

# SICSS-Lignan University

## Day 4

### LLM-Agents and Classification

**PEDRO  
SEGUEL**

# TEXT ANALYSIS AND LLM FOR CLASSIFICATION

*Tutorial SICSS*



**Pedro Seguel**

**Support:**  
**Artin Majd, Tharu Yakkala Arachchilage Don, Afeef Shaikh**

# Goals

## Text analysis and LLM for classification

- Setting up the environment and foundations.
- Calling the OpenAI model to respond to a question.
- Calling the OpenAI model to respond and use a Tool.
- Input a CSV file and use the OpenAI model to classify it.

## Calculating the performance of a model

- Confusion Matrices and Evaluation Metrics

## Bonus: Fuzzy Matching for Duplicate Identification

- Setting up a pipeline to identify and inspect duplicates in data.

# Tutorials

Tutorials 1,2, and 3 are available with OpenAI API or Deepseek. You will need to input the KeyChain to work. Tutorials 2 and 3 will also require Tavily KeyChain.

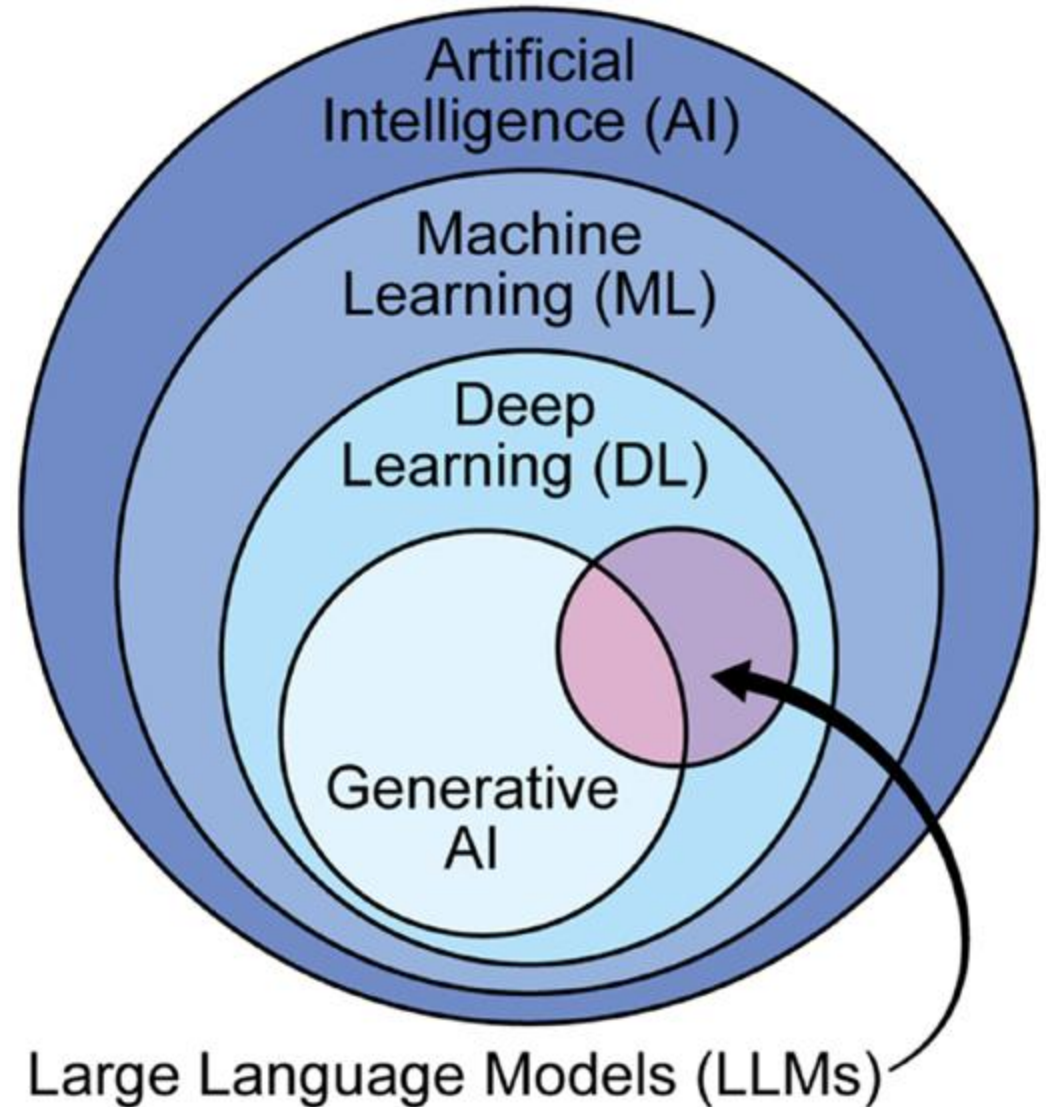
1. Set up and use LLM agents through APIs (e.g., OpenAI, DeepSeek)
  - Customize prompts and manage API calls responsibly
2. Conduct guided web search tasks with agent-based tools
3. Classify structured datasets using agents
4. Evaluating Model Performance with Precision, Recall, and Confusion Matrices.
5. Bonus: Deduplication using ML tools

<https://github.com/SegueLab/tutorial-sicss>

# Overview

# What do we mean by AI, ML, DL, GenAI?

Image Source:  
[https://www.researchgate.net/publication/378394229\\_Large\\_language\\_models\\_a\\_primer\\_and\\_gastroenterology\\_applications](https://www.researchgate.net/publication/378394229_Large_language_models_a_primer_and_gastroenterology_applications)



# What are LLMs?

Large Language Model: An autoregressive Transformer neural Network Trained on a large corpus (hundreds of billions of tokens) and a large parameter space (billions) to predict the next word from a fixed-size context.

- It is usually later aligned to work as a user assistant using techniques such as Reinforcement Learning From Human Feedback or supervised fine-tuning.
- Some are private (access via API or Web Browser): Gemini, ChatGPT, DeepSeek.
- Others are open (model's weights can be downloaded): Llama, Llama2, Falcon, Hugging Face.

# What are LLMs?

- An LLM is a type of text model used in Natural Language Processing, in order to allow computers to process text.
- LLMs are typically built with deep neural networks, using a type of neural network architecture called transformers.
  - They are able to effectively “understand” the context in text, and thereby have more accurate “reasoning” capabilities.
- LLM stands for Large Language Model, because it uses many weights and layers, which are the building blocks of the model.
- A simple NLP model might have a few hundred or thousand weights, while an LLM will have hundreds of thousands or even billions of weight parameters.
  - The most common one that we know is ChatGPT, which is a type of LLM that responds to queries.



# What is an LLM-Agent?

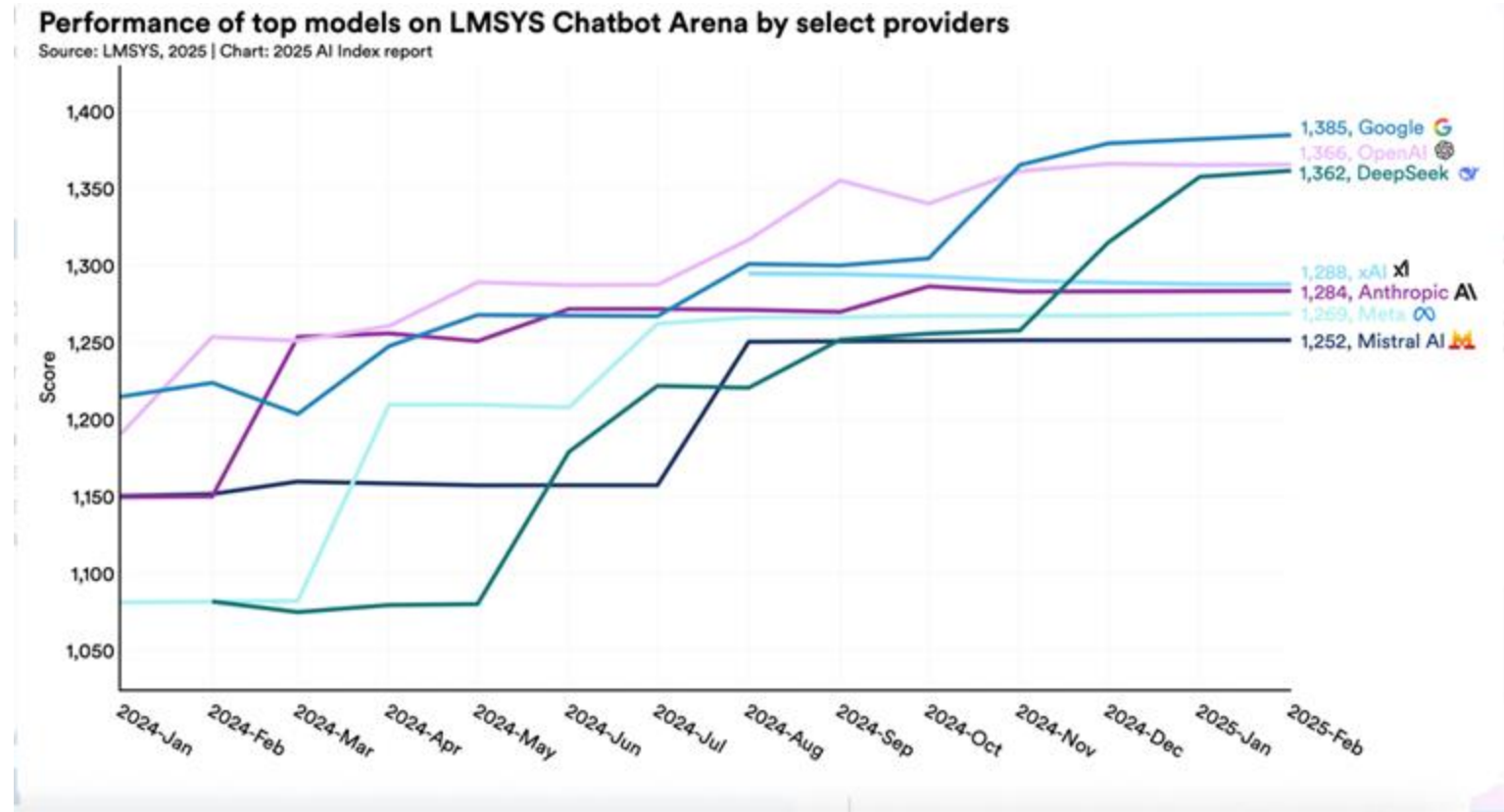
- It's a type of system that goes beyond just a single response.
- It includes LLMs by using reasoning-like features, other tools, and even combining multiple AI APIs.
- At its core, it uses LLMs to solve problems, by also using other tools, and a set of steps to get the final response.
- Example:
  - In the API tutorial with LangChain, we ask it a question, and it then uses a web search tool, first from Tavily to get current results. With this, it proceeds to the OpenAI API, adding the data to the question, and then finally passes it to Chat GPT.

# Industry is racing ahead in AI—but the frontier is tightening.

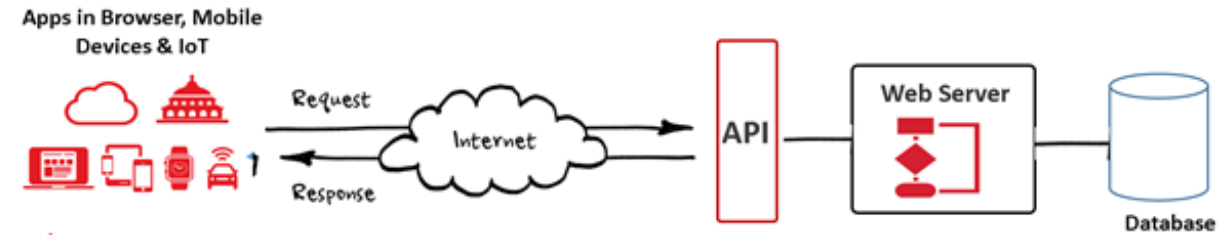
**Nearly 90% of notable AI models in 2024 came from industry**, up from 60% in 2023, while academia remains the top source of highly cited research.

Model scale continues to grow rapidly—training compute doubles every five months, datasets every eight, and power use annually. Yet performance gaps are shrinking: the score difference between the top and 10th-ranked models fell from 11.9% to 5.4% in a year, and the top two are now separated by just 0.7%. The frontier is increasingly competitive—and increasingly crowded.

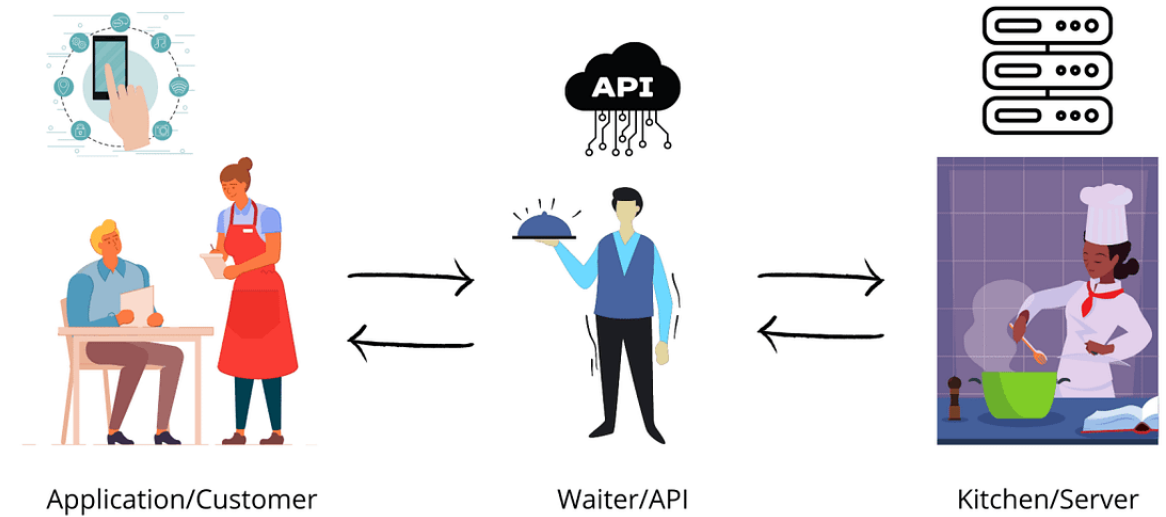
<https://hai.stanford.edu/ai-index/2025-ai-index-report>



# API Tutorials



## What is API ?



<https://medium.com/@pawan329/what-is-an-api-a-step-by-step-guide-e6858b6e1016>

# LLM APIs

- Application Programming Interfaces (APIs) act as translators, enabling seamless exchange between Large Language Models (LLMs) and artificial intelligence (AI) applications.
- These interfaces facilitate the integration of natural language processing (NLP) and natural language understanding capabilities into software systems.
  - The machine learning model leverages its NLP skills—whether it involves content generation, question answering, sentiment analysis, text generation, or text summarization—to produce a response that it relays to the API.
- To access an LLM API, users will need to sign up with their chosen provider and generate API keys for authentication.
- Source: <https://www.ibm.com/think/insights/llm-apis>

# WebApp vs API (ChatGPT vs OpenAI API)

Dimension	ChatGPT (Web App)	OpenAI API
Access Method	Interactive web interface (chat.openai.com)	Programmatic access via HTTP requests
Customization	Limited to settings and instructions	Full control over prompts, temperature, max tokens, etc.
Automation	Manual interaction only	Enables batch processing, integration with pipelines
Use Cases	Exploratory queries, idea generation, live feedback	Data annotation, fine-tuned workflows, tool development
Data Privacy	Inputs stored and may be used to improve models (unless opted out)	Greater control over data handling
Cost Structure	Subscription-based (e.g., ChatGPT Plus)	Pay-as-you-go based on token usage
Model Version Control	Limited (latest model auto-selected for Plus users)	Explicit model selection (e.g., gpt-4, gpt-4-turbo)
Rate Limits	Implicitly managed by OpenAI	Clearly defined rate limits and quotas
Coding/Tool Integration	Not applicable	Seamless integration into Python/R scripts, apps, etc.
Reproducibility	Harder to track specific runs or results	Easier to log and reproduce queries with exact parameters

# API Basics: Securing API Key

Method 1 (What we will use):

- Getpass library - allows us to enter the key manually without putting it into the code or anywhere in the notebook.

Learn more here:

- <https://docs.python.org/3/library/getpass.html>
- <https://medium.com/@xiajiun/using-getpass-getpass-to-securely-save-api-keys-in-an-environment-variable-8b2d131d7034>

Method 2:

- Using a .env file, which has the key saved as OPENAI\_API\_KEY = "KEY"
- .env is added to .gitignore so it's not committed.
- Loaded through the dotenv and os libraries.

More about it here:

- <https://jonathansoma.com/lede/foundations-2019/classes/apis/keeping-api-keys-secret/>

# OpenAI API: Basics

- (1) Sign up and get an API key.
- (2) Create the OpenAI object

```
import openai
import getpass

api_key = getpass.getpass("Enter your OpenAI API key: ")
client = openai.OpenAI(api_key=api_key)
```

Learn more at:

- <https://help.openai.com/en/articles/4936850-where-do-i-find-my-openai-api-key>
- <https://api-docs.deepseek.com/>
- OpenAI Python Library:  
<https://pypi.org/project/openai/0.26.5/>

```
response = client.chat.completions.create(
    model="gpt-3.5-turbo", # You can change this to "gpt-4" if your account has access
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Hello, how can I learn Python?"}
    ]
)

print(response.choices[0].message.content)
```

# OpenAI API: Response parameters

- “role”: “system”
  - What do you want the LLM to be like?
  - The behaviour or tone of the LLM.
- “role”: “user”
  - Your question.
  - Simulates the back -and- forth conversation.

```
response = client.chat.completions.create(  
    model="gpt-3.5-turbo", # You can change this to "gpt-4" if your account has access  
    messages=[  
        {"role": "system", "content": "You are a helpful assistant."},  
        {"role": "user", "content": "Hello, how can I learn Python?"}  
    ]  
)  
  
print(response.choices[0].message.content)
```



# Use Cases For Classification

- Sentiment analysis.
- Fraud detection.
- Credit risk.
- Document sorting - knowledge management.
- Cybersecurity malware or malicious code detection.
- Reading medical notes and diagnosing diseases.
- Fake news detection.

Source: <https://www.projectpro.io/article/large-language-model-use-cases-and-applications/887>

# Prompt Engineering: Guiding the Language Model

Prompt engineering, or “Prompting,” is the discipline of crafting effective prompts to guide the Language Model (LM) towards generating accurate responses. Some common prompting guidelines:

- **Clarity and Conciseness:** Clearly articulate the prompt to minimize ambiguity and ensure that the LM understands the task at hand.
- **Use specific Examples:** Provide specific examples within the prompt to give the LM context.
- **Role-Based Prompts:** Incorporate into prompts (e.g., a tour guide, a teacher, a doctor, a salesperson)
- **Desired Output Specification:** Specify the desired output format (e.g., JSON, HTML, CSV, markdown, LaTeX)

# Prompt design: Good practices, frameworks and resources

- Best practices from OpenAI
  - (1) latest model - OpenAI suggests that prompt engineering is easier with latest models.
  - (2) Put instructions at the top.
  - (3) Separate context or additional info in triple quotation marks or hashtags.
    - Ex. What sport does this team belong to? `""" team: maple leaves"""`
  - (3) Avoid ambiguity, provide as much info as possible, and be specific.
  - (4) Add example outputs.
- These are only a few recommendations made from OpenAI you can find more on their help site:  
<https://help.openai.com/en/articles/6654000-best-practices-for-prompt-engineering-with-the-openai-api>

See specific advice for DeepSeek here:

- <https://docs.together.ai/docs/prompting-deepseek-r1>

# Risks, and considerations

- Hallucinations
  - The responses generated by these LLMs/Agents might not be factual information.
  - They may seem like a good idea, but we must be cautious, especially when providing advice that could lead to legal complications, such as medical or financial advice.
- Dependency on an API
  - Bad actors could compromise APIs, which they can use to send false information or malicious code.
  - If the API goes down or is updated, it might cause problems for software or apps that depend on it, which might require changes to the original code.
- Prompt injection and privilege escalation.
  - The prompts could be used to trick the Agents, and attackers could get access to sensitive data. They could also use these prompts to get admin privileges.

# The Responsible AI (RAI) ecosystem evolves—unevenly.

## Reported safety and responsible AI benchmarks for popular foundation models

Source: AI Index, 2025 | Table: 2025 AI Index report

Responsible AI benchmark	o1	GPT-4.5	DeepSeek-R1	Gemini 2.5	Grok-2	Claude 3.7 Sonnet	Llama 3.3
BBQ	✓	✓				✓	
HarmBench							
Cybench						✓	
SimpleQA			✓	✓			
Toxic WildChat	✓	✓				✓	
StrongREJECT	✓	✓					
WMDP benchmark	✓	✓					
MakeMePay	✓	✓					
MakeMeSay	✓	✓					

AI-related incidents are rising sharply, yet standardized RAI evaluations remain rare among major industrial model developers. However, new benchmarks like HELM Safety, AIR-Bench, and FACTS offer promising tools for assessing factuality and safety. Among companies, a gap persists between recognizing RAI risks and taking meaningful action. In contrast, governments are showing increased urgency: In 2024, global cooperation on AI governance intensified, with organizations including the OECD, EU, U.N., and African Union releasing frameworks focused on transparency, trustworthiness, and other core responsible AI principles.

Source: <https://hai.stanford.edu/ai-index/2025-ai-index-report>

# Beware of “Botshit”

Figure 1. Four modes of chatbot work

Response veracity importance	Crucial	<b>Authenticated chatbot work</b> Users skeptically submit tasks to chatbots and then meticulously verify responses for factual accuracy, logical coherence, and truthfulness. Examples include legal, safety, and budgetary tasks.	<b>Automated chatbot work</b> Users systematically assign routine and standard tasks to chatbots and then use responses for efficient and detached execution. Examples include application assessment and selection tasks.
	Unimportant	<b>Augmented chatbot work</b> Users openly prompt chatbots to generate ideas and concepts and then evaluate, organize, combine, and select from the generated responses. Examples include brainstorming and idea-generation tasks.	<b>Autonomous chatbot work</b> Users selectively delegate tasks to chatbots with domain training and expertise and then allow the chatbots to learn and adapt. Examples include support and assistance tasks.
		Difficult to verify	Easy to verify
		Response veracity verifiability	

“When humans uncritically use this untruthful content, it becomes what we call botshit.”

Hannigan, T. R., McCarthy, I. P., & Spicer, A. (2024). Beware of botshit: How to manage the epistemic risks of generative chatbots. *Business Horizons*, 67(5), 471-486.



## Beware of botshit: How to manage the epistemic risks of generative chatbots



Timothy R. Hannigan<sup>a,b</sup>, Ian P. McCarthy<sup>c,d,\*</sup>, André Spicer<sup>e</sup>

<sup>a</sup> Alberta School of Business, University of Alberta, Edmonton, AB, Canada  
<sup>b</sup> Telfer School of Management, University of Ottawa, Ottawa, ON K1N 6N5, Canada  
<sup>c</sup> Beedie School of Business, Simon Fraser University, 500 Granville Street, Vancouver, BC V6C 1W6, Canada  
<sup>d</sup> Luiss, Viale Romania, 32 00197 Roma, Italy  
<sup>e</sup> Bayes Business School, City University of London, 106 Bunhill Row, London EC1Y 8TZ, UK

**KEYWORDS**  
Chatbots;  
Bullshit;  
Botshit;  
Artificial intelligence;  
Natural language processing

**Abstract** Advances in large language model (LLM) technology enable chatbots to generate and analyze content for our work. Generative chatbots do this work by predicting responses rather than knowing the meaning of their responses. In other words, chatbots can produce coherent-sounding but inaccurate or fabricated content, referred to as *hallucinations*. When humans uncritically use this untruthful content, it becomes what we call *botshit*. This article focuses on how to use chatbots for content generation work while mitigating the *epistemic* (i.e., the process of producing knowledge) risks associated with *botshit*. Drawing on risk management research, we introduce a typology framework that orients how chatbots can be used based on two dimensions: response veracity verifiability and response veracity importance. The framework identifies four modes of chatbot work (*authenticated*, *automated*, *augmented*, and *autonomous*) with a botshit-related risk (*ignorance*, *miscalibration*, *routinization*, and *black boxing*). We describe and illustrate each mode and offer advice to help chatbot users guard against the botshit risks that come with each mode. © 2024 Kelley School of Business, Indiana University. Published by Elsevier Inc. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

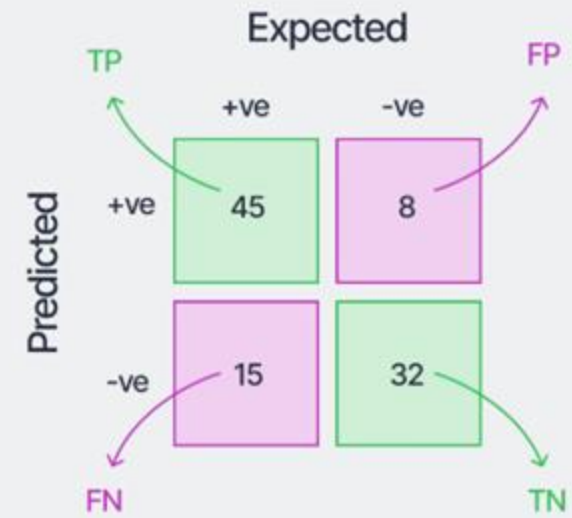
### 1. Who’s a pretty Polly?

On November 20, 2022, OpenAI released its chatbot, Chat Generative Pre-trained Transformer (ChatGPT), for public use. This AI-driven chatbot

generates text responses to human prompts, such as questions and requests. ChatGPT can undertake plagiarism checks, create content such as proposals, stories, applications, reviews, and jokes, and perform programming work such as creating and debugging code. In January 2023, Salesforce CEO Marc Benioff (2023) announced: “Just promoted #ChatGPT to our management team. It’s an

\* Corresponding author  
E-mail address: imccarth@sfu.ca (I.P. McCarthy)

# Validation Methods



# Evaluation: A starting point

- Confusion matrix:
  - A matrix that contains the labels that the model got right and wrong.
  - Accuracy: Of all the labels, the proportion that it got right.
- For each label, there are 3 measures: precision, recall, and F1-score.
- Let's use the positive labels as an example in binary classification (i.e., TRUE/FALSE or YES/NO).
  - Precision: Of all the labels the model predicted as positive, how many were actually positive?
  - Recall: Of all the positive labels, how many did the model correctly identify?
  - F1-score: The harmonized mean of the precision and recall.
- We can then repeat this process for the negative labels.

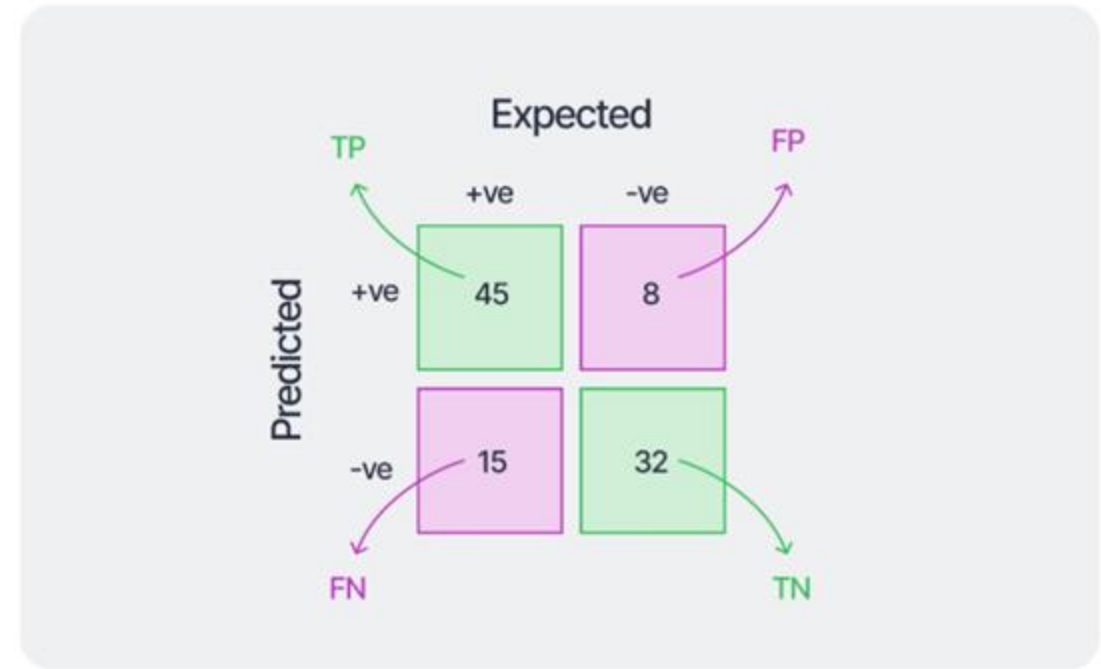


# Evaluation: Example (+ve labels)

$$\begin{aligned}\text{Precision} &= \text{TP} / (\text{TP} + \text{FP}) \\ &= 45 / (45 + 8) \\ &= 0.85\end{aligned}$$

$$\begin{aligned}\text{Recall} &= \text{TP} / (\text{TP} + \text{FN}) \\ &= 45 / (45 + 15) \\ &= 0.75\end{aligned}$$

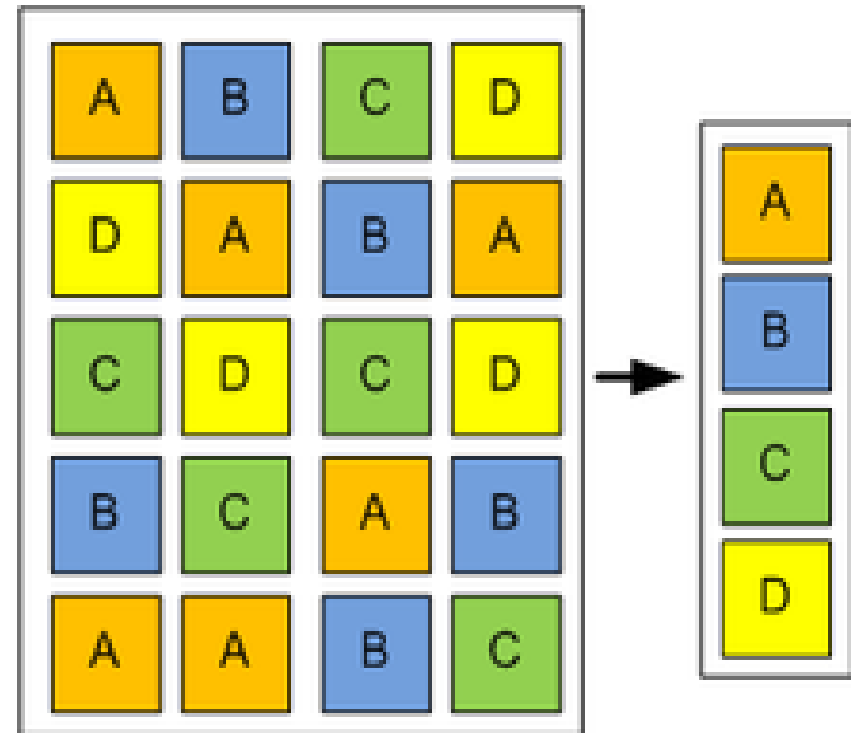
$$\begin{aligned}\text{F1-score} &= (2 \times P \times R) / (P + R) \\ &= (2 \times 0.85 \times 0.75) / (0.85 + 0.75) = 0.80\end{aligned}$$



# Building quality datasets from Data Annotators

- There is another type of evaluation that is applied to data that two or more annotators have labeled.
- There are two ways we can measure this: the Cohen Kappa Score and the Krippendorff Alpha.
- **The Kappa Score** measures the reliability between two raters.
- **Krippendorff Alpha** - measures reliability with two or more raters.
- Both are in the range of -1 (huge disagreement) to 1 (perfect agreement)
- You can use these if you create your own datasets in order to test their validity.

# Bonus: Deduplication



# Deduplication

Refers to the process of identifying and removing duplicate data or records from a dataset. In essence, it's the act of eliminating redundant copies of the same information, streamlining storage, and potentially improving efficiency.

How it works:

- Deduplication tools analyze data to identify redundant chunks (blocks of data) and replace them with a reference to the single stored copy.

Examples:

- In a file system, deduplication can eliminate multiple copies of the same image file or PDF. In a database, it can remove duplicate customer records.

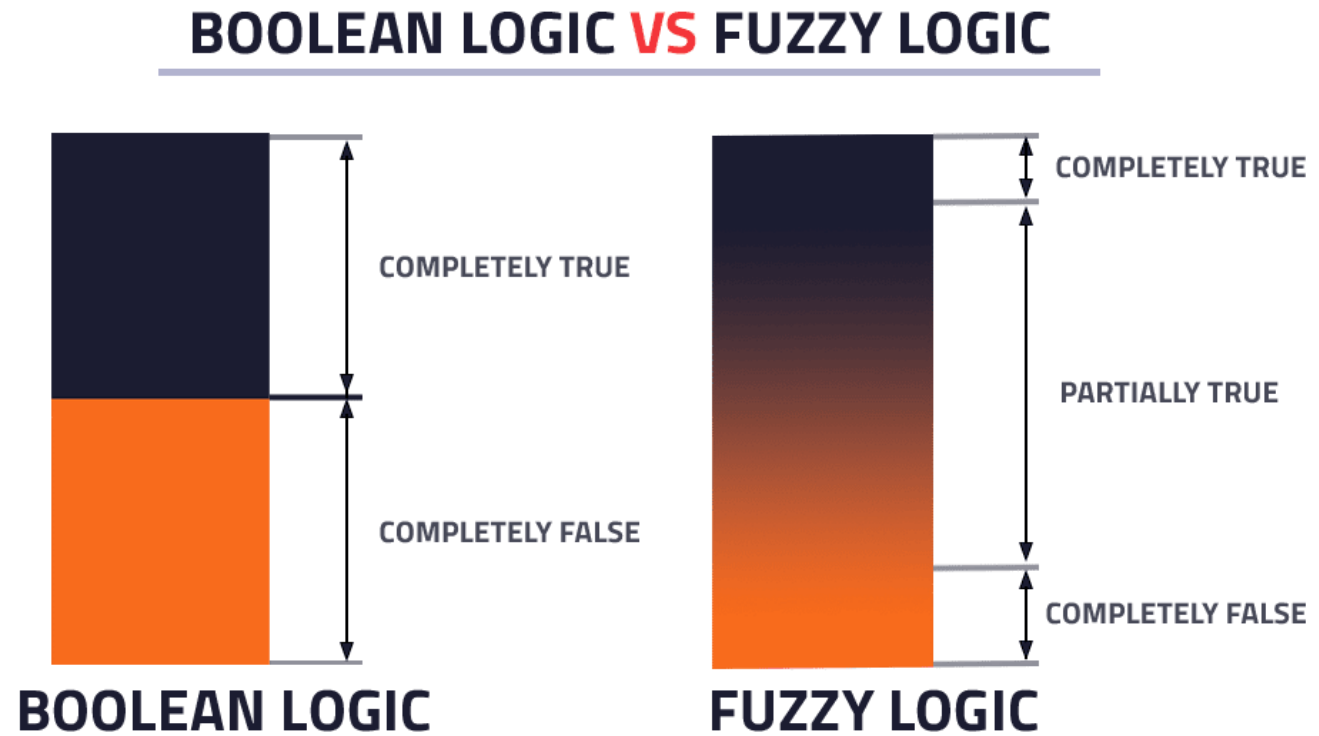
# How do we find a match between *strings*?

## Boolean Logic

- Boolean logic string matching involves using operators like AND, OR, and NOT to combine keywords and refine search results.

## Fuzzy Matching

- Fuzzy string matching is the process of finding strings that approximately match each other.



# Fuzzy Matching

Aple


Appl

Apples



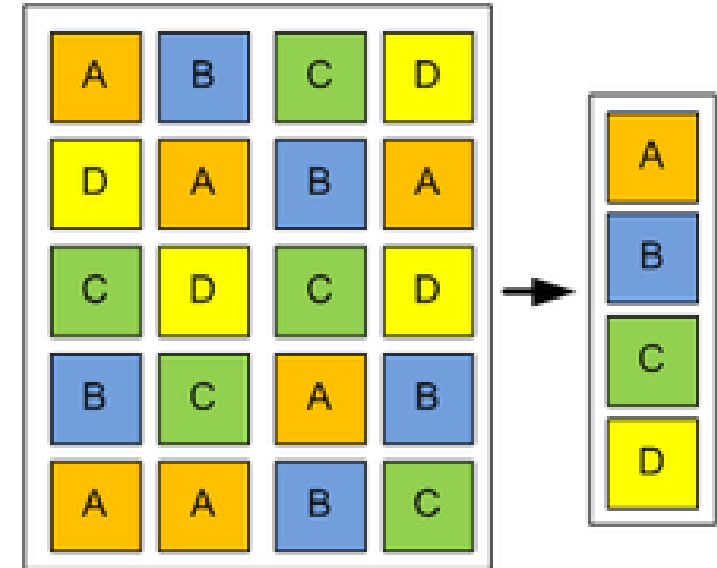
Apple

# Matching Organization Names - Example

- Why it matters:  
Clean names = accurate stats, trends, and firm-level analysis.
- The mess in real data  :
  - "Google Inc.", "GOOGLE", "Gogle", "Google, LLC" - all the same org!
- Manual effort?
  - Not scalable beyond a few hundred names.
- Exact matching?
  - Fails on typos, spacing, and punctuation.
- *We need smarter matching that understands “close enough.”*

# Matching Organization Names: Fuzzy Matching & Dedupe

- Fuzzy Matching:
  - Compares the similarity between strings, not just exact equality.
  - Uses algorithms like Levenshtein Distance (how many edits are needed).
- Why do we need it?
  - Handles typos, abbreviations, punctuation, casing, spacing, etc.
- Dedupe Library:
  - Open-source Python tool
  - Trains on labeled examples
  - Clusters of similar names
  - Picks a canonical (clean) name



## Cluster: amazon

- amaz0n
- ama zon
- AMAZON INC
- amaZOn



# Matching Organization Names

Core steps for matching and cleaning names with Dedupe

## 1. Clean text

→ lowercase, remove punctuation/suffixes

## 4. Train model

→ learns patterns from your labels

## 2. Format for dedupe

→ dictionary with IDs and cleaned names

## 5. Cluster names

→ based on similarity

## 3. Label examples

→ match vs distinct

## 6. Assign canonical names

→ map each name to its best version

- Bonus:

- Can apply to millions of rows
- Saves huge manual effort

# Matching Organization Names

Dedupe Clusters from Our Dataset

	emp	cluster_id	confidence_score	employer_original	canonical_name
1	gogle	1	0.804	Gogle	google
0	google	1	0.804	Google	google
4	amazon	2	0.973	Amazon	amazon
3	amazoncom	2	0.948	Amazon.com	amazon
2	amazone	2	0.948	Amazone	amazon
5	tesla	3	0.779	Tesla	tsla
7	teslla	3	0.784	Teslla	tsla
6	tsla	3	0.710	TSLA	tsla
8	meta	4	1.000	Meta	meta
9	oracle	5	1.000	Oracle	oracle
10	apple	6	1.000	Apple	apple
11	facebook	7	1.000	Facebook	facebook
12	ibm	8	1.000	I.B.M.	ibm
13	netflix	9	1.000	Net-flix	netflix
14	microsoft	10	1.000	Microsoft	microsoft
15	international business machines	11	1.000	International Business Machines	international business machines