上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

Lecture 8: Machine Learning Paradigm

Tao Huang

John Hopcroft Center, School of Computer Science, Shanghai Jiao Tong University

https://taohuang.info/cs3317

https://oc.sjtu.edu.cn/courses/89538

Spam Detection

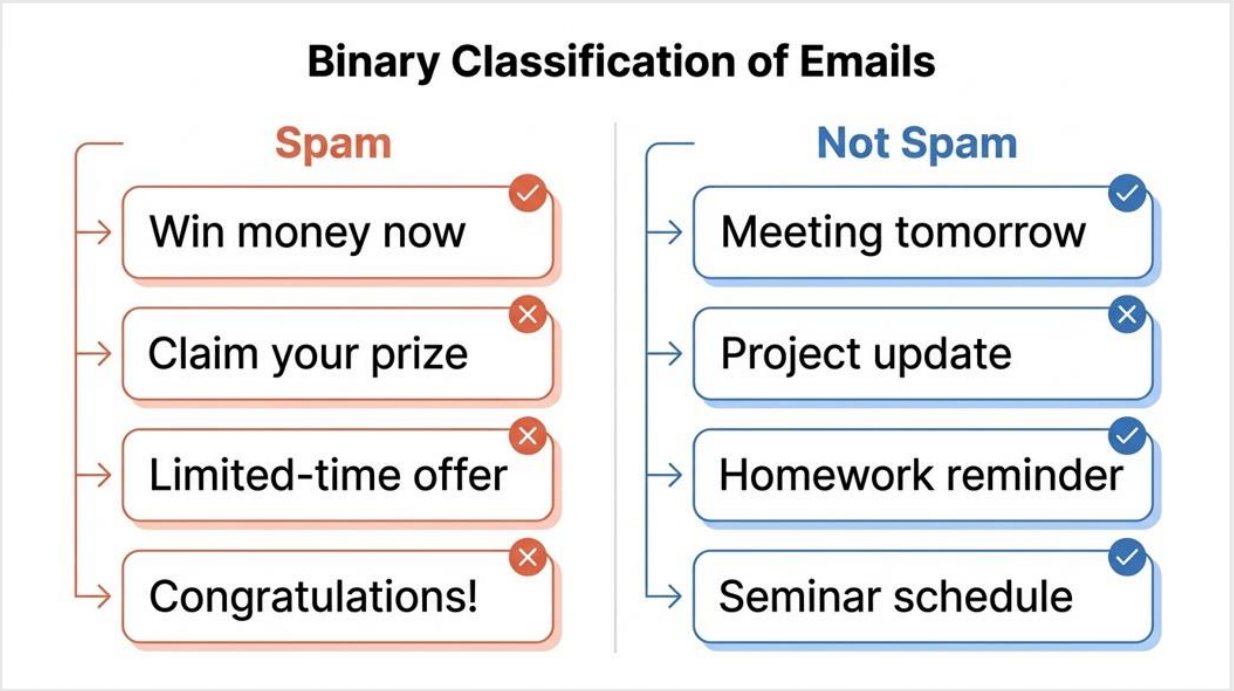
The Problem

- Emails: useful vs spam
- Manual filtering is impossible at scale

Goal

- Automatically classify emails

Question: How would you solve this?



The task is a binary classification problem.

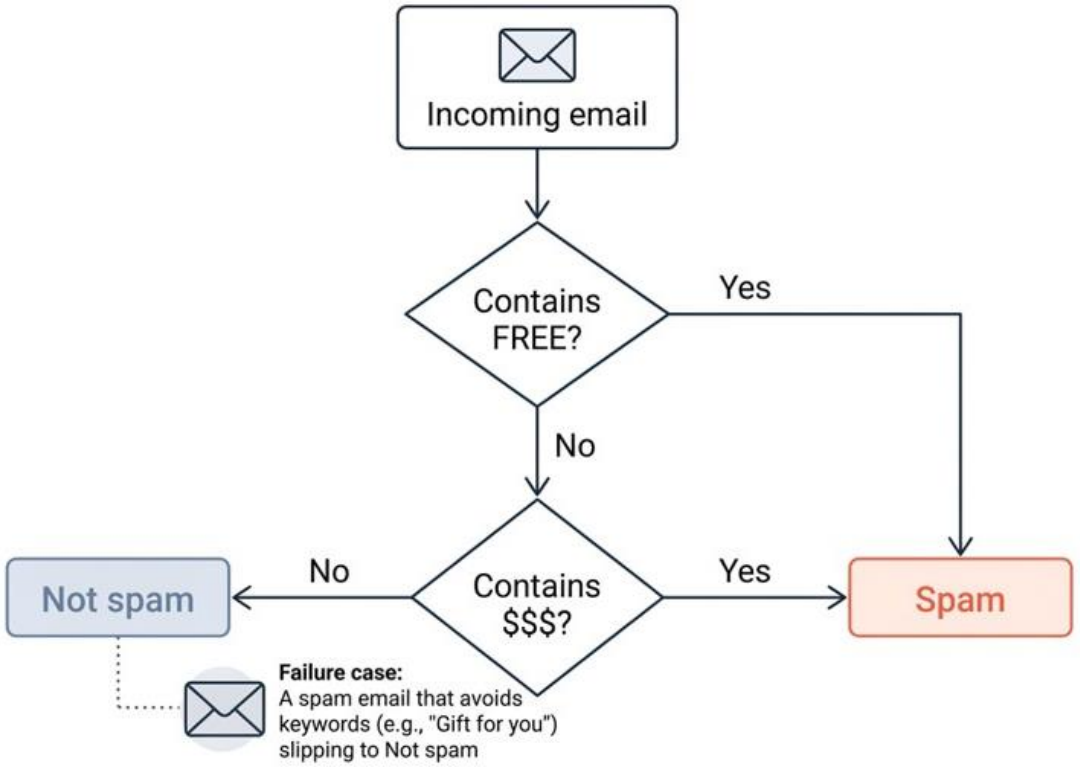
First Attempt: Rule-Based System

Possible Rules

- If contains “FREE” → Spam
- If contains “WIN $$$” → Spam

Problems

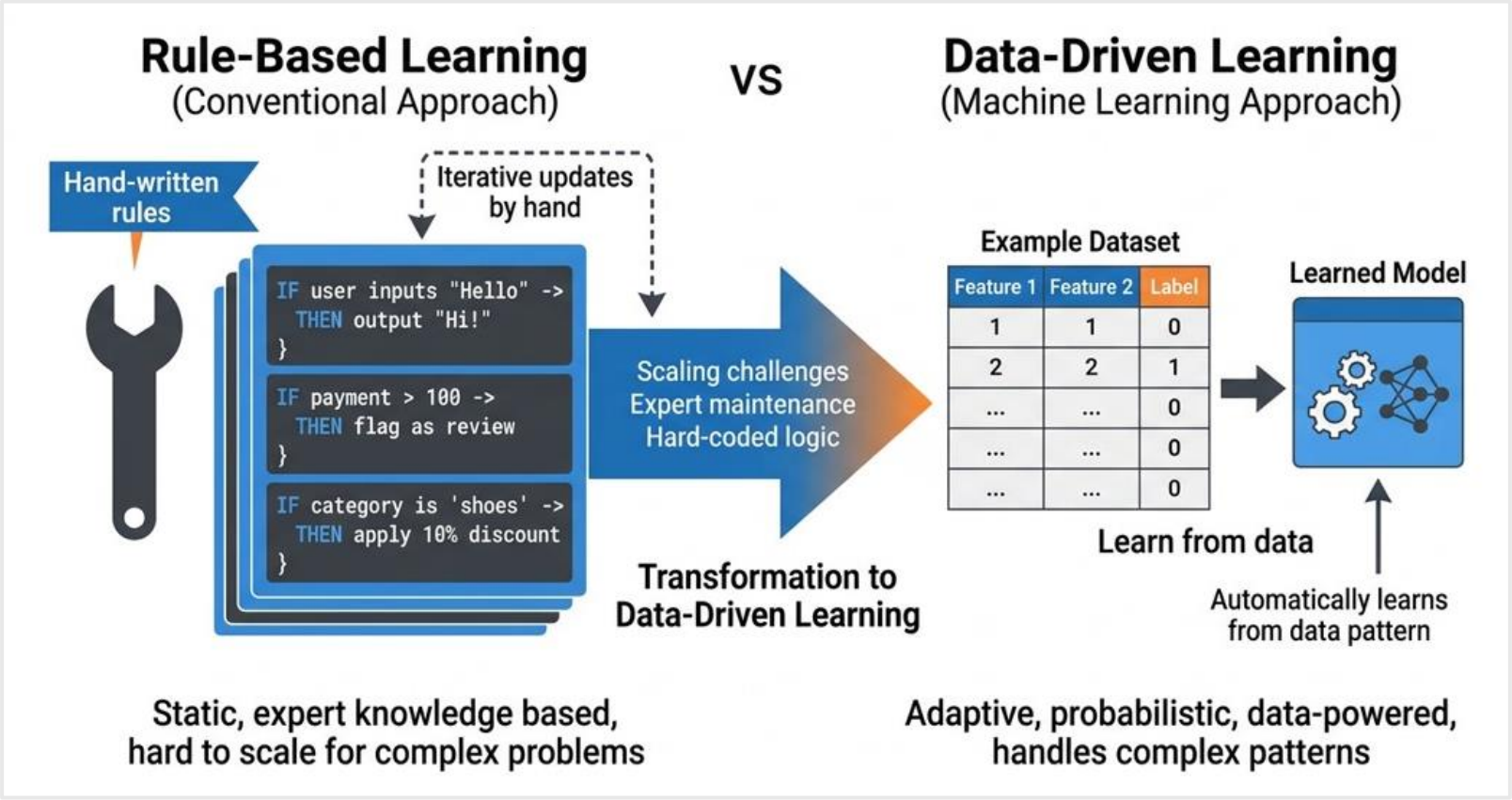
- Brittle (easy to break)
- Incomplete coverage
- Easy to bypass (e.g., "F R E E")



Hand-crafted rules are fragile and limited.

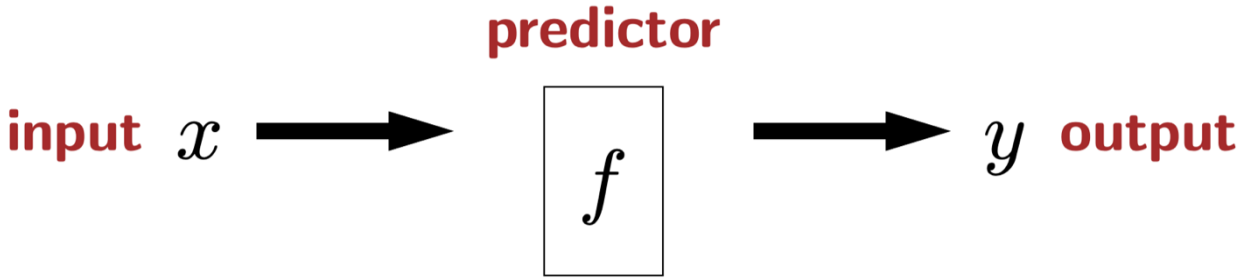
Key Idea: Let the Machine Learn

Instead of writing rules by hand, we **learn from data**.



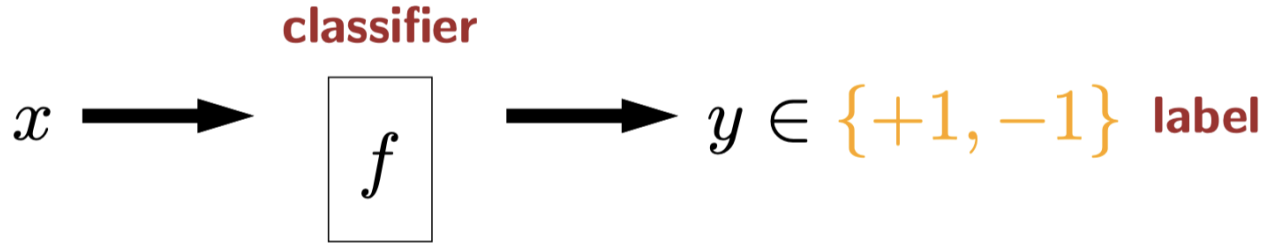
From manually designed rules to data-driven learning.

Machine Learning Models



- A **model** (f) takes some **input** x and produces some **output** y
- Input can be arbitrary (an image or sentence), output is determined by **prediction task**

Binary Classification



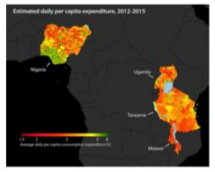
Fraud detection: credit card transaction → fraud or no fraud

Toxic comments: online comment → toxic or not toxic

Extension: multi-class classification: $y \in \{1, \ldots, K\}$

Regression

$$x \longrightarrow \boxed{f} \longrightarrow y \in \mathbb{R} \text{ response}$$



Poverty mapping: satellite image → asset wealth index

Housing: information about house → price

Arrival times: destination, weather, time → time of arrival

Structured Prediction

$x \rightarrow f \rightarrow y$ is a complex object

Machine translation: English sentence → Chinese sentence

Dialogue: conversational history → next sentence

Image captioning: image → sentence describing image

Image segmentation: image → segmentation

Supervised Learning Setup

Training Data

A set of labeled examples:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$$

- Input x_i : email
- Label y_i : the desired output

  y = 1 (Spam)

  y = 0 (Not Spam)

Goal

$$f(x) \rightarrow y$$

Training set

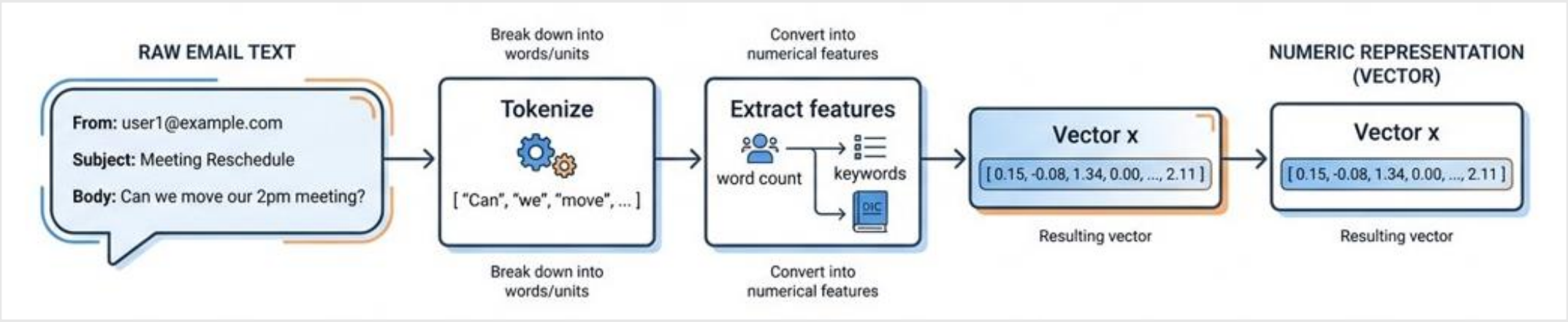
Email (text)	Label
Subject: CONGRATULATIONS! You've WON a NEW iPhone! Click NOW to claim...	Spam
Subject: Meeting confirmation: Project Update (Tuesday, Oct 26th at 10 AM)	Not spam
Subject: Urgent: Account security alert - Action required immediately! (Sign in to update info)	Spam
Subject: Re: Vacation rental inquiry - Available dates for June 2024?	Not spam
Subject: Limited time offer! Weight loss pills - Guaranteed results in 1 week!	Spam

Training data for supervised learning.

Representation Problem

- Computers cannot calculate with raw strings.
- We need a feature extractor $\boldsymbol{\phi(x)}$ to convert input into numbers.

$$\boldsymbol{x \rightarrow \phi(x) \in \mathbb{R}^d}$$



Raw inputs must be converted into numerical vectors.

Feature Examples

Defining Features

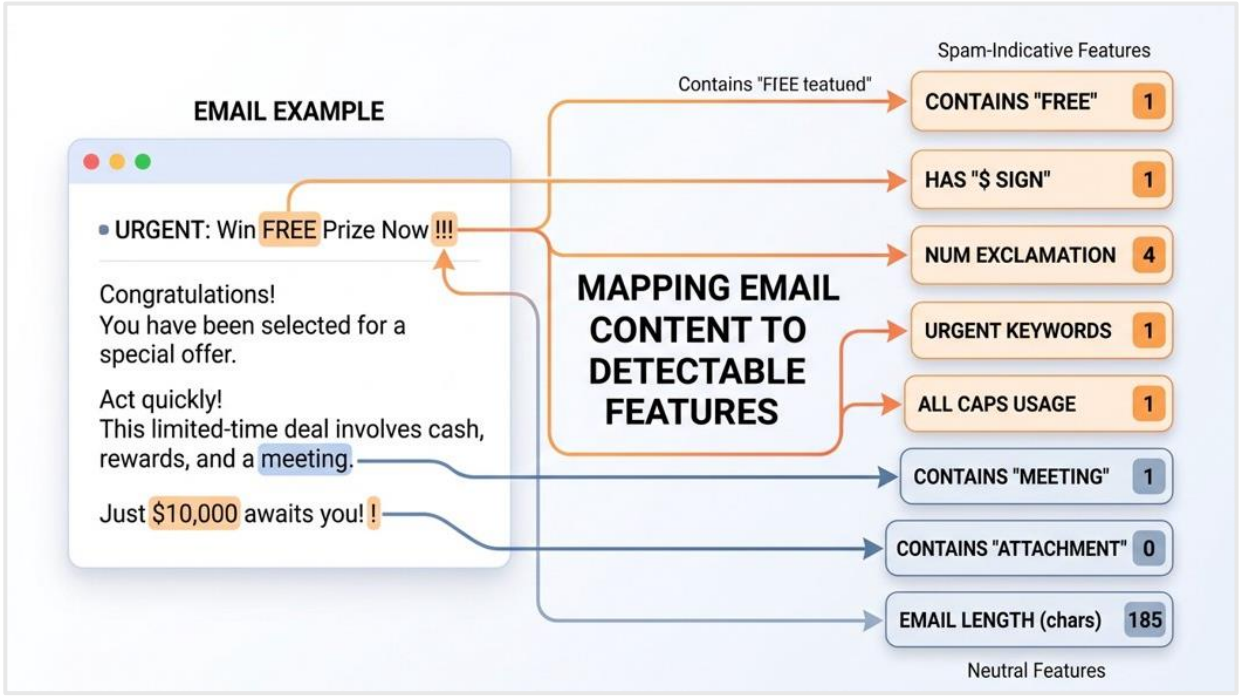
We can define many indicators:

- $\phi_1(x) = 1$ if "FREE" in x, else 0
- $\phi_2(x) = 1$ if "$" in x, else 0
- $\phi_3(x) =$ NUM_EXCLAMATIONS

…

Result is a vector:

$$\phi(x) = [1, 1, 4, \cdots]$$

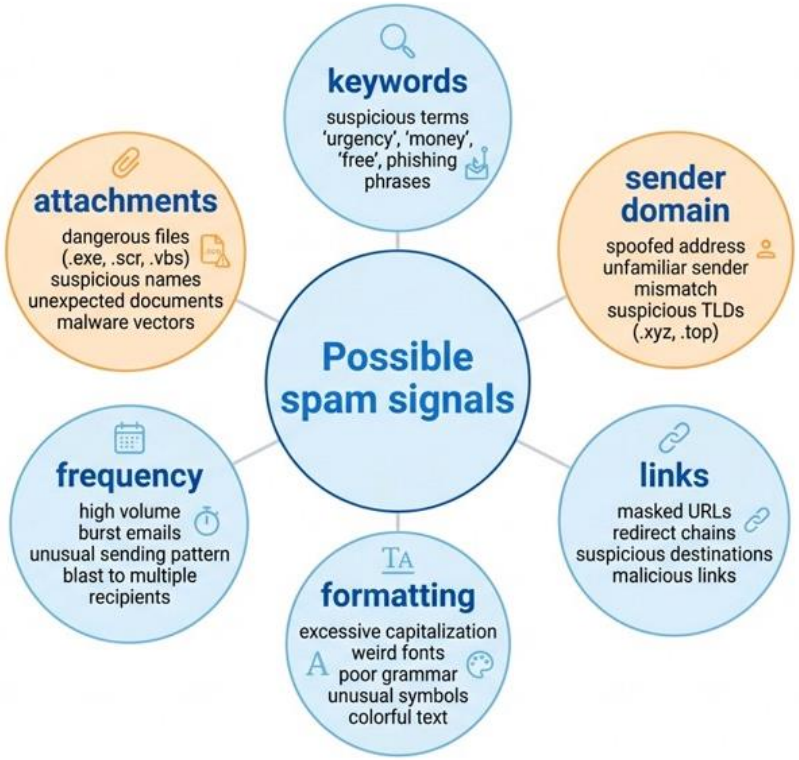


Feature extraction turns raw text into useful signals.

Think: Feature Engineering

Designing features requires **domain intuition**.

What properties distinguish spam from normal emails?



Feature engineering depends on intuition.

Feature Vector Definition

Vectors

- Feature vector $\phi(x)$: data state
- Weight vector w: model parameters

Score

The dot product aggregates evidence:

$$w \cdot \phi(x) = \sum_j w_j \phi_j(x)$$

Feature	Value (x_i)	Weight (w_i)	Contribution ($w_i x_i$)
contains FREE	1	2.0	2.0
contains meeting	0	-1.5	0
has $	1	1.2	1.2
num !	2	0.6	1.2
sender known	1	-2.0	-2.0

Each feature contributes to the final decision.

Feature Templates

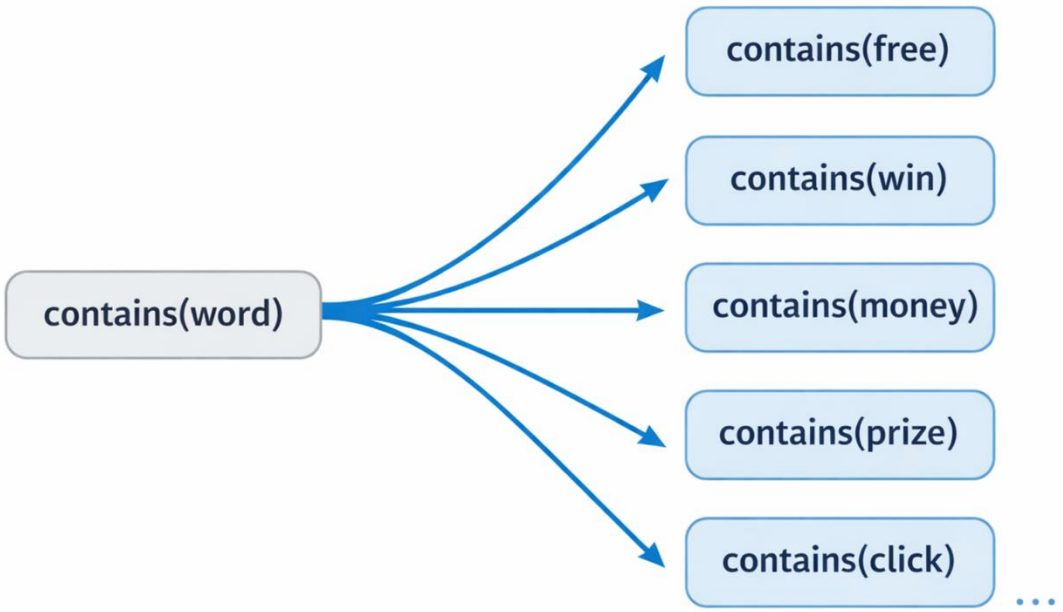
Scaling Up

We don't manually list "FREE", "$", "Meeting"...

Templates

- Define a pattern: contains(word)
- For every word in the English dictionary, create a feature.

Result: **d ≈ 50,000+**



Feature templates generate large feature spaces

Sparse Features

- Dimensionality d is very large (e.g., 50,000+)
- But an email only as ~100 unique words.
- Most entries in are $\phi(x)$ zero. This is sparsity.

$$\phi(x) \in \mathbb{R}^{20}$$

Only a few features are active (4 of 20)

The Linear Model

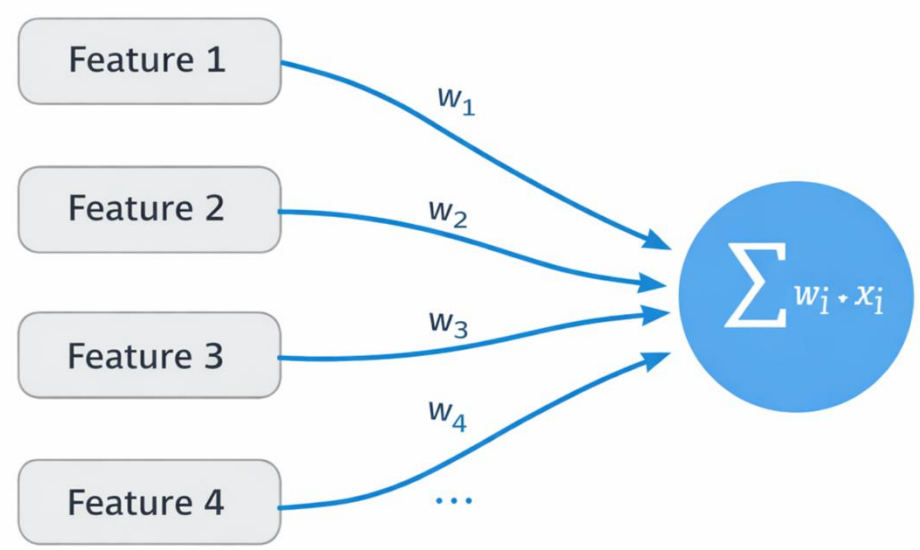
Model Definition

We assume a linear scoring function:

$$o = w \cdot \phi(x)$$

Interpretation

- It aggregates weighted evidence.
- $w_i > 0$: feature i indicates Spam
- $w_i < 0$: feature i indicates Not Spam



Model aggregates weighted evidence from features

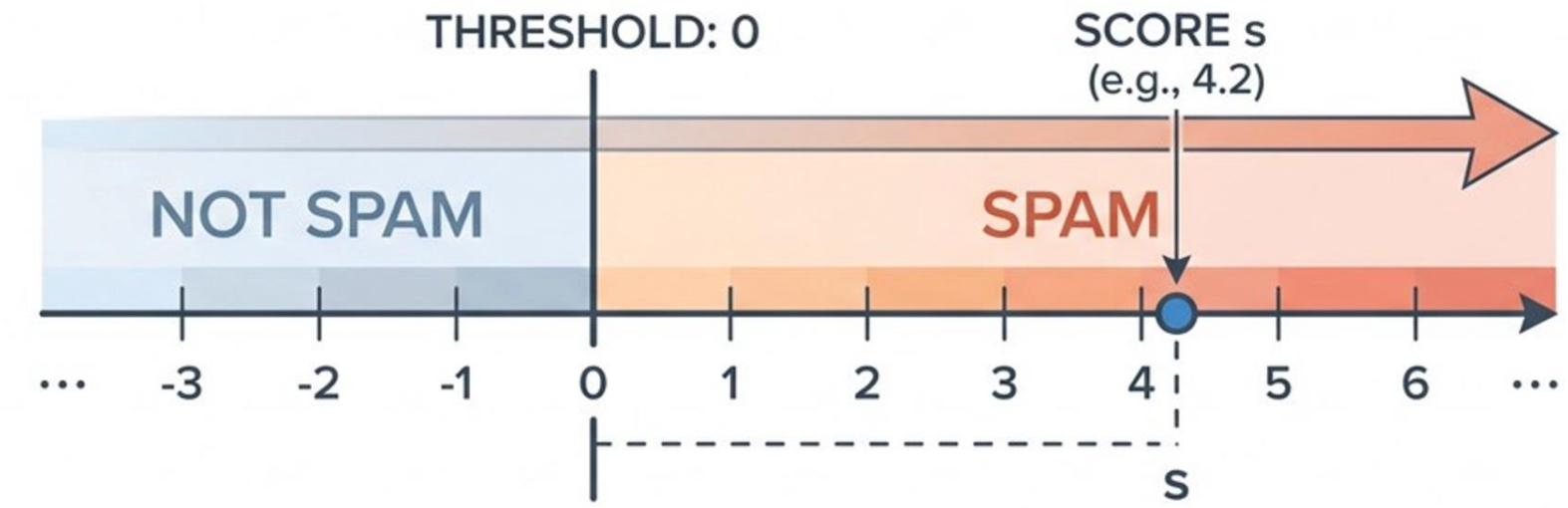
Decision Rule

- From score to class:

Predict Spam if $w \cdot \phi(x) > 0$

Predict Not Spam if $w \cdot \phi(x) \leq 0$

(Or use a threshold T instead of 0)



Classification is made by thresholding the score.

Hypothesis Class

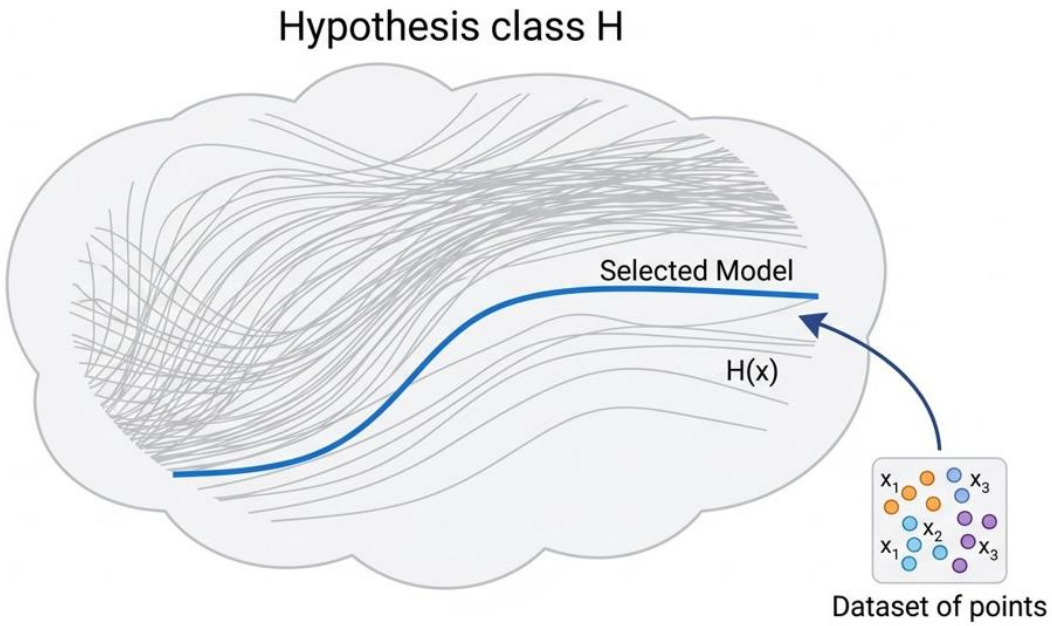
Which w?

- We haven't found w yet.
- We defined the set of all possible predictors.

The Hypothesis Class $\mathcal{H}$:

$$\mathcal{H} = \{f_w(x) = w \cdot \phi(x)\}$$

Learning = Choosing a function from a set.



Two Design Problems

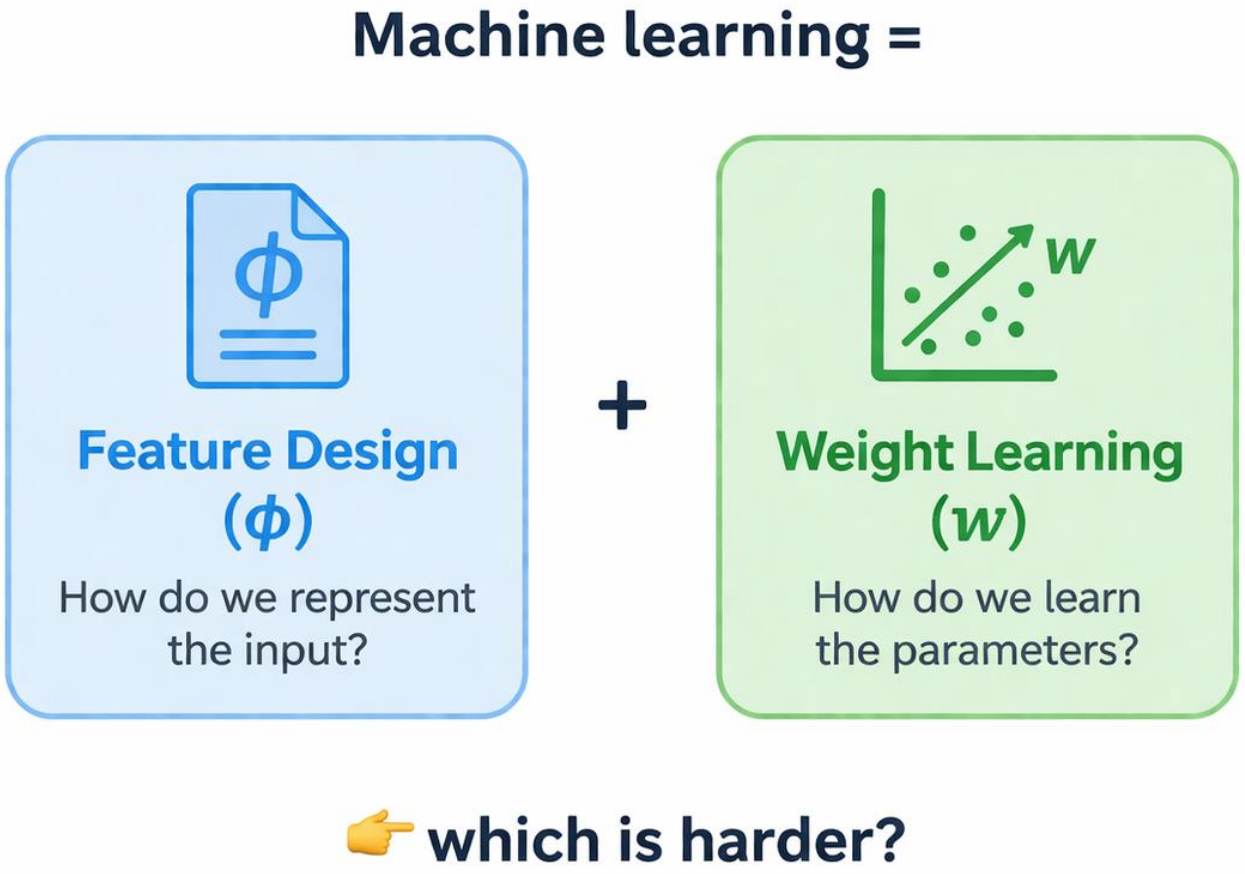
1. Representation

How to map $x \rightarrow \phi(x)$?

(Feature Engineering)

2. Weight Learning

- Which hypothesis class?
- How to define "best"?



What is a Good Model?

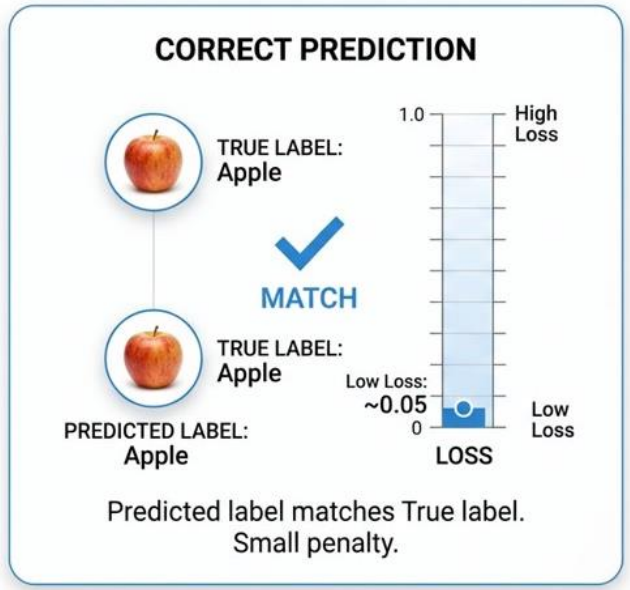
Loss Function

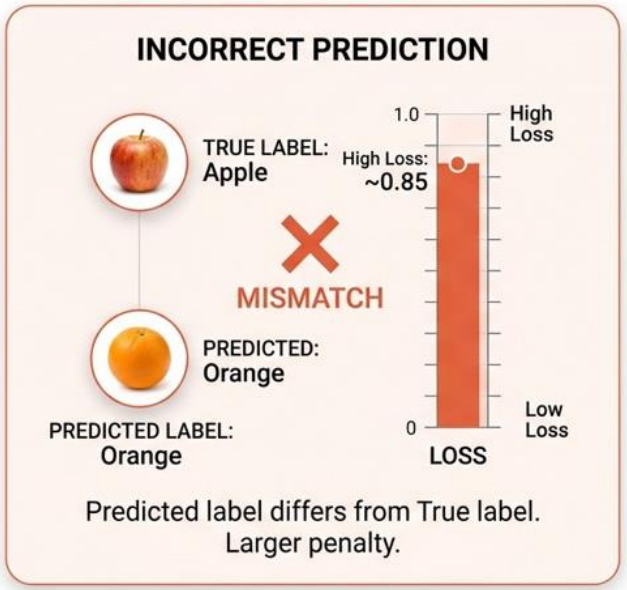
Quantifies "badness" of prediction.

$$\mathcal{L}(f_w(x), y)$$

– correct: small loss

– wrong: large loss





Example: 0-1 Loss

1 if predict != y

0 if predict == y

Empirical Risk

Training Goal

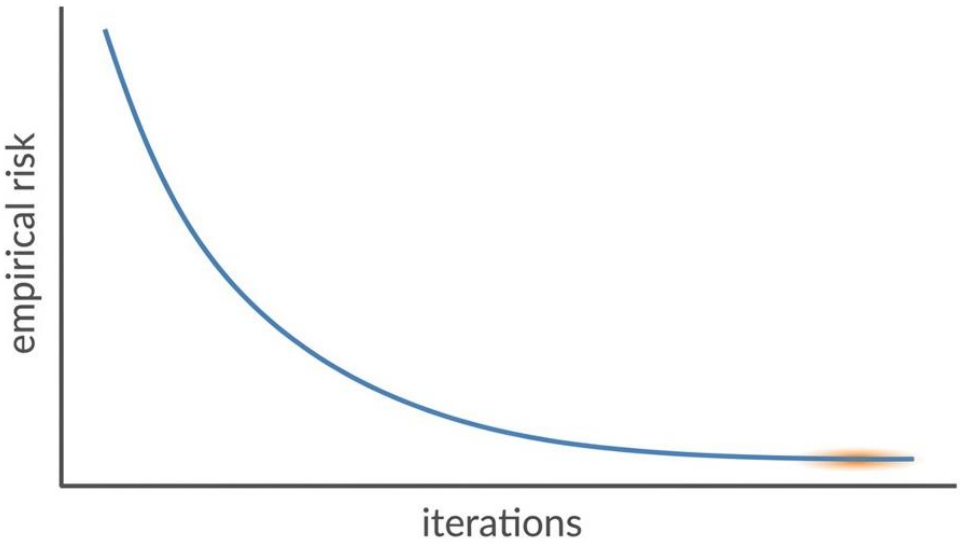
Minimize average loss on training data.

Empirical Risk

$$\mathcal{R}(w) = \frac{1}{|\mathcal{D}|} \sum_{i} \mathcal{L}(f_w(x_i), y_i)$$

Optimization

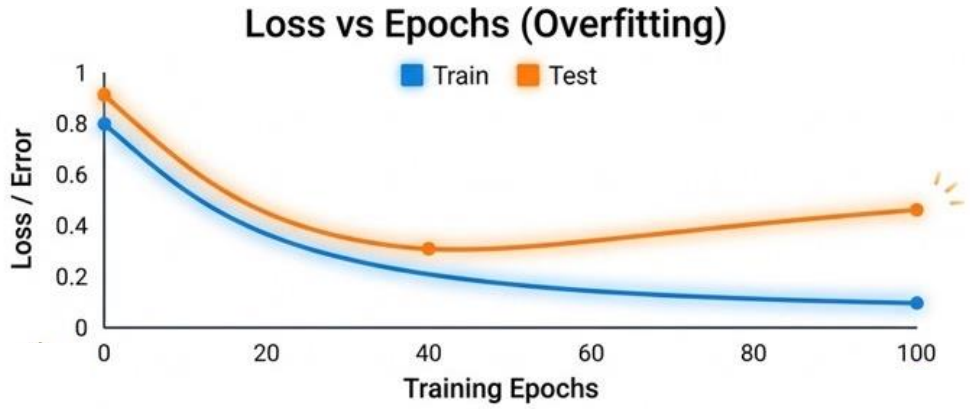
Find $\mathrm{w}^* = \arg\min \mathcal{R}(w)$

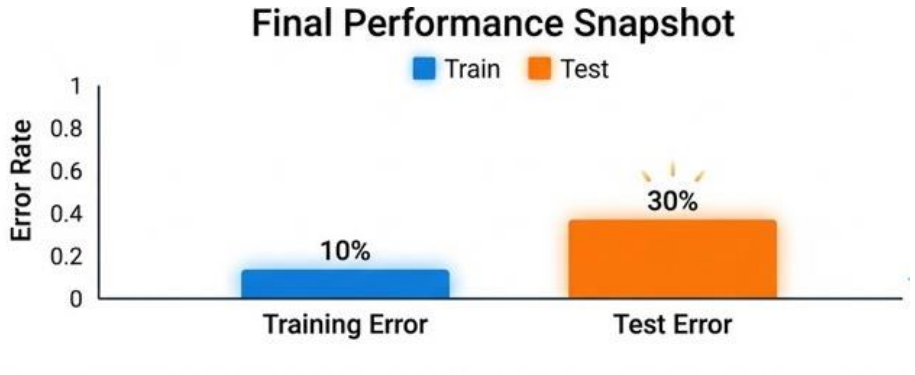


Training seeks parameters that minimize risk.

What Could Go Wrong?

- Minimizing training loss is not enough.
- We care about performance on unseen data (future emails).



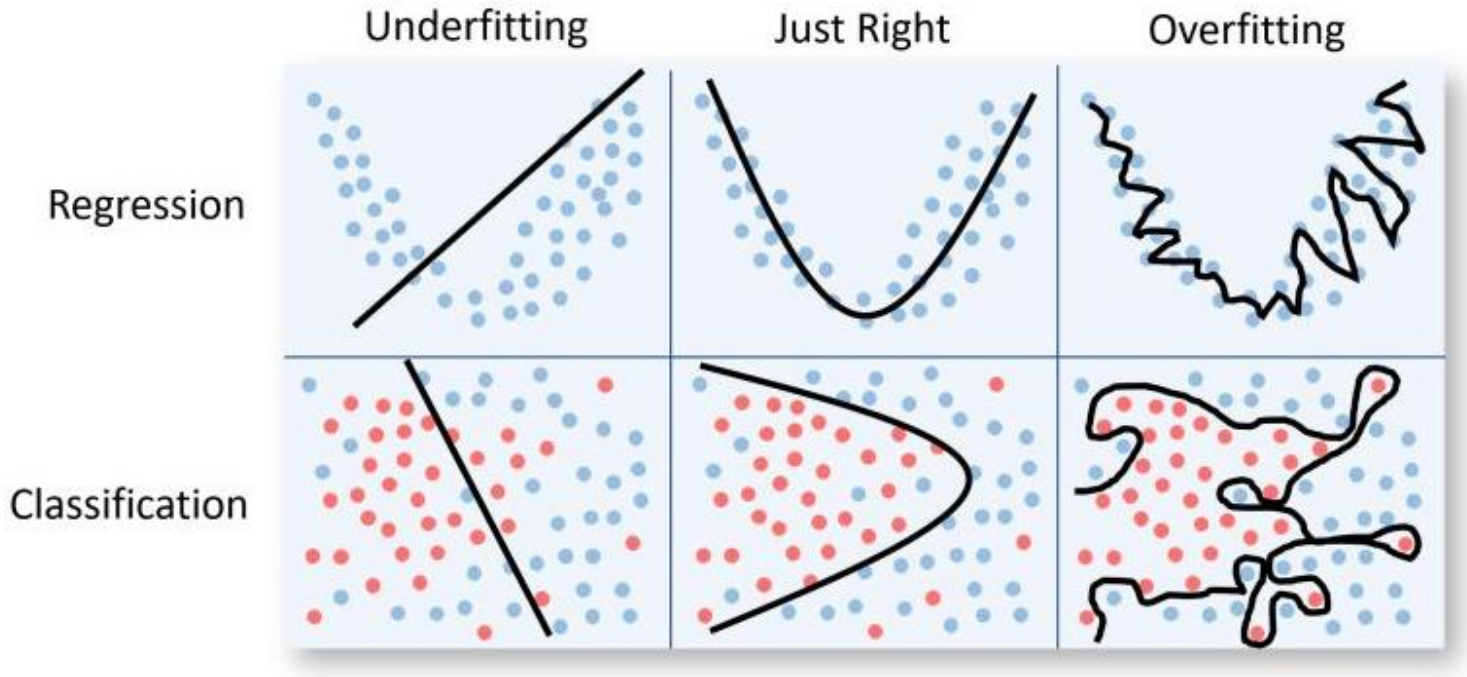


Good training performance does not guarantee future success.

Overfitting

Definition

- Fitting noise instead of signal.
- Model is too flexible (memorizes the training data)



Generalization

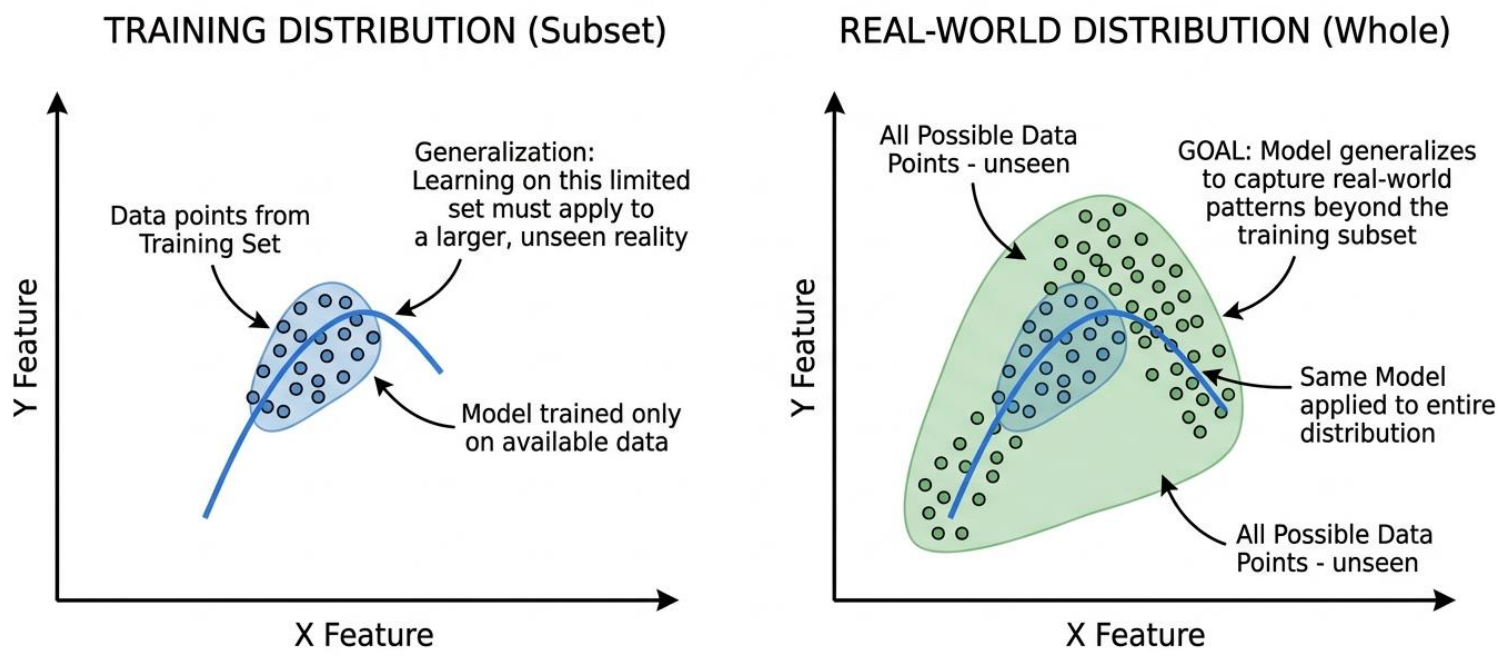
Goal

Good performance on new, unseen examples.

Assumption

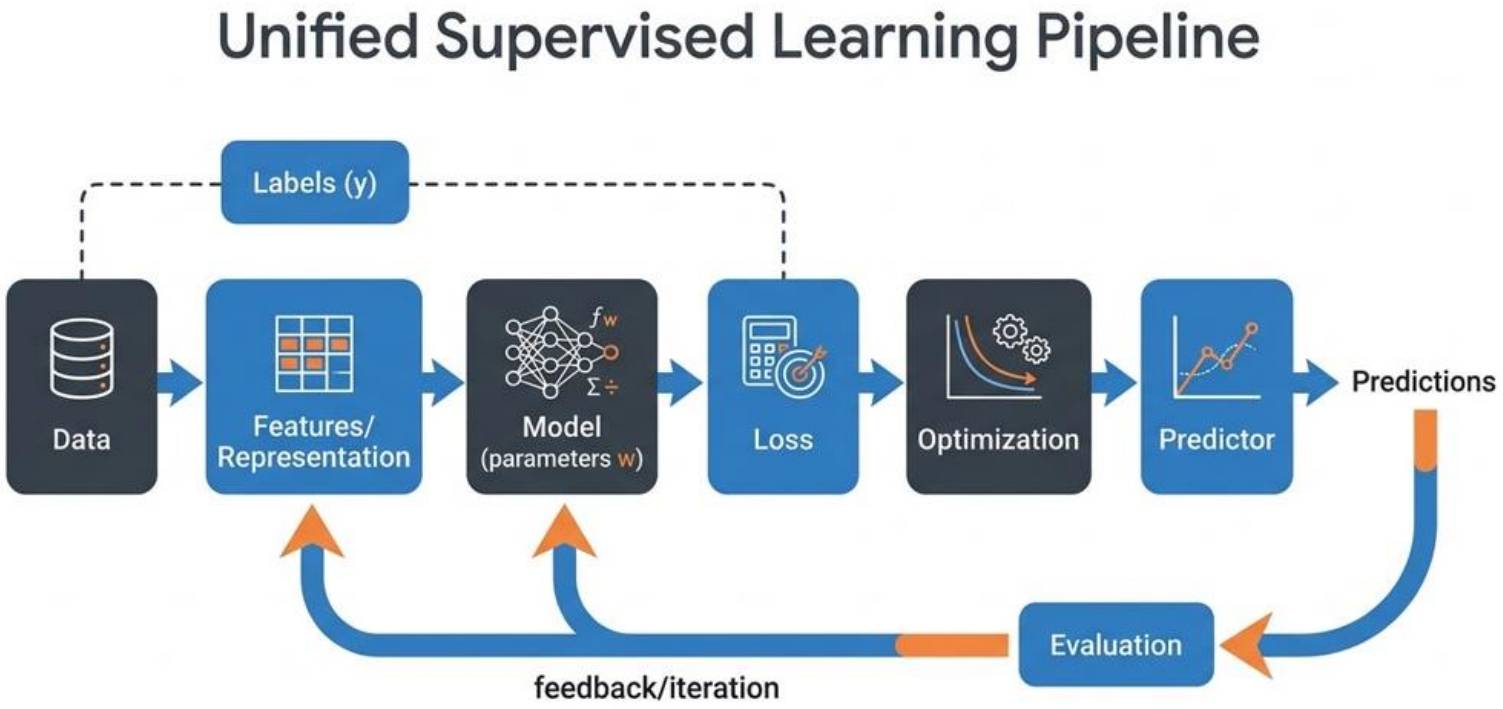
Training and test data are drawn i.i.d. from the same distribution.

- We want
  $\mathcal{R}(w) = \mathbb{E}[\mathcal{L}(f_w(x), y)]$
- We only observe
  $\hat{\mathcal{R}}(w)$



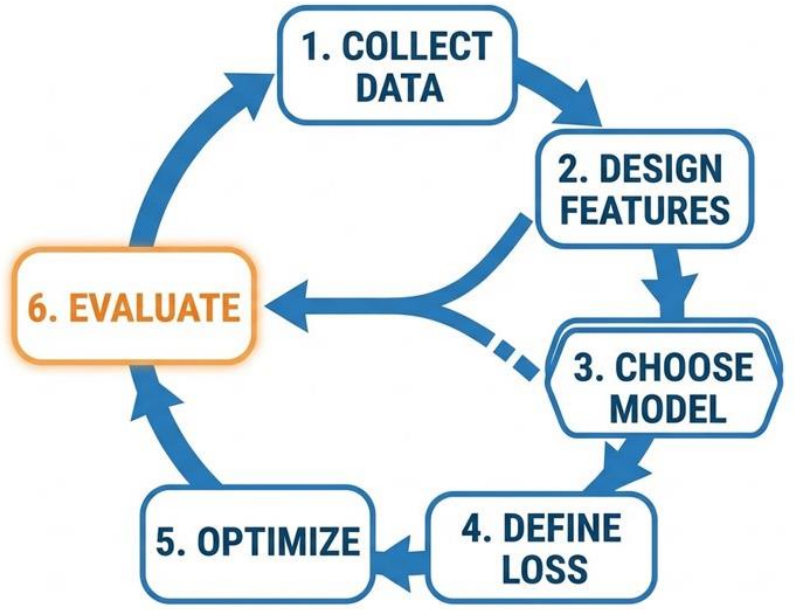
Big Picture

- The 5 core components:
  Data, Features, Model, Loss, Optimization



Pipeline Summary

1. Collect Data
2. Design Features
3. Choose Model
4. Define Loss
5. Optimize
6. Evaluate



Machine learning is an iterative process.

What We Still Need?

- We defined Model & Loss.
- But how do we actually find the best w?

Optimization

Next Lecture

Linear Models

Linear Regression · Logistic Regression