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?

```
┌─────────────────────────────────────────────────────────────────┐
│                     YOUR DAILY LIFE                             │
├─────────────────────────────────────────────────────────────────┤
│                                                                 │
│   Morning:  Phone unlocks with your face                        │
│   Commute:  Google Maps predicts traffic                        │
│   Email:    Gmail filters spam, suggests replies                │
│   Music:    Spotify recommends songs you'll love                │
│   Shopping: Amazon shows "You might also like..."               │
│   Photos:   Google Photos groups by faces, finds "beach"        │
│   Evening:  Netflix suggests what to watch                      │
│   Chat:     You ask ChatGPT a question                          │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

Each of these is a different ML task!

The Big Insight

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

```
"What are you trying to PREDICT?"
```

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?"

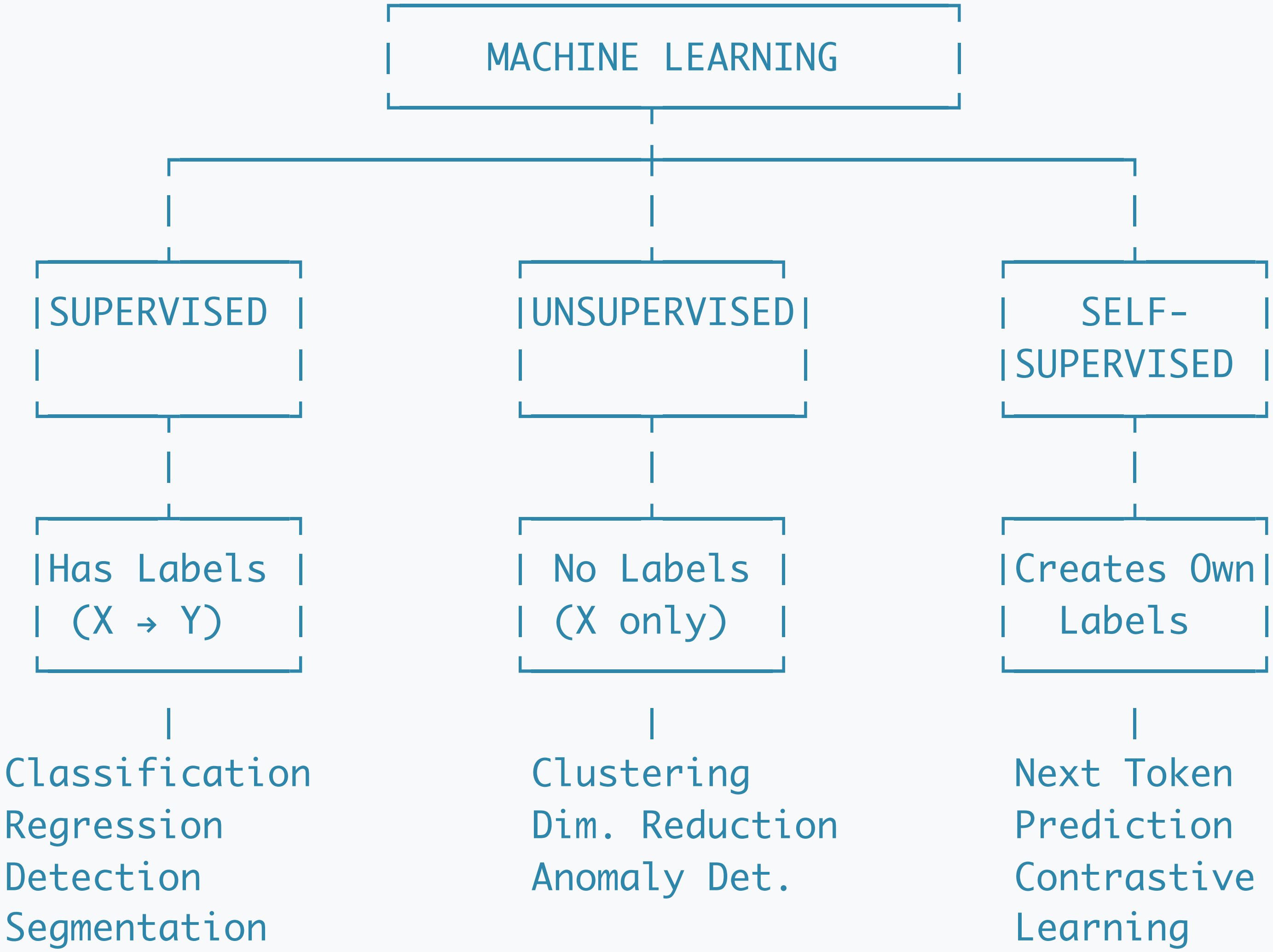
Creating Something New?

→ Generative

"Draw me a cat in space"

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

The Master Taxonomy



MACHINE LEARNING

SUPERVISED

UNSUPERVISED

SELF-SUPERVISED

Has Labels (X → Y)

No Labels (X only)

Creates Own Labels

Classification
Regression
Detection
Segmentation

Clustering
Dim. Reduction
Anomaly Det.

Next Token Prediction
Contrastive Learning

Section 1: Classification

"Which Bucket Does This Belong To?"

Classification: You Already Know This!

Think about how YOU classify things every day:

```
┌─────────────────────────────────────────────────────────────────┐
│                                                                   │
│   "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

center

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

Example: How Does Email Spam Detection Work?

```
Step 1: TRAINING (Learning from examples)
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│  Email: "Meeting at 3pm tomorrow"              Label: NOT SPAM │
│  Email: "You won $1,000,000! CLICK NOW!!!"     Label: SPAM     │
│  Email: "Your Amazon order has shipped"        Label: NOT SPAM │
│  Email: "Hot singles in your area"             Label: SPAM     │
│  ... (millions more examples)                                   │
│                                                                 │
│  Model learns: ALL CAPS, "won", "click", "$$$" → probably SPAM │
│                Normal sentences, known senders → probably OK    │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘

Step 2: INFERENCE (Using the model)
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│  New email: "CONGRATULATIONS! You're selected for a FREE gift!"│
│                                                                 │
│  Model thinks: ALL CAPS ✓, "FREE" ✓, excitement ✓              │
│  Prediction: SPAM (95% confident)                               │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

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

```
Input x ──► Neural Network ──► Softmax ──► Probabilities
                                              │
                                              ▼

                                    ┌─────────────────┐
                                    │ Cat:    0.85    │
                                    │ Dog:    0.10    │
                                    │ Bird:   0.05    │
                                    └─────────────────┘

                                              │
                                         Sum = 1.0
                                              │
                                              ▼
                                        Pick highest
                                              │
                                              ▼
                                           "Cat"
```

Softmax converts raw scores 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

```
┌────────────────────────────────────────────────────────────────┐
│                                                                │
│   "How old is this person?"           → 27.3 years             │
│                                                                │
│   "What's this house worth?"          → $425,000               │
│                                                                │
│   "How many units will sell?"         → 1,247 units            │
│                                                                │
│   "What temperature tomorrow?"        → 28.5°C                 │
│                                                                │
│   "How long until the bus arrives?"   → 7.2 minutes            │
│                                                                │
└────────────────────────────────────────────────────────────────┘
```

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

Regression: The Core Idea

center

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

Example: House Price Prediction

```
FEATURES (Input):                         PREDICTION (Output):

┌──────────────────────────────────┐
| Bedrooms:         3              |
| Bathrooms:        2              |
| Square Feet:      1,500          |     ────────►    $425,000
| Year Built:       2005           |
| Location Score:   8.5/10         |
| Has Pool:         No             |
└──────────────────────────────────┘


The model learns patterns like:
  - More bedrooms → higher price (usually)
  - Better location → higher price (definitely!)
  - Older house → lower price (sometimes)
  - Has pool → depends on the climate!

Then combines all these patterns into ONE number.
```

Regression is Actually Everywhere!

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

```
┌────────────────────────────────────────────────────────────────┐
│ BOUNDING BOX DETECTION                                          │
│                                                                 │
│   ┌─────────────────────────────────┐                          │
│   │   ┌──────────┐                  │                          │
│   │   │   DOG    │ ← This box needs │                          │
│   │   │          │   4 numbers:     │                          │
│   │   └──────────┘                  │                          │
│   └─────────────────────────────────┘                          │
│                                                                 │
│   x = 50 (left edge)     ← Regression!                         │
│   y = 30 (top edge)      ← Regression!                         │
│   w = 100 (width)        ← Regression!                         │
│   h = 80 (height)        ← Regression!                         │
│                                                                 │
│   DETECTION = Classification (what?) + Regression (where?)     │
│                                                                 │
└────────────────────────────────────────────────────────────────┘
```

Classification vs Regression: Side by Side

```
┌─────────────────────────────────────────────────────────────┐
│                       CLASSIFICATION                        │
│                                                             │
│   Input ───► Model ───► [0.1, 0.2, 0.7] ───► Class "C"     │
│                               ▲                             │
│                          Probabilities                      │
│                          must sum to 1                      │
│                                                             │
│   Loss Function: Cross-Entropy (compares probability dists) │
└─────────────────────────────────────────────────────────────┘

┌─────────────────────────────────────────────────────────────┐
│                         REGRESSION                          │
│                                                             │
│   Input ───► Model ───► 425000.00 ───► $425,000            │
│                               ▲                             │
│                        Any real number                      │
│                        (no constraints)                     │
│                                                             │
│   Loss Function: MSE / MAE (measures distance from true value) │
└─────────────────────────────────────────────────────────────┘
```

The Confusion: Age Prediction

Is predicting someone's age classification or regression?

```
OPTION A: Classification (Age Groups)

┌──────────────────────────────┐
│  ○ Child (0-12)              │
│  ○ Teenager (13-19)          │    Loses information!
│  ● Adult (20-59)             │    "25" and "55" are same class
│  ○ Senior (60+)              │
└──────────────────────────────┘

OPTION B: Regression (Exact Age)

┌──────────────────────────────┐
│                              │
│  Prediction: 27.3 years      │    More precise!
│                              │    But harder to predict exactly
└──────────────────────────────┘
```

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:

center

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

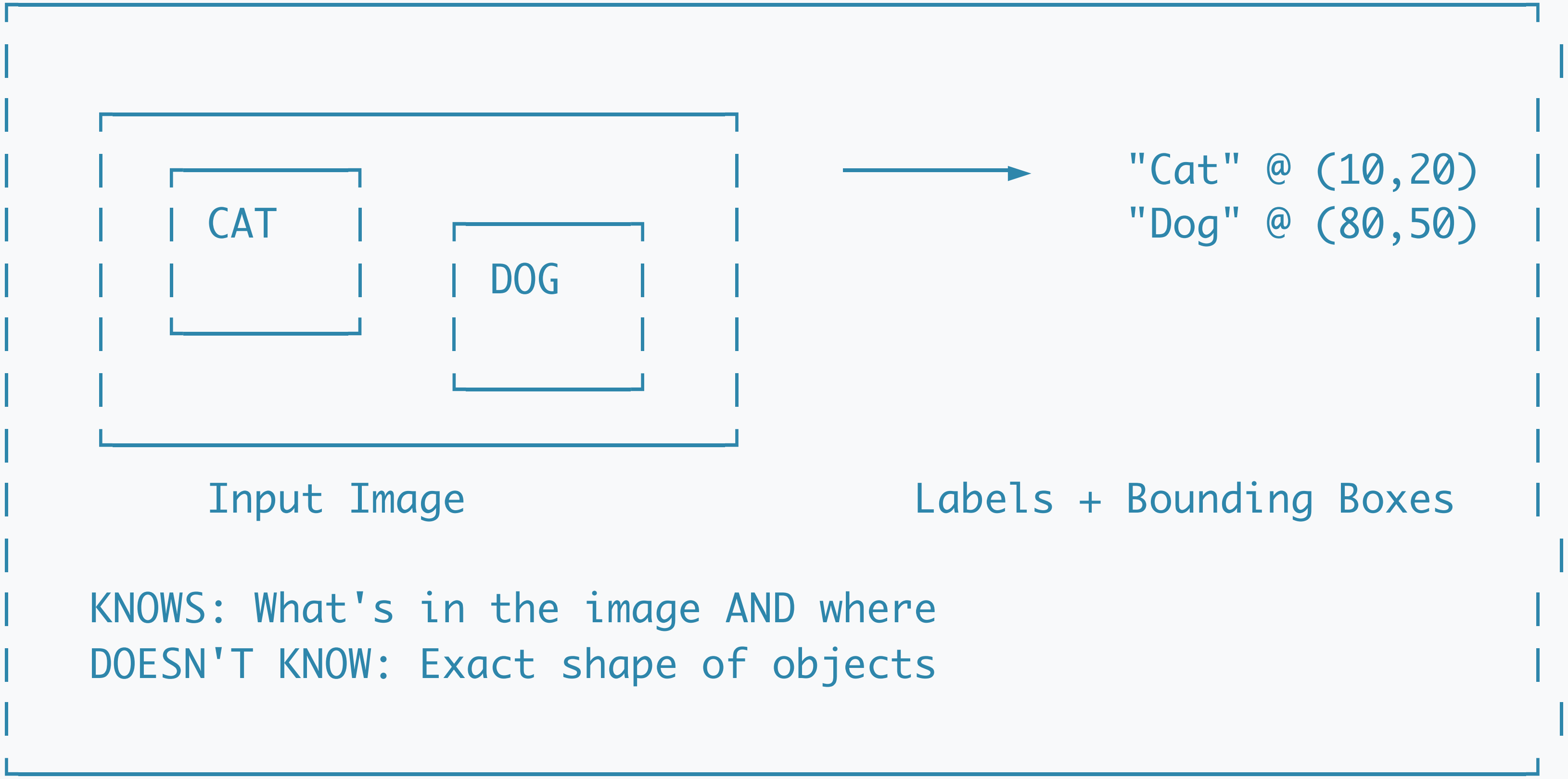
Level 1: Image Classification

```
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│   ┌───────────────────────────┐                                 │
│   │                           │                                 │
│   │      🐱                    │   ─────────▶    "Cat"          │
│   │   (somewhere in here)     │                                 │
│   │                           │                                 │
│   └───────────────────────────┘                                 │
│        Input Image                      Single Label            │
│                                                                 │
│   KNOWS: What's in the image                                    │
│   DOESN'T KNOW: Where it is                                     │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

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



KNOWS: What's in the image AND where
DOESN'T KNOW: Exact shape of objects

Detection = Classification (what) + Regression (where)

Level 3: Semantic Segmentation

```
┌────────────────────────────────────────────────────────────────┐
│                                                                 │
│   Input Image:                   Output Mask:                   │
│   ┌──────────────────┐          ┌──────────────────┐           │
│   │  [Sun] Sky       │          │ SSSSSSSSSSSSSSS  │  S = Sky  │
│   │                  │          │ SSSSSSSSSSSSSSS  │  T = Tree │
│   │  Tree Car Tree   │   -->    │ TTT CCCCC TTT    │  C = Car  │
│   │                  │          │ TTT CCCCC TTT    │  R = Road │
│   │  ═══════════════ │          │ RRRRRRRRRRRRRRR  │           │
│   └──────────────────┘          └──────────────────┘           │
│                                                                 │
│   EVERY pixel gets a class label!                               │
│   Perfect for self-driving cars, medical imaging              │
│                                                                 │
└────────────────────────────────────────────────────────────────┘
```

Level 4: Instance Segmentation

What if there are TWO cars?

```
SEMANTIC SEGMENTATION:               INSTANCE SEGMENTATION:

┌────────────────────┐               ┌────────────────────┐
│                    │               │                    │
│   CCCCC    CCCCC   │               │   11111    22222   │
│   CCCCC    CCCCC   │               │   11111    22222   │
│                    │               │                    │
└────────────────────┘               └────────────────────┘

   Both labeled "Car"                  Car #1 vs Car #2

Semantic: "These pixels are CAR"
Instance: "These pixels are CAR #1, those are CAR #2"
```

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

Real-World Vision Hierarchy Example

```
┌─────────────────────────────────────────────────────────────────┐
│                    AUTONOMOUS DRIVING                            │
├─────────────────────────────────────────────────────────────────┤
│                                                                  │
│   Classification:  "There are cars and people in this scene"    │
│                    (Not enough! Where are they?)                │
│                                                                  │
│   Detection:       "Car at (100,200), Person at (300,150)"      │
│                    (Better! But how close to lane?)              │
│                                                                  │
│   Segmentation:    "The drivable road area is these pixels"     │
│                    (Great! Now I know where to drive)            │
│                                                                  │
│   Instance Seg:    "This is Car #1, that is Car #2, tracking..."│
│                    (Perfect! I can predict each car's movement) │
│                                                                  │
└─────────────────────────────────────────────────────────────────┘
```

Section 4: Sequence Tasks

When Order Matters

Why Sequences Are Special

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

```
┌────────────────────────────────────────────────────────────────┐
|                                                                  |
|   TEXT:    "I love you"   vs    "You love I"                   |
|             (Sweet)                (Grammatically wrong!)        |
|                                                                  |
|   DNA:     ATCGATCG  vs  GATCATCG                              |
|             (Different gene!)                                    |
|                                                                  |
|   AUDIO:  ♪ Do-Re-Mi  vs  ♪ Mi-Re-Do                           |
|             (Different melody!)                                  |
|                                                                  |
|   VIDEO:  Frame1→Frame2→Frame3  vs  Frame3→Frame2→Frame1       |
|             (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!)

center

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:

```
┌────────────────────────────────────────────────────────────────┐
|                                                                 |
|   Input:    "Sundar  Pichai   visited  New    York   yesterday" |
|              |       |        |        |      |      |         |
|              ▼       ▼        ▼        ▼      ▼      ▼         |
|   Output:   PER     PER       O       LOC    LOC     O         |
|                                                                 |
|   PER = Person Name                                             |
|   LOC = Location                                                |
|   O   = Other (not an entity)                                   |
|                                                                 |
└────────────────────────────────────────────────────────────────┘
```

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

```
SUPERVISED:                        UNSUPERVISED:
┌─────────────────────────┐        ┌─────────────────────────┐
│                         │        │                         │
│ Data: X (features)      │        │ Data: X (features)      │
│ Labels: Y (answers)     │        │ Labels: NONE!           │
│                         │        │                         │
│ Learn: f(X) → Y         │        │ Find: patterns in X     │
│                         │        │                         │
│ "Teach by example"      │        │ "Learn by exploration"  │
│                         │        │                         │
└─────────────────────────┘        └─────────────────────────┘
```

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

Clustering: Finding Natural Groups

```
BEFORE (Unlabeled data):           AFTER (Discovered clusters):

    ●    ■                              ○    □
  ●   ●    ■ ■                        ○   ○    □ □
    ●        ■                          ○        □

        ▲   ▲                               △   △
    ▲          ▲                        △          △
      ▲    ▲                              △    △

No one told the algorithm these     The algorithm discovered
are 3 groups – it figured it out!   3 natural groupings.
```

Real applications:

- Customer segmentation: VIPs vs Bargain hunters vs Occasional buyers
- Gene expression: Which genes behave similarly?
- Document clustering: Group news articles by topic

Dimensionality Reduction

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

```
Original: 1000-dimensional data
         (Can't visualize 1000 axes!)

          ┌──────────────────────────────────────────────┐
          │ [0.23, 0.11, 0.87, 0.45, 0.32, ... 1000]     │
          └──────────────────────┬───────────────────────┘
                                 │
                            PCA / t-SNE
                                 │
                                 ▼
          ┌──────────────────────────────────────────────┐
          │          [0.45, -0.23]  ← Just 2D!           │
          └──────────────────────────────────────────────┘
                                 │
                                 ▼
                       Can now plot it!

              ●  ●  ●                ← Cluster 1
             ●    ●

               ▲  ▲  ▲               ← Cluster 2
                 ■ ■ ■               ← Cluster 3
```

Anomaly Detection

Find the weird ones.

```
Normal Transaction Pattern:

Amount: $50  $120  $45  $200  $75  $90  $15000  $80  $110
         ●     ●    ●     ●    ●    ●      ★     ●     ●
                                           ▲
                                           |
                                    ANOMALY DETECTED!
                                    (Unusual transaction)
```

Applications:

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

Section 6: Generative Models

Creating New Data

Generative vs Discriminative

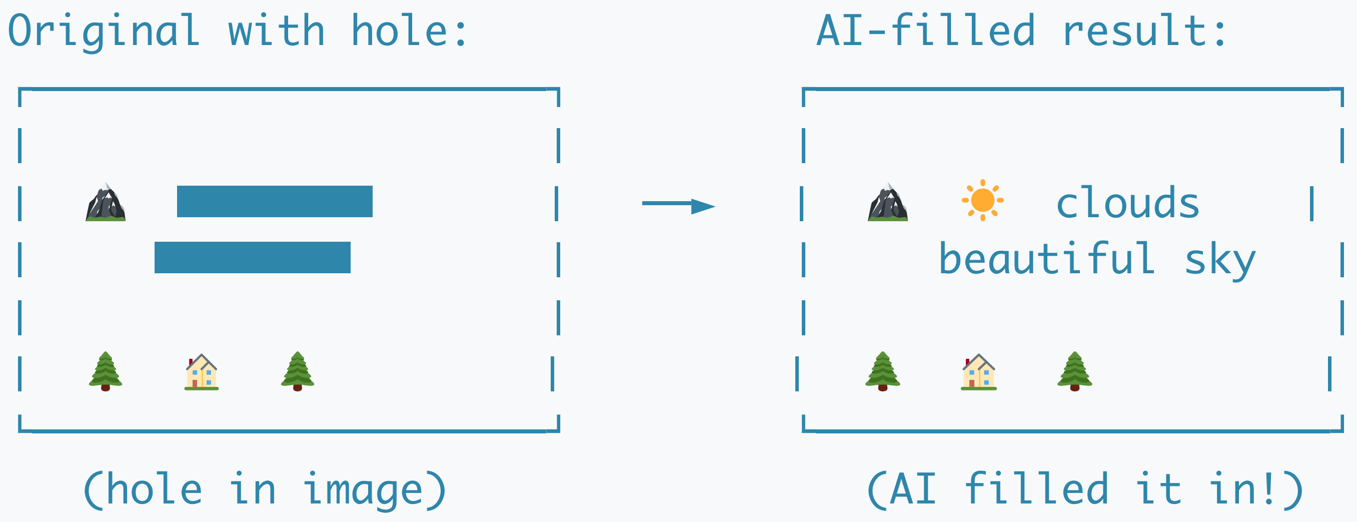
```
DISCRIMINATIVE (What we've seen so far):
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│   [Image of cat] ───► Model ───► "Cat" or "Dog"                │
│                                                                 │
│   Given X, predict Y (which category)                           │
│   "What IS this?"                                               │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘

GENERATIVE (The magic):
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│   "Draw a cat"  ───► Model ───► [NEW image of a cat!]          │
│   or just noise                                                 │
│                                                                 │
│   Create NEW X from scratch                                     │
│   "Make something that LOOKS LIKE this"                         │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

The Generative AI Revolution

```
┌─────────────────────────────────────────────────────────────────┐
│ TEXT GENERATION (ChatGPT, Claude)                               │
│                                                                 │
│ Prompt: "Write a poem about AI"                                 │
│ Output: "In silicon dreams, we think and grow..."              │
│                                                                 │
├─────────────────────────────────────────────────────────────────┤
│ IMAGE GENERATION (DALL-E, Midjourney, Stable Diffusion)         │
│                                                                 │
│ Prompt: "A cat wearing a tiny hat, oil painting style"         │
│ Output: [Beautiful AI-generated artwork!]                       │
│                                                                 │
├─────────────────────────────────────────────────────────────────┤
│ MUSIC GENERATION (Suno, Udio)                                   │
│                                                                 │
│ Prompt: "Upbeat pop song about summer"                          │
│ Output: ♪ [Complete song with lyrics!]                          │
│                                                                 │
├─────────────────────────────────────────────────────────────────┤
│ VIDEO GENERATION (Sora, Runway)                                 │
│                                                                 │
│ Prompt: "A dog running through a meadow, slow motion"          │
│ Output: [Realistic video that never existed!]                   │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

Image Inpainting: Fill in the Blanks



Original with hole:

AI-filled result:

(hole in image)

(AI filled it in!)

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.

```
SINGLE-MODAL:                    MULTI-MODAL:
┌──────────────────────┐         ┌──────────────────────────────────┐
│                      │         │                                  │
│ Image → Model → Cat  │         │ Image + Question → Model → Answer│
│ (just images)        │         │                                  │
│                      │         │ [Photo of 3 dogs]                │
│ Text → Model → Sent  │         │ "How many dogs?"                 │
│ (just text)          │         │          ↓                       │
│                      │         │       "Three"                    │
└──────────────────────┘         └──────────────────────────────────┘
```

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

Visual Question Answering (VQA)

```
┌─────────────────────────────────────────────────────────────────┐
│                                                                 │
│   Image:                      Questions & Answers:              │
│   ┌───────────────────┐                                         │
│   │                   │      Q: "What color is the car?"        │
│   │    [Red car on    │      A: "Red"                           │
│   │     a road with   │                                         │
│   │     trees]        │      Q: "Is it daytime or night?"       │
│   │                   │      A: "Daytime"                       │
│   └───────────────────┘                                         │
│                              Q: "How many trees are visible?"   │
│   Requires BOTH:             A: "Four trees"                    │
│   - Understanding image                                         │
│   - Understanding language                                      │
│   - Reasoning about both!                                       │
│                                                                 │
└─────────────────────────────────────────────────────────────────┘
```

Reinforcement Learning

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

center

Goal: Maximize total reward over time through trial and error.

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

center

The Decision Flowchart

```
START: What do you want to predict?
            │
    ┌───────┴───────┐
    │               │
Category?       Number?
    │               │
    ▼               ▼
Classification  Regression
                    │
    ┌───────────────┤
    │               │
Sequence?       Location?
    │               │
    ▼               ▼
Seq2Seq       Detection
              (Class + Reg)
                    │
            Exact pixels?
                    │
                    ▼
              Segmentation

No labels available?  →  Unsupervised (Clustering, etc.)
Want to create new data?  →  Generative
Learning from trial/error?  →  Reinforcement Learning
```

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!

What's Next?

In the **ML Tasks Zoo** lecture, we'll dive deeper into:

- 40+ specific tasks across all domains
- Computer Vision tasks in detail
- NLP tasks explained
- Audio processing
- And much more!

Thank You!

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

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

Questions?