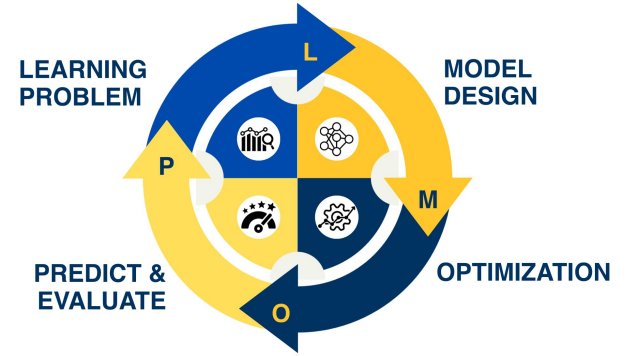


# EECS 189 Discussion 1

## The Machine Learning Lifecycle

Ozan Bayiz, Fall 2026

# The ML Lifecycle



# The ML Lifecycle

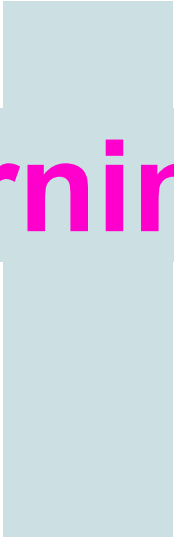
**Learning Problem:** We frame our real-world setting as a learning problem.

**Model Design:** We select & design an appropriate parametric model.

**Optimization:** We adjust our model's parameters to minimize the loss.

**Predict & Evaluate:** We see how well our model performs.



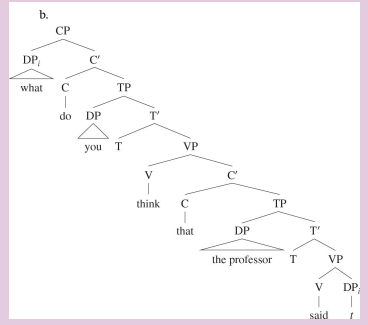


# The Learning Problem

# The Learning Problem

Suppose you want an autonomous system that can perform an intelligent task.

You could manually program the rules.



Or you could try to *learn* the rules from **data**.

# Different Learning Problem Setups

## Supervised Learning



Trying to **learn a relationship** between **observed**  $\{(X,Y)\}$  pairs.

### Classification



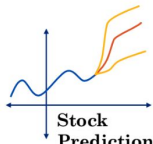
Image Labeling

**X:** Image  
**Y:** {Hot Dog, ...}



**X:** Prompt  
**Y:** Next Word

### Regression



**X:** History  
**Y:** Next Value

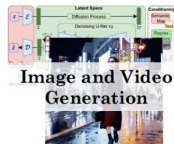


Image and Video Generation

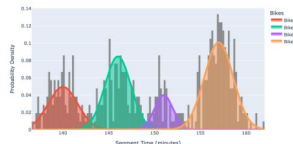
**X:** Prompt+Noise  
**Y:** Pixel Values

## Unsupervised Learning

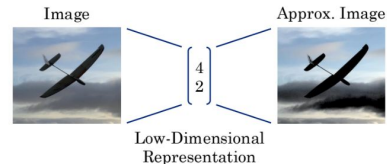


Trying to **model the data** in the **absence** of **explicit labels**.

### Clustering & Density Estimation



### Dimensionality Reduction



## Reinforcement Learning



Learning from **reward signals** often with complex **multi-step** (discrete or continuous) **action** sequences.



**Action:** next move,  
**Reward:** Win/Lose



**Action:** change in joint angles  
**Reward:** Fold quality

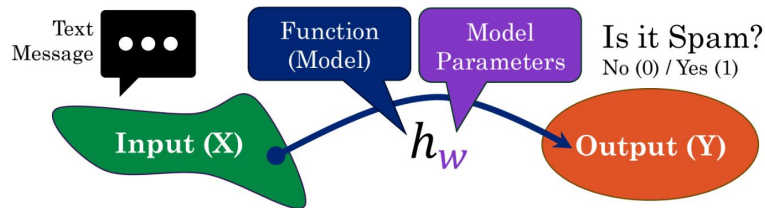


**Action:** next token  
**Reward:** answer quality

# Supervised Learning

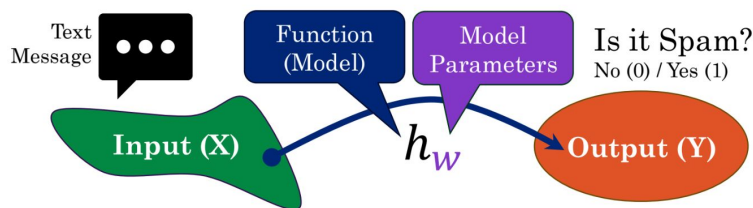


as **Learned Function Approximation**



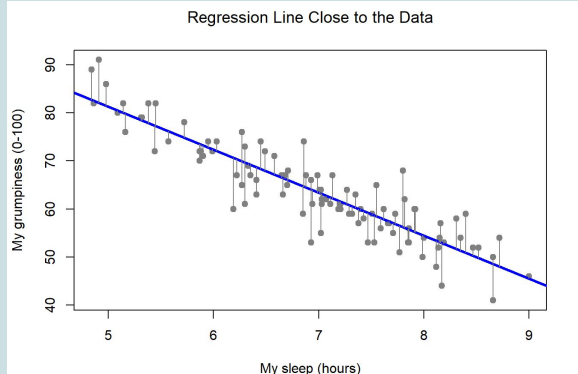
Machine learning becomes the process of “**learning**” the **model parameters** from data or interaction with the world.

as an **Optimization Problem**



$$\text{Goal: } \arg \min_w \mathbb{E}_X [\text{Loss}(h_w(X))]$$

In **Linear Regression**, we approximate the relationship between X and Y as a linear function



high when predictions are bad,  
low when predictions are good.

We obtain the parameters of our linear regression model by **minimizing our loss function.**

# Self-supervised Learning?

Also, humans aren't very good / consistent when labeling data

**Problem:** collecting labeled data is labor-intensive, making it challenging to scale.



“Workers at the headquarters of Ruijin Technology Company in Jiaxian, Henan Province”  
Photo: Yan Cong for The New York Times, 2018

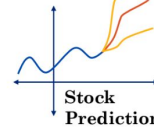
Trying to learn a relationship between observed  $\{(X,Y)\}$  pairs.

## Classification



X: Prompt  
Y: Next Word

## Regression



X: History  
Y: Next Value

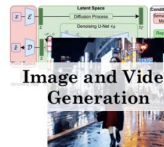
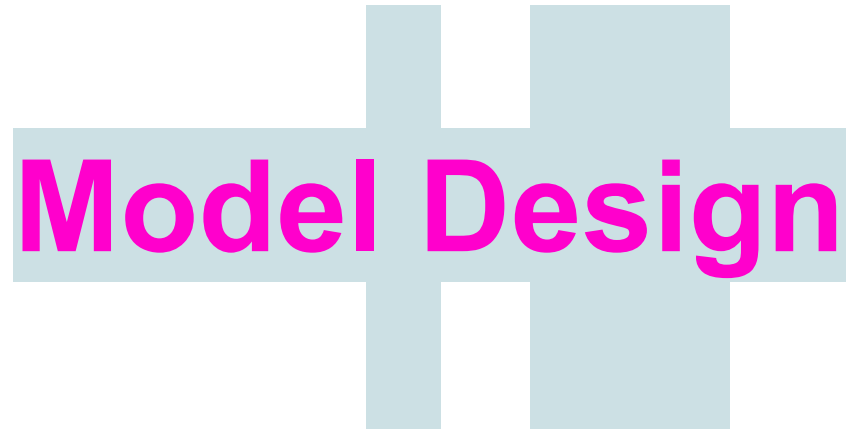


Image and Video Generation

X: Prompt+Noise  
Y: Pixel Values

**Self-supervised learning** constructs labels from data without requiring explicit labeling.

This enables more scalable + robust learning.



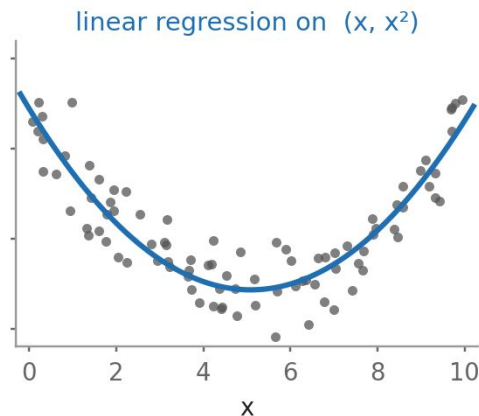
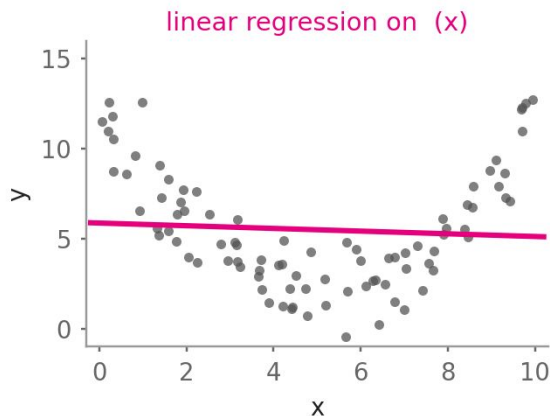
**Model Design**

# Which *features* do we use?

**Features** are the inputs provided to our model.

**Feature engineering** is computing new features from existing features.

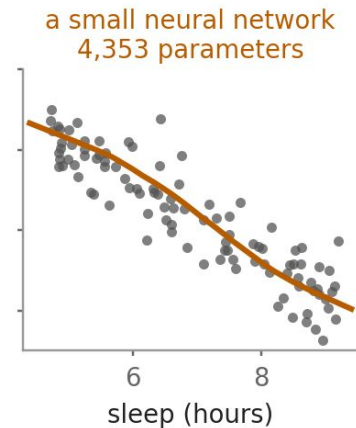
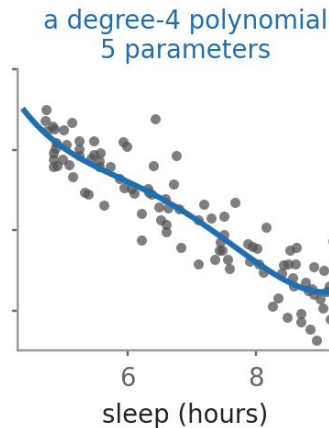
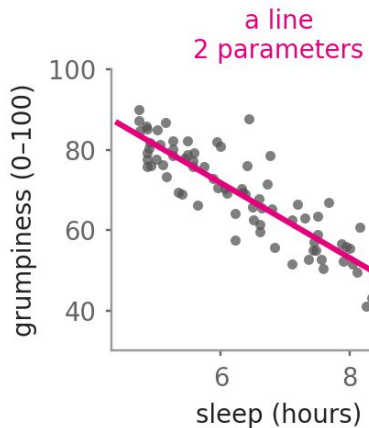
In *classic ML*, features are manually designed.  
In *Deep Learning*, features are **learned from data**.



Using non-linear features  $(x, x^2)$  lets a linear model capture nonlinear relationships!

# Which *model* do we use?

**A model family** is the set of functions you allow (e.g. linear models)



**More parameters = more expressive power**

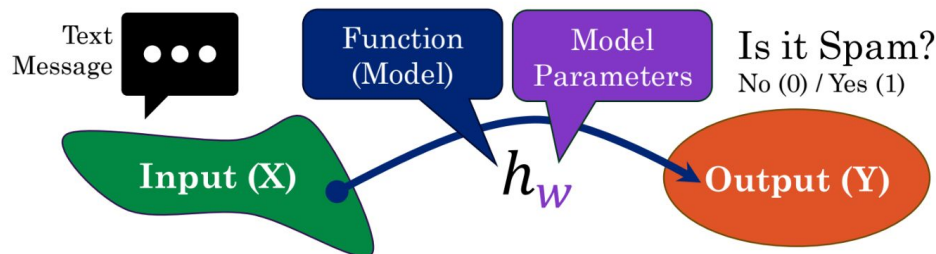


AKA Training

# Goal: Find $w^*$ that minimizes training loss.

as an **Optimization Problem**

18348

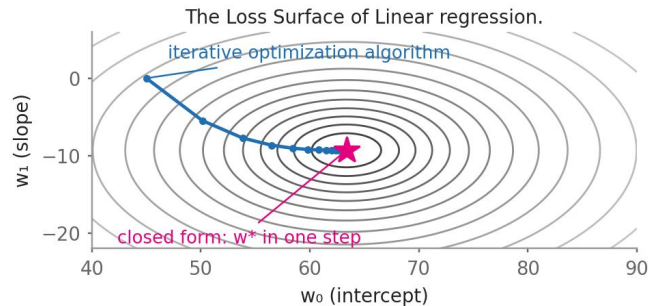
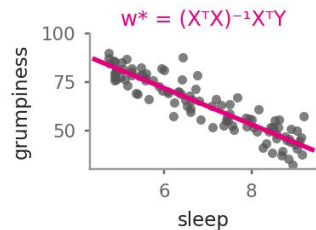


$$\text{Goal: } \arg \min_w \mathbb{E}_X [\text{Loss}(h_w(X))]$$

# Closed-form Solution vs Iterative Optimization

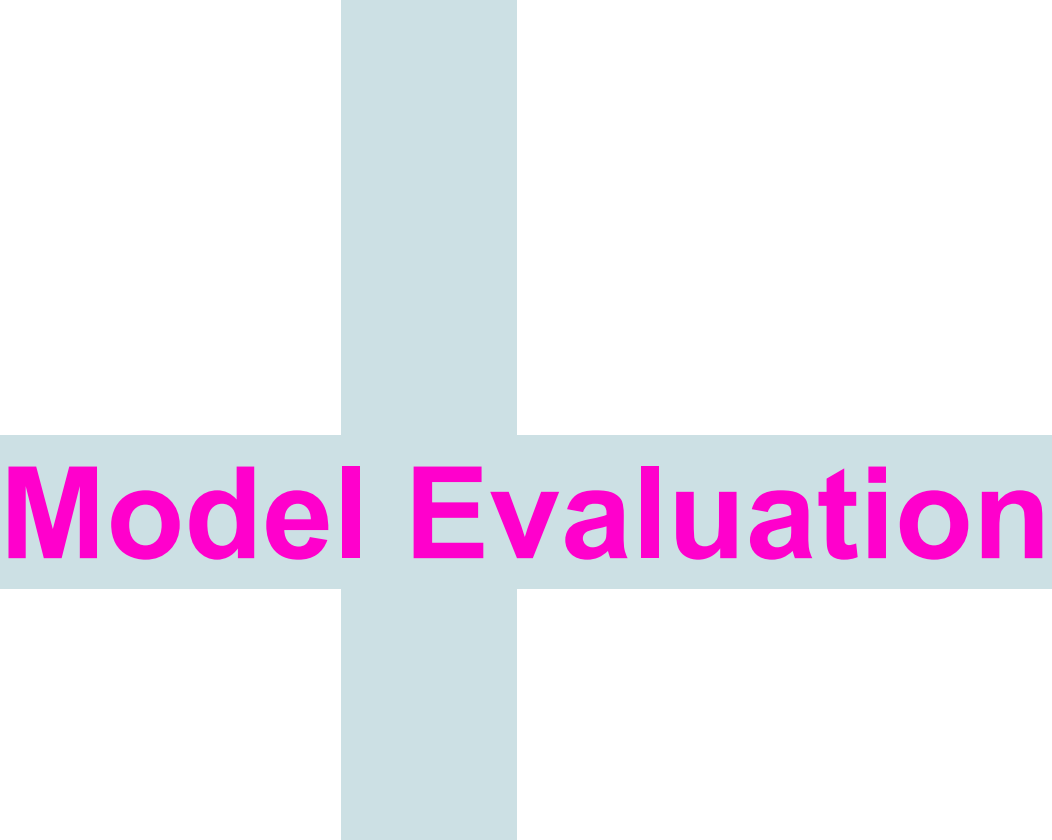
For linear regression with Mean Squared Error loss, we have a formula for the optimal solution.

$$w^* = (X^T X)^{-1} X^T Y$$



For most machine learning models, there is no formula to obtain the model parameters that minimize the loss.

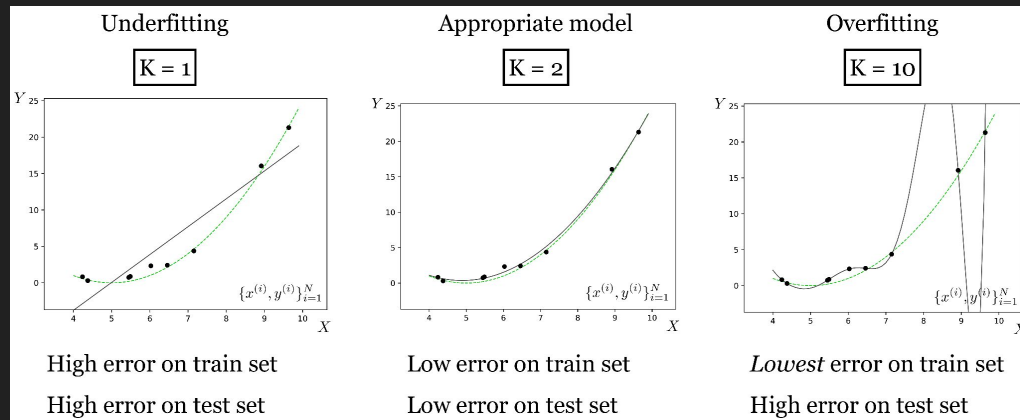
In these cases, we must use an algorithm that **iteratively improves** the model's parameters.



# Model Evaluation

# We want generalization.

Generalization is how well the fitted model does on data it has never seen.



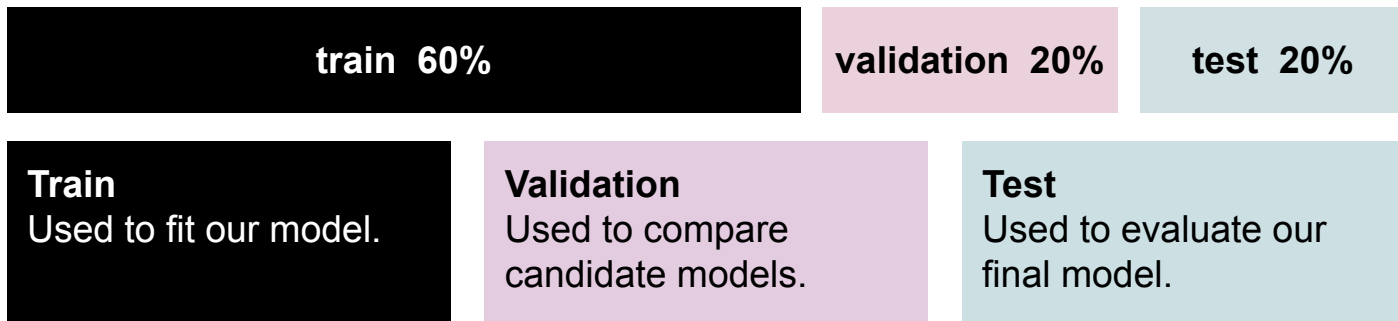
Source: Torralba, Isola & Freeman, Foundations of Computer Vision (MIT Press, 2024), Fig. 11.1 — [visionbook.mit.edu](http://visionbook.mit.edu)

**Underfitting** is when the model lacks the expressive power to capture the true relationship in the data.

**Overfitting** is when the model has too much expressive power, and fits to noise in the dataset.

# Train/Validation/Test Splits

To understand how well our model **generalizes** to unseen data, we can just split up our dataset into chunks that we don't train on.



# The ML Lifecycle

**Learning Problem:** We frame our real-world setting as a learning problem.

**Model Design:** We select & design an appropriate parametric model.

**Optimization:** We adjust our model's parameters to minimize the loss.

**Predict & Evaluate:** We see how well our model performs.

Using our insights to *re-evaluate* our learning problem setup, model design, and optimization method.

```
graph TD; A[Learning Problem] --> B[Model Design]; B --> C[Optimization]; C --> D[Predict & Evaluate]; D --> E[Using our insights to re-evaluate our learning problem setup, model design, and optimization method.]; E --> A;
```



**Thank you!**