Machine Learning Tasks & Taxonomy

From Classification to Deep Learning

Nipun Batra | IIT Gandhinagar

ML is Everywhere in Your Life

Time	What You Do	ML Behind It
Morning	Phone unlocks with your face	Face Recognition
Commute	Google Maps predicts traffic	Time Series Prediction
Email	Gmail filters spam	Text Classification
Music	Spotify recommends songs	Recommendation
Photos	Google Photos groups by faces	Clustering
Evening	Netflix suggests what to watch	Collaborative Filtering
Chat	You ask ChatGPT a question	Language Models

The Big Question

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

What are you trying to PREDICT?

What Are You Predicting?

If you're predicting...	It's called...	Example
A **category**	Classification	"Is this spam?"
A **number**	Regression	"What's the price?"
A **sequence**	Seq2Seq	"Translate to French"
Something new	Generation	"Draw a cat"

But There's Another Question...

Do you have labeled examples?

This determines your **learning paradigm**.

Part 1: Learning Paradigms

Supervised, Unsupervised, & Reinforcement

The Three Learning Paradigms

Supervised

Learn from labeled examples

- X: Images
- Y: Labels

Unsupervised

Find patterns in data

- X: Data only
- Y: None

Reinforcement

Learn from rewards

- Actions → Rewards
- Trial & error

Paradigm 1: Supervised Learning

"Learning with a teacher"

You have:

- **Input data** (X): features describing each example
- **Labels** (Y): the correct answer for each example

The model learns to map X → Y

Supervised Learning: The Setup

Training Data:

Features (X)	Label (Y)
Email text: "You won $1000..."	Spam
Email text: "Meeting at 3pm"	Not Spam
Email text: "Buy now! Sale!"	Spam
Email text: "Your order shipped"	Not Spam
...	...

Supervised Learning: The Goal

Learn a function **f** such that:

$$\hat{y} = f(x)$$

where $\hat{y}$ is close to the true label y

Supervised Learning: Examples

Task	Input (X)	Output (Y)
Spam Detection	Email text	Spam / Not Spam
House Pricing	Size, location, rooms	Price in $
Medical Diagnosis	Symptoms, tests	Disease / Healthy
Image Classification	Pixel values	Cat / Dog / Bird
Stock Prediction	Historical prices	Future price

Paradigm 2: Unsupervised Learning

"Learning without a teacher"

You have:

- **Input data** (X): features describing each example
- **No labels!**

The model finds **patterns** and **structure** in the data

Unsupervised Learning: The Setup

Training Data:

Features (X)	Label (Y)
Customer: age=25, spent=$500	?
Customer: age=45, spent=$2000	?
Customer: age=22, spent=$300	?
Customer: age=50, spent=$1800	?
...	?

No one tells you the groups - you discover them!

Unsupervised Learning: Examples

Task	Input (X)	What it Discovers
Customer Segmentation	Purchase history	Groups of similar customers
Topic Modeling	Documents	Topics in text corpus
Anomaly Detection	Transactions	Unusual/fraudulent behavior
Dimensionality Reduction	High-dim data	Lower-dim representation

Paradigm 3: Reinforcement Learning

"Learning from trial and error"

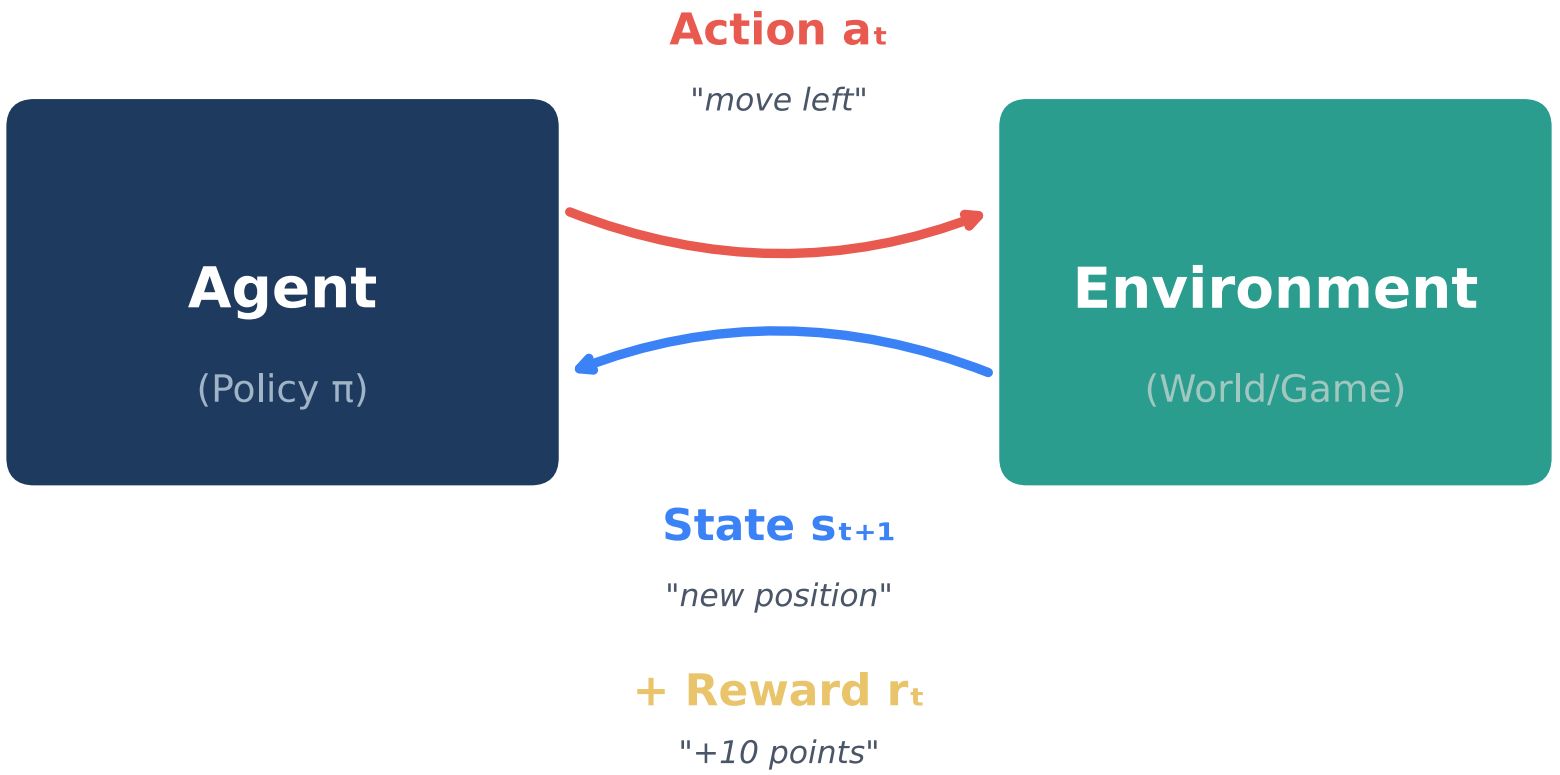
You have:

- An **agent** that takes actions
- An **environment** that responds
- **Rewards** that tell if actions were good

The agent learns to maximize cumulative reward

Reinforcement Learning: The Loop

Goal: Maximize cumulative reward Σr_t



Reinforcement Learning: Key Concepts

Concept	Meaning	Example (Game)
Agent	The learner	Game-playing AI
Environment	The world	The game itself
State	Current situation	Game screen
Action	What agent does	Move left, jump
Reward	Feedback signal	+1 for coin, -1 for death

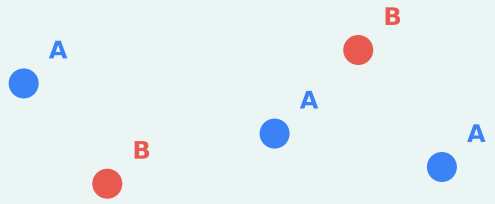
Reinforcement Learning: Examples

Domain	Agent	Reward
Games	AlphaGo	Win = +1, Lose = -1
Robotics	Robot arm	Task completed = +1
Finance	Trading bot	Profit = reward
RLHF	ChatGPT	Human preference = reward

Supervised vs Unsupervised

Supervised

"Teacher shows correct answers"



Unsupervised

"Find patterns on your own"

Summary: Learning Paradigms

Paradigm	Has Labels?	Goal
Supervised	✓ Yes	Predict labels for new data
Unsupervised	✗ No	Find patterns/structure
Reinforcement	Rewards	Maximize cumulative reward

Part 2: Supervised Learning

Classification & Regression

Two Types of Supervised Learning

It depends on what you're predicting:

If Y is...	Task Type	Example
Discrete (categories)	Classification	Cat or Dog?
Continuous (numbers)	Regression	Price = $350,000

Classification: The Idea

Goal: Assign input to one of several categories

```
Input: [Image of animal]
       ↓
   Classifier
       ↓
Output: "Cat" (from set: {Cat, Dog, Bird, Fish})
```

Binary Classification

Only **two** possible outcomes:

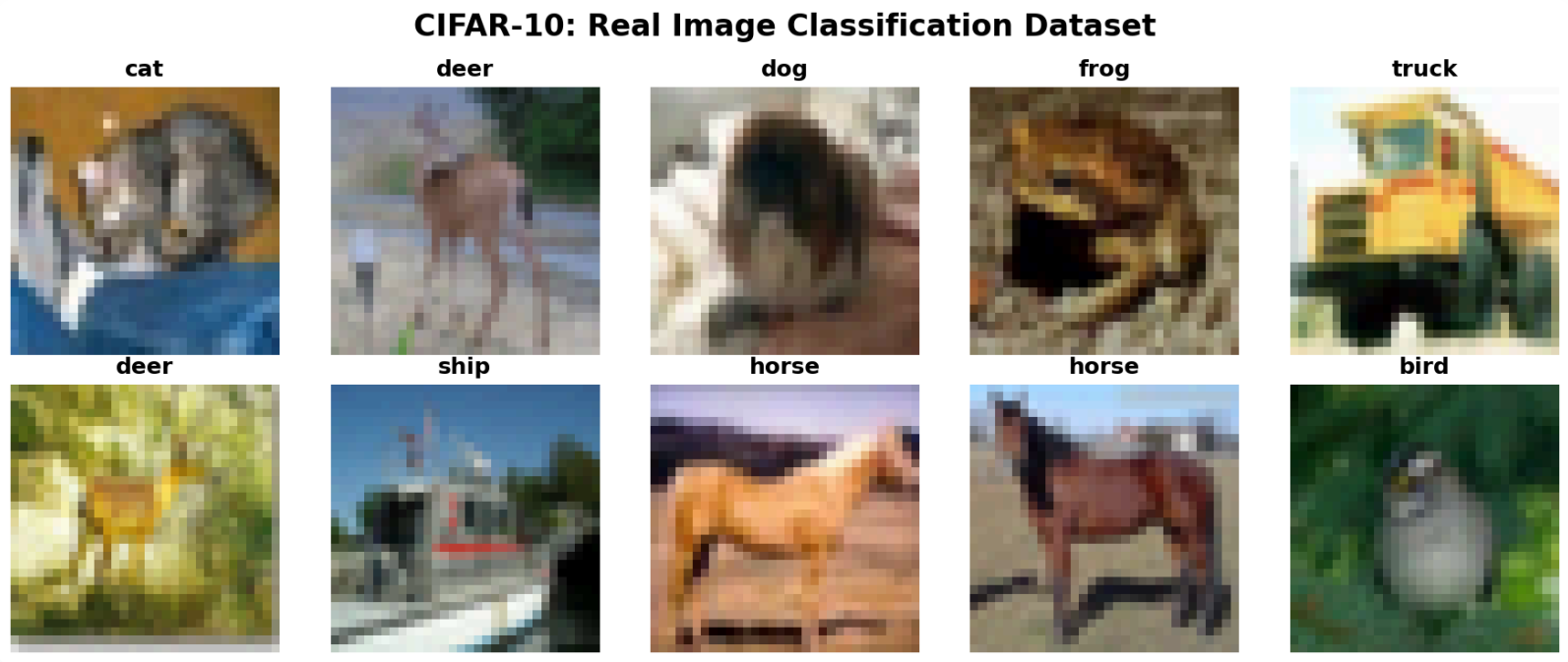
Example	Class 0	Class 1
Email filtering	Not Spam	Spam
Medical test	Healthy	Disease
Fraud detection	Legitimate	Fraud
Loan approval	Reject	Approve

Multi-class Classification

More than two possible outcomes:

Example	Classes
Digit recognition	0, 1, 2, ..., 9 (10 classes)
Animal classification	Cat, Dog, Bird, ...
Emotion detection	Happy, Sad, Angry, ...
ImageNet	1000 different objects

Classification in Pictures



Each image gets exactly **one label** from the set of possibilities.

Regression: The Idea

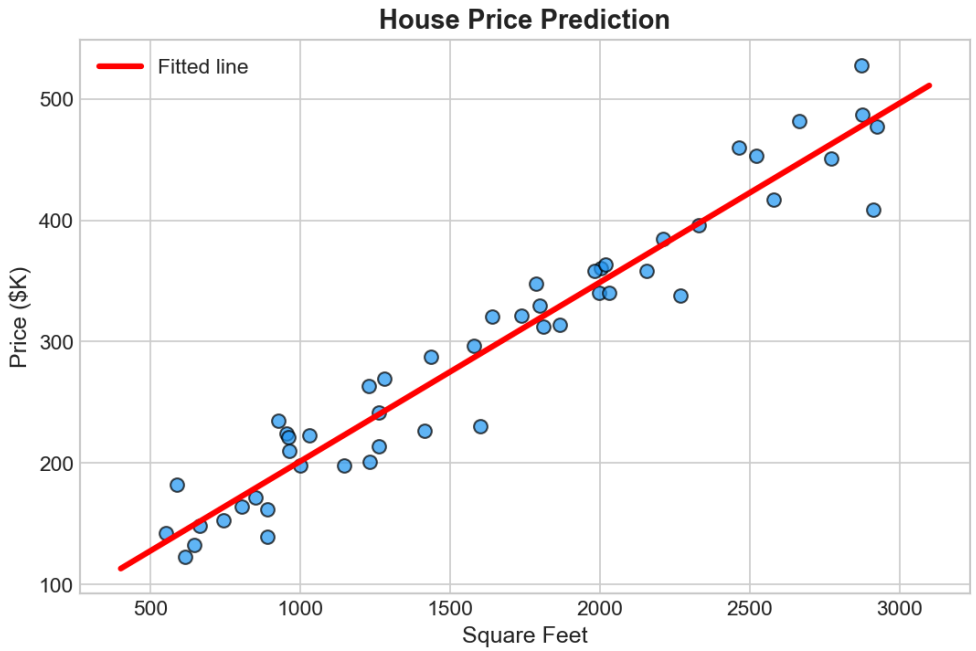
Goal: Predict a continuous numerical value

```
Input: [House features: 1500 sqft, 3 beds, ...]
       ↓
   Regressor
       ↓
Output: $425,000 (any number!)
```

Regression Examples

Input (X)	Output (Y)
House features	Price ($)
Person's photo	Age (years)
Stock history	Tomorrow's price
Weather data	Temperature (°C)
Car specs	Fuel efficiency (mpg)

Regression in Pictures



Linear Regression Formula:

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

Fitted: price = 54, 241 + 147 * sqft

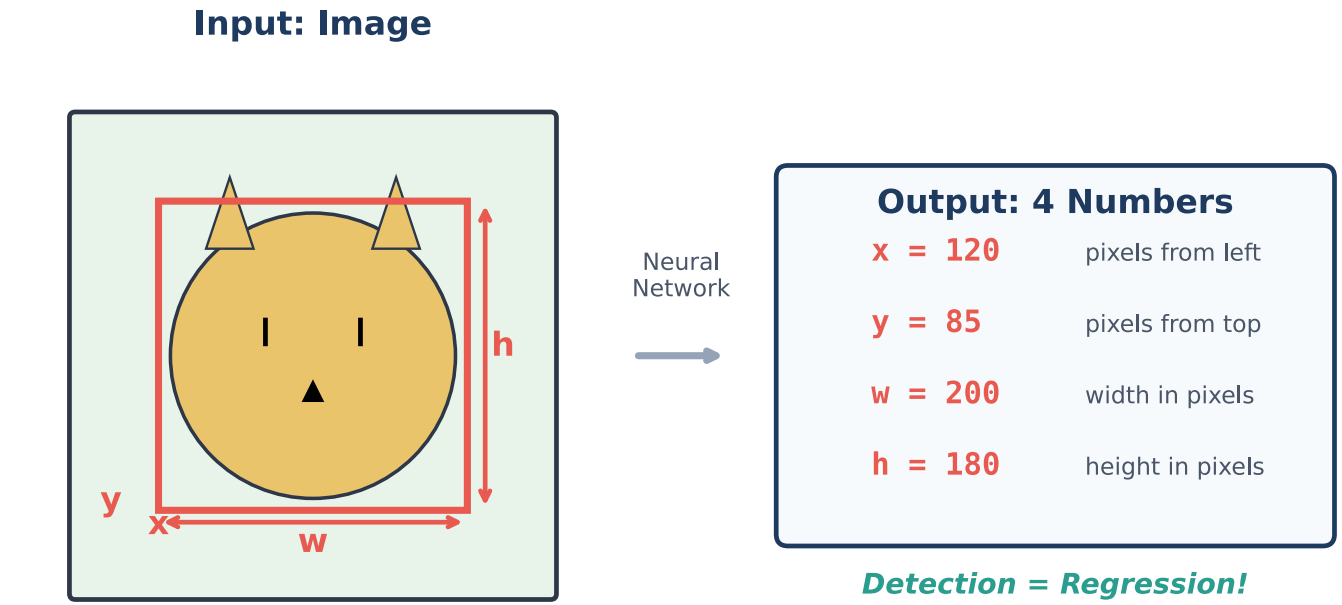
Each extra sqft adds ~$150 to price!

Key Difference: Classification vs Regression

Aspect	Classification	Regression
Output type	Category (discrete)	Number (continuous)
Example output	"Cat"	27.5
Loss function	Cross-entropy	Mean Squared Error
Metric	Accuracy	R^2, MAE, RMSE

A Hidden Surprise: Bounding Boxes

Object detection predicts **where** objects are...



Detection = Classification + Regression!

Part 3: Unsupervised Learning

Clustering & Dimensionality Reduction

When You Don't Have Labels

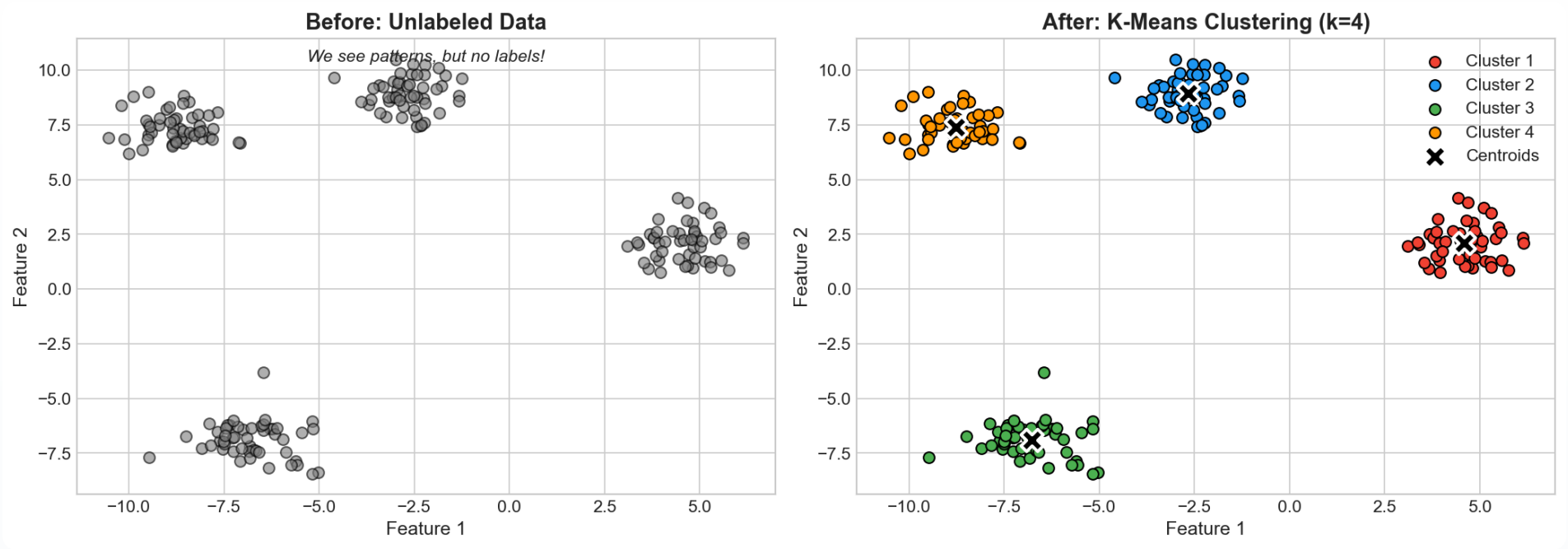
Sometimes you just have data, no labels:

- Customer transactions (who are similar customers?)
- News articles (what topics are discussed?)
- Sensor readings (what's normal vs abnormal?)

Clustering: Find Natural Groups

Goal: Group similar data points together

K-Means Clustering



Clustering Applications

Application	Data	Clusters Found
Marketing	Customer purchases	Customer segments
Biology	Gene expression	Cell types
Image	Pixels	Regions/segments
Document	News articles	Topics

Dimensionality Reduction

Problem: Data has too many features to visualize

Solution: Compress to 2-3 dimensions while preserving structure

```
Original: 1000 features → Can't visualize
                ↓ PCA / t-SNE
Reduced: 2 features → Can plot!
```

Why Reduce Dimensions?

1. **Visualization** - See patterns in high-dim data
2. **Speed** - Faster training with fewer features
3. **Noise removal** - Keep important dimensions
4. **Storage** - Less memory needed

Anomaly Detection

Goal: Find the "odd one out"

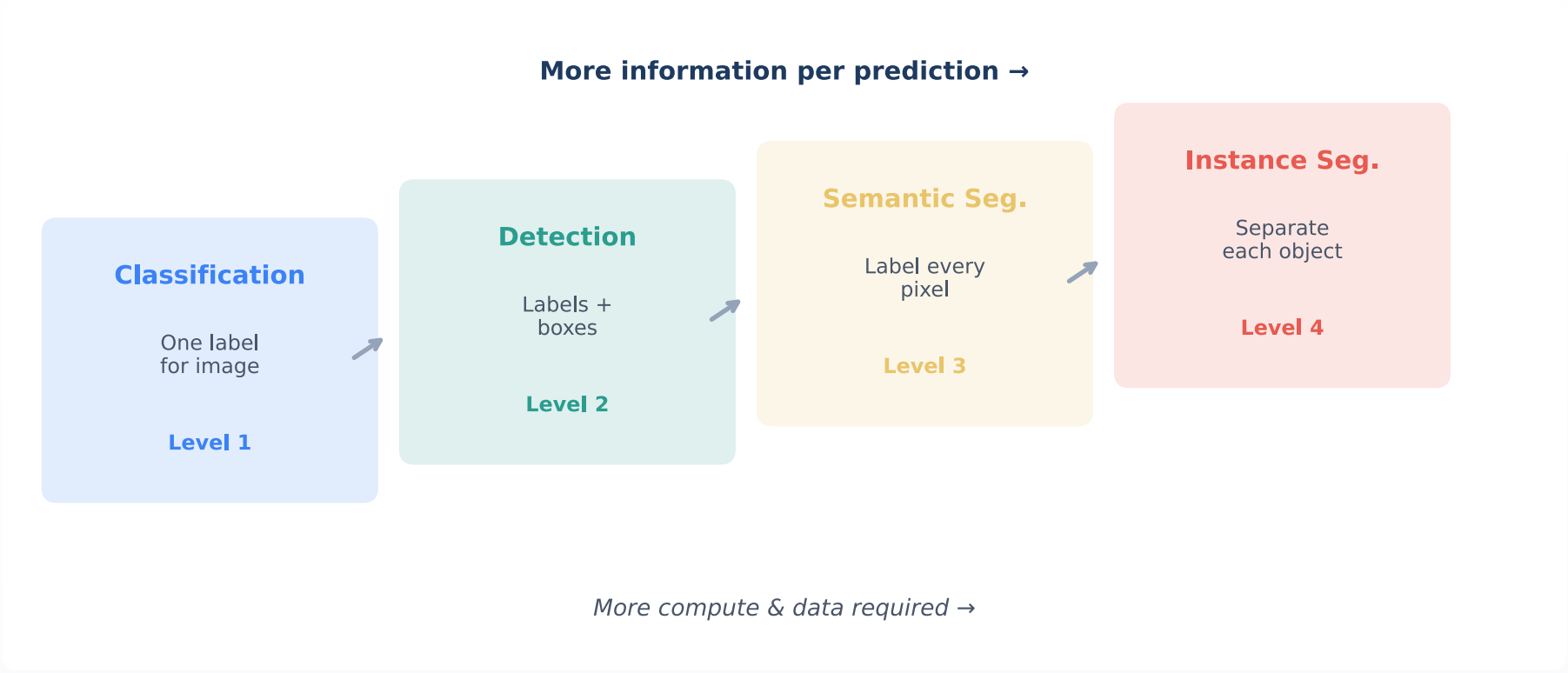
- Normal transactions look similar
- Fraud looks **different**

No labels needed - just find what's unusual!

Part 4: Computer Vision Tasks

From Classification to Segmentation

The Vision Task Ladder



Each level gives more information, but costs more!

Level 1: Image Classification

What: One label for the entire image

```
[Image of a cat]  →  "Cat"
```

Use cases:

- Photo organization ("Show me all dog photos")
- Medical imaging ("Normal or abnormal?")
- Quality control ("Defective or OK?")

Level 2: Object Detection

What: Find objects AND their locations

```
[Image with multiple objects]
    ↓
"Cat" at (x=120, y=85, w=200, h=180)
"Dog" at (x=400, y=100, w=150, h=160)
```

Detection in Action

Level 3: Semantic Segmentation

What: Label every single pixel

```
Each pixel gets a class:
- Sky pixels → "sky"
- Road pixels → "road"
- Car pixels → "car"
```

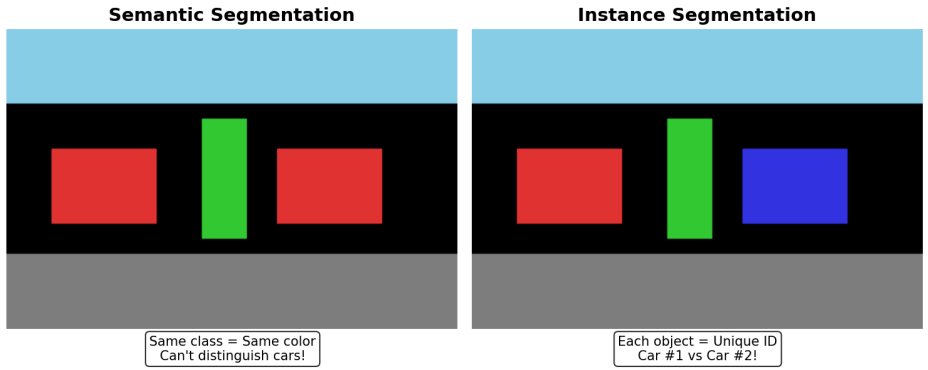
Level 4: Instance Segmentation

What: Separate each individual object

```
Not just "car" pixels, but:
- Car 1 pixels
- Car 2 pixels
- Car 3 pixels
```

Critical for self-driving cars!

Segmentation Comparison



Key Difference

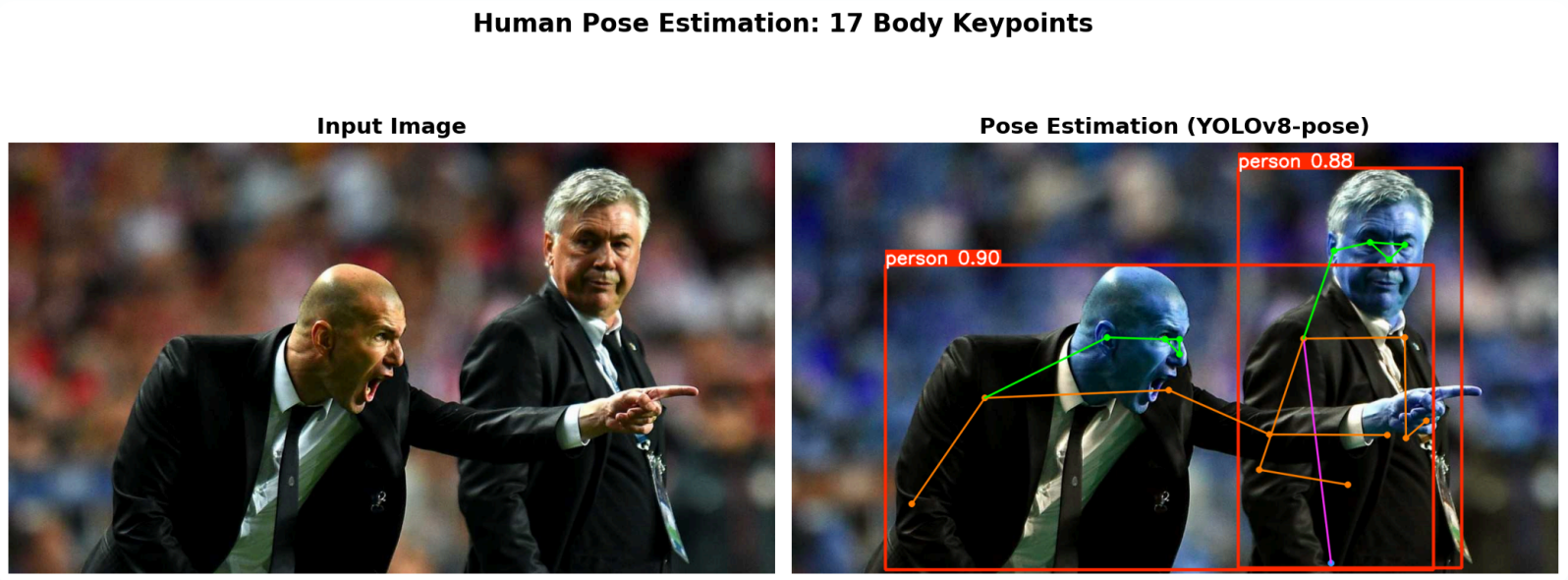
Semantic: pixel -> class

Instance: pixel -> class + object ID

Self-driving cars need Instance to track individual vehicles!

Pose Estimation

What: Find body keypoints (skeleton)



Uses: Fitness apps, motion capture, sign language

Vision Tasks Summary

Task	Output	Example Use
Classification	1 label	Photo tagging
Detection	Labels + boxes	Self-driving
Semantic Seg.	Per-pixel class	Scene understanding
Instance Seg.	Per-pixel instance	Counting objects
Pose Est.	Keypoint coords	Motion tracking

Part 5: NLP Tasks

Text Classification to Generation

Text Classification

Same as image classification, but with text:

Task	Input	Output
Sentiment	"Great movie!"	Positive
Spam	"You won $1M!"	Spam
Topic	"The stock market fell..."	Finance

Named Entity Recognition (NER)

Classify each word in a sentence:

```
"Sundar Pichai visited New York yesterday"
   PER    PER    O       LOC  LOC    O

PER = Person, LOC = Location, O = Other
```

It's like **semantic segmentation for text**!

Sequence-to-Sequence Tasks

Input: A sequence
Output: Another sequence

Task	Input	Output
Translation	"Hello, how are you?"	"Bonjour, comment allez-vous?"
Summarization	[Long article]	[Short summary]
Question Answering	"What is the capital?"	"Paris"

Text Generation

Create new text from a prompt:

```
Prompt: "Write a poem about AI"
        ↓
    Language Model
        ↓
"In circuits deep and networks wide,
 Where silicon dreams reside..."
```

This is what ChatGPT/Claude do!

Part 6: Generative AI

Creating New Data

Discriminative vs Generative

Discriminative

P(label | input)

Classifies existing data

Generative

P(data) or P(data | text)

Creates new data

Discriminative Models

Learn: P(label | input)

"Given this image, what's the probability it's a cat?"

```
[Cat image] → P(cat) = 0.95, P(dog) = 0.05
```

Generative Models

Learn: P(data) or P(data | prompt)

"Generate a new image that looks like a cat"

```
"A cat sitting on a rainbow" → [New image!]
```

The Generative AI Landscape

Domain	Tools	What It Creates
Text	ChatGPT, Claude	Essays, code, poems
Images	DALL-E, Midjourney	Any image from text
Music	Suno	Songs with lyrics
Video	Sora	Video clips
Voice	ElevenLabs	Clone voices

Generative AI is Everywhere Now!

```
Prompt: "A cute robot teaching math"
        ↓
    DALL-E / Midjourney
        ↓
    [Generated image of robot teacher]
```

These models create **new content that never existed**!

Part 7: Multimodal AI

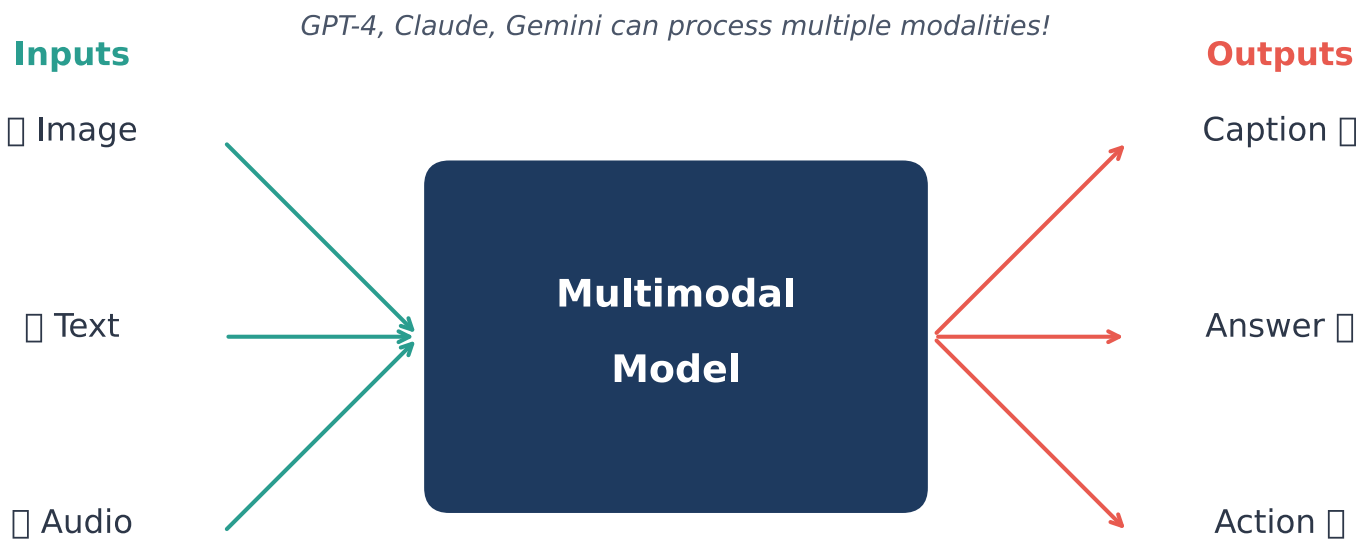
Combining Modalities

What is Multimodal?

Modality = Type of data (text, image, audio, video)

Multimodal = Multiple types combined

Multimodal Examples



Visual Question Answering

```
[Image: Red car on road]

Q: "What color is the car?"
A: "Red"

Q: "Is it daytime?"
A: "Yes, based on the lighting"
```

Needs to understand **both** image AND language!

Modern AI is Multimodal

GPT-4, Claude, Gemini can:

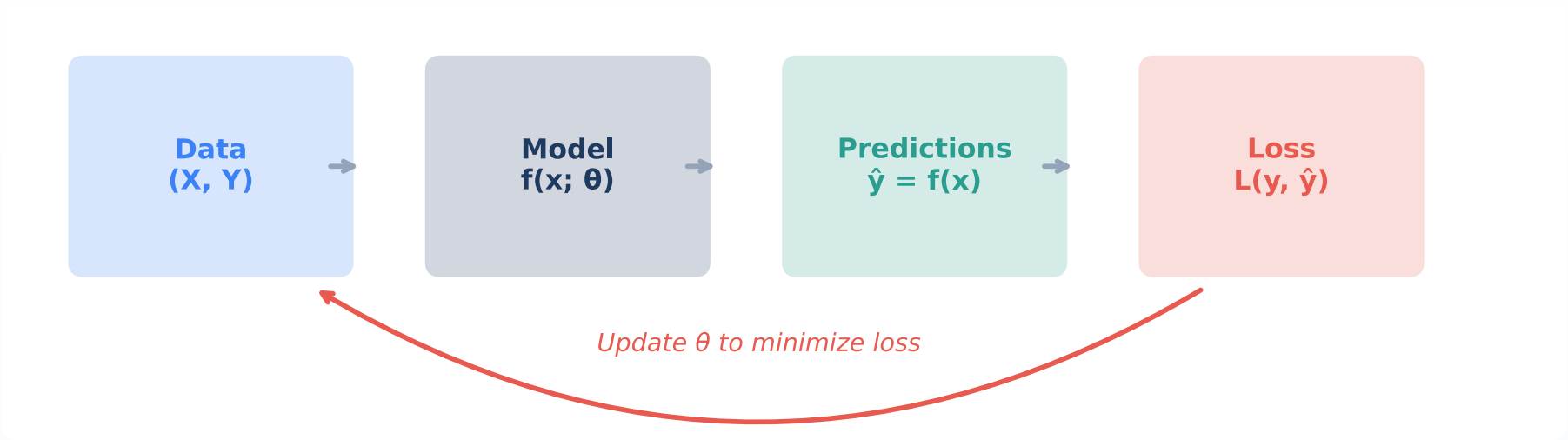
- See images
- Read text
- Understand context
- Generate responses

All in one model!

Part 8: The Universal Recipe

How All ML Works

The ML Recipe



Every ML task follows this pattern!

Step 1: Get Data

```
# Your data: features (X) and labels (y)
X = [[1500, 3, 2],    # House 1: sqft, beds, baths
     [2000, 4, 3],    # House 2
     [1200, 2, 1]]    # House 3

y = [300000, 450000, 200000]  # Prices
```

Step 2: Choose a Model

Different models for different tasks:

Task	Model Options
Classification	Logistic Regression, Decision Tree, SVM
Regression	Linear Regression, Random Forest
Complex patterns	Neural Networks, Deep Learning

Step 3: Train (Fit)

```
# The model learns from data
model.fit(X_train, y_train)

# Internally: finds parameters that minimize error
```

Step 4: Predict

```
# Use trained model on new data
new_house = [[1800, 3, 2]]
predicted_price = model.predict(new_house)
# Output: $375,000
```

Step 5: Evaluate

```
# How good is the model?
accuracy = model.score(X_test, y_test)

# Or compute specific metrics
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(y_test, predictions)
```

Part 9: Hands-On with sklearn

Your First ML Models

scikit-learn: The ML Library

```
# The pattern is always the same:
from sklearn.some_module import SomeModel

model = SomeModel()        # 1. Create model
model.fit(X_train, y_train) # 2. Train
predictions = model.predict(X_test)  # 3. Predict
```

Example: Classification with Logistic Regression

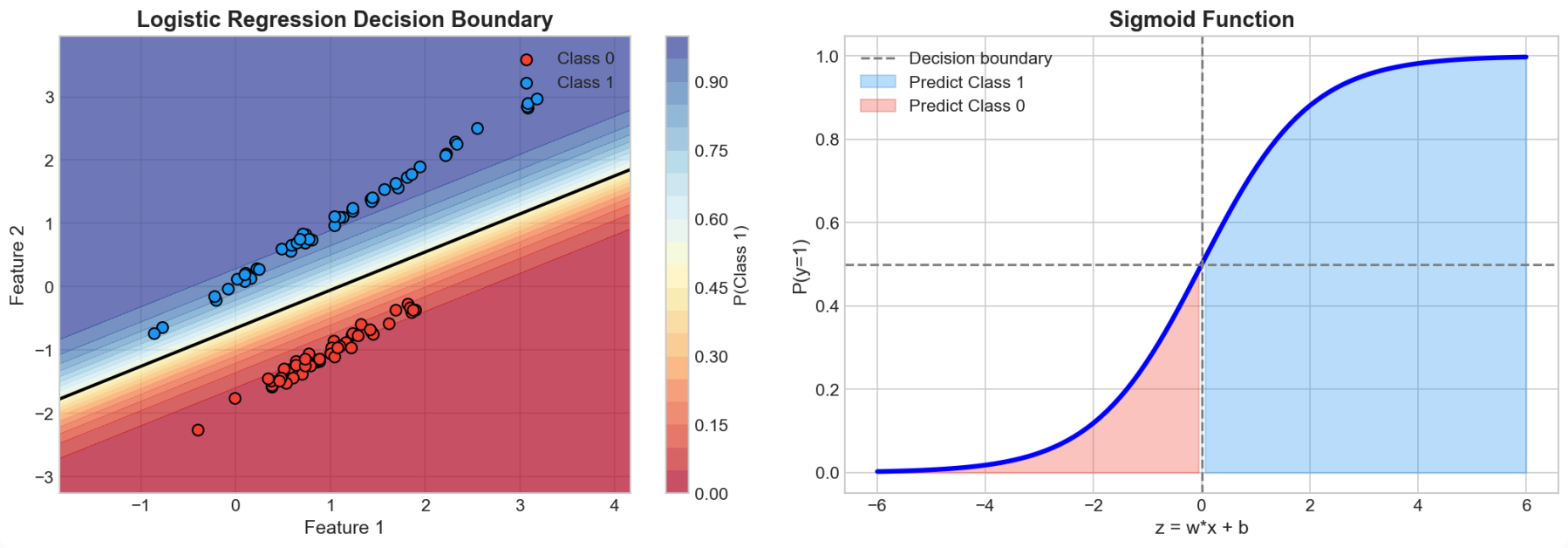
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

# Split data into train/test
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2)

# Create and train model
model = LogisticRegression()
model.fit(X_train, y_train)

# Evaluate
accuracy = model.score(X_test, y_test)
print(f"Accuracy: {accuracy:.2%}")
```

Visualizing Logistic Regression



Example: Regression with Linear Regression

```
from sklearn.linear_model import LinearRegression

# Create and train
model = LinearRegression()
model.fit(X_train, y_train)

# Predict
predictions = model.predict(X_test)

# The model learned: y = w1*x1 + w2*x2 + ... + b
print(f"Coefficients: {model.coef_}")
print(f"Intercept: {model.intercept_}")
```

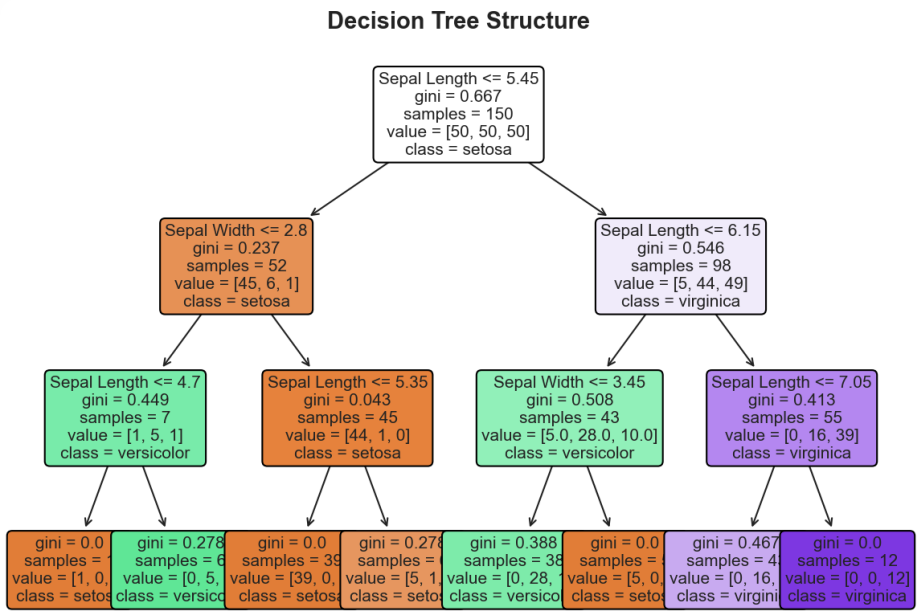
Example: Decision Tree

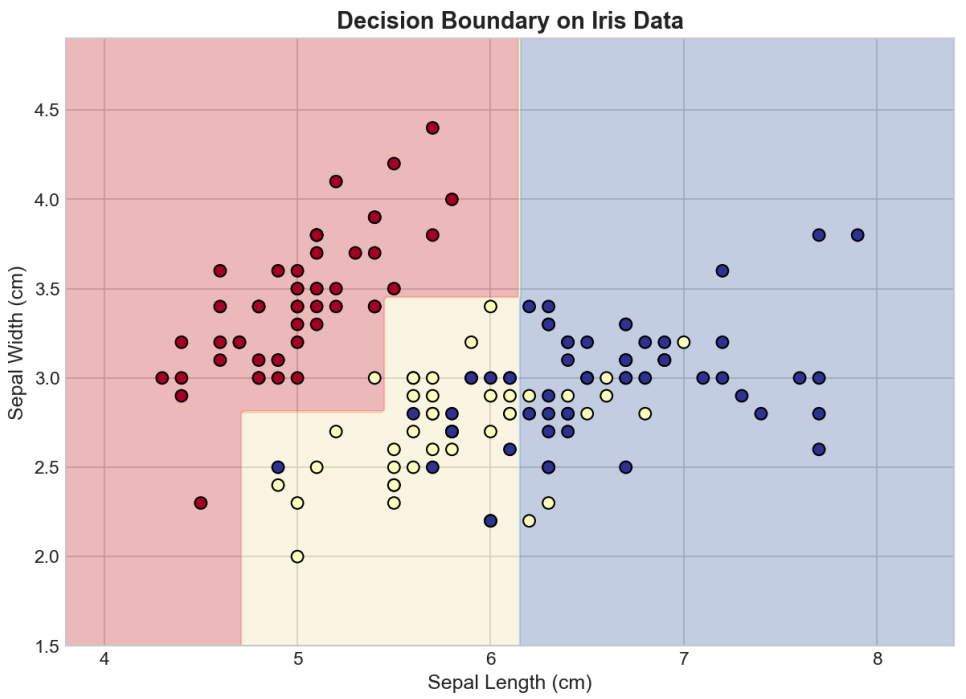
```
from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(max_depth=3)
model.fit(X_train, y_train)

# Decision trees learn IF-THEN rules!
# "If sepal_length > 5.5 AND sepal_width < 3.0
#  THEN class = versicolor"
```

Visualizing Decision Trees





Example: K-Means Clustering

```
from sklearn.cluster import KMeans

# No y labels - it's unsupervised!
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)  # Just X, no y!

# Get cluster assignments
clusters = kmeans.labels_
# Output: [0, 1, 2, 0, 1, ...]
```

The sklearn API Pattern

```
# ALL models follow this pattern:

model = SomeModel(hyperparameters)  # Create
model.fit(X_train, y_train)         # Train
predictions = model.predict(X_test)  # Predict
score = model.score(X_test, y_test)  # Evaluate
```

This works for 50+ different algorithms!

Part 10: Introduction to Neural Networks

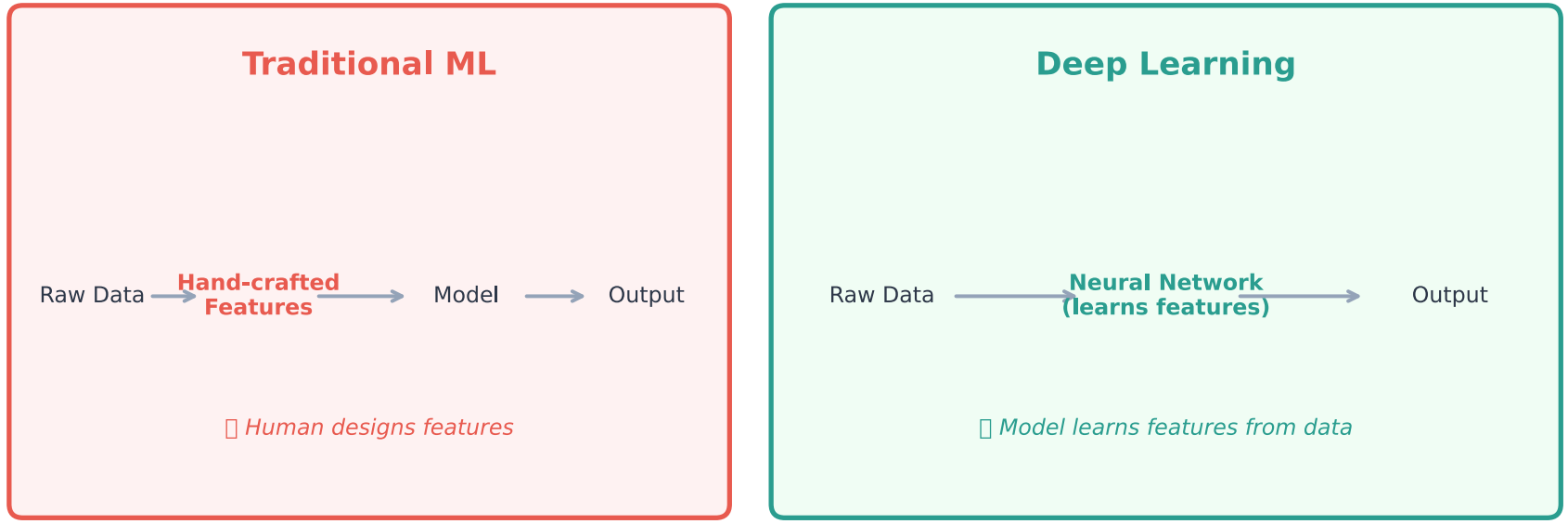
The Deep Learning Building Block

Why Neural Networks?

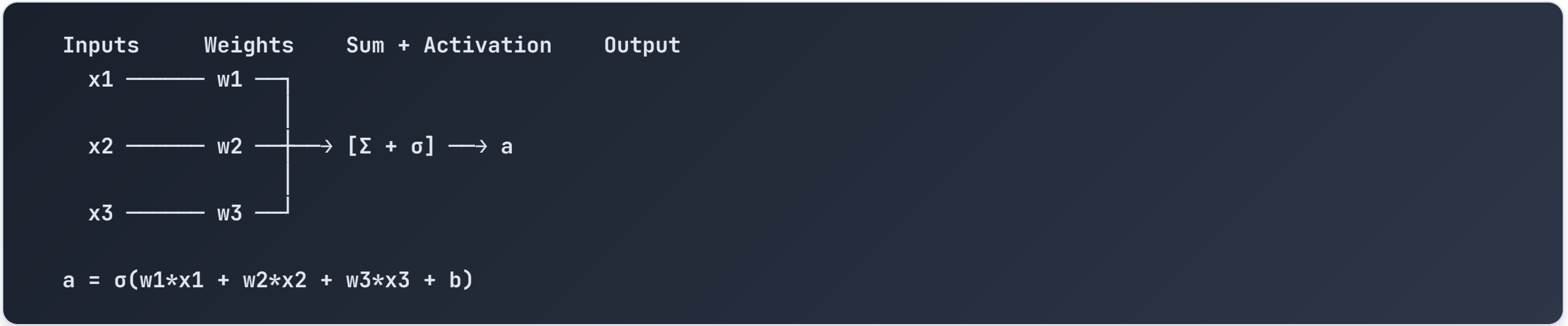
Traditional ML: You design features manually

Deep Learning: **Network learns features automatically!**

Deep Learning Revolution



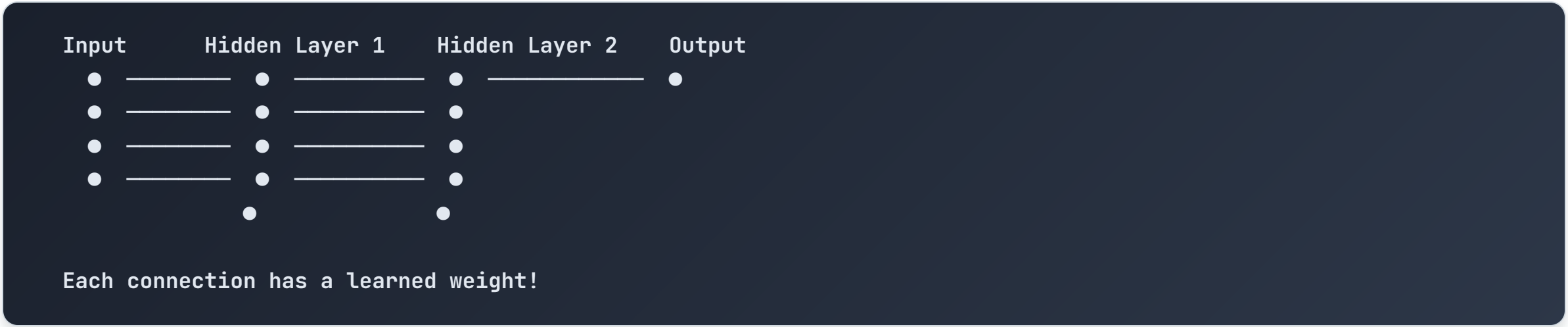
The Neuron: Basic Unit



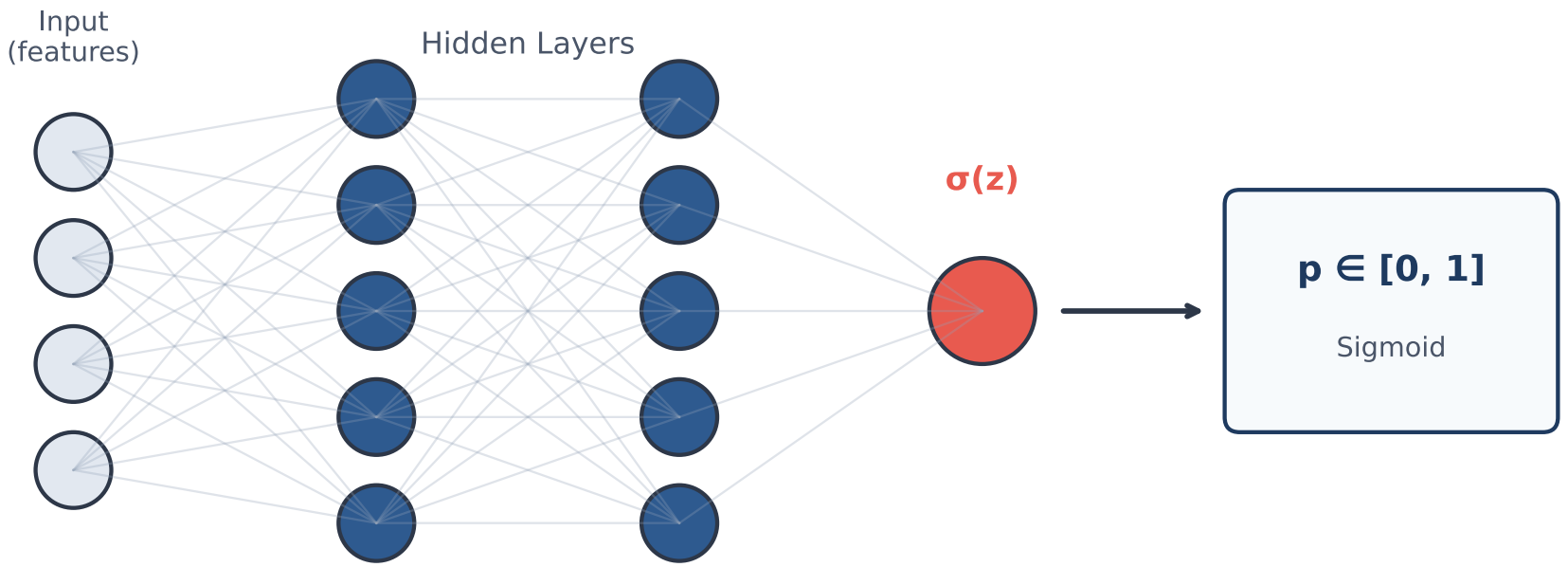
Activation Functions

Function	Formula	Use Case
ReLU	$\max(0, x)$	Hidden layers
Sigmoid	$1/(1+e^{-x})$	Binary output
Softmax	$e^{x_i}/\Sigma e^{x_j}$	Multi-class output

Neural Network = Layers of Neurons

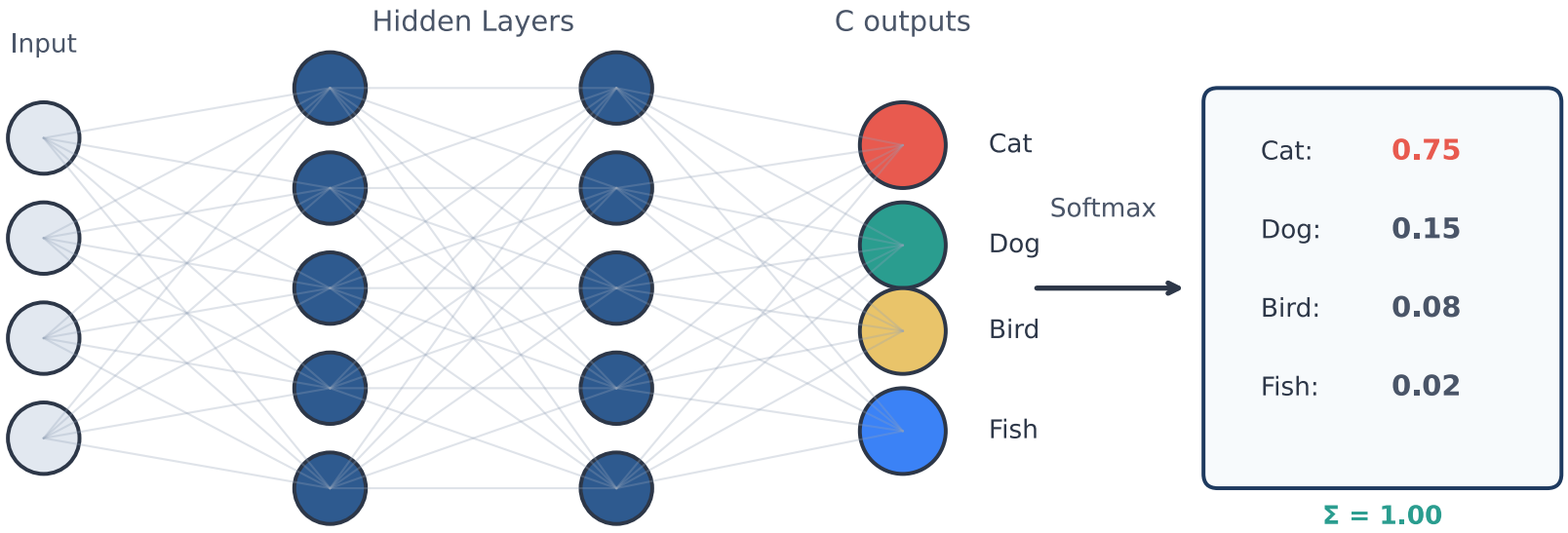


NN for Binary Classification

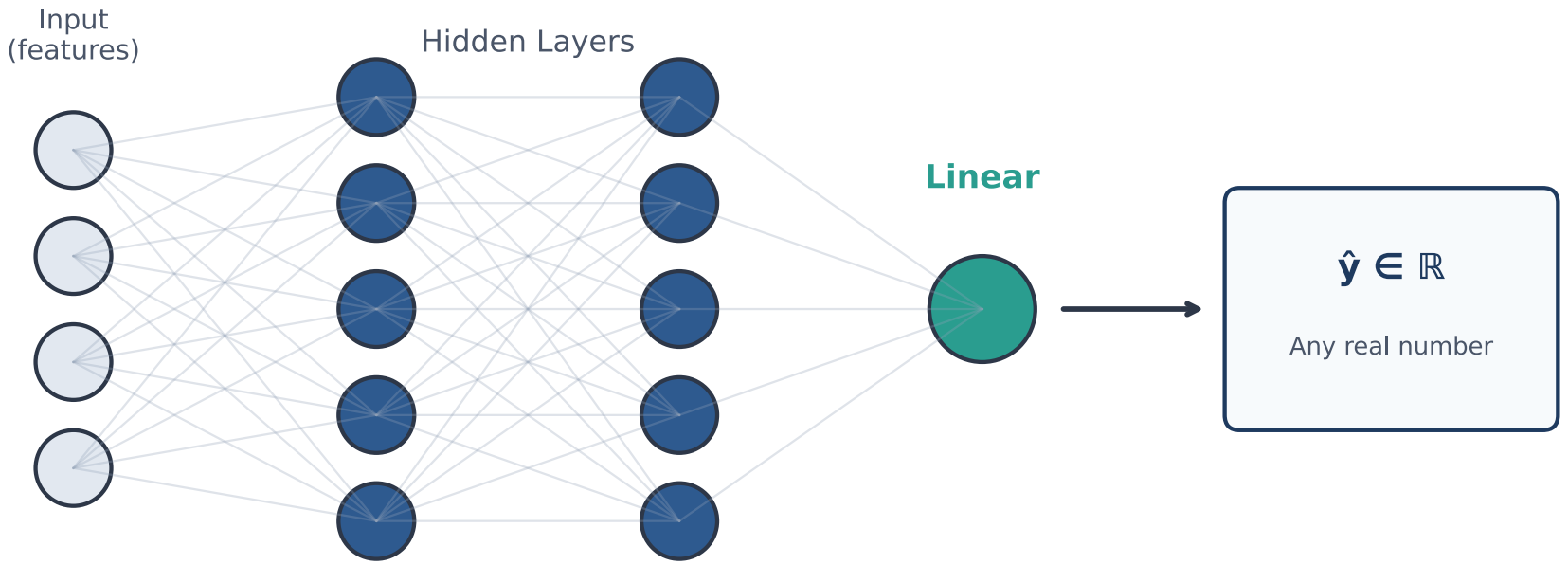


Example: p = 0.87 → "Spam" (threshold = 0.5)

NN for Multi-class Classification

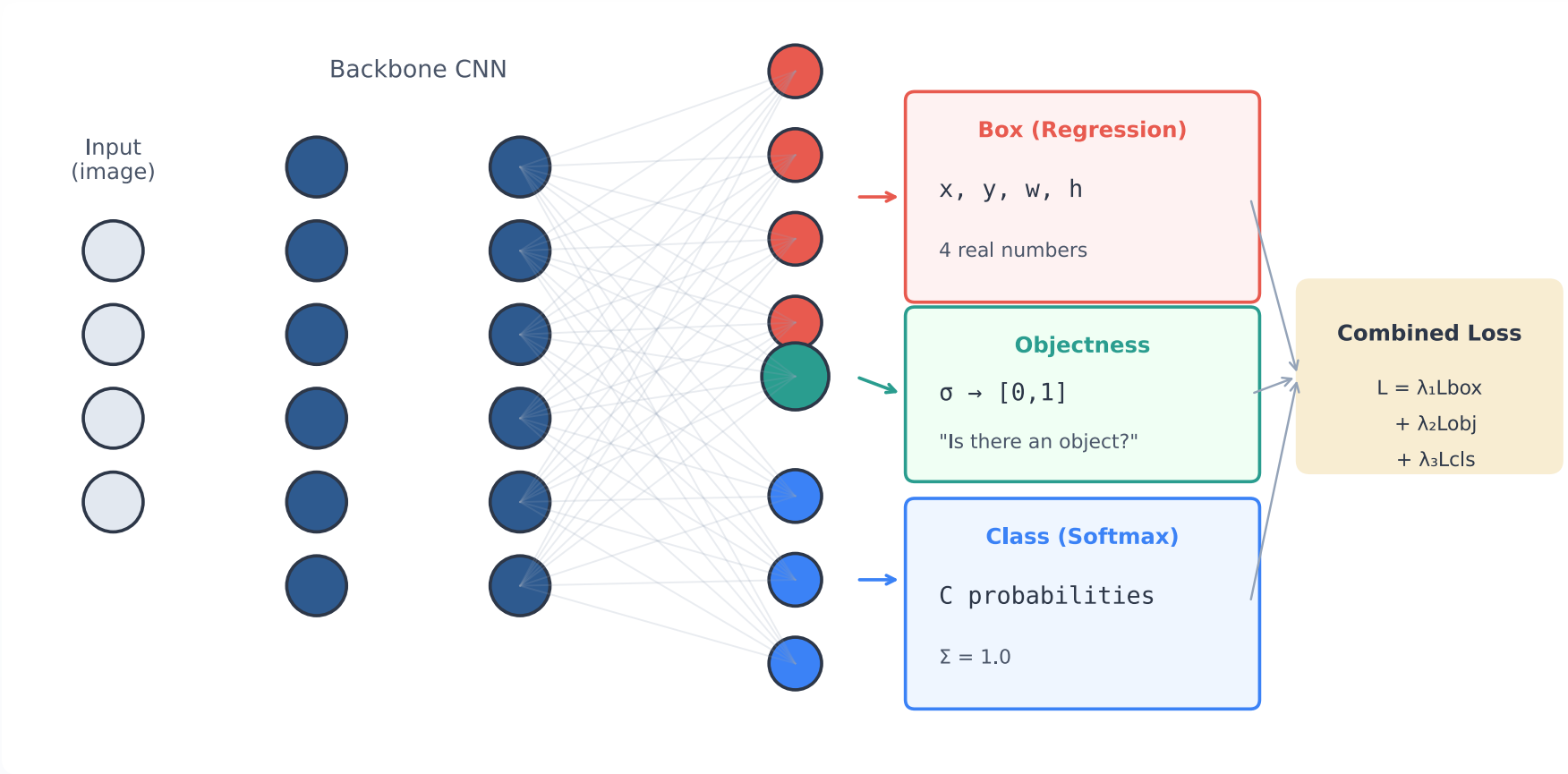


NN for Regression

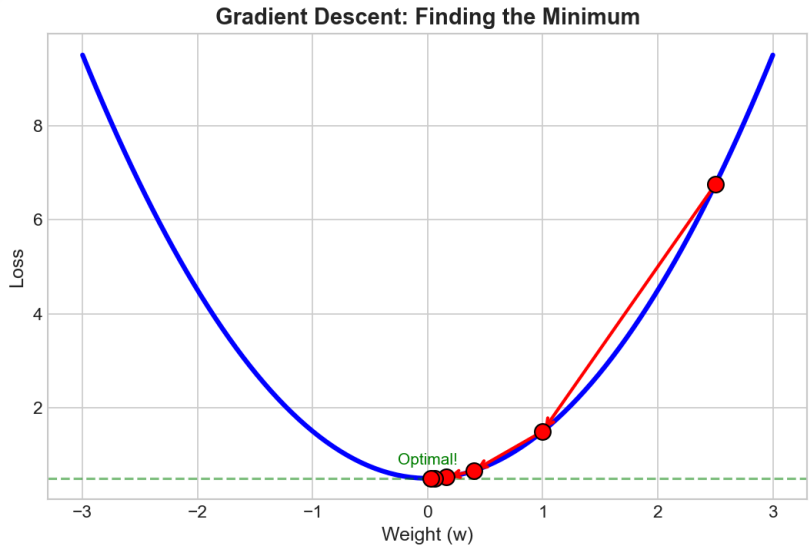


Example: ŷ = $425,000 (predicted house price)

NN for Object Detection



How 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!

Adjust weights to minimize the loss function!

Neural Networks with PyTorch

```
import torch.nn as nn

# Define a simple network
model = nn.Sequential(
    nn.Linear(4, 16),   # Input → Hidden
    nn.ReLU(),          # Activation
    nn.Linear(16, 3),   # Hidden → Output
    nn.Softmax(dim=1)   # For classification
)
```

Training Loop (Simplified)

```
for epoch in range(100):
    # Forward pass
    predictions = model(X_train)

    # Compute loss
    loss = loss_fn(predictions, y_train)

    # Backward pass (compute gradients)
    loss.backward()

    # Update weights
    optimizer.step()
```

NN Output Layer Design

Task	Output Neurons	Activation
Binary Classification	1	Sigmoid
Multi-class (C classes)	C	Softmax
Regression	1	None (linear)
Detection	4 + 1 + C	Mixed

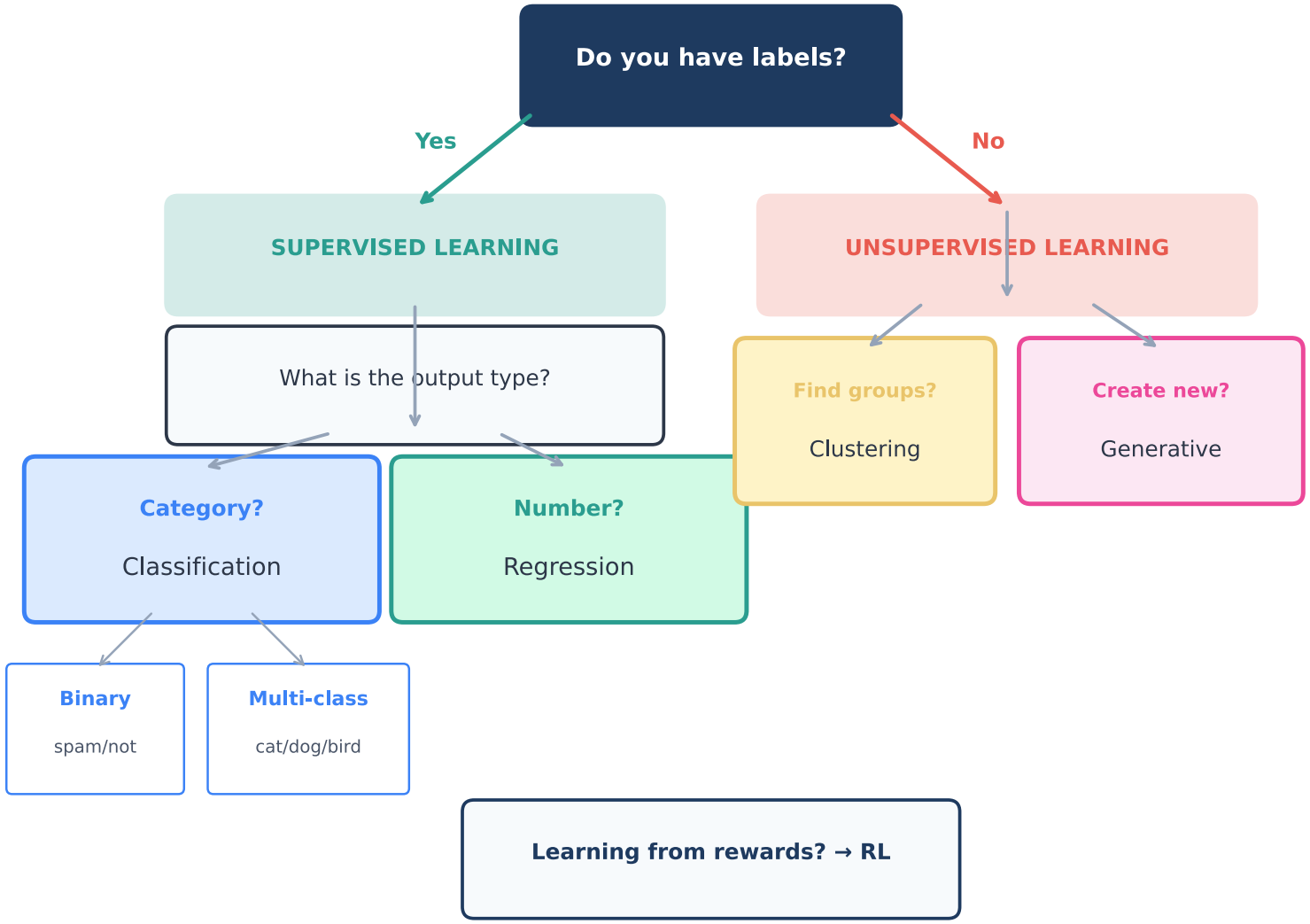
When to Use What?

Data Size	Model Choice
Small (< 1000)	sklearn: LogReg, SVM, Tree
Medium	sklearn: Random Forest, XGBoost
Large (> 10000)	Neural Networks
Images/Text	Deep Learning (CNNs, Transformers)

Part 11: Choosing the Right Task

Decision Flowchart

The ML Decision Flowchart



Quick Decision Guide

Question	Answer	Task
Have labels?	Yes →	Supervised
Have labels?	No →	Unsupervised
Predicting category?	Yes →	Classification
Predicting number?	Yes →	Regression
Need locations?	Yes →	Detection
Need pixel masks?	Yes →	Segmentation
Creating new content?	Yes →	Generation

Summary

What We Learned

Key Takeaways

1. **Learning Paradigms:** Supervised, Unsupervised, RL
2. **Supervised:** Classification (categories) vs Regression (numbers)
3. **Vision:** Classification → Detection → Segmentation
4. **NLP:** Classification → NER → Seq2Seq → Generation
5. **Generative AI:** Creates new content
6. **sklearn:** Unified API for classical ML
7. **Neural Networks:** Deep learning building block

The Output Type Tells You Everything

Output Type	Task Family
Category	Classification
Number	Regression
Boxes + labels	Detection
Pixel labels	Segmentation
Sequence	Seq2Seq
New data	Generation

Coming Up Next

Lecture 3: Language Models

- Next Token Prediction
- From GPT to ChatGPT
- RLHF

Lecture 4: Object Detection

- YOLO and beyond
- Real-time detection

Thank You!

The Big Picture

Understanding the output type helps you pick the right approach.

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

Questions?