Unleashing the power of Generative AI for tabular data in ML applications

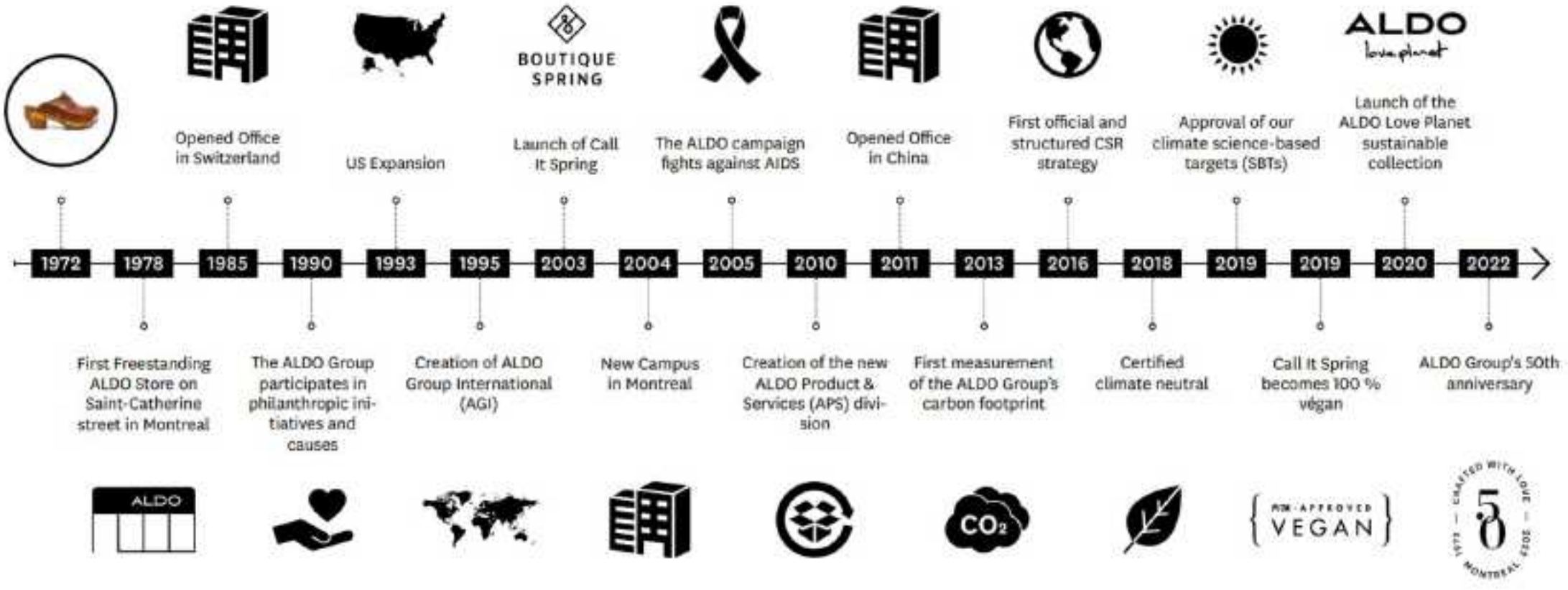
Transforming the Data Science Lifecycle with Generative Models

by Fatih Nayebi, Ph.D. | Big Data & AI Toronto | October 18, 2023

About Fatih

- Head of Data, Analytics & AI at the ALDO Group
- Professor at McGill University
- Ph.D. in Engineering specialized in ML & HCI from École de technologie supérieure
- Author of Swift Functional Programming books published by Packt

About ALDO Group



Agenda

- Introduction to Different Forms of Learning
- Generative AI & Large Language Models
- Generative vs. Predictive vs. Prescriptive Models
- Can Prediction be a Form of Generation?
- In a nutshell: Gen AI for ML Apps
- Transforming the Decision Science Lifecycle
- Advancements in LLMs for Tabular Data Processing & Prediction
- Ethical and Practical Considerations

1. Introduction to Different Forms of Learning

Supervised Learning: Utilizing labeled data to train models.

Unsupervised Learning: Discovering patterns in data without prior labeling.

Reinforcement Learning: Learning from trial and error, guided by rewards.

Semi-supervised Learning: Leveraging a small amount of labeled data to guide learning in a larger dataset of unlabeled data.

Self-supervised Learning: Learning from data without human-provided labels while enabling the model to generate its own labels.

2. Generative AI & Large Language Models

Generative AI

- AI models capable of generating new data (Text, image, voice, and video) and resembling a given dataset.

Rise of Large Language Models (LLMs)

- **Language Representation Model:**
  - Pre-trained on vast text corpora.
- **Zero-shot Model:**
  - Performs tasks without specific training data.
  - Generalizes and makes predictions for tasks it has never seen before.
- **Multimodal Model:**
  - Understands and generates content across different modalities
  - Useful for image captioning and text-based image retrieval.
- **Fine-tuned or Domain-specific Models:**
  - Improve performance for specific tasks or domains.

3. Generative vs. Predictive vs. Prescriptive Models

Predictive Models

- Predict outputs based on inputs.
- Emphasizes forecasting and pattern recognition.

Prescriptive Models

- Focus on decision-making with a blend of optimization and causal inference.
- Provides actionable recommendations by considering causal relationships and optimizing outcomes.

Generative Models

- Learn and generate data that closely resembles a specified distribution.
- Captures underlying data distribution for creativity and diversity.

4. Can Prediction be a Form of Generation?

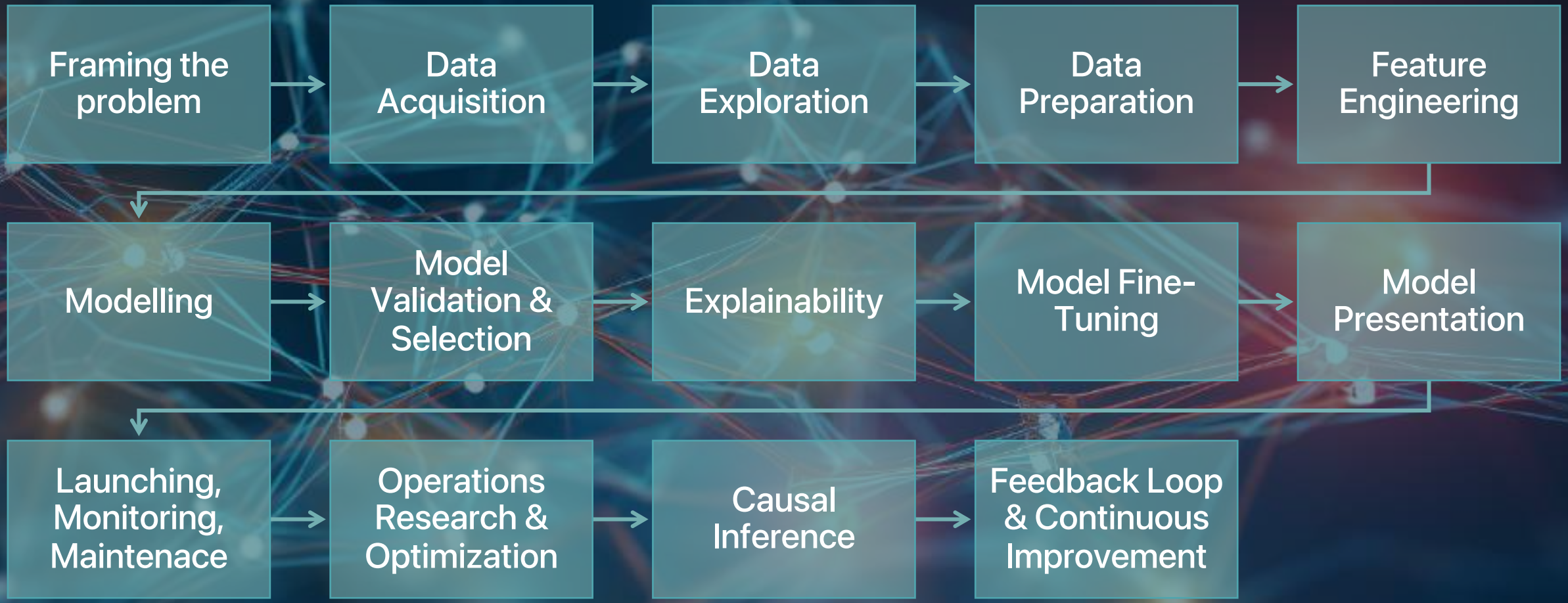
- **Philosophical Standpoints**
  - Prediction is a constrained form of generation, guided by past data, lacking the creative freedom inherent in generative processes.
- **Differentiating Prediction from Generation**
  - Metrics
    - Generative: Evaluates novelty and diversity.
    - Predictive: Prioritizes accuracy and precision.
  - Evaluation
    - Understanding the distinct methodologies for assessing predictive and generative models.
  - Applications

5. In a nutshell: Gen AI for ML Apps

Learning & Research	**Better Communication**	**Code Generation**
Data Augmentation, Synthesis, and Enrichment	**Data Anonymization**	**Insight Generation**
Question Answering	**Data Labeling**	**Summarization**

6. Transforming the Decision Science Lifecycle*

* An Extension of Aurélien Géron's Machine Learning Project Checklist



6.1. Framing the problem

Insight Generation

Process and generate text to synthesize insights from various data sources, helping in crafting a well-defined problem statement.

Stakeholder Communication

Generating intuitive reports and summaries, aiding in aligning objectives and expectations.

6.2. Data Acquisition

Automated Data Collection

Code generation for SQL, API calls, Event Sourcing, PySpark, etc.

Generating Synthetic Data

Implementing Generative Adversarial Network (**GAN**) architectures

Employing packages such as **YData Synthetic**

6.2.1. SQL Query Generation with gpt-3.5-turbo

```python
import pandas as pd
from datetime import date
import openai

# Assume you have OpenAI API access
openai.api_key = 'your-api-key'

# Load a public retail dataset (for demonstration purposes, we'll assume a simple dataset)
# Assume the dataset has columns like 'Product', 'Sales', 'Date'
data = pd.read_csv('retail_dataset.csv')

def generate_query(task_description):
    knowledge_cutoff = "2021-09-01"
    today = date.today()
    current_date = today.strftime("%Y-%m-%d")

    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        messages=[
            {
                "role": "system",
                "content": f"  \n You are a helpful assistant. Knowledge cutoff: {knowledge_cutoff} "
                           f"Current date: {current_date}",
            },
            {
                "role": "user",
                "content": f"Translate the following task into an SQL query: {task_description}"
            },
        ],
        max_tokens=100,
    )
    return response.choices[0].message.content

# Task description for data collection
task_description = "Retrieve all records of sales in the month of December 2022."
# Get SQL query
query = generate_query(task_description)
# Execute SQL query to collect data
data = pd.read_sql_query(query, conn)
print(data.head())
conn.close()
```

6.3. Data Exploration

Automated Insights

Sifting through the data, generating automated insights and visualizations.

Textual Data Exploration

Unveiling insights from textual data by summarizing, categorizing, and extracting key themes.

Leveraging **PandasAI** or **Sketch**

Conversational interface that is synergized with Pandas, permitting getting insights about the data

6.3.1. PandasAI Example

```python
from pandasai import SmartDatalake
from pandasai.llm import OpenAI

employees_data = {
    'EmployeeID': [1, 2, 3, 4, 5],
    'Name': ['John', 'Emma', 'Liam', 'Olivia', 'William'],
    'Department': ['HR', 'Sales', 'IT', 'Marketing', 'Finance']
}

salaries_data = {
    'EmployeeID': [1, 2, 3, 4, 5],
    'Salary': [5000, 6000, 4500, 7000, 5500]
}

employees_df = pd.DataFrame(employees_data)
salaries_df = pd.DataFrame(salaries_data)


llm = OpenAI()
dl = SmartDatalake([employees_df, salaries_df], config={"llm": llm})
dl.chat("Who gets paid the most?")
```

6.4. Data Preparation & Feature Engineering

Leveraging code generation with **Prompt Engineering** techniques

AUTOMATED DATA CLEANING

SYNTHETIC DATA IMPUTATION

AUTOMATED FEATURE ENGINEERING

TEXTUAL DATA PROCESSING

6.5. Modelling

Leveraging code generation with **Prompt Engineering** techniques

Automated Model Selection & Tuning

Automated Validation

Insightful Diagnostics

Narrative Explanations

Customized Visualizations

6.5.1. CAAFE: LLMs for Semi-Automated Data Science

```python
import pandas as pd
from caafe import CAAFEClassifier # Automated Feature Engineering for tabular datasets
from tabpfn import TabPFNClassifier # Fast Automated Machine Learning method for small tabular datasets
from sklearn.ensemble import RandomForestClassifier

clf_no_feat_eng = RandomForestClassifier(n_estimators=100, max_depth=2)
caafe_clf = CAAFEClassifier(
    base_classifier=clf_no_feat_eng,
    llm_model="gpt-4",
    iterations=2
)

# Example dataset description for kaggle_health-insurance-lead-prediction
dataset_description = """
For the data and objective, it is evident that this is a Binary Classification Problem data in the Tabular Data format.
A policy is recommended to a person when they land on an insurance website, and if the person chooses to fill up a form
to apply, it is considered a Positive outcome (Classified as lead). All other conditions are considered Zero outcomes.
"""
caafe_clf.fit_pandas(
    df_train,
    target_column_name=target_column_name,
    dataset_description=dataset_description
)

pred = caafe_clf.predict(df_test)
print(caafe_clf.code)
```

6.6. Optimization & Operations Research

- Leveraging code generation with **Prompt Engineering** techniques:
  - **Automated Optimization**
  - **Scenario Forecasting**
- Leveraging
  - **OPRO (Large Language Models as Optimizers)** by **Google DeepMind**
  - **Connecting Large Language Models with Evolutionary Algorithms yields powerful prompt optimizers** by **Qingyan Guo** et al.

6.6.1. OPRO (Large Language Models as Optimizers) by Google DeepMind

I have some texts along with their corresponding scores. The texts are arranged in ascending order based on their scores, where higher scores indicate better quality.

text:
Let's figure it out!
score:
61

text:
Let's solve the problem.
score:
63

(... more instructions and scores ...)

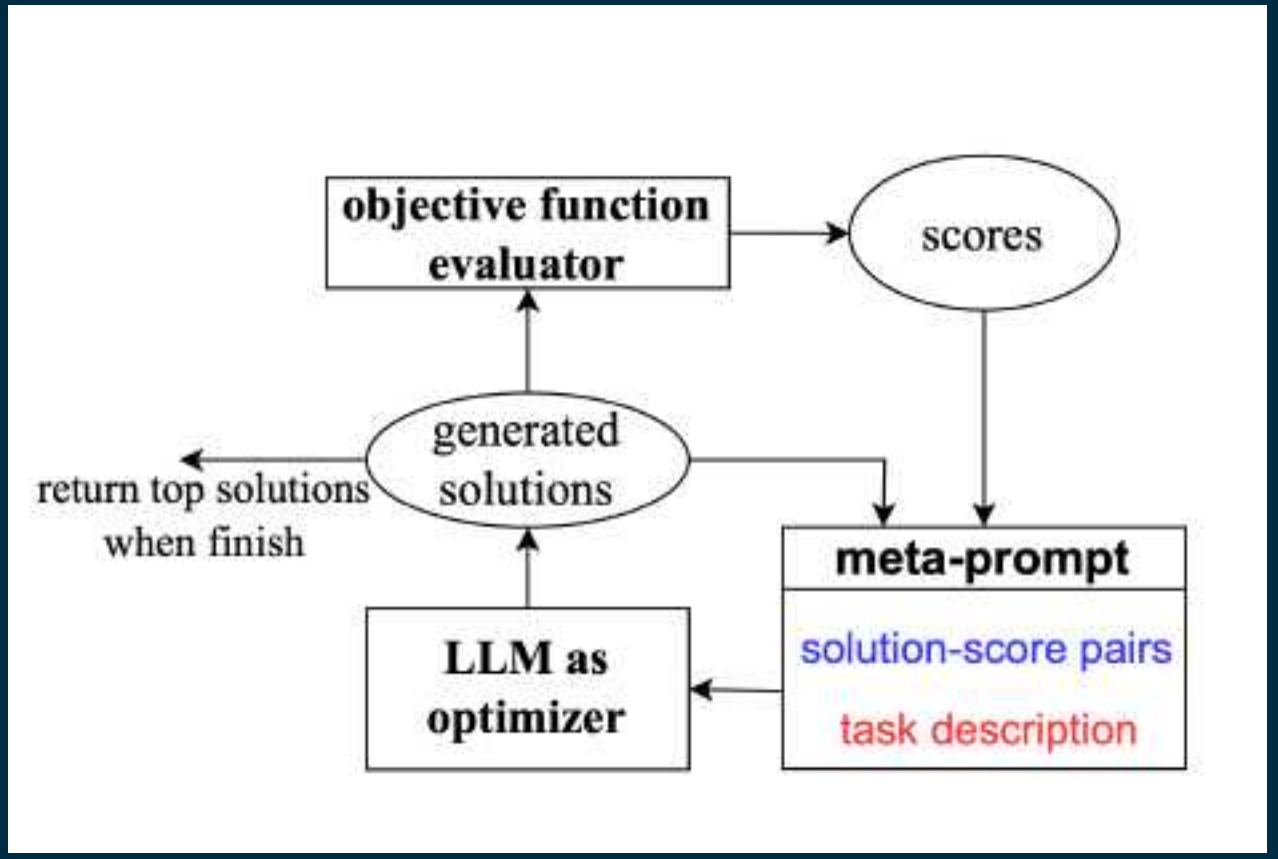
The following exemplars show how to apply your text: you replace <INS> in each input with your text, then read the input and give an output. We say your output is wrong if your output is different from the given output, and we say your output is correct if they are the same.

input:
Q: Alannah, Beatrix, and Queen are preparing for the new school year and have been given books by their parents. Alannah has 20 more books than Beatrix. Queen has 1/5 times more books than Alannah. If Beatrix has 30 books, how many books do the three have together?
A: <INS>
output:
140

(... more exemplars ...)

Write your new text that is different from the old ones and has a score as high as possible. Write the text in square brackets.

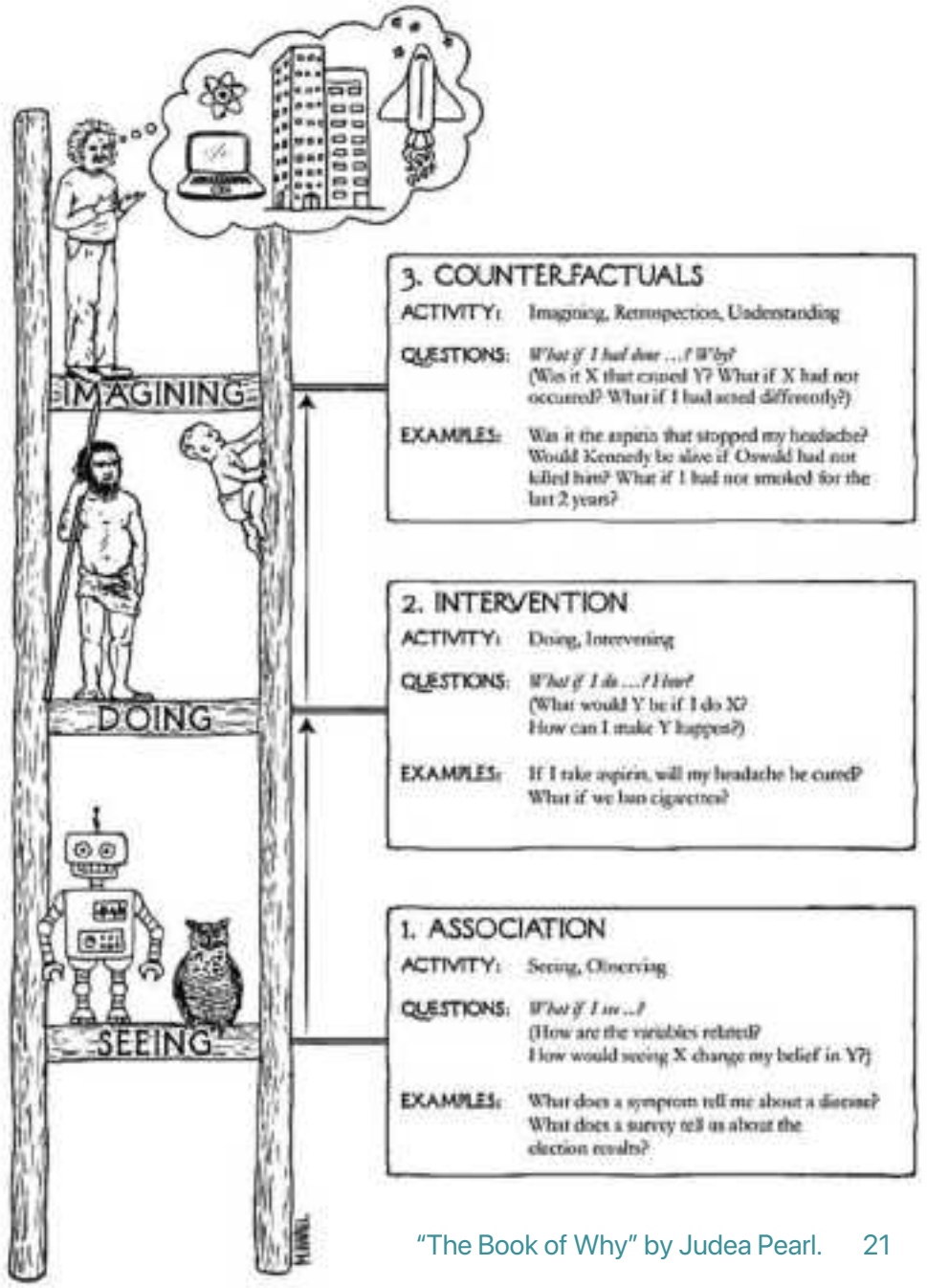
Meta-prompt



An overview of the OPRO framework

6.7. Causal Inference

- **Hypothesis Generation**:
  - Generating hypotheses regarding potential causal relationships.
- **Causal Analysis**:
  - Assisting in the analysis of causal relationships, providing insights into the potential impact of different factors.



"The Book of Why" by Judea Pearl.

7. Advancements in LLMs for Tabular Data

1. TableGPT

Enables various functions: question answering, data manipulation, visualization, and automated prediction

2. MediTab

Utilizes a data engine and LLMs to consolidate varying tabular samples, align out-domain data with a "learn, annotate, and refinement" pipeline

3. TABLET

A benchmark with 20 diverse tabular datasets annotated with varying instructions to test LLMs' effectiveness in tabular prediction

8. Ethical Considerations

Data Anonymization:

- Generative AI can be harnessed to create synthetic data that maintains the statistical properties of the original data while protecting sensitive information.

Privacy and Security:

- Challenges: LLMs might inadvertently memorize and leak sensitive data.
- Solutions: Differential privacy techniques and robust deployment architectures can mitigate privacy risks.

Bias and Fairness:

- May inherit biases from training data, leading to unfair or discriminatory outcomes.

Misinformation and Disinformation:

- Can create realistic fake content, aiding misinformation spread.

Authenticity and Ownership:

- Blurs boundaries of copyright and attribution with machine-generated content.

Consent:

- Requires consent when generating representations of individuals.

Transparency and Accountability:

- Need for disclosure about Generative AI use and mechanisms for accountability.

Explainability:

- Often seen as "black boxes," posing challenges in critical domains.

Resource Consumption:

- High computational and energy costs, with environmental implications.

Psychological Impact:

- May cause trust issues in digital content.

Economic Impact:

- Potential job displacement in content-creation fields.

Long-term Societal Impact:

- Unknown long-term effects, necessitating ongoing dialogue among stakeholders.

Thank you!

- **References**
  - Long form blog post and code:
    - Leveraging LLMs in Data Science Lifecycle for Demand Forecasting - https://t.ly/hyHtX
  - Large Language Models as Optimizers by Chengrun Yang et al. - arXiv:2309.03409
  - PandasAI https://github.com/gventuri/pandas-ai
  - sketch https://github.com/approximatelabs/sketch
  - LLMs for Semi-Automated Data Science: Introducing CAAFE for Context-Aware Automated Feature Engineering" by Hollmann, Müller, and Hutter (2023) - https://github.com/automl/caafe
  - The Book of Why by Judea Pearl
  - TabletGPT: Towards Unifying Tables, Nature Language and Commands into One GPT - arXiv:2307.08674
  - MediTab: Scaling Medical Tabular Data Predictors via Data Consolidation, Enrichment, and Refinement - arXiv:2305.12081
  - TABLET: Learning From Instructions For Tabular Data - arXiv:2304.13188
- **Disclaimer**: images are generated with MidJourney