Week 1

Lecture 1

What is Machine Learning

---

1. What is Machine Learning?

Definition

We define Machine Learning (ML) as the study of computer algorithms that improve automatically through experience and the use of data.

Formally, we view ML as learning a function:

$$f : \mathcal{X} \to \mathcal{Y}$$

Where:

- $\mathcal{X}$ = input space
- $\mathcal{Y}$ = output space
- f = unknown mapping rule

Our objective in ML is to approximate f using data.

---

2. What is a Task?

We define a **Task** as a process that converts input into output.

$$\text{Task} : x \to y$$

Where:

- $x \in \mathcal{X}$
- $y \in \mathcal{Y}$

Examples

- Weather prediction: Radar Map $\to$ Rain / No Rain
- Face detection: Image $\to$ Face / No Face

- Password verification: Password $\rightarrow$ Authentication

---

3. Task Hierarchy

We organize tasks into four abstraction levels.

Level 1: Manual Labour

Input $\rightarrow$ Human $\rightarrow$ Output
We directly compute:

$$y = f(x)$$

Level 2 & 3: Tool Usage / Programming

We explicitly write:

$$y = f(x)$$

Structure:
Input $\rightarrow$ Software $\rightarrow$ Output
In this setting, we fully know f.

Level 4: Machine Learning

We do not explicitly specify f.
Instead, we provide labeled data:

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

The system learns an approximation:

$$\hat{f}(x) \approx f(x)$$

Structure:
Data + Model Design $\rightarrow \hat{f}$
Then:
$x \rightarrow \hat{f}(x) \rightarrow y$

Key Idea: We learn the function from data.

4. When Should We Use Machine Learning?

Programming / Human Labour Fails When

- Scale, speed, or cost becomes too high
- We cannot express rules mathematically
- The exact rule f is unknown

Formally, we cannot define:

$$y = f(x)$$

Machine Learning Can Succeed If

- There exists some function f
- We possess sufficient data
- There is underlying structure

We assume:

$$\exists f \text{ such that } y = f(x)$$

We approximate:

$$\hat{f} = \arg\min_{g \in \mathcal{H}} \mathcal{L}(g(x), y)$$

Where:

- $\mathcal{H}$ = hypothesis space
- $\mathcal{L}$ = loss function

---

5. Case Study: Password Verification

Task:
Password $\rightarrow$ Authentication
Rule:

$$f(x) = \begin{cases} 1 & \text{if } x = x_{\text{stored}} \\ 0 & \text{otherwise} \end{cases}$$

Conclusion:

- Programming suffices
- ML is unnecessary

---

6. Case Study: Face Detection

Task:

$$\text{Image} \rightarrow \{0, 1\}$$

Where:

- 1 = Face
- 0 = Not Face

We cannot explicitly write:

$$f(\text{pixels}) = \text{Face}$$

Instead, we collect labeled data:

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

And we learn:

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

Classification setting:

$$y \in \{0, 1\}$$

This constitutes a binary classification problem.

---

7. Case Study: Weather Prediction

Task:

Radar Map $\rightarrow$ Rain / Shine
There exists some unknown function:

$$y = f(x)$$

But:

- We do not know f
- We cannot code f
- We cannot compute f manually

We assume:

$$\exists f$$

And we approximate:

$$\hat{f}(x) \approx f(x)$$

---

8. Real-World ML Applications

Spam Classification

Email $\rightarrow \{\text{Spam}, \text{Not Spam}\}$
We learn:

$$\hat{f}(\text{email features})$$

Recommender Systems

User History $\rightarrow$ Recommended Item
We predict:

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

Smart Assistants

Audio waveform $\rightarrow$ Text
Text $\rightarrow$ Command
In both cases, we learn mappings from data.

Robotics

Environment State $\rightarrow$ Action
We learn a policy:

$$\pi(s) = a$$

Where:

- s = state
- a = action

---

9. Core Mathematical Insight

We assume:

- There exists a function f
- We cannot explicitly specify f
- Data encodes information about f

Thus we:

1. Define hypothesis space $\mathcal{H}$
2. Choose loss function $\mathcal{L}$

We learn:

$$\hat{f} = \arg\min_{g \in \mathcal{H}} \mathcal{L}(g(x), y)$$

This represents the fundamental ML framework.

****

Lecture 2

Data, Models and ML Tasks

---

1. What is Data?

In machine learning, we interpret data as a collection of vectors.
Formally, we represent a dataset as:

$$\mathcal{D} = \{x_1, x_2, \ldots, x_n\}$$

Where:

- $x_i \in \mathbb{R}^d$
- d = number of features (dimension)

We express each data point as:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$$

Example: Housing Data

We may represent each house as:

$$x = (\#\text{rooms}, \text{area}, \text{distance to metro}, \text{price})$$

Example dataset:

$$x_1 = (3, 9, 1.9, 5.0)$$

$$x_2 = (2, 7, 2.1, 3.2)$$

$$x_3 = (4, 12, 2.8, 6.6)$$

and so on.

Metadata

We define metadata as information about the data.
For example:

- Feature 1: Number of rooms
- Feature 2: Area (in 100 sq. ft.)
- Feature 3: Distance to metro (km)
- Feature 4: Price (in 10 lakhs)

We note:

- A computer requires only consistency in representation.
- We require metadata for interpretability.

---

2. What is a Model?

We define a model as a mathematical simplification of reality.
Formally:
A model is a function:

$$f : \mathcal{X} \to \mathcal{Y}$$

that approximates real-world behavior.

Scientific Examples of Models

- Ideal Gas Law:

$$PV = nRT$$

- Newton's Gravitational Law:

$$F = \frac{Gm_1m_2}{r^2}$$

- Moore's Law (trend model)
- Economic models (e.g., Cobb–Douglas)

The key philosophical insight is:

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

We understand that models are approximations rather than exact reality.

---

3. Types of Models in Machine Learning

We distinguish two major types:

- Predictive Models
- Probabilistic Models

---

4. Predictive Models

Predictive models define a mapping:

$$x \to y$$

We use them to generate predictions on unseen data.

Two major types:

- Regression
- Classification

4.1 Regression Model

We use regression when the output is real-valued.

$$y \in \mathbb{R}$$

Example: We predict house price.

A simple model form:

$$\text{Price} = a \cdot \text{Area} - \text{Distance}$$

More generally:

$$y = a \cdot x_1 + b \cdot x_2 + c \cdot x_3$$

Where:

- a, b, c = parameters
- x_1, x_2, x_3 = features

Regression produces continuous outputs.

4.2 Classification Model

We use classification when the output is discrete.

$$y \in \{0, 1\}$$

More generally:

$$y \in \{1, 2, \ldots, K\}$$

Example:

We predict whether a house lies within 2 km of the metro.
Possible outputs:

- Close
- Far
  Example model:

$$\text{Answer} = \begin{cases} \text{Close} & \text{if } 2 \cdot \text{Rooms} - \text{Price} < 1 \\ \text{Far} & \text{otherwise} \end{cases}$$

  Classification assigns category labels.

---

5. Probabilistic Models

Probabilistic models evaluate likelihood.
They do not directly output a deterministic value.
They compute:

$$P(x)$$

or

$$P(y \mid x)$$

Examples:

- We compute the probability that a randomly chosen person lies at latitude-longitude $(25^\circ N, 30^\circ E)$.
- We compute the probability that a tweet was generated by a specific individual.

Probabilistic models score configurations of reality.

---

6. Learning Algorithms

Learning algorithms perform the transformation:

$$\text{Data} \to \text{Model}$$

They select from a family of models sharing structure but differing in parameters.

Example structure:

$$\text{Price} = a \cdot (\text{Area}) + b \cdot (\#\text{Rooms}) + c \cdot (\text{Distance})$$

Parameters:

$$a, b, c$$

The learning algorithm chooses parameter values that best fit the data.

Formally:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(f_\theta(x), y)$$

Where:

- θ = model parameters
- $\mathcal{L}$ = loss function

---

7. Machine Learning Pipeline (Revisited)

We do not directly write the final model.
Instead:

1. We specify the model structure.
2. We provide data.
3. The learning algorithm estimates parameters.
4. We obtain the final model.

Pipeline:

$$\text{Human} + \text{Data} \rightarrow \text{Learning Algorithm} \rightarrow \text{Model}$$

Then:

$$\text{Input} \rightarrow \text{Model} \rightarrow \text{Output}$$

Core Idea of This Lecture

- We interpret data as a collection of vectors.
- We define a model as a mathematical abstraction.
- We classify predictive models into regression and classification.
- We understand probabilistic models as likelihood evaluators.
- We view the learning algorithm as the mechanism that converts data into a model.

****

Lecture 3

Supervised Learning – Regression

---

1. Supervised Learning

We interpret supervised learning as a form of curve fitting.
Given training data:

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

Where:

- $x_i \in \mathbb{R}^d$
- $y_i \in \mathcal{Y}$

Our objective is to construct a model

$$f : \mathbb{R}^d \to \mathcal{Y}$$

such that:

$$f(x_i) \text{ is close to } y_i$$

---

2. Notation

We use:

- $\mathbb{R}$ to denote real numbers
- $\mathbb{R}_+$ to denote positive real numbers
- $\mathbb{R}^d$ to denote d-dimensional real vectors

For a vector:

- $x = (x_1, x_2, \ldots, x_d)$
- x_j denotes the j^{th} coordinate
- $\|x\|$ denotes the Euclidean norm

We define:

$$\|x\|^2 = \sum_{j=1}^{d} x_j^2$$

$$\|x\| = \sqrt{\sum_{j=1}^{d} x_j^2}$$

For a collection of vectors:

$$x^1, x^2, \ldots, x^n$$

We interpret:

- x_j^i as the j^{th} coordinate of the i^{th} vector

We define the indicator function as:

$$\mathbf{1}(\text{predicate}) = \begin{cases} 1 & \text{if true} \\ 0 & \text{if false} \end{cases}$$

Example:

$$\mathbf{1}(2 \text{ is even}) = 1$$

$$\mathbf{1}(2 \text{ is odd}) = 0$$

---

3. Regression

We apply regression when:

$$y_i \in \mathbb{R}$$

Example:
We predict house price from:

- Rooms
- Area
- Distance to metro

The training data is:

$$\{(x^1, y^1), (x^2, y^2), \ldots, (x^n, y^n)\}$$

Where:

- $x^i \in \mathbb{R}^d$
- $y^i \in \mathbb{R}$

We construct a model:

$$f : \mathbb{R}^d \to \mathbb{R}$$

4. Loss Function (Squared Loss)

We define the loss of a model f as:

$$\mathcal{L}(f) = \frac{1}{n} \sum_{i=1}^{n} \left(f(x^i) - y^i \right)^2$$

We observe:

- The loss is always non-negative
- $\mathcal{L}(f) = 0$ if and only if $f(x^i) = y^i$ for all i

The learning algorithm seeks to:

$$\min_f \mathcal{L}(f)$$

---

5. Linear Parameterization

A commonly used regression model is:

$$f(x) = w^T x + b$$

In expanded form:

$$f(x) = \sum_{j=1}^{d} w_j x_j + b$$

The parameters are:

- $w = (w_1, w_2, \ldots, w_d)$
- b = bias

Example (House price):

$$f(x) = w_1 \cdot \text{Rooms} + w_2 \cdot \text{Area} + w_3 \cdot \text{Distance} + b$$

The learning algorithm determines:

$$w, b$$

that minimize the squared loss.

---

6. Regression Illustration 1 (1D Example)

Let $d = 1$.
Training data:

$$x = [1, 2, 3, 6, 7]$$
$$y = [2.1, 3.9, 6.2, 11.5, 13.9]$$

We consider two candidate models.

Model 1:

$$f(x) = 2x$$

Model 2:

$$g(x) = x + 3$$

Predictions:

For f:

$$[2, 4, 6, 12, 14]$$

For g:

$$[4, 5, 6, 9, 10]$$

We compute the loss of f:

$$\frac{1}{5}\left[(2 - 2.1)^2 + (4 - 3.9)^2 + (6 - 6.2)^2 + (12 - 11.5)^2 + (14 - 13.9)^2\right]$$

We compute the loss of g:

$$\frac{1}{5}\left[(4 - 2.1)^2 + (5 - 3.9)^2 + (6 - 6.2)^2 + (9 - 11.5)^2 + (10 - 13.9)^2\right]$$

We observe:

$$\mathcal{L}(f) < \mathcal{L}(g)$$

Thus we prefer f.

---

7. Regression Illustration 2 (House Example)

Data:

Rooms	Area	Distance	Price
3	9	1.9	5.0
2	7	2.1	3.2
4	12	2.8	6.6
5	16	0.9	9.8
5	15	3.1	8.5
4	11	1.6	6.9

We consider two models.

Model f:

$$f(x) = 2 \cdot \text{Rooms} - 0.5 \cdot \text{Distance}$$

Model g:

$$g(x) = \text{Rooms} + 2 \cdot \text{Distance}$$

We evaluate predictions for all training points.
Then we compute:

$$\mathcal{L}(f) = \frac{1}{n} \sum_{i=1}^{n} (f(x^i) - y^i)^2$$

$$\mathcal{L}(g) = \frac{1}{n} \sum_{i=1}^{n} (g(x^i) - y^i)^2$$

We observe:

$$\mathcal{L}(f) < \mathcal{L}(g)$$

Hence we conclude that f is the better model for this dataset.

Core Idea of Regression

- We fit a function to data.
- We select a model family (for example, linear models).
- We define a loss function (for example, squared loss).
- We determine parameters that minimize the loss.

We therefore interpret supervised learning as curve fitting under a specified loss function.

****

Lecture 4

Supervised Learning – Classification

---

1. Classification Problem Setup

In classification, we focus on predicting discrete labels.

Example:
We predict whether:

$$\text{Rooms} > 3$$

from:

- Area
- Price

Training Data

We are given:

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

Where:

$$x_i \in \mathbb{R}^d$$

$$y_i \in \{+1, -1\}$$

The learning algorithm produces:

$$f : \mathbb{R}^d \to \{+1, -1\}$$

---

2. Classification Loss

In the ideal situation:

$$f(x_i) = y_i \quad \forall i$$

However, this condition may not always be satisfied.
We define the classification loss as:

$$\mathcal{L}(f) = \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(f(x_i) \neq y_i)$$

Where:

$$\mathbf{1}(A) = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

This quantity represents:

> The fraction of misclassified training examples.

---

3. Linear Classifier (Linear Separator)

A commonly used parameterization is:

$$f(x) = \text{sign}(w^\top x + b)$$

Where:

- $w \in \mathbb{R}^d$
- $b \in \mathbb{R}$

This formulation defines a hyperplane:

$$w^\top x + b = 0$$

Points on one side are assigned +1.
Points on the other side are assigned -1.

---

4. Classification Illustration 1

Dataset (d = 2)

Training points:

$$(0,0), (1,0), (0,1)$$

Label:

$$+1$$

and

$$(4,4), (3,4), (4,3)$$

Label:

$$-1$$

Model 1

$$f(x) = \text{sign}(2 - x_1)$$

Evaluating this model on the training data:

All 6 points are correctly classified.
Hence:

$$\mathcal{L}(f) = 0$$

Model 2

$$g(x) = \text{sign}(x_1 - 2x_2)$$

One point is incorrectly classified.
Thus:

$$\mathcal{L}(g) = \frac{1}{6}$$

Since:

$$0 < \frac{1}{6}$$

We prefer:

$$f$$

---

5. Classification Illustration 2 (House Example)

We encode:

$$\text{Rooms} \leq 3 \rightarrow -1$$

$$\text{Rooms} > 3 \rightarrow +1$$

We examine three models:

Model f

$$f(x) = \text{sign}(\text{Area} - 10)$$

Loss:

$$\mathcal{L}(f) = 0$$

Model g

$$g(x) = \text{sign}(\text{Price} - 6)$$

Loss:

$$\mathcal{L}(g) = 0$$

Model h

$$h(x) = \text{sign}(\text{Price} - 9)$$

Loss:

$$\mathcal{L}(h) = \frac{3}{6}$$

Therefore:

- f and g perfectly classify the training data.
- h does not.

---

6. Important: Do NOT Evaluate on Training Data

Consider the following model:

$$f(x) = \begin{cases} +1 & \text{if } x = (0,0) \\ +1 & \text{if } x = (1,0) \\ +1 & \text{if } x = (0,1) \\ -1 & \text{otherwise} \end{cases}$$

On the training data:

$$\mathcal{L}(f) = 0$$

However, this model is clearly inadequate.
It memorizes the training points.
It will not generalize to unseen inputs.
This phenomenon is known as **overfitting**.

---

7. Train / Test Split

The learning procedure is structured as follows:

1. We use training data to learn the model.
2. We use test data to evaluate the model.

We must not evaluate performance on the training data.
Why?
Because:
Training loss provides an overly optimistic estimate of performance.

The true goal of machine learning is:
To perform well on unseen data.

---

8. Validation Data (Model Selection)

The learning algorithm identifies the best model **within a chosen family**.
Example model family:

$$f(x) = w_1(\text{rooms}) + w_2(\text{area}) + w_3(\text{distance}) + b$$

But we must ask:
Why select this family?
Why not:

$$f(x) = a \cdot \frac{\text{area}}{\text{rooms}} + b \cdot \text{distance}^2 + c$$

The process of choosing the appropriate model class is called:

Model Selection

We typically divide data as follows:

- Training data → learn parameters
- Validation data → choose model class
- Test data → final evaluation

Core Ideas of This Lecture

- We use classification to predict discrete labels.
- We measure performance using misclassification loss.
- A linear classifier has the form $\text{sign}(w^\top x + b)$.
- We must not evaluate on training data.
- We use test data for evaluation.
- We use validation data for model selection.

****

Lecture 5

Unsupervised Learning – Dimensionality Reduction

---

1. Introduction to Unsupervised Learning

In the previous lectures, we studied supervised learning and analyzed the two central tasks: regression and classification.

In this lecture, we begin the study of the unsupervised learning paradigm, focusing on its primary tasks:

- Dimensionality Reduction
- Density Estimation

Unlike supervised learning, where we had clearly defined objectives and explicit loss functions tied to labeled data, unsupervised learning is inherently less structured.

We no longer observe labeled pairs (x_i, y_i). Instead, we are given only data points:

$$\{x^1, x^2, \ldots, x^n\}$$

where

$$x^i \in \mathbb{R}^d$$

We view unsupervised learning as the task of **understanding data**.

More concretely, we attempt to build models that:

- Compress data
- Explain structure in data
- Group similar data points

---

2. Motivation: Why Unsupervised Learning?

We typically do not use unsupervised learning as an end goal.
Instead, we use it as a preprocessing stage that supports other tasks or aids human interpretation.

Example: Tweet Grouping

Suppose we collect one million tweets about a product.
It is impractical to manually examine all tweets individually.
Instead, we may wish to group these tweets into a small number of coherent clusters, say 10 groups.
We can then interpret each group:

- Users sharing personal experiences
- Promotional posts
- Brand collaborations
- Sponsored content

The unsupervised algorithm only performs grouping.
The semantic interpretation of these groups is performed by humans.
Thus, unsupervised learning extracts structure, and human reasoning assigns meaning.

---

3. Dimensionality Reduction

We now study the first major unsupervised task: **Dimensionality Reduction**.

Goal

We aim to compress high-dimensional data into a lower-dimensional representation while preserving essential information.

For example:
Suppose we measure gene expression levels for:

- 10^6 genes
- 10^6 individuals

This produces a matrix of size:

$$10^6 \times 10^6$$

Storing or transmitting such data directly is impractical.
Instead, we may wish to represent each individual using only 100 numbers.
Dimensionality reduction enables this compression.
We summarize its purpose as:

> Compression and simplification.

---

4. Formal Setup

We are given data:

$$\{x^1, x^2, \ldots, x^n\}$$

where

$$x^i \in \mathbb{R}^d$$

We aim to learn two functions:

Encoder

$$f : \mathbb{R}^d \to \mathbb{R}^{d'}$$

where typically:

$$d' \ll d$$

The encoder compresses a d-dimensional vector into a d'-dimensional representation.

Decoder

$$g : \mathbb{R}^{d'} \to \mathbb{R}^d$$

The decoder reconstructs the original vector from its compressed form.

Objective

We want:

$$g(f(x^i)) \approx x^i$$

for all training points.
We measure reconstruction error using squared Euclidean norm:

$$\left\| g(f(x^i)) - x^i \right\|^2$$

The overall loss is:

$$\mathcal{L}(f, g) = \frac{1}{n} \sum_{i=1}^{n} \left\| g(f(x^i)) - x^i \right\|^2$$

Our goal is to find encoder-decoder pair (f, g) that minimizes this loss.

---

5. Illustration: Simple Example

Let:

$$d = 2, \quad d' = 1, \quad n = 4$$

Data points:

$$x^1 = (1, 0.8)$$

$$x^2 = (2, 2.2)$$

$$x^3 = (3, 3.2)$$

$$x^4 = (4, 3.8)$$

Encoder-Decoder Pair 1

Encoder:

$$f(x) = x_1 - x_2$$

Decoder:

$$g(u) = (u, u)$$

Encoder outputs:

$$[0.2, -0.2, -0.2, 0.2]$$

Decoder reconstructions:

$$(0.2, 0.2), (-0.2, -0.2), (-0.2, -0.2), (0.2, 0.2)$$

These reconstructions are far from original points.
Thus, reconstruction error is large.
This encoder-decoder pair performs poorly.

Encoder-Decoder Pair 2

Encoder:

$$\tilde{f}(x) = \frac{x_1 + x_2}{2}$$

Decoder:

$$\tilde{g}(u) = (u, u)$$

Encoder outputs:

$$[0.9, 2.1, 3.1, 3.9]$$

Decoder reconstructions:

$$(0.9, 0.9), (2.1, 2.1), (3.1, 3.1), (3.9, 3.9)$$

These reconstructions are much closer to the original data.
Thus:

$$\mathcal{L}(\tilde{f}, \tilde{g}) < \mathcal{L}(f, g)$$

The second pair performs better.

---

6. Conceptual Understanding

Dimensionality reduction algorithms:

- Choose encoder f
- Choose decoder g
- Minimize reconstruction loss

In real applications, we do not restrict ourselves to two candidate pairs.
Instead, we search over large (often infinite) families of functions to find optimal compression.

Core Ideas of Lecture 5

- Unsupervised learning works without labels.
- We aim to understand structure in data.
- Dimensionality reduction compresses high-dimensional vectors.
- We learn encoder-decoder pairs.
- We minimize reconstruction error:

$$\frac{1}{n}\sum_{i=1}^{n}\left\|g(f(x^i))-x^i\right\|^2$$

> Dimensionality reduction is the foundation for many modern representation learning techniques.

****

Lecture 6

Unsupervised Learning – Density Estimation

---

1. Density Estimation Overview

With dimensionality reduction completed, we now move to the second major unsupervised learning problem: **density estimation**.

In density estimation, the output of the learning algorithm is a **probabilistic model**.

We recall that a probabilistic model assigns a score to different configurations of reality.

For example, suppose we want to construct a model capable of generating tweets that resemble tweets from a specific account. Assuming tweets are generated independently at

random from that account, we attempt to construct a robot account that can generate similar tweets.

A density estimation model enables exactly this: it assigns high probability to sentences that resemble the original source and low probability to unlikely sentences.

---

2. Formal Setup

Data

We are given:

$$\{x^1, x^2, \ldots, x^n\}$$

Where:

$$x^i \in \mathbb{R}^d$$

Each x^i represents one data instance (e.g., a tweet represented as a vector).

Model

The goal is to learn a probability mapping:

$$P : \mathbb{R}^d \to \mathbb{R}_+$$

such that:

1. $P(x) \geq 0$ for all x
2. The probabilities sum (or integrate) to 1

In discrete form (e.g., tweets of length 128 from an alphabet of size 26):

$$\sum_{x \in 26^{128}} P(x) = 1$$

Goal

We want:

- $P(x)$ to be large if x belongs to the data

- $P(x)$ to be small otherwise

---

3. Loss Function – Negative Log Likelihood

We define the loss as:

$$\mathcal{L}(P) = \frac{1}{n} \sum_{i=1}^{n} -\log \left(P(x^i) \right)$$

This is called the **negative log likelihood**.

Interpretation

- If $P(x^i)$ is large, then $-\log P(x^i)$ is small.
- If $P(x^i)$ is small, then $-\log P(x^i)$ is large.
- If $P(x^i) = 0$, then loss becomes infinite.

Therefore, we aim to:

$$\min_{P} \mathcal{L}(P)$$

---

4. Illustration 1 (1D Example)

Let:

$$d = 1$$

Suppose we are given 4 data points:

$$x^1 = 2.3$$

$$x^2 = 2.7$$

$$x^3 = 4.6$$

$$x^4 = 4.9$$

Candidate Models

Model P_1

$$P_1(x) = \begin{cases} \frac{1}{10} & \text{if } x \in [0, 10] \\ 0 & \text{otherwise} \end{cases}$$

Model P_2

$$P_2(x) = \begin{cases} \frac{1}{5} & \text{if } x \in [0, 5] \\ 0 & \text{otherwise} \end{cases}$$

Model P_3

$$P_3(x) = \begin{cases} \frac{1}{5} & \text{if } x \in [3, 8] \\ 0 & \text{otherwise} \end{cases}$$

Evaluate Likelihoods

For data points:

$$2.3,\ 2.7,\ 4.6,\ 4.9$$

- Under P_1: each gets $\frac{1}{10}$
- Under P_2: each gets $\frac{1}{5}$
- Under P_3: first two get 0, last two get $\frac{1}{5}$

Compute Loss

Loss of P_1

$$-\log\left(\frac{1}{10}\right) - \log\left(\frac{1}{10}\right) - \log\left(\frac{1}{10}\right) - \log\left(\frac{1}{10}\right)$$

Loss of P_2

$$-\log\left(\frac{1}{5}\right) - \log\left(\frac{1}{5}\right) - \log\left(\frac{1}{5}\right) - \log\left(\frac{1}{5}\right)$$

Loss of P_3

Since two probabilities are zero:

$$-\log(0) = \infty$$

Therefore:

$$\mathcal{L}(P_3) = \infty$$

Conclusion

We observe:

$$\mathcal{L}(P_2) < \mathcal{L}(P_1) < \mathcal{L}(P_3)$$

Thus, P_2 is the best among these three models.

---

5. Illustration 2 (2D Gaussian Mixture Intuition)

Now suppose:

$$d = 2$$

We are given 9 data points in $\mathbb{R}^2$.
We consider two Gaussian mixture models:

- Model P_1 with three centers aligned with visible clusters in the data.
- Model P_2 with three centers placed away from the visible clusters.

We compute the negative log likelihood for both.
Since P_1 assigns higher probability to regions where data clusters appear, it yields smaller negative log likelihood.
Thus:

$$\mathcal{L}(P_1) < \mathcal{L}(P_2)$$

Therefore, P_1 is the better density model.

---

6. Core Ideas of Density Estimation

- We are given unlabeled data.
- We learn a probability distribution over the data space.
- We ensure probabilities sum (or integrate) to 1.
- We maximize likelihood (equivalently minimize negative log likelihood).

- The learned model can generate or score new data.

---

7. Wrap-up for Week 1

We have now covered the two major unsupervised learning problems:

1. Dimensionality Reduction
2. Density Estimation

Across Week 1, we have introduced:

- Supervised learning (regression and classification)
- Unsupervised learning (dimensionality reduction and density estimation)
- Loss functions for each paradigm
- The general principle of learning as optimization over model families

In later lectures, we will develop concrete algorithms that search over infinite model classes and compute optimal solutions efficiently.

`**********`
