The Machine Learning Taxonomy

Organizing 40+ Tasks by their Mathematical Roots

Nipun Batra · IIT Gandhinagar

Before We Begin: A Simple Question

You use machine learning **every single day**.

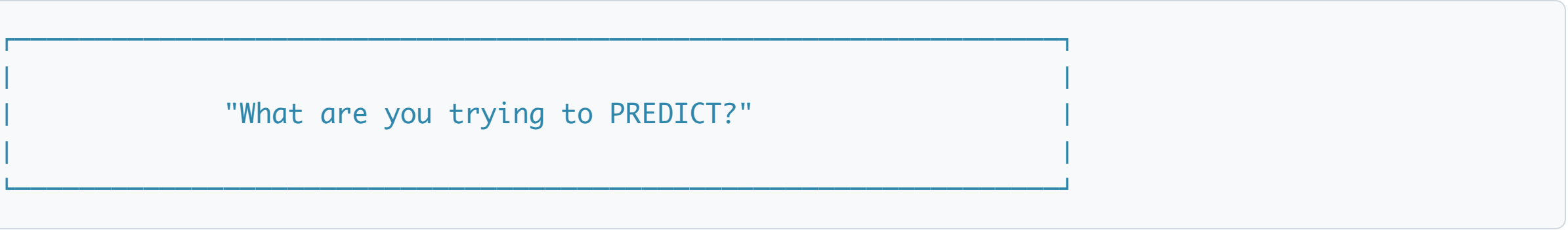
Can you identify where?

Time	Application	ML Behind It
Morning	Phone unlocks with your face	Face Recognition
Commute	Google Maps predicts traffic	Time Series Prediction
Email	Gmail filters spam, suggests replies	Text Classification + Generation
Music	Spotify recommends songs you'll love	Recommendation Systems
Shopping	Amazon shows "You might also like..."	Collaborative Filtering
Photos	Google Photos groups by faces, finds "beach"	Clustering + Image Classification
Evening	Netflix suggests what to watch	Recommendation Systems
Chat	You ask ChatGPT a question	Language Models (Generative AI)

Each of these is a different ML task!

The Big Insight

Every ML task boils down to **one question**:



Predicting a Category?

→ Classification

"Is this email spam?"

Predicting a Number?

→ Regression

"What will be the price?"

Predicting a Sequence?

→ Seq2Seq

"How do you say this in French?"

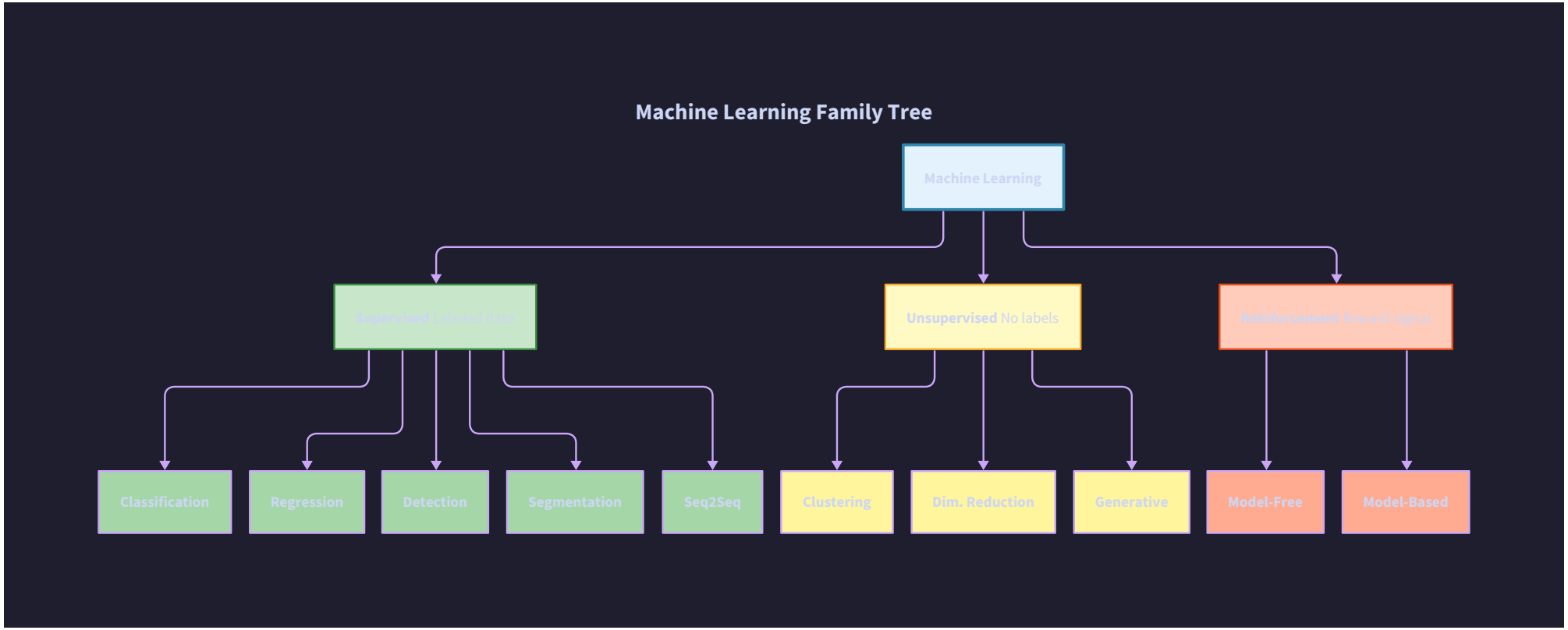
Creating Something New?

→ Generative

"Draw me a cat in space"

> Once you know the "output type", you know which family the task belongs to!

The Master Taxonomy



Section 1: Classification

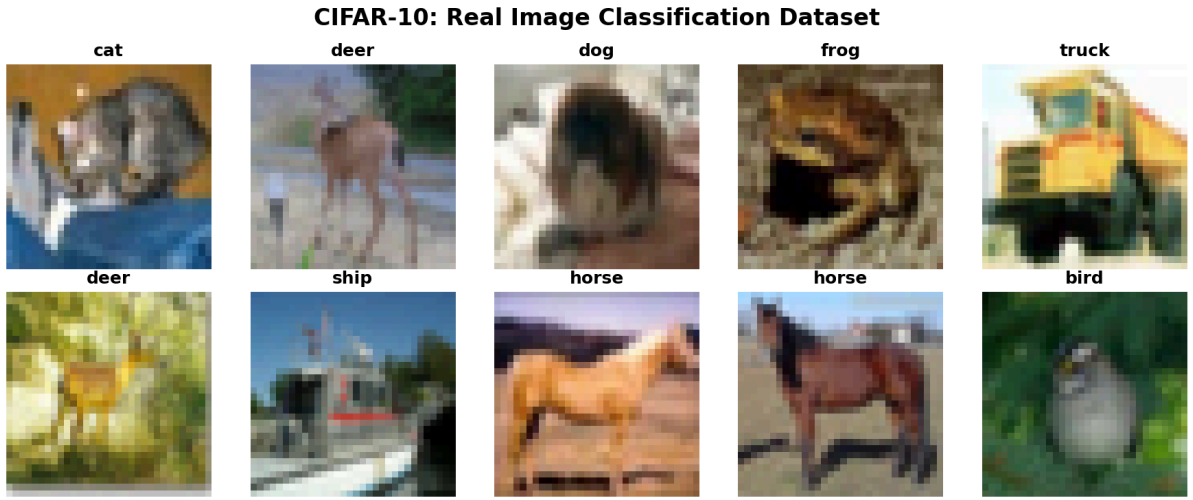
"Which Bucket Does This Belong To?"

Classification: You Already Know This!

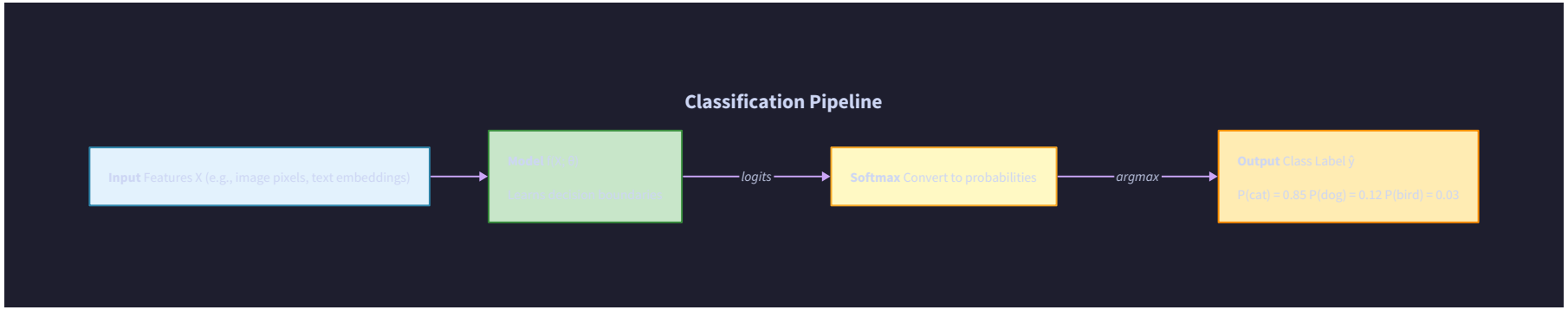
Think about how YOU classify things every day:

Question	Possible Answers
"Is this mushroom safe to eat?"	Edible / Poisonous
"What animal is in this photo?"	Dog / Cat / Bird / ...
"Should I trust this email?"	Legitimate / Spam
"What number is written here?"	0, 1, 2, ... 9

You look at the input and pick **one category** from a fixed set. That's classification!

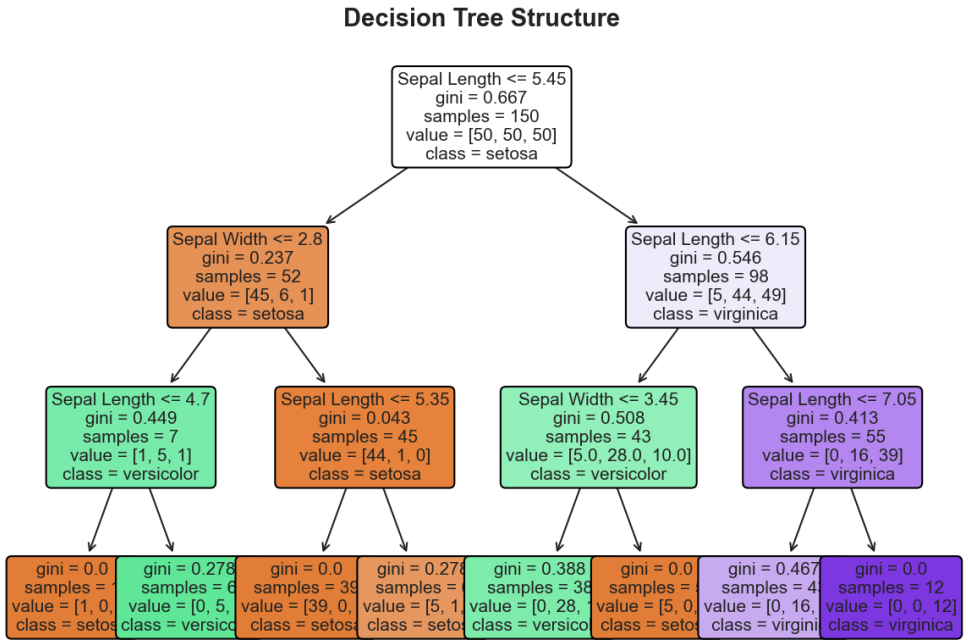


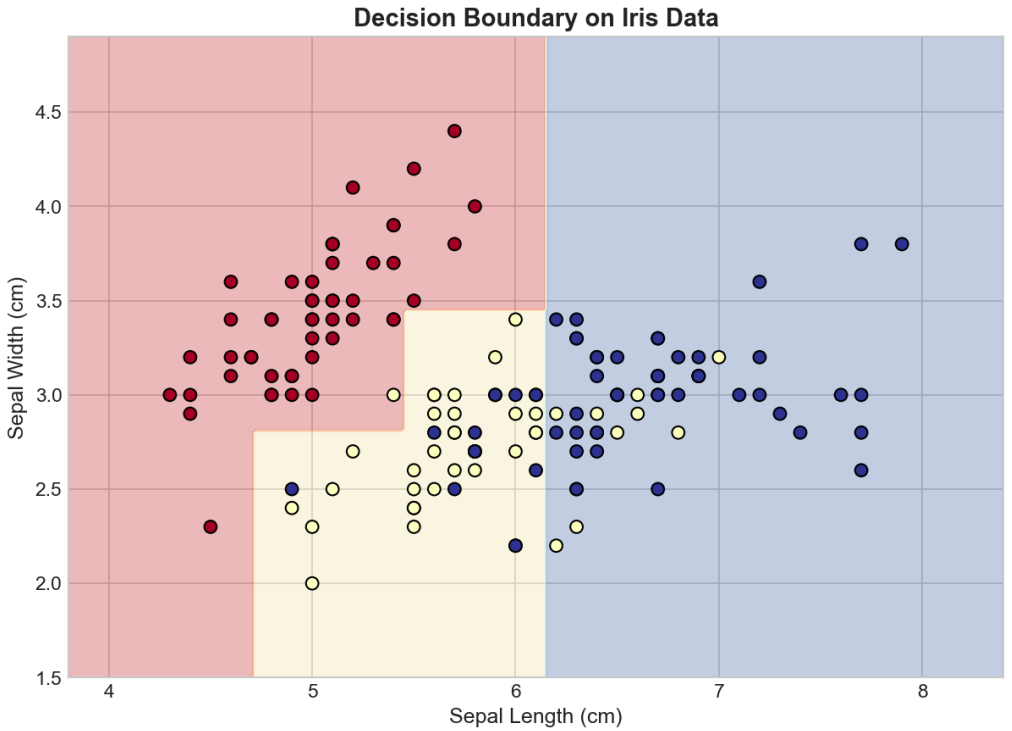
Classification: The Core Idea



The model learns patterns that distinguish categories, then applies those patterns to new inputs.

Classification in Action: Decision Trees





A decision tree learns **if-then rules** from data: "If sepal length > 5.5 AND sepal width < 3.0, then iris-versicolor"

Example: How Does Email Spam Detection Work?

Binary vs Multi-Class Classification

Binary Classification

Two possible outcomes

```
Input: Tumor image
Output:
  ○ Benign
  ● Malignant

  (Only 2 choices)
```

Examples:

- Spam / Not Spam
- Fraud / Legitimate
- Pass / Fail
- Yes / No

Multi-Class Classification

Many possible outcomes

```
Input: Animal photo
Output:
  ○ Dog
  ○ Cat
  ● Bird      ← Winner!
  ○ Fish
  ○ Horse

  (Many choices, pick ONE)
```

Examples:

- Digit recognition (0-9)
- ImageNet (1000 classes)
- Emotion detection (6+)

Multi-Label Classification

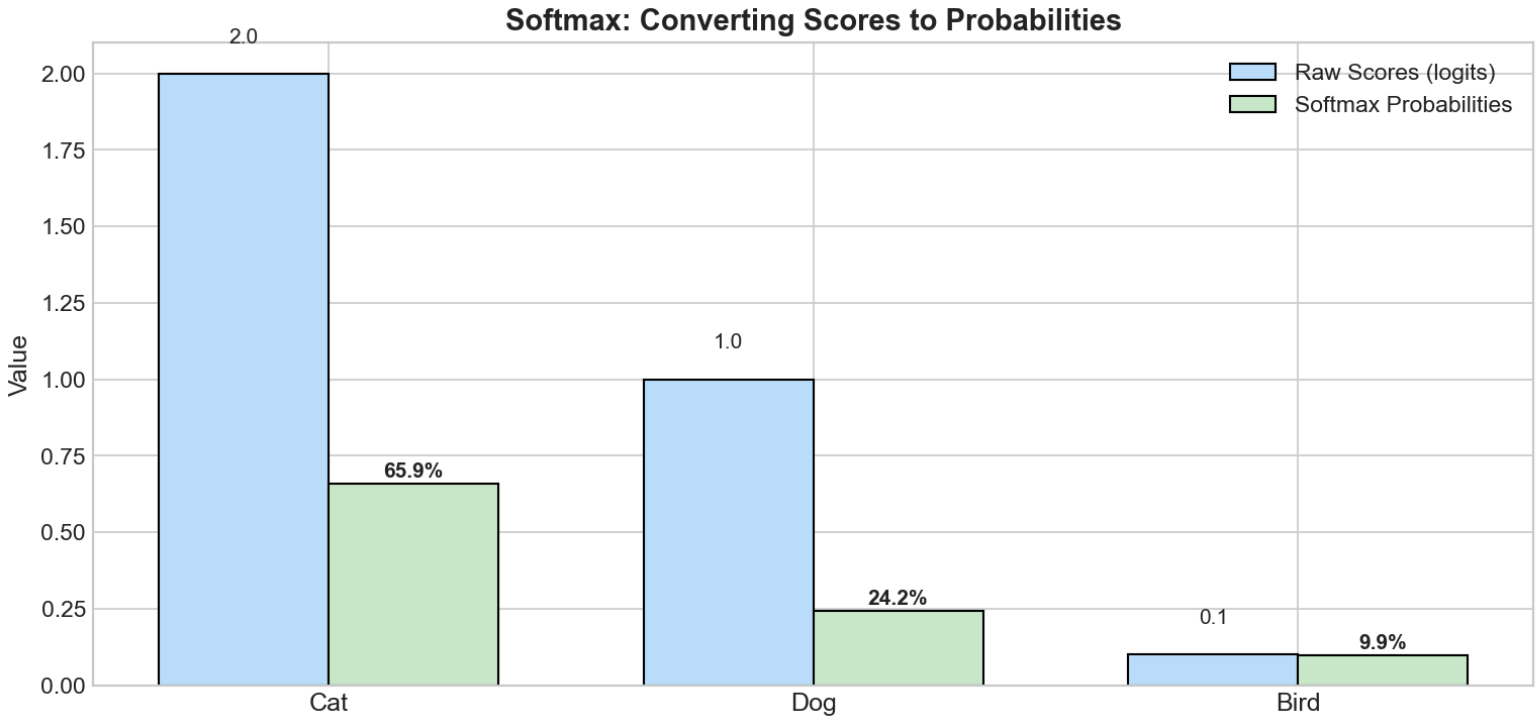
Wait, what if something belongs to MULTIPLE categories?

```
Movie Classification:

Input: "The Avengers"

Binary/Multi-class would say: "Action" (pick one!)

But actually it's: ✓ Action
                   ✓ Sci-Fi
                   ✓ Adventure
                   ○ Romance
                   ○ Documentary

(Multiple labels can be TRUE at once!)
```

Real-world multi-label examples:

- News article topics (Politics AND Economy AND International)
- Product categories (Electronics AND Computers AND Accessories)

The Math Behind Classification: Softmax



$$P(class_i) = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

Sum of probabilities = 1.0

Softmax converts raw scores (logits) to probabilities that sum to 1.

The model isn't just saying "Cat" — it's saying "85% sure it's a cat!"

Classification: Real-World Examples

Application	Input	Output	Impact
Face Unlock	Selfie	"Is this the owner?"	Security
Medical X-ray	Image	Healthy/Pneumonia/COVID	Healthcare
Credit Approval	Application	Approve/Deny	Finance
Sentiment	Tweet	Positive/Negative/Neutral	Marketing
Plant Disease	Leaf photo	38 disease types	Agriculture
Quality Control	Product photo	Pass/Fail	Manufacturing

> Classification is everywhere! It's the "Hello World" of machine learning.

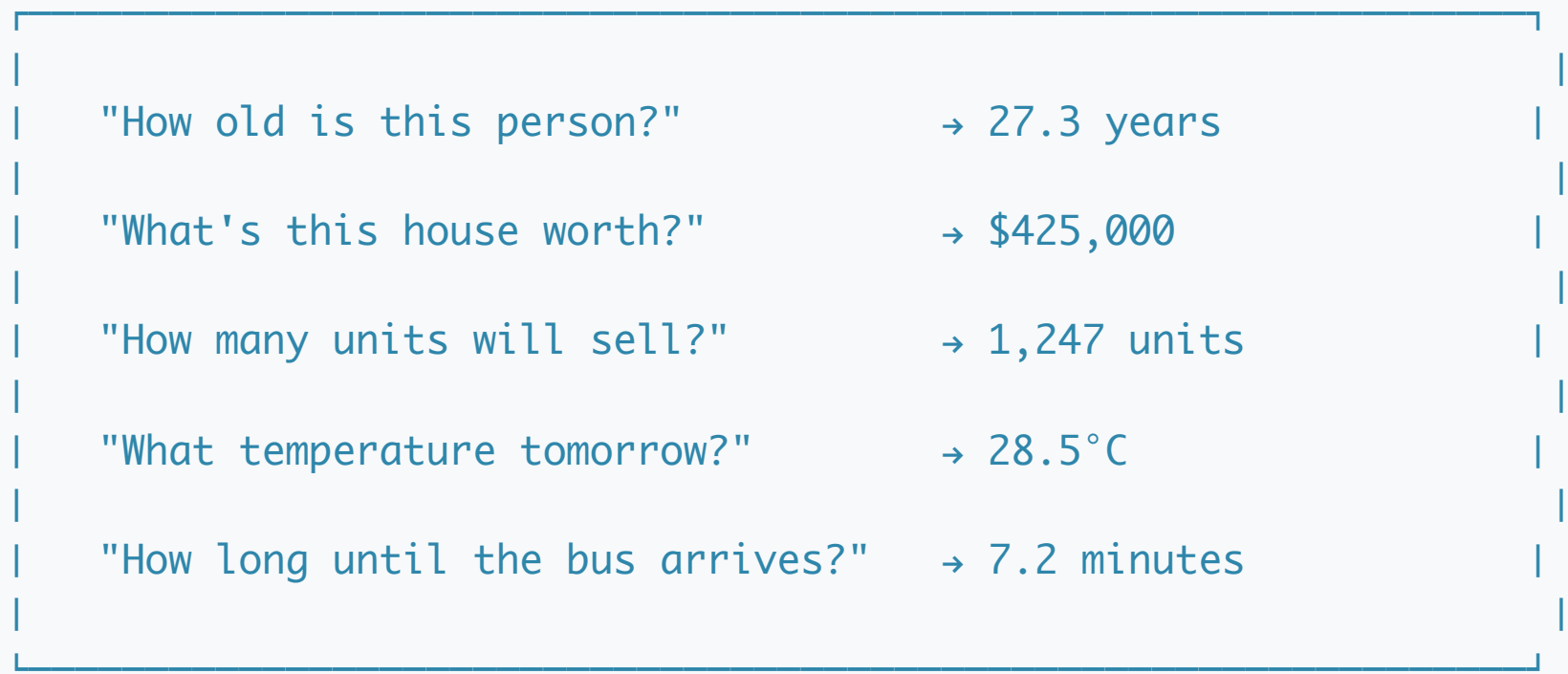
Section 2: Regression

"How Much? How Many?"

Regression: When the Answer is a Number

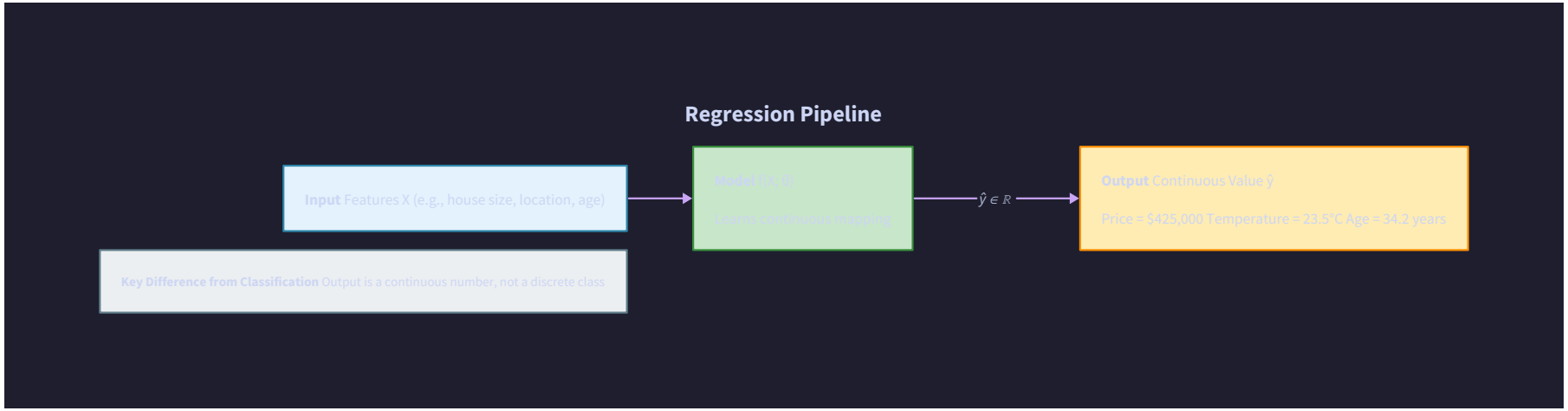
Classification: *"Which category?"* → Discrete answer

Regression: *"How much?"* → Continuous number



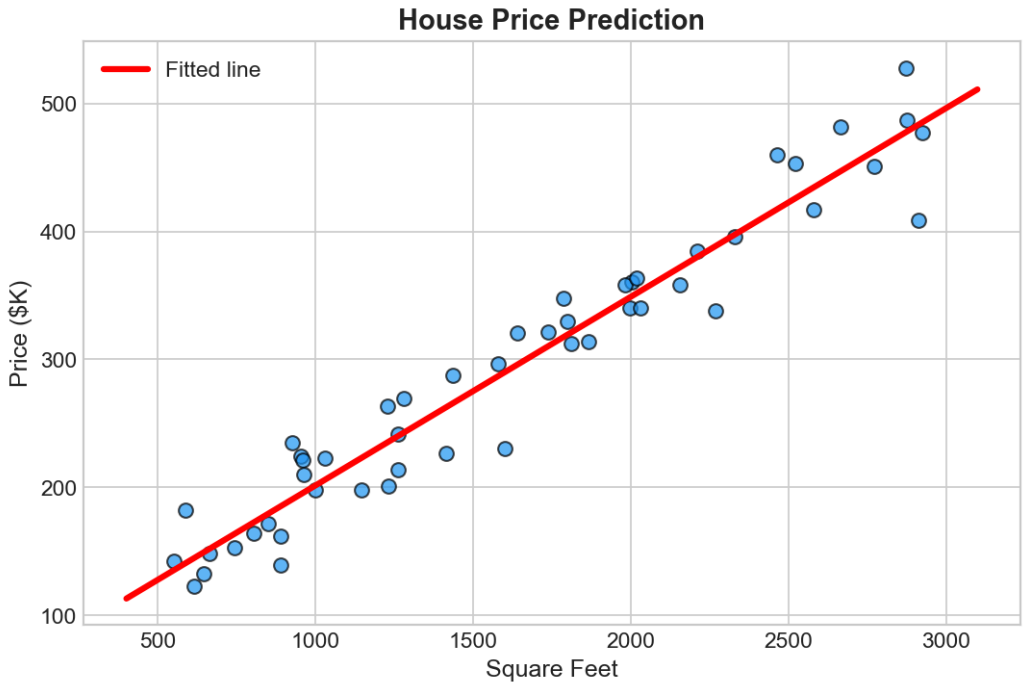
The output is **any number** on a continuous scale!

Regression: The Core Idea



Instead of choosing from buckets, we predict a specific point on a number line.

Regression in Action: Linear Regression



Linear Regression Formula:

$$\hat{y} = w_0 + w_1 \cdot x$$

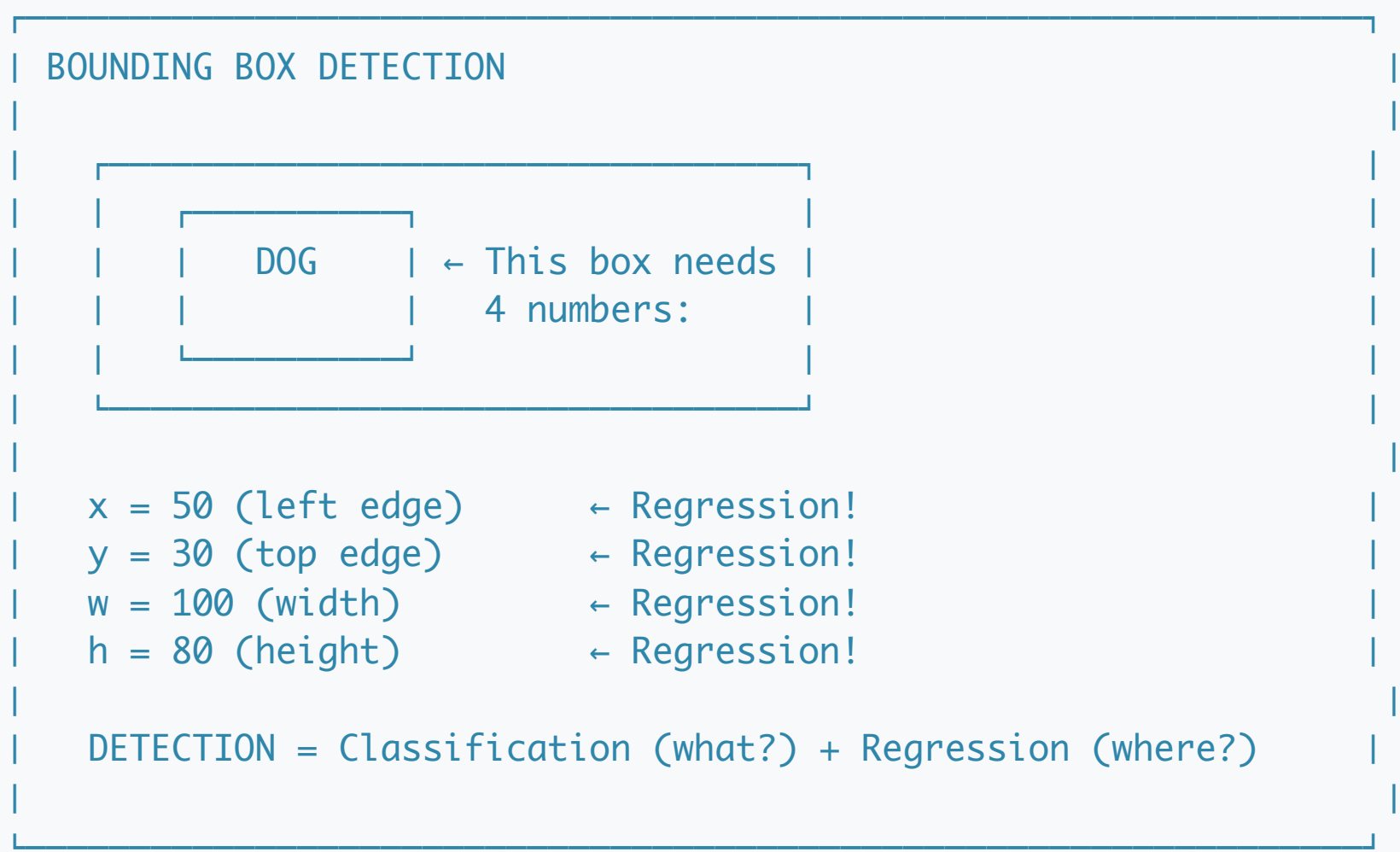
Fitted: price = 54, 241 + 147 * sqft

Each extra sqft adds ~$150 to price!

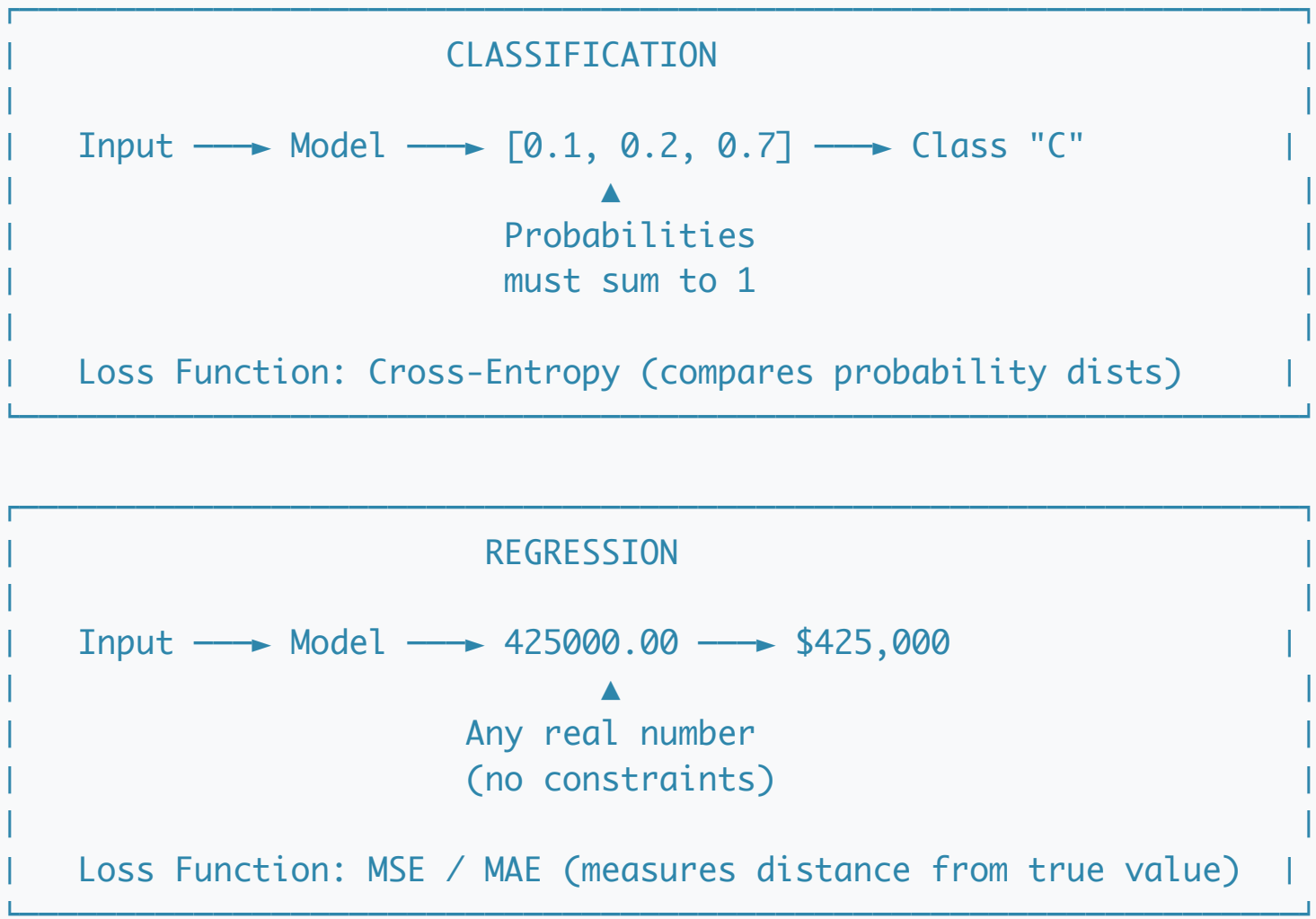
The model learns: **Price = $50,000 + $150 * (square feet)**

Regression is Actually Everywhere!

You might think you're looking at classification, but often it's regression:

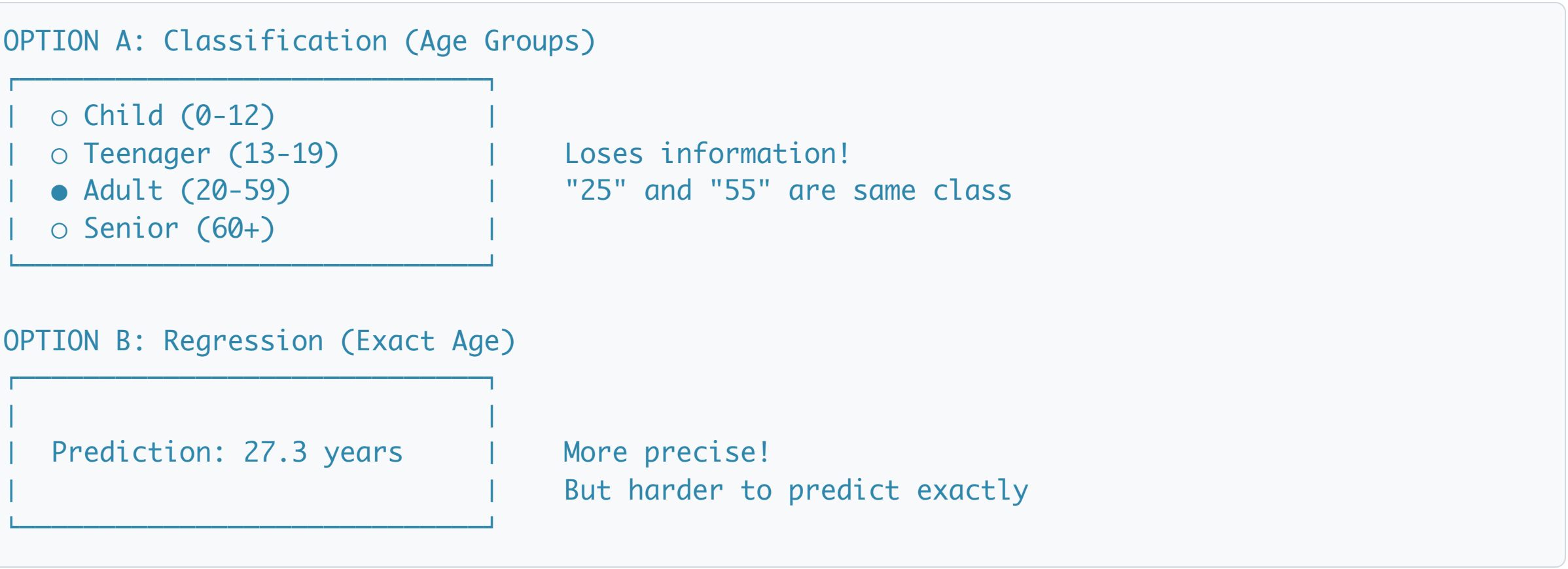


Classification vs Regression: Side by Side



The Confusion: Age Prediction

Is predicting someone's age classification or regression?



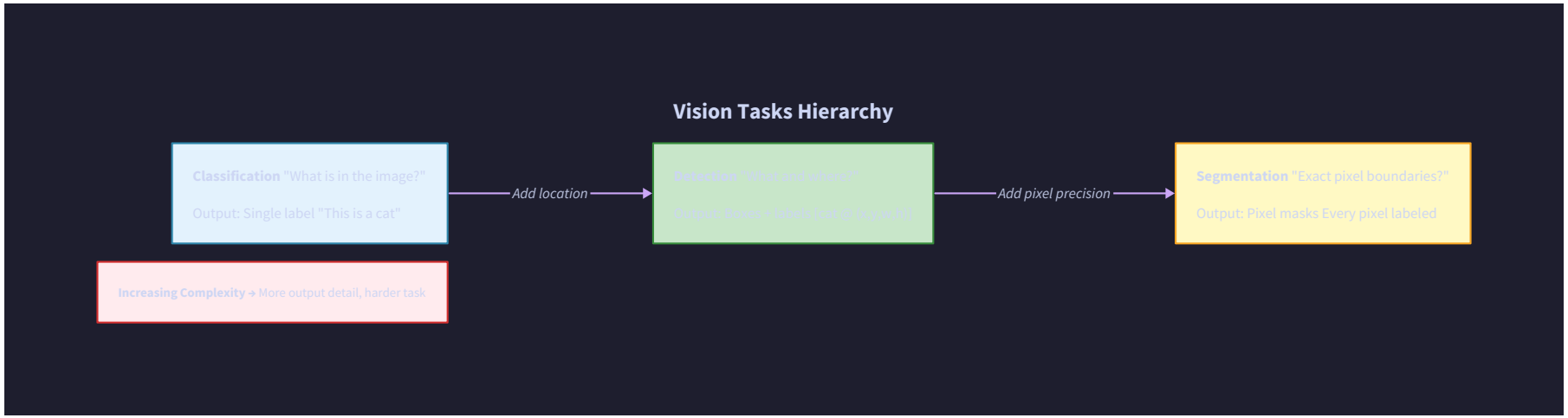
The choice depends on your application! For ID verification: regression. For marketing segments: classification might be enough.

Section 3: Vision Hierarchy

From Labels to Pixels

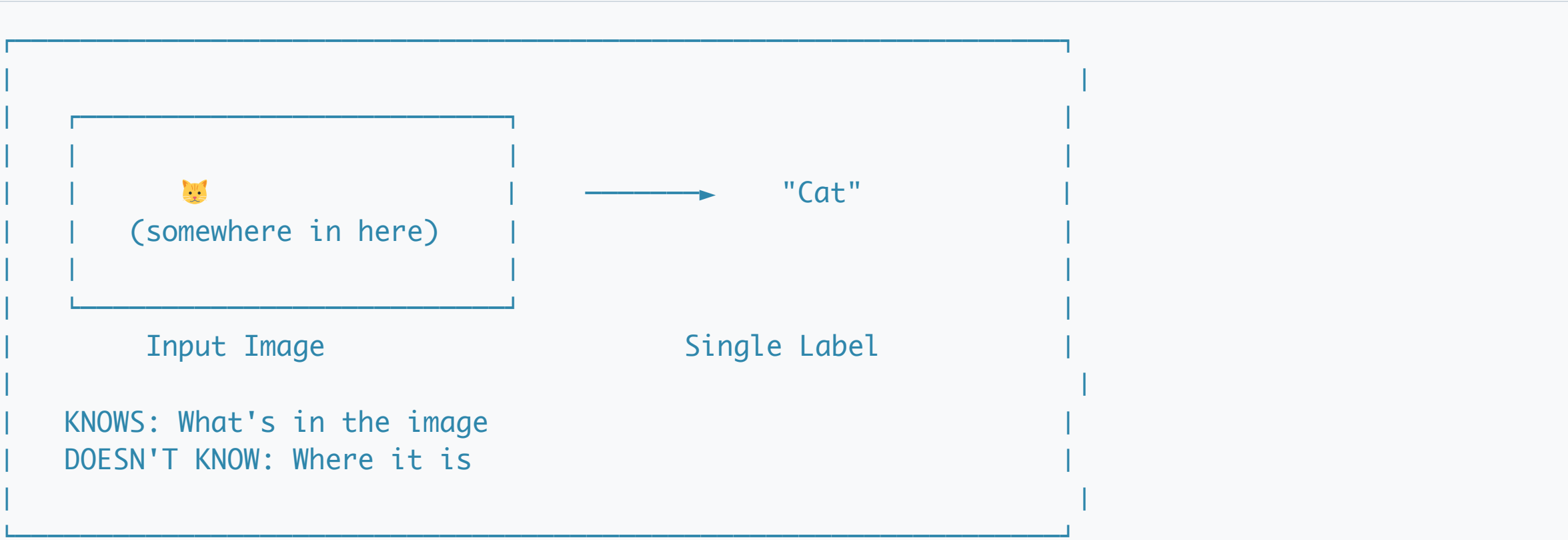
The Computer Vision Ladder

Each level gives you **more information** about what's in the image:



> Each level builds on the previous. More precision = More complexity = More data needed.

Level 1: Image Classification

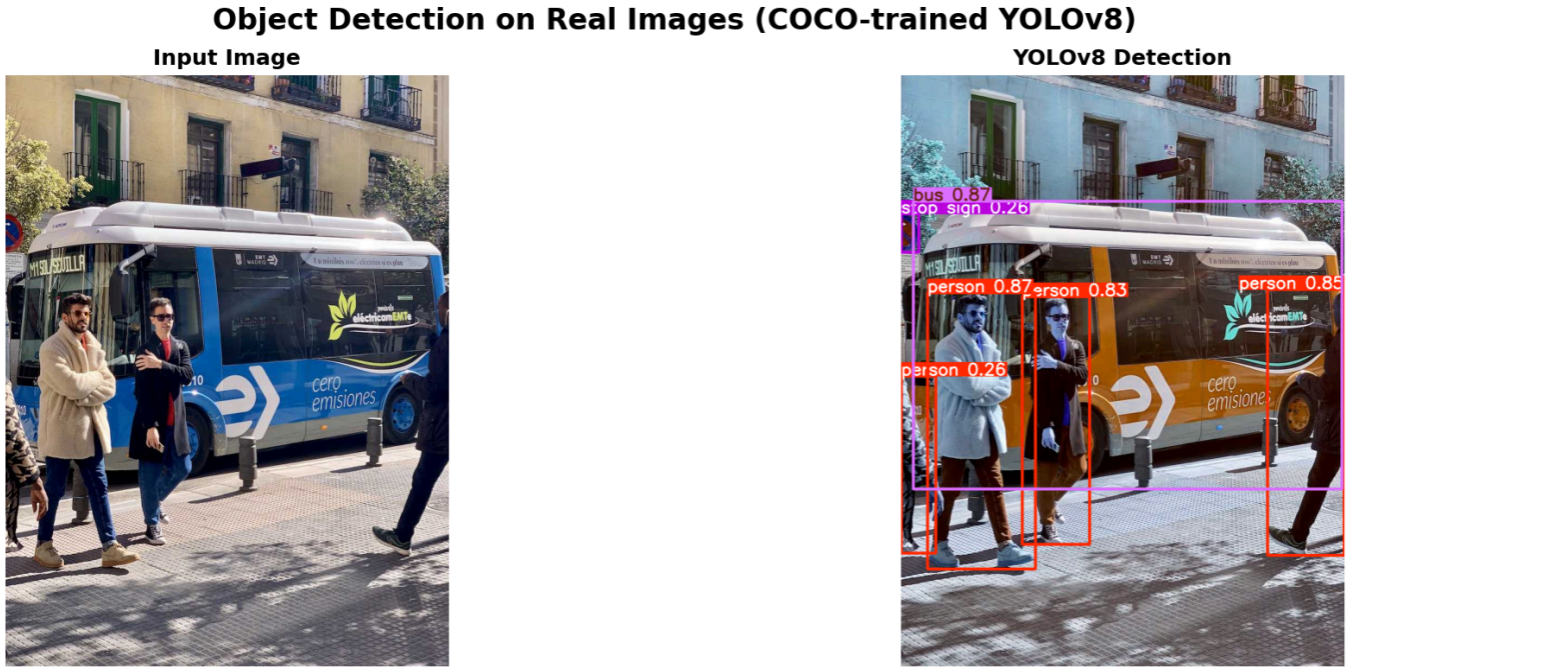


Use Cases:

- Google Photos: "Show me all photos with dogs"
- Medical: "Is this X-ray normal or abnormal?"
- Quality Control: "Is this product defective?"

Level 2: Object Detection

Detection = Classification (what) + Regression (where)



Level 3 & 4: Segmentation

Semantic Segmentation

Same class = Same color
Can't distinguish cars!

Instance Segmentation

Each object = Unique ID
Car #1 vs Car #2!

Key Difference

Semantic: pixel -> class

Instance: pixel -> class + object ID

Self-driving cars need Instance to track individual vehicles!

Instance Segmentation in Action

Instance Segmentation: Pixel-Perfect Object Boundaries

Input Image

Instance Segmentation (YOLOv8-seg)

Self-driving cars need Instance Segmentation — they must track WHICH car is doing what!

Real-World Vision Hierarchy Example

Autonomous Driving: Why Each Level Matters

Level	Task	What it Tells the Car
1	Classification	"There are cars and people in this scene"
2	Detection	"Car at (100,200), Person at (300,150)"
3	Semantic Seg.	"The drivable road area is these pixels"
4	Instance Seg.	"This is Car #1, that is Car #2" - can track each!

> Higher levels give more precise information but require more data and compute!

Section 4: Sequence Tasks

When Order Matters

Why Sequences Are Special

Some data comes in **ordered** form where **position matters**:

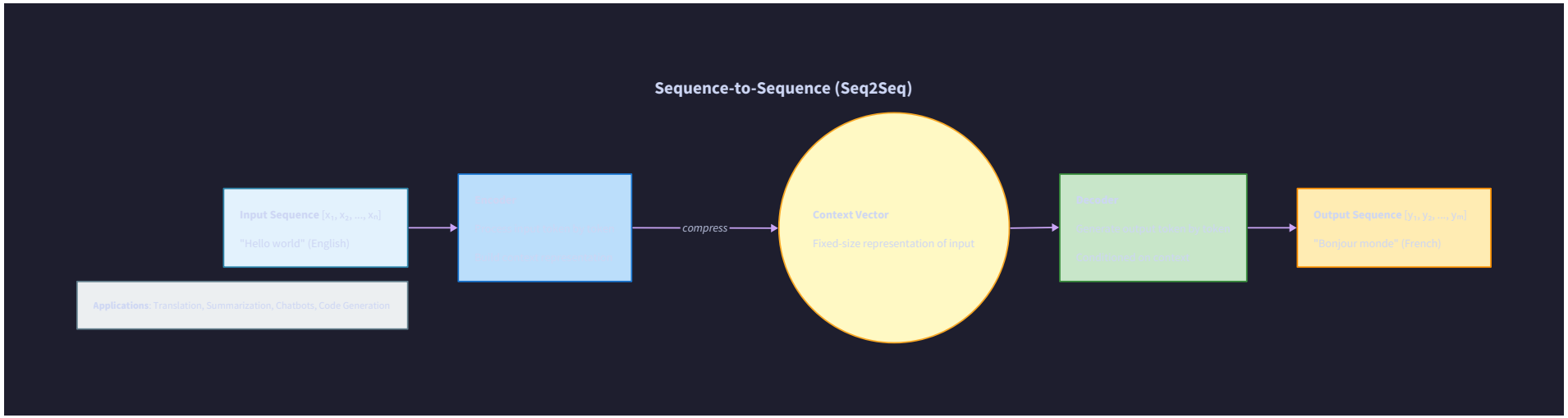
Data Type	Original	Shuffled	Problem
Text	"I love you"	"You love I"	Grammatically wrong!
DNA	ATCGATCG	GATCATCG	Different gene!
Audio	Do-Re-Mi	Mi-Re-Do	Different melody!
Video	Frame 1→2→3	Frame 3→2→1	Forward vs Backward!

> For sequences, we need models that understand ORDER, not just content!

Sequence-to-Sequence (Seq2Seq)

Input sequence → Model → Output sequence

(Lengths can be DIFFERENT!)



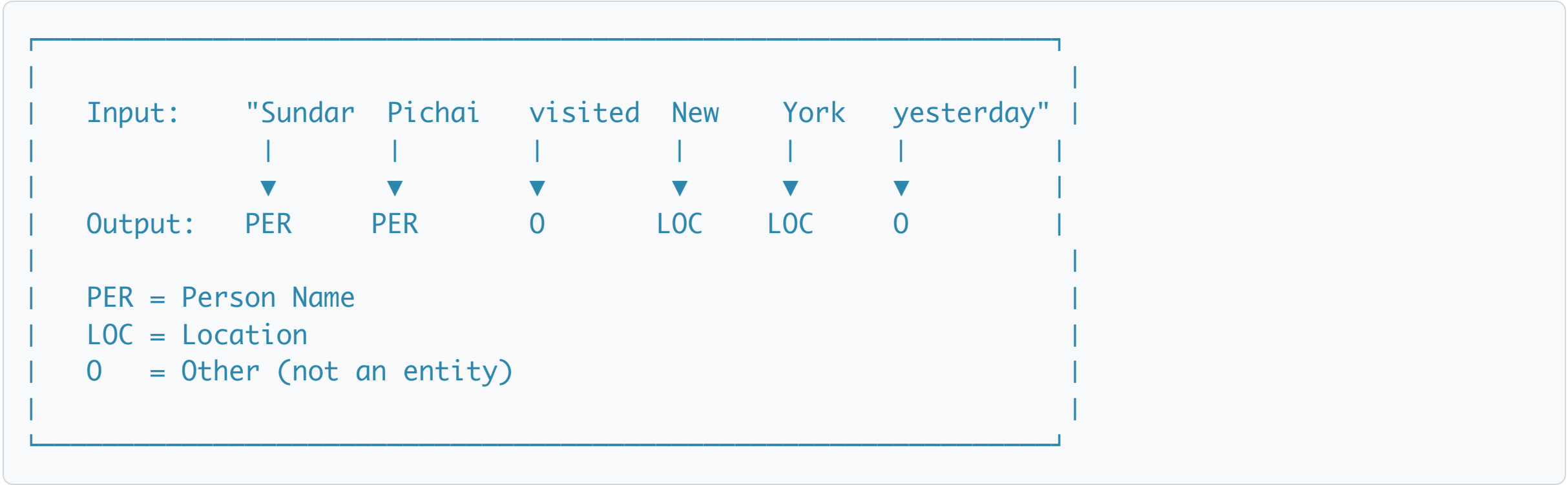
Seq2Seq Examples

Task	Input	Output	Notes
Translation	"Hello, how are you?"	"Bonjour, comment allez-vous?"	Different lengths!
Summarization	Long article (1000 words)	Short summary (50 words)	Compression
Speech-to-Text	5 seconds of audio	"Hello world"	Modality change
Text-to-Speech	"Hello world"	5 seconds of audio	Reverse direction
Code Generation	"Sort this list"	`list.sort()`	Natural → Code
Chatbot	"What's 2+2?"	"The answer is 4"	Q&A

> Google Translate, Siri, Alexa, ChatGPT — all use Seq2Seq!

Token-Level Classification (Tagging)

Sometimes we classify **each element** in the sequence:



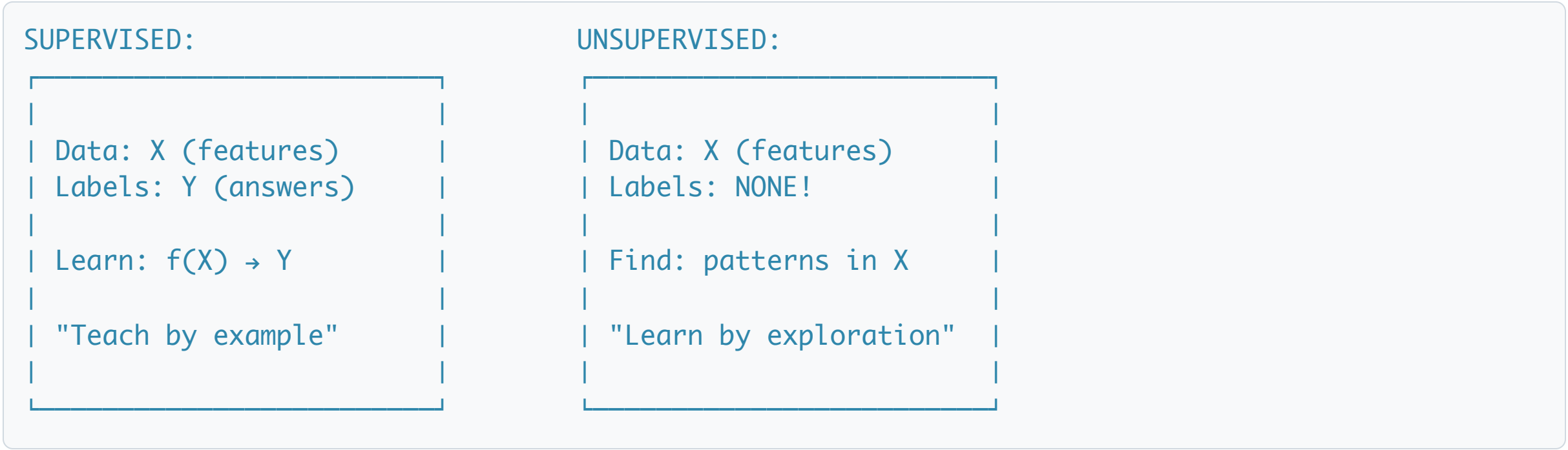
This is **Named Entity Recognition (NER)**.

> Think of it as "semantic segmentation for text" — every word gets a label!

Section 5: Unsupervised Learning

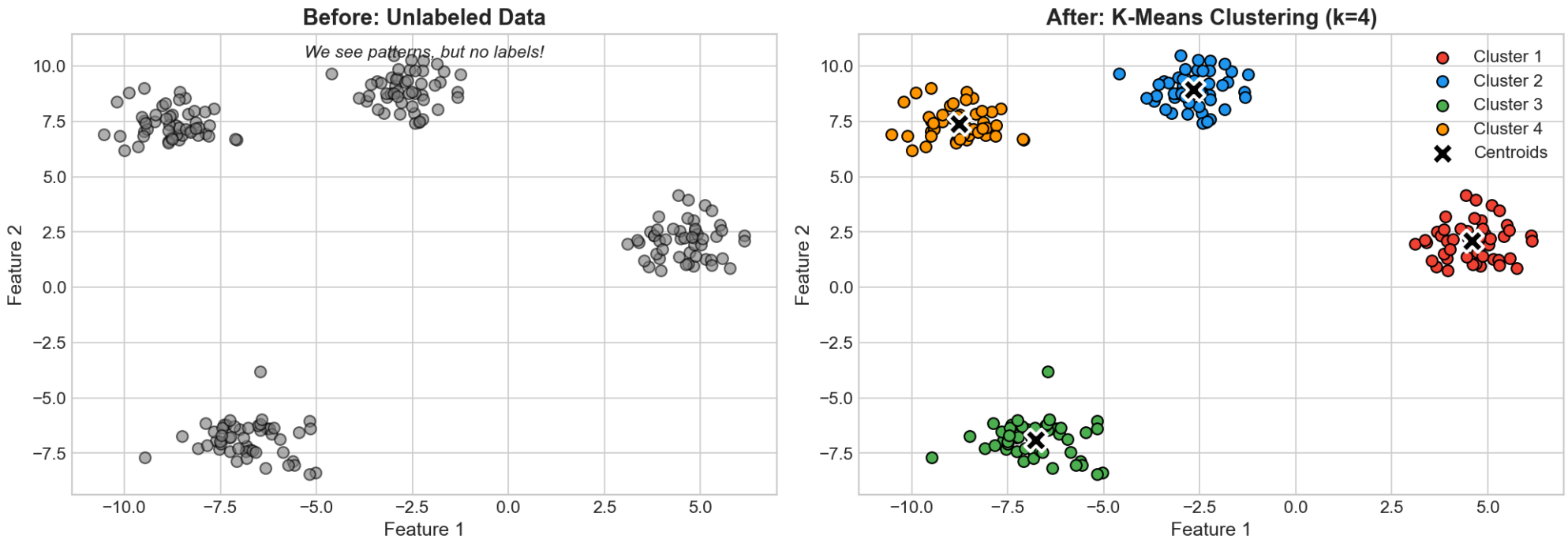
Finding Patterns Without Labels

The Unsupervised Setting



No one tells the model what to look for — it discovers structure on its own!

Clustering: Finding Natural Groups

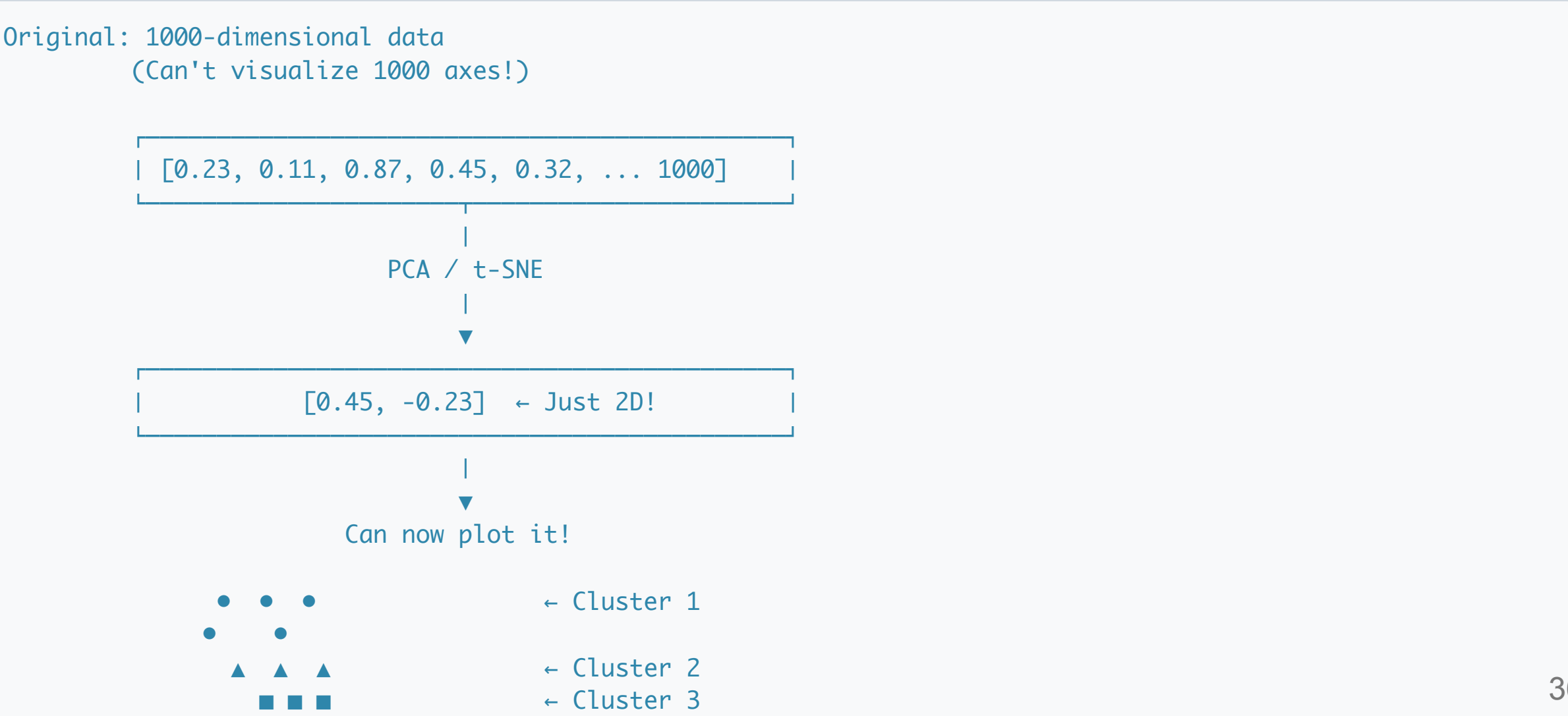


K-Means: No labels needed! The algorithm discovers natural groupings on its own.

Real applications: Customer segmentation, gene expression analysis, document clustering

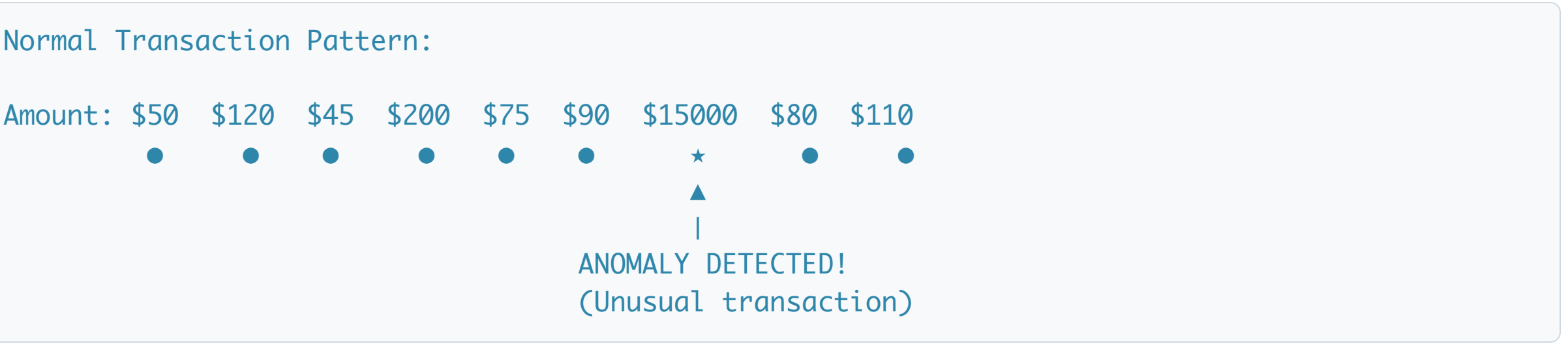
Dimensionality Reduction

Problem: High-dimensional data is hard to visualize and process.



Anomaly Detection

Find the weird ones.



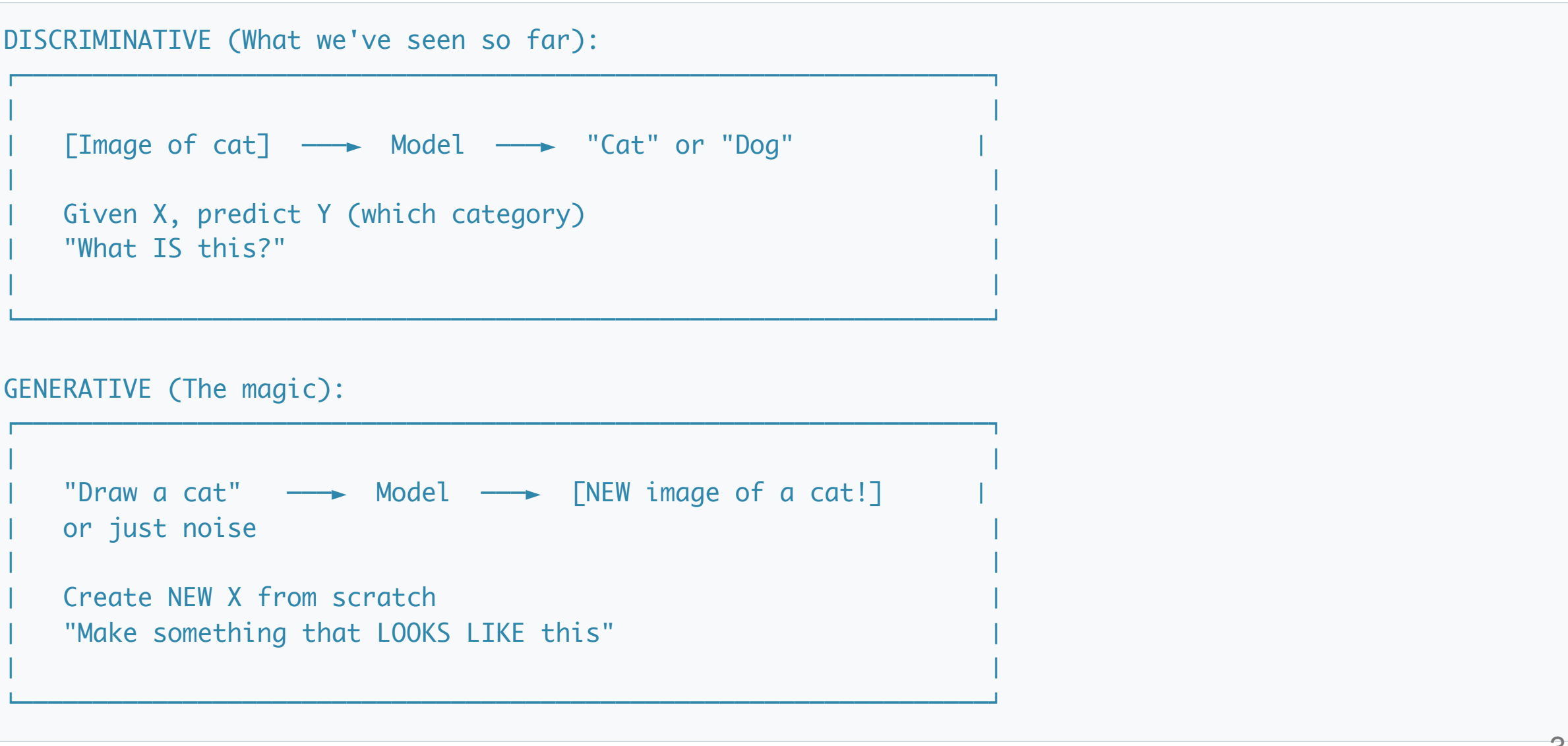
Applications:

- Credit card fraud detection
- Network intrusion detection
- Manufacturing defect detection
- Medical abnormality detection

Section 6: Generative Models

Creating New Data

Generative vs Discriminative



The Generative AI Revolution

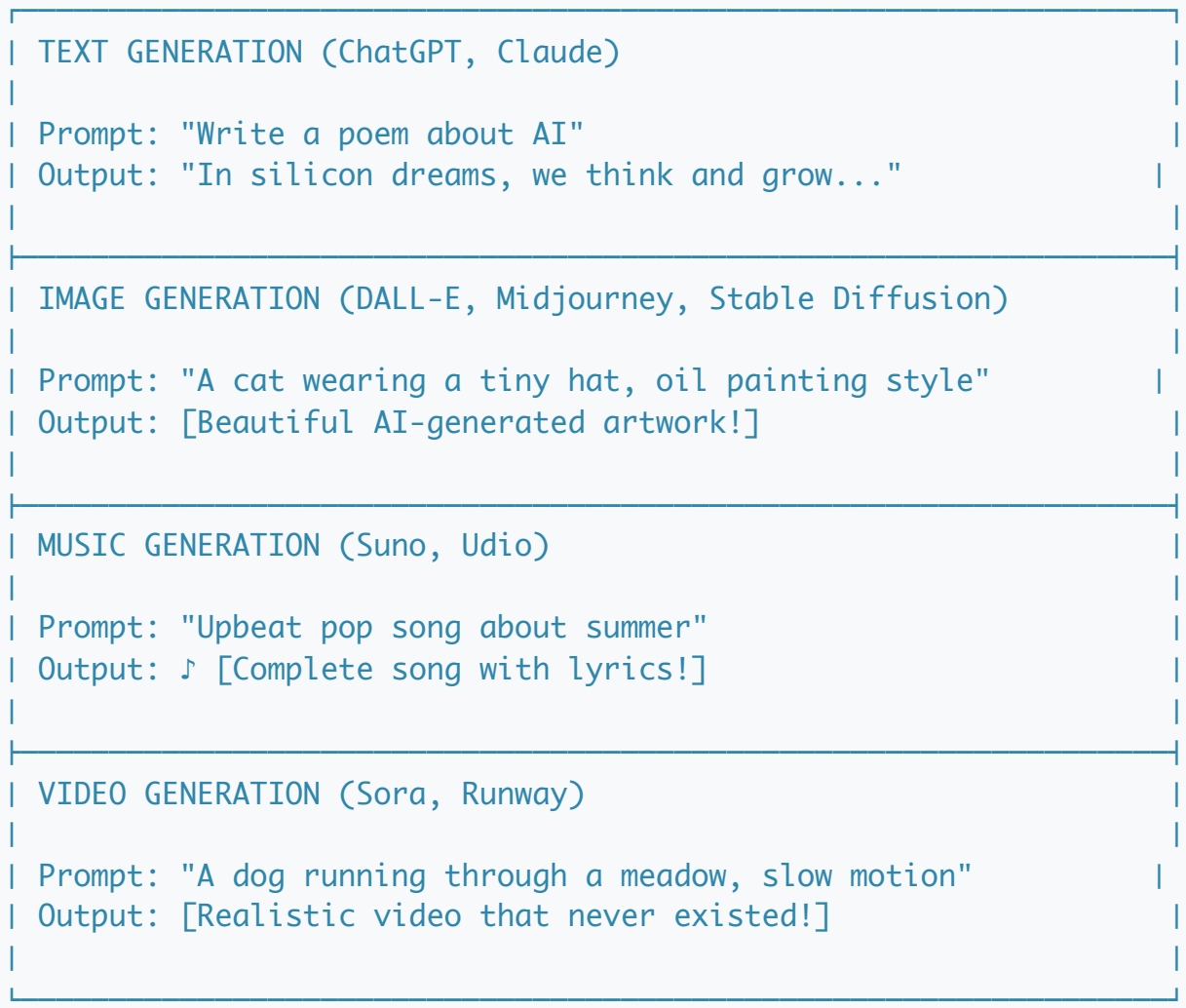
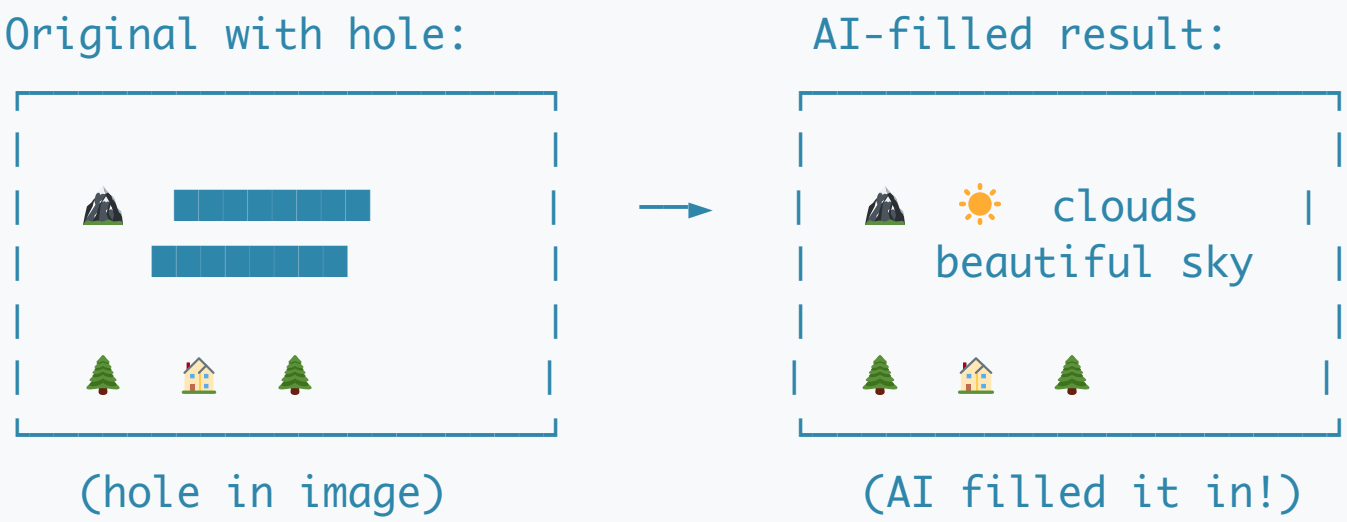


Image Inpainting: Fill in the Blanks



Applications:

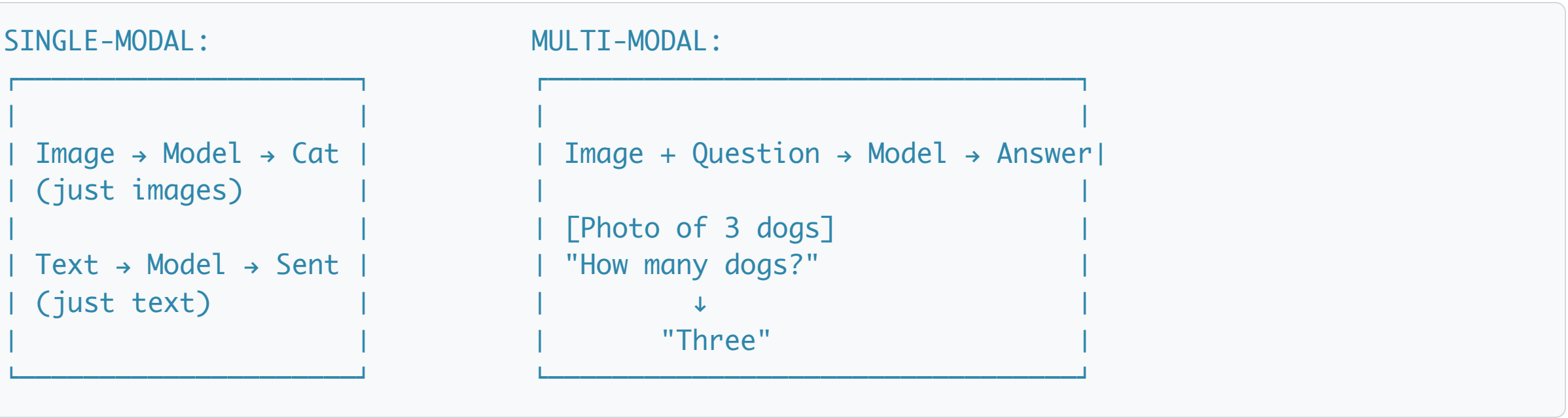
- Remove unwanted objects from photos
- Restore damaged/old photographs
- Extend images beyond their borders

Section 7: Multimodal & Complex Tasks

Combining Everything

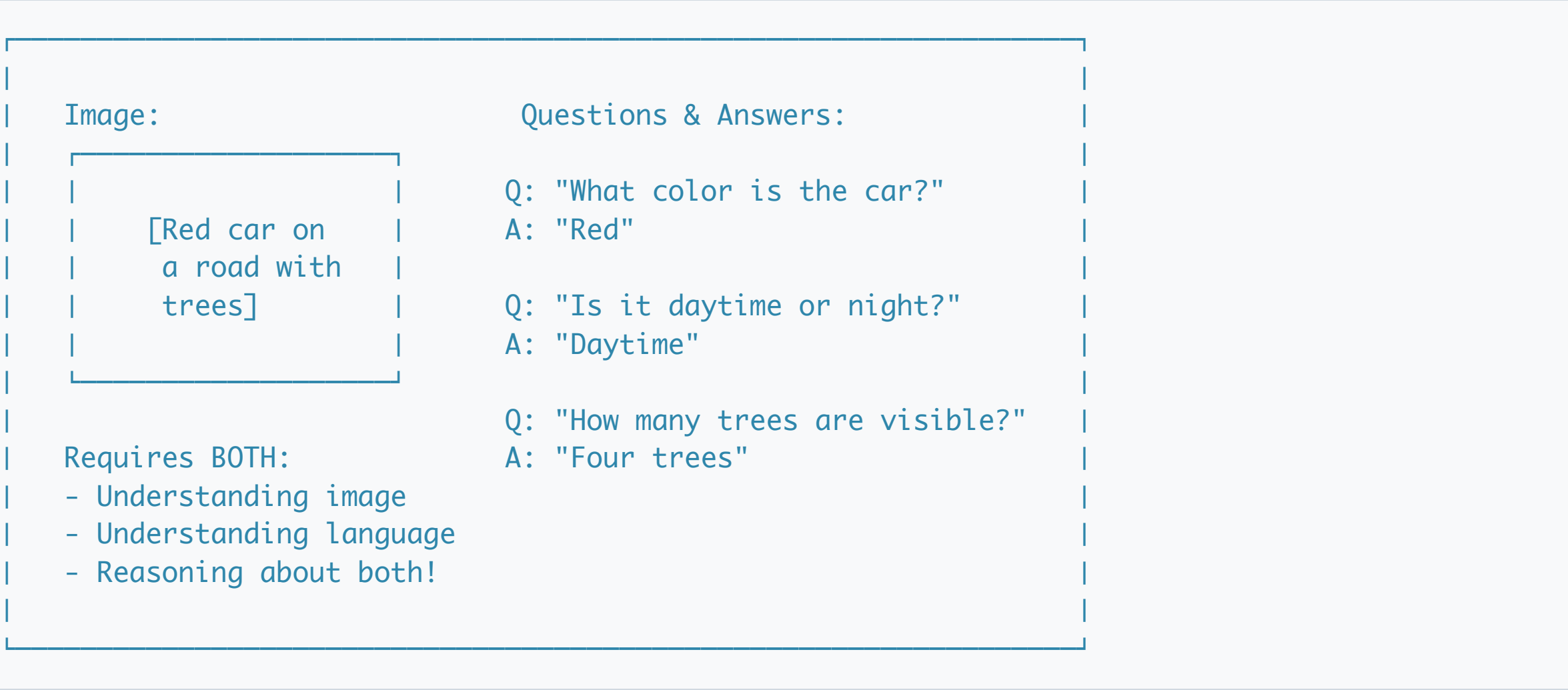
Multimodal = Multiple Modalities

Modalities: Text, Image, Audio, Video, etc.



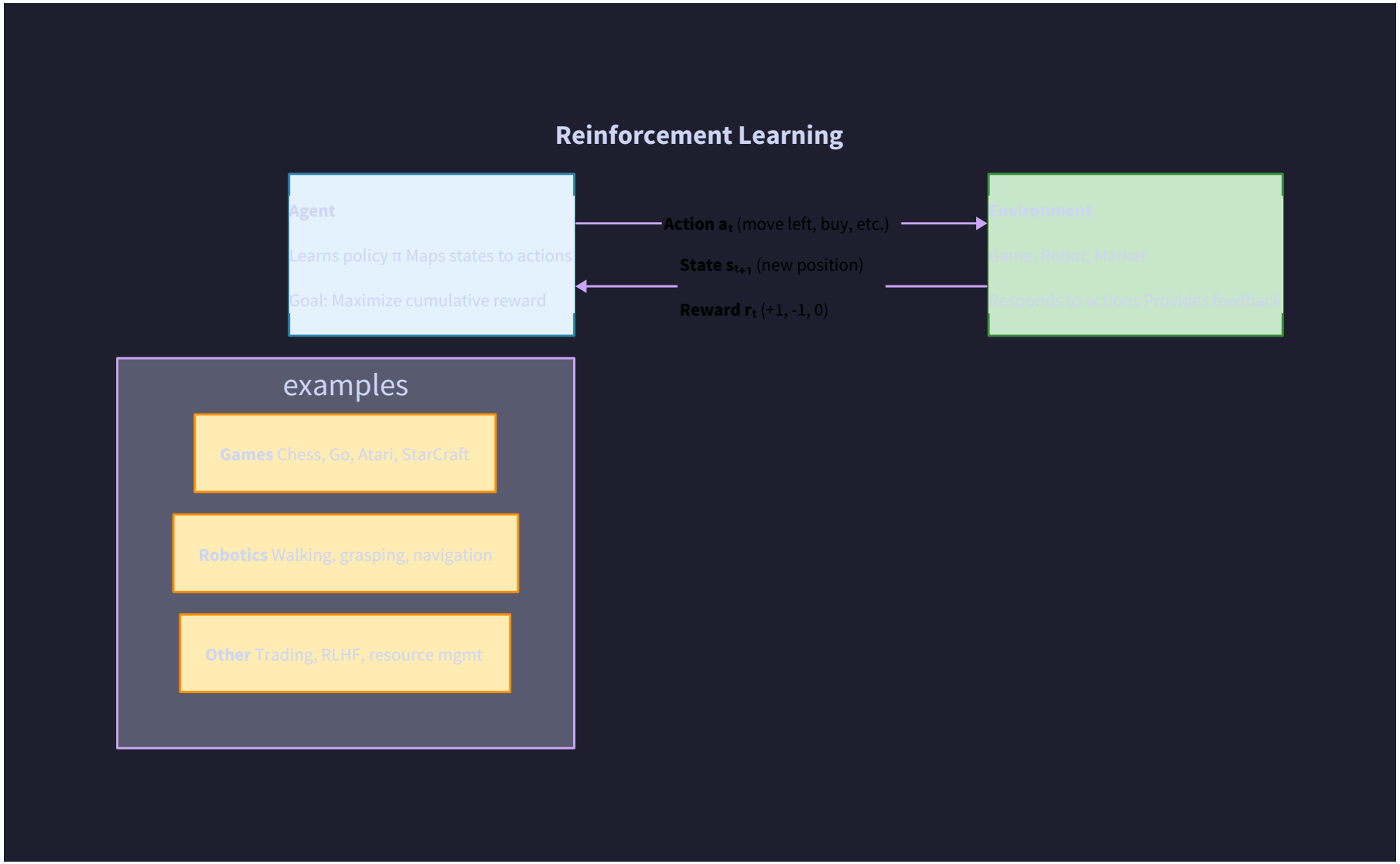
> Modern AI (GPT-4, Claude, Gemini) is multimodal — it can see AND read AND hear!

Visual Question Answering (VQA)



Reinforcement Learning

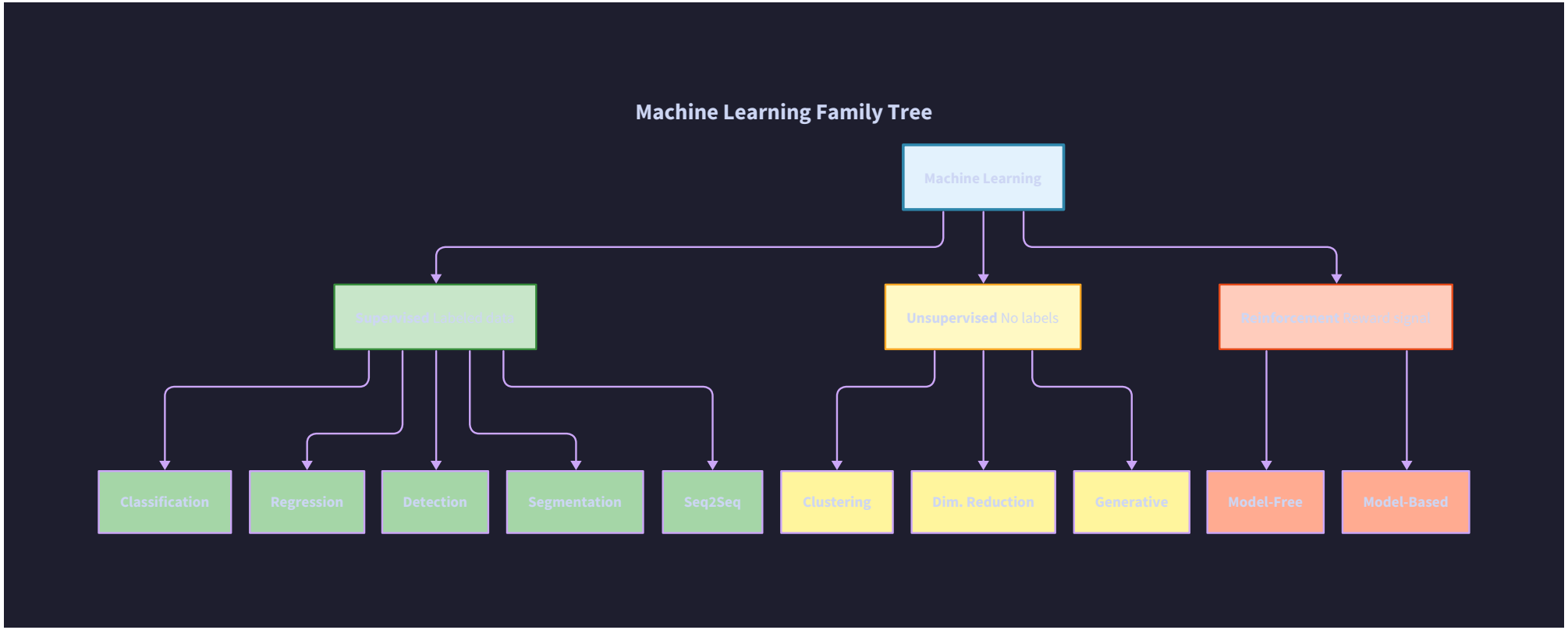
A different paradigm: **Learning through interaction**.



RL Examples

```
┌──────────────────────────────────────────────────────────────────┐
│ GAME PLAYING                                                     │
│ • AlphaGo: Beat world champion at Go                             │
│ • AlphaStar: Grandmaster level at StarCraft II                   │
│ • OpenAI Five: Beat pro teams at Dota 2                          │
│                                                                  │
├──────────────────────────────────────────────────────────────────┤
│ ROBOTICS                                                         │
│ • Boston Dynamics: Learning to walk, run, dance                  │
│ • Robot arms: Learning to pick up objects                        │
│ • Drones: Learning to navigate and avoid obstacles               │
│                                                                  │
├──────────────────────────────────────────────────────────────────┤
│ OTHER APPLICATIONS                                               │
│ • Data center cooling (Google reduced energy 40%)                │
│ • Chip design (designing better AI chips!)                       │
│ • Drug discovery (finding new molecules)                         │
│ • RLHF: Making ChatGPT helpful and safe!                         │
│                                                                  │
└──────────────────────────────────────────────────────────────────┘
```

Summary: The ML Family Tree

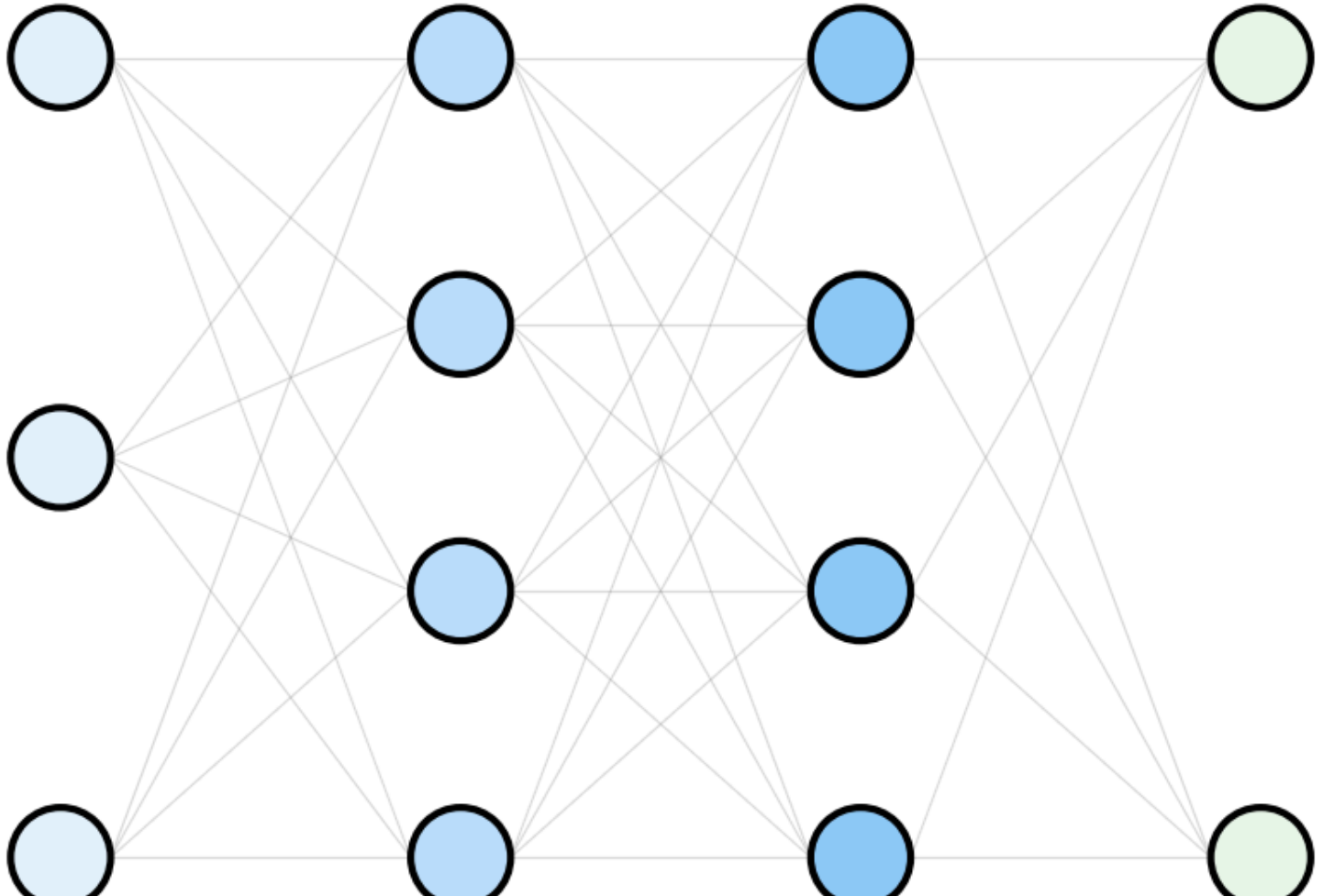


The Decision Flowchart

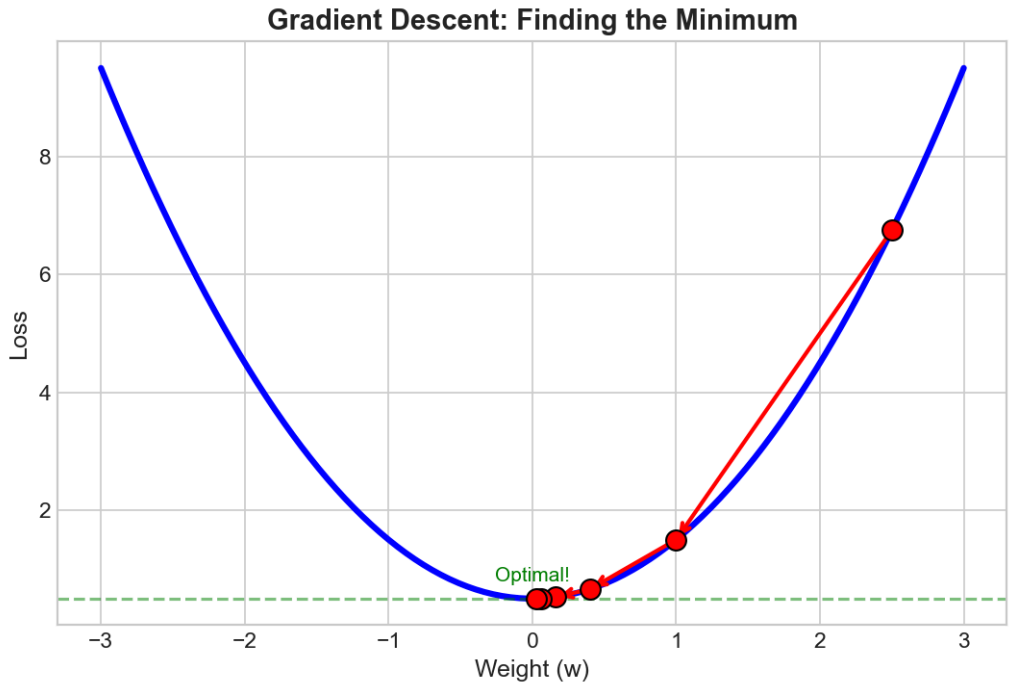
The Common Thread: Neural Networks

Neural Network: Universal Function Approximator

Each connection has a learnable weight



How Neural Networks Learn: Gradient Descent



Gradient Descent Update Rule:

$$w_{new} = w_{old} - \eta \cdot \nabla L$$

η = learning rate (step size)

∇L = gradient (direction of steepest ascent)

We go OPPOSITE to gradient to minimize loss!

Training = Finding the weights that minimize the loss function

Key Takeaways

1. **Classification** → Predict a category (discrete)
2. **Regression** → Predict a number (continuous)
3. **Detection** → Classification + Box Regression
4. **Segmentation** → Classification for every pixel
5. **Seq2Seq** → Sequence in, sequence out (translation, etc.)
6. **Unsupervised** → Find patterns without labels
7. **Generative** → Create new data
8. **Multimodal** → Combine text, images, audio, etc.
9. **RL** → Learn from rewards through interaction

> Understanding the output type tells you which family of techniques to use!

Coming Up: Deep Dives

Lecture 3: Language Models (Next Token Prediction → ChatGPT)

Lecture 4: Object Detection (YOLO and beyond)

Thank You!

"All models are wrong, but some are useful." — George Box

The key is matching the right model to the right task!

Questions?