LIVE ONLINE TRAINING

Business Analytics With Python Bootcamp

Week 3: Descriptive Business Analytics

Agenda

1. **Recap & Intro**
   (15 minutes)
2. **Introduction to Descriptive Analytics (30 minutes)**
3. **Data Wrangling with Python (60 minutes)**
   - Exercise: Data wrangling with Python
   - Break
4. **Descriptive Statistics (60 minutes)**
   - Exercise: Descriptive statistics with Python
5. **Data Visualization (60 minutes)**
   - Exercise: Data visualization in Python
   - Outlook for next week

Recap

Bootcamp overview

Learning goals:

- Derive actionable insights from data
- Perform exploratory data analysis and create meaningful visualizations
- Use value-based analysis techniques and create association rules for effective decision support
- Apply clustering techniques to discover segments in your data, e.g. different customer groups
- Build predictive models for regression and classification tasks
- Understand the key criteria for evaluating the performance of a predictive model
- Suggest specific business actions that will lead to better results

Bootcamp overview

Last week

Week 1 Jan 31	Week 2 Feb 7	Week 3 Feb 14	Week 4 Feb 21	Week 5 Feb 28	Week 6 Mar 7
Business Analytics Fundamentals	**Data Analysis With Python Fundamentals**	**Descriptive Business Analytics With Python**	**Diagnostic Business Analytics With Python**	**Predictive Business Analytics With Python**	**Prescriptive Business Analytics With Python**
Goal: Get the theoretical foundations for doing business analytics	**Goal:** Get the practical foundations for using Python for data analysis	**Goal:** Learn how to explore & describe your data (What happened?)	**Goal**: Learn how to get insights from your data (Why did it happen?)	**Goal:** Learn how to predict the future, given historical data (What will happen?)	**Goal:** Suggest better actions based on data (What to do?)

NOTE: With today's registration, you'll be signed up for all six sessions. Although you can attend any of the sessions individually, it's recommended participating in all six weeks.

Quiz time!

Which of these statements about Python is NOT true?

A) Python has a large ecosystem

B) Python is one of the world's most popular programming languages

C) Python is one of the most performant programming languages

D) Python is relatively easy to learn

Which of these is NOT a Python package?

A) Numpy

B) Pandas

C) Seaborn

D) Mewtwo

What's NOT a good use case for a Jupyter Notebook?

A) Exploratory data analysis

B) Present work to a less technical audience

C) Write functions that are used in other projects

D) Collaborate with others

When should you use code versioning?

A) Whenever you write code

B) Only when you’re working in a larger team

C) Only when you’re working in a distributed team

D) When you have enough time

What's a good name for a variable that subtracts two timestamps?

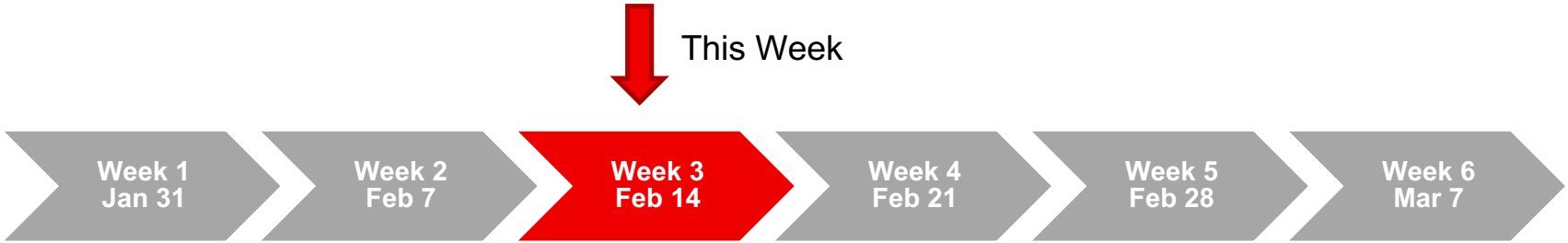
A) d

B) TimeAMinusTimeB

C) elapsed_time_in_days

D) elapsed_time

Bootcamp overview



Week 1 Jan 31	Week 2 Feb 7	Week 3 Feb 14	Week 4 Feb 21	Week 5 Feb 28	Week 6 Mar 7
Business Analytics Fundamentals	**Data Analysis With Python Fundamentals**	**Descriptive Business Analytics With Python**	**Diagnostic Business Analytics With Python**	**Predictive Business Analytics With Python**	**Prescriptive Business Analytics With Python**
Goal: Get the theoretical foundations for doing business analytics	**Goal:** Get the practical foundations for using Python for data analysis	**Goal:** Learn how to explore & describe your data (What happened?)	**Goal**: Learn how to get insights from your data (Why did it happen?)	**Goal:** Learn how to predict the future, given historical data (What will happen?)	**Goal:** Suggest better actions based on data (What to do?)

NOTE: With today's registration, you'll be signed up for all six sessions. Although you can attend any of the sessions individually, it's recommended participating in all six weeks.

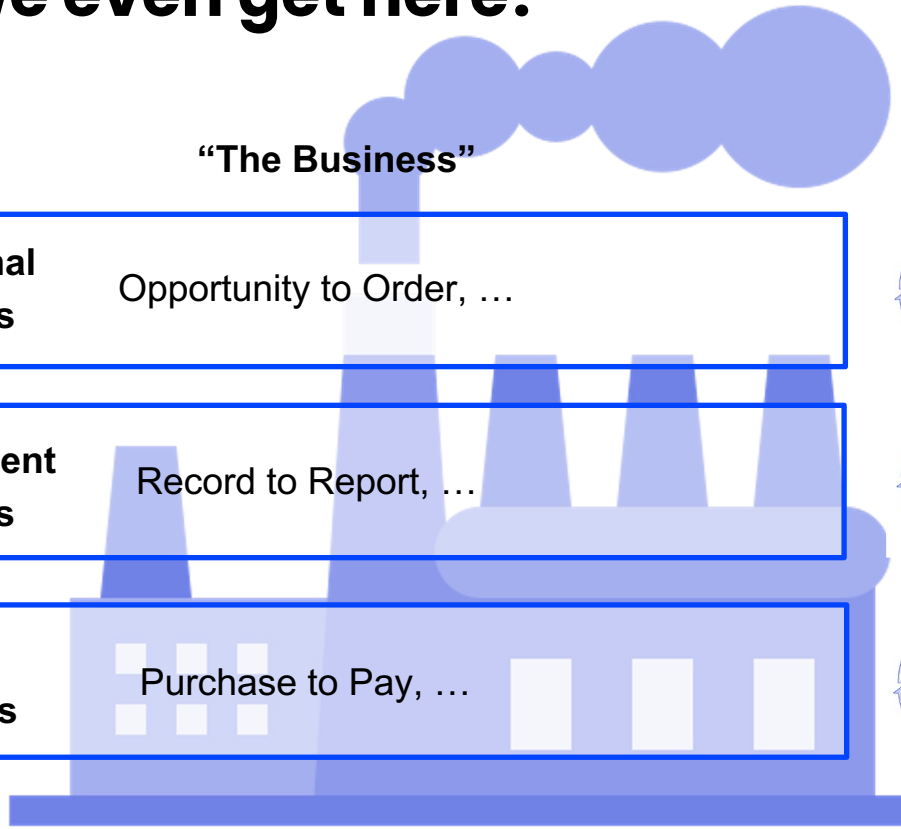
Bootcamp overview

Learning goals Week 3:

- Understand the essence of descriptive analytics
- Understand the concept of exploratory data analysis (EDA)
- Choose the right data model for data analysis (tidy)
- Transform data with Python
- Load data with and without SQL
- Write / Export data with Python
- Conduct descriptive statistics
- Conduct summary statistics & Anscombe's quartet
- Understand Data visualization techniques
- Use visualization frameworks in Python

Introduction to Descriptive Analytics

How did we even get here?

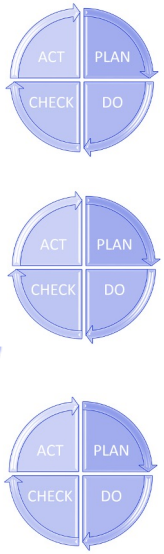


“The Business”

Operational Processes — Opportunity to Order, …

Management Processes — Record to Report, …

Support Processes — Purchase to Pay, …

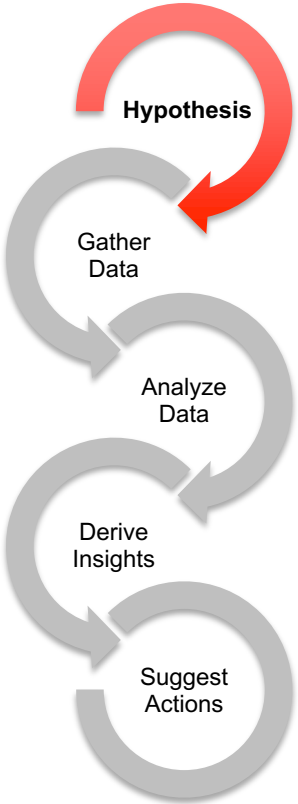


Business Analytics

How did we even get here?

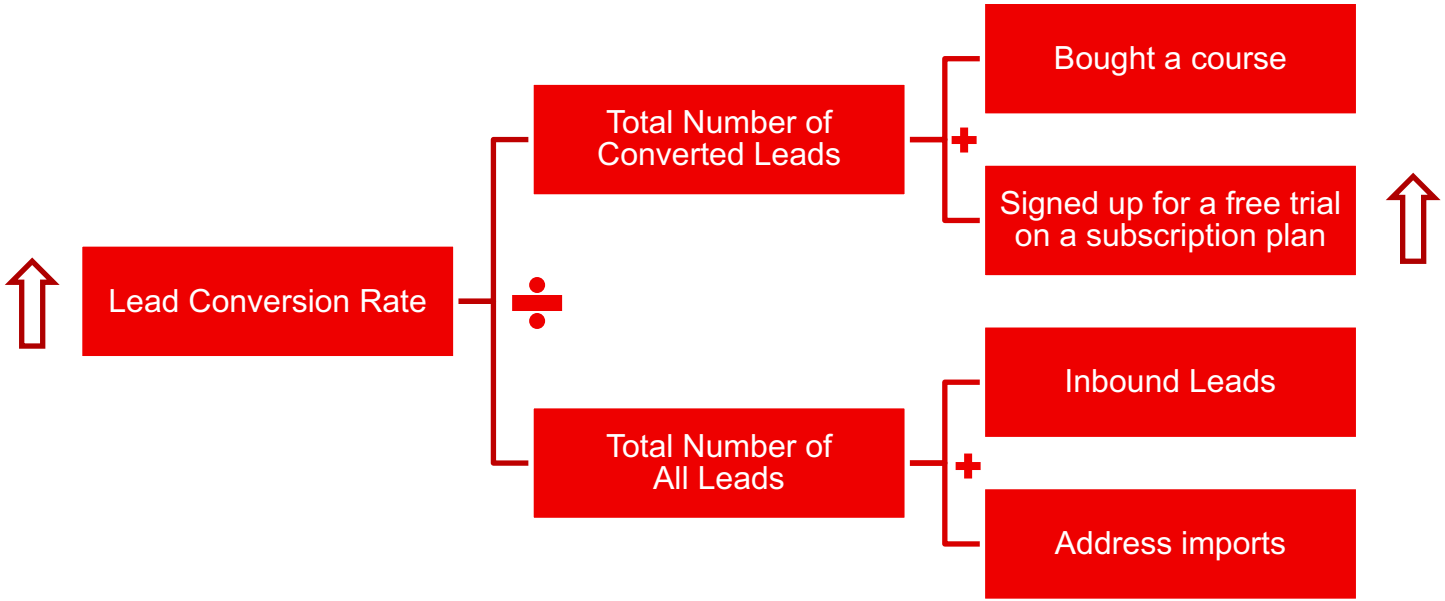
Hypothesis-Driven Analytics Framework

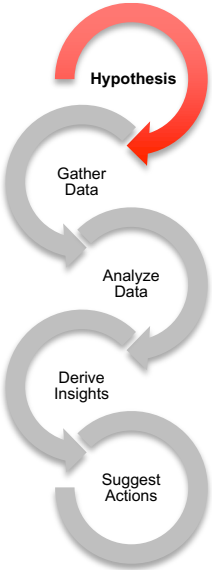
- What is the problem you're facing?
- Understand the unique challenges
- How do you describe the problem?
- Gather expert knowledge
- **Avoid analysis paralysis**: Gather your thinking around a centralized theme (hypothesis)
    - SMART Problem Statement
    - Issue Trees



How did we even get here?

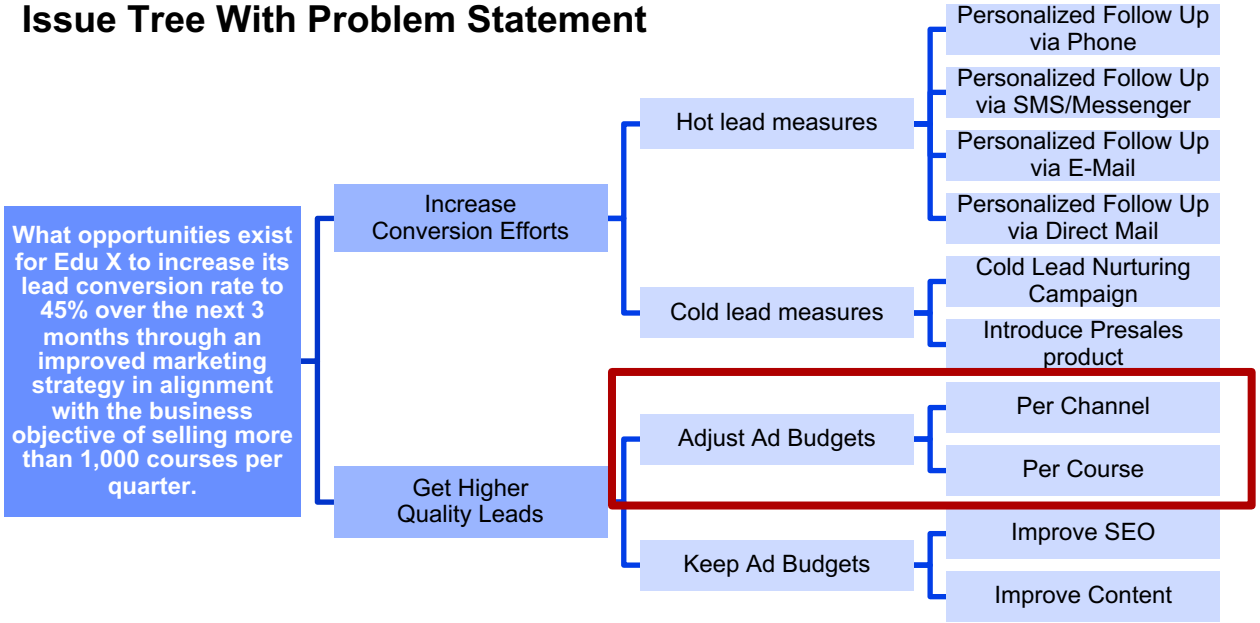
Value Driver Tree

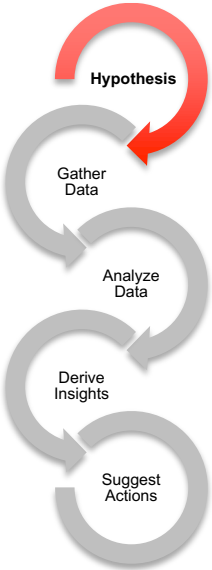




How did we even get here?

Issue Tree With Problem Statement



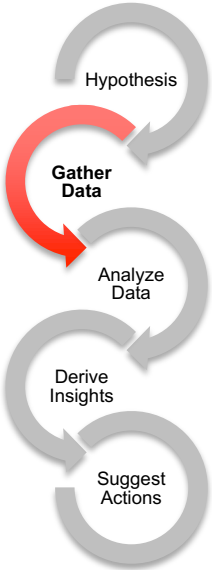


Time to do data analysis!

Hypothesis-Driven Analytics Framework

Step 2: Gather data

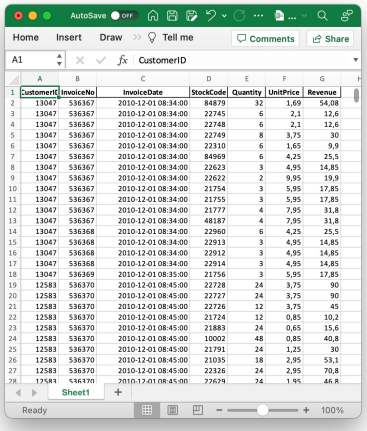
- Which data sources are needed to answer questions in the hypothesis?
- Get access to data
- Collect data if necessary



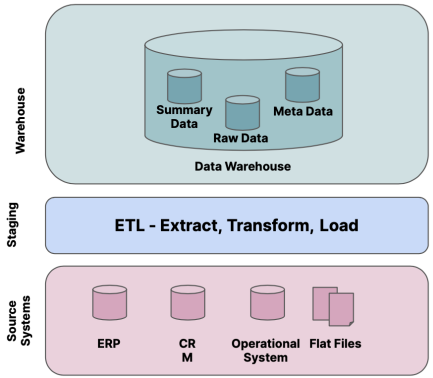
Accessing data

You'll typically access data through one of these sources:

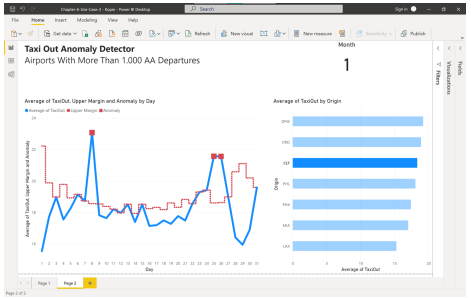
File

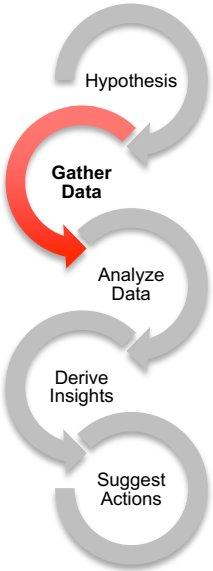


Data Warehouse / Data Lake



BI Tool





Files

Popular file types

.XLSX	- Proprietary file format used by Microsoft Excel to store spreadsheet data. - Supports a wide range of data types and features such as formulas, charts, and pivot tables. - Compatibility issues may arise.
.CSV	- Comma-separated values = Structure of a text file for storing or sharing simply structured data. - Separation also possible using other characters (e.g. semicolon). - Usually used for the exchange of tables or lists.
.JSON	- **J**ava**S**cript **O**bject **N**otation = compact format, primarily for data exchange between applications. - JSON can be read by all common languages.
.AVRO	- Apache Avro: **row-based** storage format, often used in the context of Big Data. - Stores schema in JSON format, making it easy to read and interpret by various programs. - Data itself is stored in binary from (no text), making it compact and efficient.
.PARQUET	- Apache Parquet: **column-oriented** storage format, optimized for very large files (100MB+) - Allows more efficient data compression (similar data in columns) & accelerated reads

→ Python can read and write all of these (usually through a package).

Data Warehouse & Data Lake

Data Warehouse:

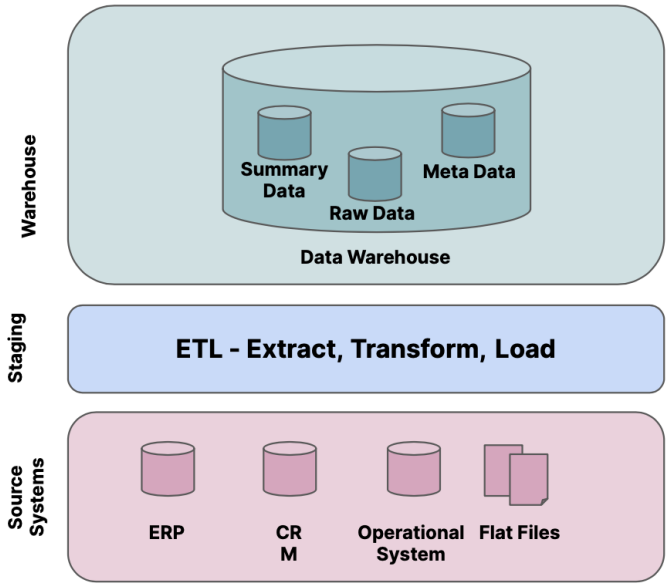
- Data is typically stored in a relational schema (e.g., STAR schema)
- Data is often organized into smaller subsets (“data marts”, “cubes”) for improved access and performance
- Data is normalized (avoid redundancies)

Data Lake:

- Data is usually accessible in raw form (schema-free or schema-less)
- Schema enforcement happens closer to the application layer

Data Access:

- Primarily SQL in both cases
- Python supports SQL (e.g., `pd.read_sql`)

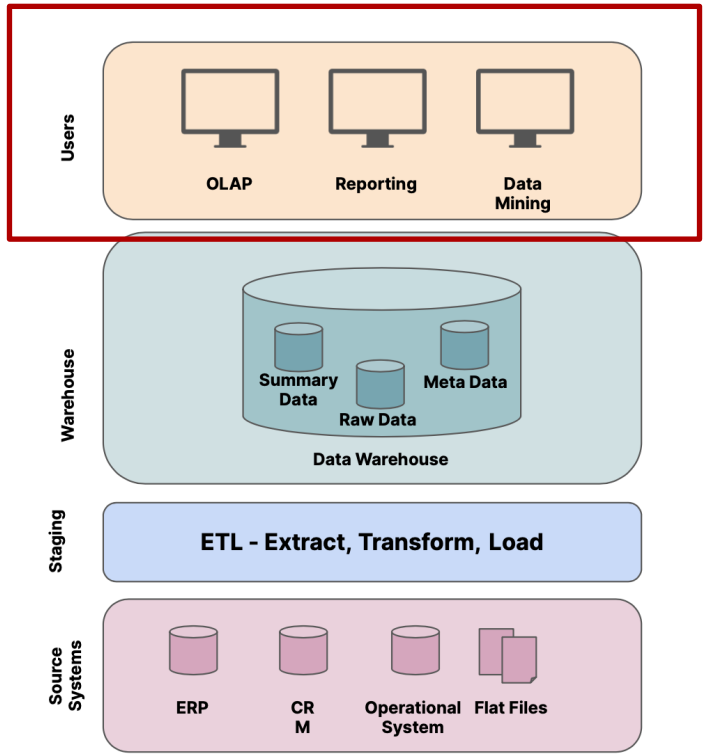


BI Tools

- BI Tools (e.g, Power BI, Tableau, Qlik, etc.) are typically the application layer on top of a data warehouse or data lake
- Let you analyze data directly within the BI tool

Options for analyzing data from a BI tool with Python:

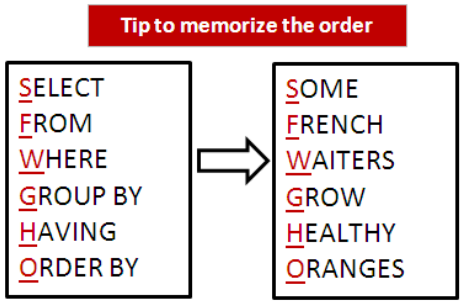
1. Write / run Python code directly in the BI Tool (e.g., Power BI)
2. Export the data (e.g., CSV) and analyze outside BI tool
3. Use SQL to access data from the DWH directly



SQL

What is SQL?

- A programming language that is considered a de-facto standard for querying data from databases (and can do much more than that...).
- ...has different "dialects", depending on which database is used.
- Basic structure always identical:




```
SELECT *
FROM covid_cases;
```

All data of the table covid_cases.

```
SELECT date, cases
FROM covid_cases
WHERE country = “DE“;
```

The columns "Date" and "Cases" from the table covid_cases, where the country is "DE” (Germany).

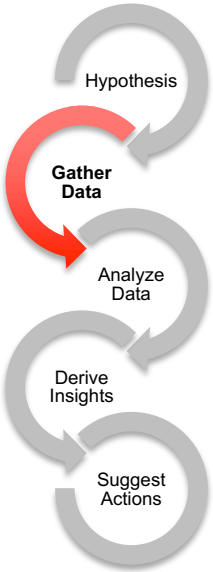
```
SELECT date, cases
FROM covid_cases
WHERE country = “DE“
ORDER BY date DESC;
```

... sorted in descending order by date.

```
SELECT year, SUM(cases)
FROM covid_cases
WHERE country = “DE“
GROUP BY year
ORDER BY year DESC;
```

... cases cumulated by year for Germany.

Now that we have access...

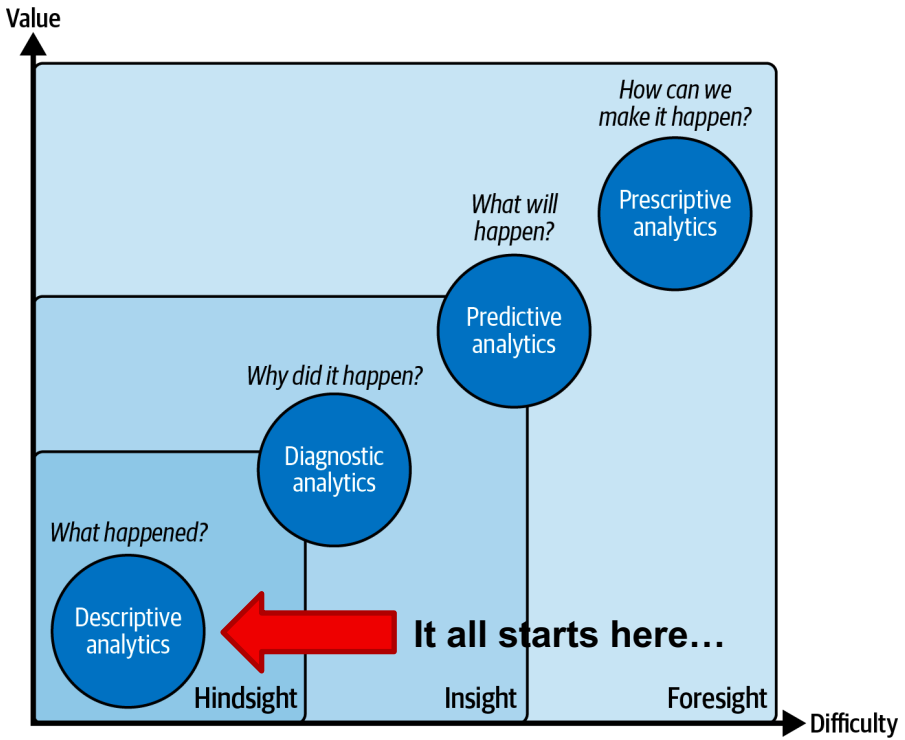


What do we do with the data?

→ Going from data data-informed decisions typically involves multiple steps:

- **Descriptive**: Understand what happened
- **Diagnostic**: Understand why it happened (and what levers you can pull)
- **Predictive**: Understand what will happen in the future
- **Prescriptive**: Suggest actions to achieve a desired outcome

→ These steps build on top of each other!



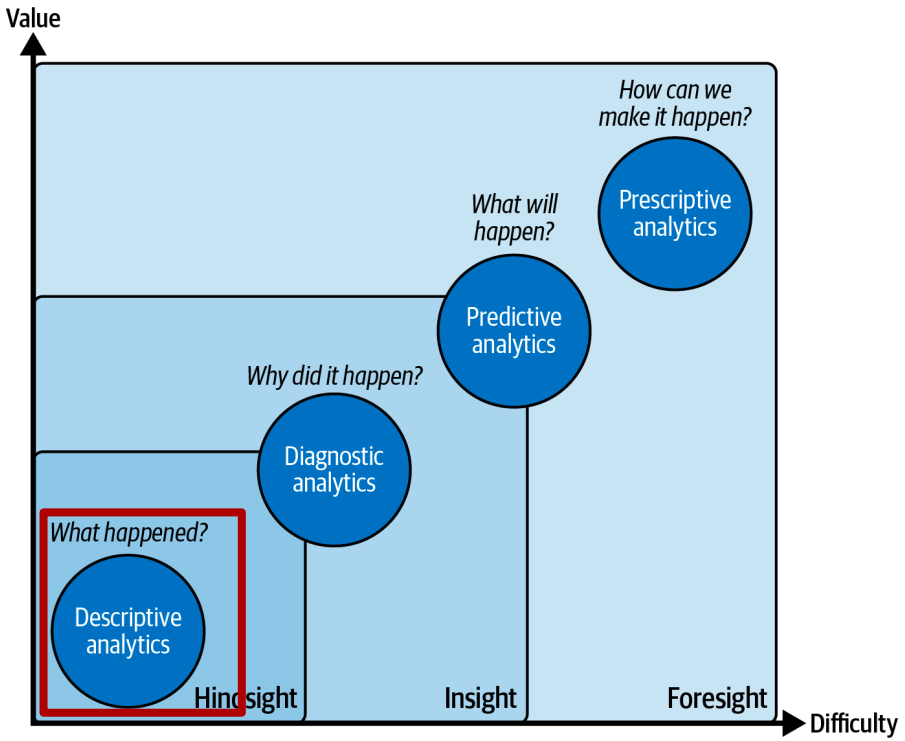
Introduction to Descriptive Analytics

Descriptive analytics allows us to…

- Describe what happened in the past
- Find unusual patterns / flaws in our data
- Monitor an ongoing process

Descriptive analytics does not allow us to…

- Interpret why things have happened (mostly)
- Make inference beyond the data we have seen



Exploratory Data Analysis

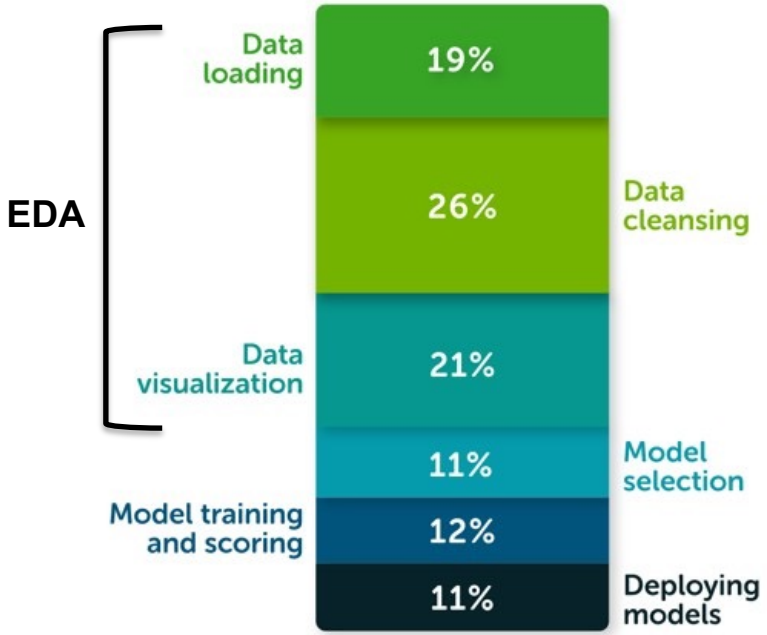
What is Exploratory Data Analysis (EDA?)

- EDA is an essential first step in every data analytics project
- It's part of the descriptive analytics phase (no "conclusions")
- The goal is to **generate more detailed hypotheses** (e.g. "Getting more traffic from Facebook ads does not increase lead conversion rate.")
- **Searching for the right questions** before providing answers (e.g., "Which fixed asset has the largest share and how has it evolved over the past 3 years?")
- → **Finding "surprises"** (e.g., data quality issues, outliers).

Key ingredients:

- Grouping variables into continuous, categorical, etc.
- Summarizing variables with descriptive statistics (mean, variance, etc.)
- Visualization of data with charts

→ **Exploratory data analysis is the basis for a successful analytical project!**



- Source: Amazon Web Services Blog

Data Modeling

- The same data can be modeled in different ways
- Data models help us to make sense of the data
- Not all data models are equally well suited for analytics

1 Table 6 x 4

Country	Year	Cases	Population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

1 Table 12 x 4

Country	Year	Variable	Value
Afghanistan	1999	Cases	745
Afghanistan	1999	Population	19987071
Afghanistan	2000	Cases	2666
Afghanistan	2000	Population	20595360
Brazil	1999	Cases	37737
...	...	...	...

2 Tables, each 3 x 3

Country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

Country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

Relational Data

- Relational data is data that is related to each other **(= relations**).
- A relational database is a collection of data organized according to the relational model and serving a specific purpose.
  Example 1: "Customer A belongs to company B", customer A is stored in the "Contacts" table, company B in the "Accounts" table. A third table might contain customer ID, Account ID and some measures (e.g., total sales) → Star schema
- Opposite: Hierarchical principle (e.g., file system of a PC).

→ Relational data is structured (fixed schema, rarely changed).

→ Relational data is not nested.

→ Use NULL values for non-existent data (opposite XML: entry does not exist).

Table "Cases"

Country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

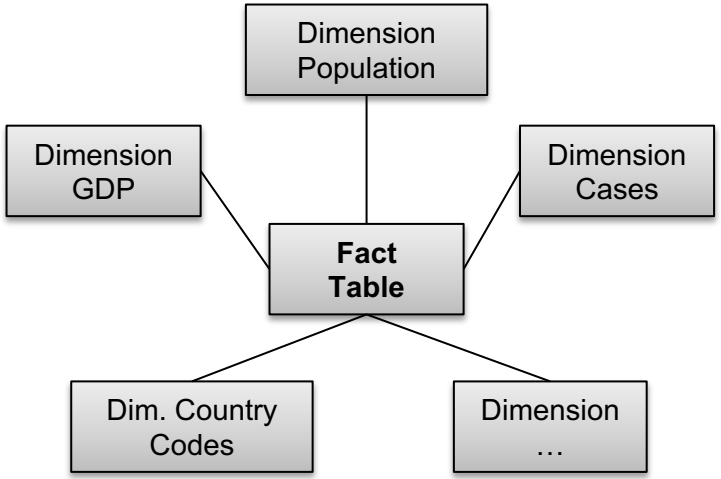
Table "Population"

Country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

Question: What is problematic about this data model?

Star Schema

- Star schema concept is widely used in data warehousing (Often used synonymously: data cube / cube, multidimensional schema).
- Advantages: Few redundancies, data is normalized.
- Organization into fact and dimension tables
- Fact tables store primary measures / metrics / business activities.
  → Goal: Fast calculation
- Dimension tables contain further information that impacts these activities.
  → Goal: Slice & Dice



Organizing Data For Analytics

- Which of these data models is most effective for analytics?

1 Table 6 x 4

Country	Year	Cases	Population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

1 Table 12 x 4

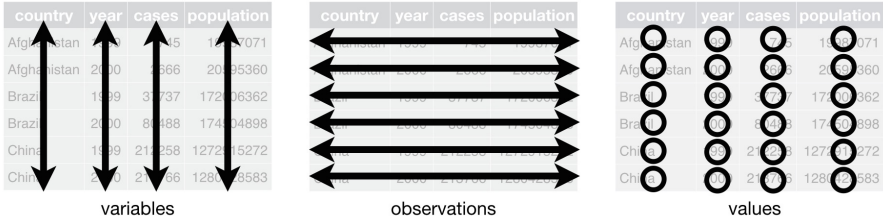
Country	Year	Variable	Value
Afghanistan	1999	Cases	745
Afghanistan	1999	Population	19987071
Afghanistan	2000	Cases	2666
Afghanistan	2000	Population	20595360
Brazil	1999	Cases	37737
...	...	...	...

2 Tables, each 3 x 3

Country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

Country	1999	2000
Afghanistan	19987071	20595360
Brazil	172006362	174504898
China	1272915272	1280428583

Tidy Data



There are three interrelated rules that make a dataset tidy:

- **Each variable is a column**
- **Each observation is a row**
- **Each value is a cell**

Country	Year	Cases	Population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

- Source: R For Data Science, Hadley Wickham (https://r4ds.hadley.nz/data-tidy.html)

Why Tidy Data?

Two main advantages:

- **General advantage**: Uniform model of data storage leads to consistent data structures. When data has a consistent data structure, it is easier to analyze. Most analysis tools handle this format well.
- **Specific advantage**: Script languages such as R or Python can process data vectorized (same data type in a row) and thus enable particularly simple and fast arithmetic operations

Country	Year	Cases	Population	**Rate**
Afghanistan	1999	745	19987071	**0.373**
Afghanistan	2000	2666	20595360	**1.29**
Brazil	1999	37737	172006362	**2.19**
Brazil	2000	80488	174504898	**4.61**
China	1999	212258	1272915272	**1.67**
China	2000	213766	1280428583	**1.67**

Python:

```
df["Rate"] = df[”Cases"] / df[”Population"] * 10000
```

Q&A

Data Wrangling With Python

Data Wrangling

What is data wrangling?

- Data wrangling describes the process of getting the data into the right shape for analysis and ensuring that its quality is good enough (fit for use principle).

- Descriptive statistics help you identify the areas in which your data is not fit for use! Examples: Find the % of missing values, different types of name spellings, etc.

Data Wrangling

Basic tasks

- **Cleaning**: Resolve data quality issues

- **Integration**: Combine data from different sources

- **Transformation**: Transform data into a useful format

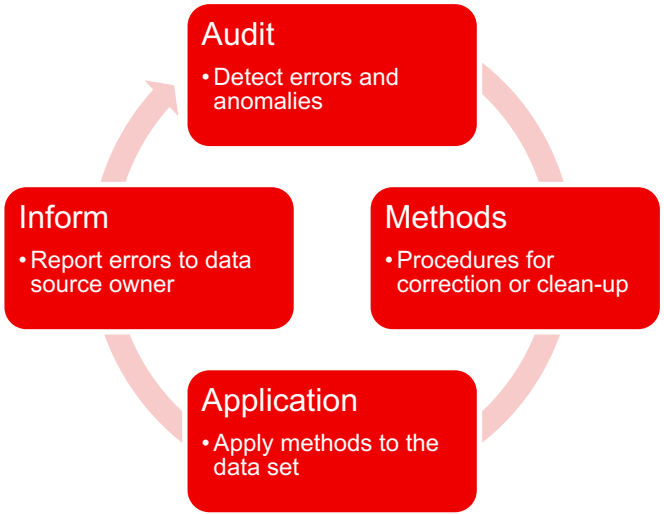
- **Enrichment**: Create new meaningful attributes

- **Reduction**: Reduce amount of data (sampling)

Data Cleaning Process

Data Quality = Degree to which data meets user requirements

- **Main principl**e: The goal is NOT to fix all ”errors” in the data. The goal is the find the relevant errors according to the problem you’re dealing with (Fit-for-use principle)
- Fixing data in a reliable, reproducible way while ideally informing the data source owner to prevent these errors from happen again.



Data Cleaning Audit

How to find errors in your data (automatically):

Pattern profiling:
- Text patterns (e.g., email address)
- Time patterns (e.g., date)
- Numerical patterns (e.g., phone number)

Single attribute profiling:
- Cardinality (count of attribute values)
- Frequency (frequency of attribute values)
- Mean, Median
- Max / Min
- Variance, Standard Deviation
- Count Null / Missing values

Multi attribute profiling:
- Functional dependencies (e.g., number of zip codes vs. number of cities)
- Logic rules (e.g., customer must be older than 18 years)
- Clustering (similarity over multiple attributes)

Data Cleaning Audit

Typical areas of conflicts

Conflicts within one data source:

- Illegal value (e.g., month 13).
- Violated attribute dependency (e.g., age and year of birth)
- Violated uniqueness (e.g., ID assigned multiple times)
- Missing value
- Spelling error
- Cryptic value
- Duplicate
- Wrong field entry
- Contradictory entry
- Incorrect reference

Conflicts with multiple data sources:

Schema conflicts

- Name conflicts (synonyms for same fields)
- Structural conflicts (e.g., first and last name separated)

Data conflicts

- Different representation (e.g. 0 / 1 vs. True / False)
- Duplicates
- Different scaling (e.g. °C vs. F)
- Different granularity (e.g. sales per product vs. sales per region)
- Different times (e.g. sales from yesterday vs. last week)

Data Cleaning Audit

Which errors are included in the following table?

Last Name	First Name	Birthdate	Age	Phone	Email	ZIP
Emma	Smith	1973-01-01	99	-1	emma.smith@gmail.com	08701
Miller		1983-31-10	37	089549123	gabriela.miller@live.com	803310
John	Snow		41		5l@x503.##+§	
JOHN	Snow		41		5l@x503.##	

Attribute dependency violation

Dummy value

Wrong field entry

Formatting error

Missing value

Illegal value

Cryptic value

Duplicate record

Data Cleaning Methods

Correct data errors:

- Obvious errors can be corrected or removed
- Be careful of what “obvious errors” are!

Fixed Error

Name	Vorname	Birthdate	Age	Phone	Email	ZIP
Emma	Smith	1973-01-01	50	-1	emma.smith@gmail.com	08701
Miller	Gabriela	1983-31-10	37	089549123	gabriela.miller@live.com	803310
John	Snow		41		5l@x503.##+§	
JOHN	Snow		41		5l@x503.##+§	

Data Cleaning Methods

Remove duplicates

- Duplicate observations should be removed, otherwise can cause wrong descriptive statistics
- Sometime hard to define a criteria what a duplicate is

Name	Vorname	Birthdate	Age	Phone	Email	ZIP
Emma	Smith	1973-01-01	50	-1	emma.smith@gmail.com	08701
Miller	Gabriela		37	089549123	gabriela.miller@live.com	803310
John	Snow		41		5l@x503.##+§	
~~JOHN~~	~~Snow~~	_	~~41~~	_	~~5l@x503.##+§~~	_

Remove duplicate

Data Cleaning Methods

Unification: Represent same values in the same format

- **Capitalization**: Adjust capitalization of categorical data
- **Concatenation**: Merge multiple string attributes through concatenation of their values (e.g. address, street number → ”address street number”)
- **Representation format**: Adjust date representation format
- **Character clean-up**: Remove / exchange unnecessary or confusing characters (e.g., $, %, &, \, …)

Name	Vorname	Birthdate	…
Emma	Smith	1973-01-01	…
Miller	Gabriela	31.10.1983	…
John	Snow	01. Feb 1979	…

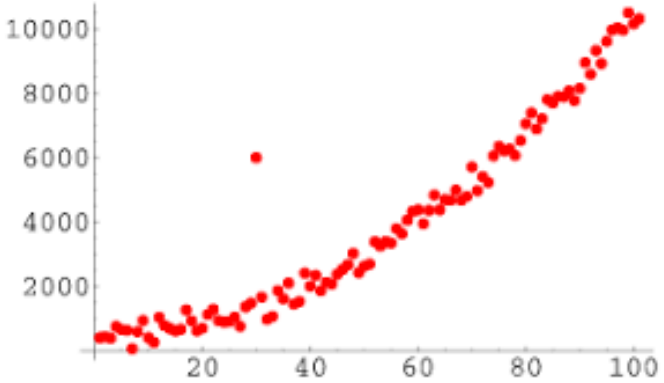
Name	Vorname	Birthdate	…
Emma	Smith	1973-01-01	…
		1983-10-31	…
John	Snow	1979-02-01	…

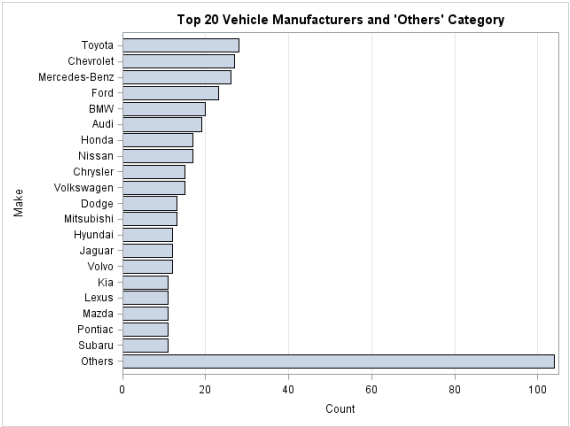
Data Cleaning Methods

Handle Outliers

- Decide whether to keep or remove outliers as they can have a substantial effect on your analysis
- What's an outlier? No universal definition. Rule of thumb: 3 times standard deviation
- Be careful: Sometimes "outliers" are the actual observations of interest!

- Also applies to categories with few or single observations → Lump together ("other")?





Data Cleaning Methods

Handle Missing Values

You have three options:

1. **Keep** missing values:
   - Sometimes a missing value has a meaning
2. **Remove** missing values
   - Remove by row (observation)
   - Remove by column (variable)
3. **Replace** missing values
   - Standard value (e.g., 0)
   - Mean of the attribute
   - Striking value (outside standard range, “unknown”)
   - Model based (inferred from other attributes)

Original Data

ID	Age	Weight	Goal
1	34	normal	general fitness
2	57	?	Weight loss
3	19	normal	Gain muscles

Striking value

ID	Age	Weight	Goal
1	34	normal	general fitness
2	57	**unknown**	Weight loss
3	19	normal	Gain muscles

Model based

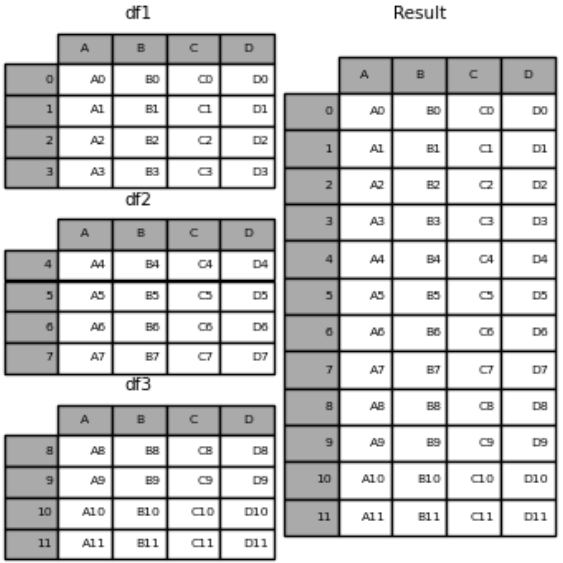
ID	Age	Weight	Goal
1	34	normal	general fitness
2	57	**overweight**	Weight loss
3	19	normal	Gain muscles

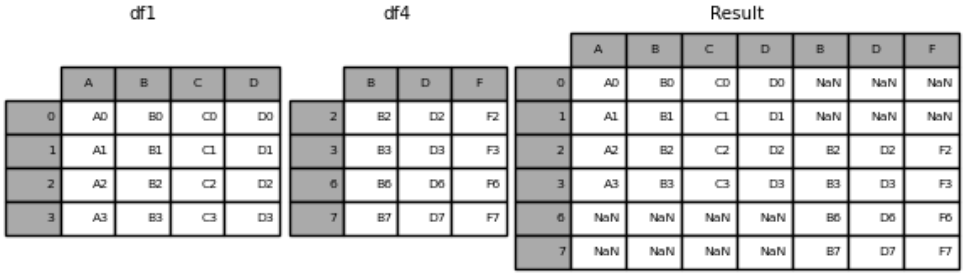
Data Integration

Goal: Combining data from different sources

Two main approaches:

- Concatenating
- Joining



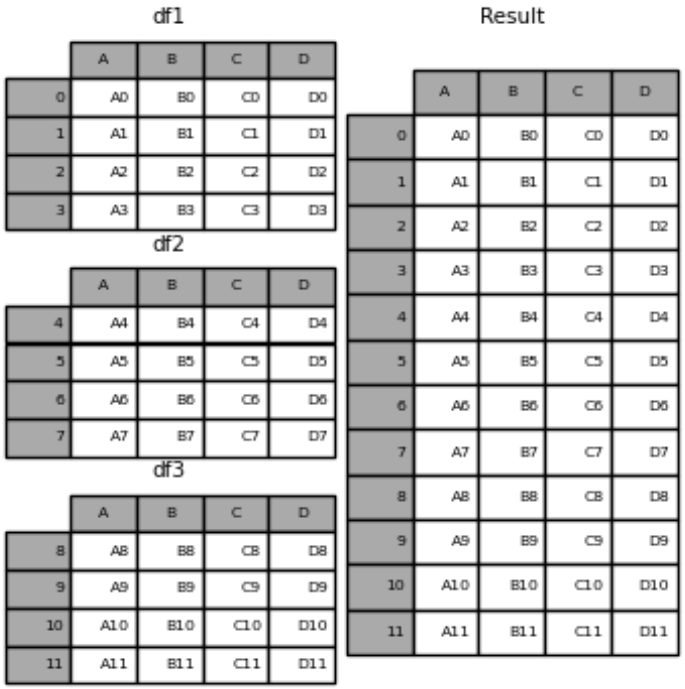


- Source: https://pandas.pydata.org/pandas-docs/dev/user_guide/merging.html

Data Integration

Concatenating

- Row-wise concatenation: Append rows to an existing table (must have the same schema / columns!)
- Column-wise concatenation: Append columns to an existing table (must have the same number of rows!)



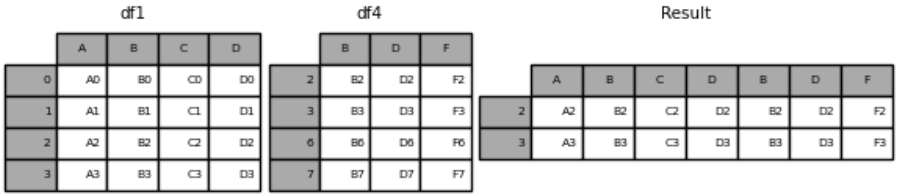
Row-wise concatenation

- Source: Pandas Documentation https://pandas.pydata.org/pandas-docs/dev/user_guide/merging.html

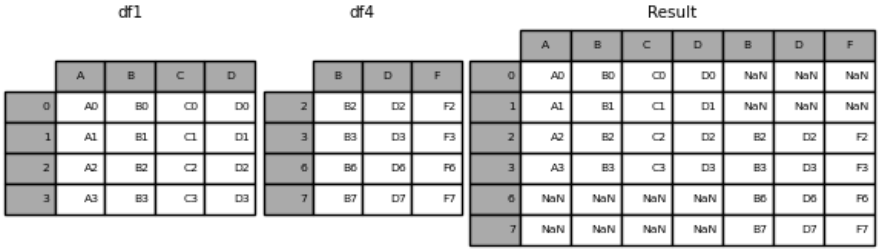
Data Integration

Joining (Merging)

- Join data from tables with different columns based on some shared key
- **Inner join**: Keep instances which are in both tables
- **Left join**: Keep instances in left table, will generate missing values for instances that were not in the right table
- **Right join**: Keep instances in right table, will generate missing values for instances that were not in the left table
- **Full (outer) join**: Combine instances of both tables



Inner join



Outer join

- Source: Pandas Documentation https://pandas.pydata.org/pandas-docs/dev/user_guide/merging.html

Data Transformation

	Scale	Characteristics	Allowed Operations	Example
Categorical	Nominal	Classes of distinguishable objects	=, ≠	red, green, yellow
	Ordinal	Classes with a rank order relationship	=, ≠, <, >	small, medium, large
Numerical	Interval	Numbers without a natural 0 value	=, ≠, <, >, +, -	3°C, 4°C 2021-10-01
	Ratio	Numbers with a unique 0 and unit	=, ≠, <, >, +, -, *, /	30 minutes 42.5 kg

Data Transformation Inter-Scale

Size
1.96
1.75
1.55

Size
Tall
Medium
Small

Gender
Male
Female
Female

Gender
0
1
1

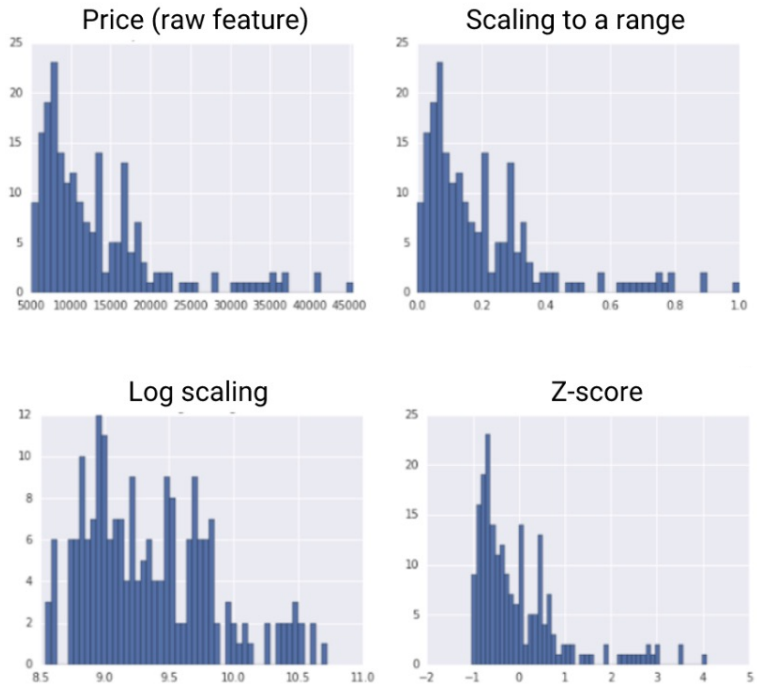
Motivation

- Convert numbers where math does not make sense into nominal
- Convert nominal values with an order into ordinal
- Some analysis methods require categorical or numerical data

Methods

- Discretization: Convert continuous numeric attributes into discrete categorical ones (binning)
- Scale augmentation: Convert categorical to numerical attributes

Data Transformation Intra-Scale



Motivation

- Attributes with large ranges overweigh attributes with small ranges (e.g. age vs. income)
- Some analysis algorithms require same scales / similar ranges
- Transform original data ranges into a unified range

Methods

- Standardization, Normalization

Data Enrichment

Quantity	Price	Revenue
250	1.70	425.00
83	29.90	2481.70
463	2.55	1180.65

Costs	Units	Cost per Unit
350	1000	0.35
750	5000	0.15
750	900	0.83

Motivation

- Create new meaningful variables from the available attributes
- Incorporate expert knowledge into data analysis

Methods

- Calculate quotients, products, differences, powers, etc. between different attributes
- Example: Cost per unit

Exercise:
Data Wrangling in Python

Q&A
How do you feel?

Descriptive Statistics

Descriptive Statistics

- There are many different types of descriptive statistics (mean, mode, range, variance, ...)
- Which one you need depends on…
  - What you want to achieve
  - The data you have
  - The number of attributes (variables) you look at

	Scale	Eigenschaften	Allowed Operations	Example
Categorical	Nominal	Classes of distinguishable objects	=, ≠	red, green, yellow
	Ordinal	Classes with a rank order relationship	=, ≠, <, >	small, medium, large
Numerical	Interval	Numbers without a natural 0 value	=, ≠, <, >, +, -	3°C, 4°C 2021-10-01
	Ratio	Numbers with a unique 0 and unit	=, ≠, <, >, +, -, *, /	30 minutes 42.5 kg

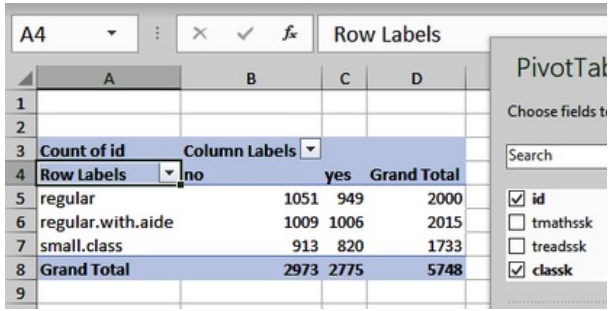
Categorical variables

→ Absolute and relative frequencies (% values).

→ Tools: pivot tables, bar and column charts

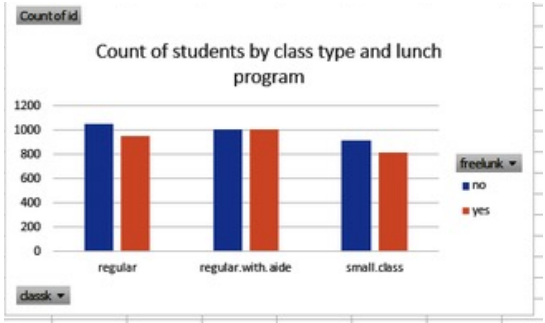
	A	B
1		
2		
3	**Row Labels**	**Count of id**
4	regular	2000
5	regular.with.aide	2015
6	small.class	1733
7	**Grand Total**	**5748**
8		

One-dimensional frequency table



	A	B	C	D
1				
2				
3	**Count of id**	**Column Labels**		
4	**Row Labels**	**no**	**yes**	**Grand Total**
5	regular	1051	949	2000
6	regular.with.aide	1009	1006	2015
7	small.class	913	820	1733
8	**Grand Total**	**2973**	**2775**	**5748**
9				

Two-dimensional frequency table



Bar charts

- Source: Mount, G., Advancing into Analytics (2021), O'Reilly

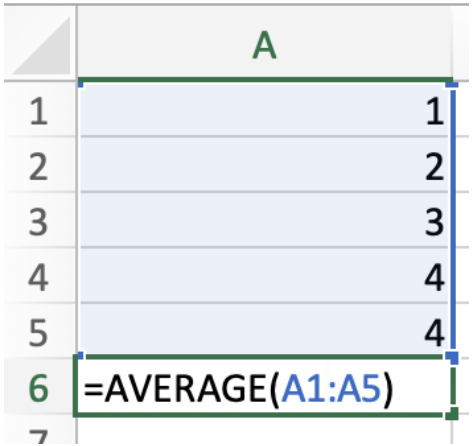
Numerical Variables

Measures of central tendency:

- **Mean ("average"):**

- The arithmetic mean divides the sum of all observations and by the number of observations.

- [1, 2, 3, 4, 4] → 2.8

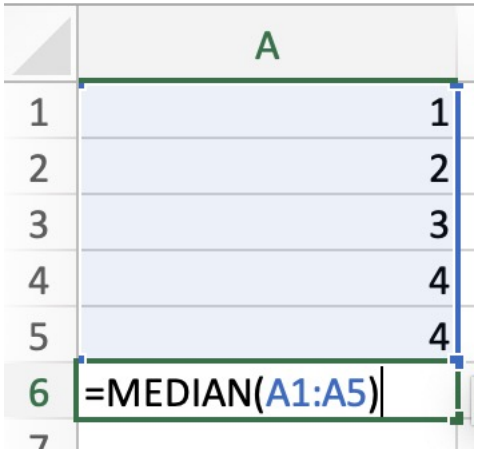


Numerical Variables

Measures of central tendency:

- **Median**

- Returns the value of the observation in the middle of the sorted data set. If the total number of observations is even, take the mean from the two middle values.
- [1, 2, 3, 4, 4] → 3



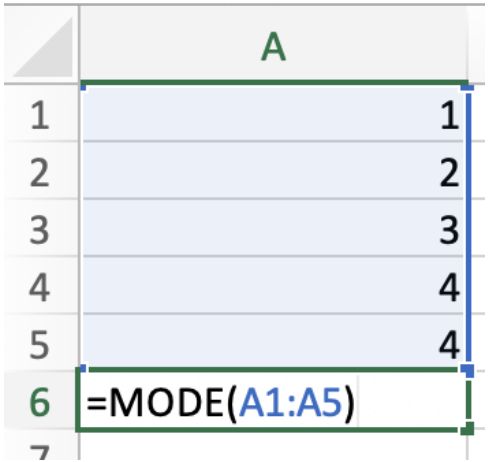
Numerical Variables

Measures of central tendency:

- **Mode**

- Returns the most frequent value of the observations. A variable can have one, several or no modal value.
- [1, 2, 3, 4, 4] → 4

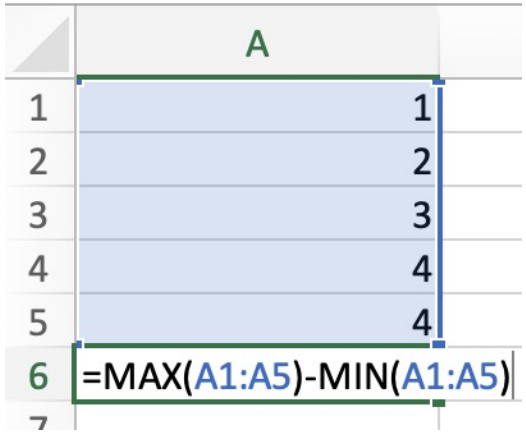
→ *Works also for categorical data!*



Numerical Variables

Measures of dispersion:

- **Range**
- Difference between the largest and smallest value of an observation.
- [1, 2, 3, 4, 4] → 3

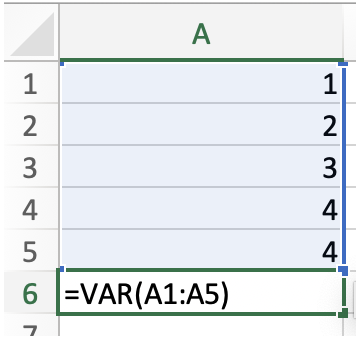


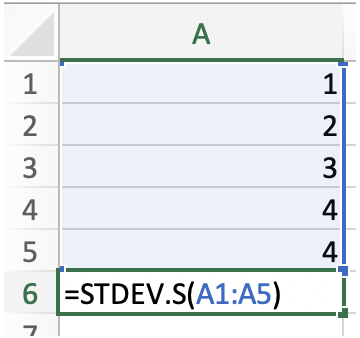
Numerical Variables

Measures of dispersion:

- **Variance** measures how much observations vary around the mean:
  - Calculate mean
  - Subtract mean from each observation (differences)
  - Sum the squares of all differences
  - Divide sum of squares by number of observations.
- [1, 2, 3, 4, 4] → 1,7

- **Standard deviation** is the root of the variance and represents the original scale unit.
- [1, 2, 3, 4, 4] → 1,3





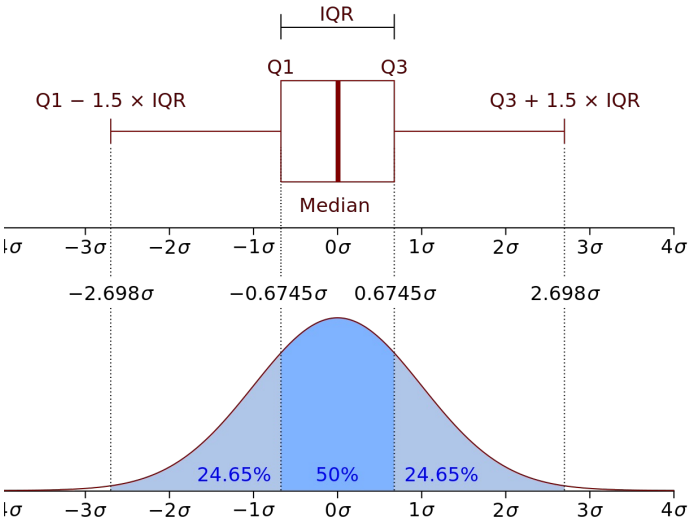
Numerical Variables

Measures of dispersion:

- **Interquartile range (IQR):**
  → Difference between the 25th and 75th percentile (= first and third **quartile***) i.e., IQR describes the middle 50% of observations.

  → Large IQR: The middle 50% of observations are spaced wide apart.

  → Advantage: Not affected by extreme values.



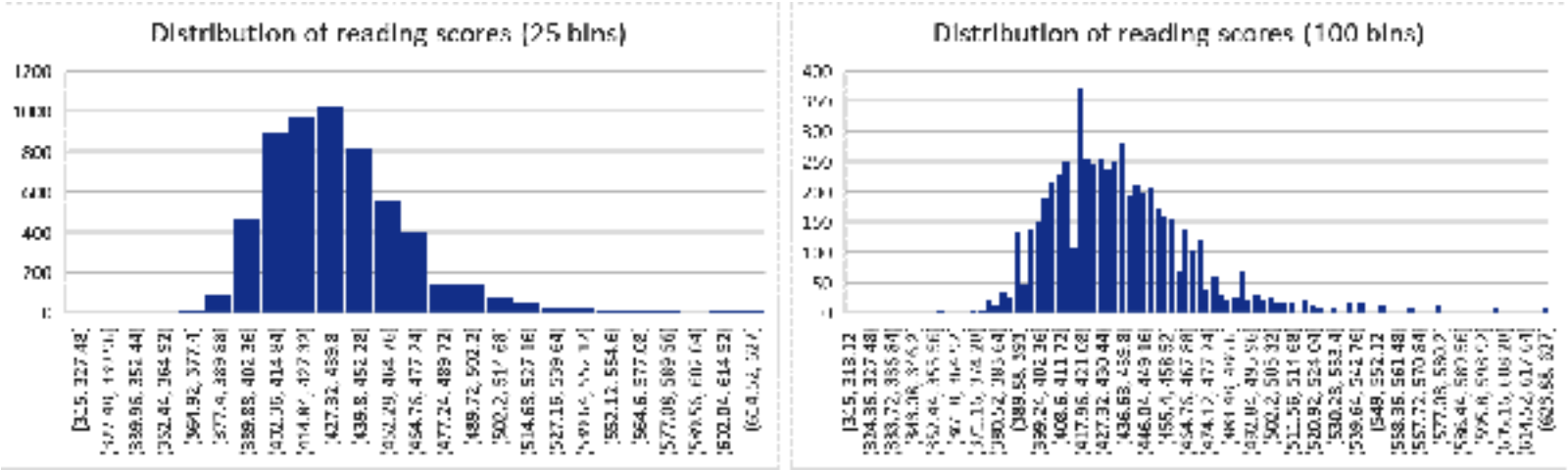
*Quartile: Divides the number of data points into four parts (quarters) of (more-or-less) equal size

- Source: Wikipedia

Numerical Variables

Histogram (one numerical variable)

→ approximate representation of the data distribution using “bins” (”buckets”)

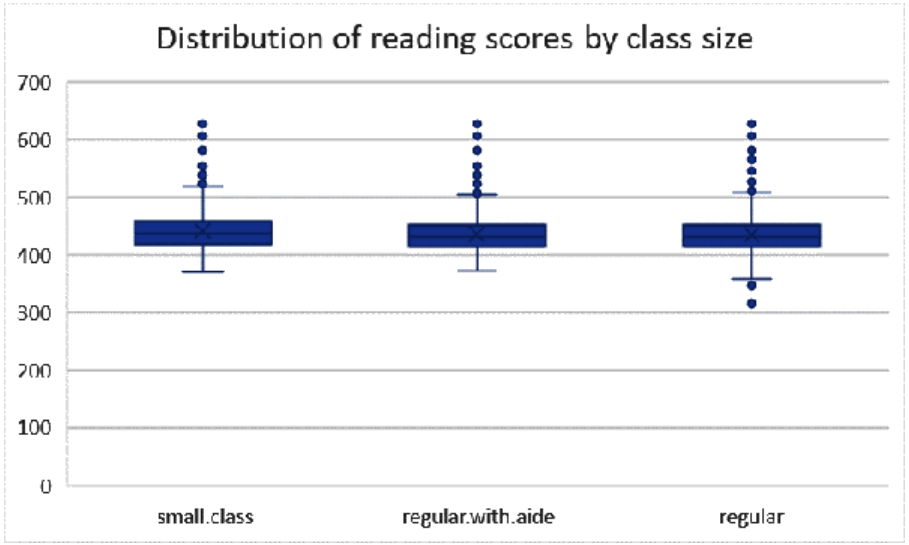


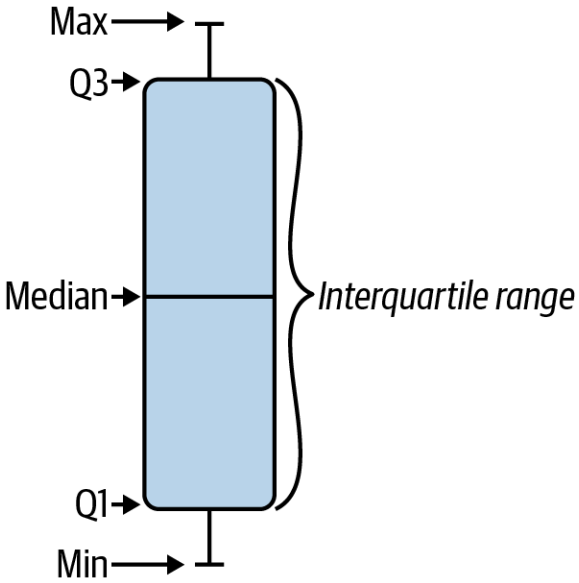
- Source: Mount, G., Advancing into Analytics (2021), O’Reilly

Numerical Variables

Box plots (one or many numerical variables)

→ Show spread, skewness and outliers of numerical data through their quartiles.



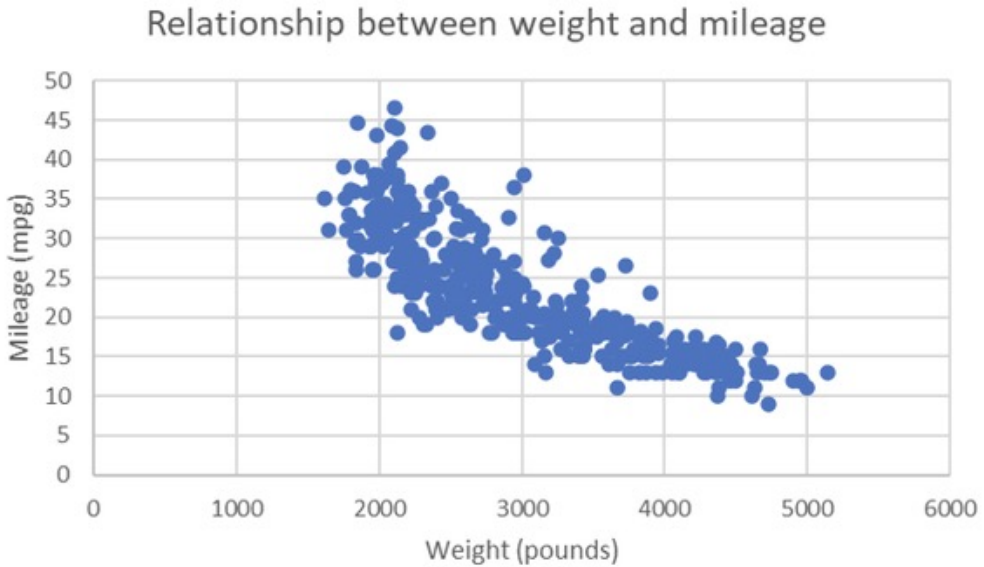


- Source: Mount, G., Advancing into Analytics (2021), O'Reilly

Numerical Variables

Scatter plots

→ Show relationship between two numerical variables

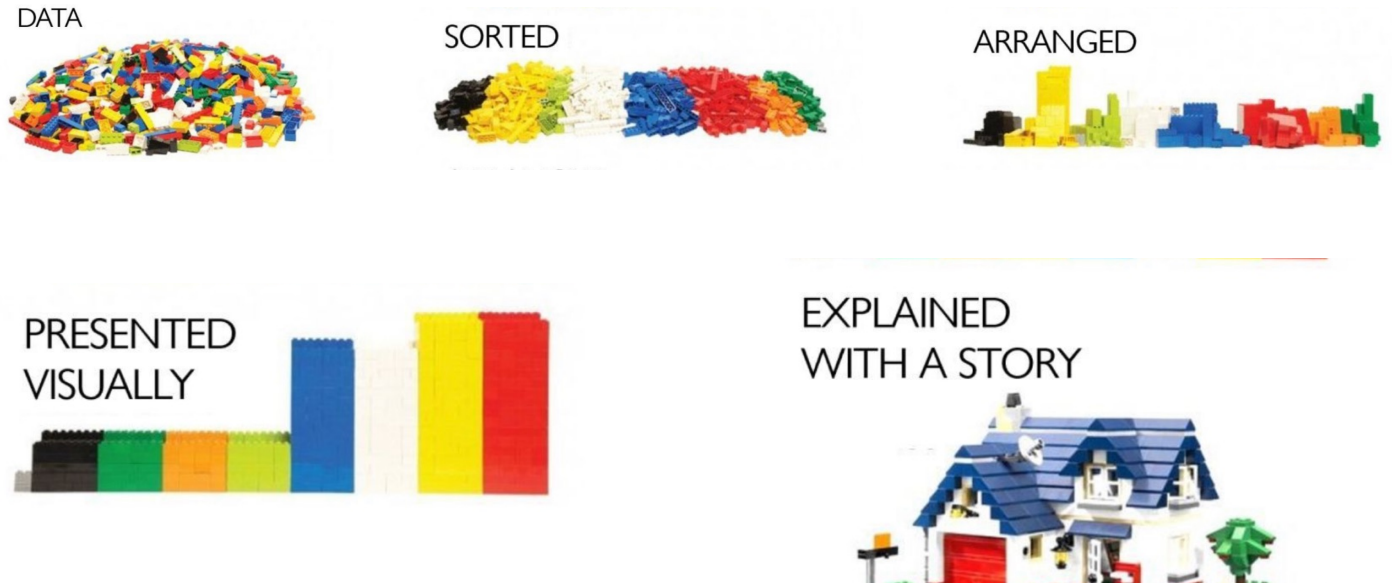


- Source: Mount, G., Advancing into Analytics (2021), O'Reilly

Exercise:
Descriptive Statistics With Python

Data Visualizations

Introduction to Data Visualization



Why Data Visualization?

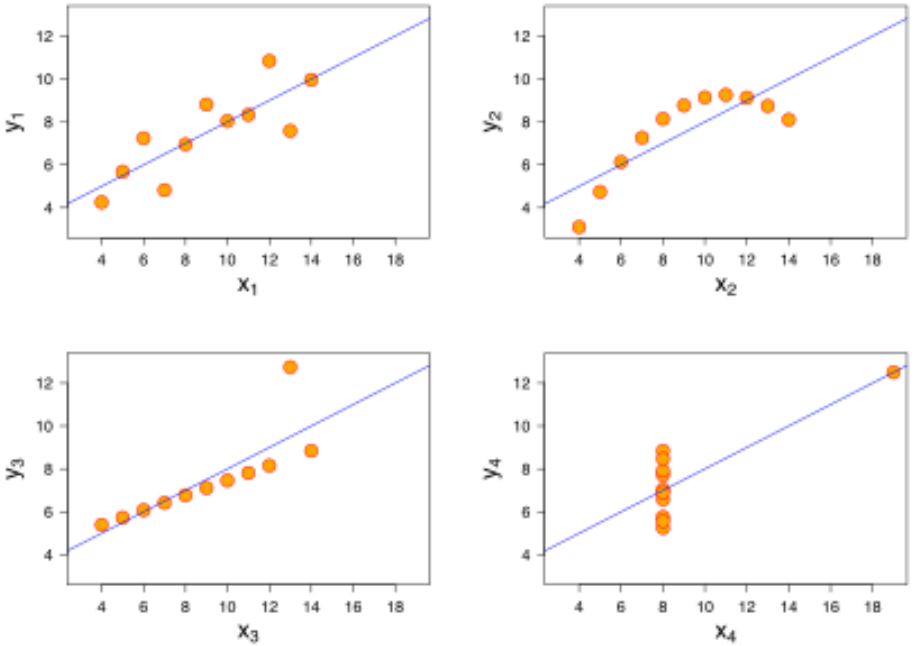
Visualizing data is essential to make it interpretable for humans.

→ **Anscombe's quartet**

Anscombe's quartet

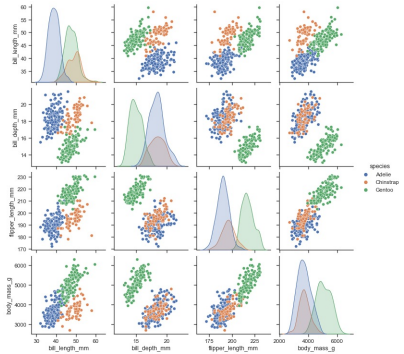
I		II		III		IV	
x	*y*	*x*	*y*	*x*	*y*	*x*	*y*
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	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

Mean of x	9
Mean of y	7.5
Variance of x	11
Variance of y	4.125
Correlation	0.816
Regression	y = 3 + 0.5x
R2	0.67



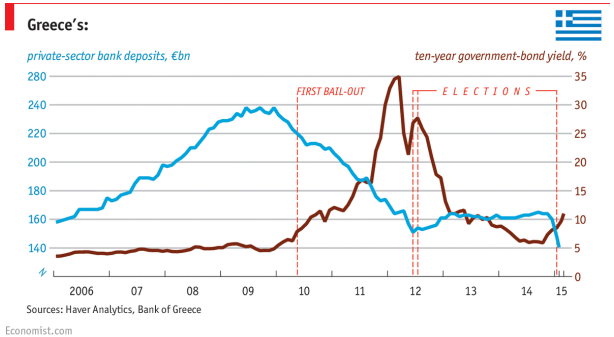
- Source: Wikipedia

Two Types of Data Visualization



Exploratory

Goal: Get familiar with the data. “Turning over 100 rocks to find perhaps 1 or 2 precious gemstones.”



Explanatory

Goal: Show something specific to an audience, e.g., those “1 or 2 precious gemstones”.

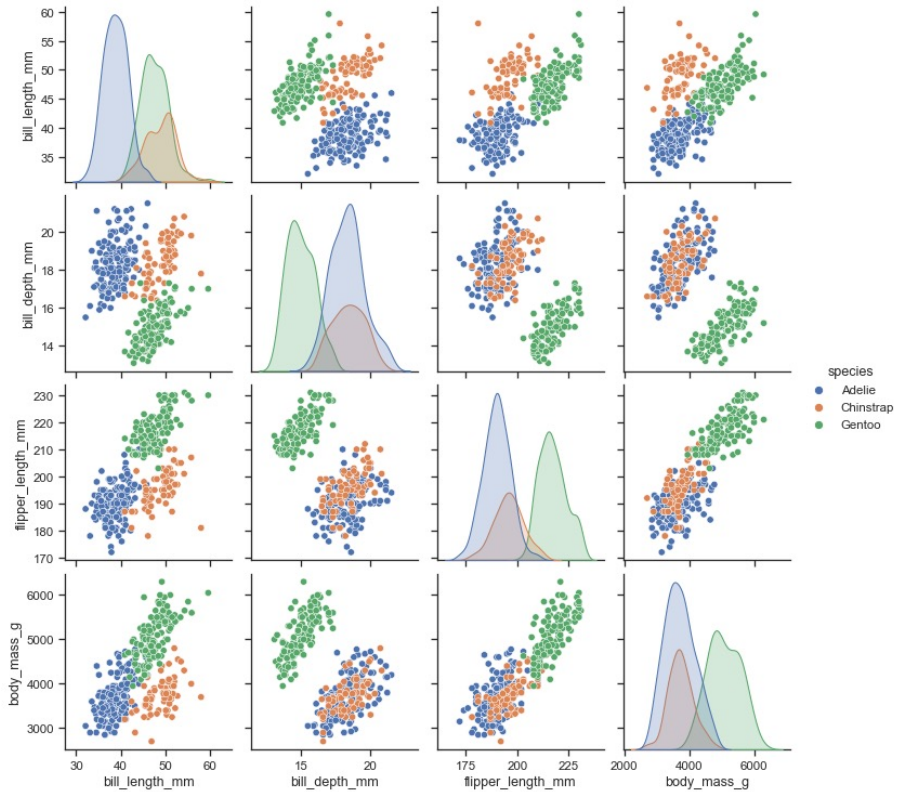
- Source: https://www.storytellingwithdata.com/blog/2014/04/exploratory-vs-explanatory-analysis

Data Visualization

Exploration & Confirmation

Finding the story the data tells or verifying a story with data.

- → "Reader driven”
- → Weak order
- → "Light messaging”
- → Interactive and flexible
- → Exploratory
- → ”A lot"

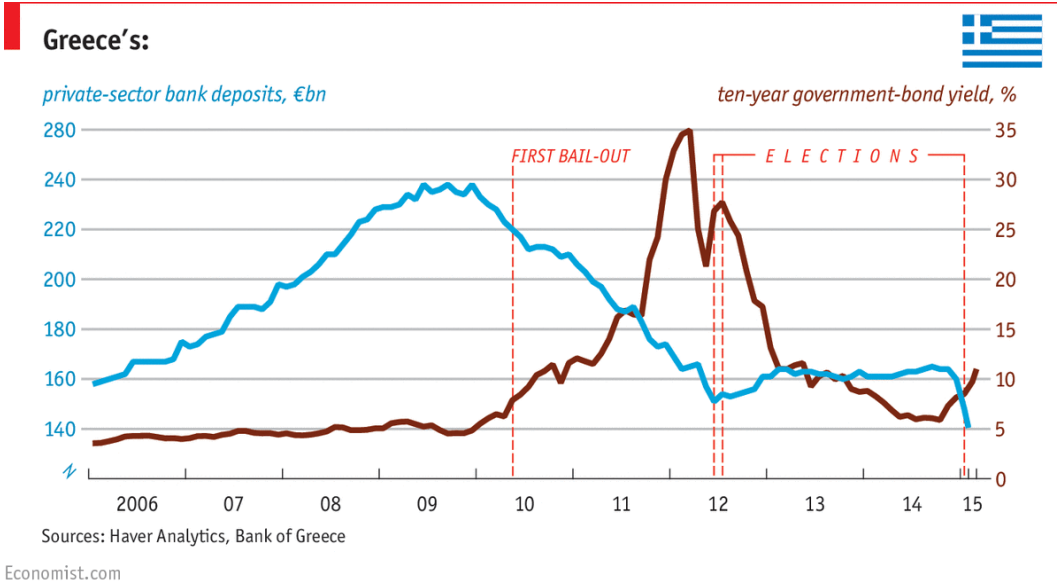


Data Visualization

Explanation & Reporting

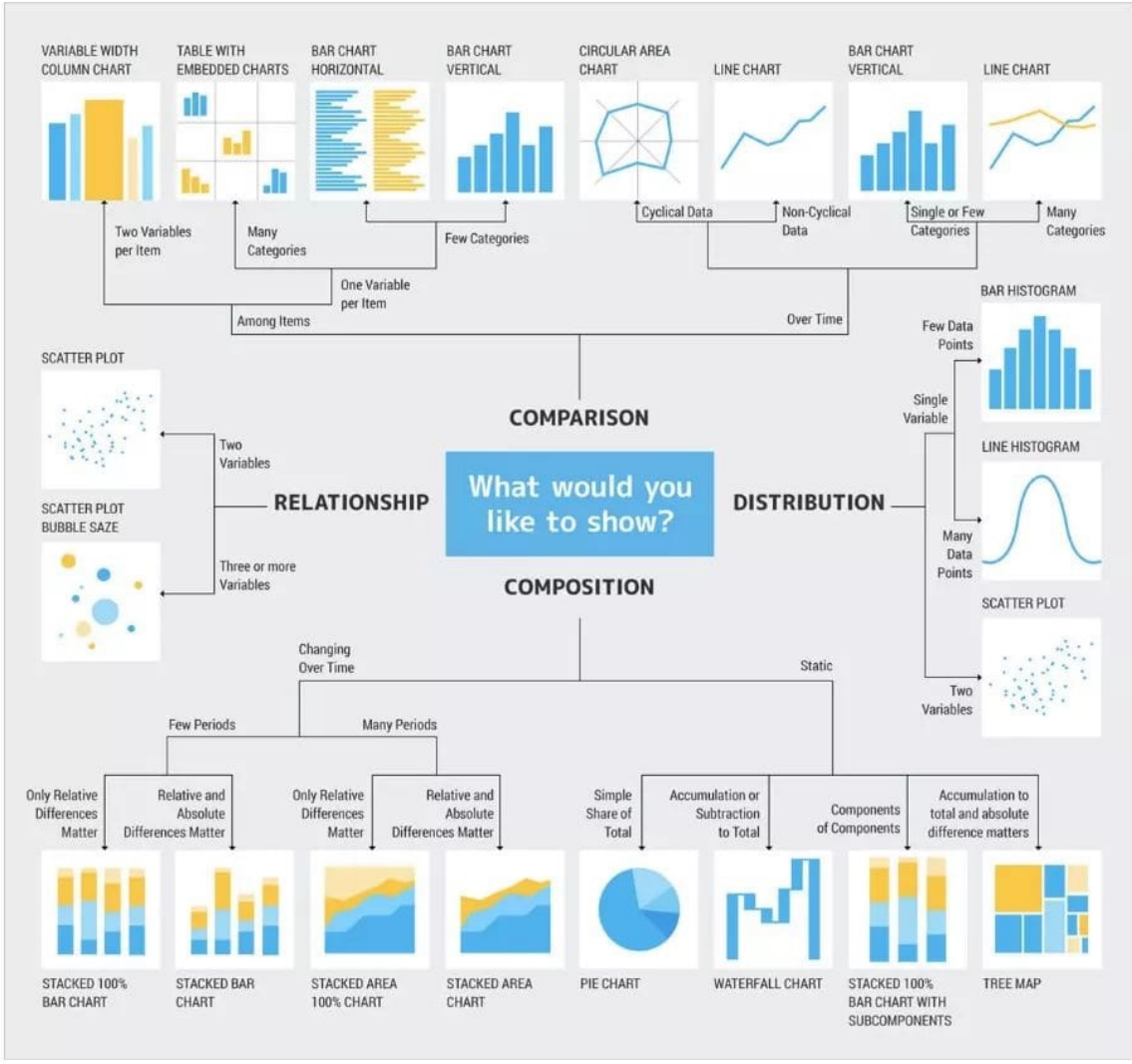
Reporting the story, the data tells.

- → "Author driven”
- → Strict order (linear)
- → "Heavy messaging”
- → Continuous narrative
- → Limited interactivity
- → "A few"



Data Visualization

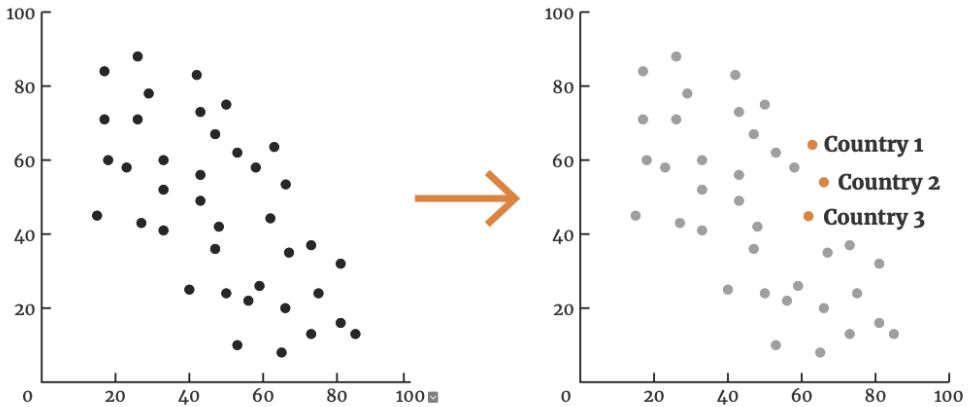
- Choose the visualization according to your data and goal
- Source: https://www.kaggle.com/getting-started/160583



Principles of Data Visualization

Show the data

Data should be presented as clearly as possible. However, this does not mean that all data must be shown - in fact, many charts show too much.



- Source: PolicyViz

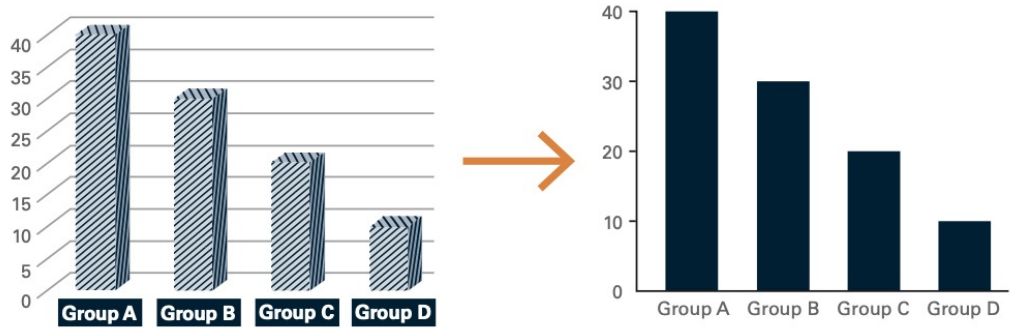
Principles of Data Visualization

Keep it simple

Cluttered diagrams, unnecessary or distracting visual elements, reduce effectiveness.

Clutter is created by e.g.:

- dark or heavy grid lines
- unnecessary labels or text,
- unnecessary symbols or images, shading and gradients, and
- unnecessary dimensions.



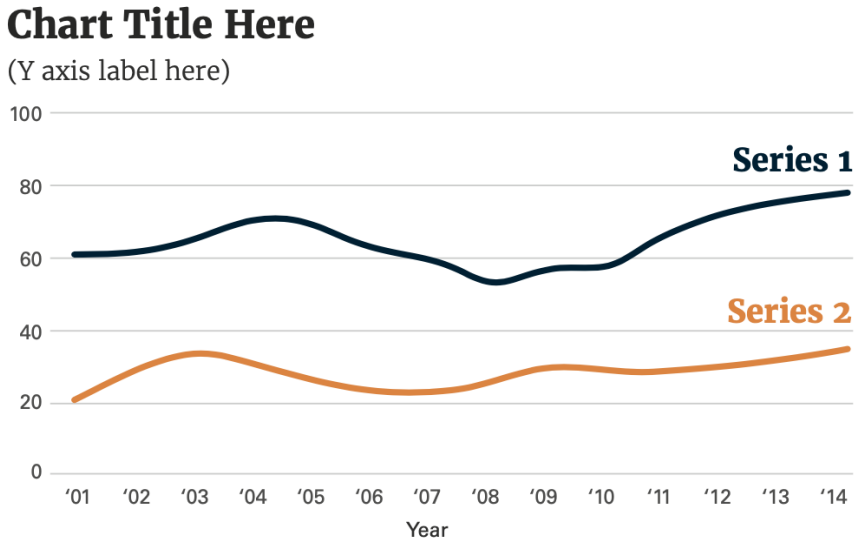
- Source: PolicyViz

Principles of Data Visualization

Integrate text and graph

Visualizations should be designed to complement text (language) while providing enough information to stand on their own.

Example: legends that explain a line, bar, or point should be integrated close to the object.



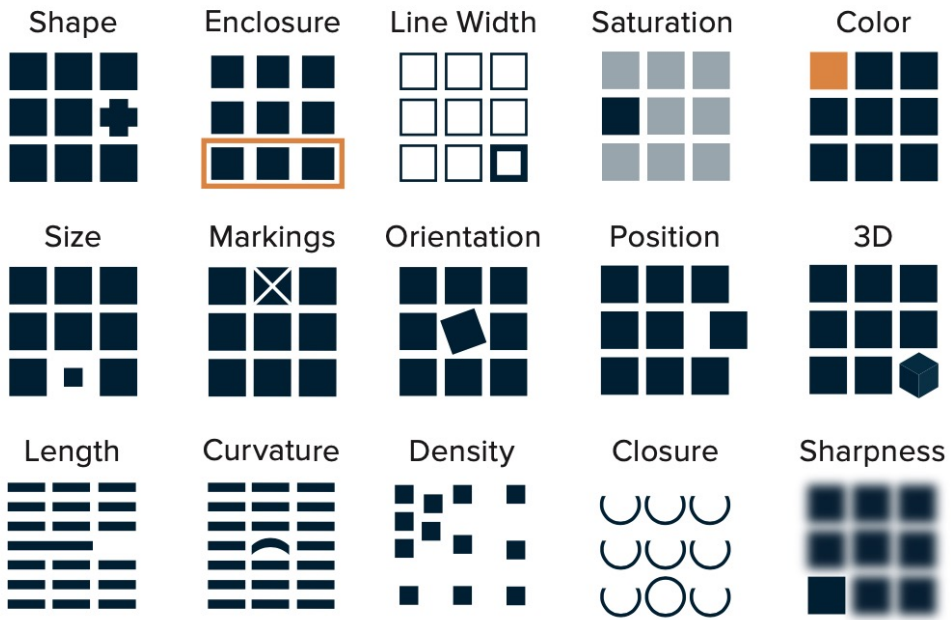
- Source: PolicyViz

Principles of Data Visualization

Use pre-attentive perception

Pre-attentive perception = subconscious processes in the brain that process information before giving attention.

Effective visualizations take advantage of this effect.

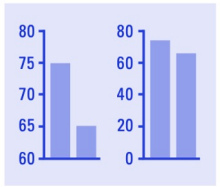


- Source: PolicyViz

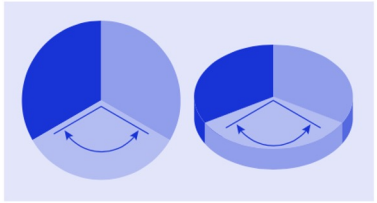
Principles of Data Visualization

Always keep your audience in mind - whether they need a short, written report, a more detailed paper, or an online data exploration tool.

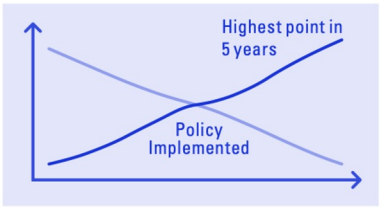
Use pie charts with caution: up to three categories are still practicable, beyond that it is difficult to distinguish ratios. Bar charts or tree diagrams are better suited for this purpose.



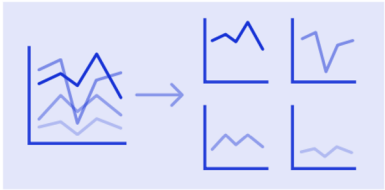
Bar and column charts should **always start at zero**. Otherwise, they overly emphasize the differences between values. Use relative / percentage changes to show small absolute value changes instead.



Avoid 3D representations unless you really have a third variable. 3D representation distorts the perception of the data and should therefore be avoided.



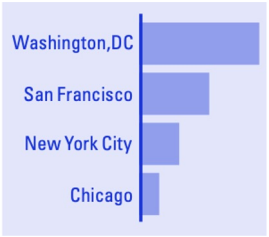
Adding **annotations and explanatory text** to help readers understand how to interpret or - if necessary - use the visualization.



Breaking a complicated chart into **smaller parts** can be an effective way to visualize your data.

- Source: PolicyViz

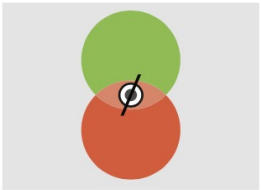
Principles of Data Visualization

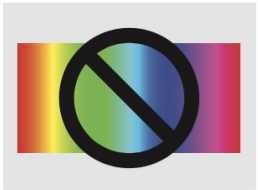


Make labels easy to read: Rotate bar and column charts so that labels are horizontal, if necessary. Labels should be clear, concise, and easy for readers to understand.

Use maps with care: Always make sure that it's really the geographic point you are trying to represent. Column and bar charts, are often better suited for comparisons between regions.

Fonts and colors

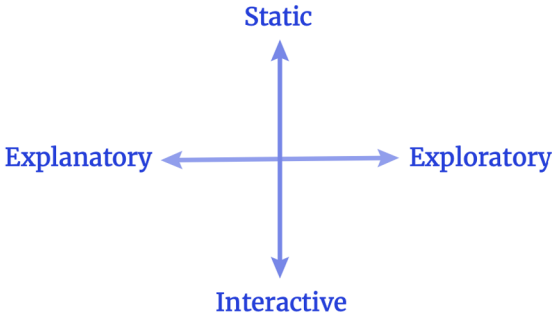




Use custom colors and fonts to attract more attention. Consider color blindness - about 10% of people are affected in some way. Choose a color palette that supports your message. Avoid rainbow color palettes with no logical order.

- Source: PolicyViz

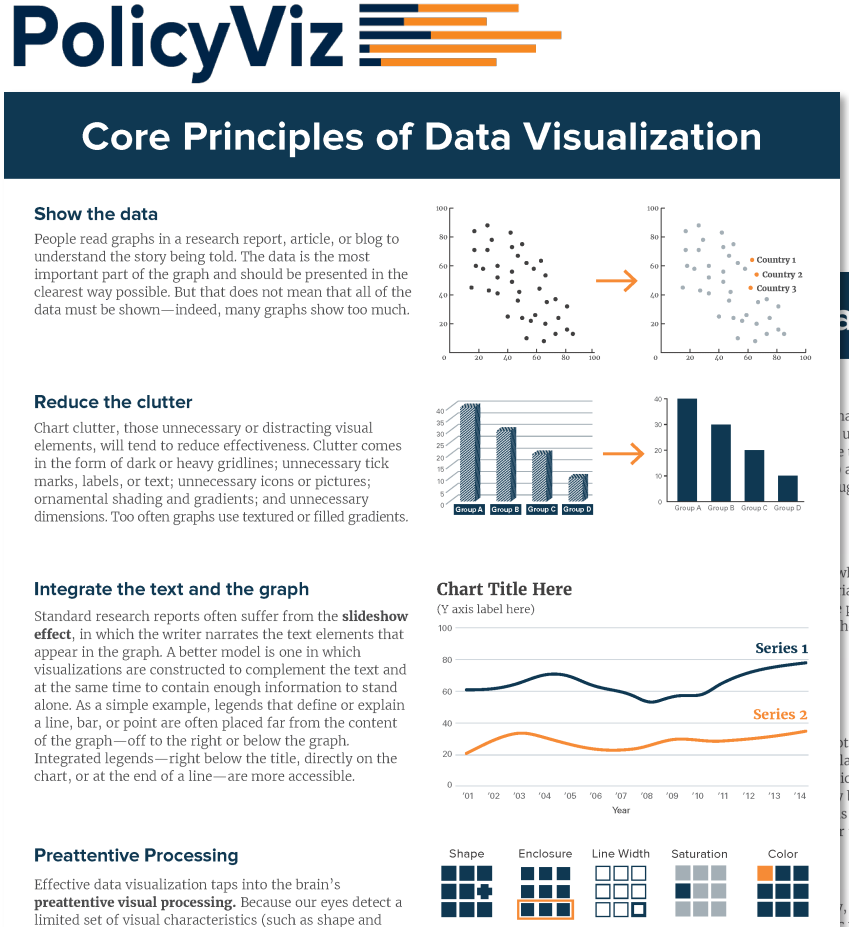
Balance form and function as needed



Get the cheat sheets

https://policyviz.com/2018/08/07/dataviz-cheatsheet/

- Source: PolicyViz

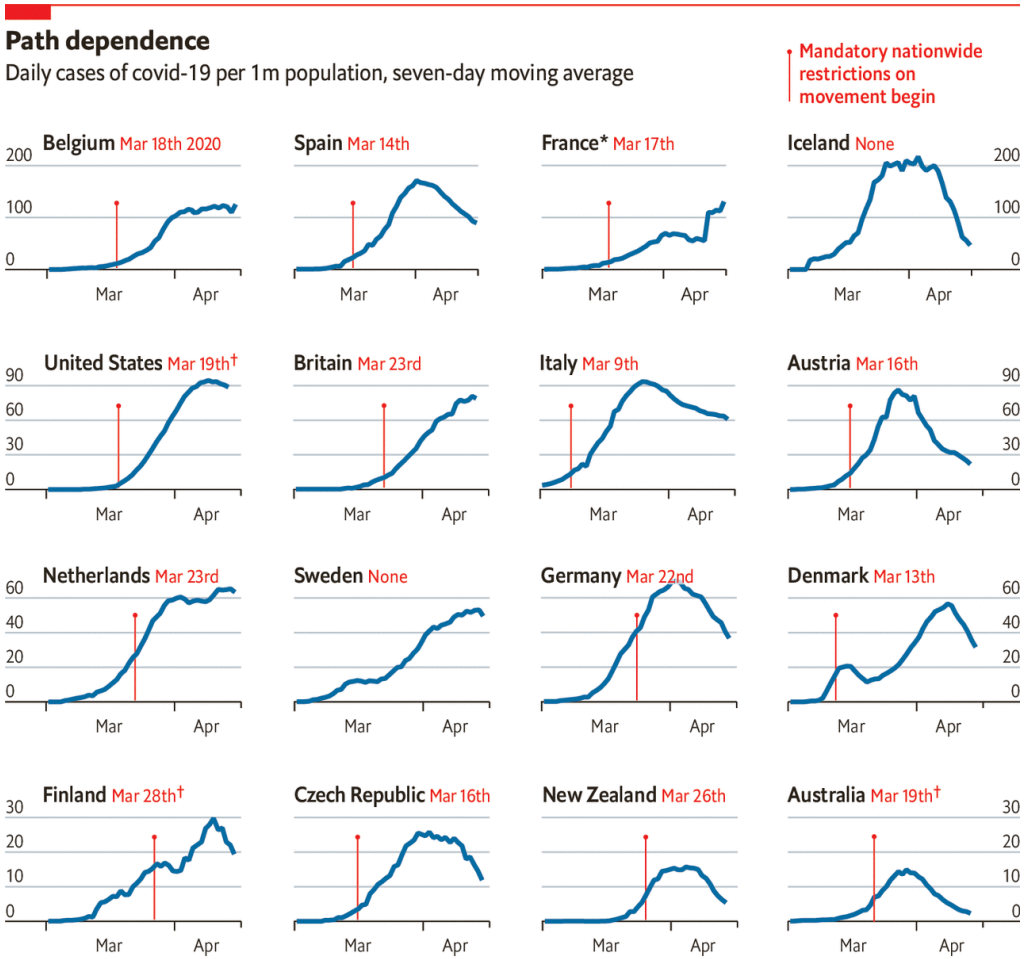


Data Visualization

Examples

Break up complex charts into multiple smaller pieces

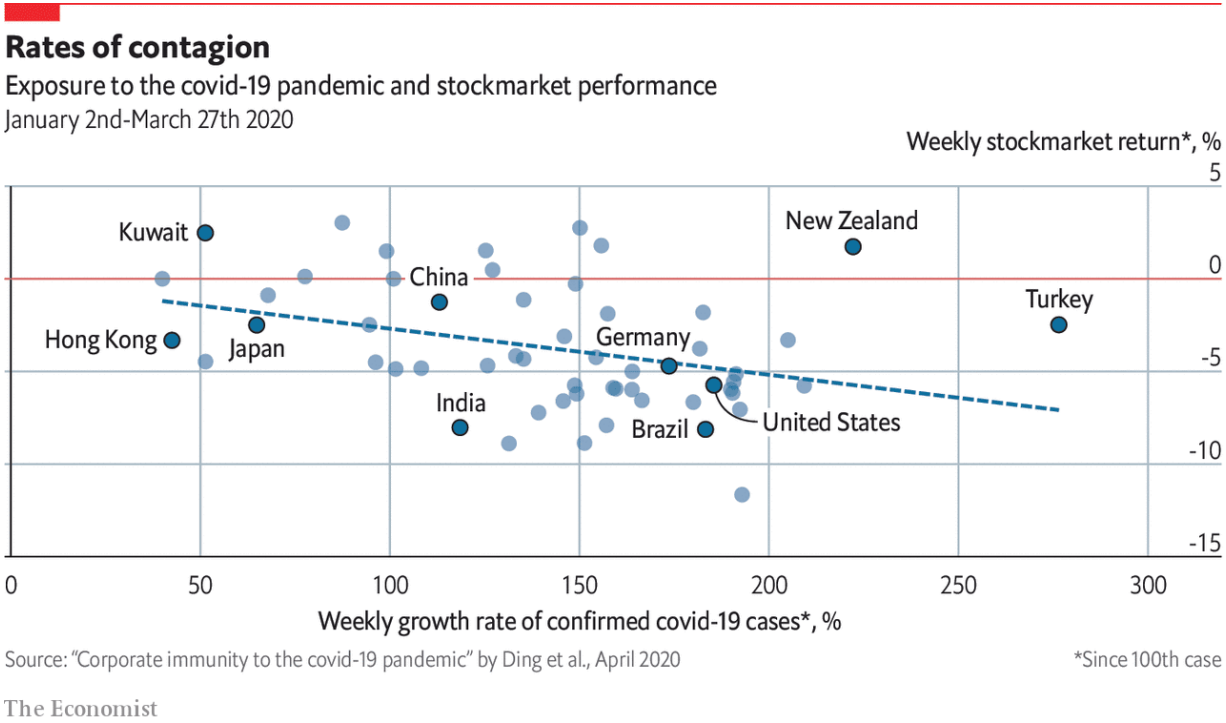
- Source: PolicyViz



Data Visualization

Examples

Highlight selected points only, legend close to data points

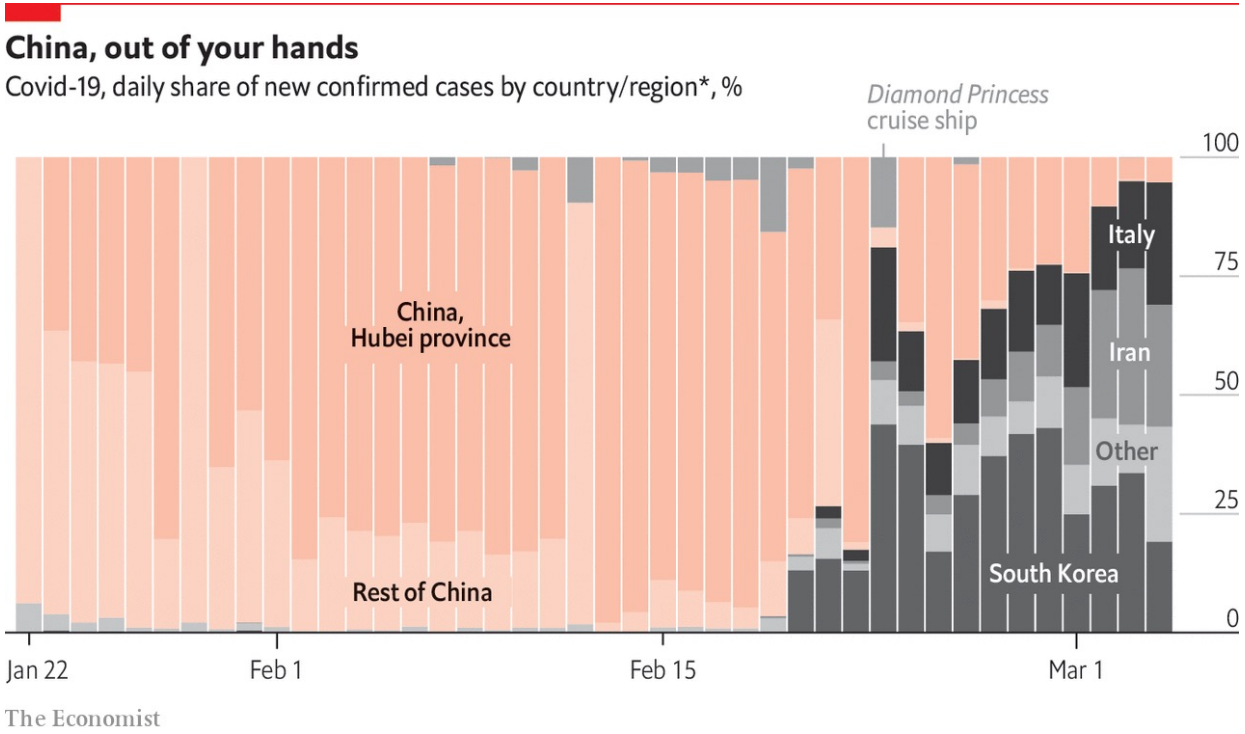


- Source: PolicyViz

Data Visualization

Examples

Pre-attentive perception with annotations in graph and explanative text



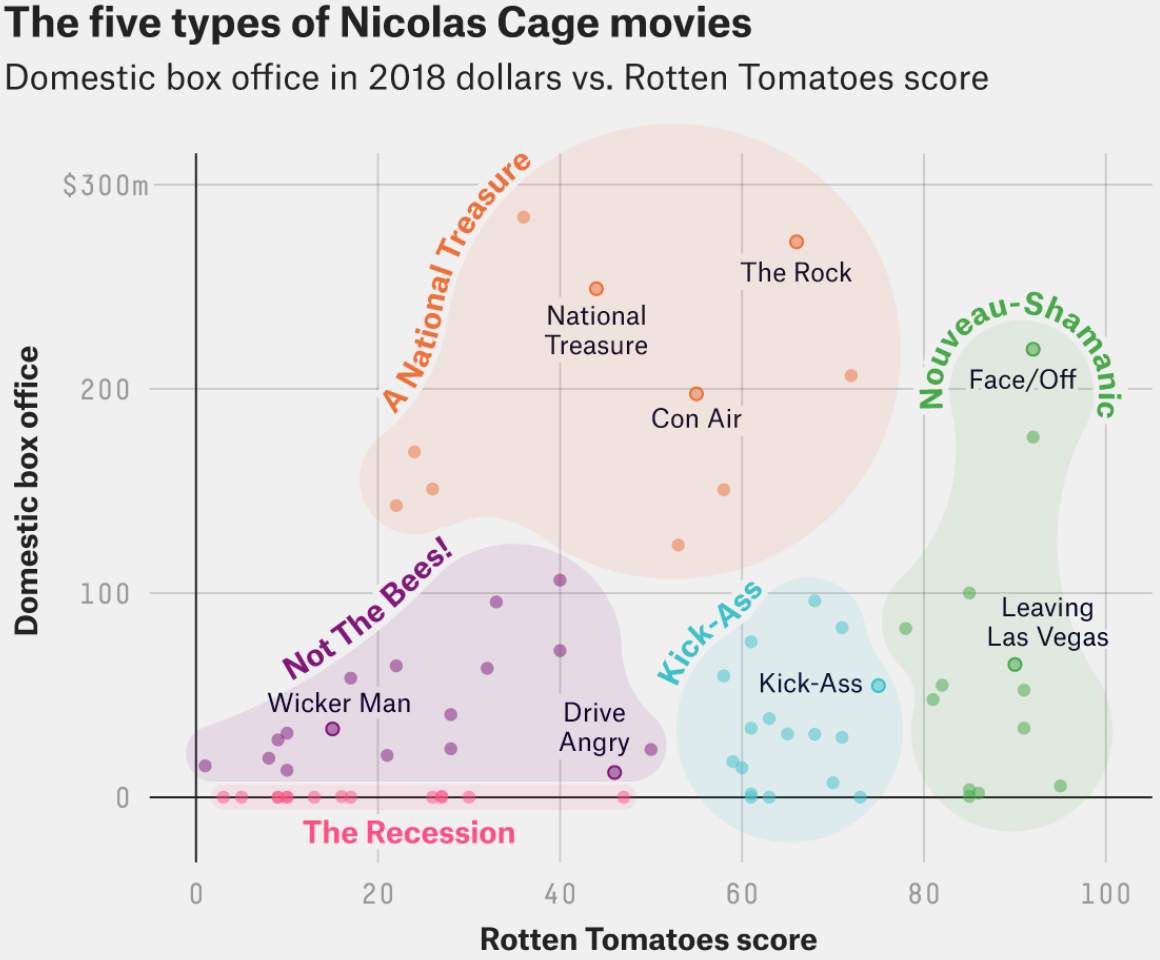
- Source: PolicyViz

Data Visualization

Examples

- Annotations
- Colors
- Selected data

- Source: PolicyViz

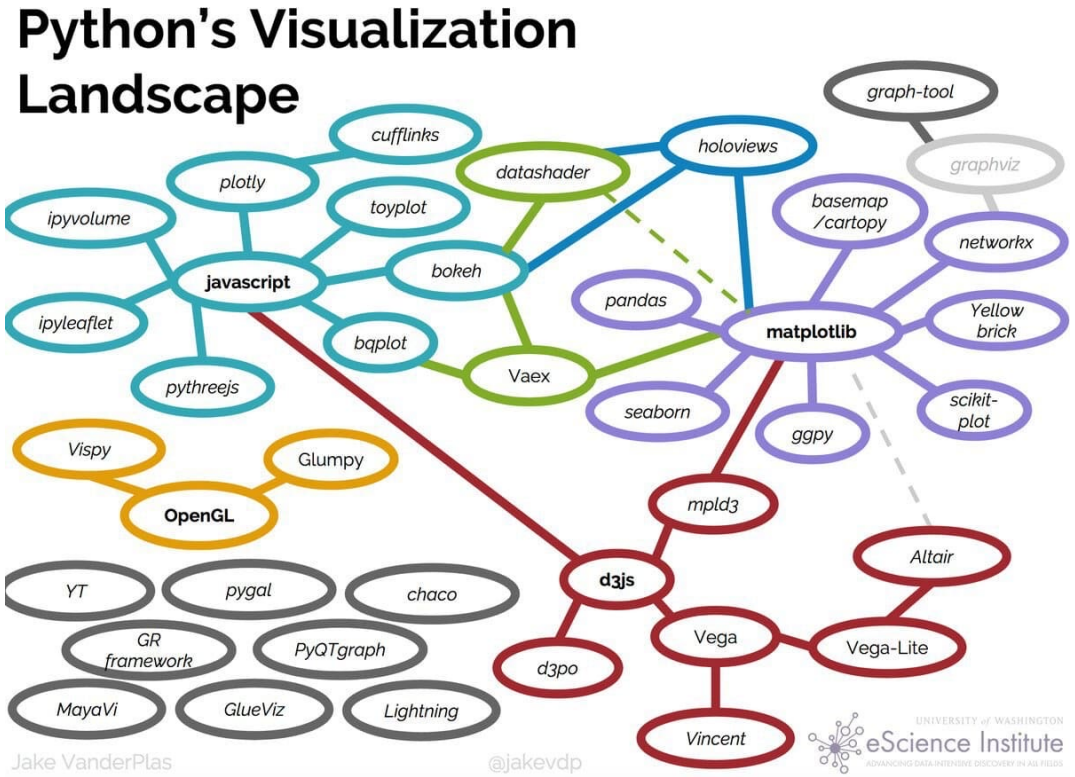


Data Visualization in Python

There's no shortage of data viz frameworks in Python!

The most popular include…

- Matplotlib
- Seaborn
- Plotly
- Bokeh
- …

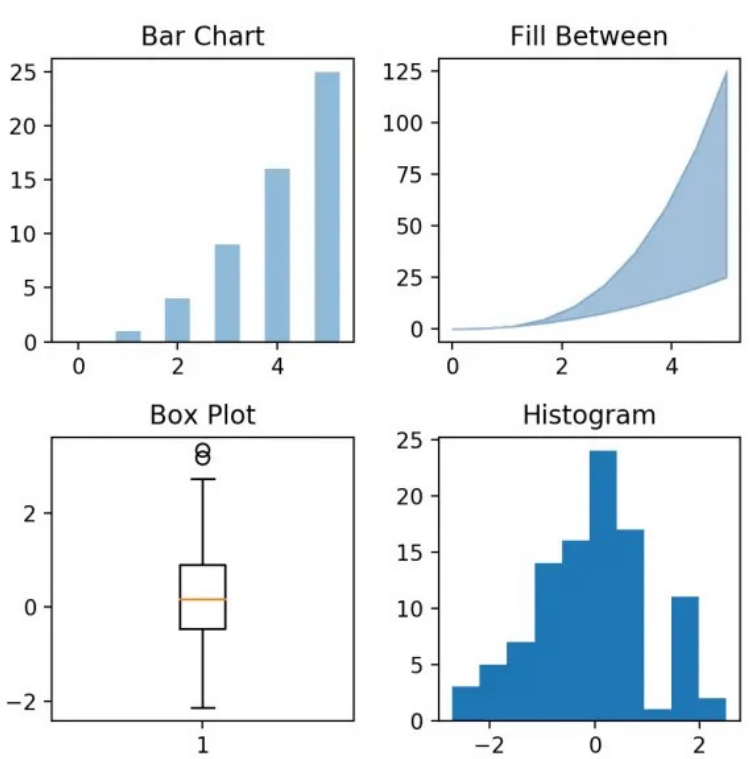


- Source: https://www.anaconda.com/blog/python-data-visualization-2018-why-so-many-libraries

Data Visualization in Python

Matplotlib

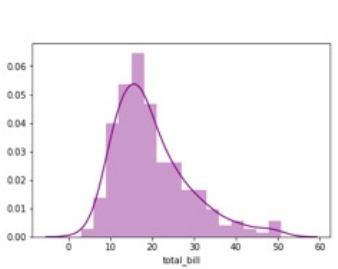
- The “classic”
- Static plots
- Integrated in pandas
- Essential customizations



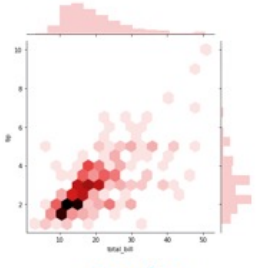
Data Visualization in Python

Seaborn

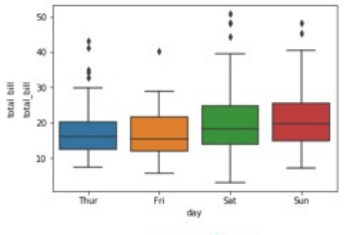
- Built on top of matplotlib
- More plot types
- Even more customization



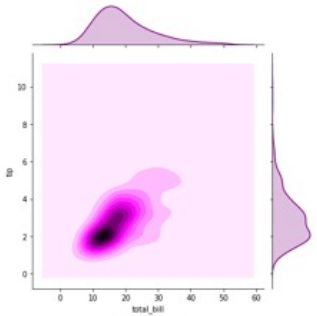
distplot



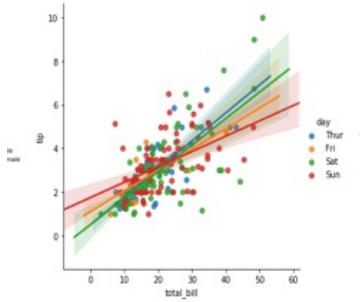
Hexplots



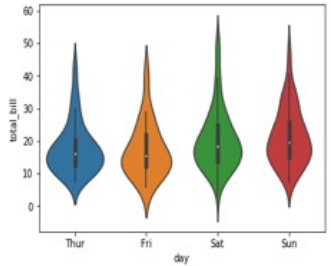
Boxplots



KDE Plot



LM Plots

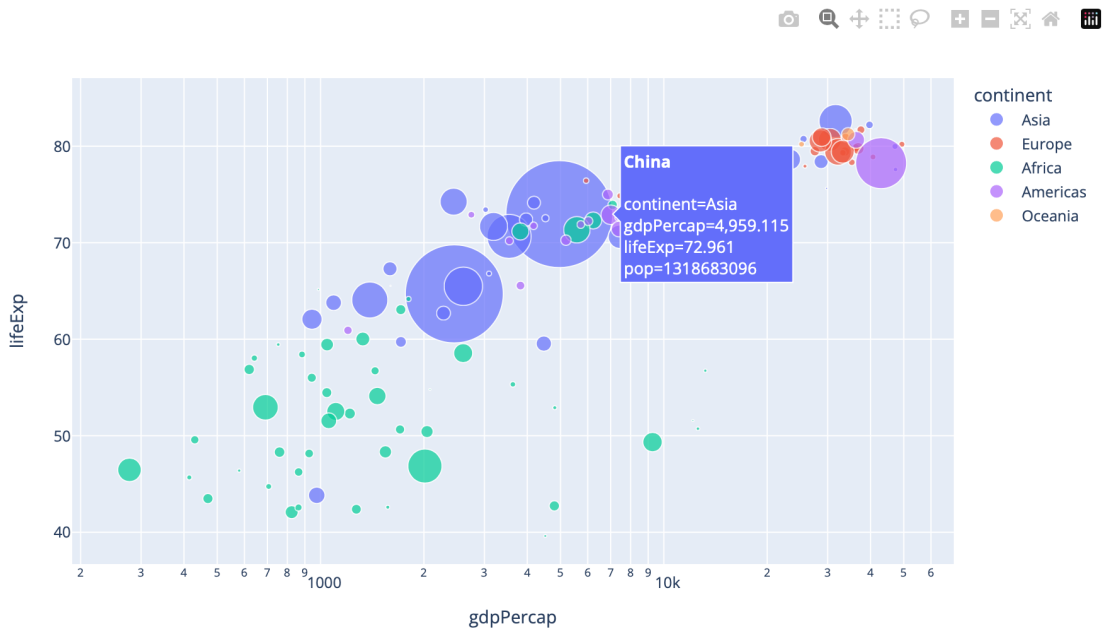


Violin Plots

Data Visualization in Python

Plotly

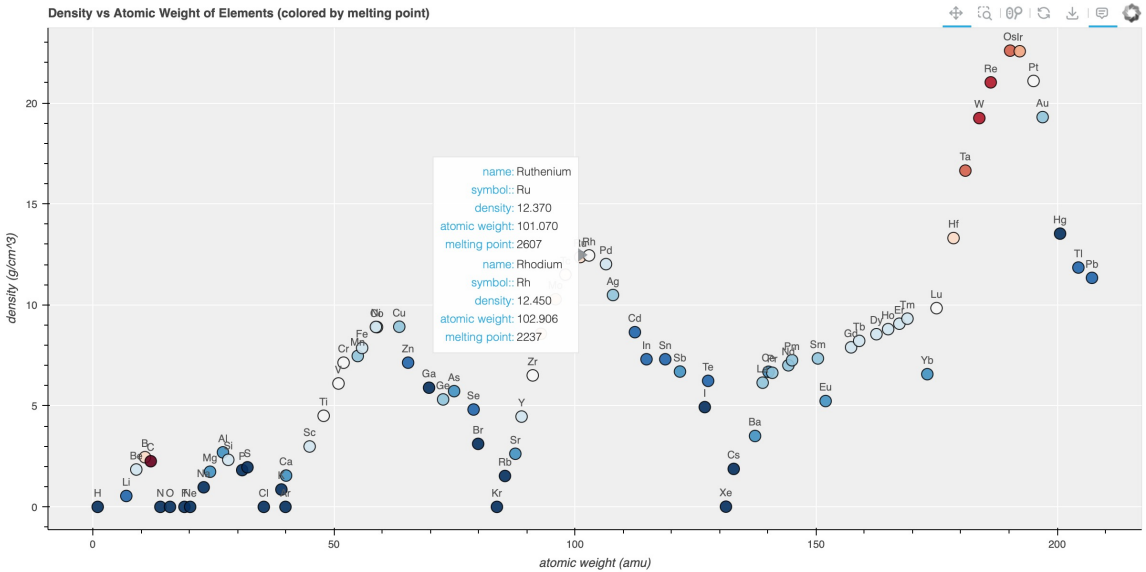
- JavaScript library
- High interactivity
- Many plot types



Data Visualization in Python

Bokeh

- JavaScript library
- High interactivity
- Easy to learn



Exercise: Explore Data Visualizations in Python

Q&A

How do you feel?

Wrap-up

What did we learn today?

- ❑ Understand the essence of descriptive analytics
- ❑ Understand the concept of exploratory data analysis (EDA)
- ❑ Choose the right data model for data analysis (tidy)
- ❑ Transform data with Python
- ❑ Load data with and without SQL
- ❑ Write / Export data with Python
- ❑ Conduct descriptive statistics
- ❑ Conduct summary statistics & Anscombe's quartet
- ❑ Understand Data visualization techniques
- ❑ Use visualization frameworks in Python

Outlook for next week

Week 4: Diagnostic Business Analytics with Python

- Introduction to Diagnostic Analytics
- Essential Diagnostic Analytics Techniques
- Correlation vs. causation
- Principle of the 5-Whys and root-cause analysis
- Rule mining techniques and association rules
- Support, confidence, lift
- Introduction to customer segmentation
- RFM analysis
- Clustering techniques

Thank you!

Keep in touch:

linkedin.com/in/tobias-zwingmann
tobias@rapyd.ai