AMATH 445

Scientific Machine Learning

Lecture Notes

Winter 2026

Contents

Lecture 1

1.1 What is Machine Learning

Definition 1.1 *(Machine Learning).* Machine learning is decision making based on data. The algorithm improves its performance on tasks based on experience (data).

Note 1.2 *(Traditional Algorithm Workflow).*

$$\left.\begin{array}{l}\text{data/info}\\ \text{recipe}\end{array}\right\} \rightarrow \text{traditional algorithm} \rightarrow \text{output}$$

Note 1.3 *(Machine Learning Workflow).*

$$\left.\begin{array}{l}\text{data/info}\\ \text{output}\end{array}\right\} \rightarrow \text{ML} \rightarrow \text{recipe}$$

What, When and Why

What does ML do really well:

- Prediction: Forecasting unknown outcome (temperature).
- Representation: Finding structure in data (clustering).
- Decision making: Choosing the optimal action.
- Generation: ChatGPT, new data, image generation.

When

- Complex patterns: When relationships in data are too complex for traditional programming and we are not able to find the patterns ourselves.
- Large datasets: When there is an abundance of data that can be leveraged for learning.

Why

- It can generalize well to unseen data, making accurate predictions or decisions based on learned patterns (not memorize only).

Important: ML and Data Quality

Machine Learning is only as good as the data it is trained on. Poor quality or biased data can lead to inaccurate or unfair outcomes.

1.2 Factors Driving Popularity of ML

There are several factors driving the popularity of ML:

1. Data availability.
2. Parallel computing (GPU).
3. Algorithm & infrastructure maturity.
    - CNN, RNN.
    - Backward differentiation.

1.3 Mechanics of ML

The mechanics of ML is as follows

1. Data representation $\rightarrow$ turn things into feature
2. Pattern discovery $\rightarrow$ structure emerges from randomness
3. Pick a model
4. Training – optimization
5. Generalize $\rightarrow$ good model works on new data
6. Evolution $\rightarrow$ test on new data (unseen)

1.4 Data

1.4.1 Data Splitting

A common convention for data splitting is 70% training, 15% validation, and 15% test.

Important

TEST DATA is NEVER used in training!

1.4.2 Label Availability

This table shows the availability of features and outputs during the modeling phases versus real-world deployment.

	$F_1 \ldots F_n$	output
training	known	known
validation	known	known
test	known	known
new	known	unknown

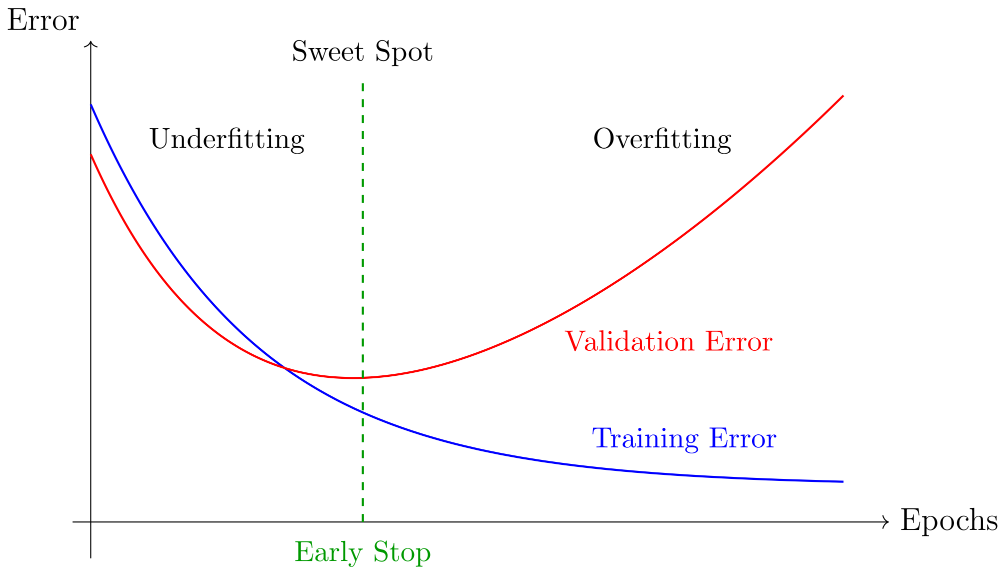
1.5 Types of Error

There are three types of error to track during model development:

- Training error: error on the training set. It measures how well the model fits the training data.
- Validation error: error on the validation set. It guides hyperparameter tuning and model selection.
- Test error: error on the test set. It provides an unbiased estimate of model performance on unseen data.

1.6 Training Dynamics

As training progresses, training and validation errors evolve differently. The figure below illustrates the typical behavior.



Note 1.4. Three key behaviors emerge from the training dynamics:

- **Underfitting**: Both training and validation errors are high. The model is too simple to capture the underlying patterns.
- **Overfitting**: Training error continues to decrease while validation error starts to increase. The model memorizes the training data but fails to generalize.
- **Early Stopping**: Stop training when the validation error begins to increase. This achieves the best generalization by stopping at the sweet spot.

1.6.1 Underfitting

A model underfits when it is too simple (high bias) to capture the underlying structure of the data. Both training and validation errors remain high. Remedies include:

- Increase model complexity.
- Add more or better features.
- Obtain more or better training data.

1.6.2 Overfitting

A model overfits when it learns the training data too closely — including its noise — and fails to generalize to new data. Training error is low but validation error is high. Remedies include:

- Get more data.
- Regularization.
- Dropout.
- Batch normalization.
- Data augmentation.
- Early stopping.

Lecture 2

2.1 Categories of Machine Learning

2.1.1 Supervised Learning

Definition 2.1 *(Supervised Learning)*. In supervised learning, the model is trained on a labeled dataset to learn a map

$$f : \mathbb{R}^d \to \mathcal{Y}, \tag{2.1}$$

where $\boldsymbol{X}_i \in \mathbb{R}^d$ is the feature vector of the ith data point (with d features) and $y_i \in \mathcal{Y}$ is its label. The dataset consists of N labeled pairs

$$\{(\boldsymbol{X}_i, y_i)\}_{i=1}^{N}, \tag{2.2}$$

and the goal is to learn a mapping that generalizes to unseen data.

Supervised learning is used for:

- **Classification** (binary, multi-class or multi-label): Predict discrete class label.
- **Regression**: Predict continuous outputs (MSE, MAE are used commonly as loss function in regression).

2.1.2 Unsupervised Learning

Definition 2.2 *(Unsupervised Learning)*. The model is trained on unlabeled dataset:

$$\{\boldsymbol{X}_i\}_{i=1}^{N}. \tag{2.3}$$

The goal is to learn the underlying structure or distribution $p(\boldsymbol{X})$ of the data, without any label information.

Common tasks include:

- **Clustering:** partition the data into groups of similar points,
- **Dimensionality Reduction:** learn a mapping $f : \mathbb{R}^d \to \mathbb{R}^{d'}$ with $d' \ll d$ that preserves important structure,
- **Anomaly Detection:** identify points $\boldsymbol{X}_i$ where $p(\boldsymbol{X}_i) \ll 1$.

2.1.3 Reinforcement Learning

The key elements of reinforcement learning are:

- **Agent:** The learner or decision-maker that interacts with the environment and learns to improve its behaviour over time.
- **Environment:** The external system the agent interacts with, which responds to the agent's actions and produces new states and rewards.
- **State $\boldsymbol{s}_t \in \mathcal{S}$:** The current situation of the agent in the environment, capturing all relevant information needed to make a decision.
- **Action $a_t \in \mathcal{A}$:** The choices available to the agent at each timestep, which can be discrete (e.g. legal moves in chess) or continuous (e.g. joint angles of a robot arm).
- **Reward r_t:** A feedback signal received after each action, indicating how successful that action was in progressing toward the goal.
- **Policy $\pi : \mathcal{S} \to \mathcal{A}$:** A strategy that defines the agent's actions based on the current state, which the agent refines over time to maximise cumulative reward.

Definition 2.3 *(Reinforcement Learning)*. In reinforcement learning, an agent learns a policy

$$\pi : \mathcal{S} \to \mathcal{A}, \tag{2.4}$$

where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. At each timestep t, the agent:

1. observes state $\boldsymbol{s}_t \in \mathcal{S}$,
2. takes action $a_t \in \mathcal{A}$,
3. receives reward r_t, generating a trajectory of tuples

$$\{(\boldsymbol{s}_t,\, a_t,\, r_t,\, \boldsymbol{s}_{t+1})\}_{t=0}^{T}. \tag{2.5}$$

The goal is to learn a policy π that maximizes the cumulative discounted reward

$$R = \sum_{t=0}^{T} \gamma^t r_t, \tag{2.6}$$

where $\gamma \in [0, 1]$ is a discount factor controlling the importance of future rewards.

Note 2.4 *(State and Action Space)*. The state space $\mathcal{S}$ is the set of all possible situations the agent can observe. For example, in a chess game $\mathcal{S}$ is the set of all possible board configurations, while in a self-driving car $\mathcal{S}$ might encode the car's position, speed, and surrounding obstacles.
The action space $\mathcal{A}$ is the set of all possible actions the agent can take. The action space can be:

- **Discrete:** a finite set of distinct actions, $\mathcal{A} = \{a_1, a_2, \ldots, a_K\}$. For example, in a chess game $\mathcal{A}$ is the set of all legal moves.
- **Continuous:** an infinite range of actions, $\mathcal{A} \subseteq \mathbb{R}^k$. For example, in a robot arm $\mathcal{A}$ might be the set of all possible joint rotation angles.

Together, they define the full decision-making problem — the policy $\pi : \mathcal{S} \to \mathcal{A}$ maps each observed state to an action.

2.1.4 Semi-supervised and Self-supervised Learning

Definition 2.5 *(Semi-supervised Learning)*. In semi-supervised learning, the model is trained on a small set of labeled data

$$\{(\boldsymbol{X}_i,\, y_i)\}_{i=1}^{N_l}, \tag{2.7}$$

combined with a large set of unlabeled data

$$\{\boldsymbol{X}_j\}_{j=1}^{N_u}, \quad N_u \gg N_l. \tag{2.8}$$

The model leverages the labeled data to learn initial patterns and then uses the unlabeled data to refine its understanding and improve generalization and performance.

Definition 2.6 *(Self-supervised Learning)*. Self-supervised learning is a type of unsupervised learning in which the model generates its own pseudo-labels from the input data via a *pretext task*. Given an unlabeled dataset

$$\{\boldsymbol{X}_i\}_{i=1}^{N}, \tag{2.9}$$

the pretext task automatically constructs labeled training pairs $\{(\tilde{\boldsymbol{X}}_i,\, \hat{y}_i)\}_{i=1}^{N}$ by augmenting, masking, or transforming the original data, where $\hat{y}_i$ is the pseudo-label. The model then trains on these pairs in a supervised fashion, learning useful internal representations without requiring manual annotations.

Note 2.7 *(Pretext Task).* A pretext task is an artificial task designed to generate pseudo-labels from unlabeled data. The model does not ultimately care about solving the pretext task itself, the goal is to learn rich internal representations that can be transferred to downstream tasks. Common approaches include:

- **Masking:** Hide part of the input and train the model to predict the missing piece (e.g., masked language modeling in BERT).
- **Transformation prediction:** Apply a known transformation (e.g., rotation by $0°, 90°, 180°, 270°$) and train the model to predict which transformation was applied.
- **Next-token prediction:** Train the model to predict the next element in a sequence given all previous elements (e.g., GPT).

Note 2.8 *(Self-supervised Learning vs. Unsupervised Learning).* Both self-supervised and unsupervised learning operate on unlabeled data $\{\boldsymbol{X}_i\}_{i=1}^N$, but they differ in mechanism:

- **Unsupervised learning** discovers structure directly from the data distribution $p(\boldsymbol{X})$ without any labels (e.g., clustering, dimensionality reduction).
- **Self-supervised learning** manufactures pseudo-labels via a pretext task and then trains in a supervised fashion on the resulting (input, pseudo-label) pairs.

In other words, unsupervised learning finds patterns without labels, while self-supervised learning creates its own labels and uses the supervised learning framework to learn representations.

Note 2.9 *(Integrating Mechanistic Models with ML).* Integrating Mechanistic Models with Machine Learning: Using physical laws or domain knowledge to inform and constrain machine learning models, enhancing their interpretability and reliability.

2.2 Supervised Learning and Classification

2.2.1 The Problem Statement of Classification

The input (feature vector) is $\boldsymbol{X} \in \mathbb{R}^d$, the output (class label) is a discrete variable $y \in \{1, 2, 3, \ldots, K\}$. Given a new $\boldsymbol{X}_{\text{new}}$, the model should predict which class it belongs to, y_{new}.

2.2.2 A Probabilistic View of Classification

Definition 2.10 *(Posterior Probability).* The posterior probability is defined as

$$P(y = k \mid \boldsymbol{X}), \tag{2.10}$$

where $y = k$ is the hypothesis and $\boldsymbol{X}$ is the data.

Posterior Probability vs. Conditional Probability

The distinction between conditional and posterior probability is not mathematical, it is conceptual. Posterior emphasizes learning from data, while conditional is a neutral probabilistic relationship.

We want to minimize the probability of misclassifications.

Theorem 2.11 *(Bayes Decision Rule).* The Bayes classifier assigns an input $\boldsymbol{X}$ to the class with largest posterior probability

$$\hat{y}(\boldsymbol{X}) = \underset{k \in \{1,2,\ldots,K\}}{\arg\max}\; P(y = k \mid \boldsymbol{X}). \tag{2.11}$$

Theorem 2.12 *(Bayes Theorem).* The Bayes theorem is

$$P(y = k \mid \boldsymbol{X}) = \frac{P(\boldsymbol{X} \mid y = k)\, P(y = k)}{P(\boldsymbol{X})}. \tag{2.12}$$

It flip which variable is being conditioned on.

Definition 2.13. The conditional probabilit is given by

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}. \tag{2.13}$$

Note 2.14 *(Posterior, Likelihood, and Conditional Probability).* All three are conditional probabilities $P(A \mid B)$, but differ in context:

- **Conditional probability:** The general mathematical concept $P(A \mid B)$, with no particular interpretation attached.
- **Posterior probability $P(y = k \mid \boldsymbol{X})$:** A conditional probability where we con-

dition on *data* to learn about a *hypothesis*. "Posterior" means *after* — it is our belief about the class *after* observing the evidence.

- **Likelihood $P(\boldsymbol{X} \mid y = k)$:** A conditional probability where we condition on a *hypothesis* to evaluate how well it explains the *data*.
- **Prior probability $P(y = k)$:** Our belief about the class *before* observing any data.

The distinction is conceptual, not mathematical. Bayes Theorem connects them:

$$\underbrace{P(y = k \mid \boldsymbol{X})}_{\text{posterior}} = \frac{\overbrace{P(\boldsymbol{X} \mid y = k)}^{\text{likelihood}} \cdot \overbrace{P(y = k)}^{\text{prior}}}{\underbrace{P(\boldsymbol{X})}_{\text{evidence}}}.$$

Using theorem 2.11 and theorem 2.12, since the denominator is k independent, the Bayes decision rule is implied to be

$$\hat{y}(\boldsymbol{X}) = \underset{k \in \{1,2,\ldots,K\}}{\arg\max}\; P(\boldsymbol{X} \mid y = k)\, P(y = k), \tag{2.14}$$

where $P(\boldsymbol{X} \mid y = k)$ is the class-conditional likelihood and $P(y = k)$ is the prior probability of class k.

2.2.3 Linear Discriminant Analysis (LDA)

LDA is based on the following assumptions:

- $P(\boldsymbol{X} \mid y = k) = \mathcal{N}(\boldsymbol{X} \mid \boldsymbol{\mu}_k, \Sigma)$, a multivariate Gaussian.
- All classes share the same covariance matrix $\Sigma_k = \Sigma$.
- $\Pi_k = P(y = k)$, the prior probability.
- $\boldsymbol{\mu}_k \in \mathbb{R}^d$ is the mean vector of class k.

Therefore

$$P(\boldsymbol{X} \mid y = k)\, P(y = k) = P(\boldsymbol{X} \mid y = k)\, \Pi_k, \tag{2.15}$$

so that eq. (2.14) is

$$\hat{y}(\boldsymbol{X}) = \underset{k \in \{1,2,\ldots,K\}}{\arg\max}\; P(\boldsymbol{X} \mid y = k)\, \Pi_k. \tag{2.16}$$

Taking the log of eq. (2.16)

$$\hat{y}(\boldsymbol{X}) = \underset{k \in \{1,2,\ldots,K\}}{\arg\max}\; \left(\log P(\boldsymbol{X} \mid y = k) + \log \Pi_k\right). \tag{2.17}$$

Given that

$$P(\boldsymbol{X} \mid y=k) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{X}-\boldsymbol{\mu}_k)^\top \Sigma^{-1}(\boldsymbol{X}-\boldsymbol{\mu}_k)\right).$$

Calculate $\log P(\boldsymbol{X} \mid y=k)$ to get

$$\log P(\boldsymbol{X} \mid y=k) = -\frac{1}{2}\boldsymbol{X}^\top \Sigma^{-1}\boldsymbol{X} + \boldsymbol{X}^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\log|\Sigma| - \frac{d}{2}\log(2\pi). \quad (2.18)$$

Substitute into $\hat{y}(\boldsymbol{X})$ and drop terms that are independent of k to obtain the LDA

$$\hat{y}(\boldsymbol{X}) = \arg\max_k \ \delta_k(\boldsymbol{X})$$

where

$$\delta_k(\boldsymbol{X}) = \boldsymbol{X}^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k + \log \Pi_k.$$

Definition 2.15 *(LDA).* The LDA is defined as

$$\hat{y}(\boldsymbol{X}) = \underset{k\in\{1,2,\ldots,K\}}{\arg\max} \ \delta_k(\boldsymbol{X}),$$

where δ_k is defined as

$$\delta_k(\boldsymbol{X}) = \boldsymbol{X}^\top \Sigma^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\top \Sigma^{-1}\boldsymbol{\mu}_k + \log \Pi_k.$$

Suppose we have a case where $K=2$, i.e., two classes

$$\delta_1(\boldsymbol{X}) - \delta_2(\boldsymbol{X}) \geq 0.$$

Substitute from

$$\delta_k(\boldsymbol{X}) \implies \boldsymbol{W}^\top \boldsymbol{X} + b \geq 0$$

Then

$$\boldsymbol{W} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \quad b = -\frac{1}{2}\left(\boldsymbol{\mu}_1^\top \Sigma^{-1}\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2^\top \Sigma^{-1}\boldsymbol{\mu}_2\right) + \log\frac{\Pi_1}{\Pi_2}$$

2.2.4 Logistic Regression

Logistic Regression is a supervised algorithm for classification. We approximate the posterior probability $P(y=k \mid \boldsymbol{X})$.

The problem we solve is a binary classification. We have features $\boldsymbol{X} \in \mathbb{R}^d$ and $y \in \{0,1\}$ as outputs. The goal is to model the probability that an observation belongs to class 1, given its features, i.e., we need

$$P(y=1 \mid \boldsymbol{X}).$$

Recall in linear regression, we try to fit a line to data or hyperplane to higher dimension data with

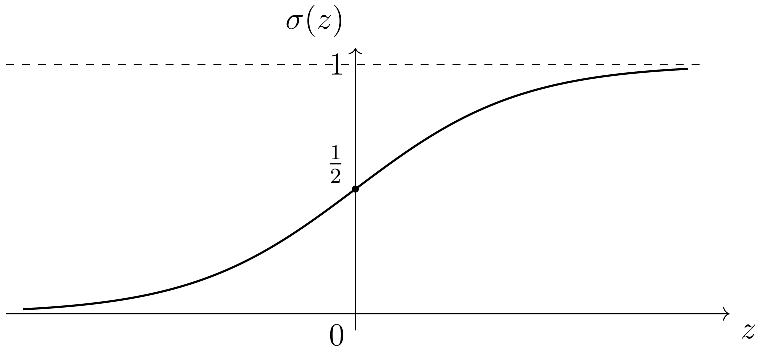
$$\boldsymbol{\beta}^\top \boldsymbol{X}, \quad \beta_0 + \beta_1 x_1 + \cdots + \beta_{d-1} x_d, \quad \boldsymbol{\beta} \in \mathbb{R}^d$$

for

$$\boldsymbol{X} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}.$$

To obtain results as probability, we need to transform with a sigmoid function

$$\boldsymbol{\beta}^\top \boldsymbol{X} \to \sigma\left(\boldsymbol{\beta}^\top \boldsymbol{X}\right) = \frac{1}{1+e^{-\boldsymbol{\beta}^\top \boldsymbol{X}}}.$$



The new probability after the sigmoid transformation is

$$\begin{cases} P(y=1 \mid \boldsymbol{X}) = \dfrac{1}{1+e^{-\boldsymbol{\beta}^\top \boldsymbol{X}}} \\ P(y=0 \mid \boldsymbol{X}) = 1 - P(y=1 \mid \boldsymbol{X}). \end{cases}$$

Likelihood function: Given the dataset $\{(\boldsymbol{X}_i, y_i)\}_{i=1}^N$ and $y \in \{0,1\}$, the likelihood function $L(\boldsymbol{\beta})$ is the product of the probabilities of observing each data point:

$$P(y_i \mid \boldsymbol{X}_i, \boldsymbol{\beta}) = \big(P(y=1 \mid \boldsymbol{X}_i)\big)^{y_i} \big(P(y=0 \mid \boldsymbol{X}_i)\big)^{1-y_i}$$

$$L(\boldsymbol{\beta}) = \prod_{i=1}^N P(y_i \mid \boldsymbol{X}_i, \boldsymbol{\beta}) = \prod_{i=1}^N \big(\sigma(\boldsymbol{\beta}^\top \boldsymbol{X}_i)\big)^{y_i} \big(1 - \sigma(\boldsymbol{\beta}^\top \boldsymbol{X}_i)\big)^{1-y_i}$$

We work with the log-likelihood function

$$\hat{L}(\boldsymbol{\beta}) = \sum_{i=1}^{N} \left[y_i \log \sigma(\boldsymbol{\beta}^\top \boldsymbol{X}_i) + (1 - y_i) \log \big(1 - \sigma(\boldsymbol{\beta}^\top \boldsymbol{X}_i) \big) \right],$$

where σ is the sigmoid function.

It is difficult to find a closed-form solution like the LDA, so gradient-descent etc. will be used.

Lecture 3

3.1 Review And Summary of Lecture 2

3.1.1 Bayes Decision Rule

Let $y \in \{0,1\}$ denote the class label and $\boldsymbol{X}$ the feature vector. The Bayes classifier is given by

$$\hat{y}(\boldsymbol{X}) = \arg\max_k \mathbb{P}(y = k \mid \boldsymbol{X}),$$

or using Bayes' theorem, this can be written as

$$\hat{y}(\boldsymbol{X}) = \arg\max_k \mathbb{P}(\boldsymbol{X} \mid y = k)\, \mathbb{P}(y = k),$$

where $\mathbb{P}(\boldsymbol{X} \mid y = k)$ is the class-conditional likelihood and $\mathbb{P}(y = k) = \Pi_k$ is the prior probability of class k.

3.1.2 Linear Discriminant Analysis (LDA)

In Linear Discriminant Analysis (LDA), it is assumed that

$$\mathbb{P}(\boldsymbol{X} \mid y = k)$$

is Gaussian for each class k, with all classes sharing a common covariance matrix.

3.1.3 Logistic Regression

The logistic regression

$$\mathbb{P}(y = 1 \mid X) = \sigma(\boldsymbol{\beta}^T \boldsymbol{X}) = \frac{1}{1 + e^{-\boldsymbol{\beta}^T \boldsymbol{X}}}$$

directly models the posterior probability and is typically used for binary classification problems with

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

and labeled data

$$\{(\boldsymbol{X}_i, y_i)\}_{i=1}^n .$$

3.1.4 Log-Likelihood and Loss Function

The likelihood formulation is

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \big(\sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i)\big)^{y_i} \big(1 - \sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i)\big)^{1-y_i}.$$

We want to choose $\boldsymbol{\beta}$ that maximizes the log-likelihood:

$$\log L(\boldsymbol{\beta}) = \sum_{i=1}^{n} \Big(y_i \log \sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i) + (1-y_i) \log\big(1 - \sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i)\big)\Big).$$

Equivalently, we minimize the negative log-likelihood, which defines the binary cross-entropy loss

$$J(\boldsymbol{\beta}) = -\log L(\boldsymbol{\beta}).$$

The binary cross-entropy loss $J(\beta)$ has no closed-form solution. We therefore turn to iterative optimization methods. For this course, we use gradient descent.

3.2 Gradient Descent

3.2.1 General Update Rule

Suppose

$$\min_{\beta \in \mathbb{R}^d} f(\boldsymbol{\beta}) \quad f : \mathbb{R}^d \to \mathbb{R},$$

where f is differentiable. We need to find $\boldsymbol{\beta}^*$ such that

$$\boldsymbol{\nabla} f(\boldsymbol{\beta}^*) = 0.$$

The directional derivative is minimized when

$$\hat{\boldsymbol{u}} = -\frac{\boldsymbol{\nabla} f(\boldsymbol{\beta}^*)}{\|\boldsymbol{\nabla} f(\boldsymbol{\beta}^*)\|}.$$

The basic idea is to use gradient descent to minimize the loss function

$$f\left(\boldsymbol{\beta}^{(k)} + \Delta\boldsymbol{\beta}\right) \simeq f\left(\boldsymbol{\beta}^{(k)}\right) + \underbrace{\nabla f\left(\boldsymbol{\beta}^{(k)}\right)^T \Delta\boldsymbol{\beta}}_{<0},$$

where

$$\Delta\boldsymbol{\beta} = -\eta\, \nabla f\left(\boldsymbol{\beta}^{(k)}\right).$$

The parameter update rule is

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} - \eta \underbrace{\boldsymbol{\nabla} f\left(\boldsymbol{\beta}^{(k)}\right)}_{\equiv \boldsymbol{\nabla} J(\boldsymbol{\beta})}. \tag{3.1}$$

where η is the learning rate.

3.2.2 Gradient Descent for Logistic Regression

Applying eq. (3.1) to logistic regression requires computing the gradient of $J(\boldsymbol{\beta})$ with respect to $\boldsymbol{\beta}$. We differentiate the binary cross-entropy loss term by term. We need

$$\boldsymbol{\nabla}_{\boldsymbol{\beta}} J(\boldsymbol{\beta}) = -\sum_{i=1}^{n} \nabla_{\boldsymbol{\beta}} \Big[y_i \log \sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i) + (1-y_i) \log\big(1 - \sigma(\boldsymbol{\beta}^T \boldsymbol{X}_i)\big) \Big].$$

Let

$$z_i = \boldsymbol{\beta}^T \boldsymbol{X}_i, \quad \frac{d}{dz} \log \sigma(z) = 1 - \sigma(z), \quad \frac{d}{dz} \log\big(1 - \sigma(z)\big) = -\sigma(z), \quad \boldsymbol{\nabla}_{\boldsymbol{\beta}} z_i = \boldsymbol{X}_i,$$

then,

$$\left.\begin{aligned} \nabla_{\boldsymbol{\beta}}\big[y_i \log \sigma(z_i)\big] &= y_i\big(1 - \sigma(z_i)\big)\boldsymbol{X}_i \\ \nabla_{\boldsymbol{\beta}}\big[(1-y_i)\log\big(1-\sigma(z_i)\big)\big] &= -(1-y_i)\sigma(z_i)\boldsymbol{X}_i \end{aligned}\right\} \implies \boldsymbol{\nabla}_{\boldsymbol{\beta}} J(\boldsymbol{\beta}) = \sum_{i=1}^{n} \big(\sigma(z_i) - y_i\big)\boldsymbol{X}_i.$$

Note that the update rule is now

$$\boldsymbol{\beta}^{(k+1)} = \boldsymbol{\beta}^{(k)} - \eta \, \boldsymbol{\nabla}_{\boldsymbol{\beta}} J(\boldsymbol{\beta}^{(k)}).$$

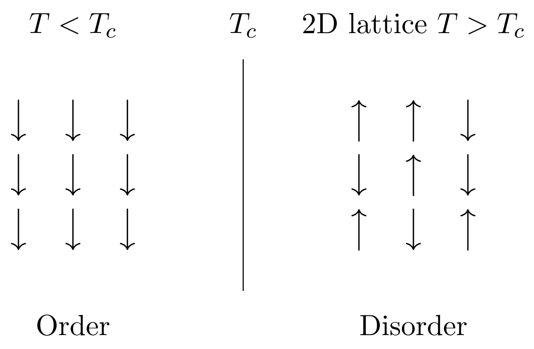
3.3 Example: Ising Model

To illustrate classification in a physics context, consider the Ising model phase transition. Given a spin configuration, can we classify whether the system is above or below the critical temperature?

The Hamiltonian of the system is

$$H = -J \sum_{\langle ij \rangle} \sigma_i \sigma_j,$$

where $\langle ij \rangle$ denotes nearest-neighbor pairs (2D lattice) and $\sigma_i \in \{-1, +1\}$. For $J = 1$ in the 2D square-lattice Ising model: $T_c \approx 2.269$.



In this setting, the feature vector $\boldsymbol{X}$ could consist of statistics computed from the spin configuration, such as the magnetization, energy, or spatial correlation functions. The label $y \in \{0, 1\}$ indicates whether the configuration was sampled from the ordered phase ($T < T_c$) or the disordered phase ($T > T_c$). Logistic regression can then be trained on labeled configurations to predict the phase from the features.

3.4 Example: Prediction of Immunotherapy Response

Consider the problem of predicting patient response to immunotherapy with pre-treatment clinical data. This example is motivated by recent work on LORIS Immunotherapy, reported in Nature Cancer (2024).

Question: Can we predict, prior to treatment, the probability that a patient will respond to immunotherapy using measured clinical variables?

We are given labeled clinical data consisting of six measured features per patient. Let

$$\boldsymbol{X} \in \mathbb{R}^6$$

denote the feature vector for a single patient, and let

$$\{(\boldsymbol{X}_i, y_i)\}_{i=1}^{n}$$

be the full dataset, where n is the number of data points and

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

indicates whether patient i responds to immunotherapy. The goal is to model the probability of response given the clinical features.

We use a logistic regression model to estimate the probability of response:

$$\mathbb{P}(y = 1 \mid \boldsymbol{X}) = \sigma\left(\boldsymbol{\beta}^T \boldsymbol{X}\right) = \sigma\left(\beta_0 + \sum_{j=1}^{6} \beta_j X_j\right),$$

where β_0 is the bias (intercept) term. The quantity

$$\mathbb{P}(y = 1 \mid \boldsymbol{X})$$

represents the predicted probability that a patient responds to immunotherapy given their pre-treatment clinical variables.

3.4.1 Log-Odds of Response Approach

An equivalent and often more interpretable form of logistic regression is obtained by considering the *odds* and *log-odds* of response. The odds of response given the clinical features $\boldsymbol{X}$ are defined as

$$\frac{\mathbb{P}(y = 1 \mid \boldsymbol{X})}{\mathbb{P}(y = 0 \mid \boldsymbol{X})} = \frac{\mathbb{P}(y = 1 \mid \boldsymbol{X})}{1 - \mathbb{P}(y = 1 \mid \boldsymbol{X})}.$$

Using the logistic regression model

$$\mathbb{P}(y = 1 \mid \boldsymbol{X}) = \sigma\left(\beta_0 + \sum_{j=1}^{6} \beta_j X_j\right),$$

we obtain the log-odds:

$$\begin{aligned} \log \frac{\mathbb{P}(y = 1 \mid \boldsymbol{X})}{\mathbb{P}(y = 0 \mid \boldsymbol{X})} &= \log \frac{\mathbb{P}(y = 1 \mid \boldsymbol{X})}{1 - \mathbb{P}(y = 1 \mid \boldsymbol{X})} \\ &= \beta_0 + \sum_{j=1}^{6} \beta_j X_j. \end{aligned} \tag{3.2}$$

Lecture 4

4.1 Support Vector Machines

SVM is a form of supervised learning, we have labeled data

$$\{(\boldsymbol{X}_i, y_i)\}_{i=1}^{n}.$$

(FILL IN)

We have a classifier

$$f(x) = \omega^T \boldsymbol{X} + b$$

that outputs

$$\operatorname{sign} f(\boldsymbol{X}).$$

We have a decision boundary

$$\omega^T \boldsymbol{X} + b = 0.$$

Note that

- $\boldsymbol{\omega}$ is the normal vector to this decision boundary.
-