Machine Learning for Economics Research

History, Theory and Practice

Giulia Caprini – Oxford University

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Some vocabulary: What is Machine Learning?

- **Definition:** Machine learning (ML) is a subset of artificial intelligence (AI) involving algorithms that enable computers to learn from data and improve performance over time *without explicit programming*.

Key idea: Instead of us specifying a formula, we let the computer learn from examples

Examples in everyday life: recommendation systems, image recognition, spam detection in emails, autonomous driving.

What is Machine Learning?

- ML learns patterns from data *instead* of using explicit rules
- It excels at prediction and pattern recognition
- ML is useful for high-dimensional and unstructured data (like a text, or images)

Econometrics often focuses on causal inference;

ML is complementary

Some vocabulary: algorithms

- **Algorithms**: a step-by-step set of instructions for solving a problem or completing a task.

Think of it like a recipe in cooking:

Ingredients → input data

Cooking steps → instructions (the algorithm)

Dish → the result (output)

Some vocabulary: algorithms

- **Algorithms**: a step-by-step set of instructions for solving a problem or completing a task.

Example from economics: "Calculate average income"

- Step 1: Add up all income values
- Step 2: Count the number of entries
- Step 3: Divide total income by number of entries
  → That's an algorithm for computing the mean.

Some vocabulary: algorithms

- **Algorithms**: a step-by-step set of instructions for solving a problem or completing a task.

Example from ML: "Predict house price"

- Step 1: Input features: size, location, age
- Step 2: Multiply each feature by a learned weight (learned via regression)
- Step 3: Add them up and return a price
  → The algorithm follows these steps for every new house.

Some vocabulary: training data

Training data is the set of **examples** we give to a machine learning model so it can **learn** a pattern or rule.

Think of it like a **teacher giving graded homework** to a student:

- The **input** (question) and the **correct answer** (label) are provided.
- The model learns: “When I see this kind of input, the answer should be that.”

Email text	Label
"You won a prize!"	spam
"Meeting at 3pm"	not spam

Some vocabulary: prediction

- Once the model has been trained, you can give it **new, unseen data —** and it will use what it learned to make out of sample «**predictions**».

- In ML literature prediction = forecasting future or unseen outcomes using patterns in the data;
- In Econometrics literature, prediction= usually refers to fitting a model, not necessarily forecasting future observations

Prediction is a tool. In ML, it’s often the goal. In economics, it’s often a step toward explaining or testing a theory.

Some vocabulary: supervised learning

Supervised Learning: Algorithms trained with labeled examples (e.g., classification)

→ You give the model the questions *and* the answers:

The algorithm learns from **labeled data —** each input has a known correct output. The goal is to **learn a mapping** from input → output so it can make predictions on new data.

Some vocabulary: supervised learning

Supervised Learning: Algorithms trained with labeled examples (e.g., classification)

→ You give the model the questions *and* the answers:

The algorithm learns from **labeled data** — each input has a known correct output. The goal is to **learn a mapping** from input → output so it can make predictions on new data.

Examples: Classifying satellite images as urban vs. rural; Predicting whether a household is poor based on image features.

Like a student studying past exams with the answer key — they learn how to solve problems by example.

Some vocabulary: unsupervised learning

Unsupervised Learning: Algorithms trained on unlabeled data to find hidden structures (e.g., clustering, dimensionality reduction)

→ You give the model the questions, but no answers.

The algorithm tries to **find structure or patterns** in the data on its own. No labels — just input data. Common goals: grouping similar observations (clustering), reducing dimensionality.

Some vocabulary: unsupervised learning

Unsupervised Learning: Algorithms trained on unlabeled data to find hidden structures (e.g., clustering, dimensionality reduction)

→ You give the model the questions, but no answers.

The algorithm tries to **find structure or patterns** in the data on its own. No labels — just input data. Common goals: grouping similar observations (clustering), reducing dimensionality.

Examples: Clustering regions based on nightlight patterns; Grouping newspaper articles by topic.

Like giving students a pile of unlabeled essays and asking to group them by topic — they have to figure out patterns without help.

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

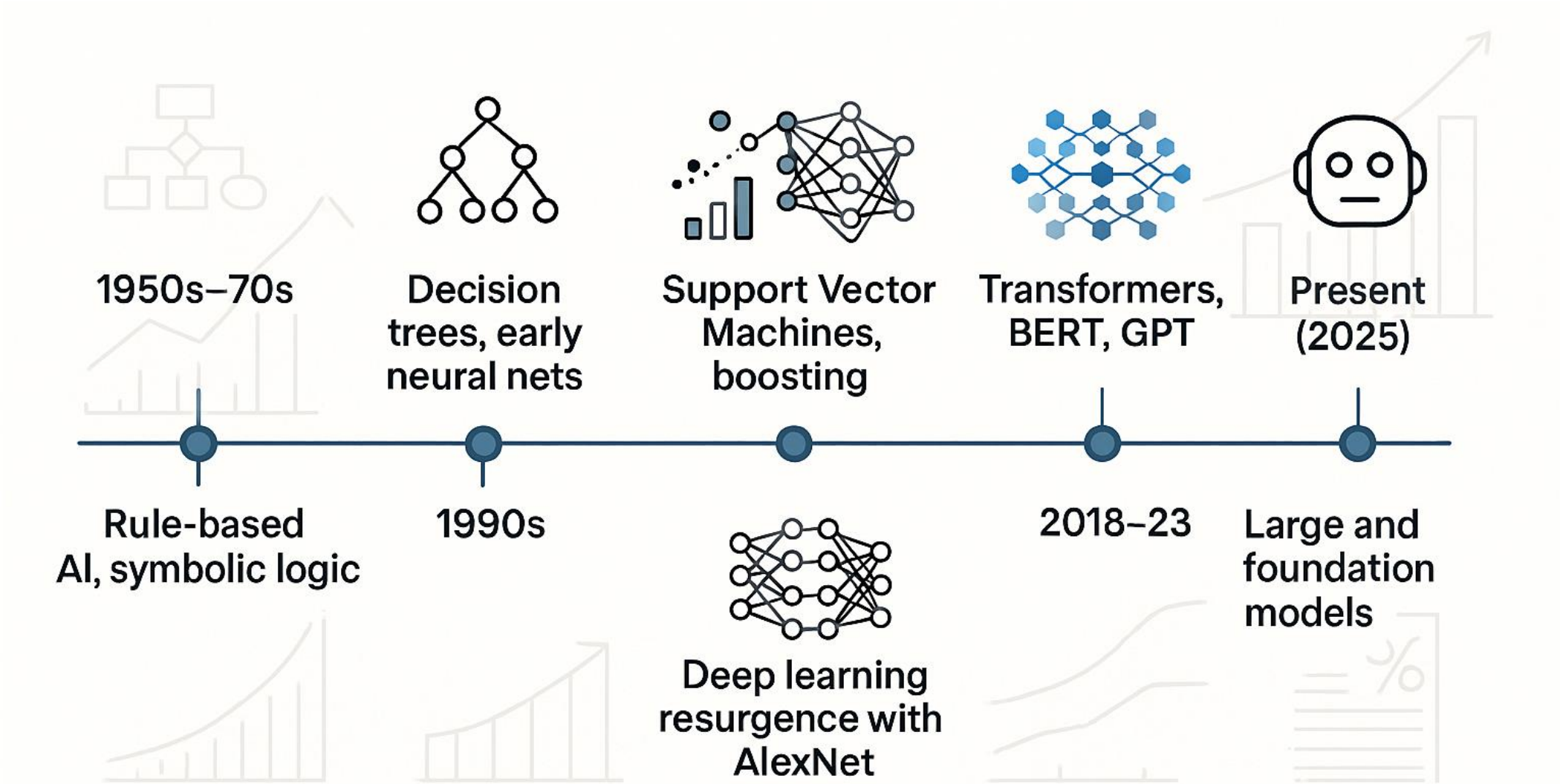
+ Wrap-up and Q&A

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

A brief history of ML



1950s–1970s: Symbolic AI and Early Ideas

- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.

The test:

A human judge has text-based conversations (e.g., via chat) with two unseen entities: one is a human, the other is a machine.

If the judge can't reliably tell which is which, the machine passes the test.

1950s–1970s: Symbolic AI and Early Ideas

- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.
- **1957** – *Perceptron (Frank Rosenblatt)*:
  - First algorithm modeled after neurons.
  - Could learn simple patterns but failed on complex prob (e.g., XOR).

1950s–1970s: Symbolic AI and Early Ideas

- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.
- **1957** – *Perceptron (Frank Rosenblatt)*:
  - First algorithm modeled after neurons.
  - Could learn simple patterns but failed on complex problems.

A **perceptron** is the simplest type of *neural network* — a kind of yes/no decision-maker.

- Imagine you're building a system to decide whether to **approve a loan**.
  You give it inputs: income, credit score, debt. The perceptron multiplies each by a weight, adds them up, and uses a rule: *"if total is above threshold, say YES"*.

This works for problems that can be separated with a straight line, but it **can't solve more complex problems** (like XOR).

1950s–1970s: Symbolic AI and Early Ideas

- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.
- **1957** – *Perceptron (Frank Rosenblatt)*:
  - First algorithm modeled after neurons.
  - Could learn simple patterns but failed on complex problems.

XOR ("exclusive or") is a logic function returning true if **exactly one** input is true.

- Example:

  (0, 0) → 0 (1, 0) → 1 (0, 1) → 1 (1, 1) → 0

Why is this famous? Because no straight line can separate the ones and zeroes in a 2D plot — this showed that a single perceptron wasn't enough for all problems, which led to **multi-layer neural networks**.

1950s–1970s: Symbolic AI and Early Ideas

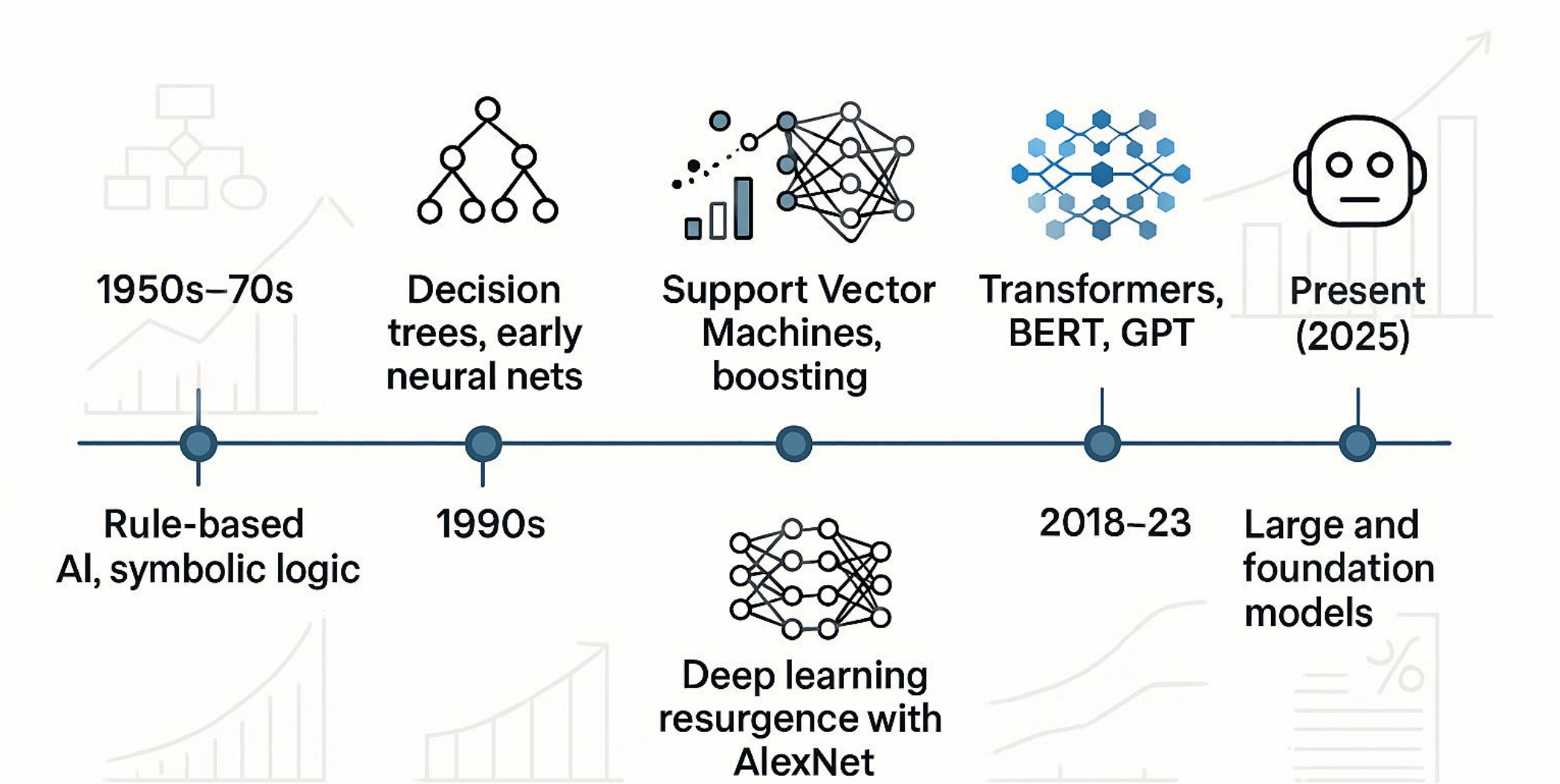
- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.
- **1957** – *Perceptron (Frank Rosenblatt)*:
  - First algorithm modeled after neurons.
  - Could learn simple patterns but failed on complex prob (e.g., XOR).
- **1970s** – *Expert systems, rule-based AI*:
  - Programs encoded human rules ("if income > X, then rich").
  - Required manual specification of rules, not scalable.

1950s–1970s: Symbolic AI and Early Ideas

- **1950** – *Alan Turing proposes the "Turing Test"*: Lays philosophical groundwork for thinking about machine intelligence.
- **1957** – *Perceptron (Frank Rosenblatt)*:
  - First algorithm modeled after neurons.
  - Could learn simple patterns but failed on complex prob (e.g., XOR).
- **1970s** – *Expert systems, rule-based AI*:
  - Programs encoded human rules ("if income > X, then rich").
  - Required manual specification of rules, not scalable.

- This laid the theoretical foundations. But rule-based systems couldn't learn from data — which ML would later solve.

A brief history of ML



1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.

A **neural network** is a bunch of perceptrons stacked together, in layers.

Imagine a **network of decision-makers:** each does a small job, one neuron detects **edges** in an image, another recognizes **shapes**.

Each layer transforms and passes on the information.

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.

In a **neural network** each layer transforms and passes on the information.

- **Input layer** → Takes in raw data (e.g., pixels, numbers)
- **Hidden layers** → Process and transform data (multiple steps)
- **Output layer** →Gives a prediction or decision (e.g., “cat”, 72% poverty)

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective "deep" learning for small problems.

Inside, each neuron in a layer does the following:

- Takes numbers in (features),
- Multiplies them by **weights**,
- Adds them up,
- Applies a function to decide what signal to send forward.

This flow is called **"forward propagation."**

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective "deep" learning for small problems.

Example: Predicting Loan Default

- **Inputs**: income, age, credit score
- **Hidden layers**: detect nonlinear patterns (e.g. young + low income = risky?)
- **Output**: probability of default

The network figures this out **by learning from training data**, not by hand-coded rules.

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.

Backpropagation is how the NN learns:

- The model makes a prediction.
- It compares the result to the true answer (error).
- It sends that error **backward** through the network and **adjusts the weights**.
- This process repeats many times (called **training**).

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective "deep" learning for small problems.

What Makes NN Powerful?

•It can learn **complex, non-linear patterns**.

Downsides of Neural Networks:

•Often a **"black box"** — it's hard to interpret *why* it made a prediction.

•Needs **lots of data and computing power**.

•Prone to **overfitting** if not trained carefully.

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.

1980s: First Learning Algorithms

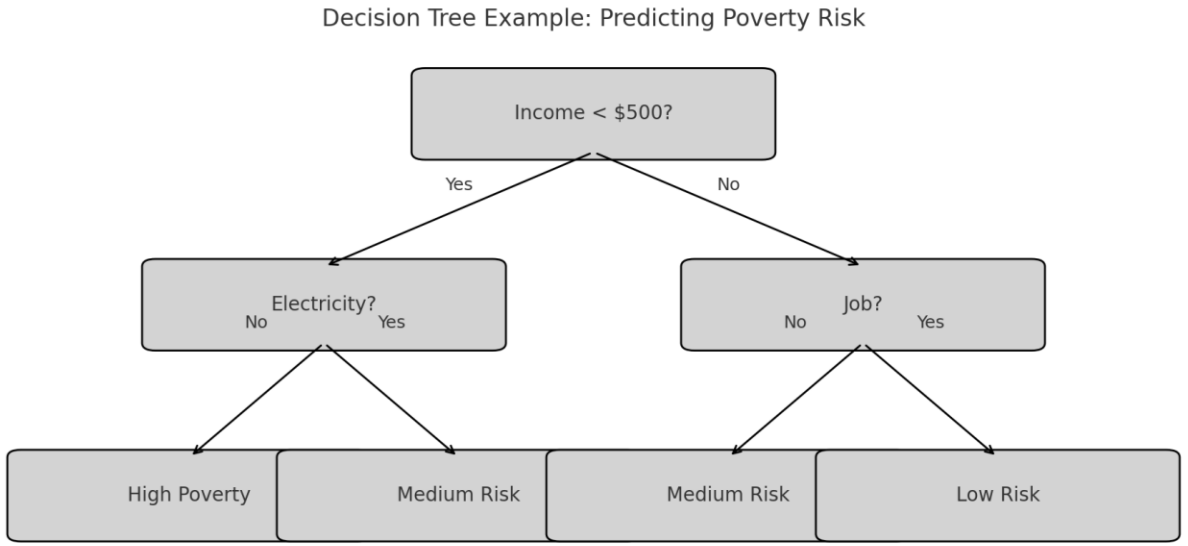
- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.

NN is the basis for **deep learning** (the word *deep* refers to its many layers)

Note that in the 1990s the neural networks were mostly **shallow** — just 1–2 hidden layers, because deeper NN would have been hard to train.

1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.
- **1980s–1990s** – Rise of **decision trees**, **k-nearest neighbors**, **naive Bayes**:
  - Easy to understand, fast.

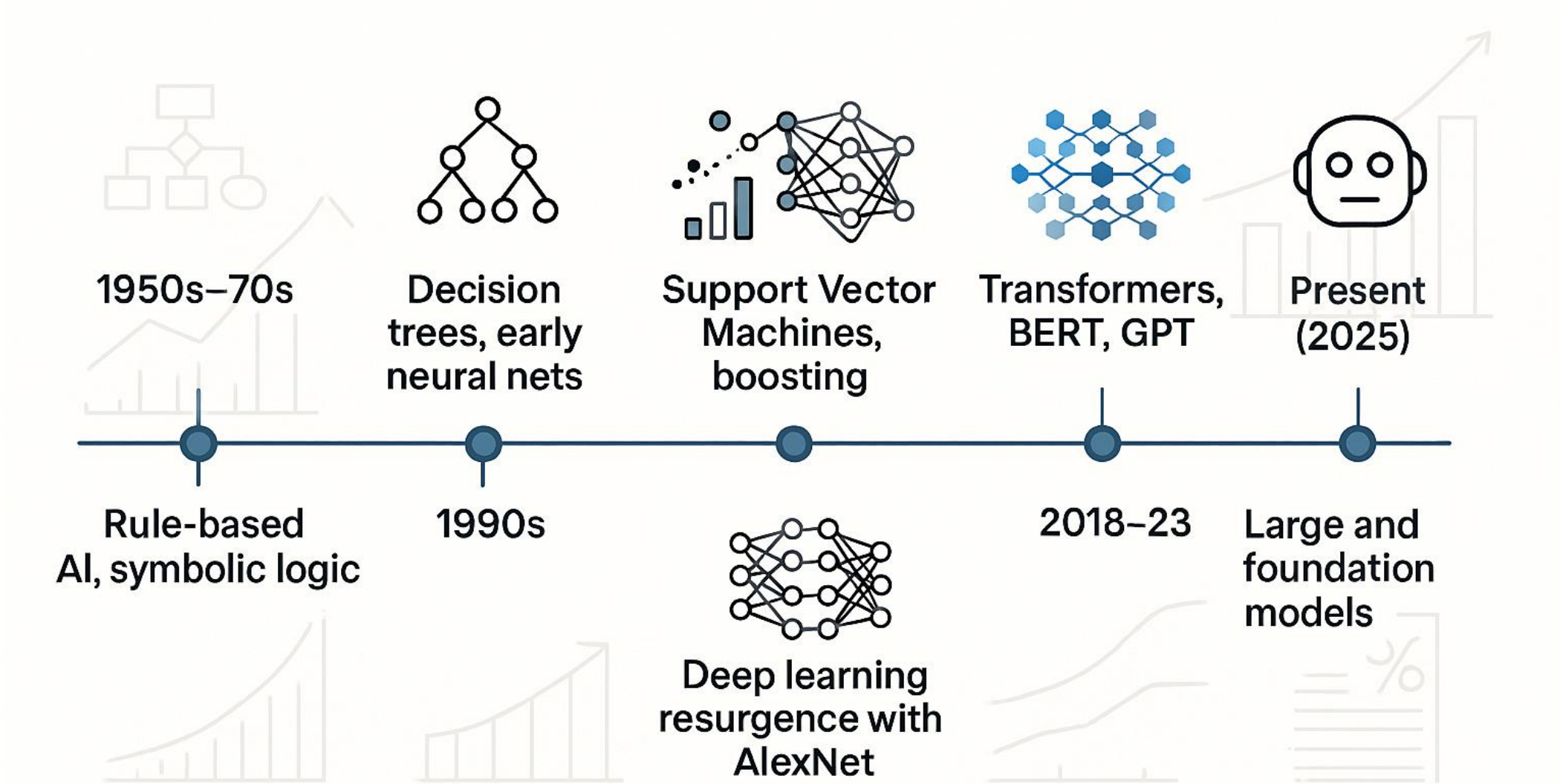


1980s: First Learning Algorithms

- **1986** – *Backpropagation rediscovered (Rumelhart et al.)*:
  - Enabled multi-layer neural networks to be trained.
  - First effective “deep” learning for small problems.
- **1980s–1990s** – Rise of **decision trees**, **k-nearest neighbors**, **naive Bayes**:
  - Easy to understand, fast.

This was the first wave of algorithms that could learn patterns from data with general-purpose use cases.

A brief history of ML



1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM):**
  - Powerful linear classifiers with margin maximization.

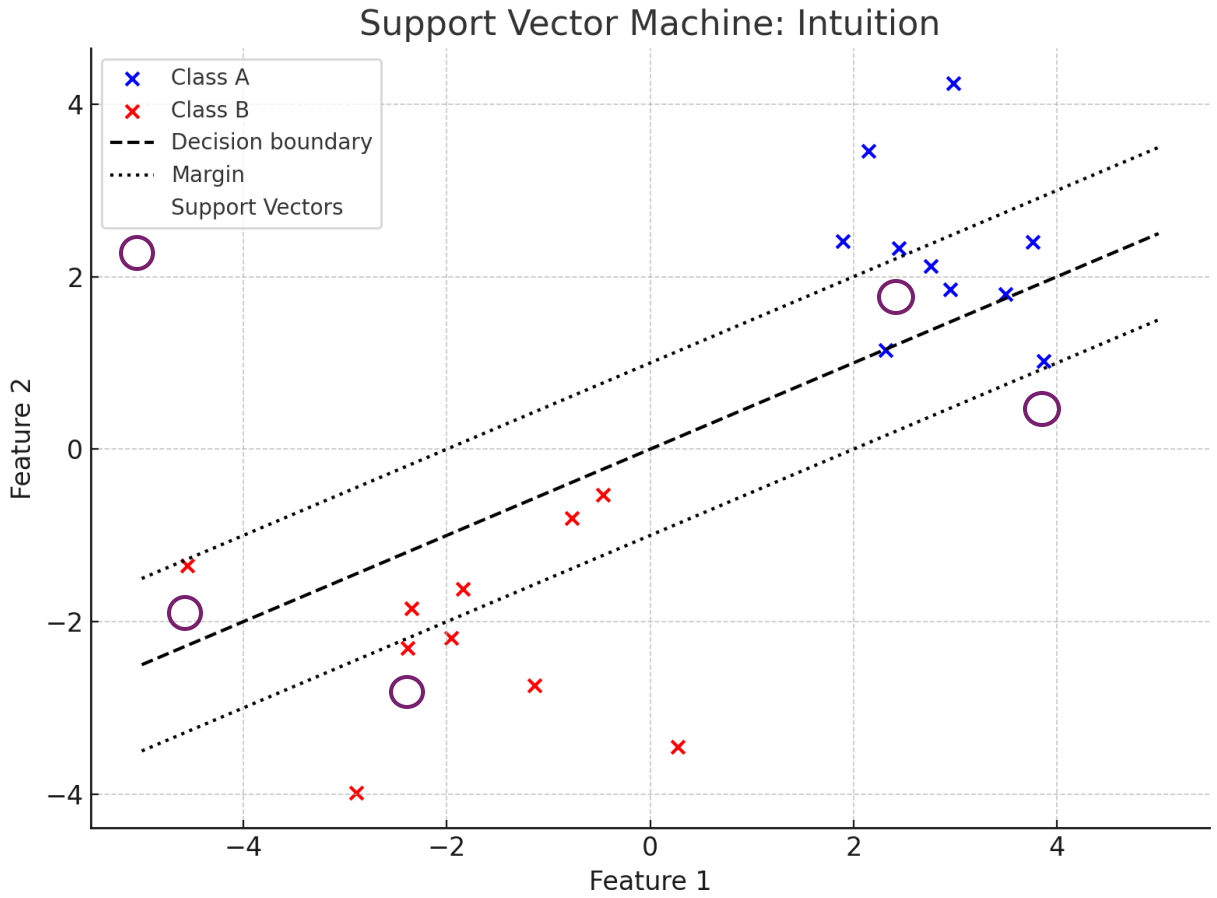
A SVM is classifier that finds **the best boundary** (or hyperplane) between classes.

Imagine plotting two types of data points (e.g. fraud vs. not fraud).

An SVM finds the **line (or surface)** that separates them **with the widest margin** — the “safest” separation.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM):**



The **solid dashed line** is the **decision boundary** — the best line that separates the two classes.

The **dotted lines** are the **margins;** the model tries to **maximize the distance** between them and the nearest data points from each class.

The **points touching the margin** are called **support vectors** — they are critical to defining the boundary.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:
  - Combined many weak models (e.g., shallow trees) into a strong predictor.
  - Inspired ensemble learning.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:

Boosting is a method for turning many weak learners into one strong learner.

•A **weak learner** is a model that does just slightly better than random guessing (like a shallow decision tree).

•Boosting **trains a sequence** of these weak models, each one focusing more on the **mistakes** of the previous ones.

•Then it **combines** their outputs into a final prediction, often by a weighted vote.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:

Boosting is like a panel of experts: Imagine you're trying to predict poverty in villages using satellite images. You ask several simple analysts (weak learners):

- The first analyst says: “Low vegetation → high poverty.” Not bad, but not great.
- You find out where the analyst got it wrong.
- The next analyst says: “Look more closely at road density in the mistakes.”
- The third analyst says: “Also check for roof material.”

At the end, you combine their advice — and you get a pretty smart prediction!

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:

AdaBoost stands for **Adaptive Boosting** — it's the most famous boosting algorithm. It's key idea is that it adaptively focuses more on the data points that are **hard to classify**.

1990s: Statistical ML and Theoretical Foundations

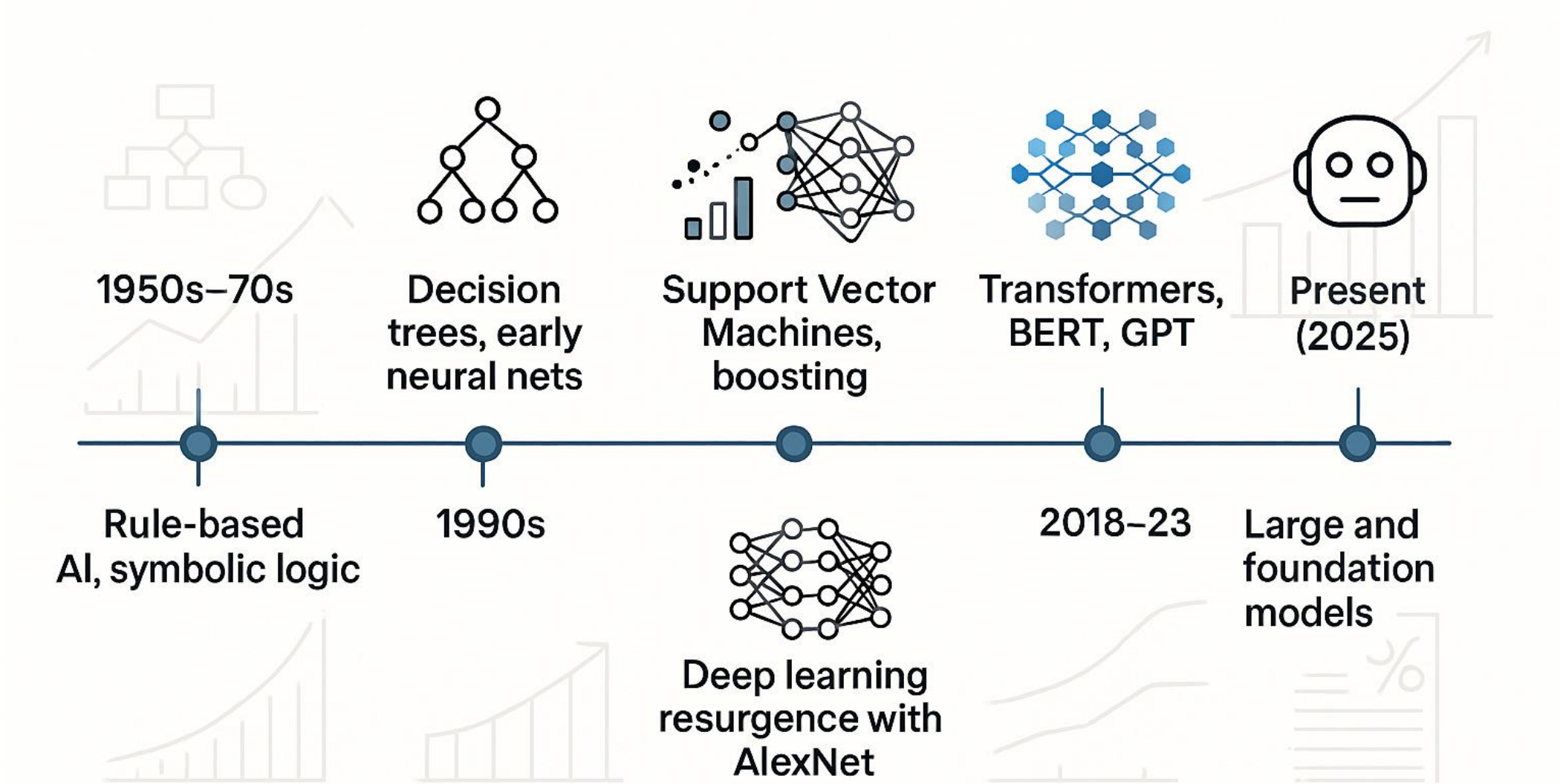
- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:
  - Combined many weak models (e.g., shallow trees) into a strong predictor.
  - Inspired ensemble learning.
- **Bayesian methods**: Probabilistic models gained traction in modeling uncertainty.

1990s: Statistical ML and Theoretical Foundations

- **Support Vector Machines (SVM)**:
  - Powerful linear classifiers with margin maximization.
- **Boosting (AdaBoost)**:
  - Combined many weak models (e.g., shallow trees) into a strong predictor.
  - Inspired ensemble learning.
- **Bayesian methods**: Probabilistic models gained traction in modeling uncertainty.

There was a shift from symbolic to statistical thinking. Tools from this era are still robust for tabular/structured data (common in econ).

A brief history of ML



2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).

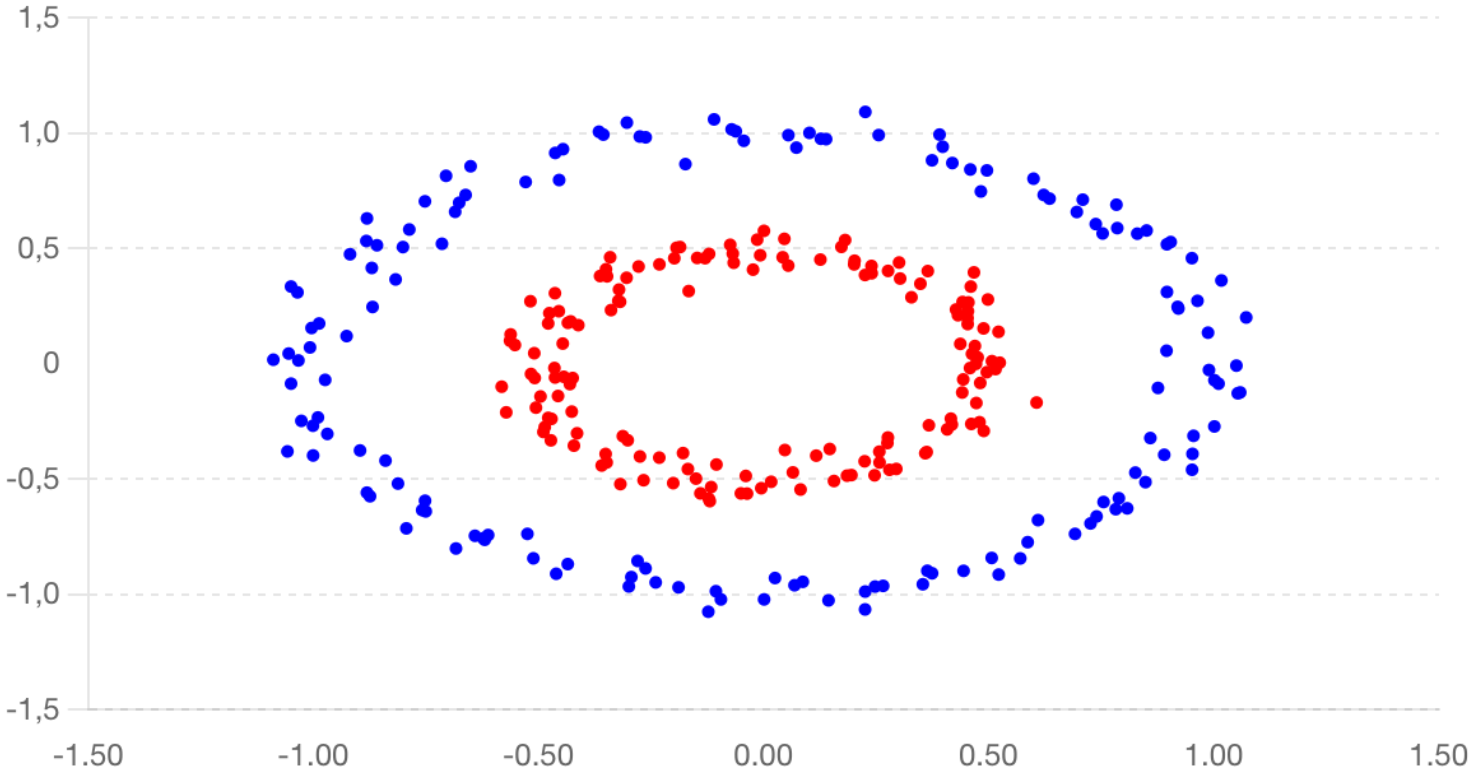
What is a Kernel Method?

Imagine this:
You have data that’s not easily separated by a straight line. Blue dots and red dots are all mixed up.

What do you do?

2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).



2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).

The Trick : A New Dimension!

- **Kernel methods** help by "secretly" sending your data into a **higher-dimensional space** where things **can** be separated!

It’s like unfolding a paper to separate dots that were overlapping.
A **kernel** is a smart **math function** that lets us compare two data points without actually doing all the hard math of changing dimensions.

2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).

Imagine you're solving a puzzle on a flat table, but it’s impossible to separate the red and blue pieces with a ruler. Now imagine you could **lift** the puzzle into the **air** (higher dimension), and suddenly, **the separation is easy**.
Kernel methods do this lifting — but with math, not actual coordinates.

- A **kernel method** lets a linear model (like SVM) **act nonlinear** — by cleverly computing inner products in **imaginary high-dimensional spaces**.

2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).
- **Random Forests** (Breiman, 2001):
  - Extremely powerful ensemble method.
  - Robust, interpretable, and easy to use (popular in applied economics).

A random forest builds **many trees** on random subsets of data and **averages their results**.

Think of it as **"many weak opinions combined into one strong opinion."**

Imagine asking 100 friends: "Should I wear a jacket today?"

Each one gives their opinion based on what they know (temperature, clouds, wind, etc.) You go with the **majority** decision

2000s: Scaling Algorithms and Data

- **Kernel methods** for non-linear patterns (used with SVMs).
- **Random Forests** (Breiman, 2001):
  - Extremely powerful ensemble method.
  - Robust, interpretable, and easy to use (popular in applied economics).
- **Graphical models**: For structured dependencies (Bayesian networks, HMMs).
- These methods combined solid theory with practical performance — still widely used for prediction tasks.

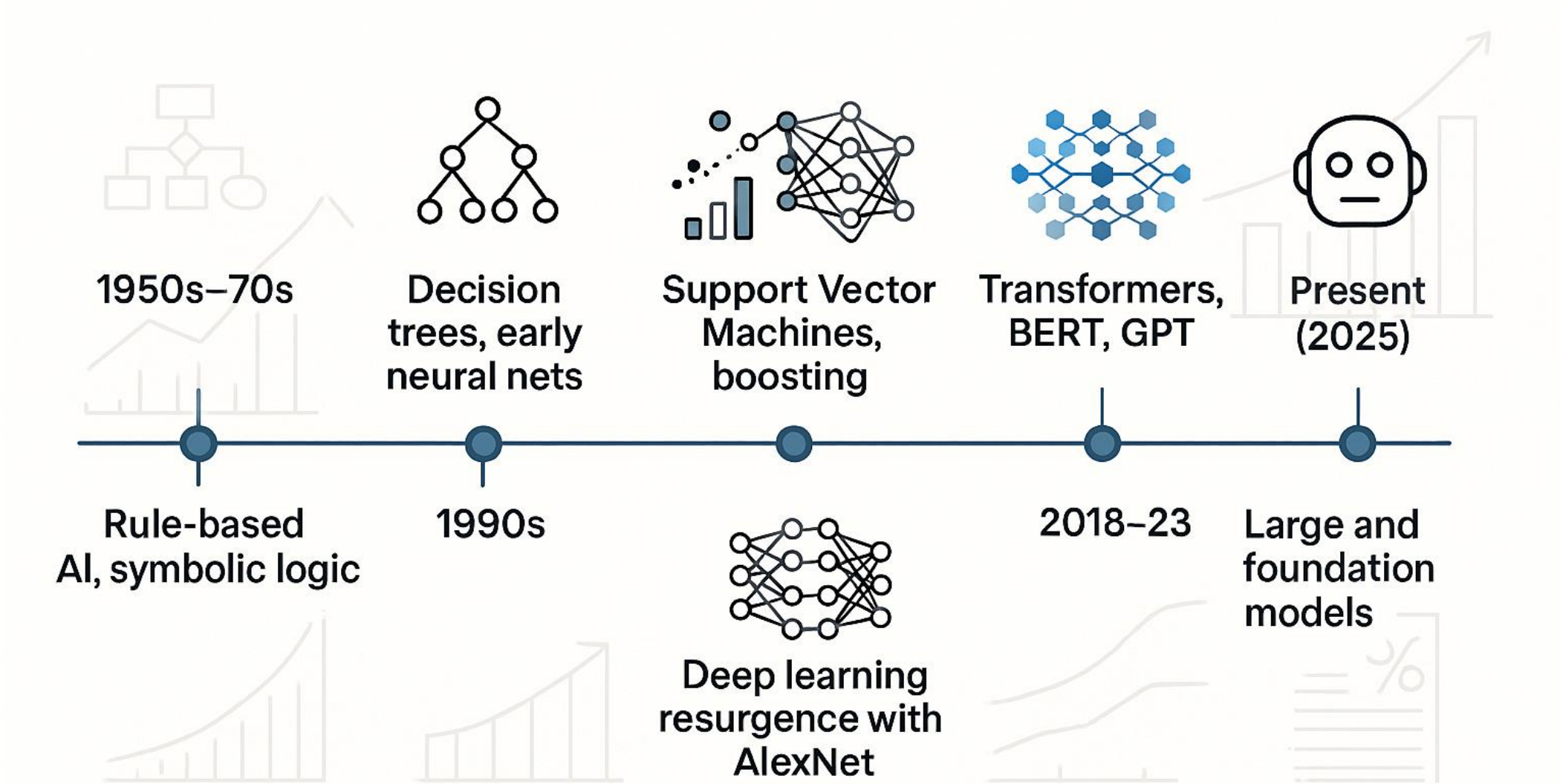
Bayesian Network

- A **graph-based model** that shows **how variables influence each other** probabilistically.
- Each node is a variable (e.g., education, income, employment).
- Arrows represent conditional dependencies: e.g., education → income.
- It's like a map of how things probabilistically depend on one another, useful for inference under uncertainty.

Hidden Markov Model (HMM)

- A **sequence model** used to model systems that **change over time** but have **hidden states**.
- Example: You observe someone's actions (e.g., buying behavior) but not their mood or financial state — that's the "hidden" part.
- HMM assumes current state depends on the **previous one**, and you only see **signals**, not the actual state.
- Used in speech, bioinformatics, and time series analysis.

A brief history of ML



2010–2012: Deep Learning Breakthrough

- **2012** – *AlexNet (Krizhevsky et al.) wins ImageNet*:
  - Used GPUs to train a deep convolutional neural network.
  - Achieved massive improvement in image classification.

ILSVRC = ImageNet Large Scale Visual Recognition Challenge, held from **2010 to 2017**, this was a yearly competition where teams competed to:

- Classify images (1,000 categories)
- Detect objects (identify and locate)
- Evaluate performance using **top-1** and **top-5 accuracy**

2010–2012: Deep Learning Breakthrough

- **2012** – *AlexNet (Krizhevsky et al.) wins ImageNet*:
  - Used GPUs to train a deep convolutional neural network.
  - Achieved massive improvement in image classification.

"AlexNet" (2012) a deep neural network built by Alex Krizhevsky, used **convolutional neural networks (CNNs)** + **GPU acceleration,** crushing previous records in ILSVRC 2012. This sparked the **deep learning revolution** in image recognition.

2010–2012: Deep Learning Breakthrough

- **2012** – *AlexNet (Krizhevsky et al.) wins ImageNet*:
  - Used GPUs to train a deep convolutional neural network.
  - Achieved massive improvement in image classification.
- Convolutional Neural Networks (CNNs):
  - Core tool for image data, mimicking how humans detect visual features.
- Recurrent Neural Networks (RNNs):
  - Used for data where order matters (e.g., text, time series).

It was proved that deep learning could scale and outperform traditional ML — leading to rapid adoption in computer vision, speech, and more.

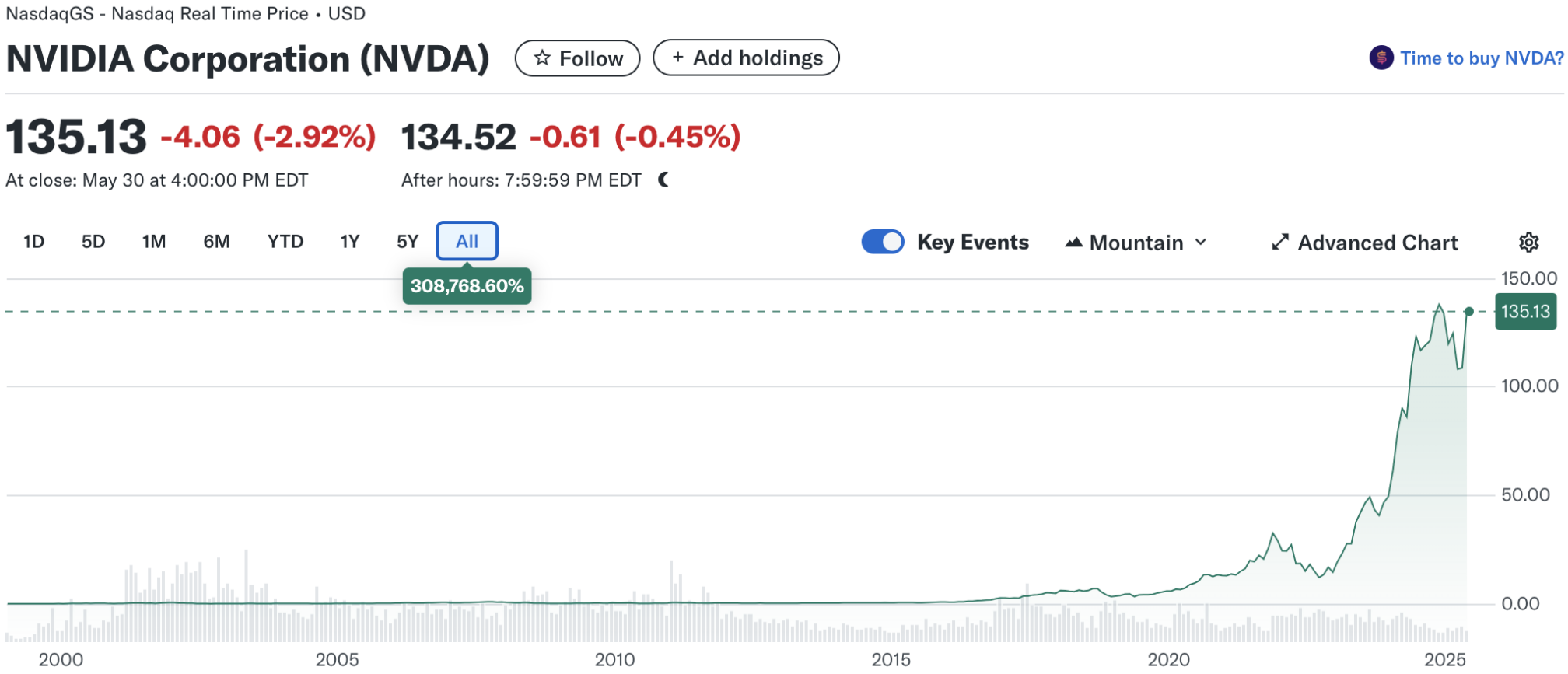
Why GPUs Changed the Game

- **CPU (Central Processing Unit)**: General-purpose, does a few tasks quickly.
- **GPU (Graphics Processing Unit)**: Originally made for video games, it can handle **thousands of simple operations in parallel**.

- Neural networks involve millions of **tiny, repetitive math operations** (like multiplying matrices). GPUs are built to do exactly that — **in parallel** — much faster than CPUs.

GPUs were key to the deep learning revolution because they made training massive models (like CNNs) feasible in hours instead of weeks.

NVIDIA bets on GPU production...



Modern (DL) algorithms:
CNN (Convolutional Neural Network)

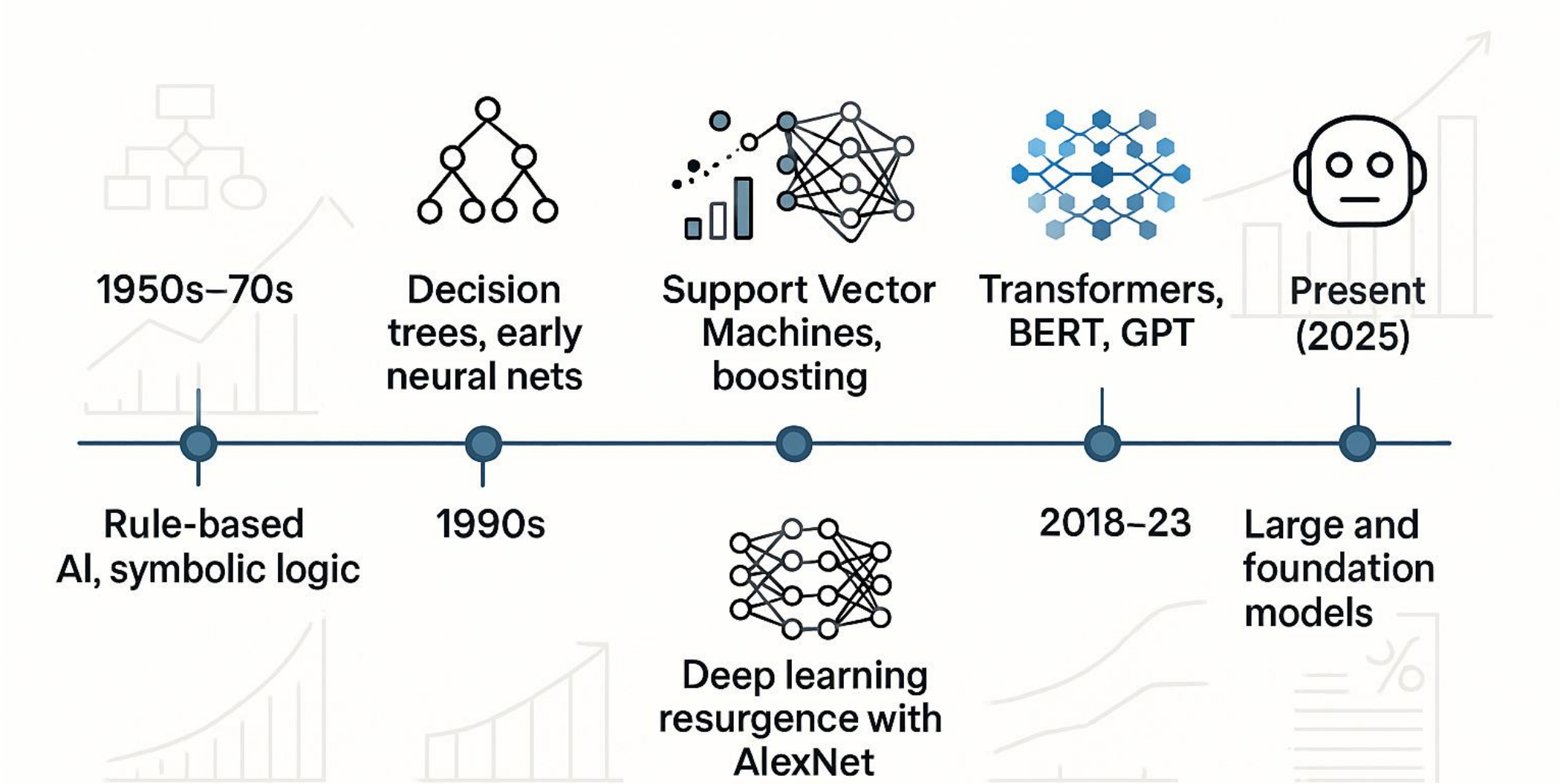
- Special neural network for **image data**.
- Instead of looking at all pixels at once, a CNN scans the image in **small patches** (filters or “convolutions”) and learns features like edges, shapes, textures.
- It stacks these layers to detect more complex patterns (like a face or a city).
- Very effective for image classification and object detection — used in satellite analysis, medical imaging, etc.

RNN (Recurrent Neural Network)

Designed for **sequence data** (like time series or text).

- An RNN has a “memory”: it considers **previous inputs** when predicting the next one.
- Good for modeling things where order matters (e.g., sentences, stock prices).
- But RNNs struggle with long sequences → replaced by **Transformers** (like GPT) in NLP tasks.

A brief history of ML



2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.

A **Transformer** is a **neural network architecture** designed to handle **sequential data** (like text), but **without relying on recurrence (RNNs)**.

Before Transformers, models like RNNs were used to process sequences **step by step** (like reading one word at a time). Transformers process **the entire sequence at once**, using a mechanism called **attention** to decide which parts of the input are most important.

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.

Consider the sentence:

"*The cat sat on the mat.*"

When the model is thinking about the word "sat," attention might work like this→

It focuses on **"cat"** and **"mat"** to understand what's happening with "sat."

Word	Attention Weight
the	low
cat	**high** ✅
sat	medium
on	low
the	low
mat	medium

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.

Why attention is powerful:

•It lets the model **see the whole sentence at once**.

•It learns **which words affect each other** — great for grammar, pronouns, meaning (a map of the relationships)

•It's **flexible**: works for short sentences, long documents, or code.

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual (i.e. flexible) word embeddings.

BERT stands for: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers.

It's a **language model** made by Google in **2018** that helps computers understand **what words mean** in context.

Before BERT, most models read text **left to right** (like GPT) or **right to left**.
But BERT reads in **both directions at the same time,** because it sees the **whole sentence** to understand meaning.

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual (i.e. flexible) word embeddings.

Self-supervised models learn by **predicting parts of the data** from other parts. Example (text):

"The cat sat on the ___."

The model tries to guess the missing word ("mat").

It's like playing **fill-in-the-blank** — over and over — on **millions of sentences**!

Word Embeddings

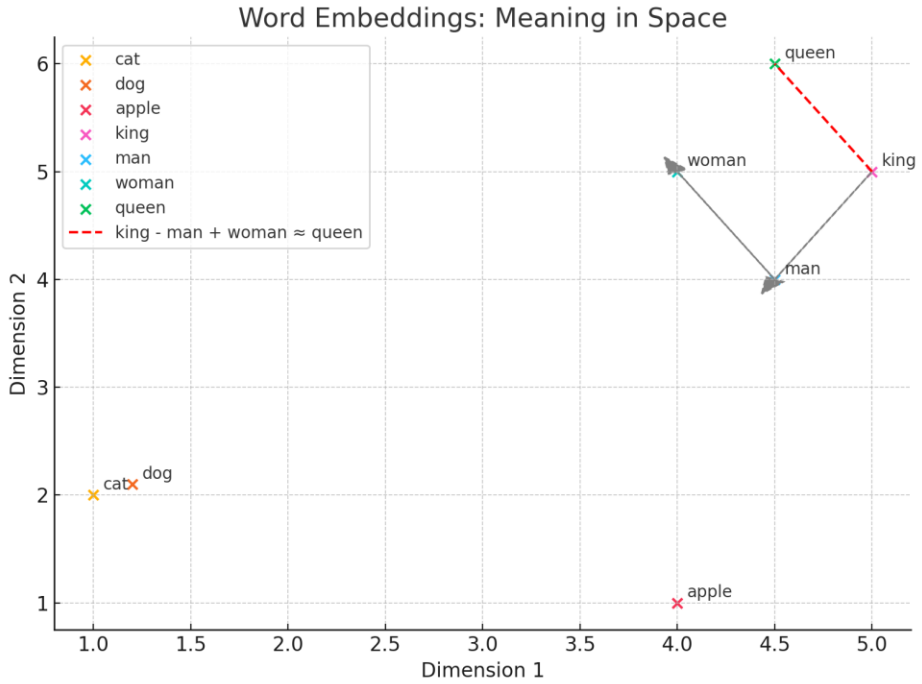
Embeddings are a way to **turn words into numbers** so computers can work with them.

- **Embedding**: places each word in a **vector space**, where similar words are **closer together**.

Here's a 2D visualization of word embeddings:

- Words like **"cat"** and **"dog"** are close together (similar meaning).
- **"apple"** is far away (different meaning).
- The **vector math** in gray shows:

"king" - "man" + "woman" ≈ "queen"



2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (paper by Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual (i.e. flexible) word embeddings.

BERT was the first famous **self-supervised** Transformer model.
BERT learned by itself by hiding random words in sentences and then predicting what those words were.

It was the **first model** to understand text well using self-supervised learning.
Moreover, BERT generates flexible embeddings, that is, different embeddings **based on sentence meaning.** ("bank" in a river sentence is different from "bank" in a money sentence).

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual word embeddings.
- **2019–2020** – *GPT-2 / GPT-3 (OpenAI)*:
  - General-purpose models that can generate coherent text and perform multiple tasks with zero/few-shot learning.

2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual word embeddings.
- **2019–2020** – *GPT-2 / GPT-3 (OpenAI)*:
  - General-purpose models that can generate coherent text and perform multiple tasks with zero/few-shot learning.

In 2020, models like **GPT-2 and GPT-3** took self-supervised learning even further. They learned by **predicting the next word** in billions of text documents. With enough data, they learned **grammar, facts, reasoning**, and even some **creativity** — just from text!

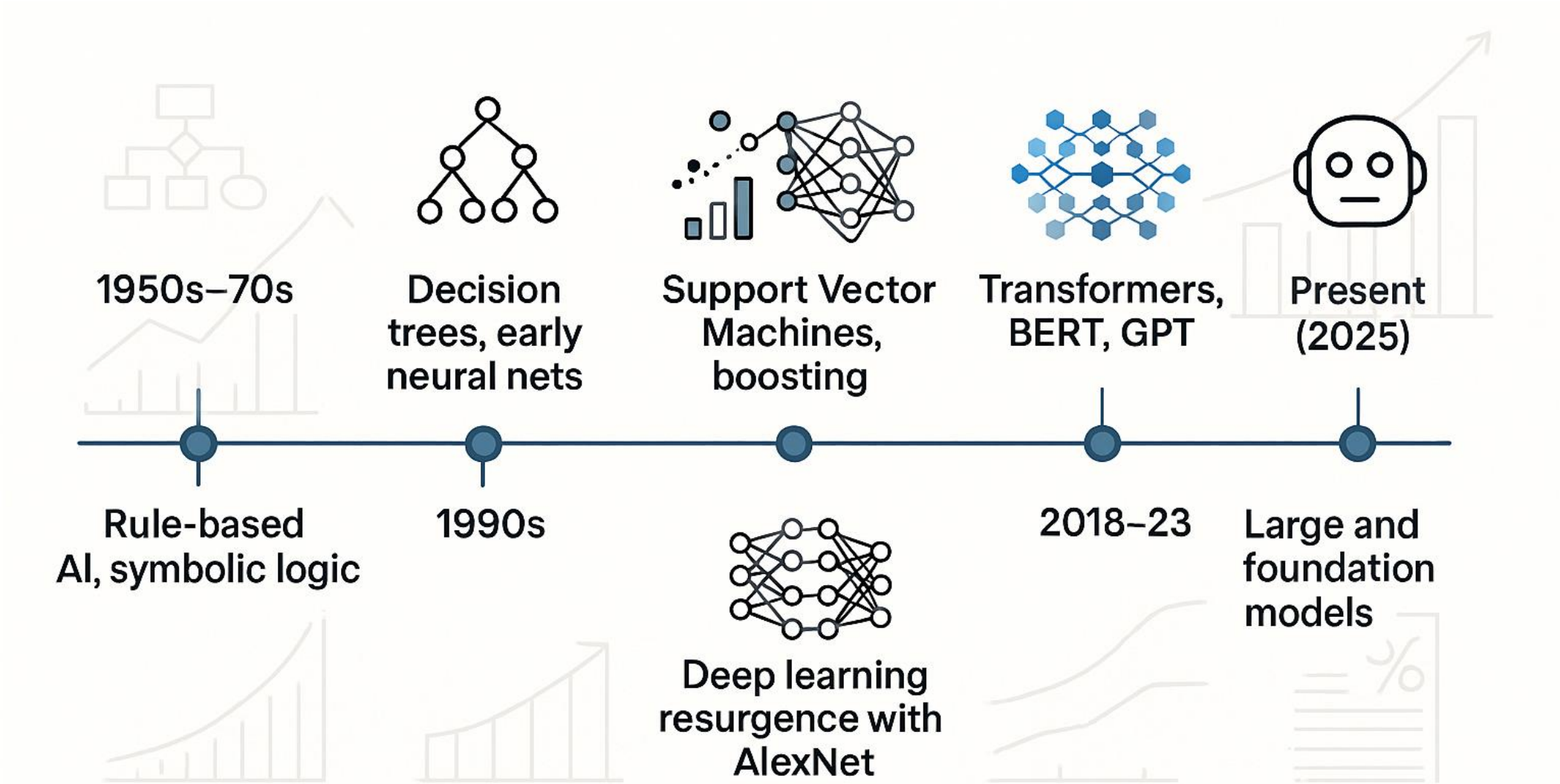
2017–2020: Transformers & NLP Revolution

- **2017** – *Attention is All You Need* (Vaswani et al.):
  - Introduced the Transformer architecture.
  - Replaced RNNs for language tasks; more parallelizable, scalable.
- **2018** – *BERT (Google)*: Contextual word embeddings.
- **2019–2020** – *GPT-2 / GPT-3 (OpenAI)*:
  - General-purpose models that can generate coherent text and perform multiple tasks with zero/few-shot learning.

Key gains:

- Breakthroughs in text understanding and generation.
- Foundation models became possible: train once, use everywhere.

A brief history of ML



2021–Today: Foundation Models & LLMs

- **GPT-4, Claude, Gemini, LLaMA**, etc.:
  - Trained on diverse modalities (text, code, images).
  - Useful for code generation, reasoning, tutoring, and even economics research workflows (e.g., scraping, summarizing).
- **Multimodal models** (e.g., CLIP, DALL·E, Flamingo):
  - Understand and generate across both image and text.
- **Open-source wave**:
  - HuggingFace, PyTorch Lightning, LLaMA — more democratized access.

An important novelty: LLMs and multimodal models have blurred the line between ML user and developer. Everyone can code!

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Why using ML? Main Tasks

- Classification – e.g., urban vs. rural image
- Regression – e.g., "predict" GDP from image brightness
- Clustering – e.g., group regions by visual similarity
- Object detection – e.g., count cars in street photos
- Segmentation – e.g., identify farmland pixel by pixel

Note: We nearly always use *pre-trained* models for these tasks – we don’t have to build an image recognizer from scratch, we can adapt existing ones!

A data revolution in Economics

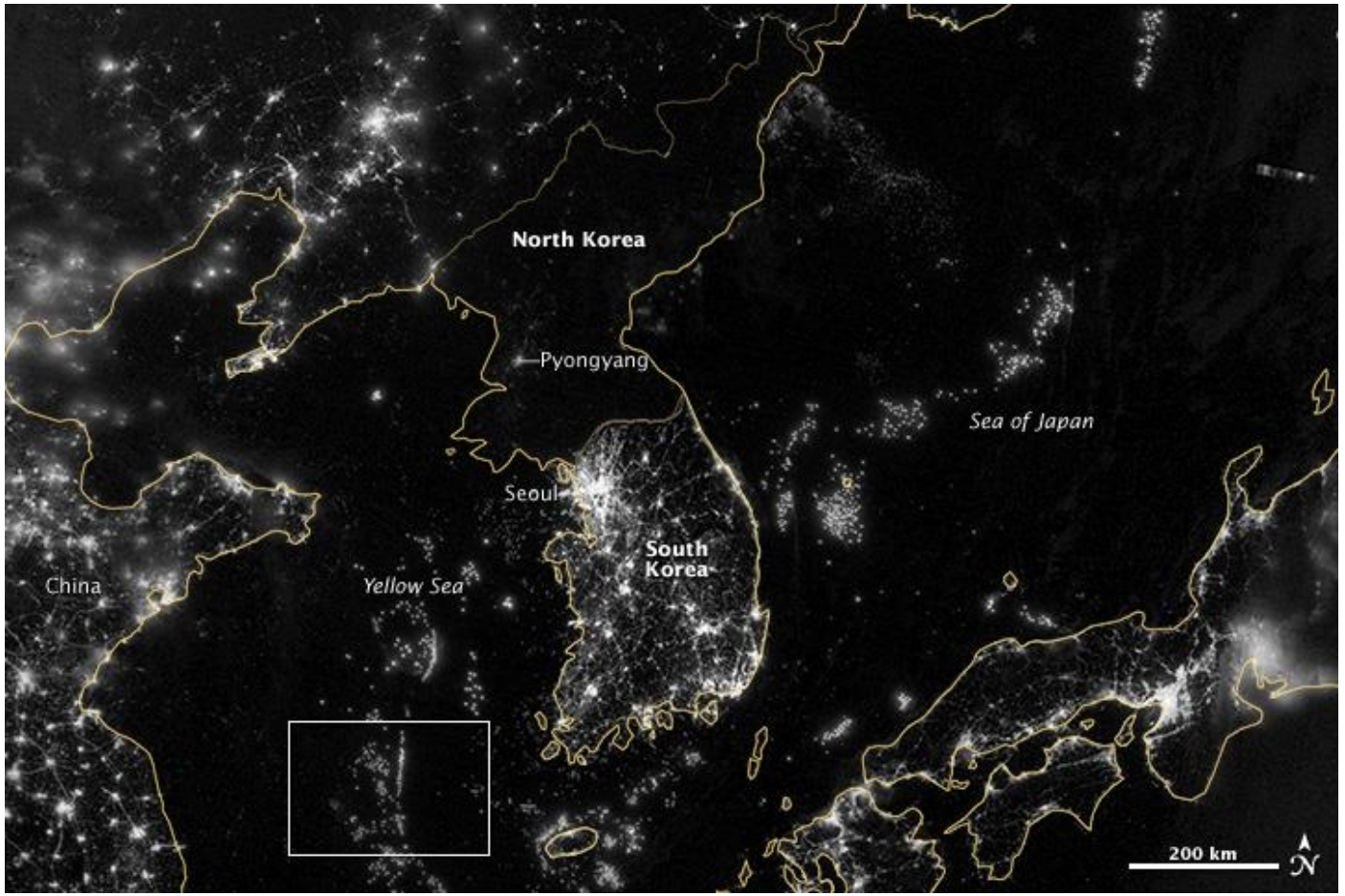
- Traditional data: surveys, aggregated stats.
- Modern data: images (satellite, street), text (social media, news), sensor data, etc.

The availability of large datasets plus ML methods is expanding what we can study. Economists now extract info from these data (e.g., text analysis for sentiment, image analysis for urban change,etc.).

Case study: Satellite images as Data

Satellite Imagery: A Goldmine of data!

Why useful?



(Image source: NASA Earth Observatory and NOAA National Geophysical Data Center)

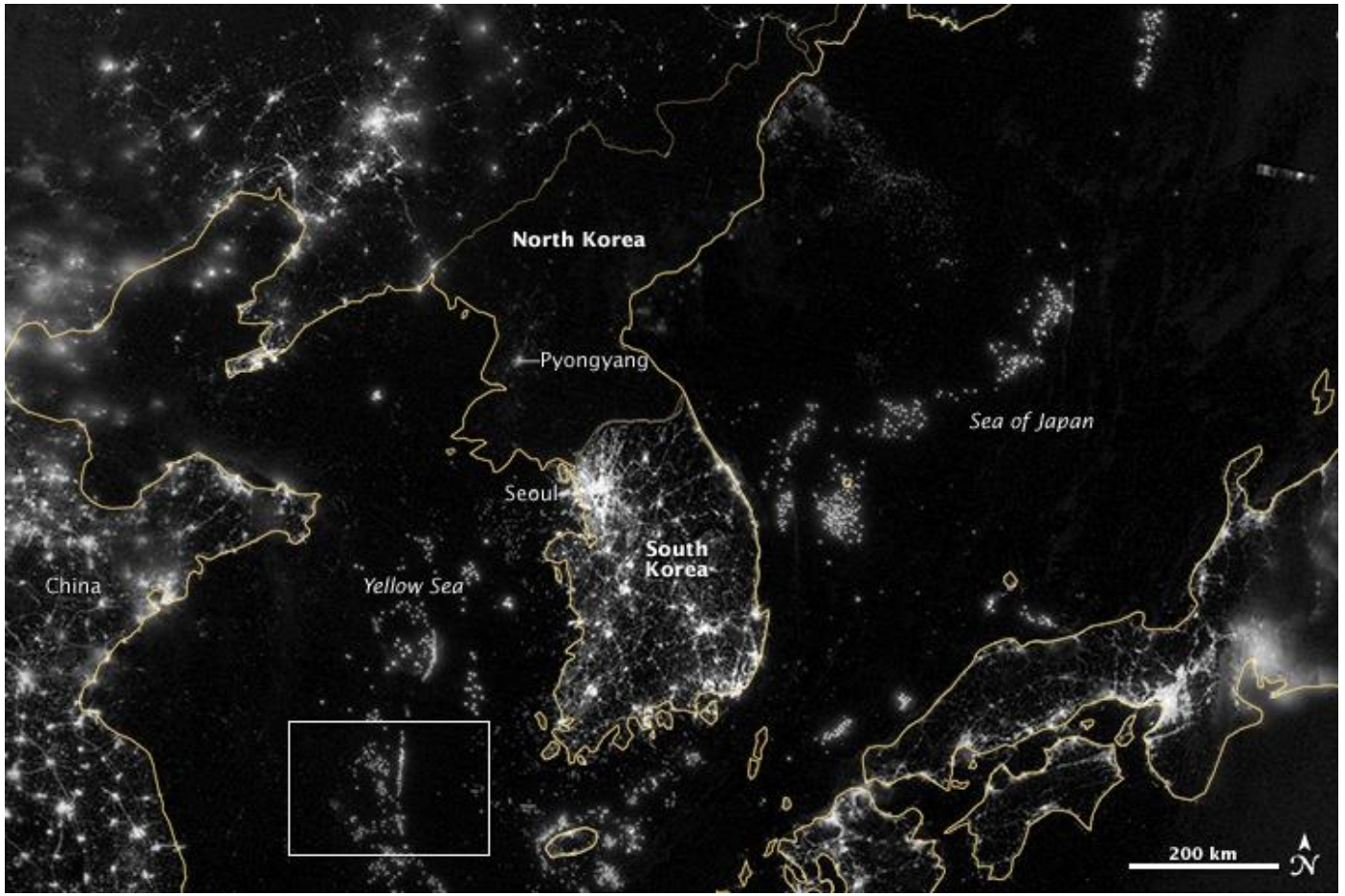
Case study: Satellite images as Data

Satellite Imagery: A Goldmine of data!

Why useful?

- Covers *everywhere on Earth*, consistent measurement, often free and updated.

- Can proxy things like infrastructure, economic activity, population density.



(Image source: NASA Earth Observatory and NOAA National Geophysical Data Center)

Case study: Satellite images as poverty Data

- **Problem:** Lack of local economic data in Africa.

Case study: Satellite images as poverty Data

- **Problem:** Lack of local economic data in Africa.
- **Idea:** Use high-res images + ML to predict wealth (Jean et al. 2016).
- **How?**

Case study: Satellite images as poverty Data

- **Problem:** Lack of local economic data in Africa.
- **Idea:** Use high-res images + ML to predict wealth (Jean et al. 2016).
- **How:** Train ML on what bright night lights correlate with in daytime images. ML learned features (roads, sturdy roofs, etc.) that indicate development.
- **Outcome:** Generated poverty heatmaps more detailed than ever before, guiding resource allocation.

Case study: Satellite images as poverty Data

- **Problem:** Lack of local economic data in Africa.
- **Idea:** Use high-res images + ML to predict wealth (Jean et al. 2016).
- **How:** Train ML on what bright night lights correlate with in daytime images. ML learned features (roads, sturdy roofs, etc.) that indicate development.
- **Outcome:** Generated poverty heatmaps more detailed than ever before, guiding resource allocation.

This doesn't replace surveys entirely, but it greatly augments them.

Case study: evaluation of policy impact

Does building roads cause deforestation?

Case study: evaluation of policy impact

Does building roads cause deforestation?

Economists have examined satellite-derived forest cover data near new road projects (aiddata.org).

One study in Cambodia combined satellite forest-cover images with administrative data on where roads and irrigation were built.

Case study: evaluation of policy impact

Does building roads cause deforestation?

Economists have examined satellite-derived forest cover data near new road projects (aiddata.org).

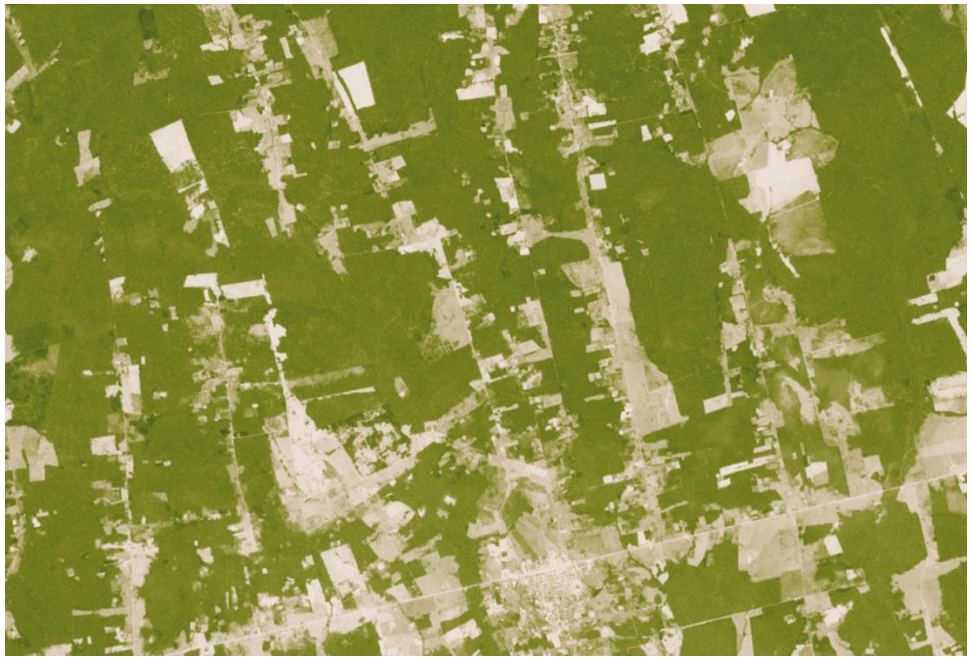
One study in Cambodia combined satellite forest-cover images with administrative data on where roads and irrigation were built.
It found that new irrigation infrastructure actually *improved* nearby forest cover (perhaps by intensifying agriculture on existing farmland, reducing pressure to clear new land), and that new roads, on average, did not immediately lead to forest loss except in very dense forests.

Case study: evaluation of policy impact

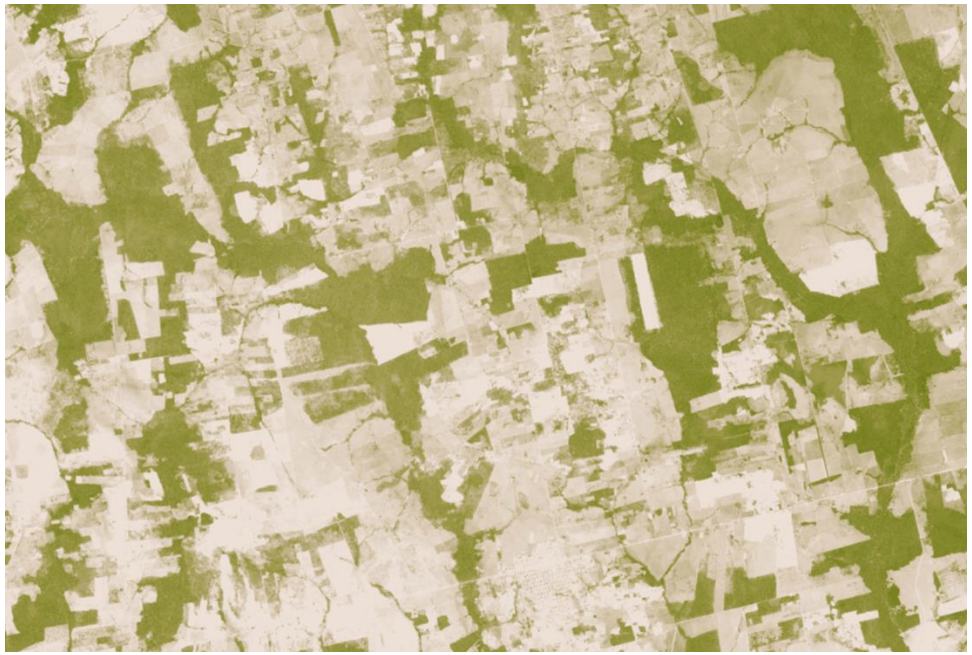
This kind of analysis would be impossible without remote-sensing data – satellites provide an objective, **consistent** way to measure forest cover change at scale.

A pair of satellite images of Rondônia, Brazil (in the Amazon)
Forested areas appear green; deforested land appears tan.

2000

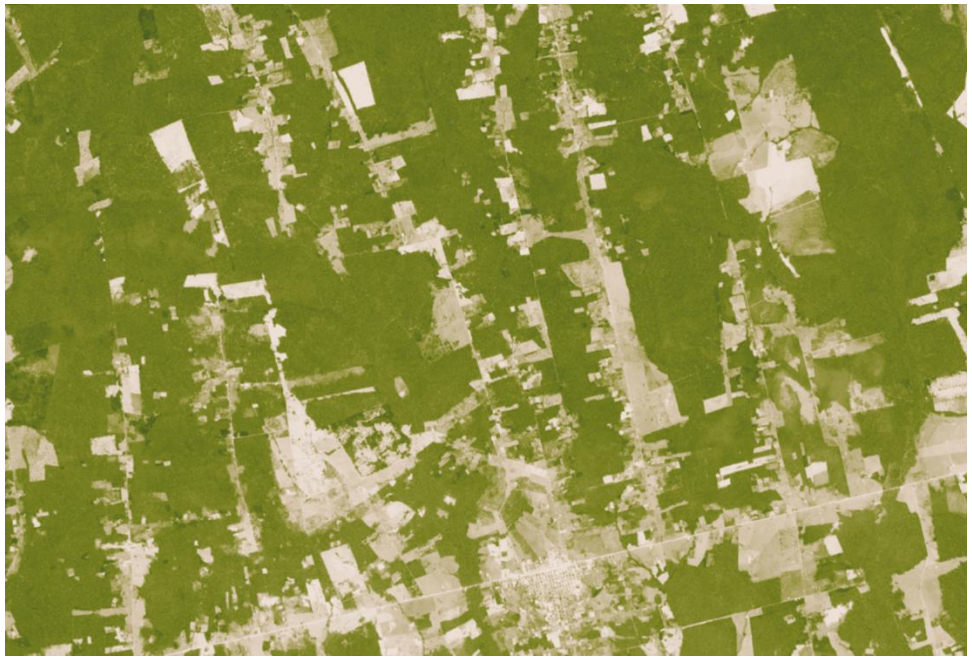


2006

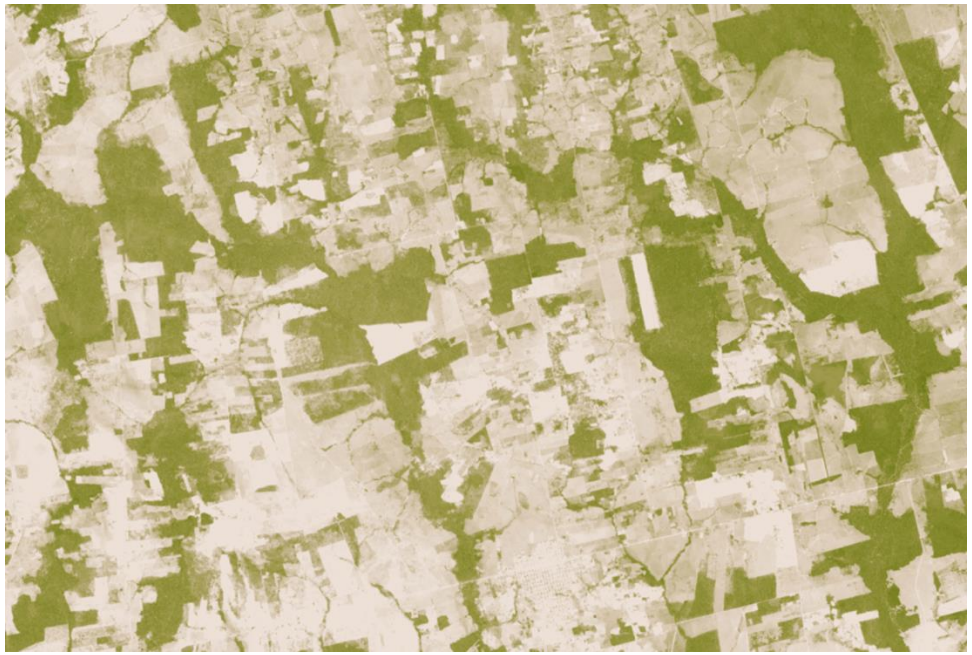


A pair of satellite images of Rondônia, Brazil (in the Amazon)
Forested areas appear green; deforested land appears tan.

2000



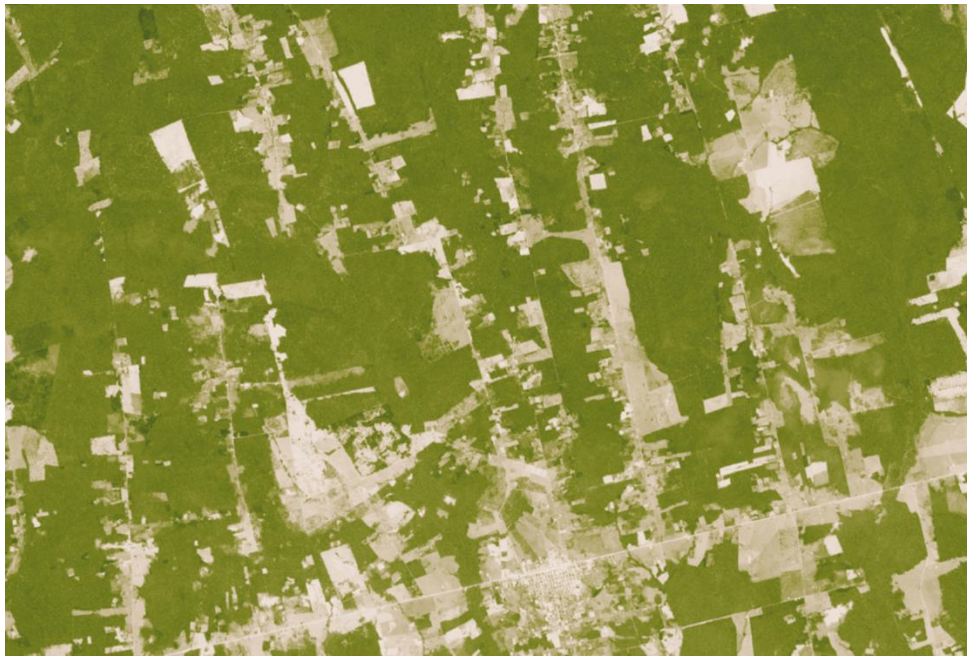
2006



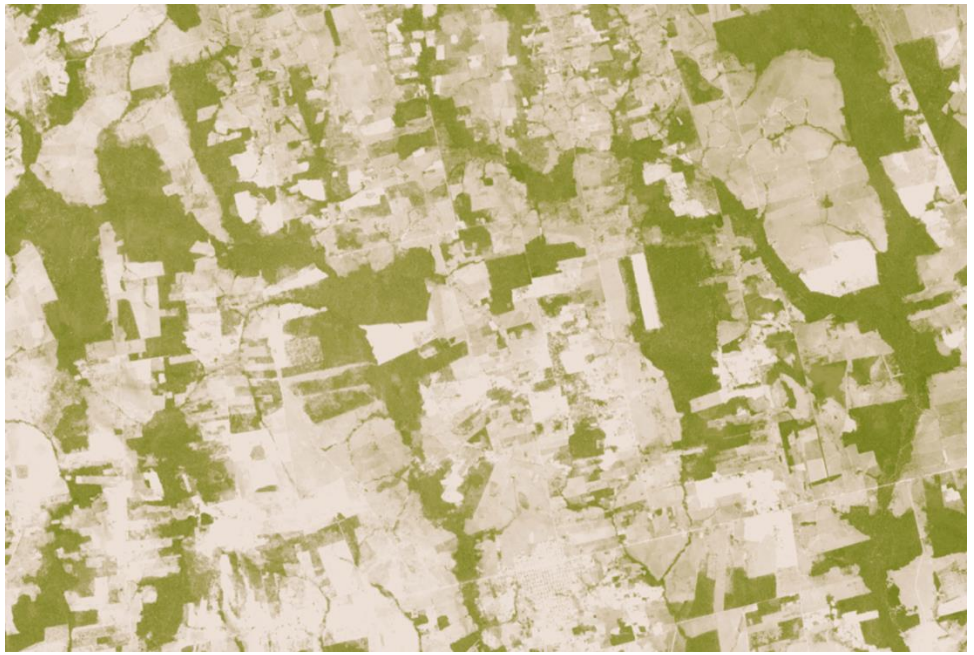
This type of image evidence can be ***quantified***: we can calculate the percentage of green pixels over time to measure how much forest was lost and where;

A pair of satellite images of Rondônia, Brazil (in the Amazon)
Forested areas appear green; deforested land appears tan.

2000



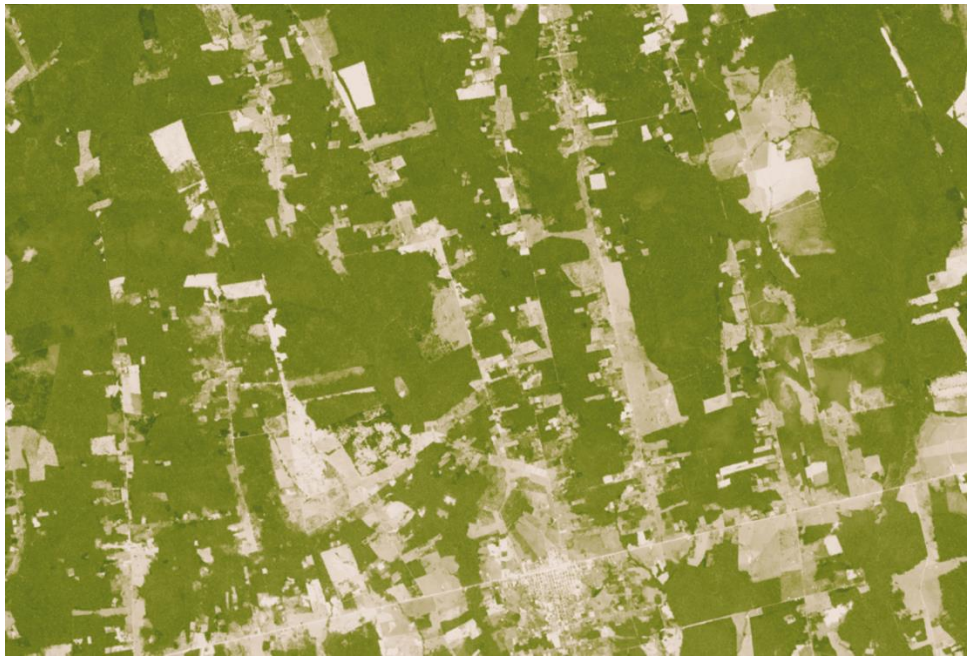
2006



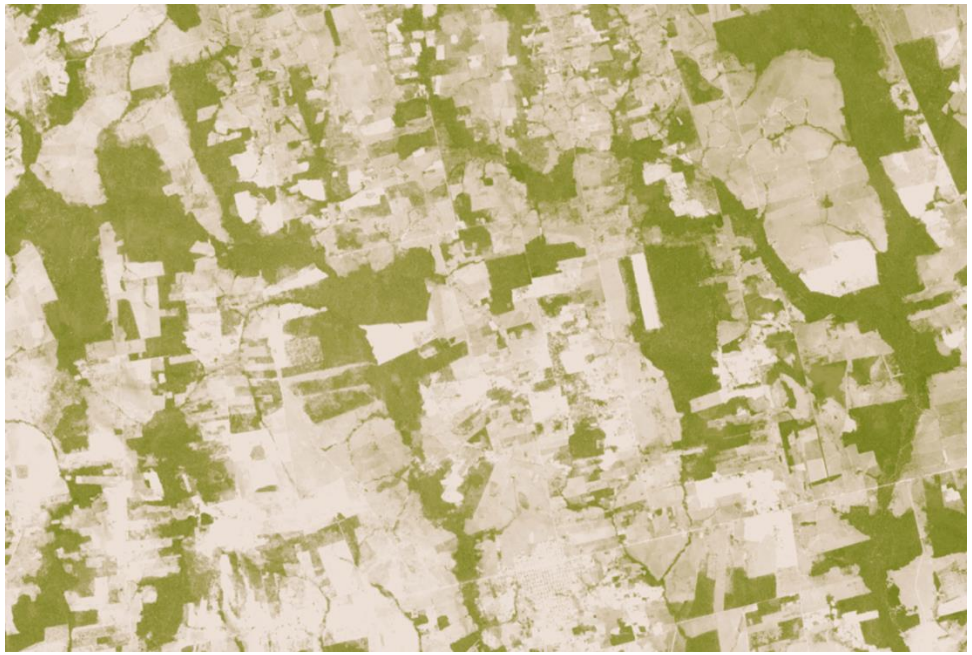
This type of image evidence can be ***quantified***: we can calculate the percentage of green pixels over time to measure how much forest was lost and where; then, we can address our research question by **correlating** these changes **with economic activities** (like farming expansion or road construction).

A pair of satellite images of Rondônia, Brazil (in the Amazon)
Forested areas appear green; deforested land appears tan.

2000



2006



Governments and researchers can now use ML to automatically analyze these images and evaluate the effectiveness of policies.

Other Examples (Street View & Beyond)

- Some quick-fire examples to inspire you:
  - **Street View & Cars:** ML analyzed 50 million street images to infer neighborhood demographics

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Case study "Using DL and Google Street View to estimate the demographic makeup of neighborhoods across the United States"

«The United States spends more than $250 million each year on the American Community Survey (ACS), a labor-intensive door-to-door study that measures statistics relating to race, gender, education, occupation, unemployment, and other demographic factors. Although a comprehensive source of data, the lag between demographic changes and their appearance in the ACS can exceed several years. As digital imagery becomes ubiquitous and machine vision techniques improve, automated data analysis may become an increasingly practical supplement to the ACS. Here, we present a method that estimates socioeconomic characteristics of regions spanning 200 US cities by using 50 million images of street scenes gathered with Google Street View cars. Using deep learning-based computer vision techniques, we determined the make, model, and year of all motor vehicles encountered in particular neighborhoods. Data from this census of motor vehicles, which enumerated 22 million automobiles in total (8% of all automobiles in the United States), were used to accurately estimate income, race, education, and voting patterns at the zip code and precinct level. (The average US precinct contains ~1,000 people.) The resulting associations are surprisingly simple and powerful. For instance, if the number of sedans encountered during a drive through a city is higher than the number of pickup trucks, the city is likely to vote for a Democrat during the next presidential election (88% chance); otherwise, it is likely to vote Republican (82%). Our results suggest that automated systems for monitoring demographics may effectively complement labor-intensive approaches, with the potential to measure demographics with fine spatial resolution, in close to real time.»

(https://pubmed.ncbi.nlm.nih.gov/29183967/)

Other Examples (Street View & Beyond)

- Some quick-fire examples to inspire you:
  - **Street View & Cars:** ML analyzed 50 million street images to infer neighborhood demographics
  - **Social Media Photos:** Researchers correlate Instagram photos' features with tourism or happiness indices (e.g., colors of photos in a city correlated with happiness surveys).

Other Examples (Street View & Beyond)

- Some quick-fire examples to inspire you:
  - **Street View & Cars:** ML analyzed 50 million street images to infer neighborhood demographics
  - **Social Media Photos:** Researchers correlate Instagram photos' features with tourism or happiness indices (e.g., colors of photos in a city correlated with happiness surveys).
  - **Satellite & Urban Change:** Tracking construction of buildings (urban growth) or post-disaster damage using before/after images.
  - **Historical images:** Analyzing historical maps with ML to study long-run development (e.g., how night lights grew in the Nile Delta since 1992 could reflect impact of policies).

Other Examples (Street View & Beyond)

- Some quick-fire examples to inspire you:
  - **Street View & Cars:** ML analyzed 50 million street images to infer neighborhood demographics
  - **Social Media Photos:** Researchers correlate Instagram photos' features with tourism or happiness indices (e.g., colors of photos in a city correlated with happiness surveys).
  - **Satellite & Urban Change:** Tracking construction of buildings (urban growth) or post-disaster damage using before/after images.
  - **Historical images:** Analyzing historical maps with ML to study long-run development (e.g., how night lights grew in the Nile Delta since 1992 could reflect impact of policies).

The only limit is creativity – if there's a visual aspect to your question, there may be data in images you can use!

Case study: Media Bias in Images (Visual Bias, Caprini 2023)

Do news outlets bias their audience not just with words, but with pictures?

- Context: People increasingly consume news via snippets (headline + lead image on social media). Images might carry ideological tones.


A
News source name @newssource · 27m
"Lorem ipsum dolor sit amet consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur."
B
gettyimages
Brooks Kraft
C
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
"Lorem ipsum dolor sit amet consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim"...
newssource.com
12
15
71
D

Case study: Media Bias in Images (Visual Bias, Caprini 2023)

Do news outlets bias their audience not just with words, but with pictures?

- Context: People increasingly consume news via snippets (headline + lead image on social media). Images might carry ideological tones.
- Data: 300k+ news images from 2019–2020, major US outlets. Method: visual vocabulary & dictionary approach (ML + manual tagging of features).

Case study: Media Bias in Images (Visual Bias, Caprini 2023)

The paper shows systematic differences by outlet ideology. Take away: Left vs Right media *don't choose the same pictures* even for similar topics.

Example: in BLM protests, right-leaning outlets more likely to show images of riots and loothing for a given protest story, whereas left-leaning outlets might show images emphasizing police violence.

Key takeaway: *Images have an ideological signature, just like text*.

From the Left

The Fed holds interest rates steady for third consecutive time

CNN Business L L C R R See rating details

From the Right

Fed leaves interest rates unchanged again and signals three cuts coming next year

Fox Business L L C R R See rating details

(Visual Bias, Caprini 2023)

From the Left

New York may use national guard to replace unvaccinated health wor...

Governor outlines plans as Monday mandate deadline looms, saying: 'We are in a battle against Covid to protect our loved ones'

theguardian.com

From the Right

New York could use National Guard to fill ranks of unvaccinated healt...

New York's health department issued an order for all health care workers to be vaccinated by Sept. 27, and the state has a plan to fill t...

fox29.com

(Visual Bias, Caprini 2023)

Can we measure visual bias?

News @sourcename

President Trump extended the administration's social-distancing guidelines for another 30 days through the end of April. These guidelines include avoiding non-essential travel, going to work, or gathering in groups of 10 or more. The extension of social-distancing restrictions came after Dr. Fauci and other members of the COVID-19 task force made the recommendation

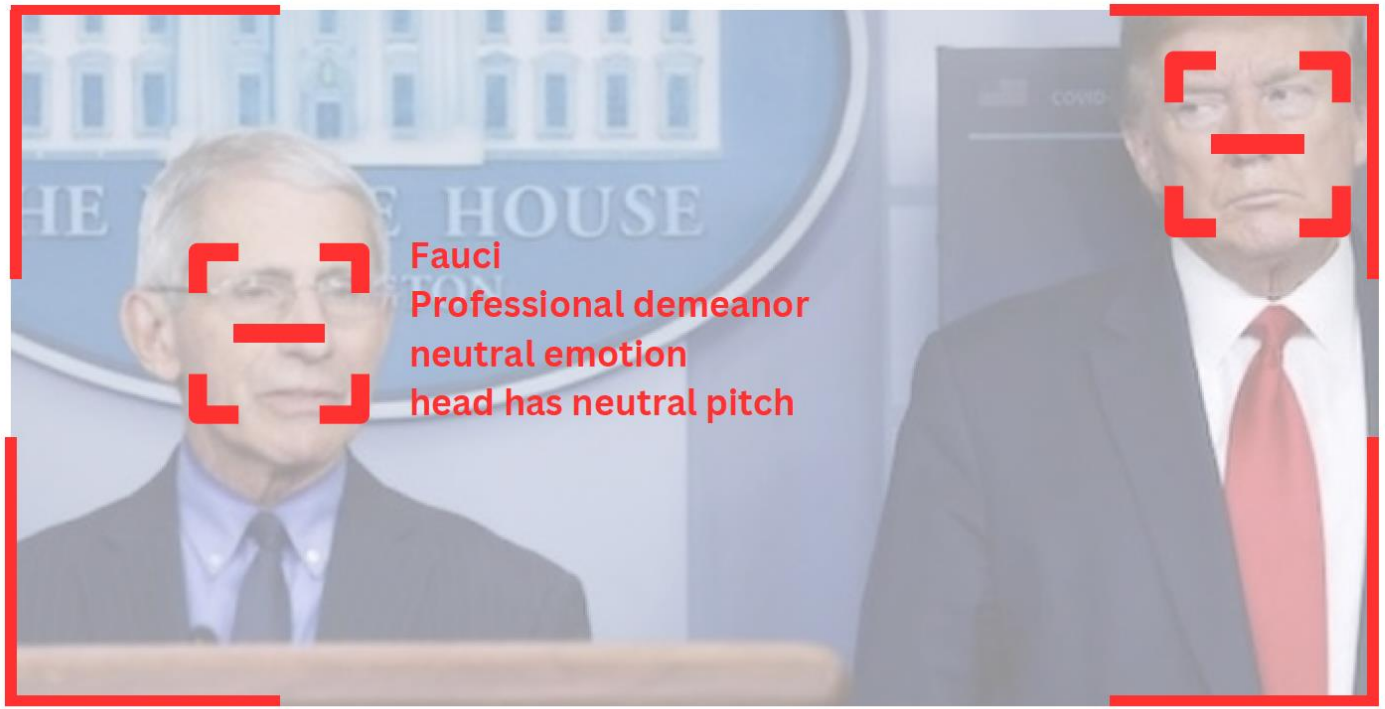
(Credits for this slide goes to Bocconi students Cicala Iorgus, Formato Francesca and Girace Ines)

Case study: Media Bias in Images (Visual Bias, Caprini 2023)

PART 1 measuring idological distance, «partisanship» in pics

- I develop a dictionary of visual features, which includes all graphic aspects deemed important in political communication by the literature.
  Using this dictionary, I measure the ideological distance across images.

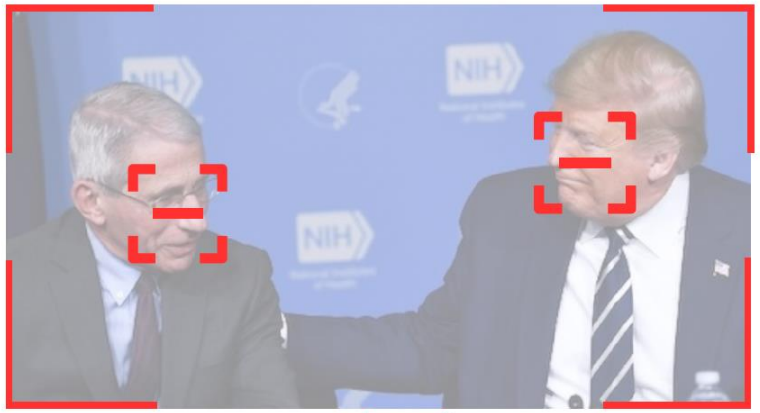
Visual Tokens



(Credits for this slide goes to Bocconi students Cicala Iorgus, Formato Francesca and Girace Ines)

The effectiveness of anti-Covid measures

Dem-leaning



Rep-leaning

	Dem-leaning	Rep-leaning
Subject tokens	Fauci, Trump	Fauci, Trump
Adjective Features	**Fauci**: professional demeanor, neutral emotion, head has neutral pitch. **Trump**: Neutral-to-angry expression, Medium-Low centrality, looking towards Fauci.	**Trump**: Positive emotion (smiling); There is body contact
Contextual Features	Institutional backdrop: White House press room, neutral color tones	Institutional backdrop: NIH branding; bright lighting and color tones.

(Credits for this slide goes to Bocconi students Cicala Iorgus, Formato Francesca and Girace Ines)

Intuition on the method:

1 ▷ 2 ▷ 3

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

(Visual Bias, Caprini 2023)

Intuition on the method:

1

Pass images to computer vision algorithms, get "*raw information*"

2

3

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

(Visual Bias, Caprini 2023)

Intuition on the method:

1

Pass images to computer vision algorithms, get "*raw information*"

2

3

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

- face detect & recognition
- object detection,
- emotion recognition,
- complement with other data (voteview.com) etc.

(Visual Bias, Caprini 2023)

Example from “face detection” algorithm


```
Risultato del rilevamento:
detection_01
JSON:
 [
  {
    "faceId": "3d06d80f-0324-4896-8b94-20bc
    "faceRectangle": {
      "top": 197,
      "left": 579,
      "width": 176,
      "height": 176
    },
    "faceAttributes": {
      "hair": {
        "bald": 0.11,
        "invisible": false,
        "hairColor": [
          {
            "color": "gray",
            "confidence": 0.97
          },
          {
            "color": "blond",
            "confidence": 0.92
          },
          {
            "color": "other",
            "confidence": 0.41
```


(Visual Bias, Caprini 2023)

Intuition on the method:

1 ▶ 2 ▶ 3

Pass images to computer vision algorithms, get "*raw information*"

Use annotated meaning to tag elements by their syntax role

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

(Visual Bias, Caprini 2023)

Intuition on the method:

1 ▷ 2 ▷ 3

Pass images to computer vision algorithms, get "*raw information*"

Use annotated meaning to tag elements by their syntax role

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

- **subject:** people, McCarty
- **adj. (kinesics):** glance, smile
- **adj. (size):** close shot
- **context**: flags, speech

(Visual Bias, Caprini 2023)

Intuition on the method:

1 ▷ 2 ▷ 3

Pass images to computer vision algorithms, get "*raw information*"

Use annotated meaning to tag elements by their syntax role

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

Exploit grammar to build <u>interpretable and meaningful</u> combinations

(Visual Bias, Caprini 2023)

Intuition on the method:

1 ▶ 2 ▶ 3

Pass images to computer vision algorithms, get "*raw information*"

Use annotated meaning to tag elements by their syntax role

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

tokens made of single features (circa 520k)

Exploit grammar to build <u>interpretable and meaningful</u> combinations

- **subject:** people, McCarty
- **adjectives (kinesics):** glance, smile
- **adjectives (size):** close shot, background
- **context**: flags, speech

(Visual Bias, Caprini 2023)

Intuition on the method:

1 ▷ 2 ▷ 3

Pass images to computer vision algorithms, get "*raw information*"

Use annotated meaning to tag elements by their syntax role

Kevin McCarthy	Three Republicans	…	Two smile and look McCarty
1	1	0	1

Exploit grammar to build <u>interpretable and meaningful</u> combinations

▽

tokens made of features combinations (circa 2.6M)

"Kevin McCarthy, observed by two people who smile"

(Visual Bias, Caprini 2023)

- I **map images** in my dataset **to the vector of tokens** in my visual vocabulary.

image *i*

Kevin McCarthy	Close shot of a man	Woman with a microphone	…
1	1	0	

$\mathbf{v_i}$ = vector representation of *i* (length: 34’647)

(Visual Bias, Caprini 2023)

Perks of creating a dictionary of visual tokens:
1) All tokens are interpretable by design.
2) Modelling the interdependence of visual language elements enables both lexical and semantic analysis (= we gain deeper reading).

(Visual Bias, Caprini 2023)

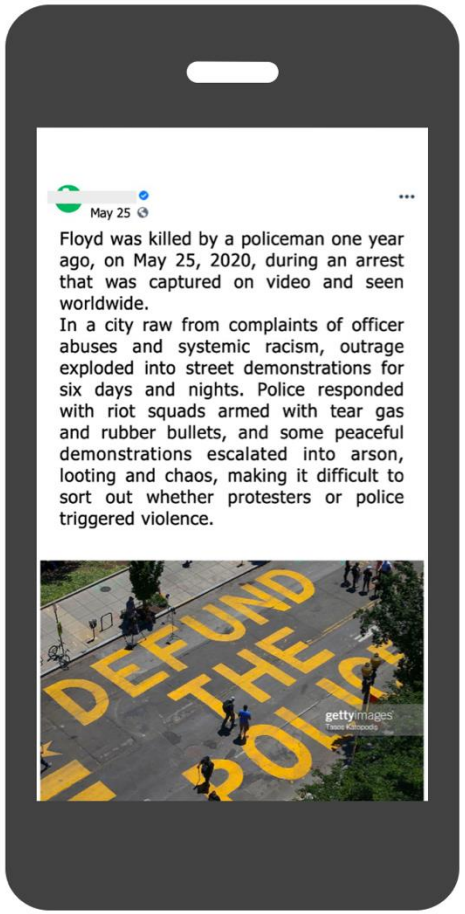
Case study: Media Bias in Images (Visual Bias, Caprini 2023)

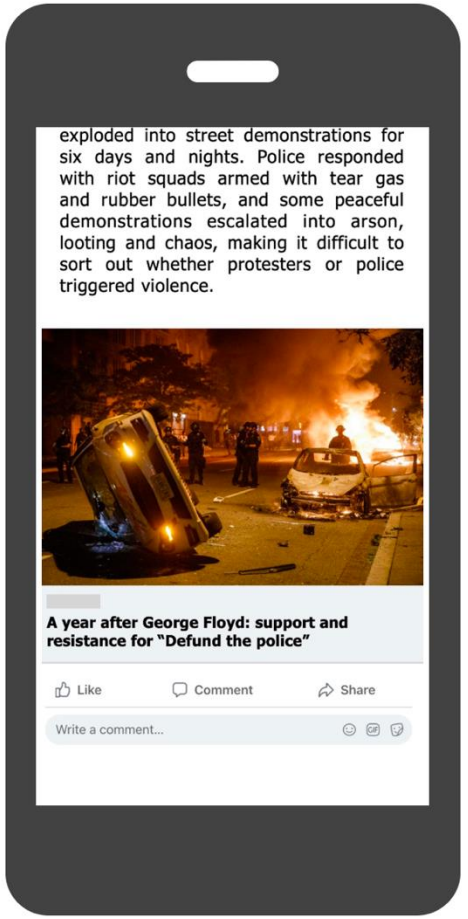
PART 2 Infuence on Audience

- Using an experiment – identical news text, different image- the paper finds that readers' opinions shifted **toward the bias of the image**

Survey experiment

- I expose 2'000 respondents to news previews on 5 issues, varying (only) the leading images: Non-partisan, Dem-leaning, Rep-leaning.





(Visual Bias, Caprini 2023)

Case study: Media Bias in Images (Visual Bias, Caprini 2023)

Take away from the paper: bias is conveyed both via text and via images.

But fact-checking today only focuses on text!

Therefore: we need to develop measurement tools and education initiatives that focus on the relationship between text and images.

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Hands-On Session – LLM + Python Intro

What is Python?

Python is a popular programming language used for:

- Building websites
- Running artificial intelligence
- Doing scientific research
- Analyzing data

It's **easy to read** and **widely used** by companies, researchers, and students!

Comparing languages

	Python	R	Stata (Mata)
Type of Language	General-purpose	Statistical-focused	Menu-driven + low-level Mata
Main Use	ML, AI, apps, web, data	Data analysis, stats, graphs	Econometrics, regressions
Machine Learning	Best choice	Some support (via add-ons)	Very limited (better in newer versions)
Language Style	Like English, clean & flexible	More statistical, less general	Mostly commands or menus
Statistics Packages	Many (but not always built-in)	Tons of built-in stats tools	Built-in stats (plus Mata)
Visualization	Good (matplotlib, seaborn)	Excellent (ggplot2)	Basic graphs
Used By	ML engineers, data scientists	Statisticians, academics	Economists, social scientists
Cost	Free (open source)	Free (usually)	Paid license

Comparing languages- which software?

Python:

- **Jupyter Notebook** (great for data & ML)
- **Google Colab** (free, runs in your browser!)
- **Visual Studio Code** (for coding everything)

R:

- **RStudio** (clean interface, good for data work)

Stata:

- Menu-based interface (click or type commands)
- Mata runs behind the scenes (for advanced users)

Live coding session using Colab

Live Coding: Let's Analyze Images!

- Using **Google Colab** (cloud notebook) – no install needed, just a browser.
- **LLM-assisted coding:** We'll use plain English to generate/explain code.
- Goals: Load an image, do simple analysis (brightness, etc.), and see ML in action on visuals.

Don't worry if you're new to coding – follow along, and focus on the concepts. The code is provided (and AI can help us!)

You can open the notebook yourself if desired,

Click here or use the QR code:

You will need to log in to a google account (gmail)

https://colab.research.google.com/github/GCapr/teaching_material/blob/Padova-2025-Summer-school-scuola-Galileiana/ML_Econ_Image_Analysis_Full_Demo.ipynb

Menu for today:

1. **Some vocabulary**
2. **A brief history of ML**
3. **Applications and case studies**: Satellite Images for Econ & Environment, “Visual Bias” in news images
4. **Hands-on**: ML with Python for image data (no coding exp. needed!)

+ Wrap-up and Q&A

Q&A