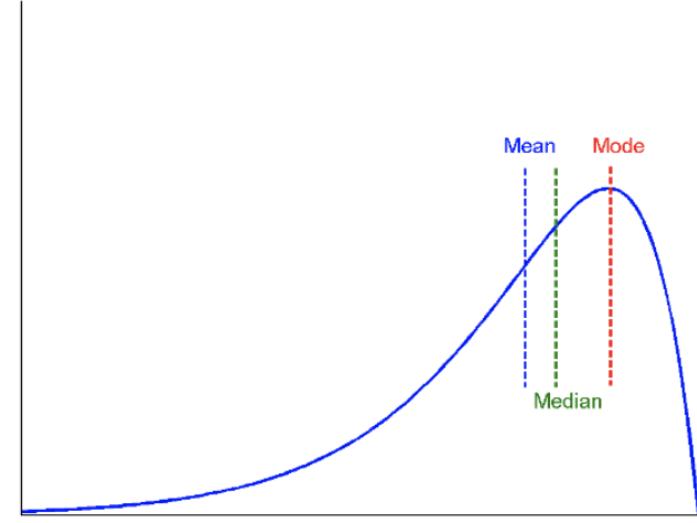
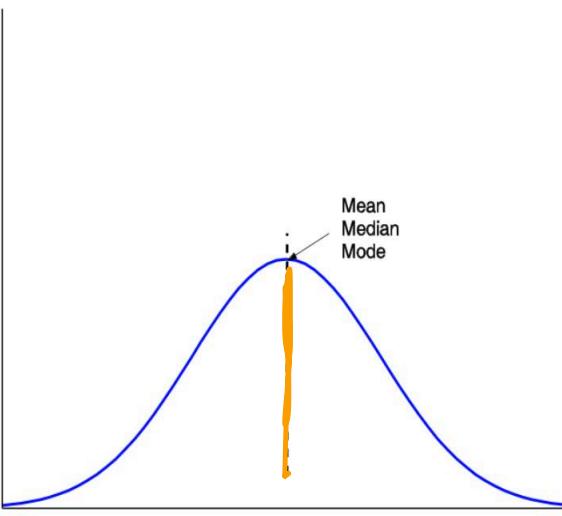
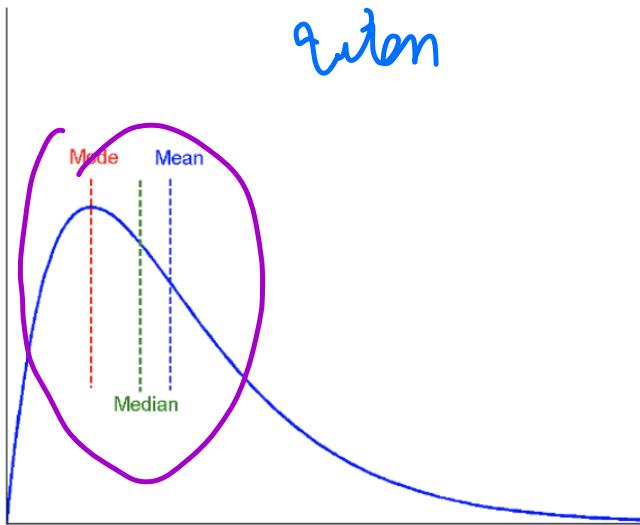
1. Look for **humps or peaks**.
  - One hump → Unimodal.
  - Two humps → Bimodal.
  - More → Multimodal.
2. If there's no clear peak, the data may lack a meaningful mode.

Would you like an example of calculating the mode or interpreting a specific histogram? Let me know!

Message



# Intuition of mean, median, mode



↓  
The relationship might not be present for multimodal or bimodal histograms.  
Mean is susceptible to extreme values.

# Range, Variance, Standard Deviation

Range = Maximum - Minimum

might be affected to outliers

Mean,  $\mu = \frac{1}{n} \sum_{i=1}^n x_i$

Variance =  $\sigma^2 = \left( \frac{1}{n} \sum_{i=1}^n (x_i - \mu)^2 \right)$

Avg of the sum of All data Pair - Mean Squared

Std. dev. =  $\sigma$

That's why

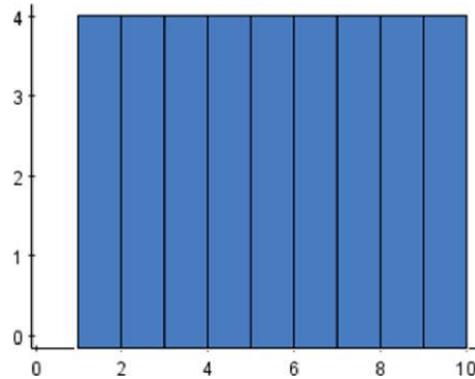
Unit of the Std. Dev.

Simply the  $\sqrt$  of the Variance



# Histograms and standard deviation

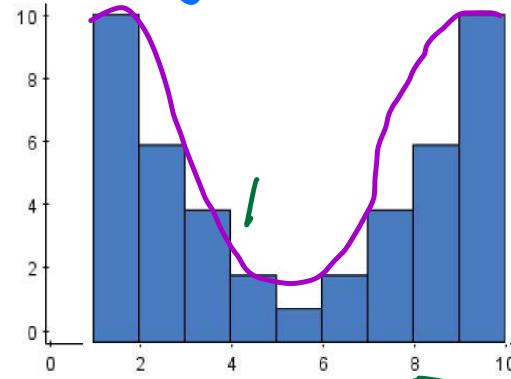
Order the histograms below from smallest standard deviation to largest standard deviation.



Non **A**  
Ordered 5

~~Small Std to large~~

**B, C, A**



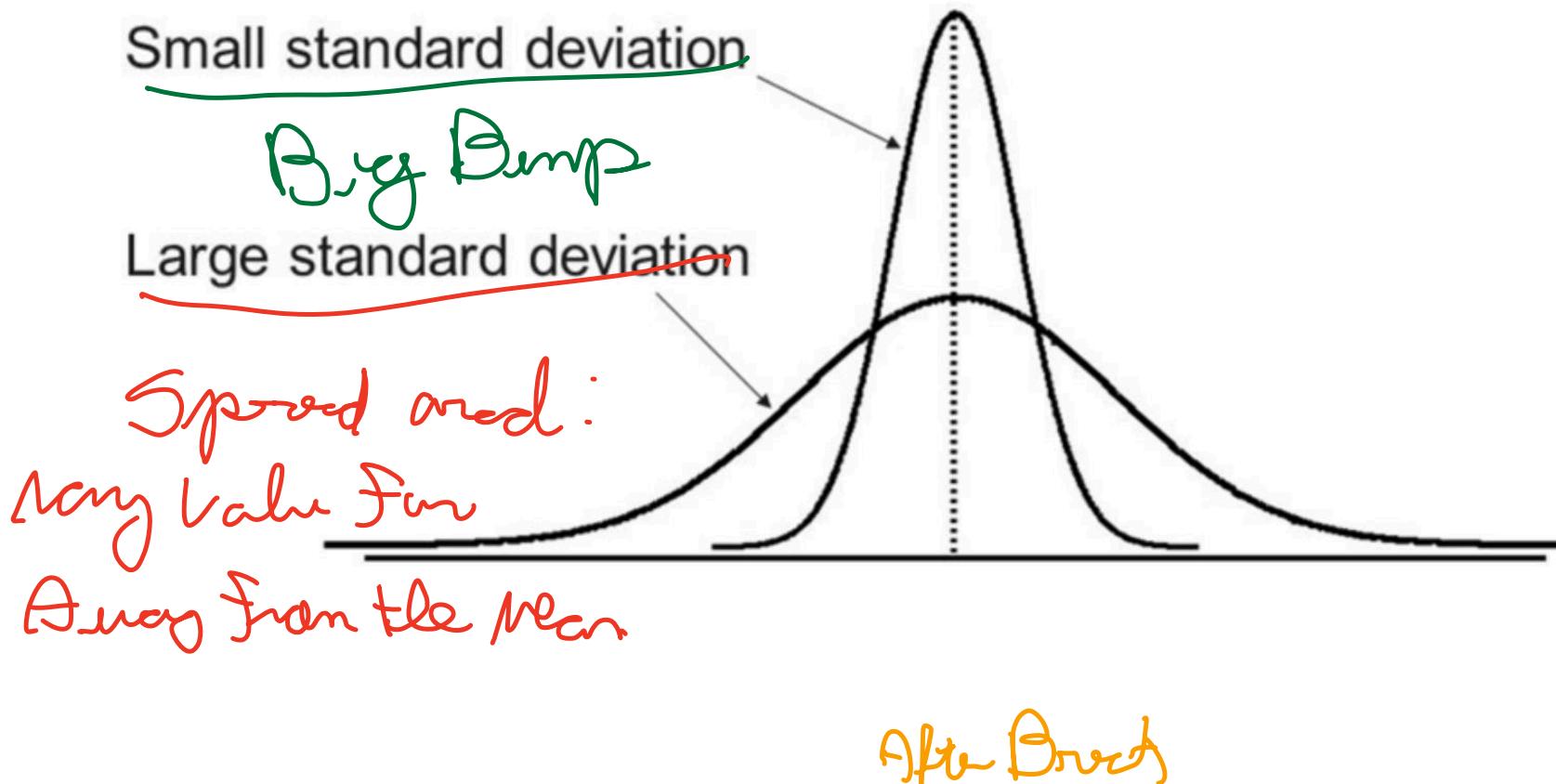
Near **5**,  
with a **gap**

**C**

of Valv far  
from the mean

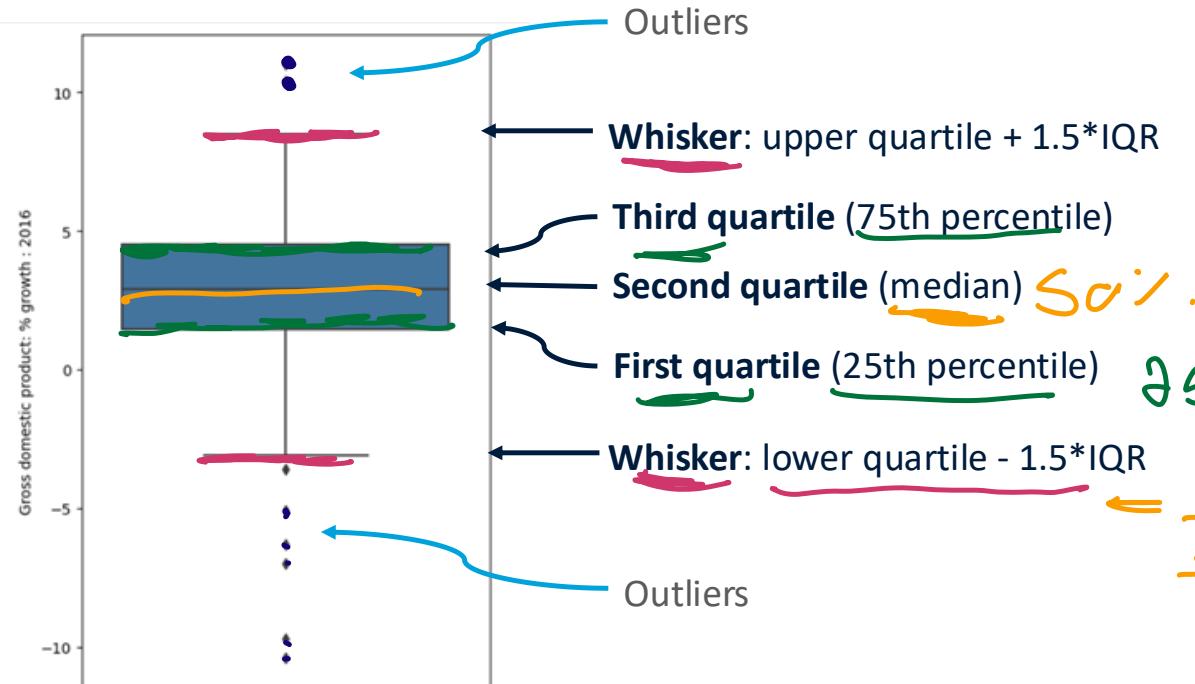
- A. A, B, C
- B. B, A, C
- C. C, A, B
- D. A, C, B
- E. B, C, A

# Histograms and standard deviation



## Boxplot

We have more data  
Popular for slow learning



Possible Outlier

`sns.boxplot(data=wb, y="Gross domestic product: % growth : 2016")`

25% of the values are lower than the others

$$IQR = Q_3 - Q_1$$

Are these really outliers? [\[link\]](#)

## Nominal and ordinal data types

Categorical

- Don't try to compute any numerical summary DO NOT
- Only (relative) frequency tables

| Class  | Count |
|--------|-------|
| First  | 325   |
| Second | 285   |
| Third  | 706   |
| Crew   | 885   |

| Class  | %     |
|--------|-------|
| First  | 14.77 |
| Second | 12.95 |
| Third  | 32.08 |
| Crew   | 40.21 |

Cont

In [20]:

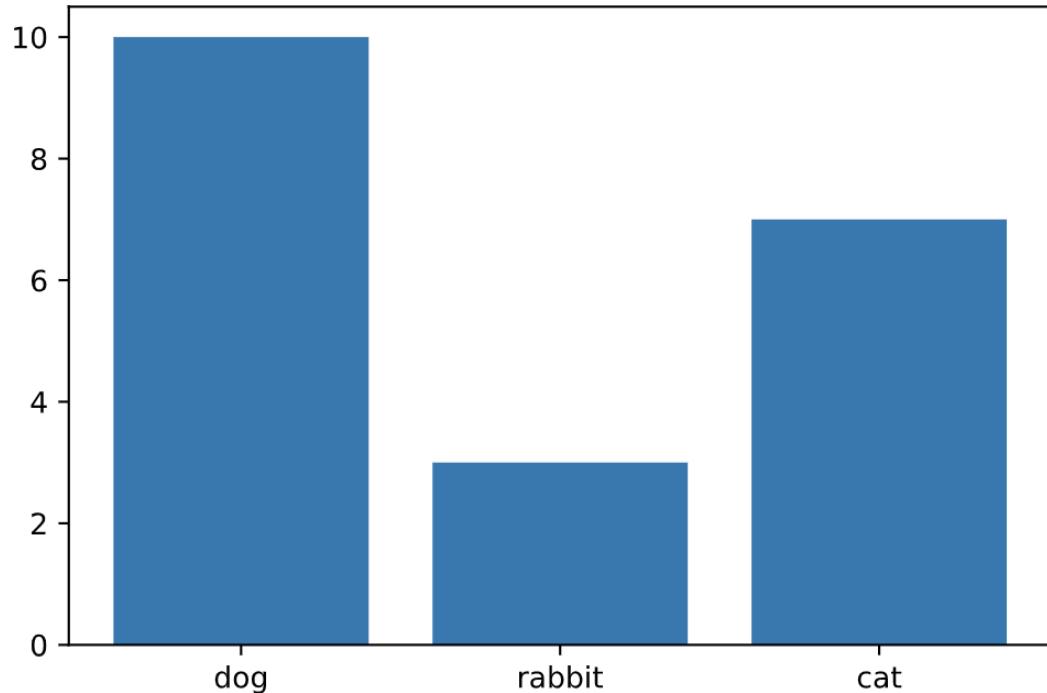
```
print(race.value_counts()) #frequency table  
print(race.value_counts()/len(race)) #relative frequency table
```

|                    |       |
|--------------------|-------|
| White              | 27816 |
| Black              | 3124  |
| Asian-Pac-Islander | 1039  |
| Amer-Indian-Eskimo | 311   |
| Other              | 271   |

As far as I can get with Numpy

# Bar chart

|          | Data |
|----------|------|
| Nominal  | ✓    |
| Ordinal  | ✓    |
| Interval | ✗    |
| Ratio    | ✗    |

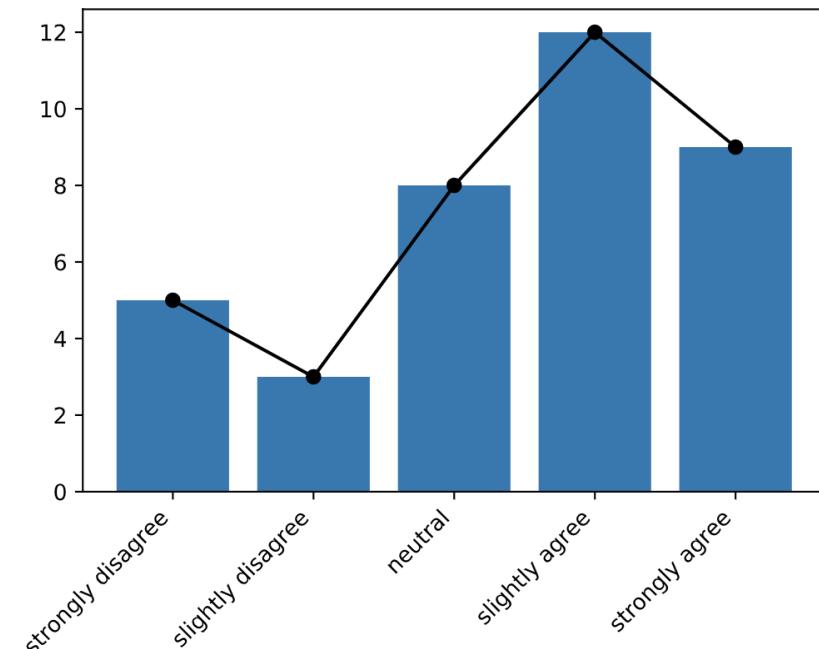
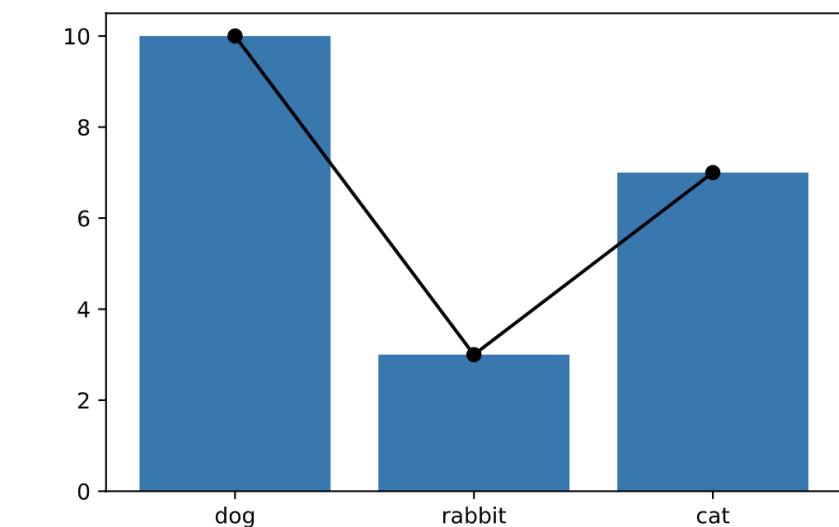


```
animals = wb["Animals"].value_counts()  
plt.bar(animals.index, animals.values);
```

```
wb["Animals"].value_counts().plot(kind='bar')
```

## Bar chart (bad)

Don't use lines within a bar chart for categorial or ordinal features!



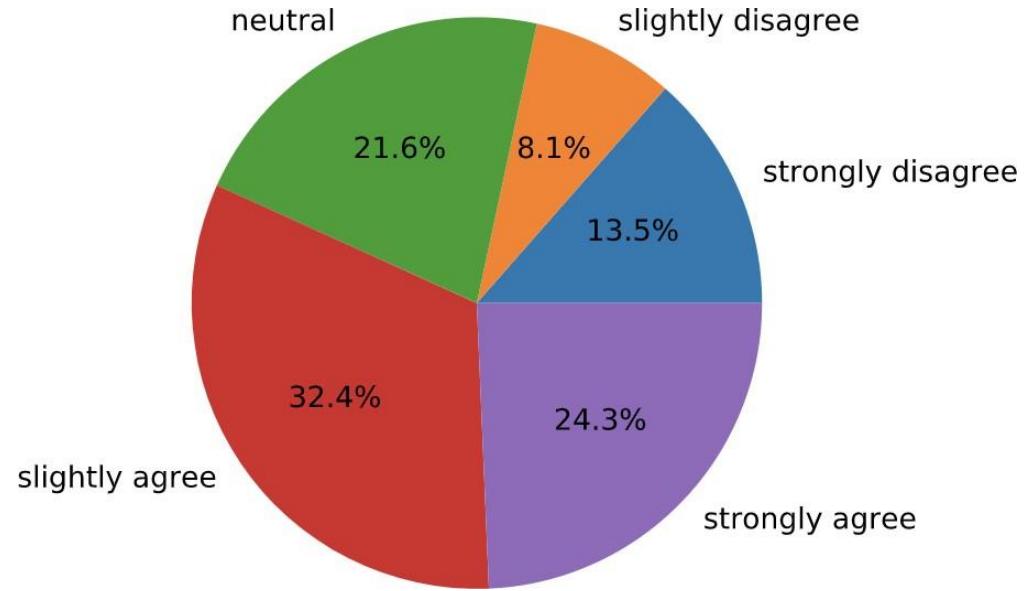
Even if there is Ordinality

These imply that this is a Corach

## Pie chart (mostly bad)

|          | Data |
|----------|------|
| Nominal  | X    |
| Ordinal  | X    |
| Interval | X    |
| Ratio    | X    |

{



Try to Avoid

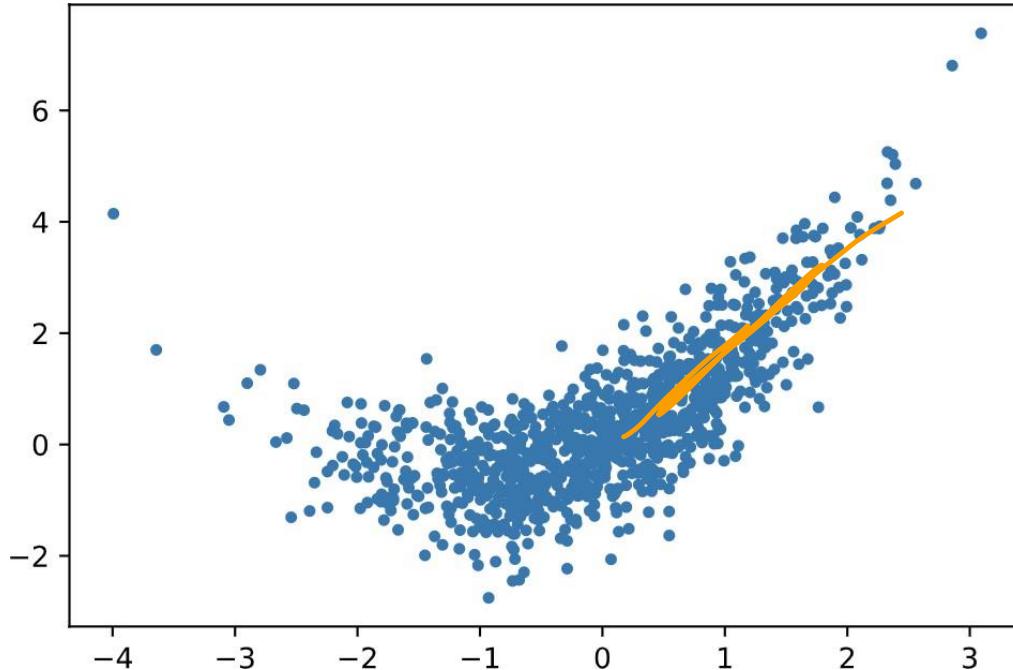
What about 2 Variable

## Scatterplot

|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | x     | x     |
| Ordinal  | x     | x     |
| Interval | ✓     | ✓     |
| Ratio    | ✓     | ✓     |

Why not ordinal data in first dimension?

general trend



```
plt.scatter(x_values, y_values)
```

```
sns.scatterplot(data=df, x="x_column", \n                 y="y_column", hue="hue_column")
```

When we group up,  
the 0th led to  
group 80

## Correlation

Put a # to the Answer

They change together

- The coefficient of two variables  $x$  and  $y$  is given by

$$\text{corr}(x, y) = \frac{\text{cov}(x, y)}{\sigma_x \sigma_y},$$

$$\text{cov}(x, y) = \frac{1}{n} \sum_{i=1}^n (x_i - \mu_x)(y_i - \mu_y),$$

- Assumptions, conditions, properties

- Correlation assumes a linear relationship between  $x$  and  $y$
- Outliers can strongly affect the value of correlation
- $\text{corr} > 0 \rightarrow$  positive association,  $\text{corr} < 0 \rightarrow$  negative association
- $-1 \leq \text{corr} \leq 1$
- Interchanging  $x$  and  $y$  does not change the correlation
- Corr has no units
- Changing the units of  $x$  or  $y$  does not affect corr

Pearson

Same Change Answer

Introducing

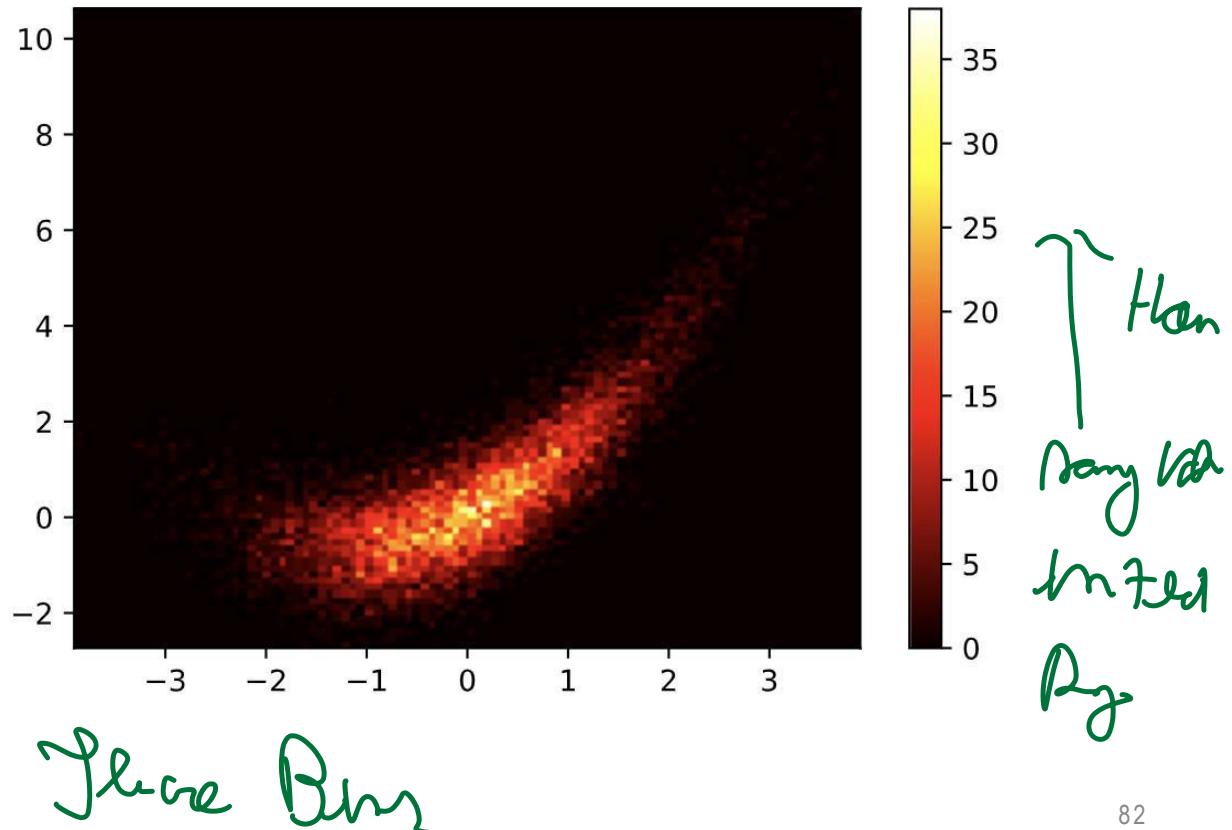
<http://guessthecorrelation.com/>

Be a part of the team

Jiving, trying

# Heatmap (density or 2D histogram)

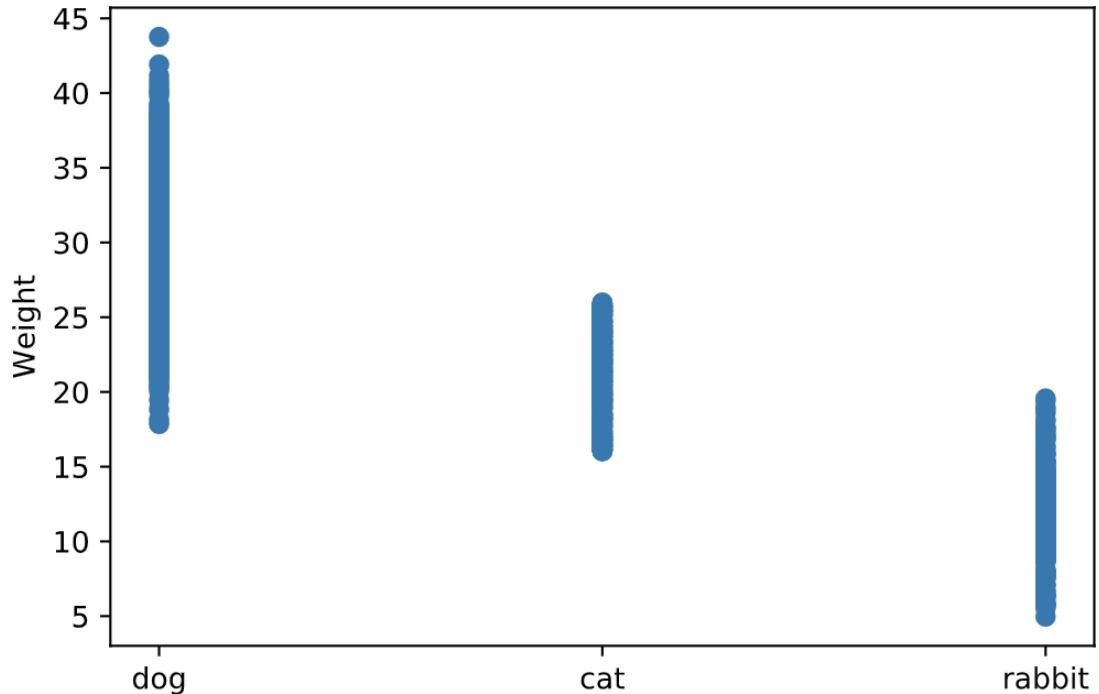
|          | Dim 1    | Dim 2    |
|----------|----------|----------|
| Nominal  | $\times$ | $\times$ |
| Ordinal  | $\times$ | $\times$ |
| Interval | ✓        | ✓        |
| Ratio    | ✓        | ✓        |



# Scatterplot (very very bad)



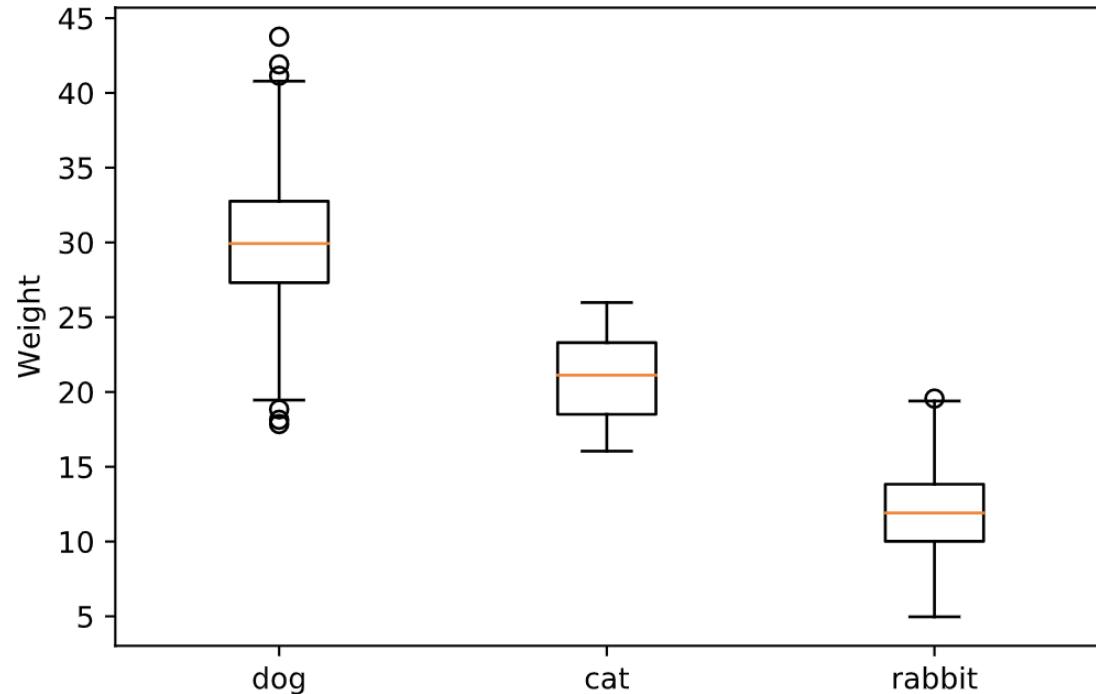
|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | x     | x     |
| Ordinal  | x     | x     |
| Interval | ✓     | ✓     |
| Ratio    | ✓     | ✓     |



# side-by-side boxplot

Küchen

|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | ✓     | ✗     |
| Ordinal  | ✓     | ✗     |
| Interval | ✗     | ✓     |
| Ratio    | ✗     | ✓     |



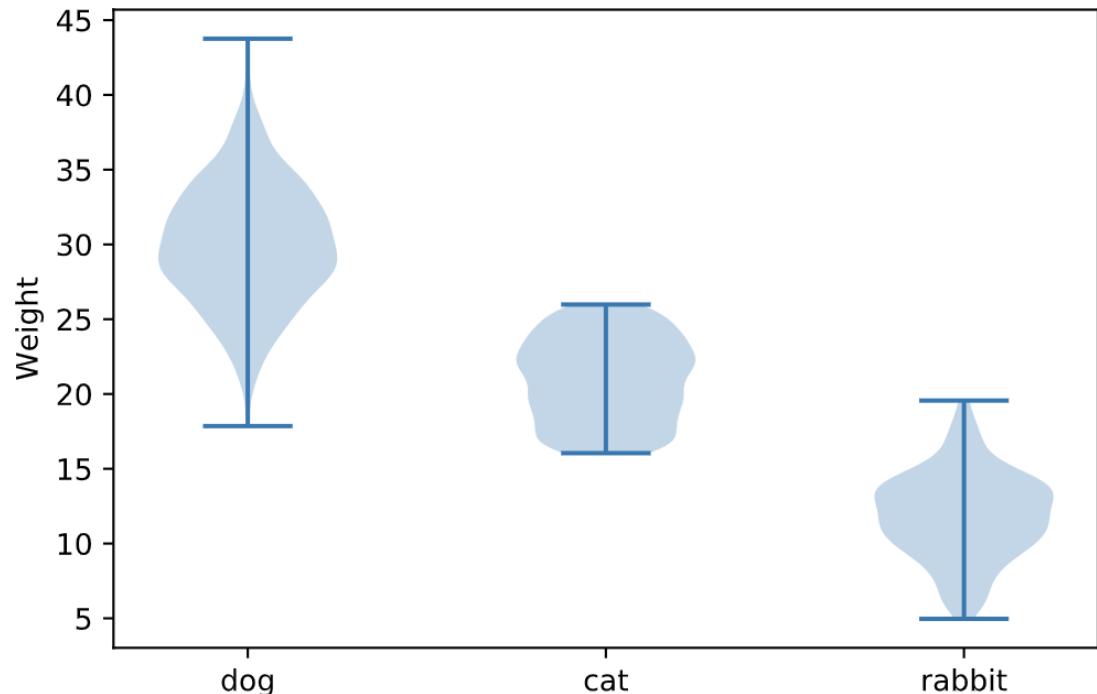
Performance of Models

```
sns.boxplot(data=wb, x="animals", y="weight");
```

## side-by-side violin plot

Also known as the  
Violin Plot

|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | ✓     | ✗     |
| Ordinal  | ✓     | ✗     |
| Interval | ✗     | ✓     |
| Ratio    | ✗     | ✓     |



```
sns.violinplot(data=wb, x="animals", y="weight");
```

spike is temporally

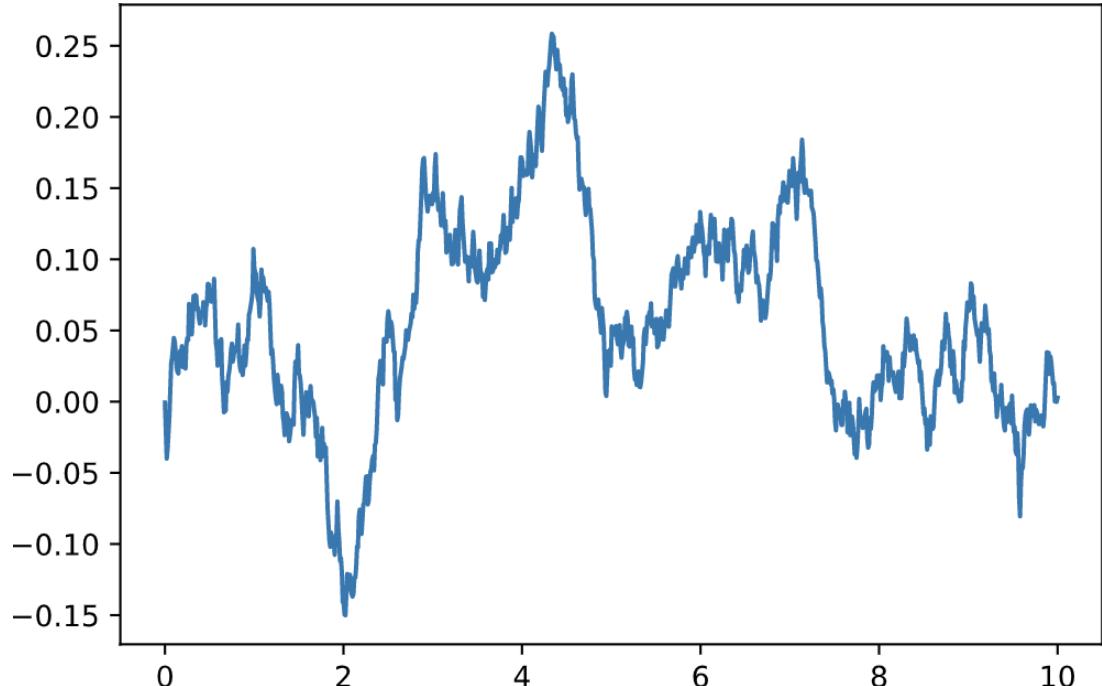
## Line plot

Plot over Time

When Dim 1 is temporal

|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | x     | x     |
| Ordinal  | x     | x     |
| Interval | ✓     | ✓     |
| Ratio    | ✓     | ✓     |

Why not ordinal data in first dimension?

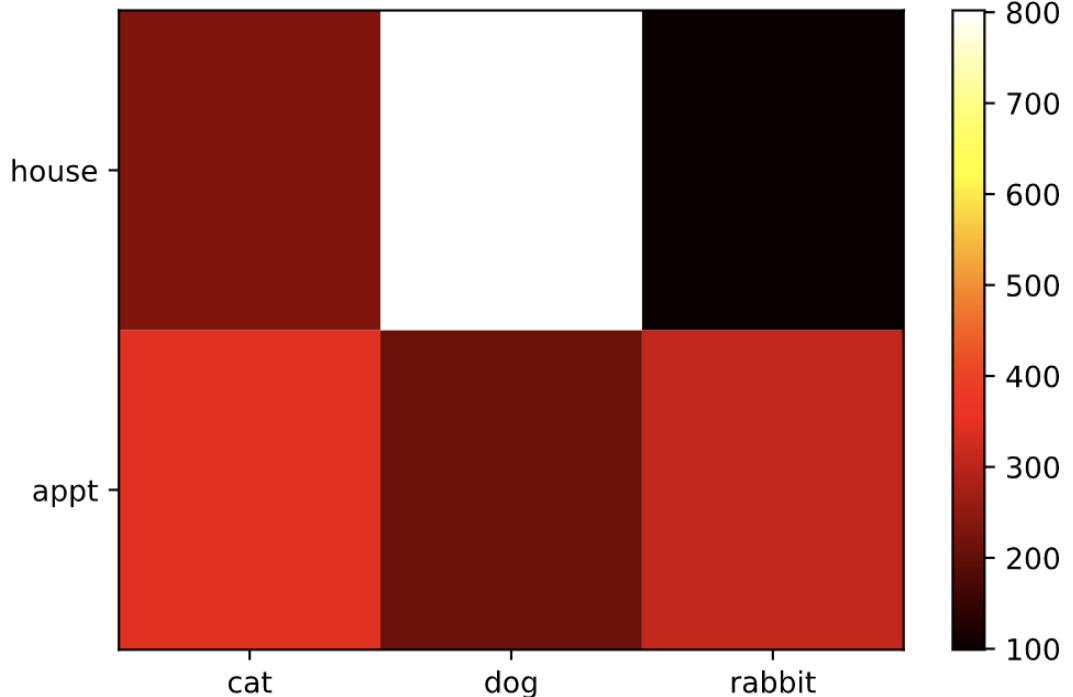


Time

# Heatmap (matrix)

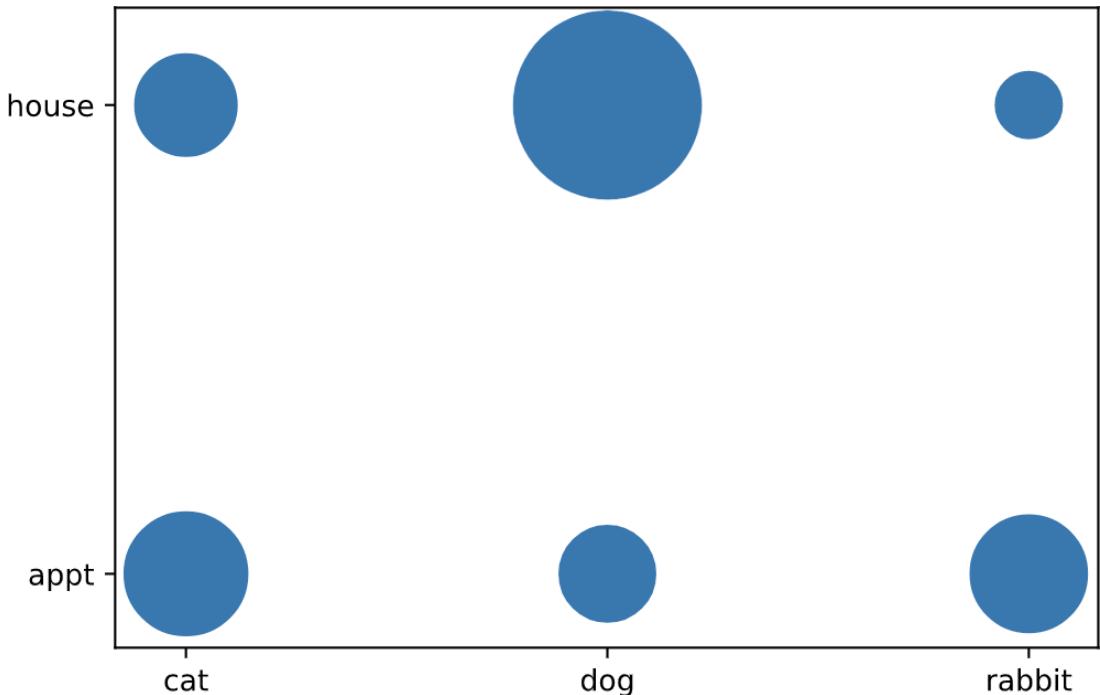
2 Categorical

|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | ✓     | ✓     |
| Ordinal  | ✓     | ✓     |
| Interval | ✗     | ✗     |
| Ratio    | ✗     | ✗     |



## Bubbleplot *Same Idea*

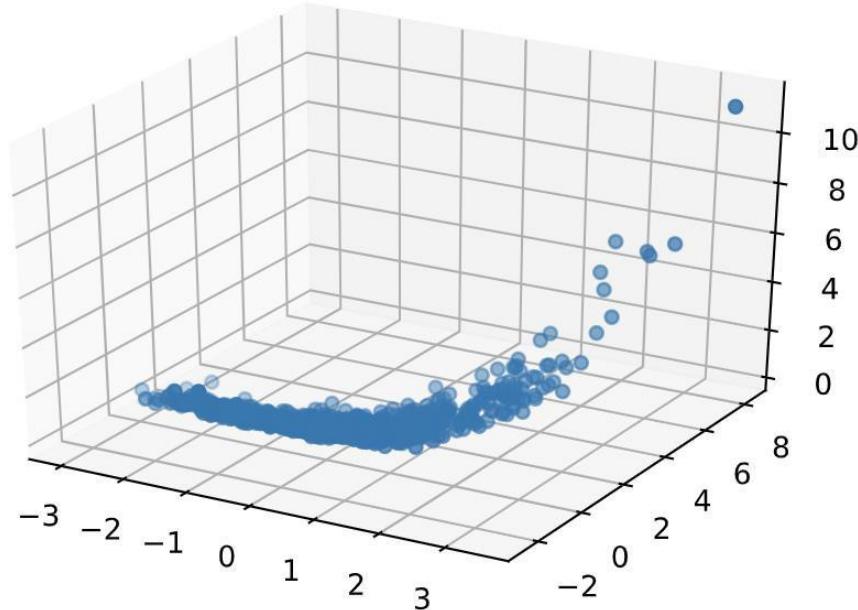
|          | Dim 1 | Dim 2 |
|----------|-------|-------|
| Nominal  | ✓     | ✓     |
| Ordinal  | ✓     | ✓     |
| Interval | ✗     | ✗     |
| Ratio    | ✗     | ✗     |



# 3D scatterplot



|          | Dim 1 | Dim 2 | Dim 3 |
|----------|-------|-------|-------|
| Nominal  | x     | x     | x     |
| Ordinal  | x     | x     | x     |
| Interval | x     | x     | x     |
| Ratio    | x     | x     | x     |



Avoid 3D!

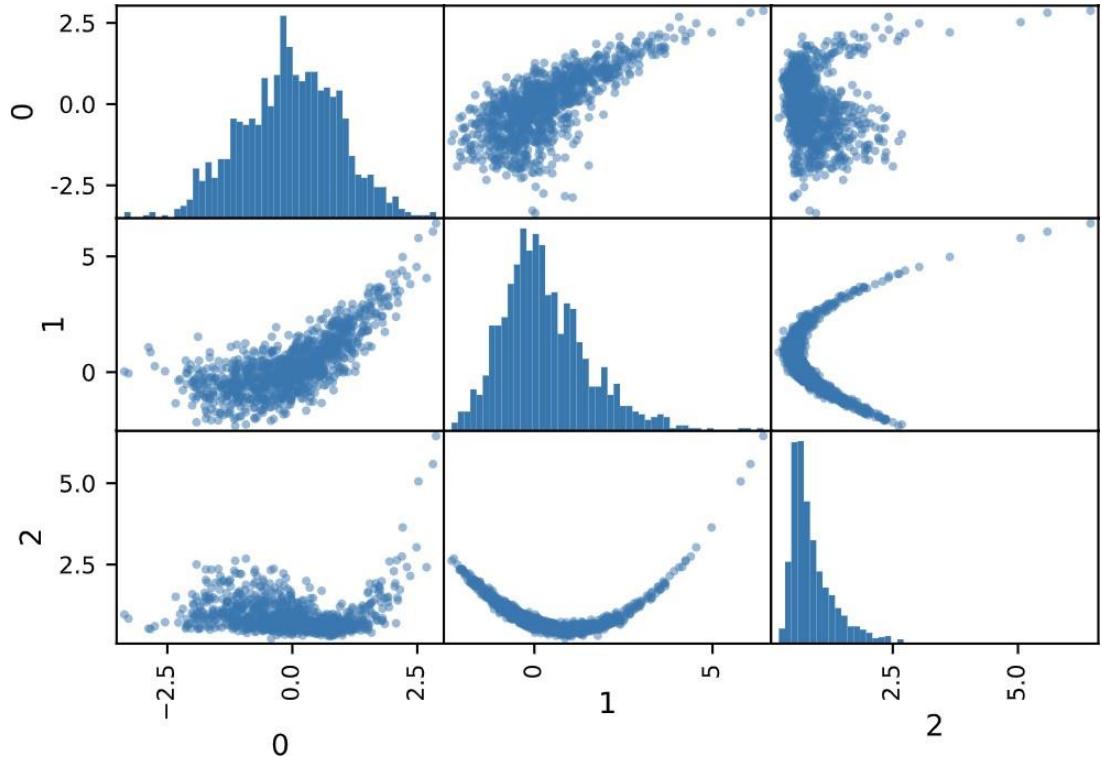


## Scatterplot matrix

Jm Pm

Diagonal + var all b/w them  
we see here

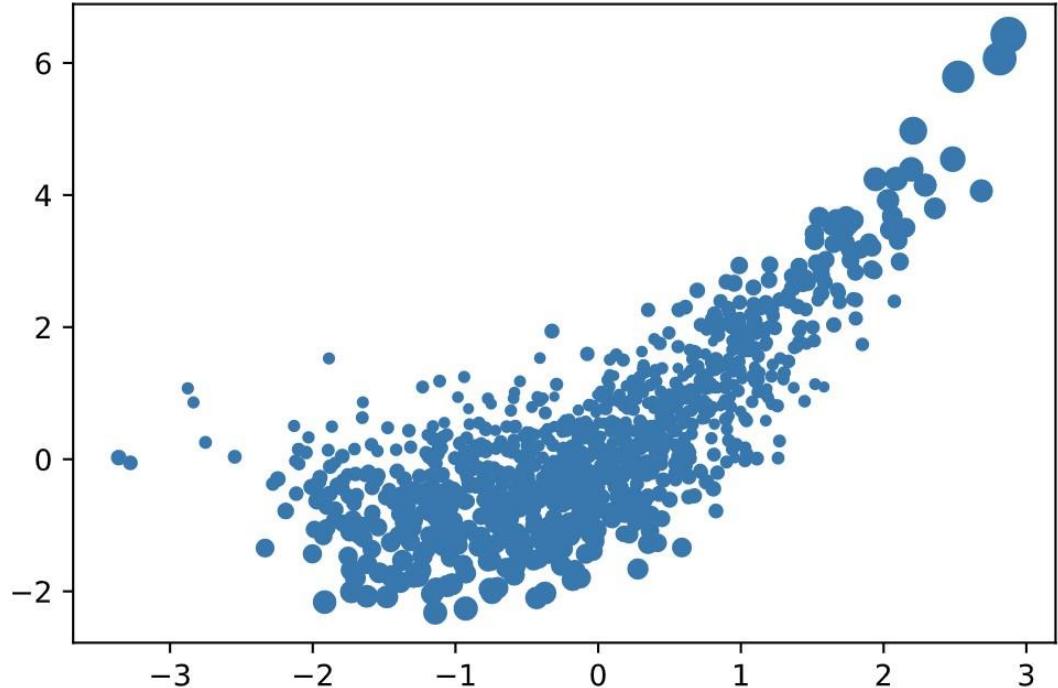
|          | Dim 1 | Dim 2 | Dim 3 |
|----------|-------|-------|-------|
| Nominal  | x     | x     | x     |
| Ordinal  | x     | x     | x     |
| Interval | ✓     | ✓     | ✓     |
| Ratio    | ✓     | ✓     | ✓     |



## Bubbleplot (“light” 3D)

floor by  $\leftarrow$  the Dots

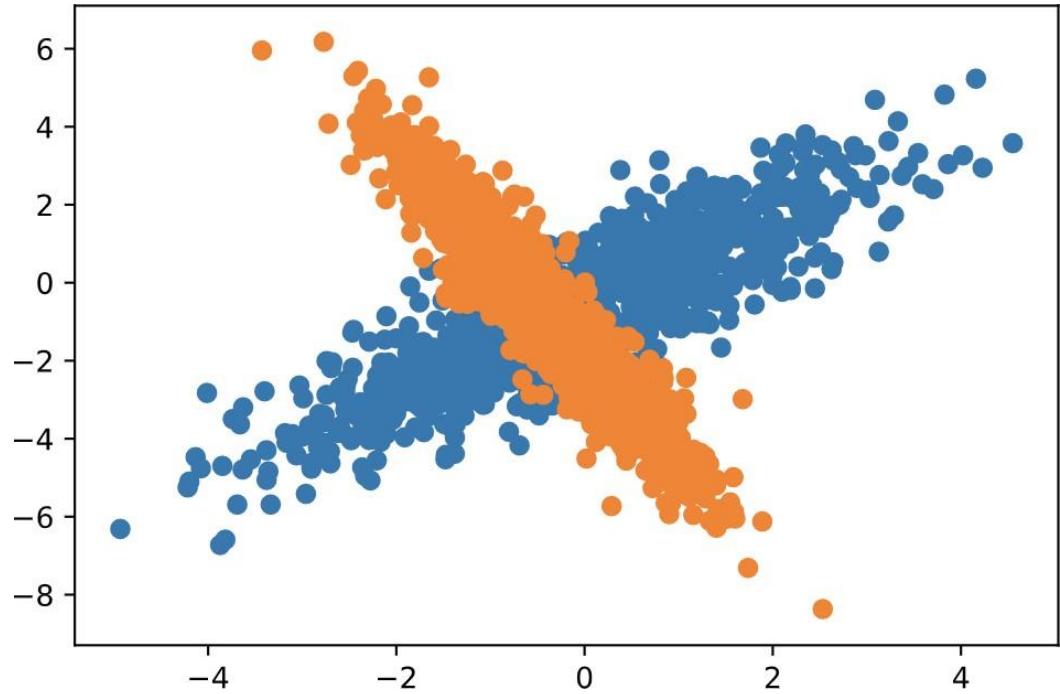
|          | Dim 1    | Dim 2    | Dim 3    |
|----------|----------|----------|----------|
| Nominal  | $\times$ | $\times$ | $\times$ |
| Ordinal  | $\times$ | $\times$ | $\times$ |
| Interval | ✓        | ✓        | ✓        |
| Ratio    | ✓        | ✓        | ✓        |



## Color scatterplot (“light” 3D)

|          | Dim 1 | Dim 2 | Dim 3 |
|----------|-------|-------|-------|
| Nominal  | x     | x     | ✓     |
| Ordinal  | x     | x     | ✓     |
| Interval | ✓     | ✓     | x     |
| Ratio    | ✓     | ✓     | x     |

Under?



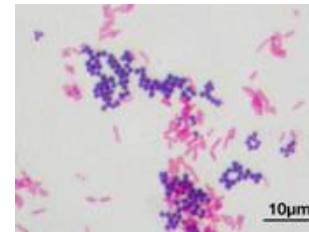
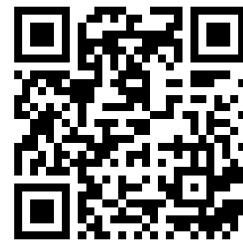
Color: Yes or No

## HARD DONTs

---

- Don't make histograms for nominal/ordinal data! *Categorical*
- Don't compute numerical summaries for nominal/ordinal data!
  - e.g. mean student i-number does not anything *No意义*
- Don't look for shape, spread etc. in a bar chart *when Plotly*
- Choose an appropriate bin for histograms *Catgo*
- Beware of outliers
- Not everything might be possible to plot/visualize

# What questions to ask?



Hantavirus

Genus, Species

Table 1: Burtin's data.

| Bacteria                               | Min.Inhibitory Concentration [ml/g] | Penicillin | Streptomycin | Neomycin | Gram Staining |
|--|-------------------------------------|------------|--------------|----------|---------------|
| <i>Aerobacter aerogenes</i>            |                                     | 870        | 1            | 1.6      | negative      |
| <i>Brucella abortus</i>                |                                     | 1          | 2            | 0.02     | negative      |
| <i>Brucella anthracis</i>              |                                     | 0.001      | 0.01         | 0.007    | positive      |
| <i>Diplococcus pneumoniae</i>          |                                     | 0.005      | 11           | 10       | positive      |
| <i>Escherichia coli</i>                |                                     | 100        | 0.4          | 0.1      | negative      |
| <i>Klebsiella pneumoniae</i>           |                                     | 850        | 1.2          | 1        | negative      |
| <i>Mycobacterium tuberculosis</i>      |                                     | 800        | 5            | 2        | negative      |
| <i>Proteus vulgaris</i>                |                                     | 3          | 0.1          | 0.1      | negative      |
| <i>Pseudomonas aeruginosa</i>          |                                     | 850        | 2            | 0.4      | negative      |
| <i>Salmonella (Eberthella) typhosa</i> |                                     | 1          | 0.4          | 0.008    | negative      |
| <i>Salmonella schottmuelleri</i>       |                                     | 10         | 0.8          | 0.09     | negative      |
| <i>Staphylococcus albus</i>            |                                     | 0.007      | 0.1          | 0.001    | positive      |
| <i>Staphylococcus aureus</i>           |                                     | 0.03       | 0.03         | 0.001    | positive      |
| <i>Streptococcus faecalis</i>          |                                     | 1          | 1            | 0.1      | positive      |
| <i>Streptococcus hemolyticus</i>       |                                     | 0.001      | 14           | 10       | positive      |
| <i>Streptococcus viridans</i>          |                                     | 0.005      | 10           | 40       | positive      |

Analysis of Burtin

# How effective are the drugs?

Half of the Bacteria  
is Treated

Gram  
Positive

Gram  
Negative

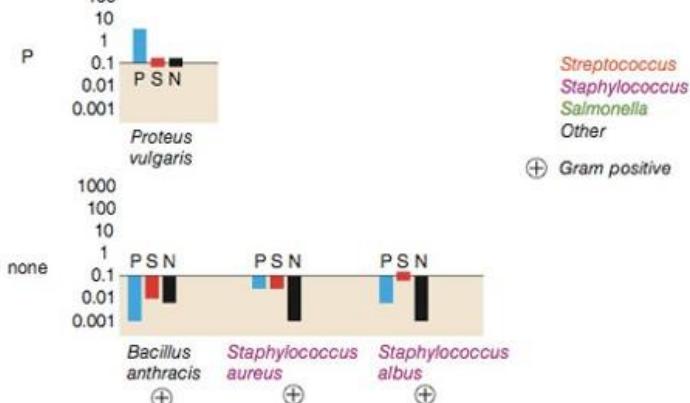
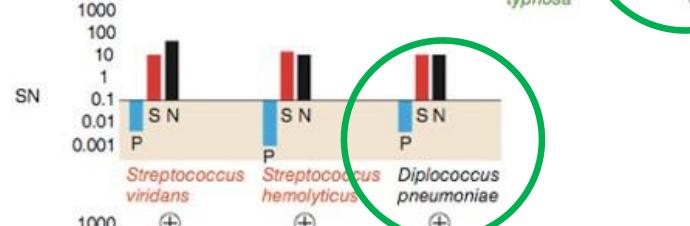
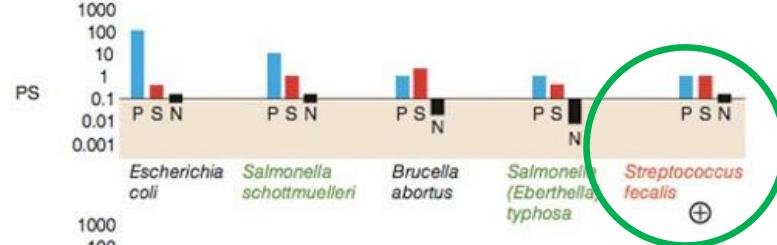
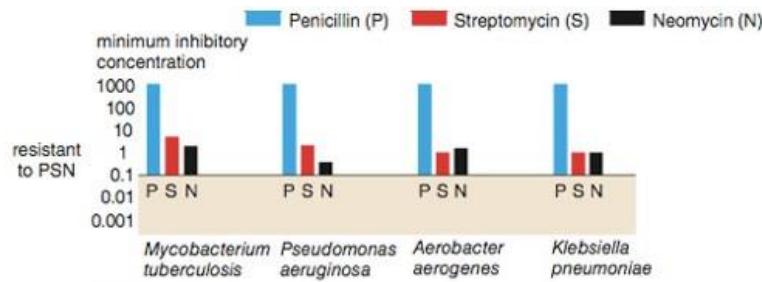
If bacteria is gram positive,  
Penicillin & Neomycin are  
most effective

If bacteria is gram negative,  
Neomycin is most effective

Half Bacteria

↑ Bar

↑ %



## How resistant are the bacteria?

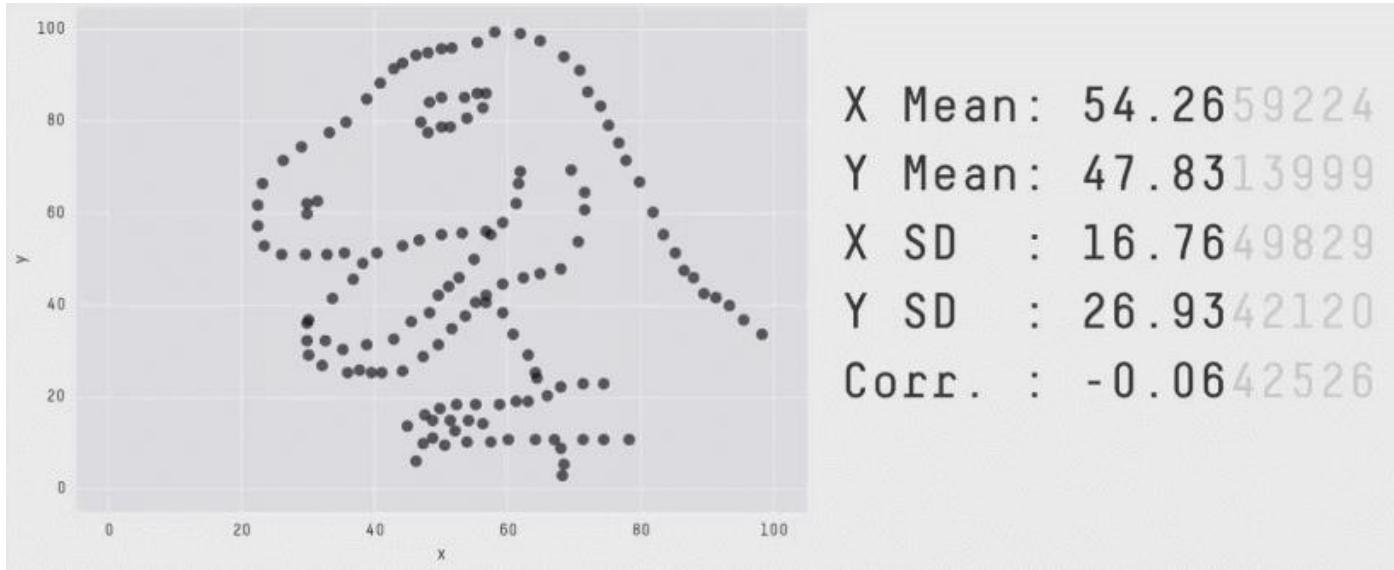
Not a streptococcus!  
(realized ~30 years later)

Really a streptococcus!  
(realized ~20 years later)

It can change its identity over time  
We need to do further analysis

# Why don't we just use numbers? Datasaurus

<https://www.autodesk.com/research/publications/same-stats-different-graphs>



Locut Sir

# Visualization motivation

Why

Visualizations help us to analyze and explore the data.

They help to:

- Identify hidden patterns and trends
- Formulate/test hypotheses
- Communicate any modeling results instead of just those numbers
  - Present information and ideas succinctly
  - Provide evidence and support
  - Influence and persuade
- Determine the next step in analysis/modeling

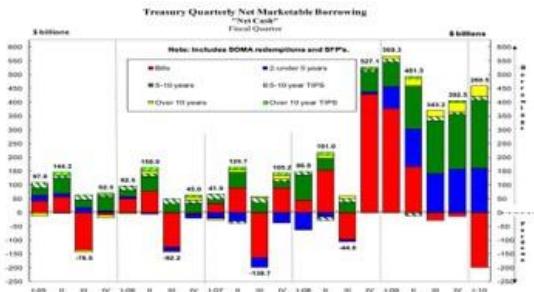
Get me created in the snow  
instead of just those numbers

Super Cool

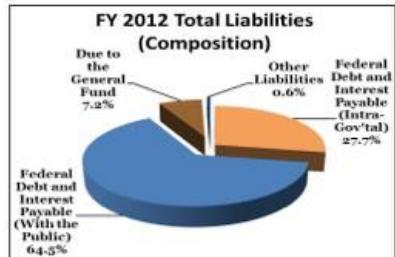
# But there are many bad examples...



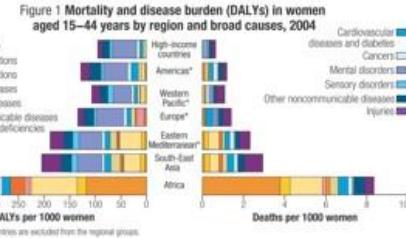
We doren  
wachten, gt  
gt kleiner



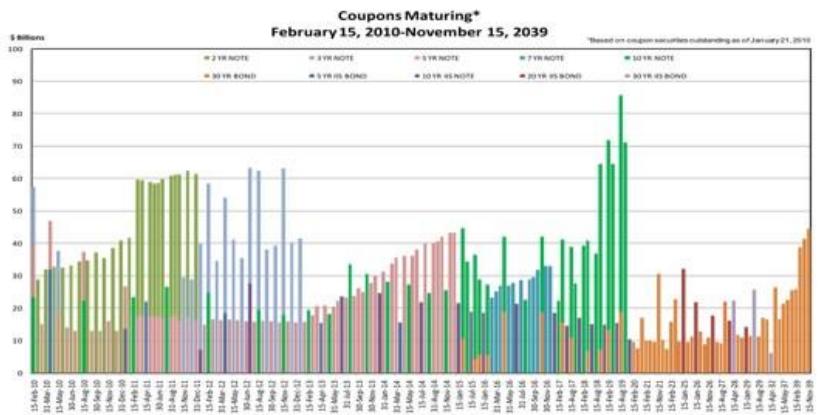
**Figure 10**



**Figure 10**



and are succeeded from the regional groups.  
With Organisation."



#### Cryptosporidium Prevalence

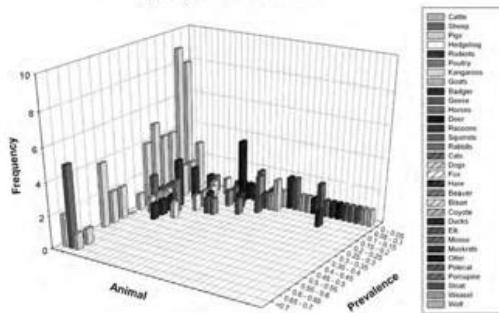
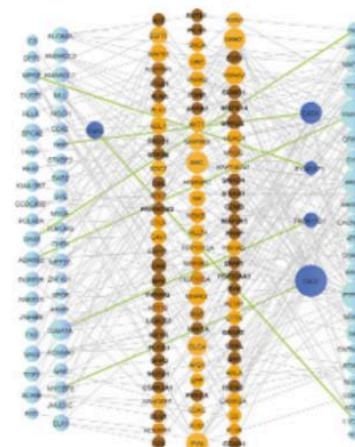
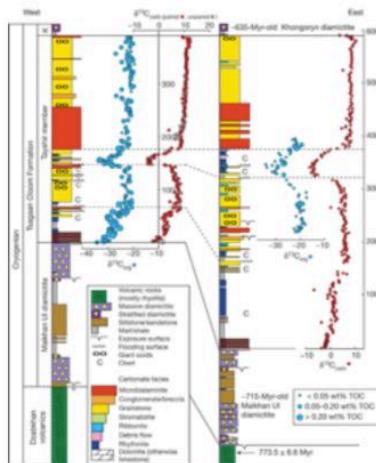
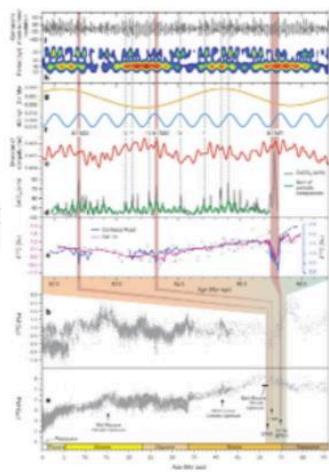
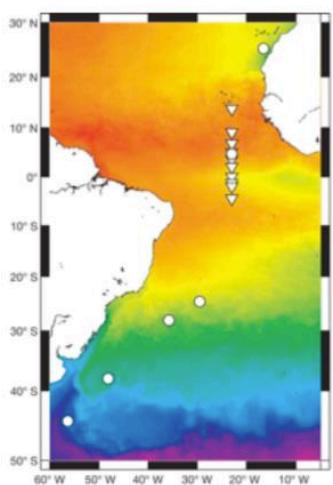
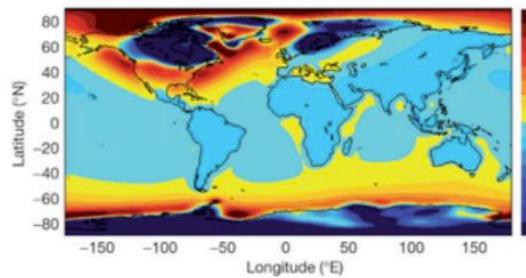
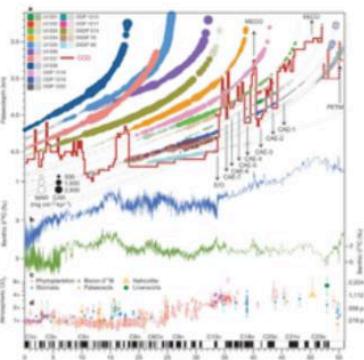
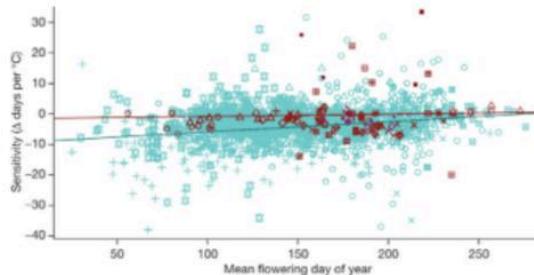
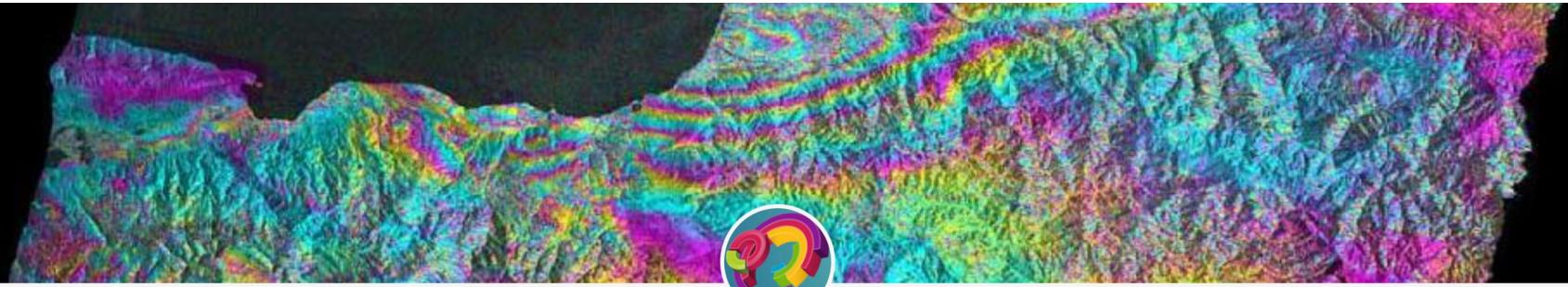


Figure 5.2 Mean prevalence rates of *Cryptosporidium* oocysts by animal species.

# Even more...



Sources: Nature



<http://viz.wtf>

# WTF Visualizations

Visualizations that make no sense.

For a discussion of what is wrong with a particular visualization, tweet at us [@WTFViz](#).  
[Submit a WTFViz you found](#).

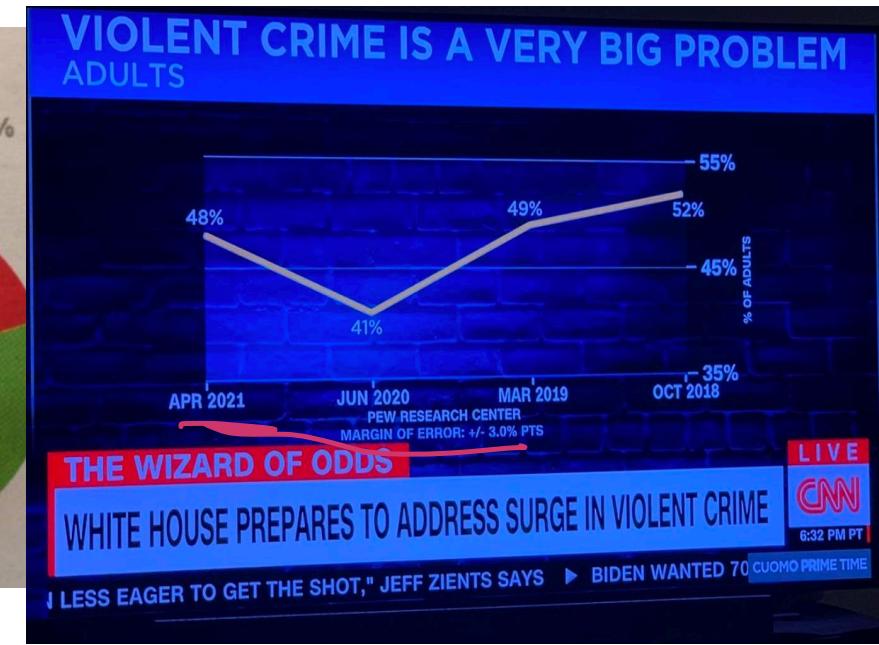
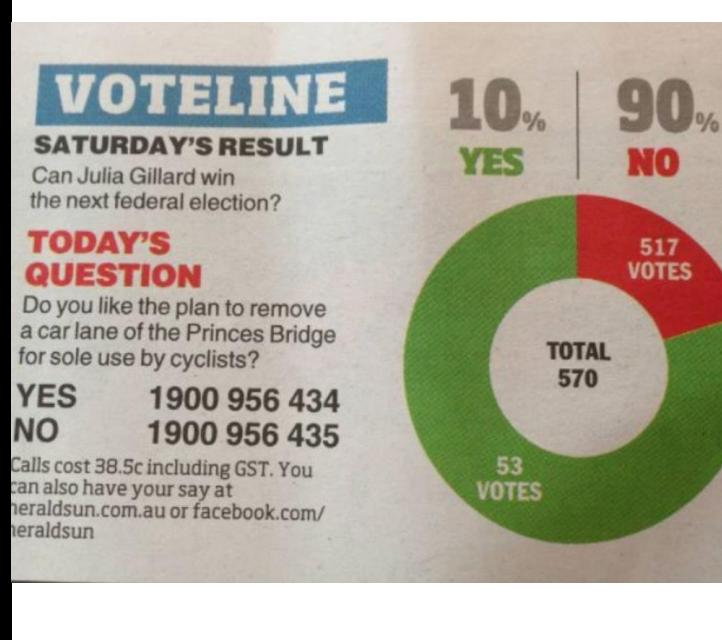
BetMGM 🐾 🐾 @BetMGM  
Mike Glennon has 1 win since 2014

Glennon's earned over \$5.3 mil per win in the NFL (6-24)

**MIKE GLENNON CAREER EARNINGS**

| Category      | Value        |
|---------------|--------------|
| 1st           | \$1,375,000  |
| 2nd           | \$1,280,132  |
| 3rd           | \$1,865,625  |
| 4th           | \$5,000,000  |
| Total         | \$18,500,000 |
| Current Total | \$3,946,782  |

503 2,770 12.6K



## Effective visualizations

Simple but effective

1. Have graphical integrity
2. Keep it simple...enough
3. Use sensible design
4. Use the right display

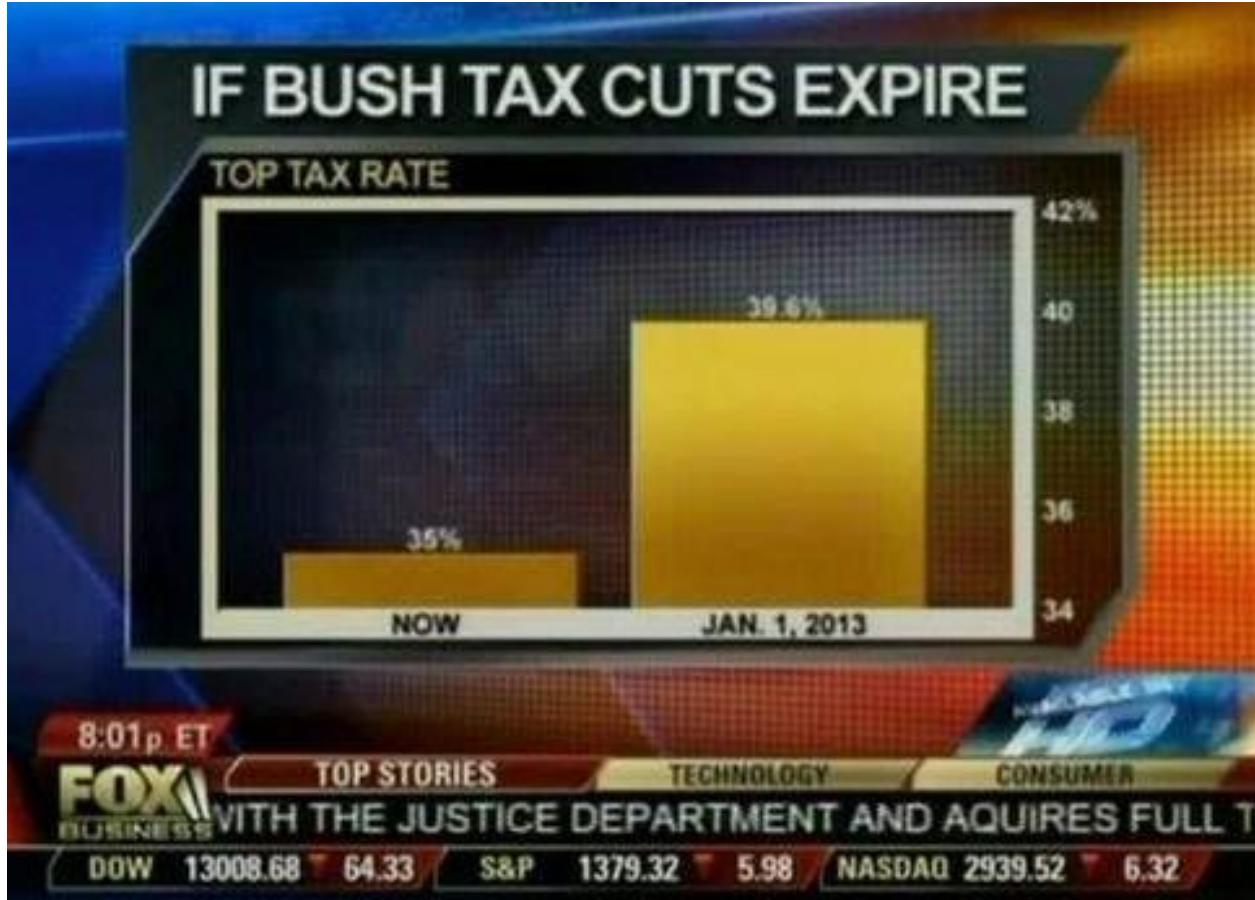
# 1. Graphical integrity

- Show data variation, **not** design variation
- Clear, detailed and thorough labeling and appropriate scales
- Plot all the data
- Be proportional

Should be Standard

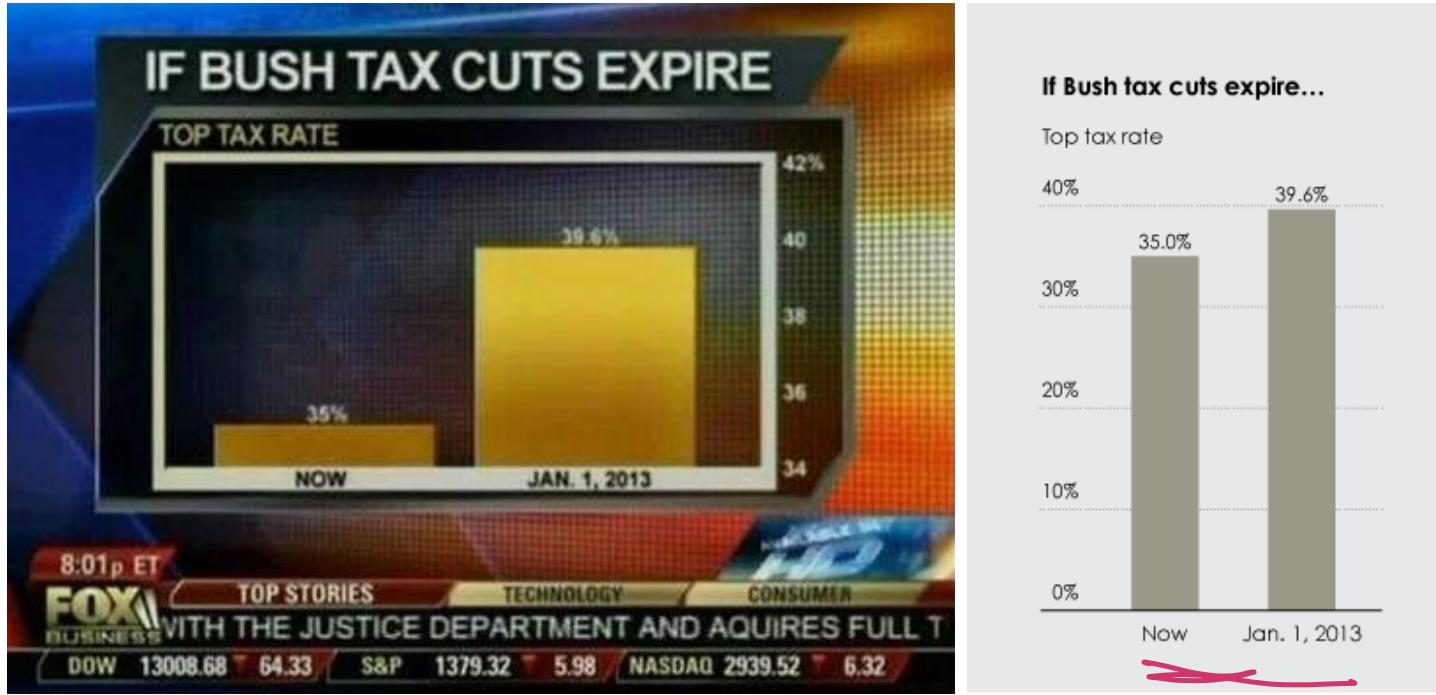
↓  
Take any ad  
Should still make  
Sense

# What's the problem here?



Should Start at 0

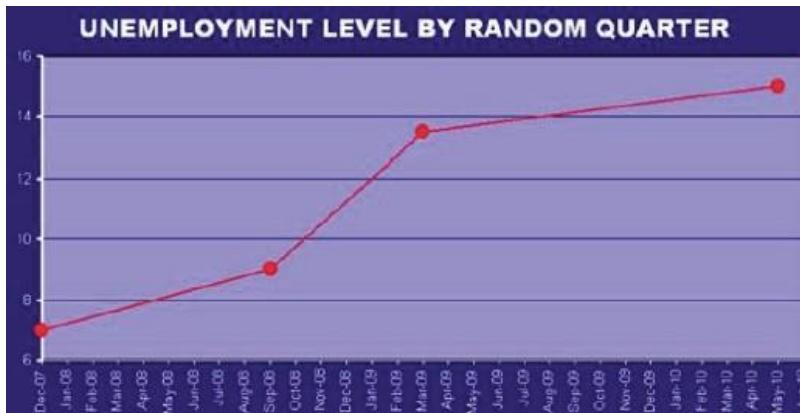
# Problem: Scale distortions



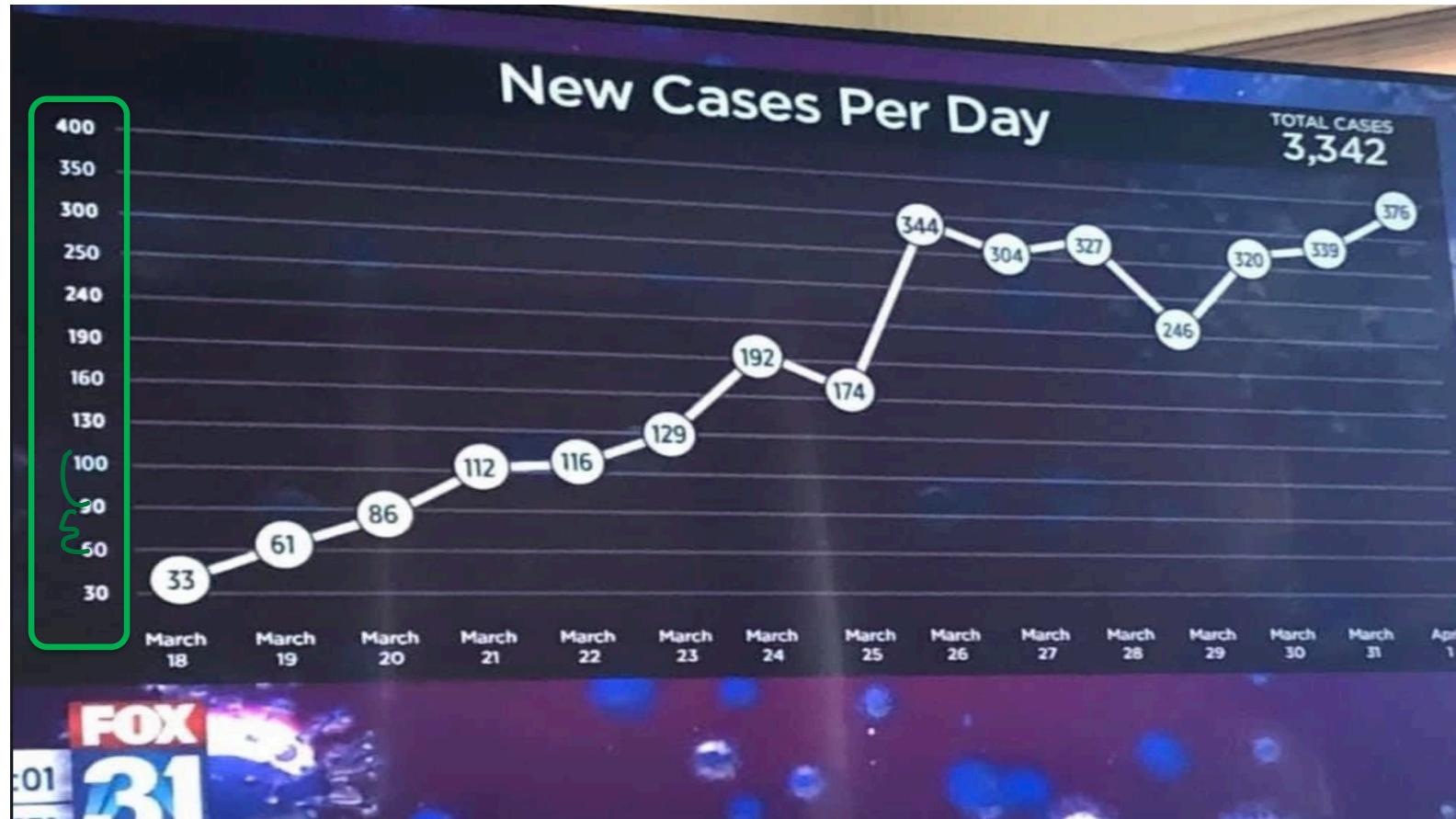
## Another example



# Problem: Scale distortions



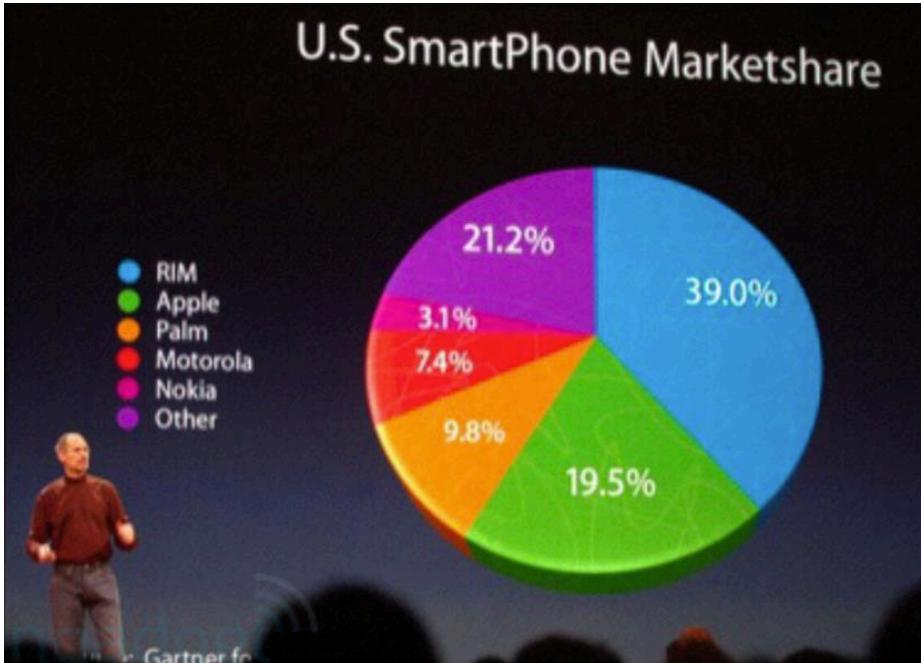
# COVID-19 gave us some really nice plots...



# What are the problems here?

- Violates the Area Principle

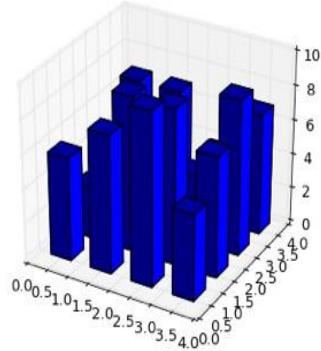
*It's tilted*



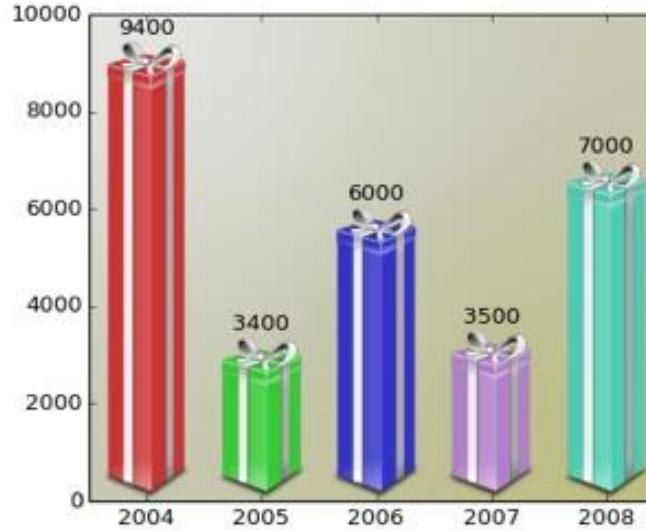
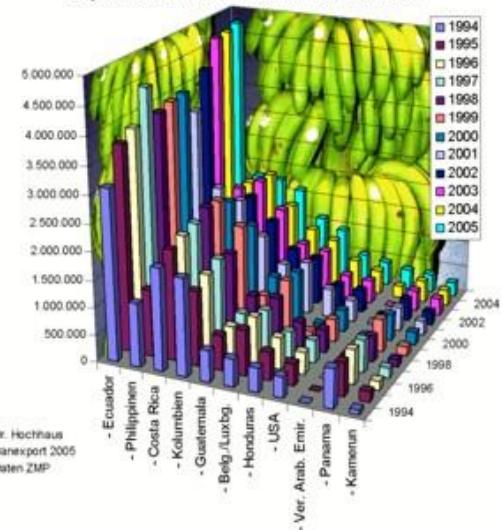
- Add up the percents!



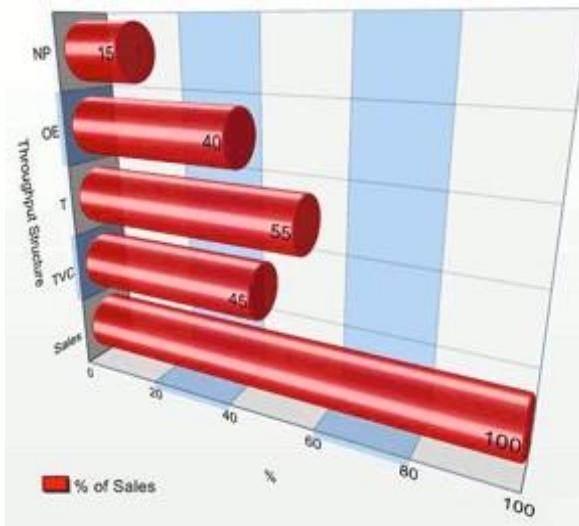
## 2. Keep it simple



Export von Bananen in Tonnen von 1994-2005



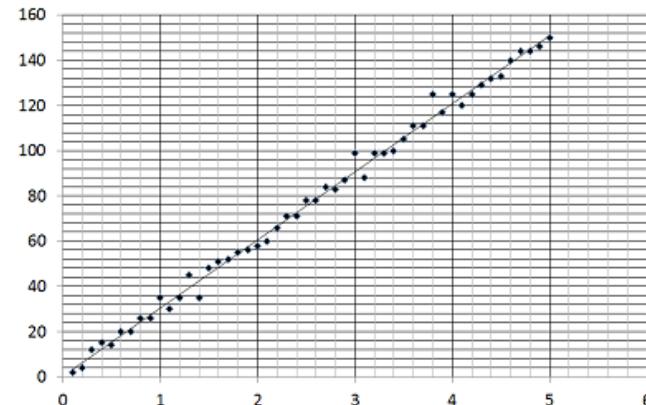
matplotlib  
gallery



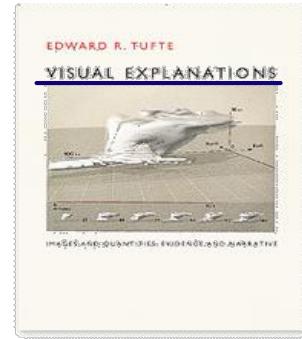
Excel Charts Blog

## 2. Keep it simple

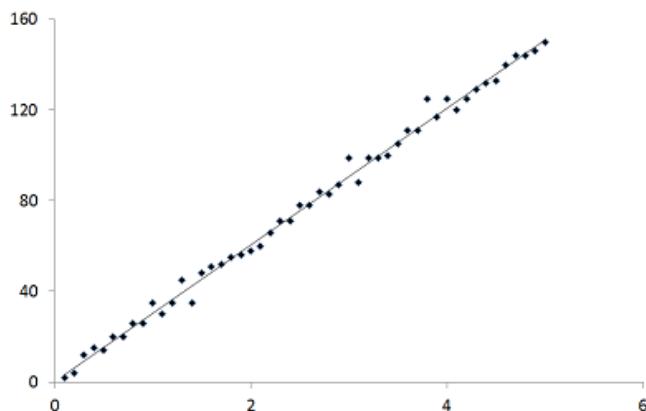
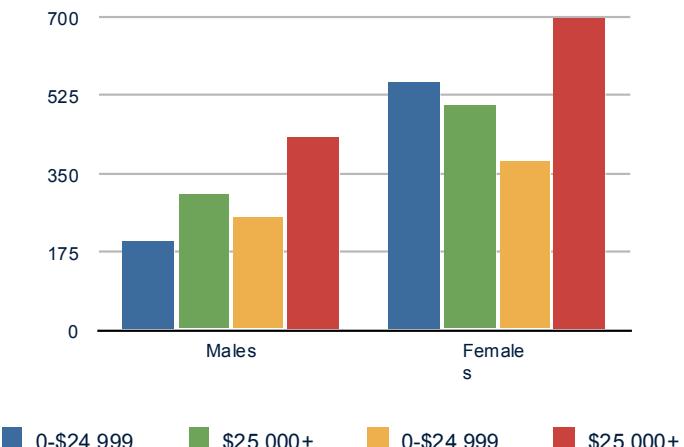
- Size of the graphic effect should be directly proportional to the numerical quantities (“lie factor”)



# Maximize data-ink ratio

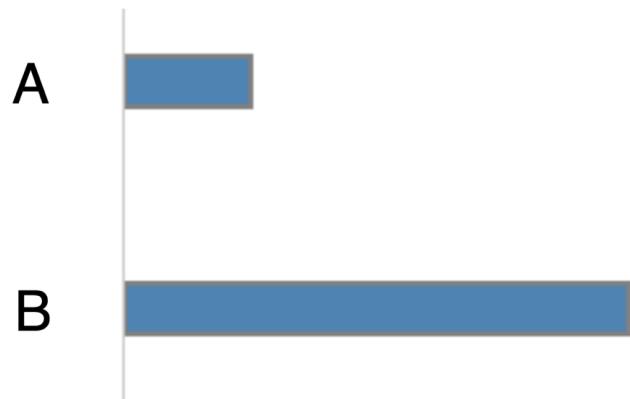


Data-Ink Ratio =   
*We Mean Money*

$$\frac{\text{Data ink}}{\text{Total ink used in graphic}}$$




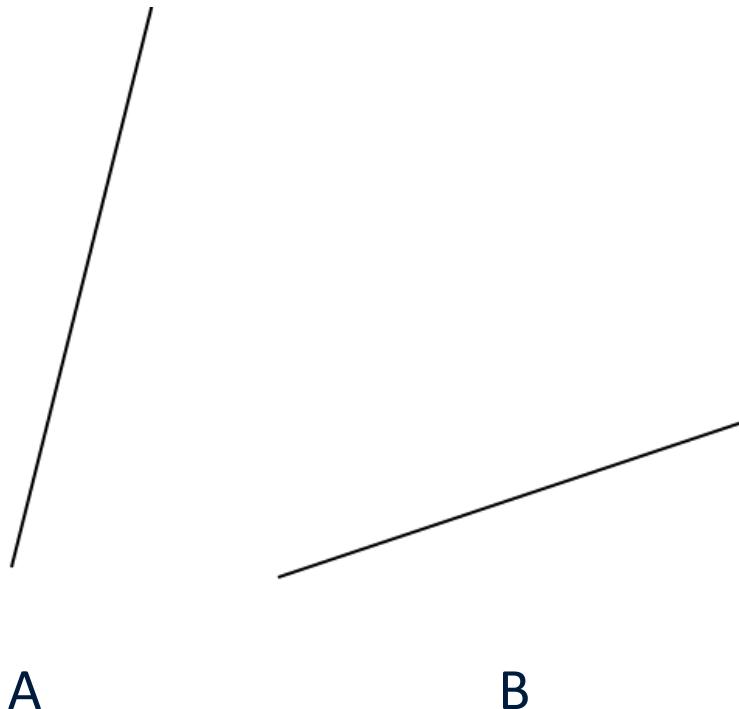
## How much longer?



- A. 2
- B. 3
- C. 4
- D. 5
- E. 6



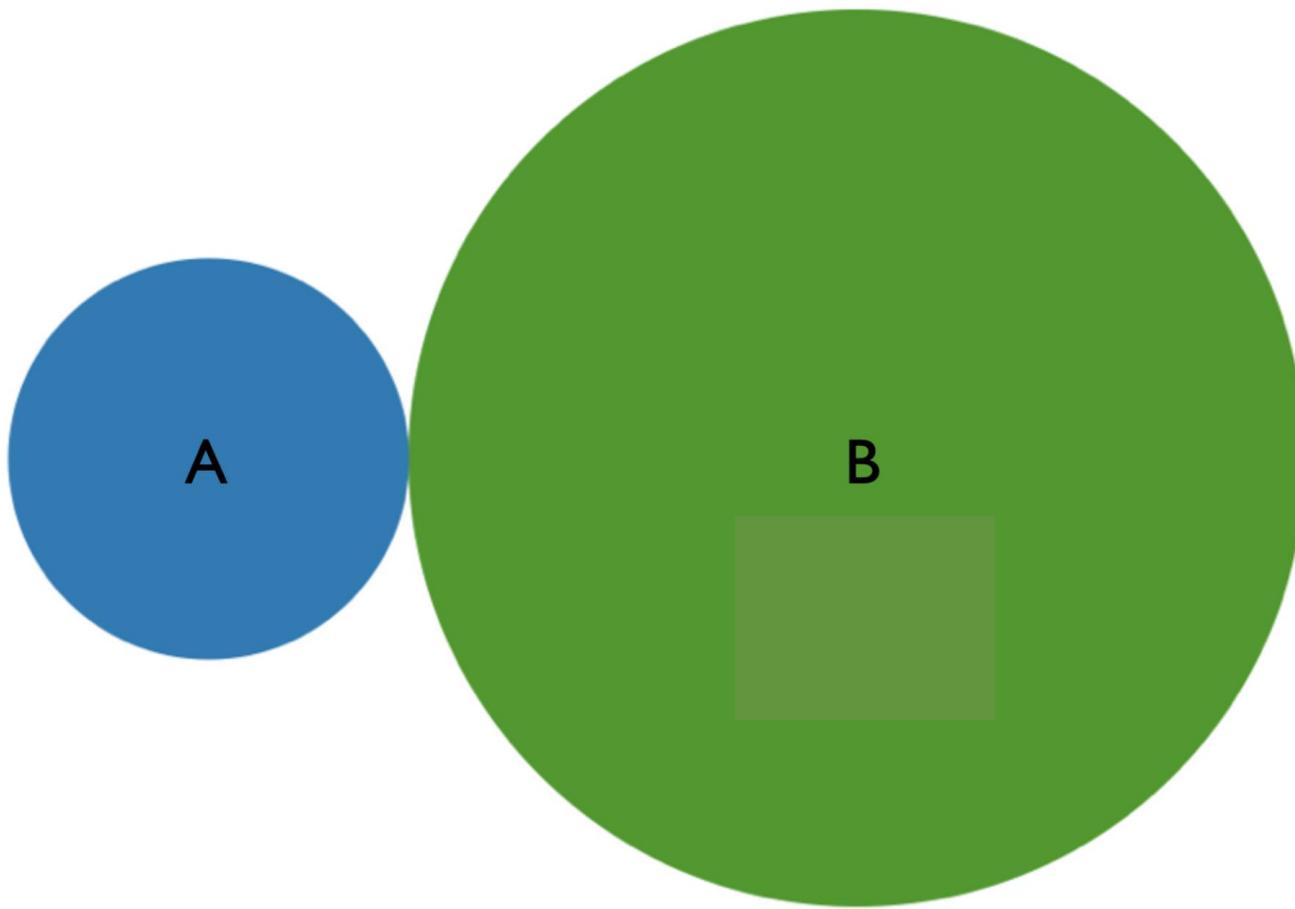
## How much steeper slope?



- A. 2
- B. 3
- C. 4
- D. 5
- E. 6

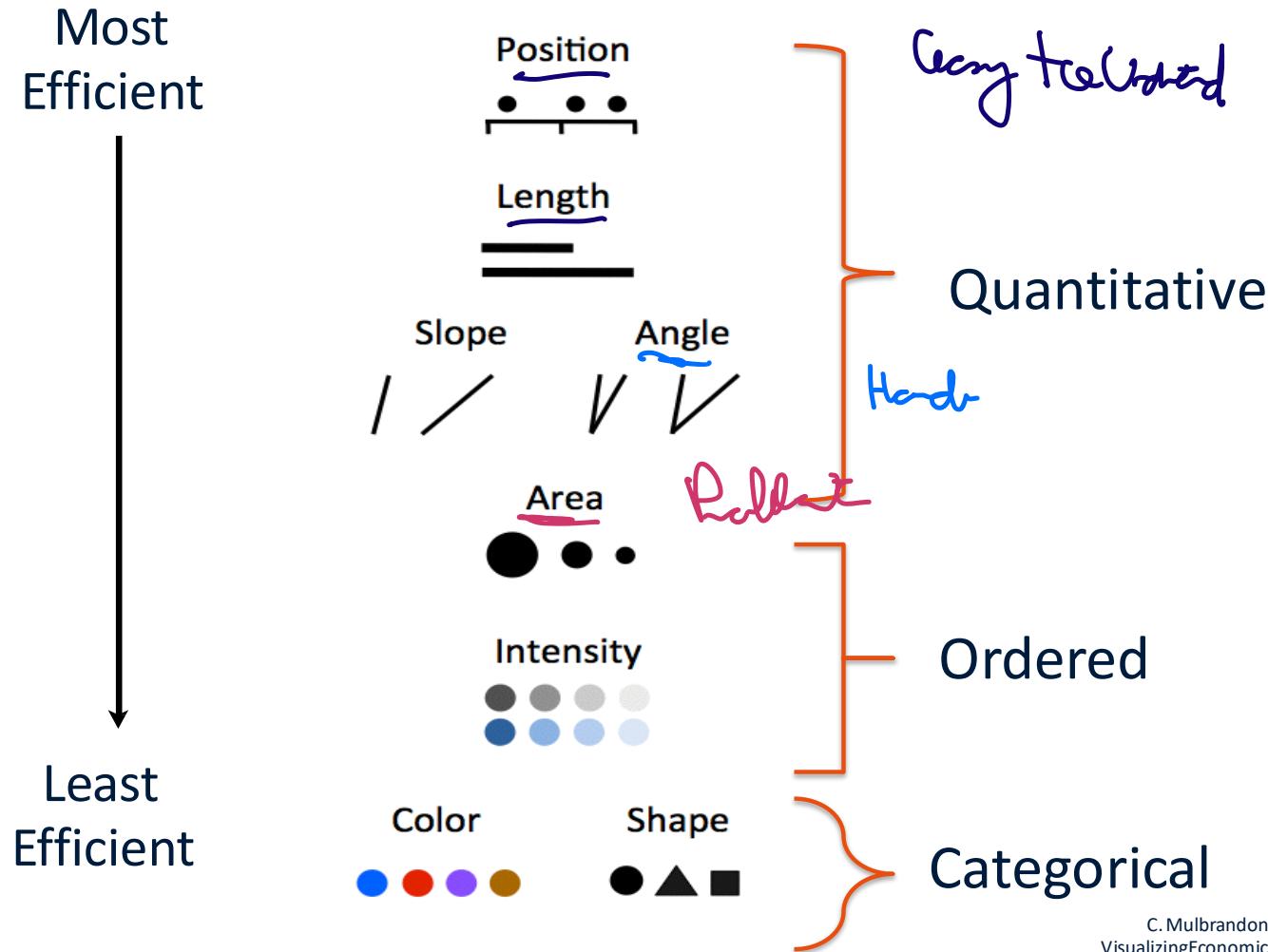


## How much larger area?



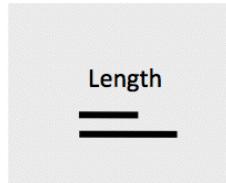
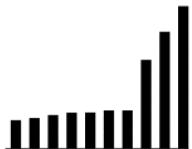
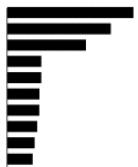
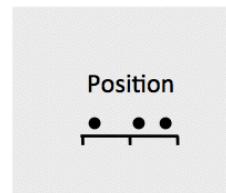
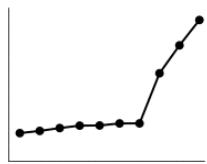
- A. 2
- B. 3
- C. 4
- D. 5
- E. 6

# 30 years of visualization research in 1 slide

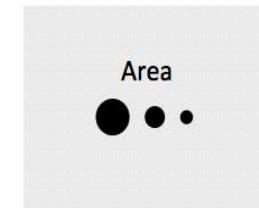
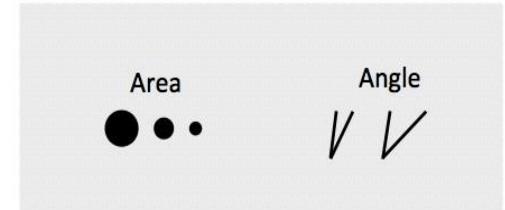
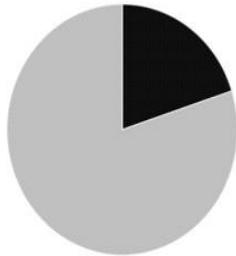


# Most vs. less effective

We Prefer This Than ^

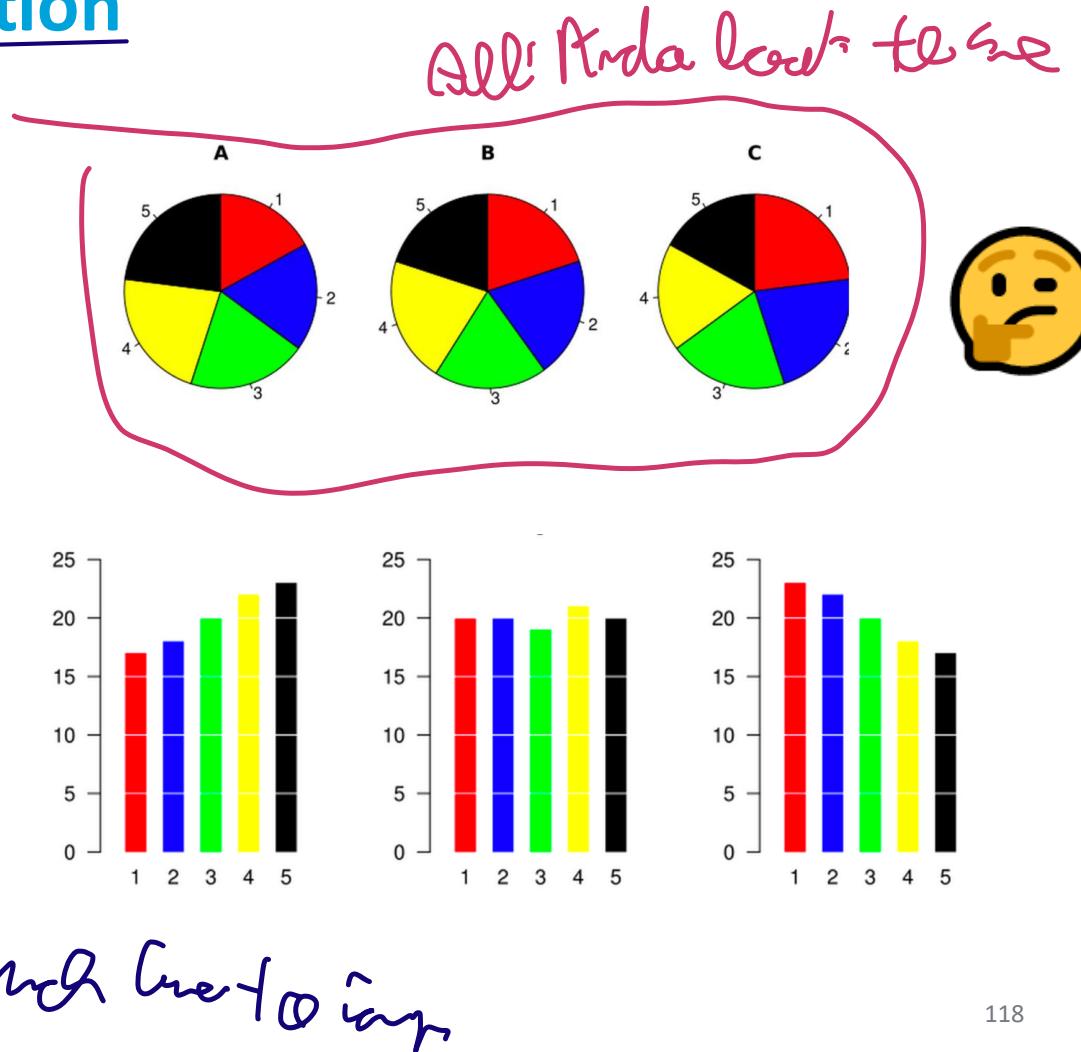
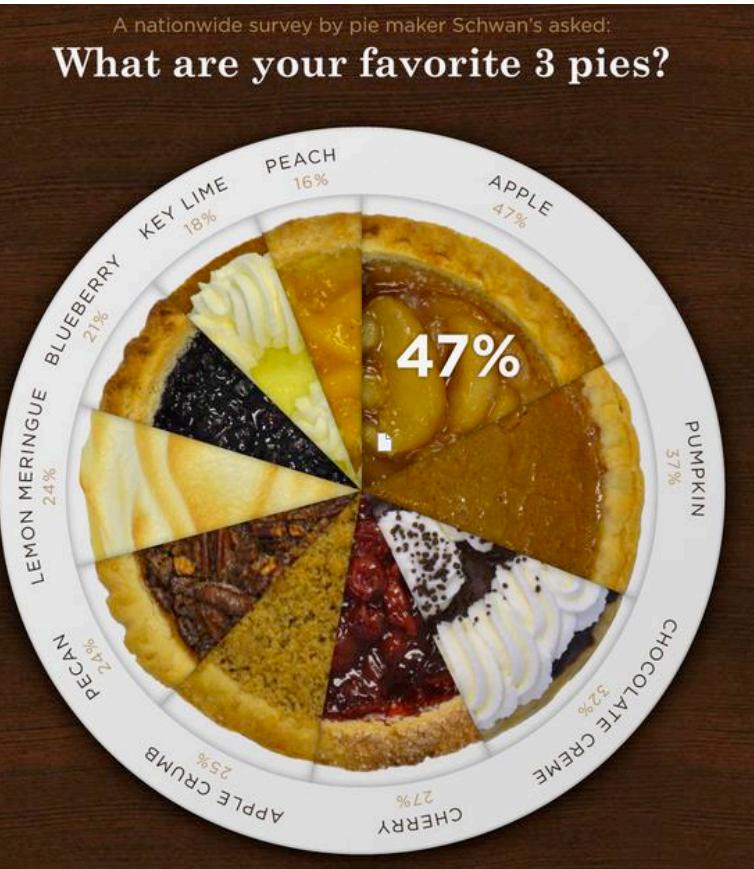


Why Percent  
is not good



# Pie charts have bad reputation

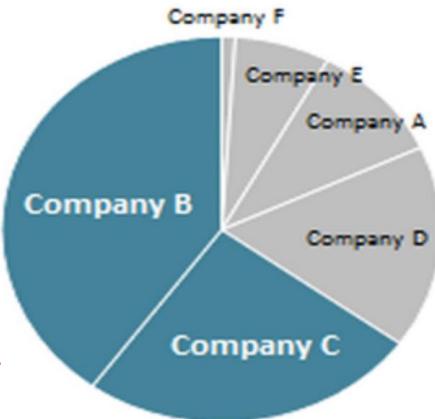
<http://eagerpies.com/>



# ...but sometimes they are useful

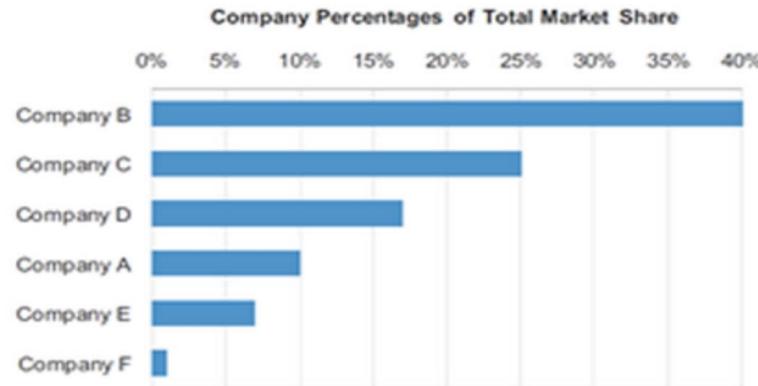


65% of the market is controlled by companies B and C



B  
B  
C  
F  
C

It's like me



Then there

### 3. Use a sensible design

Use color strategically

- For quantitative variables: Do not use color
- For ordinal data: Try varying luminance & saturation
- For categorical data: Do not use more than 5-8 colors



# Least effective

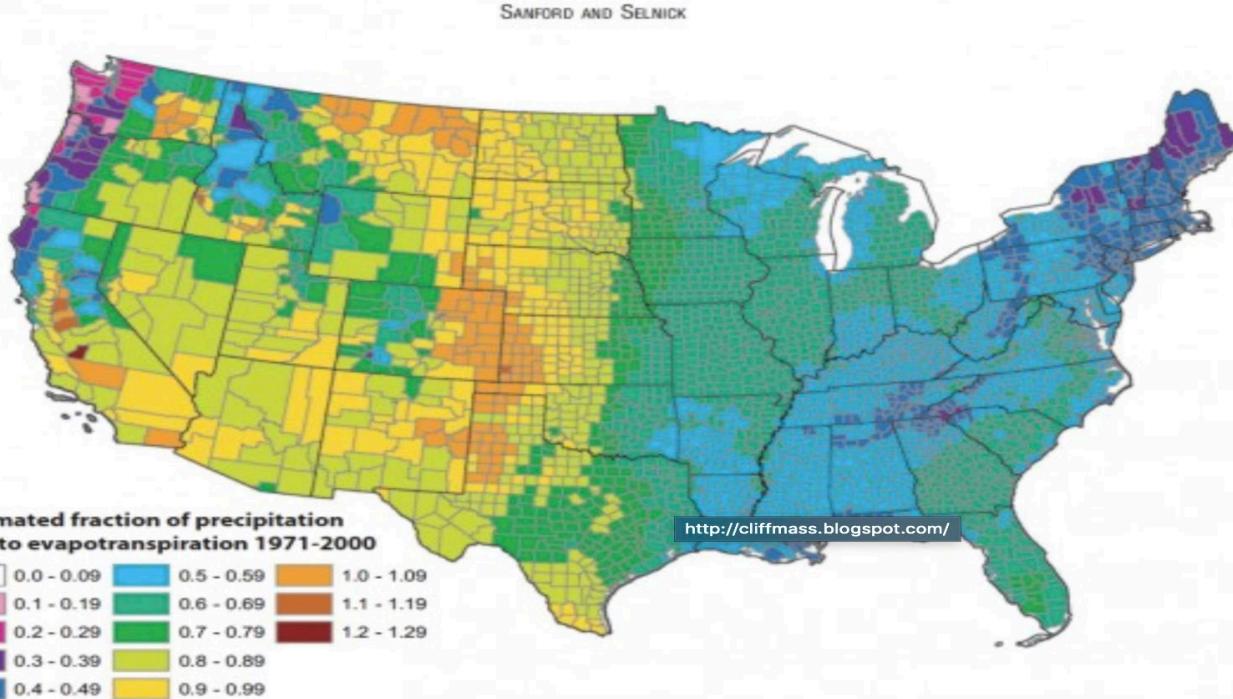
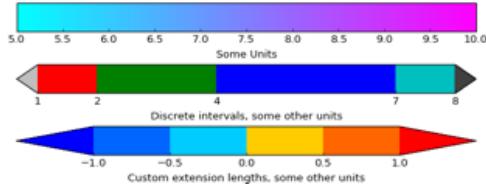


FIGURE 13. Estimated Mean Annual Ratio of Actual Evapotranspiration (ET) to Precipitation ( $P$ ) for the Conterminous U.S. for the Period 1971-2000. Estimates are based on the regression equation in Table 1 that includes land cover. Calculations of  $ET/P$  were made first at the 800-m resolution of the PRISM climate data. The mean values for the counties (shown) were then calculated by averaging the 800-m values within each county. Areas with fractions >1 are agricultural counties that either import surface water or mine deep groundwater.

# Avoid Rainbow Colors!



Nat Radley Ciffero

Very Hard to  
Read

# Color blindness

Avoiding Red

- 10% of males, 1% of females
- Most common: red-green weakness / blindness



Normal Color Perception

Deuteranopia (no green receptors)

Protanopia (no red receptors)

- There are packages that take care of this!

## **4. Choose the right display**

---

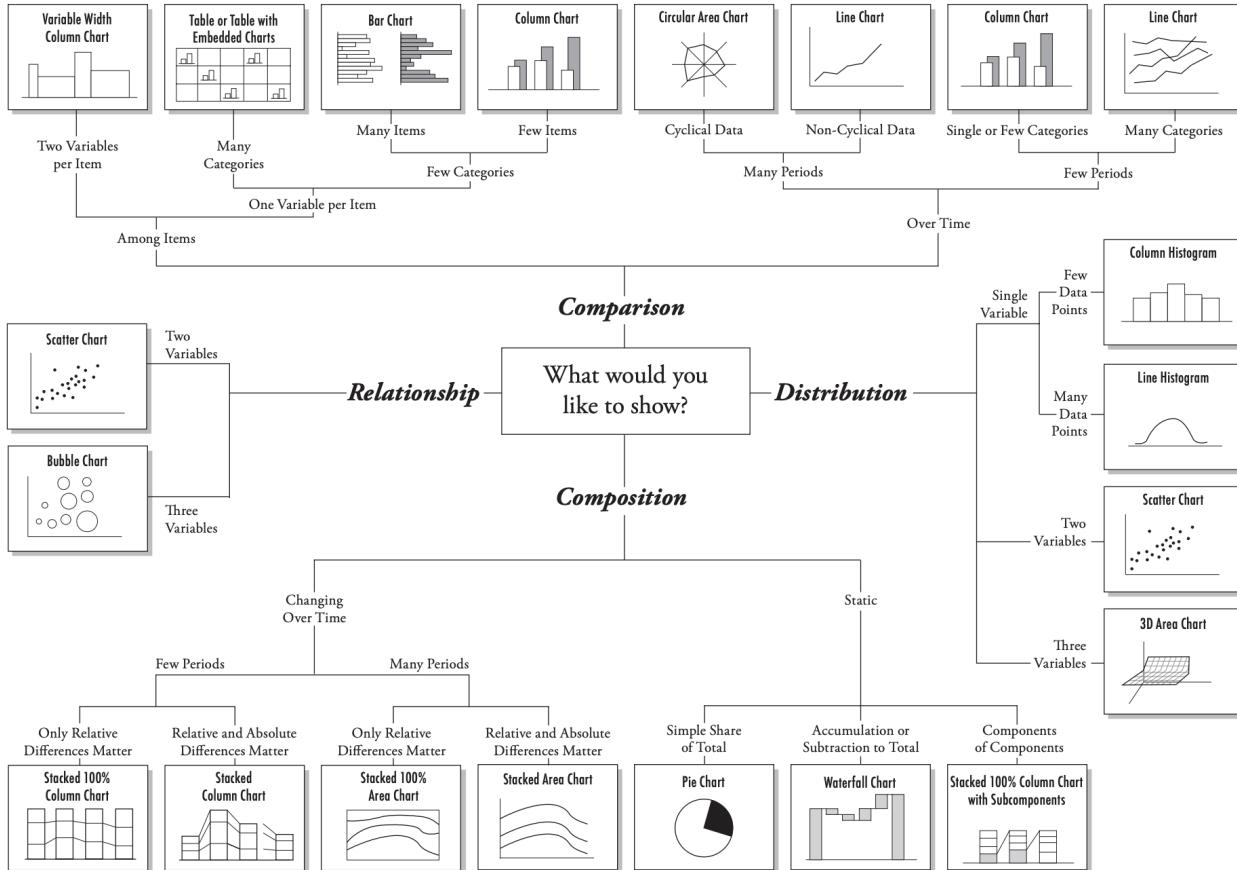
What do you want your visualization to show about your data?

- **Distribution:** how a variable or variables in the dataset distribute over a range of possible values.
- **Relationship:** how the values of multiple variables in the dataset relate
- **Composition:** how the dataset breaks down into subgroups
- **Comparison:** how trends in multiple variable or datasets compare

# 4. Choose the right display

Deven

## Chart Suggestions—A Thought-Starter



# Summary

- Reality check on your dataset
- Per variable:
  - Consistency check, missing data, outliers, noise...
  - Histograms, barplots, boxplots, etc.
  - Transformations/engineering possible
- Per pairs of variables (or more):
  - Associations (conditional tables)
  - Correlations (linearity check) for quantitative variables
- Tell your story via scientific, verbal and visual means