



DATA ENGINEERING



香港城市大學
City University of Hong Kong

Course Review Notes

(for reference only)

CS5481 Data Engineering
Instructor: Linqi Song

L1: introduction

- 1. Overview of the data engineering ecosystem
- 2. Types of data
- 3. Languages and tools for data professionals

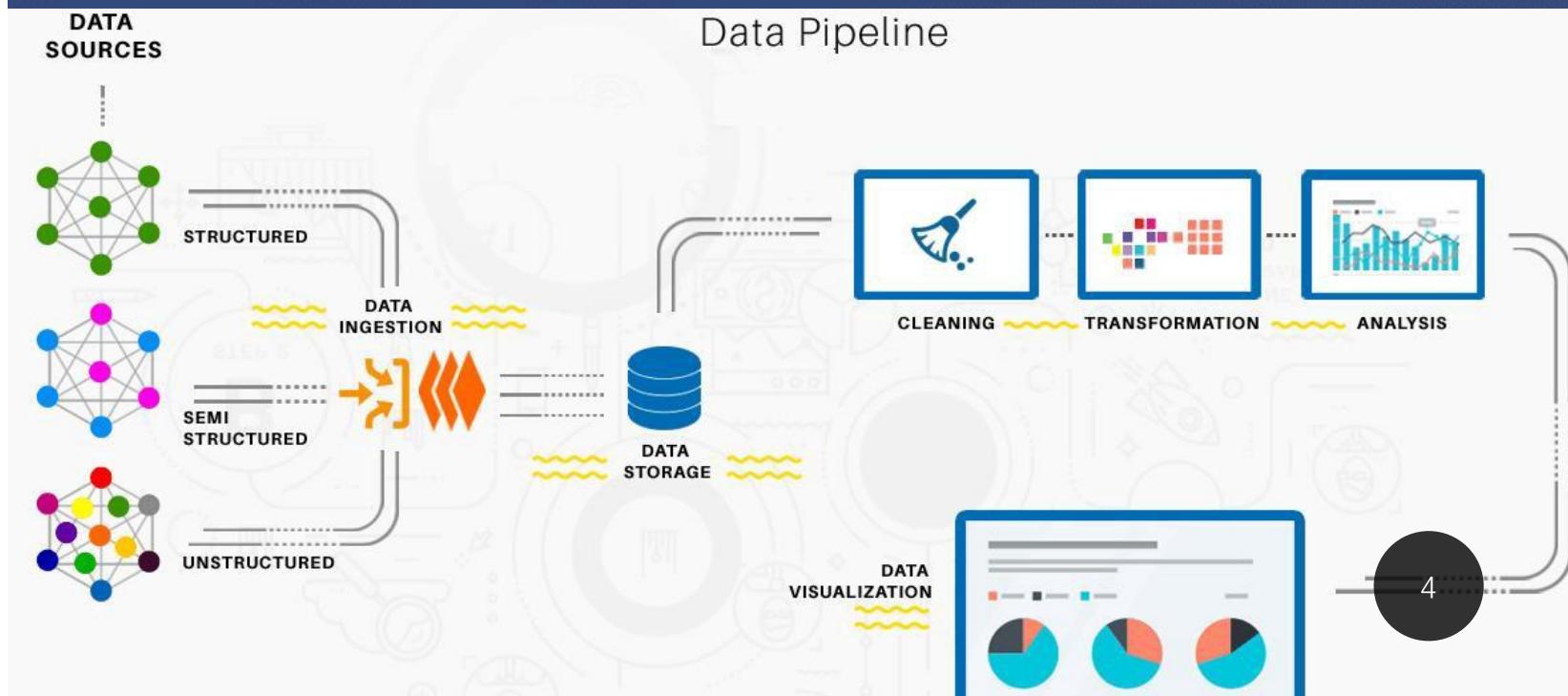
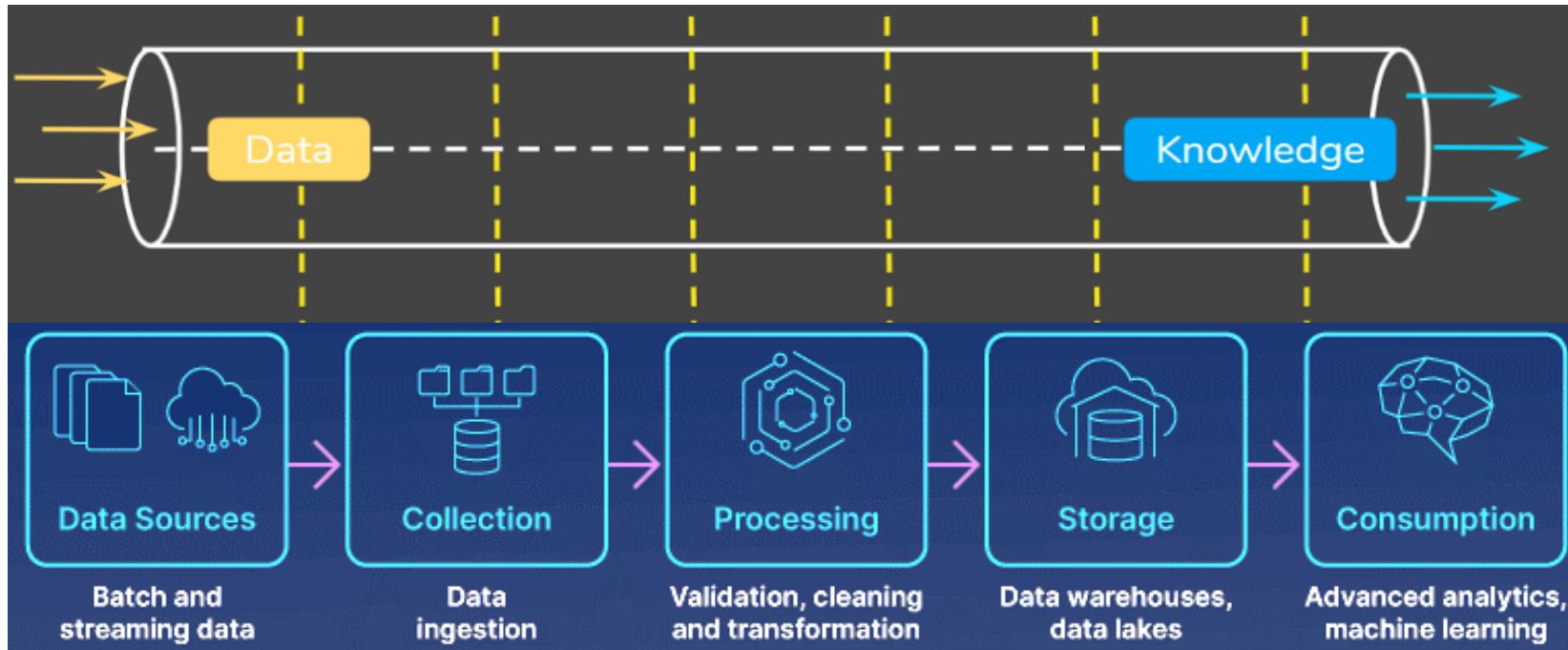
WHAT IS Data Engineering?



- **Big data** is changing the way we do business and creating a need for data engineers who can collect and manage large quantities of data.
- **Data engineering** is the practice of **designing and building systems** for collecting, storing, and analyzing data at scale.

Data engineering pipeline

- **The Goal of Data Engineering** is to provide organized, standard data flow to enable data-driven models such as machine learning models, data analysis.



Types of data

Unstructured data

The university has 5600 students. John's ID is number 1, he is 18 years old and already holds a B.Sc. degree. David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
  <Student ID="1">
    <Name>John</Name>
    <Age>18</Age>
    <Degree>B.Sc.</Degree>
  </Student>
  <Student ID="2">
    <Name>David</Name>
    <Age>31</Age>
    <Degree>Ph.D. </Degree>
  </Student>
  ...
</University>
```

Structured data

| ID | Name | Age | Degree |
|----|---------|-----|--------|
| 1 | John | 18 | B.Sc. |
| 2 | David | 31 | Ph.D. |
| 3 | Robert | 51 | Ph.D. |
| 4 | Rick | 26 | M.Sc. |
| 5 | Michael | 19 | B.Sc. |

Languages for data professionals



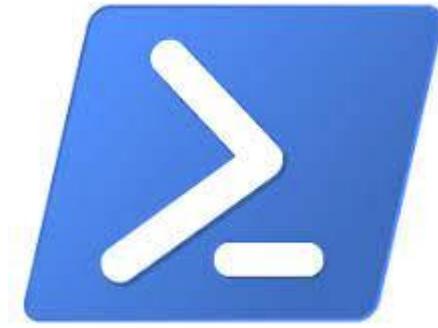
Query Languages

For example, SQL for querying and manipulating data



Programming Languages

For example, Python for developing data applications



Shell and Scripting Languages

For repetitive and time-consuming operational tasks

L2: data acquisition

- 1. Data sources
- 2. Web scraping
- 3. From web scraping to web crawling

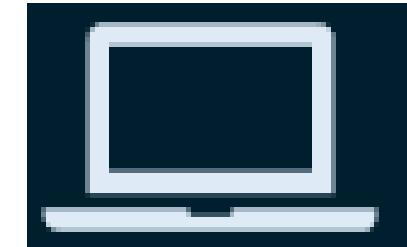
Sources of data



Relational
databases



Flat files and XML
datasets



APIs and web
services

Web scraping

- The **construction of an agent** to download, parse, and organize data from the web in an automated manner
- Extract relevant data from unstructured sources on the Internet
- Also known as screen scraping, web harvesting, and web data extraction
- Can extract text, contact information, images, videos, product items, etc.



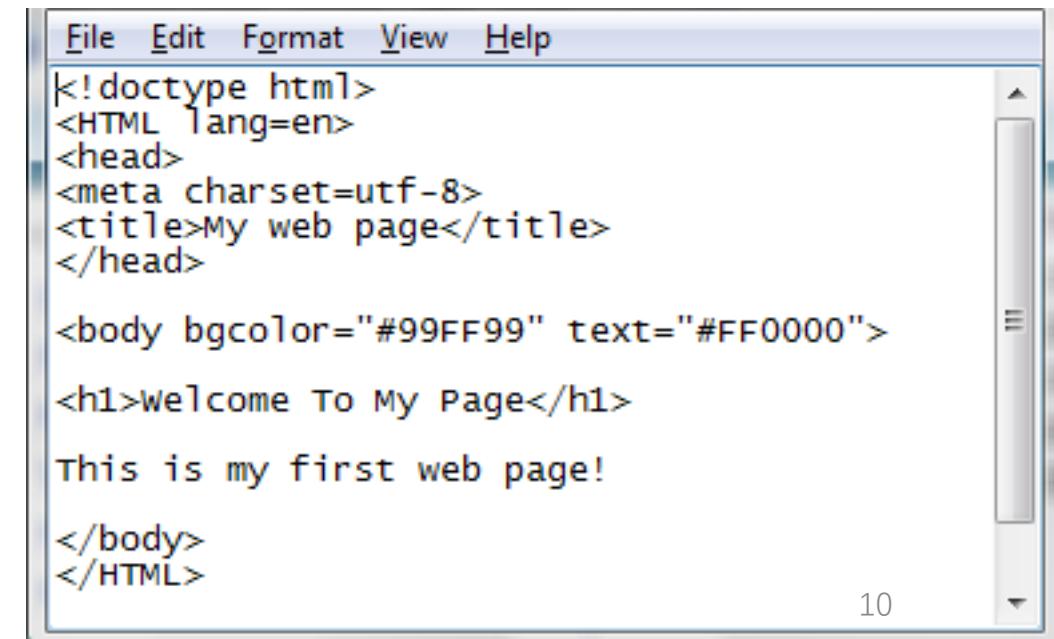
HTML file to scrape

- HTML is a standard **markup language** for **creating web pages**.
- HTML provides the building blocks to provide **structure and formatting** to documents.
- Python 'requests' library could get the html content from a webpage.

```
import requests

url = 'https://en.wikipedia.org/w/index.php' + \
      '?title=List_of_Game_of_Thrones_episodes&oldid=802553687'

r = requests.get(url)
print(r.text)
```



The image shows a screenshot of a Windows Notepad window. The window has a menu bar with File, Edit, Format, View, and Help. The main content area contains the following HTML code:

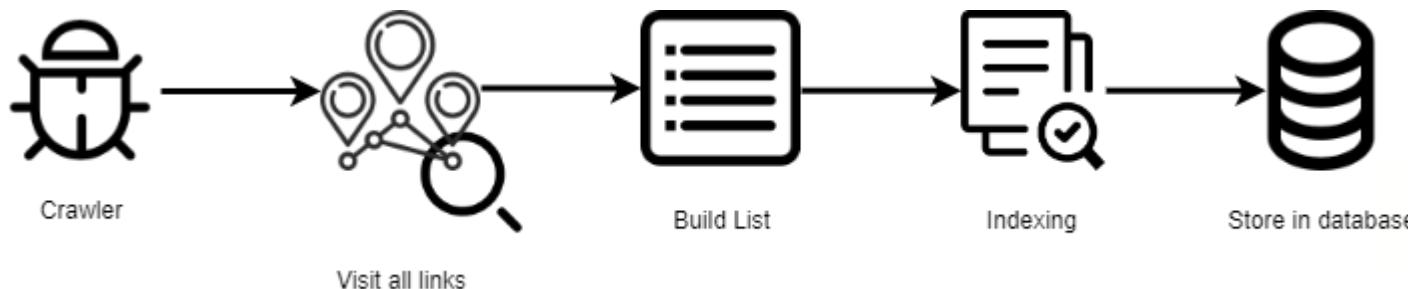
```
<!doctype html>
<HTML lang=en>
<head>
<meta charset=utf-8>
<title>My web page</title>
</head>

<body bgcolor="#99FF99" text="#FF0000">
<h1>welcome To My Page</h1>
This is my first web page!

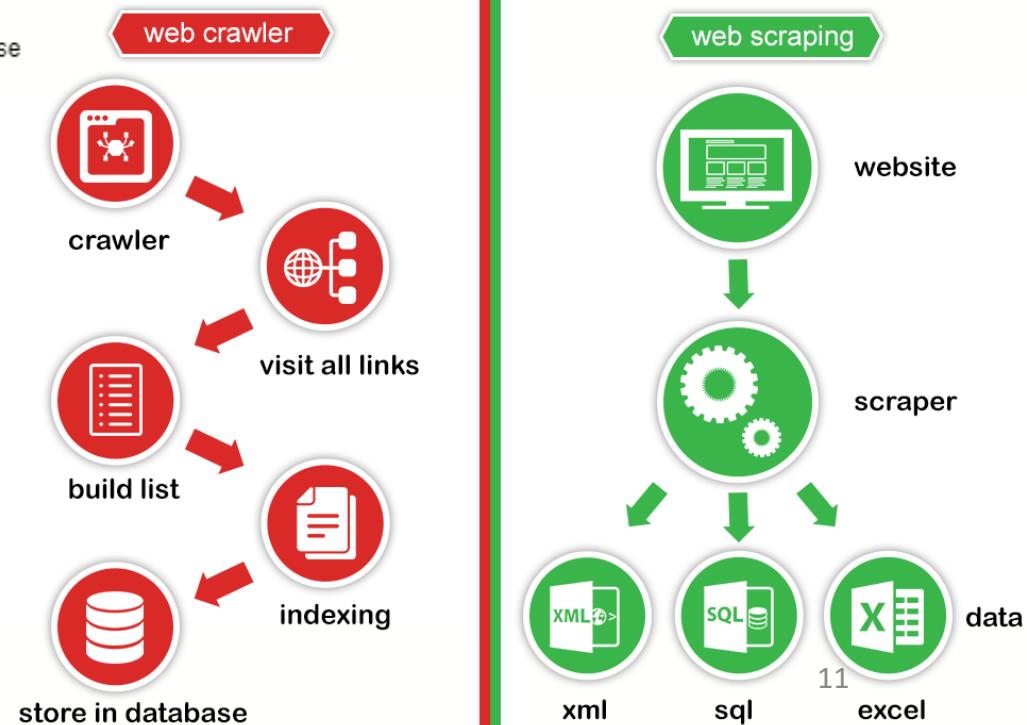
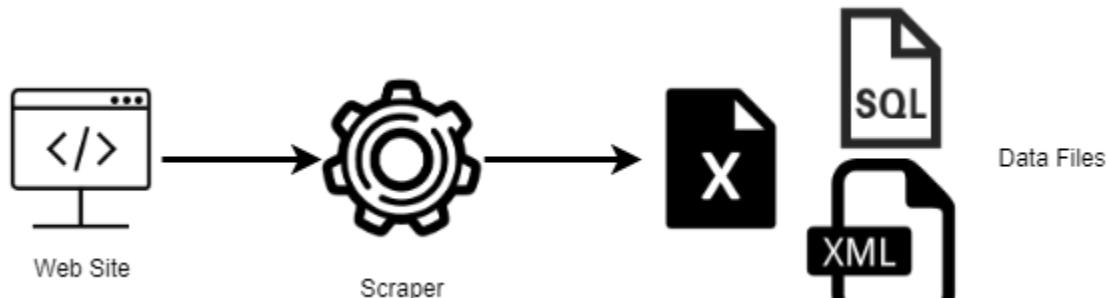
</body>
</HTML>
```

From web scraping to web crawling

Web Crawling: Using tools to read, copy and store the content of the websites for archiving or indexing purposes. Crawling usually deals with a network of webpages



Web Scraping: Extracting a large amount of specific data usually from a single webpage or a single website



L3: data preprocessing

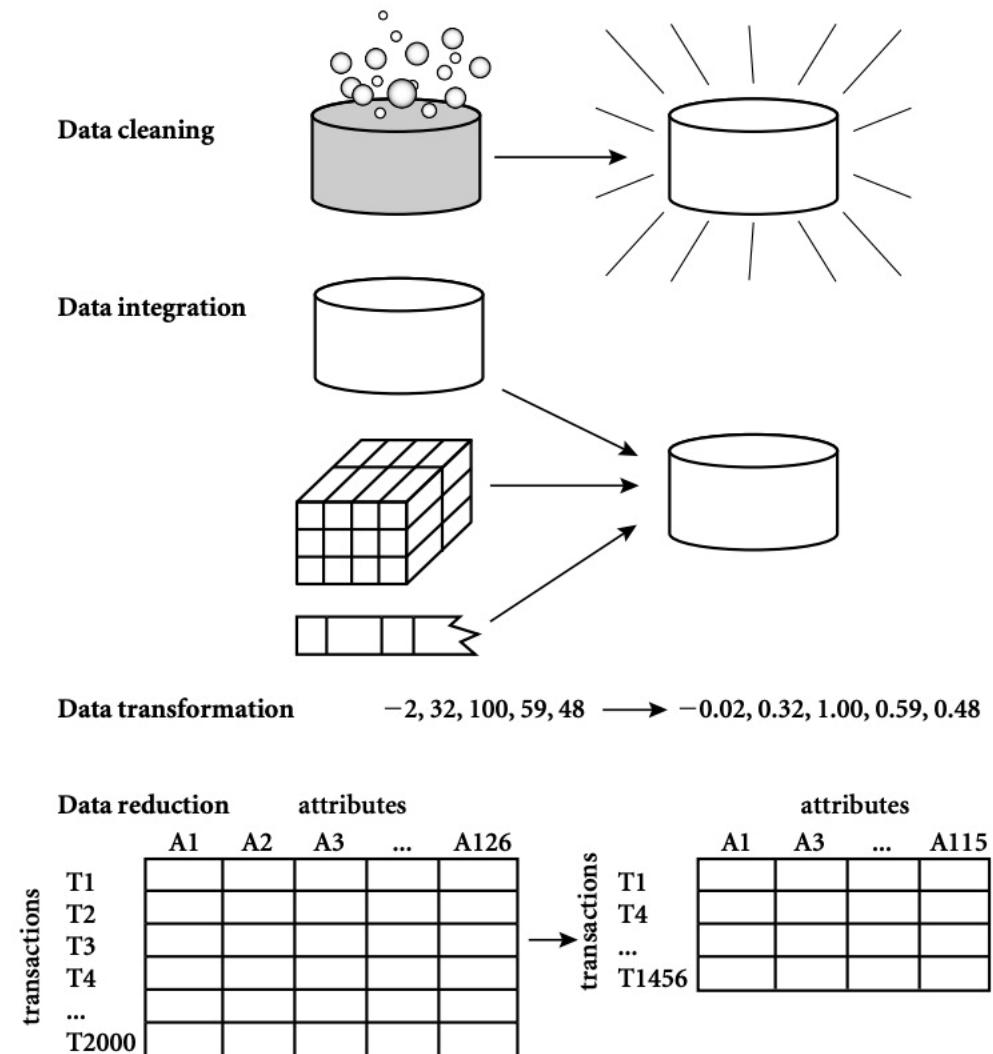
1. Why data preprocessing?
2. Data Cleaning
3. Data Integration
4. Data Transformation
5. Data Reduction
6. Data Discretization

Why data preprocessing - dirty data

- Data in the real world is dirty: Lots of potentially incorrect data, e.g., instrument faulty, human or computer error, transmission error
 - **Incomplete**: lacking attribute values, lacking certain attributes of interest, or containing only aggregate data
 - e.g., Occupation=" " (missing data)
 - **Noisy**: containing noise, errors, or outliers
 - e.g., Salary="-10" (an error)
 - **Inconsistent**: containing discrepancies in codes or names, e.g.,
 - Age="42", Birthday="03/07/2010"
 - Was rating "1, 2, 3", now rating "A, B, C"
 - discrepancy between duplicate records
 - **Intentional** (e.g., disguised missing data)
 - Jan. 1 as everyone's birthday?

Major tasks in data preprocessing

- **Data cleaning:** Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies.
- **Data integration:** Integration of multiple databases, data cubes, files, or notes.
- **Data transformation:** Normalization (scaling to a specific range), aggregation.
- **Data reduction:** Obtains reduced representation in volume but produces the same or similar analytical results.
- **Data discretization:** Reduce the number of values for a given continuous attribute by dividing the range of the attribute into intervals. Interval labels can then be used to replace actual data values.



Data cleaning – handling missing data

1. **Ignore** the tuple: usually done when class label is missing (assuming the task is classification—not effective in certain cases)
2. Fill in the missing value **manually**: tedious + infeasible
3. **Automatically**:
 - 1) Use a **global constant** to fill in the missing value: e.g., “unknown”, a new class
 - 2) Use the **attribute mean** to fill in the missing value
 - 3) Use the **attribute mean for all samples of the same class** to fill in the missing value: smarter
 - 4) Use the **most probable value** to fill in the missing value: inference-based such as regression, Bayesian formula, decision tree

Data cleaning – handling noisy data

- **Binning method**
 - First sort data and partition into bins
 - Then one can **smooth by bin means, median, and boundaries, etc.**
 - Used also for discretization
- **Clustering**
 - Detect and remove outliers
- **Semi-automated method**
 - Combined computer and human inspection
 - Detect suspicious values and check manually
- **Regression**
 - Smooth by fitting the data into regression functions

A useful tool: Regular expressions (Regex)

- **What is Regex**

- A regular expression is a sequence of characters that specifies a search pattern in text.
- It can be used when
 - testing the pattern within the string
 - identify specific text in a document, delete the text completely, or replace it with other text
 - extract substrings from strings based on pattern matching

- **Regex examples**

- `.at` matches any three-character string ending with "at", including "hat", "cat", "bat", "4at", "#at" and " at" (starting with a space).
- `[hc]at` matches "hat" and "cat".
- `[^b]at` matches all strings matched by `.at` except "bat".
- `[^hc]at` matches all strings matched by `.at` other than "hat" and "cat".
- `^[hc]at` matches "hat" and "cat", but only at the beginning of the string or line.
- `[hc]at$` matches "hat" and "cat", but only at the end of the string or line.
- `\[.\]` matches any single character surrounded by "[" and "]" since the brackets are escaped, for example: "[a]", "[b]", "[7]", "[@]", "[]]", and "[]" (bracket space bracket).
- `s.*` matches s followed by zero or more characters, for example: "s", "saw", "seed", "s3w96.7", and "s6#h%(>>>m n mQ".

Data transformation

- **Normalization:** scaled to fall within a small, specified range
 - min-max normalization
 - z-score normalization
 - normalization by decimal scaling
- **Attribute/feature construction**
 - New attributes constructed from the given ones

Data reduction example – dimensionality reduction

Feature selection (i.e., attribute subset selection):

- Select a minimum set of features such that the probability distribution of different classes given the values for those features is as close as possible to the original distribution given the values of all features.
- Nice side-effect: reduces # of attributes in the discovered patterns, easier to understand.



Data discretization

Discretization and concept hierarchy generation for **numeric data**

1. Hierarchical and recursive decomposition using:

- Binning (data smoothing)
- Histogram analysis (numerosity reduction)
- Clustering analysis (numerosity reduction)

2. Entropy-based discretization

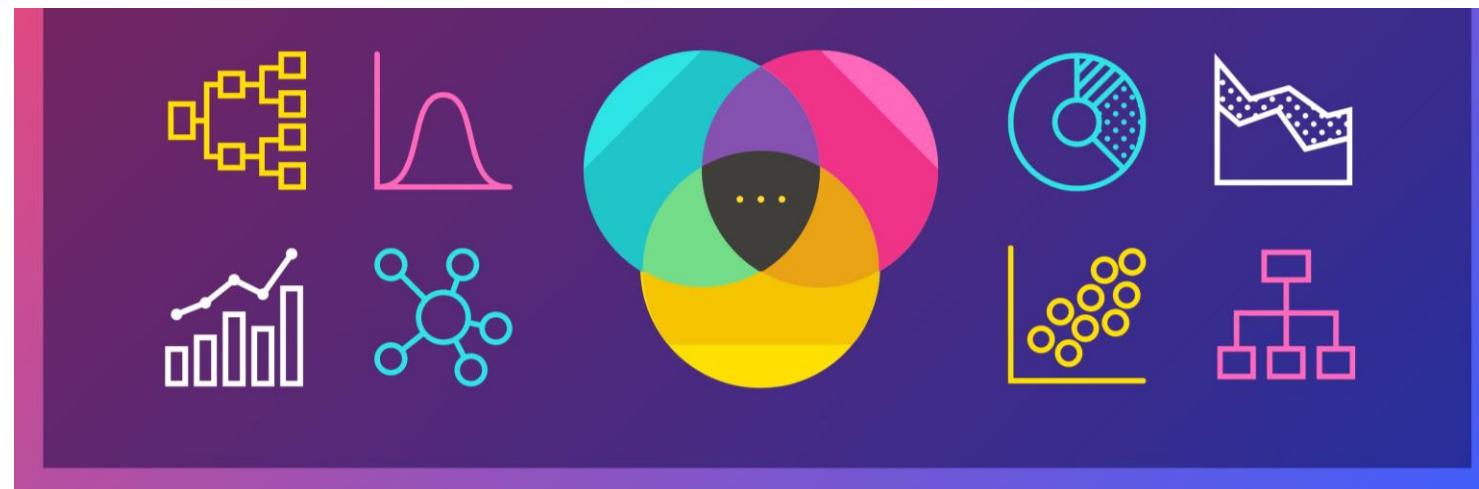
3. Segmentation by natural partitioning

L4: data visualization

1. Introduction to Data Visualization
2. Conventional Visualizations
3. Data Visualization Skills
4. Tips and Tricks
5. Data Visualization Traps
6. Visualization and Dashboard Software

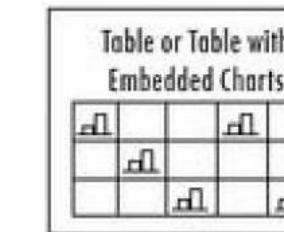
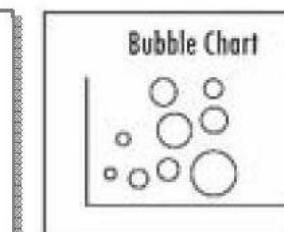
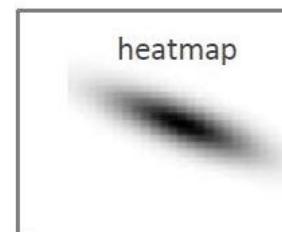
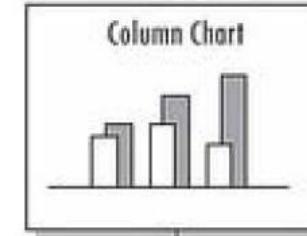
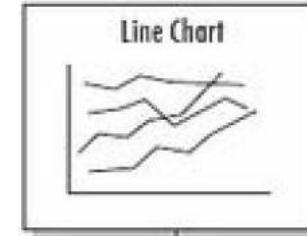
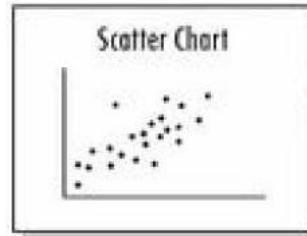
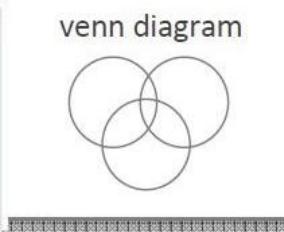
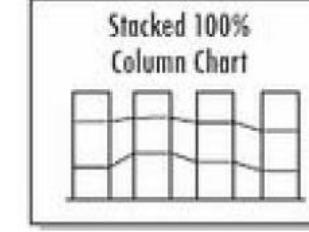
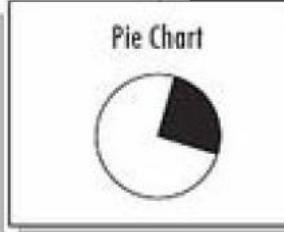
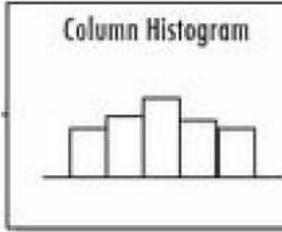
Data visualization

- **Data visualization is the study of visual representations of abstract data to reinforce human cognition.** Its goal is to make information easy to comprehend, interpret, and retain.
- The discipline of communicating information through the use of visual elements such as graphs, charts, and maps.



Conventional visualizations

more discrete dimensions

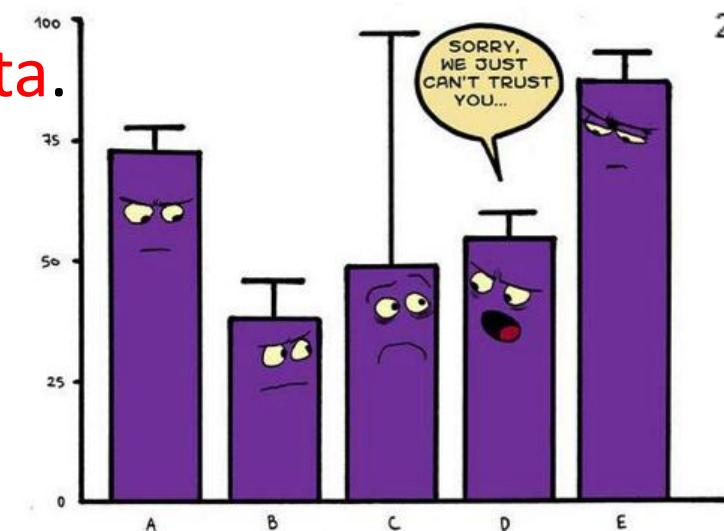


more continuous dimensions

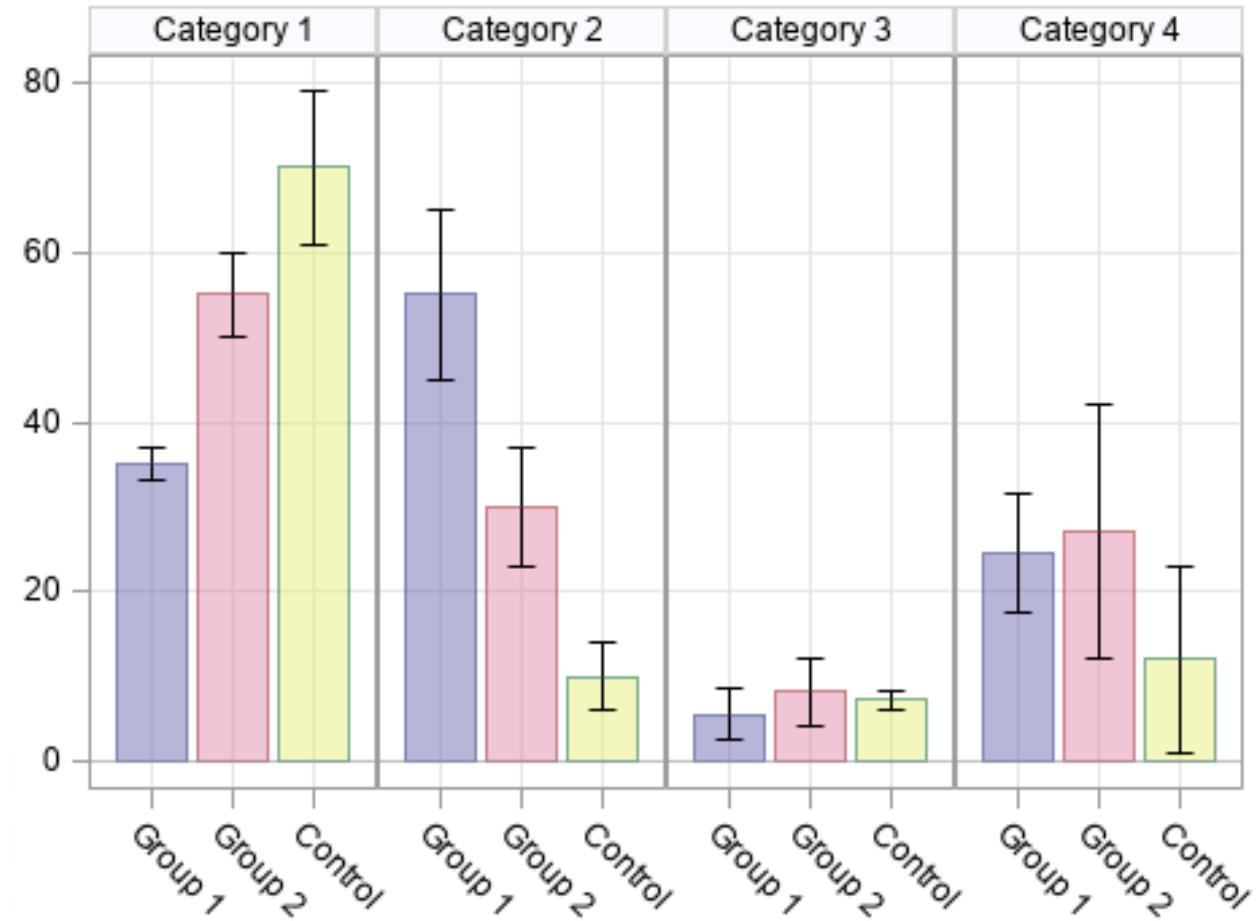
Visualization example – dynamite plots

Dynamite Plot

- Can be used for lots of discrete grouping factors.
- Natural semantics of **grouping**
- **Conceals data.**



Paneled Dynamite Plot
Vertical Bar Charts with Error Bars



Data visualization skills

There are usually 6 reasons to make a chart:

1. To compare
2. To show the distribution
3. To explain parts of the whole
4. To tell the trend over time
5. To find out the deviations
6. To understand the relationship

2 tradeoffs

- **Informativeness vs. readability**

- Too little information can conceal data.
- But too much information can be overwhelming.
- Possible solution: hierarchical organization.

- **Data-centric vs. viewer-centric**

- Viewers are accustomed to certain types of visualization.
- But novel visualization can be truer to data.

L5: LLMs for data engineering

1. LLM&Prompt Engineering
2. Large Language Models for Data Preprocessing
3. Large Language Models for Data Annotation
4. Large Language Models for Data Augmentation
5. Large Language Models for Data Visualization
6. Large Language Models for Text Analysis

ChatGPT has powerful textual processing abilities

- ChatGPT is versatile. It can **write and debug computer programs, mimic the style of celebrity CEOs and write business pitches, compose music, teleplays, fairy tales and student essays, answer test questions** (sometimes, depending on the test, at a level above the average human test-taker), **write poetry and song lyrics, emulate a Linux system; simulate entire chat rooms, play games like tic-tac-toe and simulate an ATM.**

so please write python codes to scrape news from financial times

Sure, here's an example Python code that uses BeautifulSoup and Requests to scrape news from the Financial Times:

```
python
import requests
from bs4 import BeautifulSoup

# Set the URL of the Financial Times news page you want to scrape
url = 'https://www.ft.com/'

# Send a request to the URL and get the HTML response
response = requests.get(url)
html = response.content

# Parse the HTML using BeautifulSoup
soup = BeautifulSoup(html, 'html.parser')
```

so introduce yourself

Hello! I am ChatGPT, a language model trained by OpenAI, based on the GPT-3.5 architecture. I am designed to generate human-like responses to natural language inputs, and I am constantly learning and improving my ability to understand and interact with people. My goal is to provide helpful and informative responses to any questions or

so I want to give a talk to management science professors about how to use ChatGPT to assist their research and teaching. Can you help to provide some topics to cover?

so Certainly! Here are some topics you may want to cover in your talk:

1. **Introduction to ChatGPT:** Start by introducing ChatGPT and explaining what it is, how it works, and what its capabilities are. This will give your audience a basic understanding of the technology and set the stage for the rest of your talk.
2. **Research Applications:** Discuss the various ways in which ChatGPT can be used in management science research, such as natural language processing, sentiment analysis, and data analysis. Explain how ChatGPT can help researchers to gain new insights into complex problems and make more accurate predictions.
3. **Teaching Applications:** Explain how ChatGPT can be used as a teaching tool in management science classrooms, such as for generating case studies, creating quizzes, or even assisting in grading. You may want to provide examples of how ChatGPT can be integrated into existing teaching methods.
4. **Data Collection:** Discuss how ChatGPT can be used to collect data from various sources, such as social media or customer reviews. Explain how this data can be used to gain insights into customer behavior, market trends, and more.
5. **Limitations and Ethical Concerns:** Be sure to mention the limitations of ChatGPT, such as its tendency to generate biased or inaccurate responses. Discuss the ethical concerns that arise when using ChatGPT, such as the potential for misuse or abuse of the technology.
6. **Future Developments:** Finally, discuss the potential for future developments in ChatGPT and how they might impact management science research and teaching. You may want to discuss new features, improvements in accuracy, or the development of new applications.

By covering these topics, you will provide a comprehensive overview of the ways in which ChatGPT can be used in management science research and teaching, while also highlighting its limitations and ethical concerns. Good luck with your talk!

Regenerate response

What is a language model?

- A language model is a probability distribution over sequences of words. Given such a sequence, say of length T , it assigns a probability $P(x^{(1)}, x^{(2)}, \dots x^{(T)})$ to the whole sequence.
- To determine whether a given ordering of words sounds like natural language
- Given such a sequence of length T , it assigns a probability $P(x^{(1)}, x^{(2)}, \dots x^{(T)})$ to the whole sequence. For example, consider a piece of text $(x^{(1)}, x^{(2)}, \dots x^{(T)})$, then according to the Language Model, the probability of the text could be written as:

$$P(x^{(1)}, \dots, x^{(T)}) = P(x^{(1)}) \times P(x^{(2)} | x^{(1)}) \times \dots \times P(x^{(T)} | x^{(T-1)}, \dots, x^{(1)})$$
$$= \prod_{t=1}^T P(x^{(t)} | x^{(t-1)}, \dots, x^{(1)})$$



A language model can also be interpreted as predicting what word comes next



The evolution of GPT models



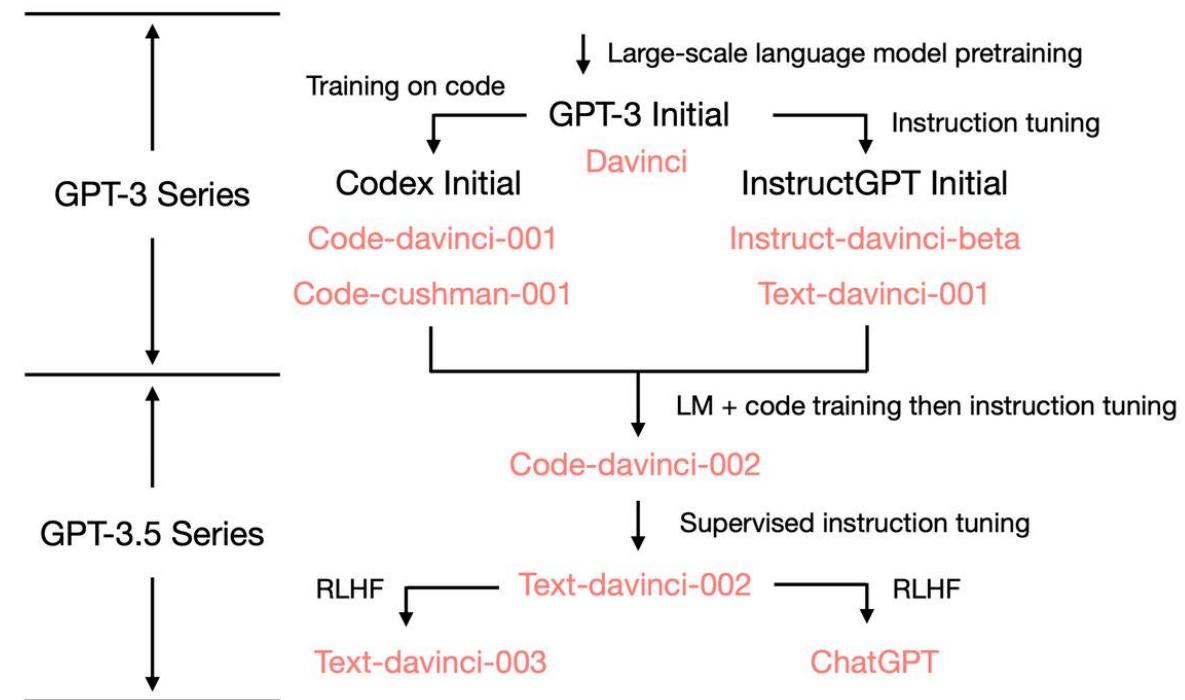
The first version of GPT was released

The second version of GPT was released

Initial GPT-3 preprint paper was published at arXiv. API became publicly available on Nov. 18th, 2021

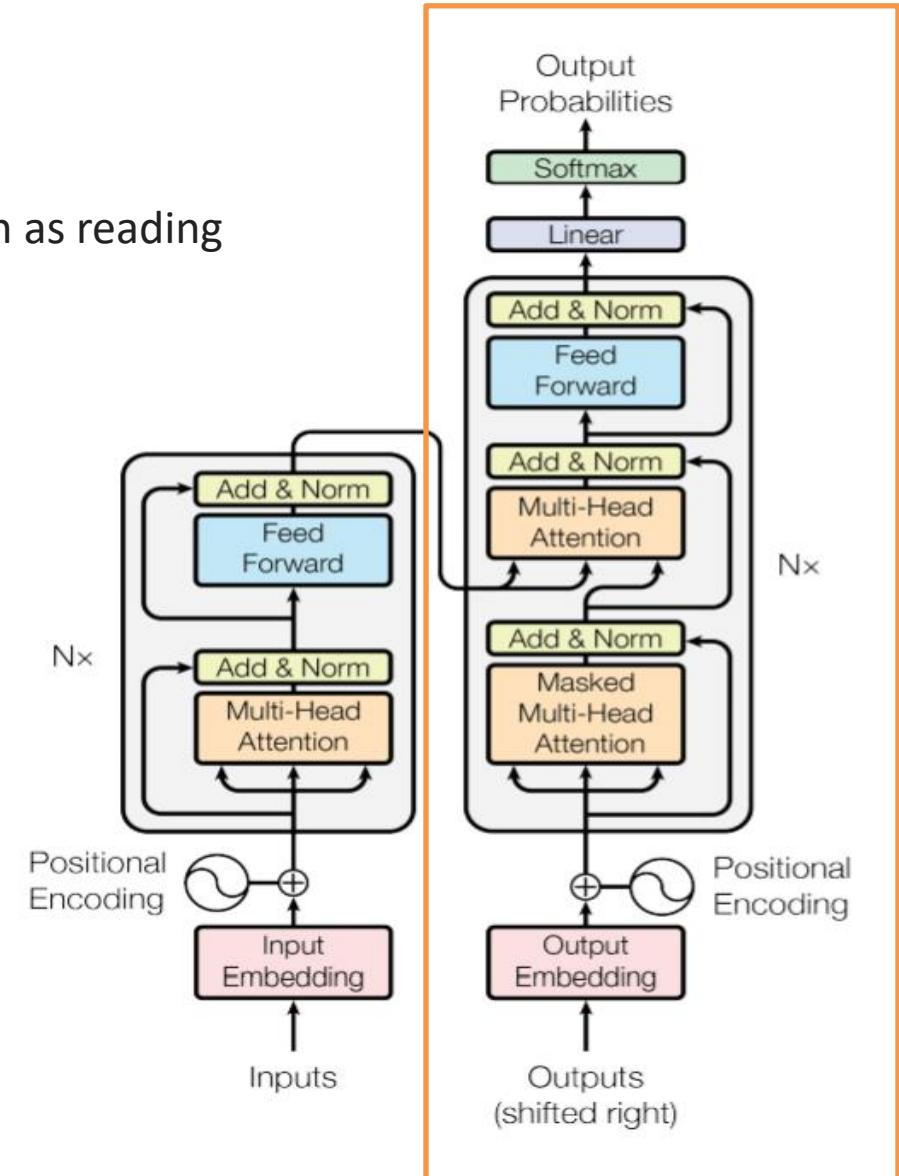
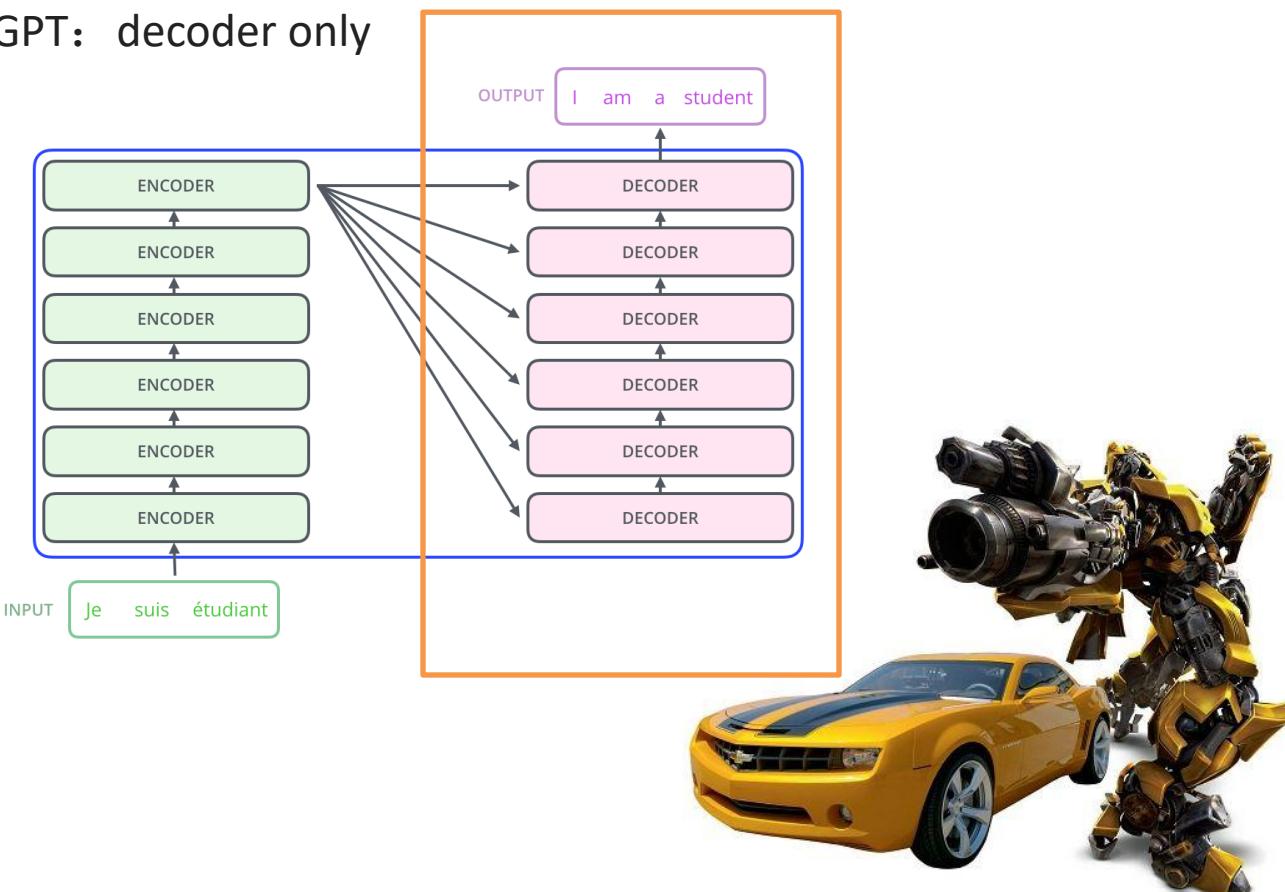
ChatGPT was announced on OpenAI blog. ChatGPT API became available on Mar. 1st, 2023

GPT-4 was released via ChatGPT. API will be publicly available soon.



ChatGPT's core technique: Transformer

- Proposed by Google: Attention is All You Need (Vaswani et al., 2017)
- Main technique: **multi-head self attention mechanism**.
- Solve sequence to sequence understanding and generation tasks, such as reading comprehension, machine translation, text generations.
- GPT: decoder only

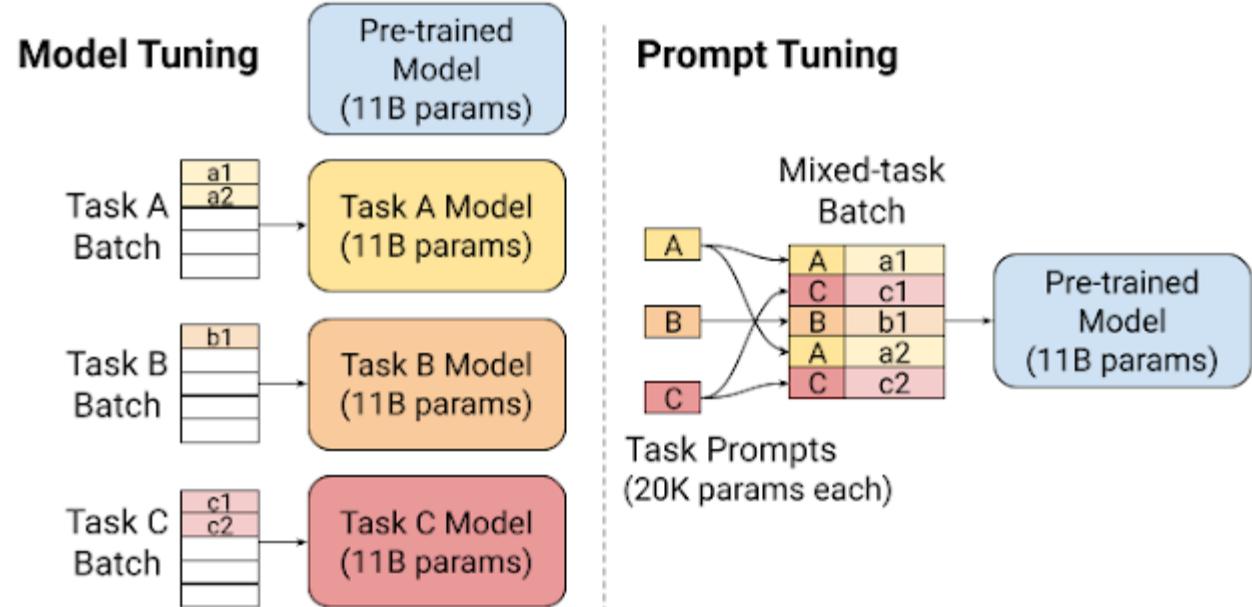
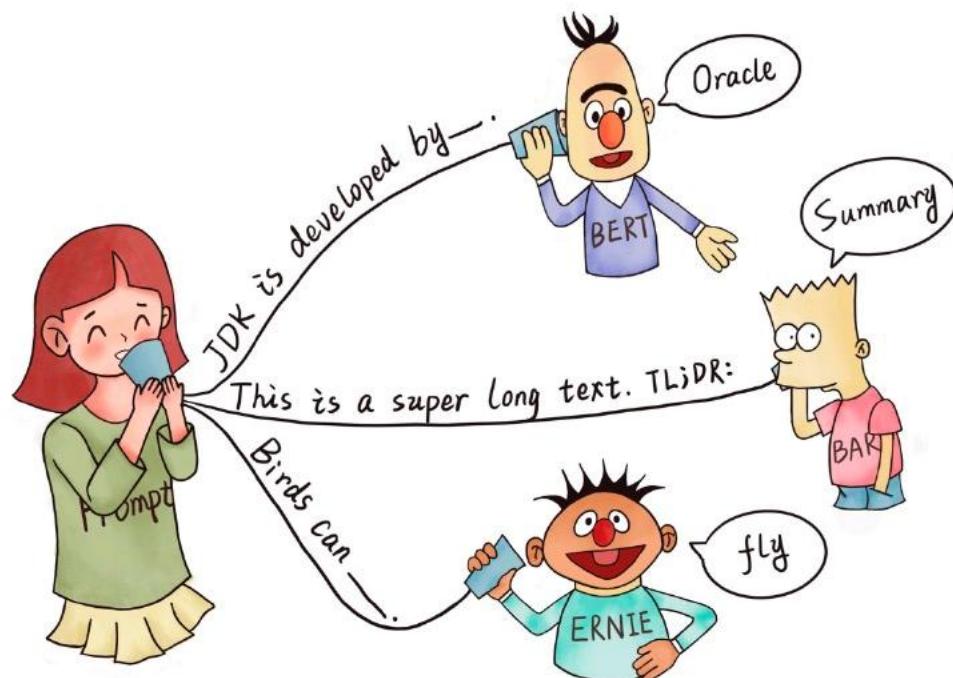


Prompting: Replacing pretraining-finetuning

During past few years, pretraining + finetuning becomes a paradigm for downstream tasks

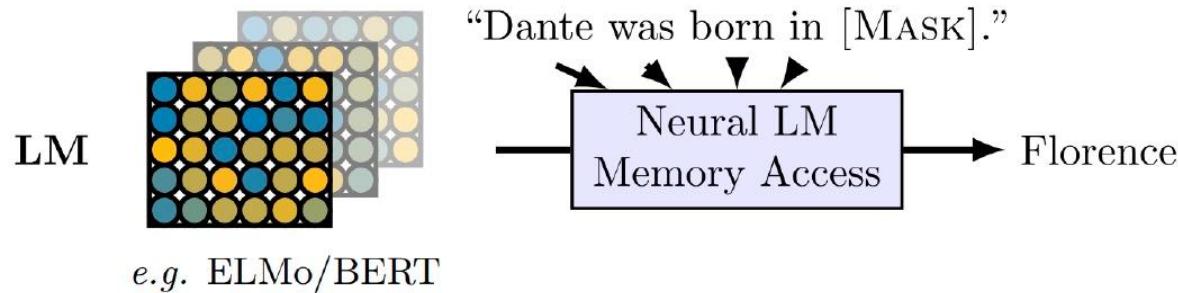
For large models, fine-tuning hundreds of billions of parameters (even a few iterations) is also costly!

How to reduce the “usage cost” of these language models?

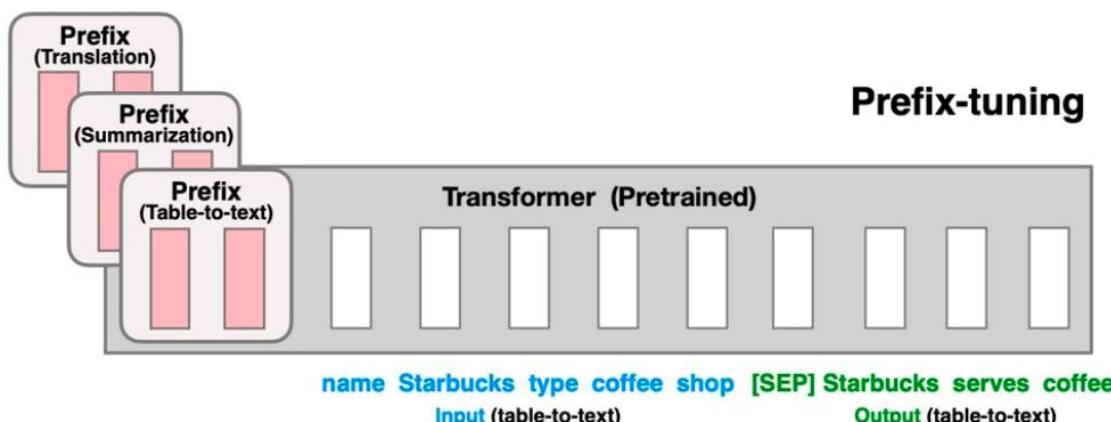


Prompt shape

Cloze prompts: fill in the blanks of a textual string



Prefix prompts: continue a string prefix

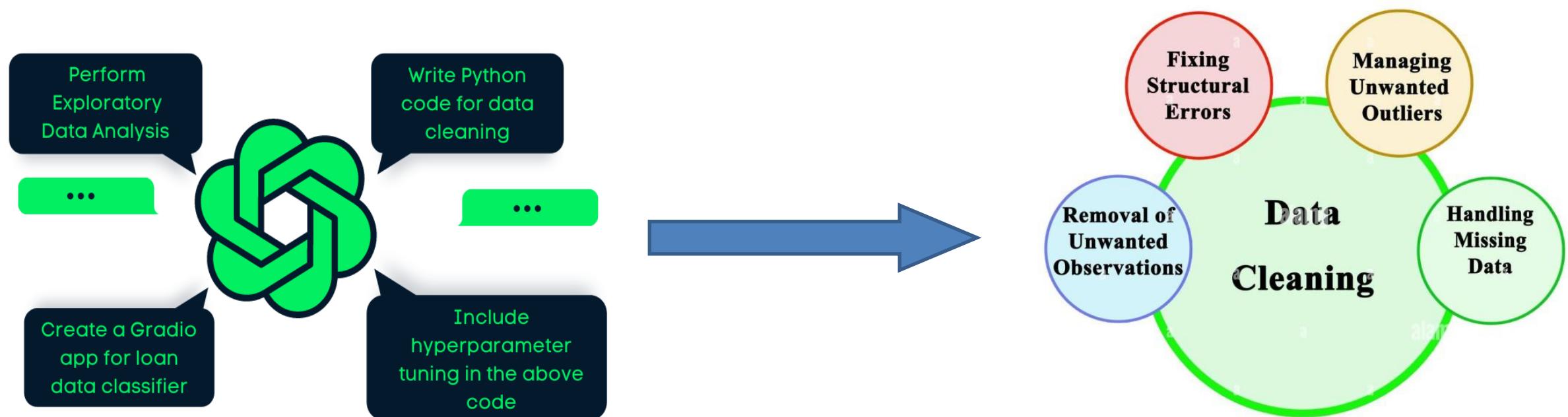


Automated prompt searching

Discrete prompts (a.k.a. **hard prompts**): templates described in a discrete space, usually corresponding to natural language phrases.

Continuous prompts (a.k.a. **soft prompts**): rather than being for human consumption, it is not necessary to limit the prompt to human-interpretable natural. Continuous prompts perform prompting **directly in the embedding space** of the model.

LLMs for data preprocessing



Data preprocessing pipelines via LLMs

Generic Data Cleaning and Preprocessing Pipeline

Can you give me a generic data cleaning and preprocessing pipeline based on what we've done so far. Get basic info on the dataset, check for and handle missing values, identify categorical columns, and encode categorical columns. Use only pandas.

Assistant Poe

Certainly! Here's a generic data cleaning and preprocessing pipeline using pandas, based on the steps you mentioned:

python

Copy

```
import pandas as pd

# Load the dataset
df = pd.read_csv('your_dataset.csv')

# Get basic information on the dataset
print("Dataset Information:")
print(df.info())

# Check for missing values
print("Missing Values:")
print(df.isnull().sum())

# Handle missing values
# Example: Replace missing values with the mean of each column
df.fillna(df.mean(), inplace=True)
```

```
# Check for missing values
print("Missing Values:")
print(df.isnull().sum())

# Handle missing values
# Example: Replace missing values with the mean of each column
df.fillna(df.mean(), inplace=True)

# Identify categorical columns
categorical_columns = df.select_dtypes(include='object').columns

# Encode categorical columns using one-hot encoding
encoded_df = pd.get_dummies(df, columns=categorical_columns)

# Print the preprocessed dataset
print("Preprocessed Dataset:")
print(encoded_df)
```

This pipeline assumes that your dataset is in a CSV file format. Here's a breakdown of the steps:

1. Load the dataset using `pd.read_csv('your_dataset.csv')`. Replace 'your_dataset.csv' with the path to your actual dataset file.
2. Get basic information on the dataset using `df.info()`. This provides an overview of the column names, data types, and non-null

1. Load the dataset using `pd.read_csv('your_dataset.csv')`. Replace 'your_dataset.csv' with the path to your actual dataset file.
2. Get basic information on the dataset using `df.info()`. This provides an overview of the column names, data types, and non-null counts.
3. Check for missing values using `df.isnull().sum()`. This counts the number of missing values in each column.
4. Handle missing values according to your desired strategy. In the example, missing values are replaced with the mean of each column using `df.fillna(df.mean(), inplace=True)`. You can modify this step to handle missing values based on your specific requirements.
5. Identify categorical columns using `df.select_dtypes(include='object').columns`. This selects columns with object (string) data type as categorical columns. Modify the `include` parameter if your categorical columns have a different data type.
6. Encode categorical columns using one-hot encoding with `pd.get_dummies(df, columns=categorical_columns)`. This creates binary columns for each unique category in the categorical columns.
7. Print the preprocessed dataset using `print(encoded_df)`. This displays the dataset with the encoded categorical columns.

LLM augmented data

Bottleneck of Data-Centric LLM:

Where does the data come from? Is the data in the world unlimited?

High-quality data is scarce, high-quality data are books and scientific papers, or well-annotated corpus. Low-quality data, such as chat records, phone calls, etc., have a relatively limited effect on training due to poor quality. Limitations brought by data security and privacy: sensitive information such as personally identifiable information, financial information, or medical records are unavailable for LLMs.

Possible Solution: LLM-synthetic / augmented data can meet the needs.

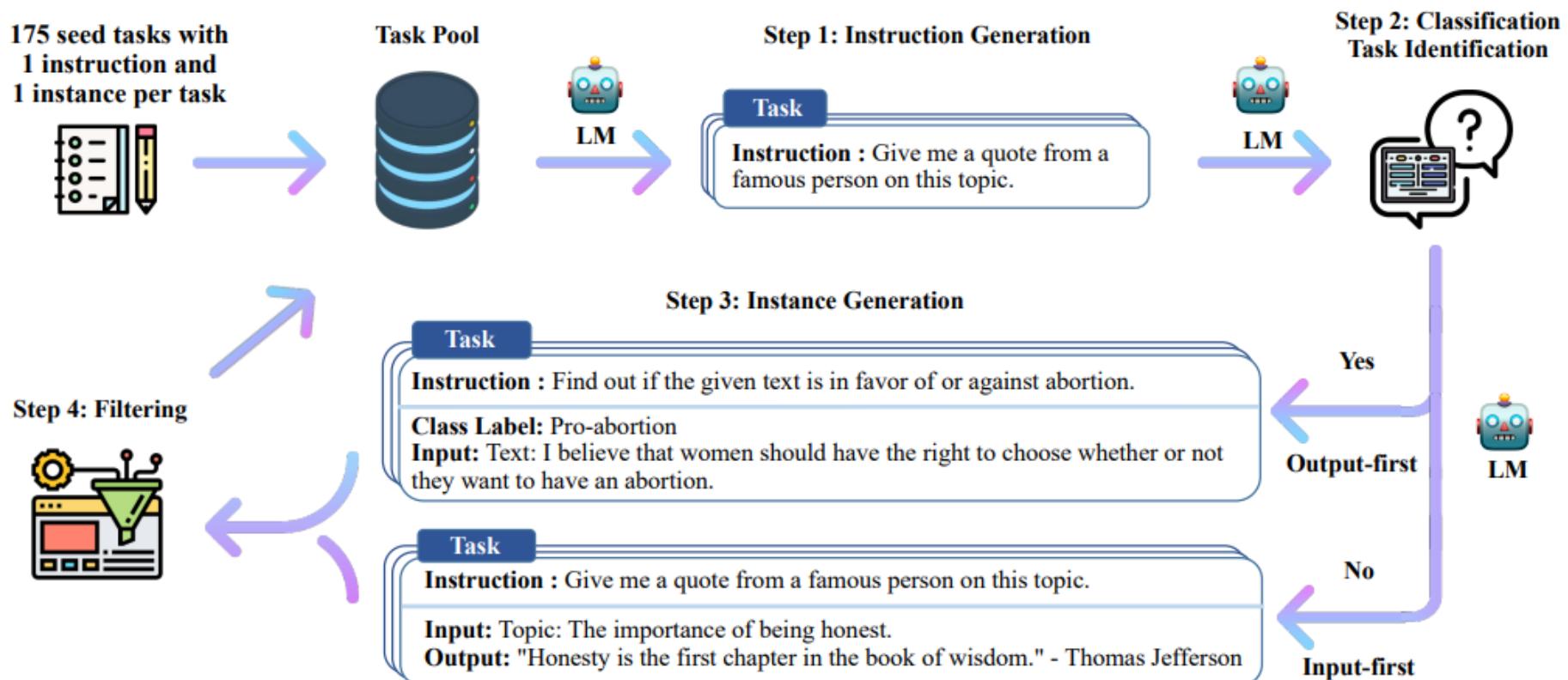
LLM-guided data Collection

Model distillation: Use the output / logits of the LLM as a label to train small models. This is often better than training a small model directly on the original dataset.

Self-Instruct: Instruction-finetune that using instruction generated by LLM itself. The process is an iterative bootstrapping algorithm.

Self Instruct

Loop from step 1 to 4 to get more data



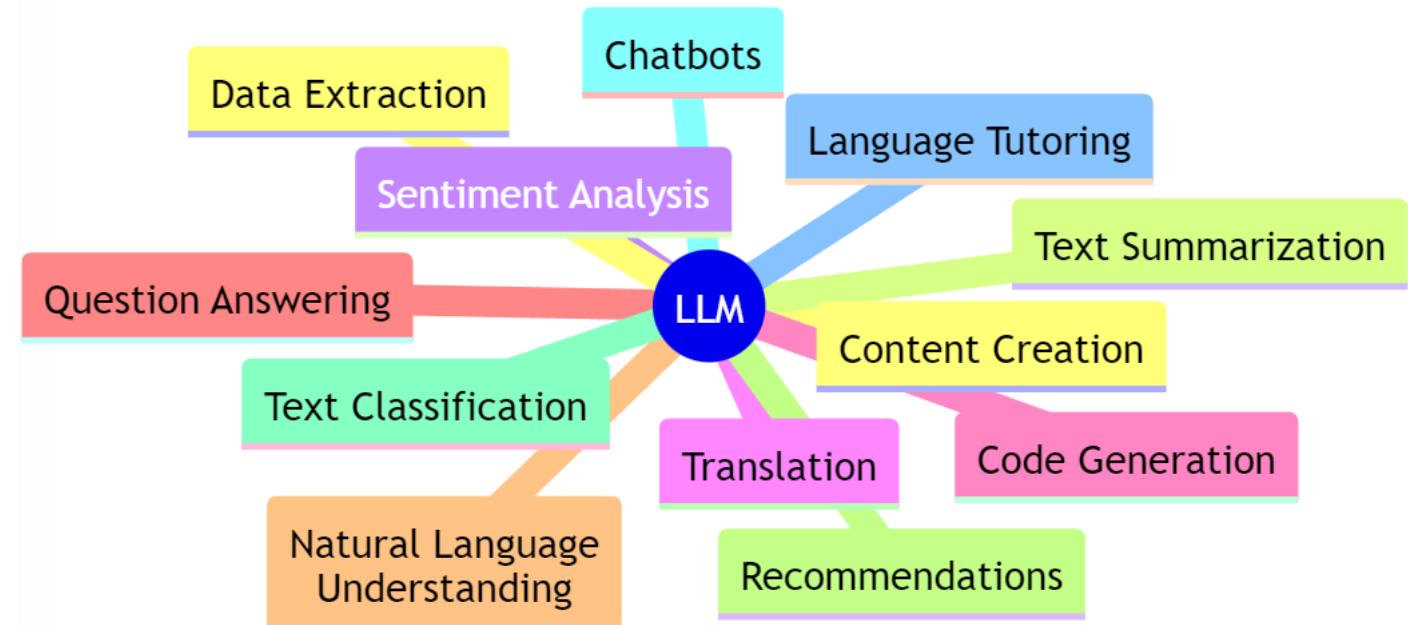
Large Language Models for Text Analysis

sentiment analysis

text classification

information extraction

machine translation

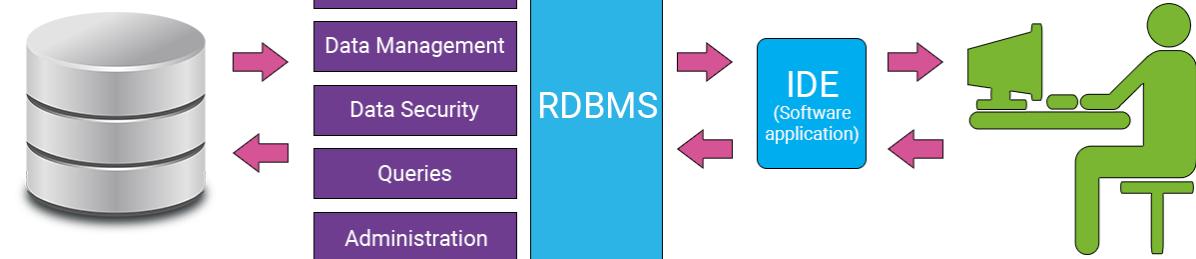
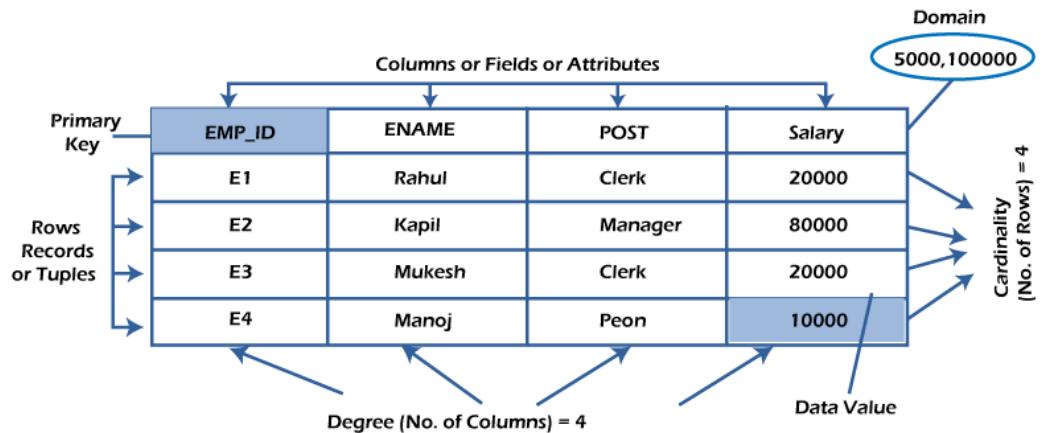
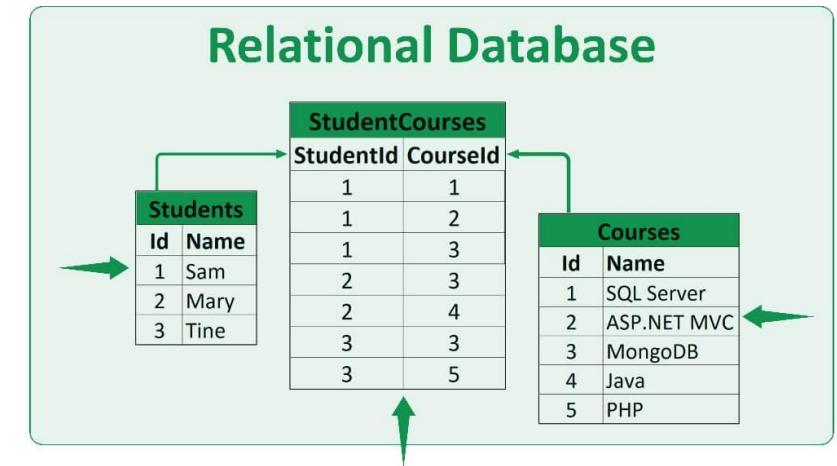


L6: data indexing

1. Data Storage
2. Records and Files
3. Hashed Files
4. Indexing Techniques
5. Tree Data Structure

Relational databases

- A **relational database** is based on the relational model of data.
- A **relational database management system (RDBMS)** is a system to maintain relational databases.
- Many relational database systems are equipped with the option of using the **SQL (Structured Query Language)** for querying and maintaining the database.



NoSQL databases

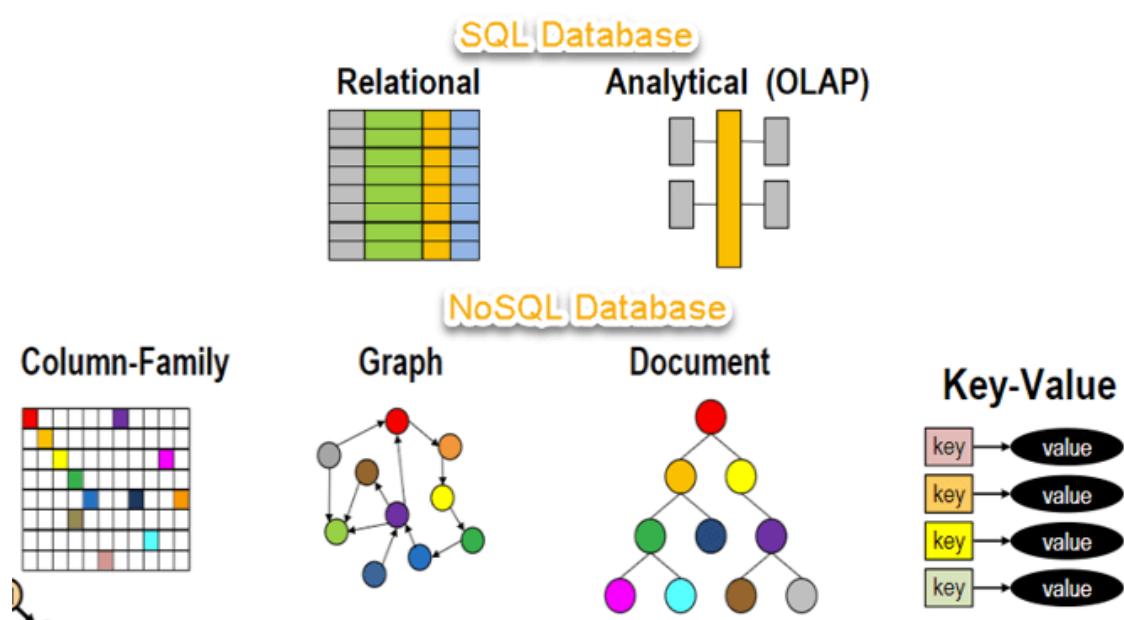
NoSQL databases are geared toward managing large sets of varied and frequently updated data, often in distributed systems or the cloud. They avoid the rigid schemas associated with relational databases. But the architectures themselves vary and are separated into four primary classifications, although types are blending over time.

- **NoSQL** stands for:

- No Relational
- No RDBMS
- **Not Only SQL**

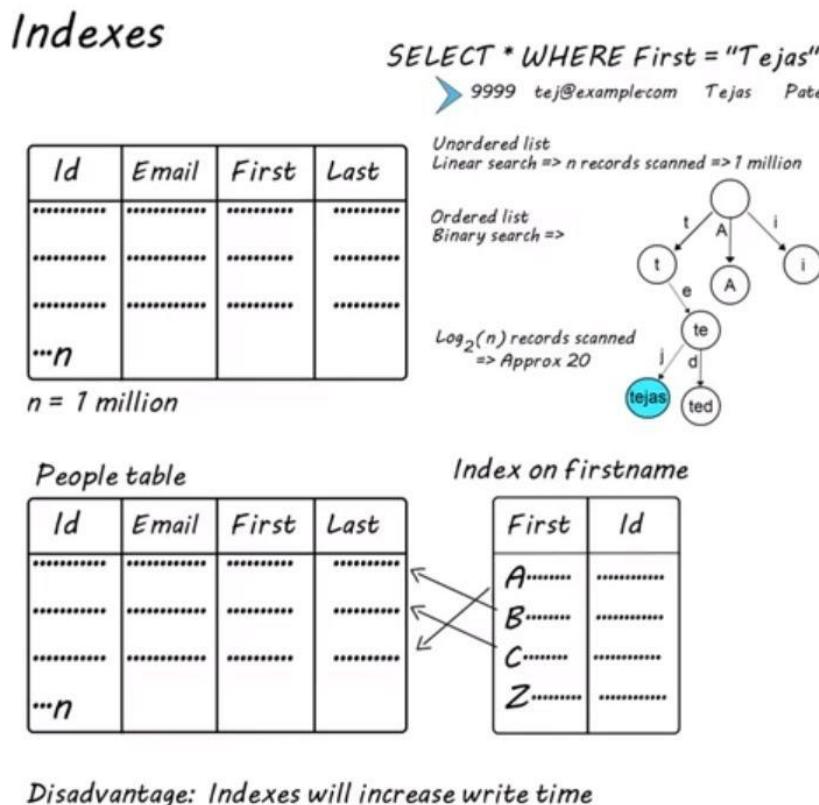
- NoSQL is an umbrella term for all databases and **data stores that don't follow the RDBMS principles**

- A class of products
- A collection of several (related) concepts about data storage and manipulation
- Often related to large data sets



Data indexing

A **database index** is a data structure that **improves the speed of data retrieval operations** on a database table at the cost of additional writes and storage space to maintain the index data structure.



Indexes

"Search faster"



Unordered records (Heap files)

Ordered records (Sorted files)

Hashed records

B and B+ tree data structure for indexing

B+ tree index

- Most implementations of a dynamic multilevel index use a variation of the B-tree data structure called a B+-tree.
- In a B+-tree, **data pointers are stored only at the leaf nodes of the tree.**
 - The **pointers in internal nodes are tree pointers** to blocks that are tree nodes.
 - The **pointers in leaf nodes are data pointers** to the data file records.
- Because entries in the internal nodes of a B+-tree does not include data pointers, more entries (tree pointers) can be packed into an internal node and thus fewer levels (**higher capacity**).
- The **leaf nodes** of a B+-tree are usually **linked** to provide ordered access on the search field to the records.

L7: data querying

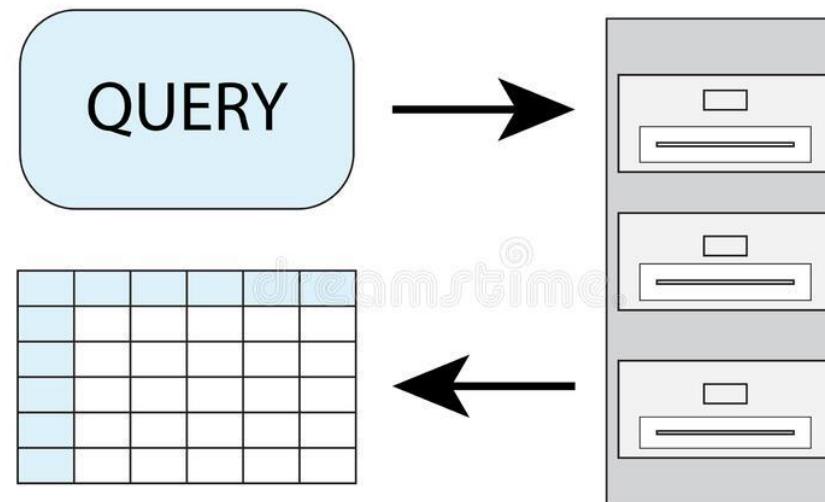
1. Overview of data querying
2. Relational algebra and SQL querying
3. NoSQL querying

Data querying

A **data query** is either an **action query** or a **select query**.

A **select query** is one that **retrieves data from a database**.

An **action query** asks for **additional operations on data**, such as **insertion, updating, deleting** or other forms of data manipulation.



Relational databases and relational algebra

| <i>ID</i> | <i>name</i> | <i>dept_name</i> | <i>salary</i> |
|-----------|-------------|------------------|---------------|
| 10101 | Srinivasan | Comp. Sci. | 65000 |
| 12121 | Wu | Finance | 90000 |
| 15151 | Mozart | Music | 40000 |
| 22222 | Einstein | Physics | 95000 |
| 32343 | El Said | History | 60000 |
| 33456 | Gold | Physics | 87000 |
| 45565 | Katz | Comp. Sci. | 75000 |
| 58583 | Califieri | History | 62000 |
| 76543 | Singh | Finance | 80000 |
| 76766 | Crick | Biology | 72000 |
| 83821 | Brandt | Comp. Sci. | 92000 |
| 98345 | Kim | Elec. Eng. | 80000 |

attributes
(or columns)

tuples
(or rows)

Six basic operators

select: σ

project: Π

union: \cup

set difference: $-$

Cartesian product: \times

rename: ρ

Additional operations that simplify common queries

set intersection

join

assignment

outer join

Relational databases

Relational algebra

SQL – Select

Select *<List of Columns and expressions (usually involving columns)>*

From *<List of Tables & Join Operators>*

Where *<List of Row conditions joined together by And, Or, Not>*

Group By *<list of grouping columns>*

Having *<list of group conditions connected by And, Or, Not >*

Order By *<list of sorting specifications>*

| Payroll | | | |
|---------------|-------------|------------|---------------|
| UserID | Name | Job | Salary |
| 123 | Jack | TA | 50000 |
| 345 | Allison | TA | 60000 |
| 567 | Magda | Prof | 120000 |
| 789 | Dan | Prof | 100000 |



```
SELECT P.Name, P.UserID  
FROM Payroll AS P  
WHERE P.Job = 'TA';
```

NoSQL querying – document databases as an example

- E.g., MongoDB querying: Mongo query language
 - Targets a **specific collection** of documents
 - Specifies **criteria** that identify the returned documents
 - May include a projection to **specify returned fields**
 - May impose **limits, sort, orders, ...**
- Basic query - all documents in the collection:
 - `db.users.find()`
 - `db.users.find({})`
- E.g.,

```
db.inventory.find({ type: "snacks" })
```

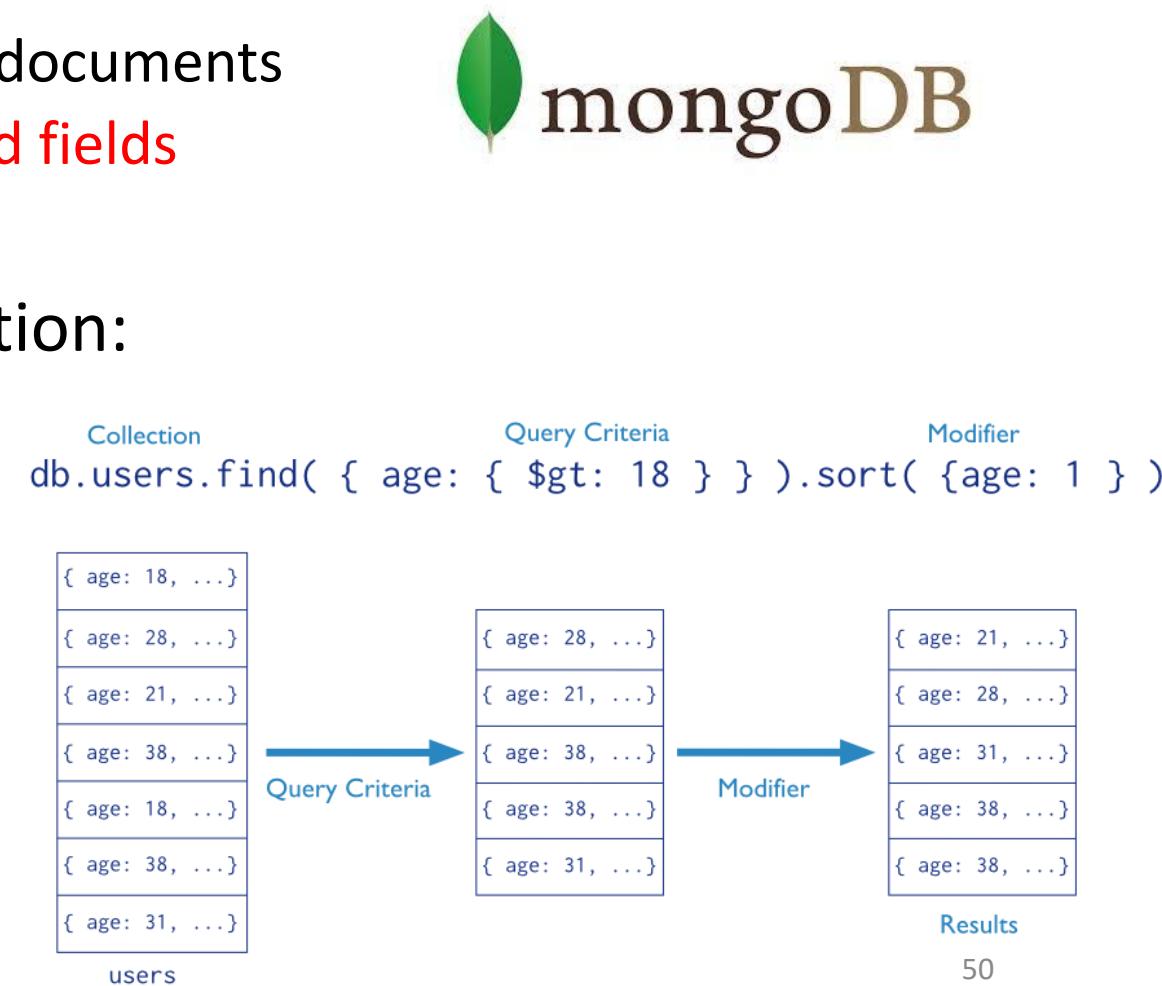
All documents from collection **inventory** where the **type** field has the value **snacks**

```
db.inventory.find({ type: { $in: [ 'food', 'snacks' ] } } )
```

All **inventory** docs where the **type** field is either **food** or **snacks**

```
db.inventory.find( { type: 'food', price: { $lt: 9.95 } } )
```

All ... where the **type** field is **food** and the **price** is less than 9.95

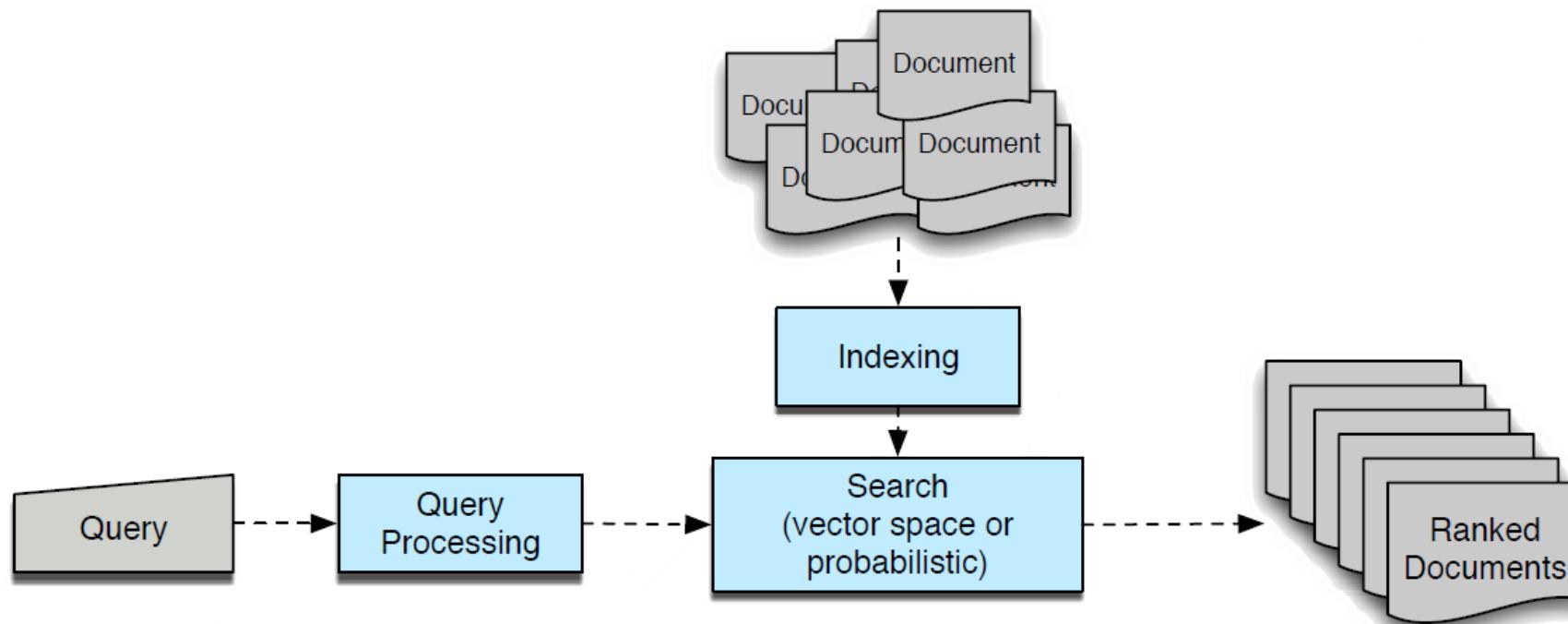


L8&L9: data-driven applications

1. Information retrieval
2. Recommendations
3. Social network analysis
4. Anomaly detection

Process of information retrieval

- Some of the key issues
 - Information need: query and query processing
 - Relevance: indexing documents and search algorithm
 - Evaluation: document ranking, precision-recall, etc.

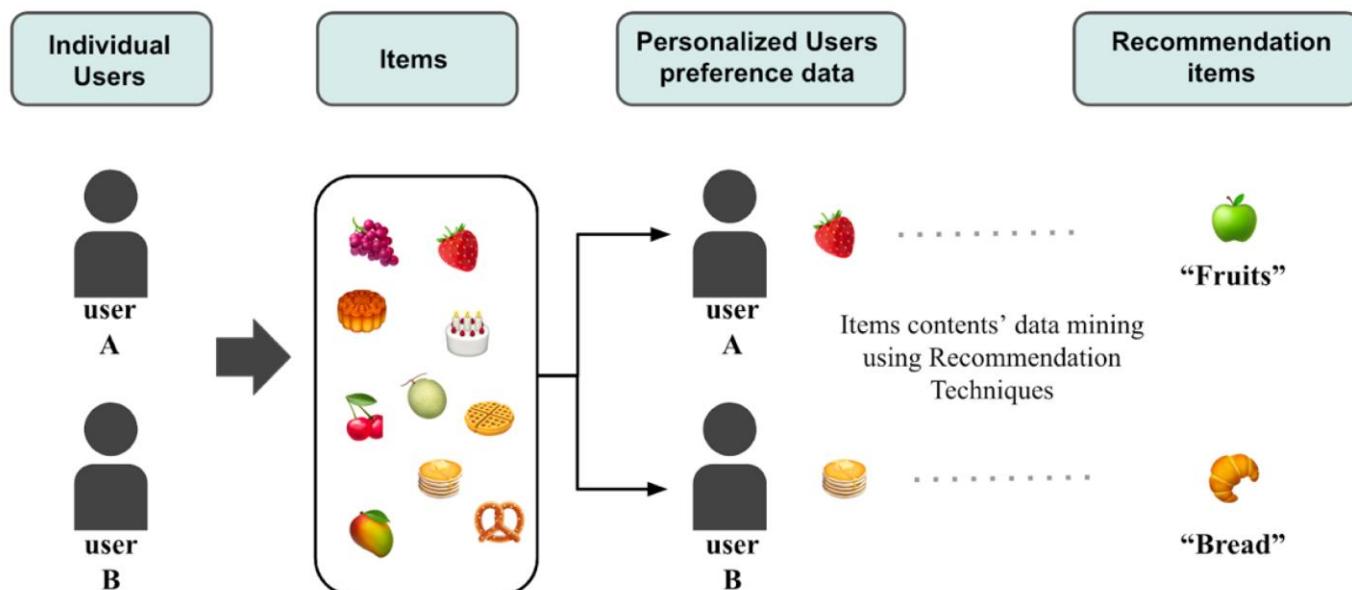


How to find relevant documents for a query?

- By keyword matching
 - Boolean model
- By similarity
 - Vector space model
- By imaging how to write out a query
 - How likely a query is written with this document in mind.
 - Generate with some randomness
 - Query generation language model
- By trusting how other web pages think about the web page
 - Pagerank, hits
- By trusting how other people find relevant documents for the same or similar query
 - Learning to rank

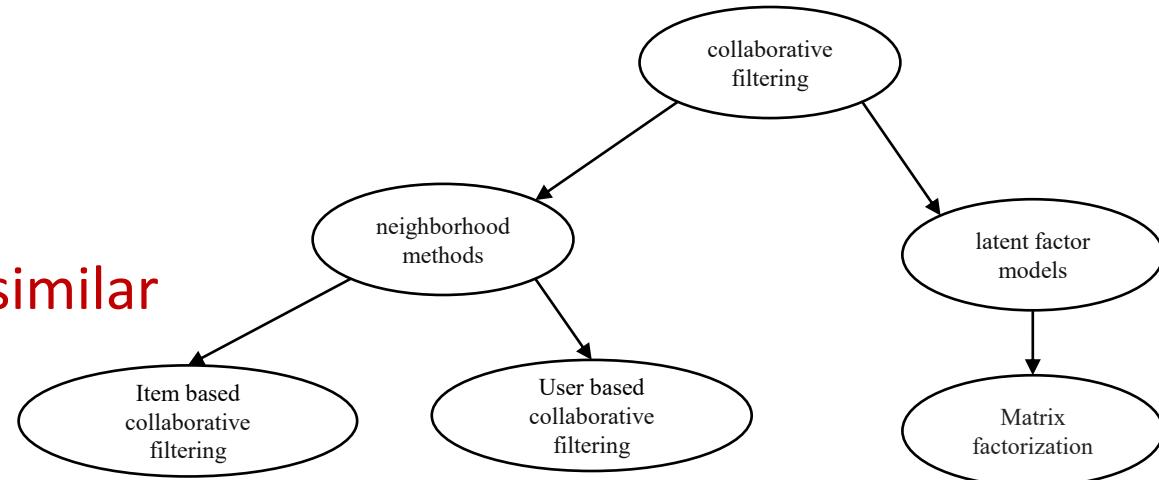
Recommender systems

- Recommender systems: a subclass of information filtering system that provide suggestions for items that are most pertinent (of interest) to a particular user.

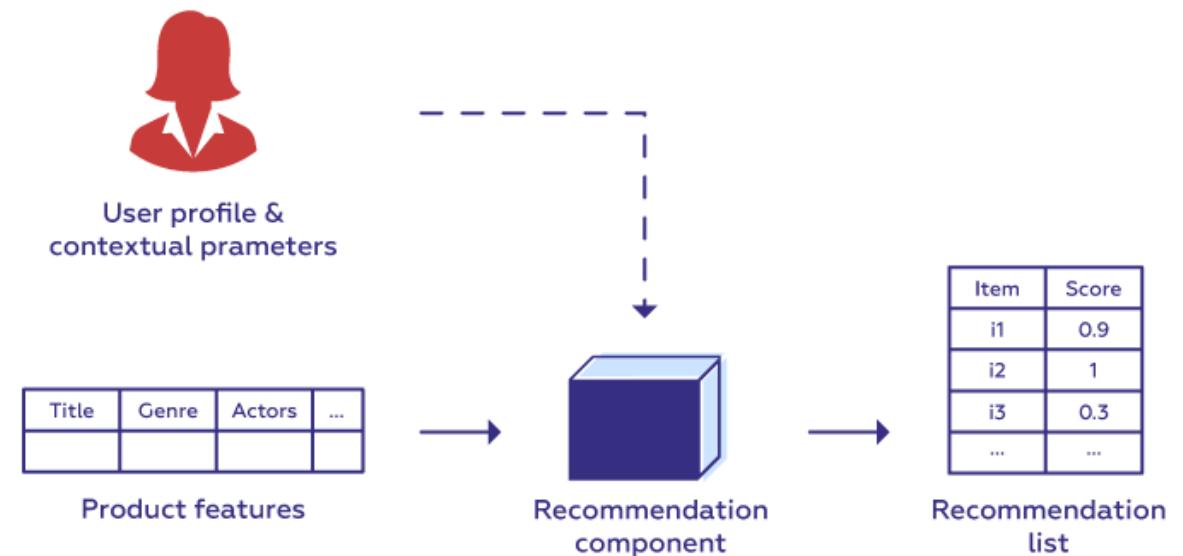


Collaborative filtering

- Item/User based collaborative filtering: find **similar items/users and recommend similar items.**
- Matrix factorization: represent each item and user as an embedding vector and calculate their preference score (e.g., inner product).



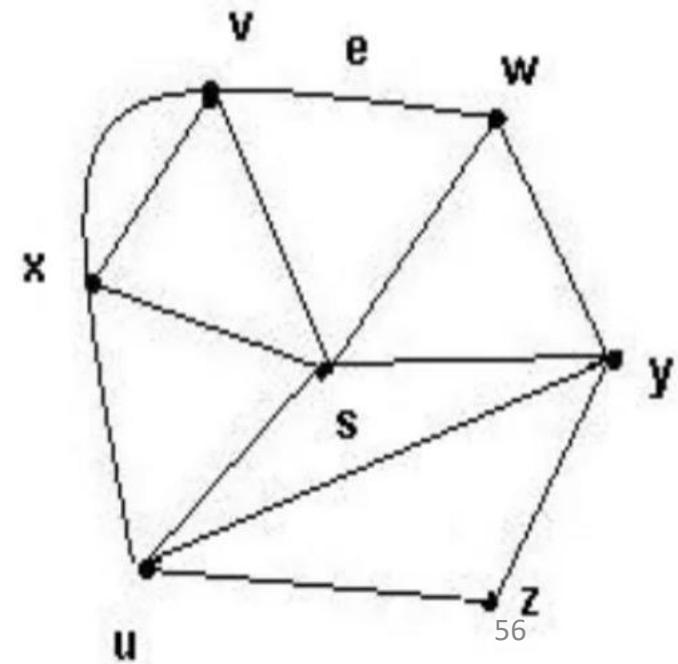
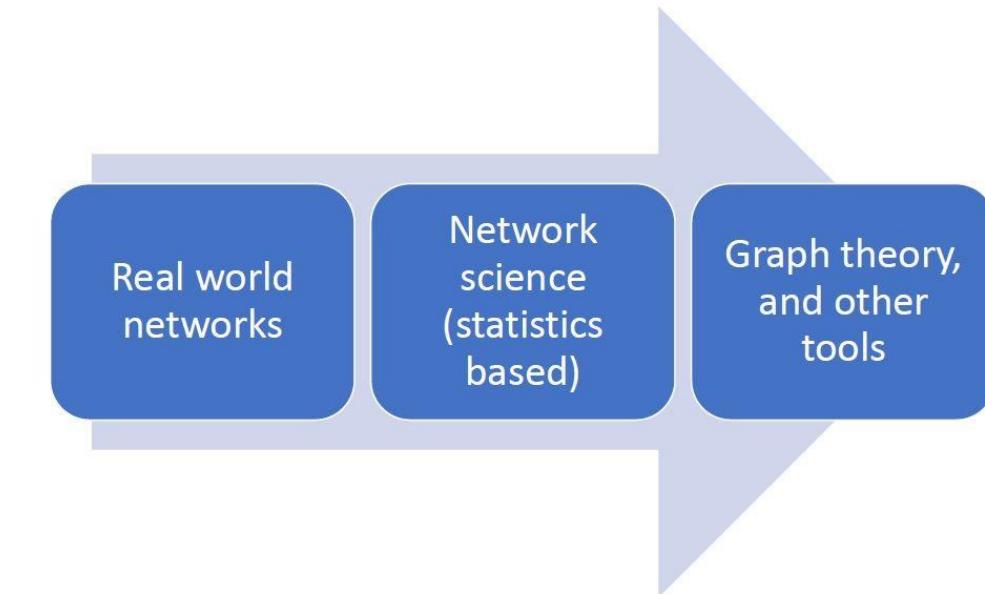
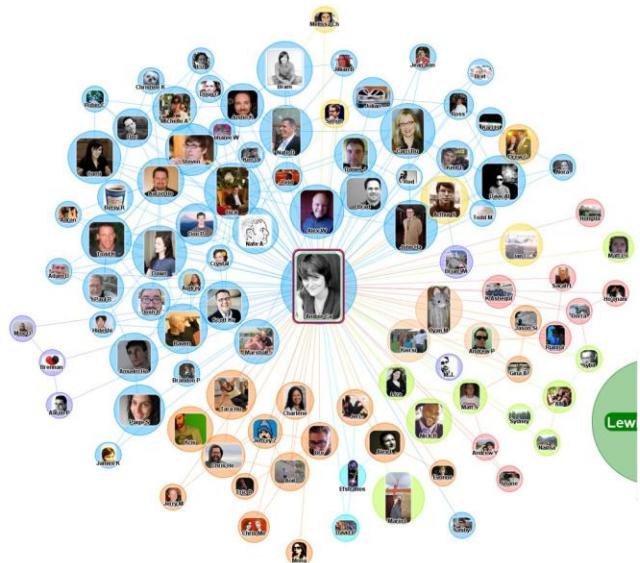
A user/An item profile is a collection of settings and information associated with a user/an item.



Social network analysis – graph theory

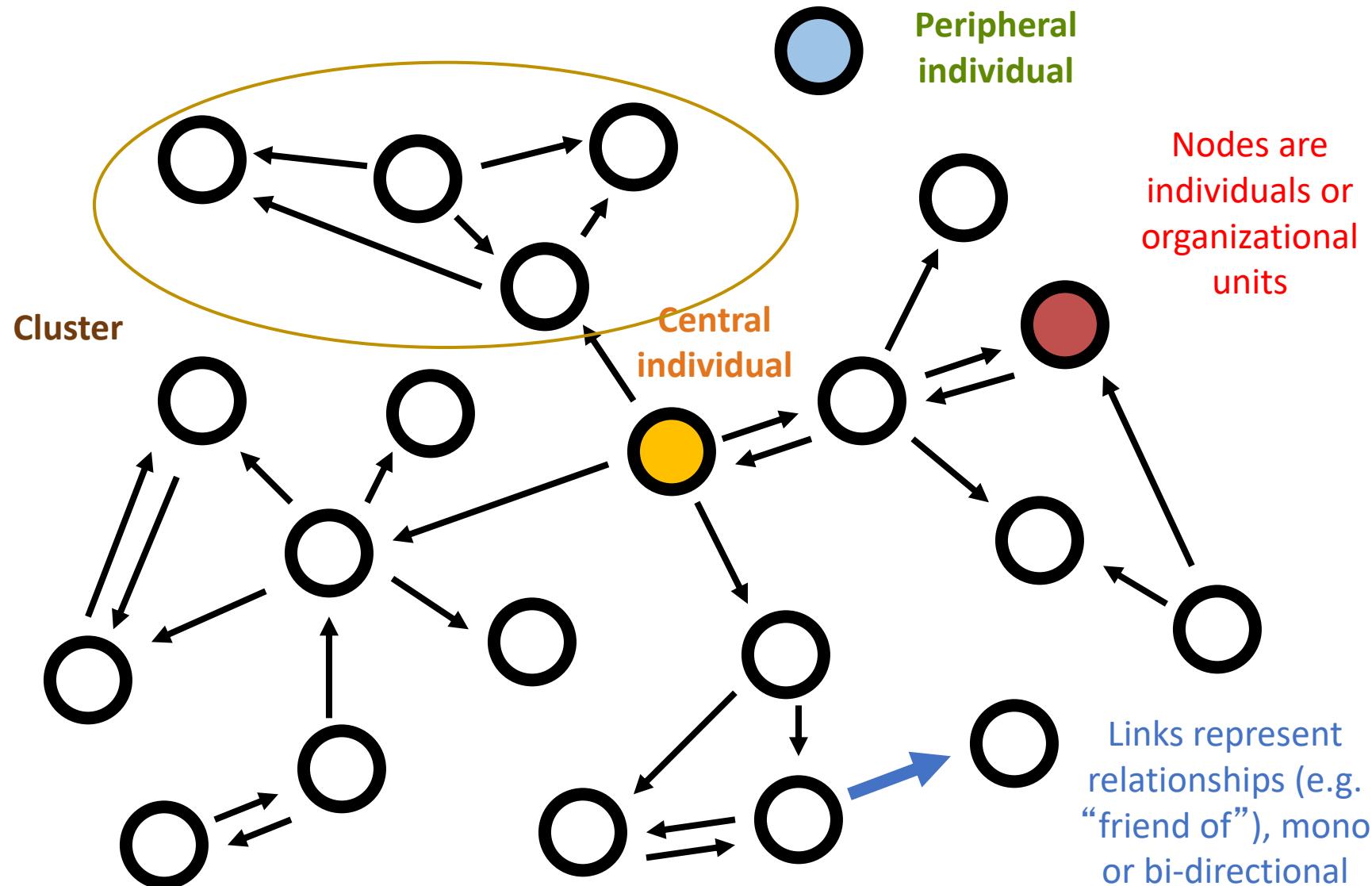
Graph theory is a branch of discrete mathematics concerned with proving theorems and developing algorithms for arbitrary graphs (e.g. random graphs, lattices, hierarchies).

Graph theory and network science



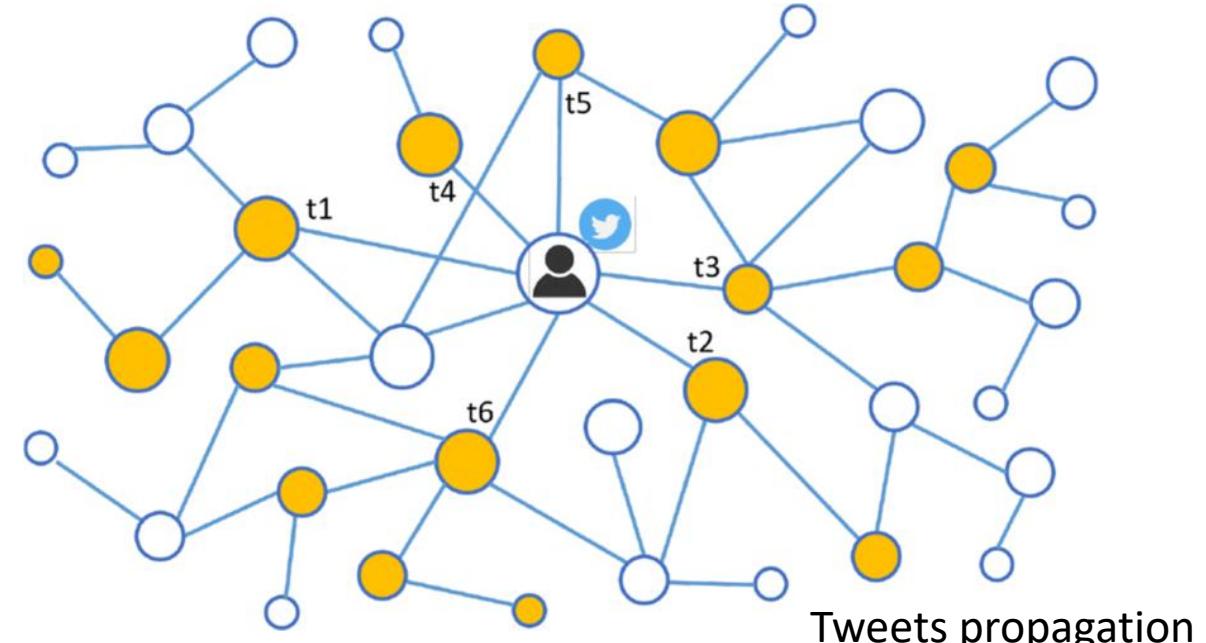
Sociograms

A **sociogram** is a graphic representation of social links that a person has. It is a graph drawing that plots the structure of interpersonal relations in a group situation.



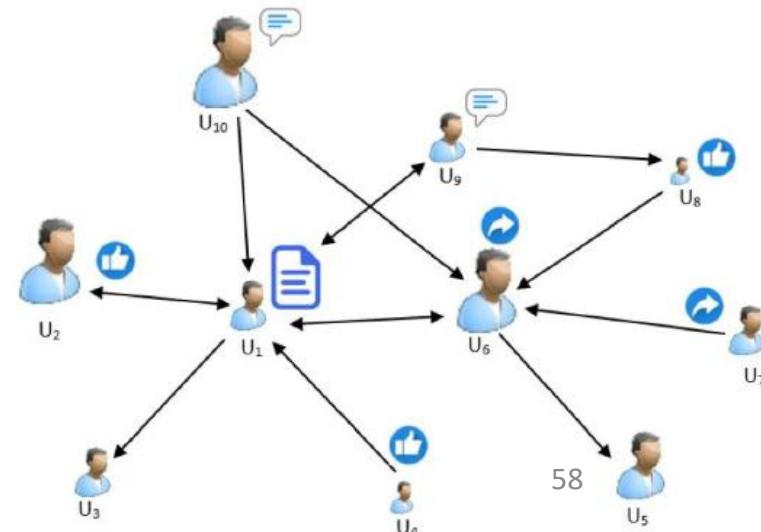
Social network analysis example – network propagation

- **Network propagation** refers to a class of algorithms that integrate information from input data across connected nodes in a given network.
- Study the **dynamics of network propagation**, i.e., how data/information is spread across networks, and how networks are influenced in the meantime.
 - Epidemic/Virus spread model
- Applications
 - News/Tweets/Memes/Trends propagation
 - Rumer/Fake news detection



Memes = social genes

Fake news detection



Anomaly detection overview

Anomaly detection: the identification of rare items, events or observations which deviate significantly from the majority of the data and do not conform to a well defined notion of normal behavior.

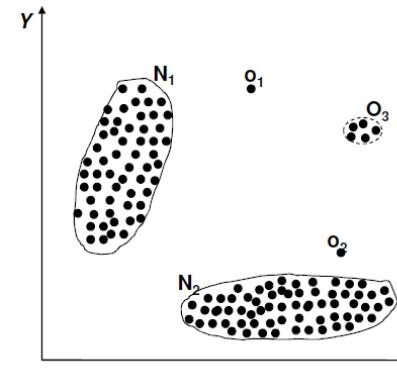
Historically, the field of statistics tried to find and remove outliers as a way to improve analyses.

There are now many fields where the outliers / anomalies are the objects of greatest interest.

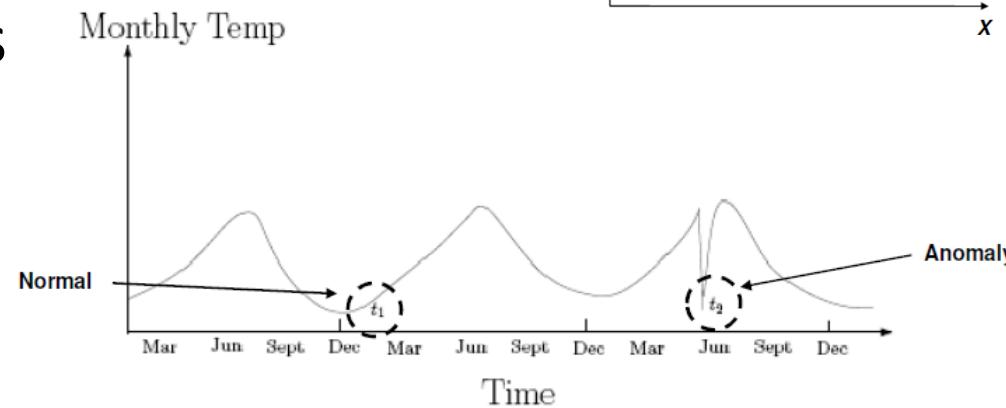
- The rare events may be the ones with the greatest impact, and often in a negative way.

Structure of anomalies

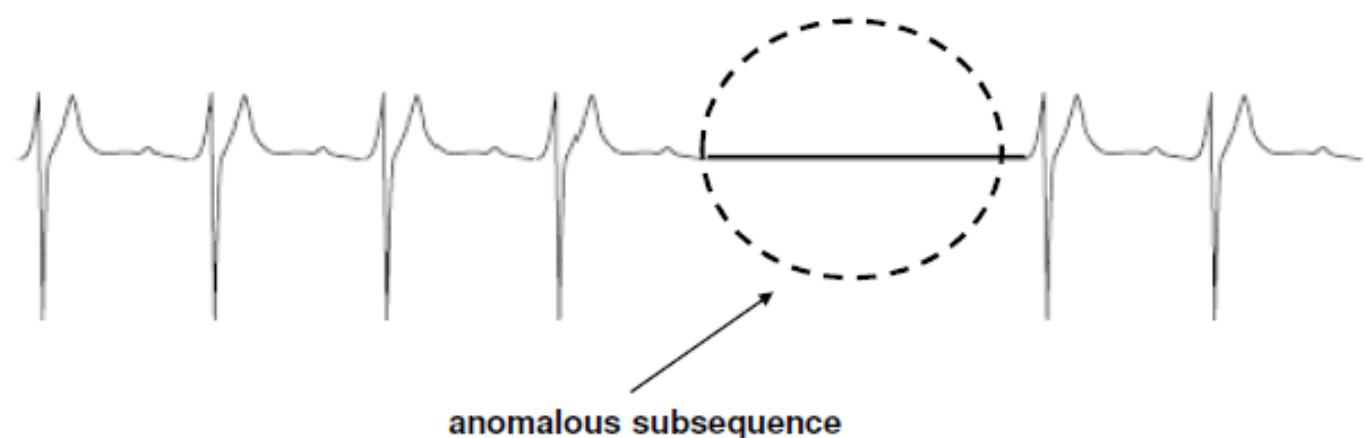
- Point anomalies



- Contextual anomalies



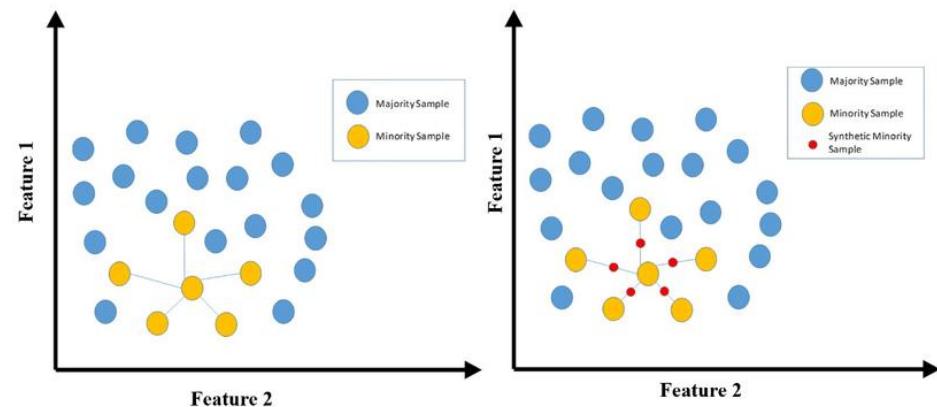
- Collective anomalies



Data labels for anomaly detection

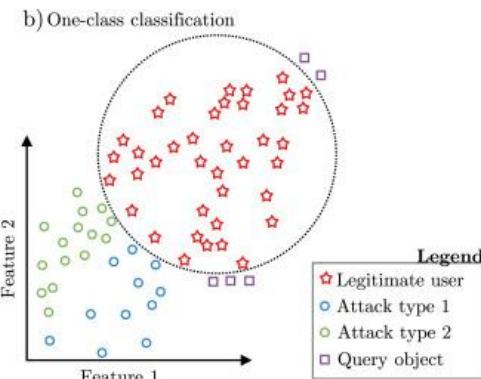
Supervised anomaly detection

- Labels available **for both normal data and anomalies**
- Similar to classification with high class imbalance
- **Data augmentation** solution: SMOTE (Synthetic Minority Oversampling TErchnique) is an oversampling technique where the synthetic samples are generated for the minority class.



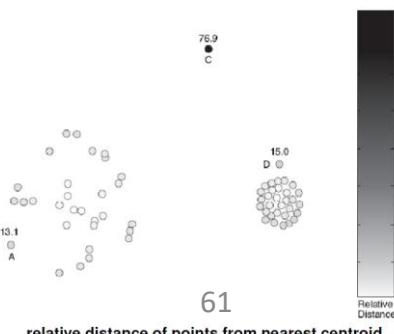
Semi-supervised anomaly detection

- Labels available **only for normal data**
- **One class classification solution**: only learning normal pattern



Unsupervised anomaly detection

- **No labels** assumed
- Based on the assumption that anomalies are very rare compared to normal data

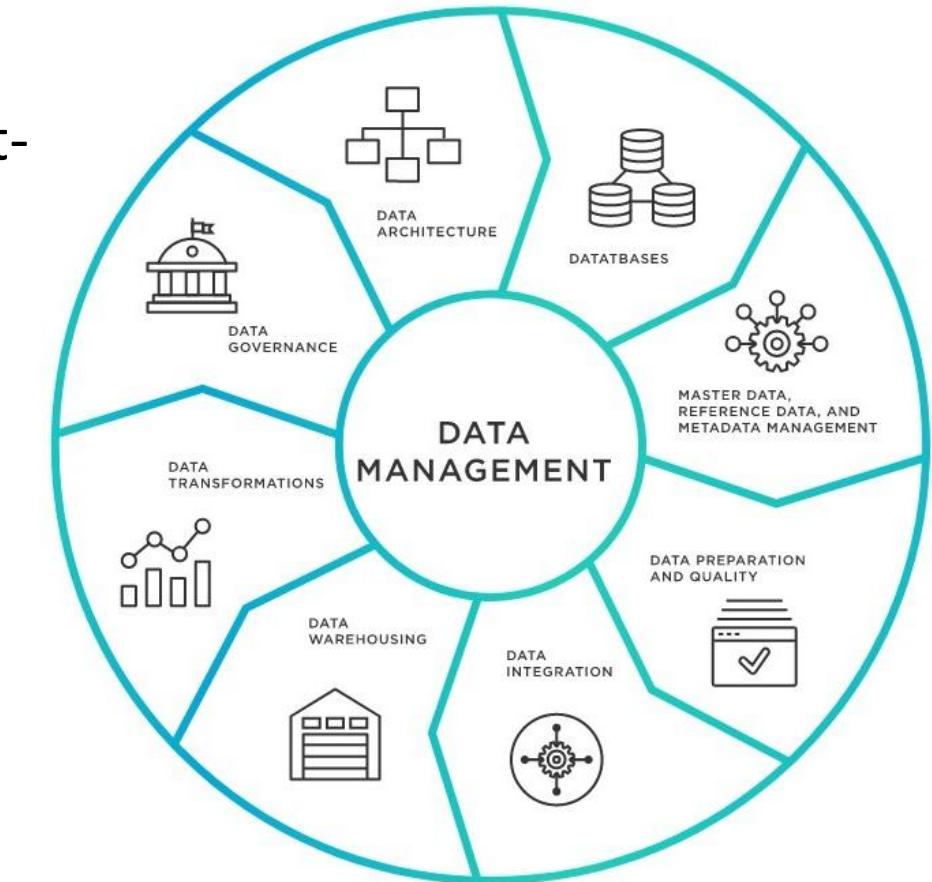


L10: data management

1. What is Data Management
2. Data Quality
3. Data Security
4. Data Privacy

What is data management?

- Data management is the practice of collecting, keeping, and using data securely, efficiently, and cost-effectively.
- As organizations create and consume data at unprecedented rates, data management solutions become essential for making sense of the vast quantities of data.
- Today's leading data management software ensures that reliable, up-to-date data is always used to drive decisions.



Data quality dimensions

Six main dimensions of data quality

Accuracy: The data should reflect actual, real-world scenarios; the measure of accuracy can be confirmed with a verifiable source.

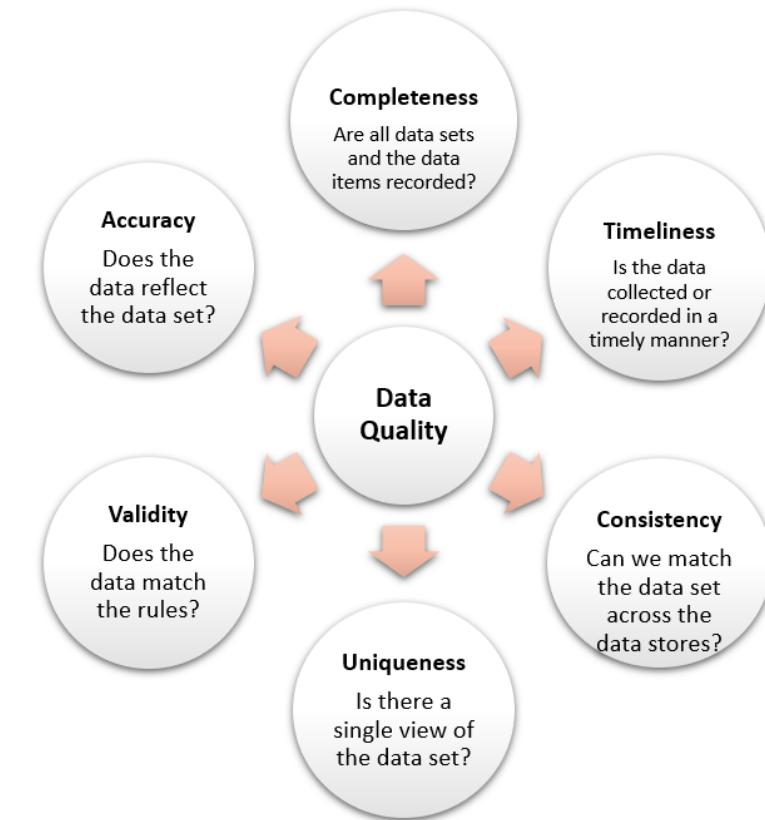
Completeness: Completeness is a measure of the data's ability to effectively deliver all the required values that are available.

Consistency: Data consistency refers to the uniformity of data as it moves across networks and applications. The same data values stored in difference locations should not conflict with one another.

Validity: Data should be collected according to defined business rules and parameters, and should conform to the right format and fall within the right range.

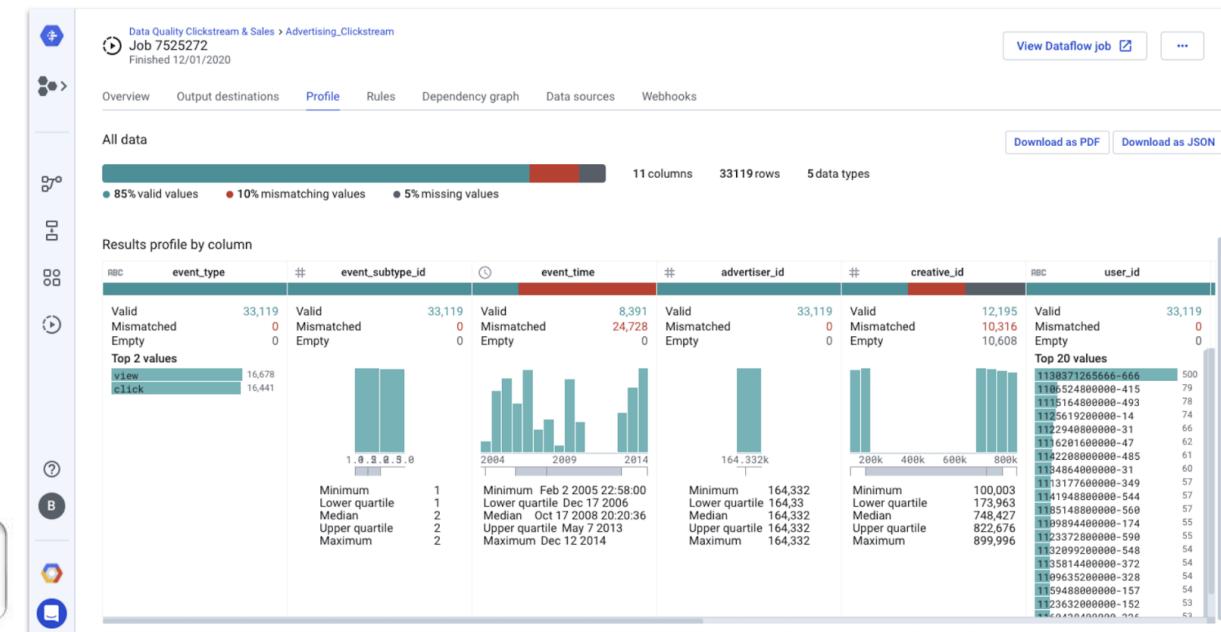
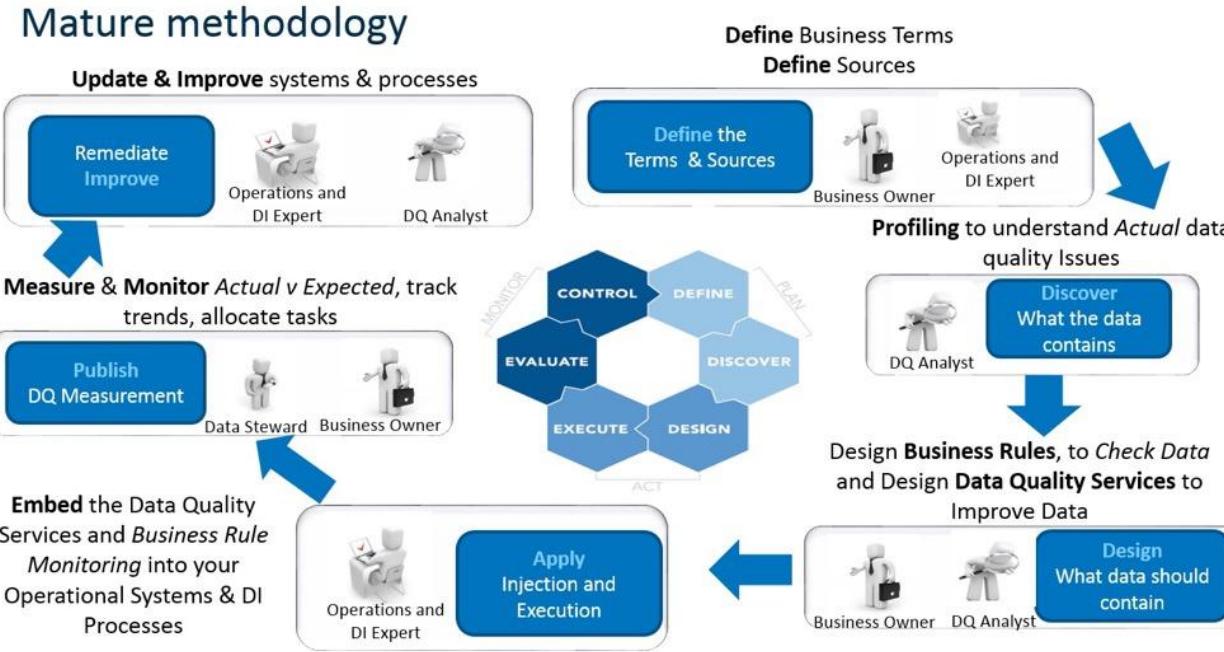
Uniqueness: Uniqueness ensures there are no duplications or overlapping of values across all data sets. Data cleansing and deduplication can help remedy a low uniqueness score.

Timeliness: Timely data is data that is available when it is required. Data may be updated in real time to ensure that it is readily available and accessible.



How to improve data quality? An example

- **Data quality monitoring** - **frequent data quality checks** are essential. Data quality software in combination with machine learning can **automatically detect, report, and correct data variations** based on predefined business rules and parameters.



What is data security?

- **Data security is the process** of safeguarding digital information throughout its entire life cycle to protect it from corruption, theft, or unauthorized access.
 - It covers everything—hardware, software, storage devices, and user devices; access and administrative controls; and organizations' policies and procedures.
- **Data security uses tools and technologies** that enhance visibility of a company's data and how it is being used. These tools can protect data through processes like data masking, encryption, and redaction of sensitive information. The process also helps organizations streamline their auditing procedures and comply with increasingly stringent data protection regulations.

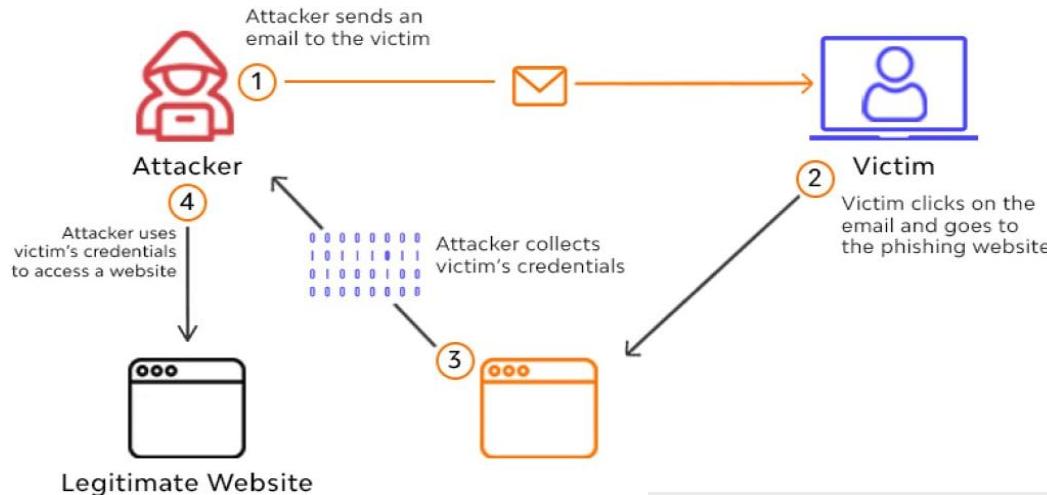
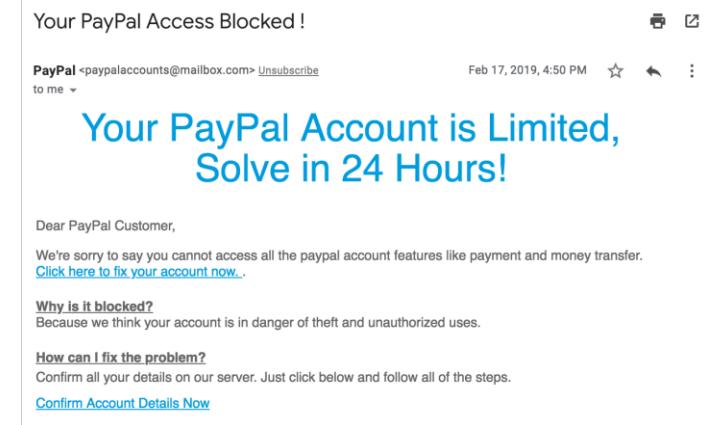
Data security risks example

• Phishing Attacks

In a phishing attack, a **cyber criminal** sends messages, typically via email, short message service (SMS), or instant messaging services, that **appear to be from a trusted sender**. Messages include **malicious links** or attachments that lead recipients to either download **malware** or visit a **spoofed website** that enables the attacker to steal their login credentials or financial information.

• Malware

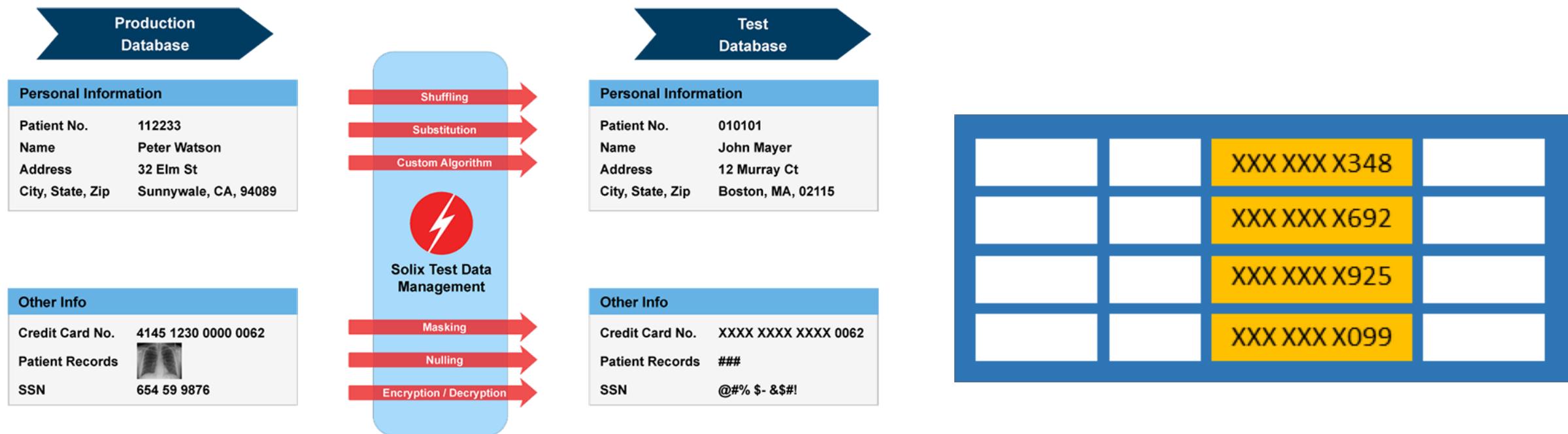
Malicious software is typically spread through email- and web-based attacks. Attackers use malware to **infect computers and corporate networks** by exploiting vulnerabilities in their software, such as web browsers or web applications. Malware can lead to serious data security events like **data theft, extortion, and network damage**.



How to achieve data security? An example

- **Data Masking**

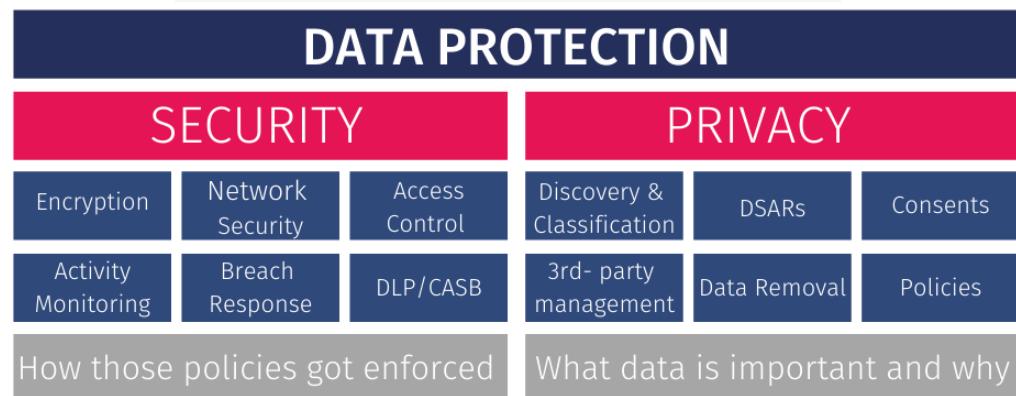
Data masking enables an organization to **hide data by obscuring and replacing specific letters or numbers**. This process is a form of encryption that renders the data useless should a hacker intercept it. The original message can only be uncovered by someone who has the code to decrypt or replace the masked characters.



Data privacy

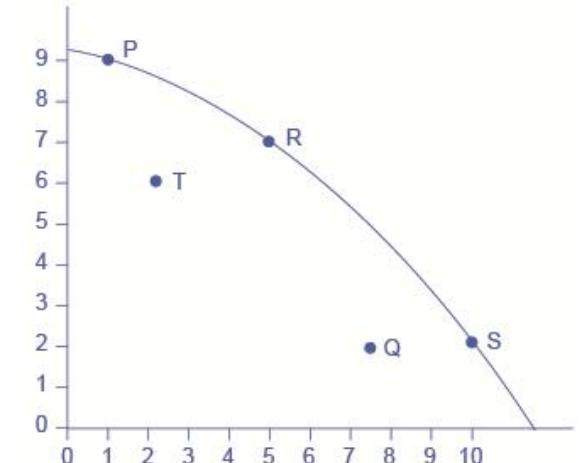
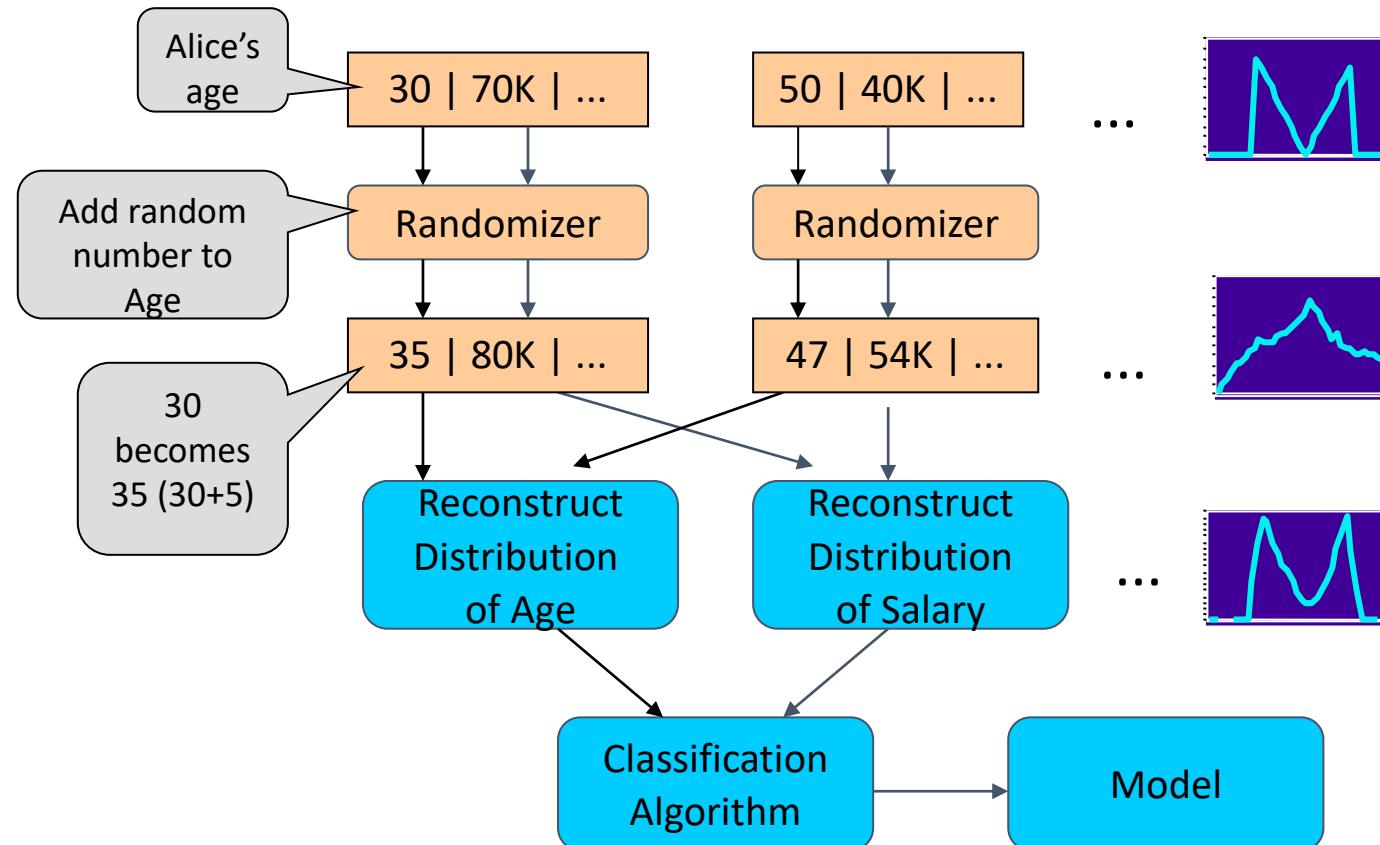
Data privacy generally means the **ability** of a person **to determine** for themselves **when, how, and to what extent personal information about them is shared with or communicated to others, or being used.**

This personal information can be one's name, location, contact information, or online or real-world behavior.



Privacy preserving method example: data perturbation

- Perturb data with value distortion
 - User provides x_i+r instead of x_i
 - r is a random value
 - Uniform, uniform distribution between $[-\alpha, \alpha]$
 - Gaussian, normal distribution with $\mu = 0, \sigma$



Privacy-performance tradeoff
More noises
higher privacy, lower performance

Thanks for your attention!