3 Basics of Math and Machine Learning

As you may know, deep learning relies extensively on mathematical principles to model and interpret complex data patterns. In this chapter, we introduce some fundamental mathematical concepts that form the backbone of deep learning. We will start with a brief review of linear algebra and probability theory, two fields that provide the tools to manipulate data and quantify uncertainties.

Linear algebra focuses on vectors, matrices, and their operations—essential for managing the vast data arrays processed by machine learning algorithms. Probability theory provides a framework for making informed predictions based on existing data-critical in developing machine learning models.

Following the review, we will provide an overview of how machine learning algorithms learn from data to make prediction. This will set the stage for further explorations into specific deep learning techniques and their applications in subsequent chapters.

By the end of this chapter, you will have refreshed your understanding of the mathematical tools vital for navigating the exciting realm of deep learning.

3.1 Linear Algebra

Linear algebra is a branch of mathematics that is widely used in engineering. It is a very comprehensive field by itself that covers numerous topics. In this section, we will review only a subset of topics that are highly related to deep learning. If you need a comprehensive overview of linear algebra, we refer you to other textbooks [1].

3.1.1 Scalars, Vectors, Matrices, and Tensors

A **scalar** is just a single number. In this book, we denote them with lower-case italics letters. For example, the systolic blood pressure of a patient can be represented with a real-valued scalar $p \in \mathbb{R}$, and the number of times a patient has stroke in the past three years can be represented with a natural number scalar $s \in \mathbb{N}$.

A **vector** is an ordered list of numbers. It can represent things such as an arrow pointing in space, the coefficients of a polynomial, or a patient’s vital signs such as temperature, pulse rate, and blood pressure. In this book, we write vectors vertically and denote them with lower-case bold letters. For example, a vector $\mathbf{x}$ might be represented as follows:

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix},$$

where x_1, x_2, x_3 are the elements of the vector $\mathbf{x}$. If each element is a real-valued scalar (i.e., $x_1, x_2, x_3 \in \mathbb{R}$), then the vector $\mathbf{x}$ lies in the set formed by taking the Cartesian product of $\mathbb{R}$ 3 times, denoted as $\mathbf{x} \in \mathbb{R}^3$.

A **matrix** is a two-dimensional array of numbers, arranged in rows and columns, and is often used to represent linear transformations or to store datasets (e.g., vital signs for 100 patients). In this book, matrices are denoted by capital bold letters. For example, matrix $\mathbf{A}$ can look like:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

where a_{ij} represents the element in the i-th row and j-th column of the matrix $\mathbf{A}$. If the matrix $\mathbf{A}$ is real-valued, we can write $\mathbf{A} \in \mathbb{R}^{3\times 3}$. Note that vectors can be viewed as matrices with only one column.

A **tensor** is generalization of matrix to higher dimensions. It is a multi-dimensional array with more than two axes. For example, a color image can be represented by a width $\times$ height $\times$ color channels tensor. In this book, we also denote tensors with capital bold letters and use the terminologies tensors and matrices interchangeably.

3.1.2 Transpose

The **transpose** of a matrix is an operation that flips a matrix over its diagonal, turning the row indices into column indices and vice versa. The transpose of a matrix $\mathbf{A}$ is denoted by $\mathbf{A}^\top$. For example, if matrix $\mathbf{A}$ is:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix},$$

then the transpose of $\mathbf{A}$, $\mathbf{A}^\top$, is:

$$\mathbf{A}^\top = \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \\ a_{13} & a_{23} \end{bmatrix}.$$

This operation changes the dimensions of $\mathbf{A}$ from 2×3 to 3×2, and each element a_{ij} in $\mathbf{A}$ moves to position a_{ji} in $\mathbf{A}^\top$.

Similarly, since (columns) vectors can be viewed as matrices with only one column, the transpose of a vector is a row vector. For instance, for a vector $\mathbf{v} \in \mathbb{R}^3$,

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix},$$

the transpose of **v** is a row vector,

$$\mathbf{v}^\top = \begin{bmatrix} v_1, v_2, v_3 \end{bmatrix}.$$

3.1.3 Addition and Subtraction

Addition and **subtraction** of vectors and matrices are performed element-wise, requiring the operands to have the same dimensions. For example, consider the matrices **A** and **B**, each of dimension 2×2:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}.$$

Their addition is:

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} a_{11} + b_{11} & a_{12} + b_{12} \\ a_{21} + b_{21} & a_{22} + b_{22} \end{bmatrix},$$

and their subtraction is:

$$\mathbf{A} - \mathbf{B} = \begin{bmatrix} a_{11} - b_{11} & a_{12} - b_{12} \\ a_{21} - b_{21} & a_{22} - b_{22} \end{bmatrix}.$$

We can also add a scalar to a matrix. For example, we write $\mathbf{A} + s$, which represents adding the scalar s to each element of the matrix **A**, as

$$\mathbf{A} + s = \begin{bmatrix} a_{11} + s & a_{12} + s \\ a_{21} + s & a_{22} + s \end{bmatrix}.$$

With a little bit abuse of notation, we can add a vector to a matrix $\mathbf{A}+b$. For example, we write $\mathbf{A} + \mathbf{v}$, which represents adding the vector **v** to each row of matrix **A**, , as

$$\mathbf{A} + \mathbf{v} = \begin{bmatrix} a_{11} + v_1 & a_{12} + v_2 \\ a_{21} + v_1 & a_{22} + v_2 \end{bmatrix}.$$

This operation of implicitly coping **v** to many locations is called broadcasting. Note that this operation presumes the dimensions are matched between **A** and **v**.

Properties of matrix addition include:

- $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$
- $(\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C})$
- $\mathbf{A} + 0 = \mathbf{A}$
- $\mathbf{A} + (-\mathbf{A}) = 0$
- $(\mathbf{A} + \mathbf{B})^\top = \mathbf{A}^\top + \mathbf{B}^\top$

3.1.4 Multiplication

Scalar Multiplication involves multiplying each element of a matrix by a scalar. If c is a scalar and **A** is a matrix, then the product $c\mathbf{A}$ is:

$$c\mathbf{A} = \begin{bmatrix} c \cdot a_{11} & c \cdot a_{12} \\ c \cdot a_{21} & c \cdot a_{22} \end{bmatrix}.$$

Matrix Multiplication is slightly more complex. To multiply two matrices, the number of columns in the first matrix must equal the number of rows in the second matrix. For matrices **A** and **B**, where **A** is an $m \times n$ matrix and **B** is an $n \times p$ matrix, the result $\mathbf{C} = \mathbf{AB}$ is an $m \times p$ matrix. The elements of **C** are computed as follows:

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

for each i from 1 to m and each j from 1 to p.

For example, consider the following matrices **A** (a 2×3 matrix) and **B** (a 3×2 matrix):

$$\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix}.$$

Their product $\mathbf{C} = \mathbf{AB}$ is:

$$\mathbf{C} = \begin{bmatrix} (1 \times 7 + 2 \times 9 + 3 \times 11) & (1 \times 8 + 2 \times 10 + 3 \times 12) \\ (4 \times 7 + 5 \times 9 + 6 \times 11) & (4 \times 8 + 5 \times 10 + 6 \times 12) \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix}.$$

The **dot product** of two vectors is a scalar obtained by multiplying corresponding elements of the vectors and summing the results. For vectors **u** and **v** in $\mathbb{R}^3$, their dot product is:

$$\mathbf{u} \cdot \mathbf{v} = u_1 v_1 + u_2 v_2 + u_3 v_3.$$

The dot product is a measure of the angle between two vectors; if the dot product is zero, the vectors are orthogonal. This operation is the same as performing matrix multiplication between **u** and $\mathbf{v}^\top$, written as $\mathbf{uv}^\top$.

Properties of matrix multiplication include:

- $\mathbf{AB} \neq \mathbf{BA}$ (in general)
- $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$
- $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$
- $\mathbf{A}0 = 0\mathbf{A} = 0$
- $(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top$

3.1.5 Identity and Inverse Matrices

The **identity matrix**, denoted as **I**, is characterized by having 1's on its diagonal and 0's in all other positions. For an $n \times n$ identity matrix, it is defined as:

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix}.$$

The identity matrix serves as the multiplicative identity in matrix multiplication, such that for any square matrix **A** of appropriate size,

$$\mathbf{AI} = \mathbf{IA} = \mathbf{A}.$$

This unique property makes the identity matrix analogous to the number 1 in scalar multiplication.

The **inverse** of a matrix is a fundamental concept in linear algebra that allows for solving systems of linear equations, among other applications. The inverse of a matrix **A**, denoted as $\mathbf{A}^{-1}$, is a matrix such that when it is multiplied by **A**, the result is the identity matrix **I** of the same dimension as **A**:

$$\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}.$$

3.1.6 Norms

Vector norms are functions that measure the size or length of a vector. Some common vector norms include:

- **Manhattan Norm (L1 Norm)** The Manhattan norm, or L1 norm, measures the sum of the absolute values of the components of the vector. It is widely used in optimization problems where sparsity is desired. For a vector **v**, the Manhattan norm is given by:
$$\|\mathbf{v}\|_1 = |v_1| + |v_2| + \ldots + |v_n|.$$
- **Euclidean Norm (L2 Norm)** The Euclidean norm, also known as the L2 norm, is the most common type of norm, which measures the standard geometric length of a vector. For a vector $\mathbf{v} \in \mathbb{R}^n$, the Euclidean norm is defined as:
$$\|\mathbf{v}\|_2 = \sqrt{v_1^2 + v_2^2 + \ldots + v_n^2}.$$

Matrix norms are a natural extension of vector norms into matrices, providing a measure of the "size" or "magnitude" of a matrix. A commonly used matrix norm is the **Frobenius Norm**. It is similar to the Euclidean norm for matrices and is defined as the square root of the sum of the absolute squares of its elements:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2},$$

where a_{ij} are the elements of the matrix **A**.

3.2 Probability

In the realm of healthcare, decisions are often underpinned by probabilistic reasoning, whether diagnosing a patient, predicting treatment outcomes, or modeling disease spread. For practitioners harnessing deep learning, understanding the fundamentals of probability is crucial. This section delves into the essential probability concepts necessary for building, interpreting, and evaluating deep learning models in healthcare.

3.2.1 Probability Space

A **probability space** is a mathematical framework that lays the foundation for probability theory and enables the formal analysis of random experiment. It consists of three main components:

- **Sample Space Ω:** This is the set of all possible outcomes of a random experiment. For instance, if the experiment is tossing a fair coin, the sample space would be $\Omega = \{\text{heads}, \text{tails}\}$.
- **Events $\mathcal{F}$:** This is a set of subsets of the sample space, which includes all possible events for which we want to assign probabilities. If the sample space Ω is finite/-countable, we can set $\mathcal{F} = 2^{\Omega}$ (i.e., the power set of Ω). For example, in the case of tossing a fair coin, $\mathcal{F} = \{\emptyset, \{\text{heads}\}, \{\text{tails}\}, \{\text{heads}, \text{tails}\}\}$.
- **Probability Measure P:** This is a function that assigns a probability to each event in $\mathcal{F}$. For example, in the case of tossing a fair coin, the probability measure P is:

$$P(\emptyset) = 0 \text{ (the probability of an impossible event)}, \tag{3.1}$$
$$P(\text{heads}) = 0.5 \text{ (there is a 50\% chance of landing heads)}, \tag{3.2}$$
$$P(\text{tails}) = 0 \text{ (there is also a 50\% chance of landing tails)}, \tag{3.3}$$
$$P(\{\text{heads}, \text{tails}\}) = 1 \text{ (the probability of the certain event)}. \tag{3.4}$$

3.2.2 Random Variable

A **random variable** is a function that assigns a real number to each outcome in the sample space of a probability experiment. It serves as a bridge between real-world observations and mathematical abstraction, allowing for quantitative analysis of random phenomena. For example, in the the case of tossing a fair coin, you can define a random variable X as,

$$X = \begin{cases} X = 0 & \text{if the outcome is heads,} \\ X = 1 & \text{if the outcome is tails.} \end{cases} \tag{3.5}$$

Random variables can be categorized into two main types:

- **Discrete Random Variables:** These variables take on a countable number of distinct values. For example, the number of patients admitted to a hospital in a day is a discrete random variable because it can be counted.
- **Continuous Random Variables:** These variables can take on any value within an interval on the number line. For instance, the duration of a patient’s hospital stay, measured in days, is a continuous random variable because it can take any positive real value.

3.2.3 Probability Distributions

Probability distributions describe how the probabilities are distributed over the values of a random variable. There are two main types of distributions that correspond to the types of random variables:

- **Probability Mass Function (PMF):** Used for discrete random variables, a PMF gives the probability that a random variable equals each of its possible values. For example, the PMF of a discrete random variable X taking values $x_1, x_2, x_3, \ldots$ can be defined as $P(X = x_i)$ for each i.
- **Probability Density Function (PDF):** Used for continuous random variables, a PDF does not give probabilities directly. Instead, the probability of the random variable falling within a particular range is given by the integral of the PDF over that range. For example, if X is a continuous random variable, the probability that X lies between a and b is $\int_a^b f(x)\,dx$, where $f(x)$ is the PDF of X.

3.2.4 Expectation and Variance

The **expectation** (or expected value, or mean) of a random variable measures the central tendency of the distribution.

For a discrete random variable X, the expectation $E[X]$ is calculated by summing the products of each value of the random variable and its corresponding probability. Mathematically, this is represented as:

$$E[X] = \sum_x x \cdot P(X = x), \tag{3.6}$$

where the sum is over all possible values x that X can take.

For a continuous random variable X, the expectation $E[X]$ is calculated using an integral over all possible values of X, weighted by their probability density. The formula for this is:

$$E[X] = \int_{-\infty}^{\infty} x \cdot f(x)\,dx, \tag{3.7}$$

where $f(x)$ is the PDF of X.

Expectation has several important properties:

- **Linearity:** The expectation operator is linear, meaning that for any two random variables X and Y, and any constants a and b,

$$E[aX + bY] = aE[X] + bE[Y], \tag{3.8}$$

- **Law of the Unconscious Statistician (LOTUS):** This property allows you to compute the expectation of a function of a random variable without knowing its entire distribution, as long as you know the distribution of the original variable:

$$E[g(X)] = \sum g(x) \cdot P(X = x) \tag{3.9}$$

for a discrete random variable, and

$$E[g(X)] = \int g(x) \cdot f(x)\,dx, \tag{3.10}$$

for a continuous random variable.

The **moment** is a generalized concept of expectation. The n-th moment of a random variable X is defined as the expected value of X^n. Mathematically, it is expressed as:

$$\mu'_n = E[X^n]. \tag{3.11}$$

The mean μ of X is its 1-st moment $E[X]$.

The **central moment** is the moment defined relative to the mean $\mu = E[X]$ of the random variable. The n-th central moment is defined as:

$$\mu_n = E[(X - \mu)^n]. \tag{3.12}$$

The **variance** σ^2 of X is its 2-nd central moment $E[(X - \mu)^2]$.

Variance has several important properties:

- **Connection with moments:** An important alternative expression for the variance of a random variable X can be derived using the properties of expectation:

$$\begin{aligned} \text{Var}(X) &= E[(X - \mu)^2] && (3.13)\\ &= E[X^2 - 2\mu X + \mu^2] && (3.14)\\ &= E[X^2] - 2\mu E[X] + \mu^2 && (3.15)\\ &= E[X^2] - E[X]^2 && (3.16) \end{aligned}$$

- **Scaling Property:** Variance scales quadratically with the scaling of the variable:

$$\text{Var}(cX) = c^2\text{Var}(X). \tag{3.17}$$

- **Effect of a Constant Shift:** Adding or subtracting a constant from a random variable affects its mean but does not affect its variance:

$$\text{Var}(X + c) = \text{Var}(X). \tag{3.18}$$

3.2.5 Jointly Distributed Random Variables

Jointly distributed random variables are those where two or more random variables are considered together. This concept is crucial in situations where the behavior of one variable is dependent on, or related to, the behavior of another.

For discrete random variables X and Y, the joint probability mass function $P(X = x, Y = y)$ specifies the probability that X takes on value x and Y takes on value y simultaneously.

Similarly, for continuous random variables, the joint probability density function $f(x, y)$ gives the density of the probability at a particular point (x, y) for X and Y. The probability that X and Y fall within a certain region R is given by the integral of $f(x, y)$ over R:

$$P((X,Y) \in R) = \int \int_R f(x,y)\,dx\,dy. \tag{3.19}$$

From the joint distribution of X and Y, the marginal distributions can be derived. For discrete variables, the marginal probability mass function of X is calculated by summing over all possible values of Y:

$$P(X = x) = \sum_y P(X = x, Y = y). \tag{3.20}$$

For continuous variables, the marginal probability density function of X is obtained by integrating over all possible values of Y:

$$f_X(x) = \int f(x,y)\,dy. \tag{3.21}$$

3.2.6 Independence

Two random variables are **independent** if the occurrence of one does not affect the probability distribution of the other. This can be mathematically represented as:

$$P(X = x, Y = y) = P(X = x)P(Y = y), \tag{3.22}$$

for discrete variables, and

$$f(x,y) = f_X(x)f_Y(y), \tag{3.23}$$

for continuous variables.

3.2.7 Correlation and Covariance

Correlation and covariance are measures that express the degree to which two random variables change together. They provide insights into the strength and direction of a linear relationship between variables.

The **covariance** between two random variables X and Y, denoted as $\text{Cov}(X,Y)$, quantifies the extent to which X and Y vary together. It is defined as:

$$\text{Cov}(X,Y) = E[(X - \mu_X)(Y - \mu_Y)], \tag{3.24}$$

where $\mu_X = E[X]$ and $\mu_Y = E[Y]$ are the expected values of X and Y, respectively. A positive covariance indicates that as X increases, Y tends to increase as well, whereas a negative covariance suggests that as X increases, Y tends to decrease. The covariance can also be expressed using using the expected product of X and Y:

$$\begin{aligned}
\text{Cov}(X,Y) &= E[(X - \mu_X)(Y - \mu_Y)], && (3.25)\\
&= E[XY - X\mu_Y - \mu_X Y - \mu_X\mu_Y], && (3.26)\\
&= E[XY] - E[X]\mu_Y - \mu_X X[Y] - \mu_X\mu_Y, && (3.27)\\
&= E[XY] - E[X]E[Y]. && (3.28)
\end{aligned}$$

Two random variables are **uncorrelated** if their covariance is zero.

Covariance is linearly distributes over addition of variables and scalar multiplication:

$$\mathrm{Cov}(aW + bX, cY + dZ) = ac\mathrm{Cov}(W, Y) + ad\mathrm{Cov}(W, Z) + bc\mathrm{Cov}(X, Y) + bd\mathrm{Cov}(X, Z). \tag{3.29}$$

The **correlation coefficient**, denoted as $\rho_{X,Y}$ or $\mathrm{Corr}(X, Y)$, normalizes the covariance by the standard deviations of X and Y to provide a dimensionless measure:

$$\rho_{X,Y} = \frac{\mathrm{Cov}(X, Y)}{\sigma_X \sigma_Y}, \tag{3.30}$$

where σ_X and σ_Y are the standard deviations of X and Y, respectively. The correlation coefficient ranges from -1 to 1. A value of 1 indicates perfect positive correlation, -1 indicates perfect negative correlation, and 0 indicates no linear relationship.

Independence implies uncorrelation. If two random variables are independent, they are also uncorrelated. This is because independence leads to:

$$E[XY] = E[X]E[Y]. \tag{3.31}$$

Using the covariance formula:

$$\mathrm{Cov}(X, Y) = E[XY] - E[X]E[Y] = E[X]E[Y] - E[X]E[Y] = 0. \tag{3.32}$$

Thus, independence, a stronger condition, ensures that the covariance (and therefore the correlation) is zero.

Uncorrelation does not imply independence. However, the converse is not true; uncorrelated variables do not necessarily imply independence. Variables can be uncorrelated yet still dependent through non-linear relationships. For example, if X is uniformly distributed between $[-1, 1]$ and $Y = X^2$, then:

$$\mathrm{Cov}(X, Y) = 0, \tag{3.33}$$

indicating uncorrelatedness. However, knowing X allows perfect determination of Y, showing that X and Y are dependent.

3.2.8 Conditional Distribution

Conditional probability measures the probability of an event occurring given that another event has already occurred. If A and B are two events within a probability space, and $P(B) > 0$, then the conditional probability of A given B is defined as:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}, \tag{3.34}$$

where $P(A \cap B)$ is the probability of both A and B occurring, and $P(B)$ is the probability of B occurring.

For random variables, the concept of conditional probability extends to conditional distributions. For discrete random variables X and Y, the conditional PMF of X given $Y = y$ is:

$$P(X = x \mid Y = y) = \frac{P(X = x, Y = y)}{P(Y = y)}, \tag{3.35}$$

assuming $P(Y = y) > 0$. This function provides the distribution of X under the condition that Y takes a specific value y.

For continuous random variables, the conditional PDF is similarly defined. If $f(x, y)$ is the joint PDF of X and Y, and $f_Y(y)$ is the marginal PDF of Y, then the conditional PDF of X given $Y = y$ is:

$$f_{X|Y}(x \mid y) = \frac{f(x, y)}{f_Y(y)}, \tag{3.36}$$

provided that $f_Y(y) > 0$.

The **conditional expectation** of X given Y is defined as the expectation of X when Y is known to be a particular value. For a discrete random variable X, the conditional expectation given $Y = y$ is calculated as:

$$E[X \mid Y = y] = \sum_x x \cdot P(X = x \mid Y = y), \tag{3.37}$$

where the sum is over all possible values of X.

For continuous random variables, the conditional expectation is:

$$E[X \mid Y = y] = \int_{-\infty}^{\infty} x \cdot f_{X|Y}(x \mid y), dx, \tag{3.38}$$

where $f_{X|Y}(x \mid y)$ is the conditional PDF of X given $Y = y$.

Conditional expectation has several important properties:

- **Linearity:** The conditional expectation is linear. For random variables X and Z, and constants a and b,

$$E[aX + bZ \mid Y] = aE[X \mid Y] + bE[Z \mid Y]. \tag{3.39}$$

- **Iterated Expectation (Law of Total Expectation):** This property relates the unconditional expectation of a random variable to its conditional expectations:

$$E[X] = E[E[X \mid Y]]. \tag{3.40}$$

3.3 Machine Learning

Machine learning (ML) is a subset of artificial intelligence that focuses on building systems that can learn from data, rather than being explicitly programmed to perform a specific task. These systems are designed to improve their performance as they are exposed to more data over time.

Broadly speaking, ML algorithms can be categorized into the following:

- Supervised Learning: This involves training a model on a labeled dataset, where the desired outputs are known. The goal is to learn a mapping from inputs to outputs that can be generalized to new, unseen data. In healthcare, supervised learning can be used for tasks such as disease diagnosis or patient risk stratification.
- Unsupervised Learning: This approach is used with data that has no labels. The goal is to discover underlying patterns or structures in the data. Clustering and dimensionality reduction are common unsupervised learning methods, useful in segmenting patient populations or reducing the dimensions of complex data sets for further analysis.

In the following, we will use linear regression as an example to give a brief review of machine learning basics.

3.3.1 Linear Regression

Formulation of Linear Regression

The general form of a linear regression model is:

$$y = \beta^\top \mathbf{x} + b + \epsilon, \tag{3.41}$$

where

- $y \in \mathbb{R}$ is the dependent variable (observed outcome);
- $x \in \mathbb{R}^d$ is the vector of d independent variables (predictors);
- $\beta \in \mathbb{R}^d$ is the vector of d slope coefficients, corresponding to each predictor;
- $b \in \mathbb{R}$ is the intercept coefficient, which adjusts the line of best fit to better match the data;
- $\epsilon \in \mathbb{R}$ is the error/residual term, capturing the difference between the observed outcome y and the model prediction $\hat{y} = \beta^\top \mathbf{x} + b$.

The goal is to learn the parameters β and b based on a set of observed samples $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$ to minimize the difference between the predicted values and the actual observed values. This process, known as fitting the model, typically involves minimizing the sum of squared residuals (SSR), which is the sum of the squares of the error terms for each observation:

$$\begin{aligned}
\text{SSR} &= \sum_{i=1}^N \epsilon^{(i)} && (3.42)\\
&= \sum_{i=1}^N (y^{(i)} - \hat{y}^{(i)})^2 && (3.43)\\
&= \sum_{i=1}^N (y^{(i)} - (\beta^\top \mathbf{x}^{(i)} + b))^2. && (3.44)
\end{aligned}$$

We can further simply SSR in matrix form as

$$\text{SSR} = \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2, \tag{3.45}$$

where $\mathbf{y} \in \mathbb{R}^N$ is the vector of dependent variables from all N samples, $\mathbf{w} \in \mathbb{R}^{d+1}$ is the augmented coefficient vector that includes both the slope β and the intercept b, and $\mathbf{X} \in \mathbb{R}^{N\times(d+1)}$ is the matrix of all N independent variables plus an additional column of ones to account for the intercept; as shown below.

$$\mathbf{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}, \mathbf{X} = \begin{bmatrix} x_0^{(1)} & \cdots & x_{d-1}^{(1)} & 1 \\ x_0^{(2)} & \cdots & x_{d-1}^{(2)} & 1 \\ \vdots & \cdots & \vdots & \vdots \\ x_0^{(N)} & \cdots & x_{d-1}^{(N)} & 1 \end{bmatrix}, \mathbf{w} = \begin{bmatrix} \beta_0 \\ \vdots \\ \beta_{d-1} \\ b \end{bmatrix}. \tag{3.46}$$

To find the values of $\mathbf{w}$ (i.e., β and b) that minimize the SSR, various methods can be used, including analytical solutions like the ordinary least squares (OLS) method, or iterative optimization techniques such as gradient descent.

Ordinary Least Squares

The OLS solution is given by:

$$\begin{aligned} \mathbf{w}^* &= \arg\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 && (3.47) \\ &= \arg\min_{\mathbf{w}} (\mathbf{y} - \mathbf{X}\mathbf{w})^\top (\mathbf{y} - \mathbf{X}\mathbf{w}) && (3.48) \\ &= \arg\min_{\mathbf{w}} \left(\mathbf{y}^\top\mathbf{y} - \mathbf{y}^\top\mathbf{X}\mathbf{w} - \mathbf{w}^\top\mathbf{X}^\top\mathbf{y} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w}\right) && (3.49) \\ &= \arg\min_{\mathbf{w}} \left(\mathbf{y}^\top\mathbf{y} - 2\mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w}\right). && (3.50) \\ & && (3.51) \end{aligned}$$

Set its partial derivative with respect to $\mathbf{w}$ to 0, we have

$$\begin{aligned} \frac{\partial}{\partial \mathbf{w}} \left(\mathbf{y}^\top\mathbf{y} - 2\mathbf{y}^\top\mathbf{X}\mathbf{w} + \mathbf{w}^\top\mathbf{X}^\top\mathbf{X}\mathbf{w}\right) &= 0, && (3.52) \\ -2\mathbf{y}^\top\mathbf{X} + 2\mathbf{X}^\top\mathbf{X}\mathbf{w} &= 0, && (3.53) \\ \mathbf{X}^\top\mathbf{X}\mathbf{w} &= \mathbf{y}^\top\mathbf{X}. && (3.54) \end{aligned}$$

When $(\mathbf{X}^\top\mathbf{X})^{-1}$ exists (usually when N > d), we can set $\mathbf{w}^* = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{y}^\top\mathbf{X}$.

Gradient Descent

Alternatively, we can also use iterative optimization techniques such as gradient descent. Gradient descent is particularly useful in scenarios where the matrix $\mathbf{X}$ is large or poorly conditioned, making the computation of the inverse matrix $(\mathbf{X}^\top\mathbf{X})^{-1}$ computationally expensive or numerically unstable. In gradient descent, the coefficients are updated iteratively according to the rule:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta\nabla\text{SSR}(\mathbf{w}_{\text{old}}), \tag{3.55}$$

where η is the learning rate—a small positive scalar determining the step size at each iteration, and $\nabla\text{SSR}(\mathbf{w})$ is the gradient of the sum of squared residuals, given by:

$$\begin{aligned}\nabla\text{SSR}(\mathbf{w}) &= -2\mathbf{y}^\top\mathbf{X} + 2\mathbf{X}^\top\mathbf{X}\mathbf{w} &(3.56)\\ &= -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) &(3.57)\\ &= -2\mathbf{X}^\top(\mathbf{y} - \hat{\mathbf{y}}). &(3.58)\end{aligned}$$

By repeatedly adjusting $\mathbf{w}$ in the direction that most rapidly decreases the SSR, gradient descent seeks to find the minimum of the SSR landscape, ideally converging to the same solution as the OLS method in cases where the solution is unique.

Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is an extension of the gradient descent optimization technique that is widely used in machine learning, especially when dealing with large datasets. Unlike the traditional gradient descent approach, which computes the gradient of the entire dataset to perform a single update, SGD updates the parameters more frequently based on the gradient of a randomly selected subset of the data (i.e., batch) at each iteration.

The formulation for SGD can be expressed as follows:

$$\mathbf{w}_{\text{new}} = \mathbf{w}_{\text{old}} - \eta\nabla\text{SSR}_{\text{batch}}(\mathbf{w}_{\text{old}}), \tag{3.59}$$

where $\nabla\text{SSR}_{\text{batch}}(\mathbf{w})$ represents the gradient computed not over the entire dataset but over a randomly chosen subset of the data, often referred to as a batch. The learning rate η still plays the role of a step-size parameter.

This method introduces randomness into the optimization process, which has several beneficial effects:

- Reduction in Computation Overhead: Each update is computationally cheaper, as it uses only a fraction of the data. This is particularly advantageous for large datasets.
- Escape from Local Minima: The inherent noise added by the random selection of data helps the algorithm to potentially escape from local minima, a common issue in complex optimization scenarios.

SGD has proven especially useful in training deep learning models, where the size of the dataset and the complexity of the model make traditional batch gradient descent computationally impractical. However, its performance is highly dependent on the choice of the initial parameters, the learning rate, and the batch size, requiring careful tuning to achieve the best results.

The pseudo-code for a basic SGD algorithm is as follows:

Probabilistic Perspective of Linear Regression

Linear regression can also be interpreted from a probabilistic perspective. This approach involves viewing the outputs y not just as direct outcomes but as random variables influenced by the predictors $\mathbf{x}$ and some inherent randomness in the system.

Algorithm 3.1 Stochastic Gradient Descent (SGD) for Linear Regression

```
1: Initialize the weight vector $\mathbf{w}$ randomly
2: for each epoch do
3:     for each batch in the dataset do
4:         Calculate the gradient: $\Delta = \nabla \mathrm{SSR}_{\mathrm{batch}}(\mathbf{w})$
5:         Update the weights: $\mathbf{w} = \mathbf{w} - \eta\Delta$
6:     end for
7: end for
```

In the probabilistic model of linear regression, we assume that the conditional distribution of the dependent variable y, given the predictors $\mathbf{x}$, is normally distributed. This is expressed mathematically as:

$$y \mid \mathbf{x} \sim \mathcal{N}(\mu, \sigma^2), \tag{3.60}$$

where $\mu = \beta^\top \mathbf{x} + b$ is the mean, which is modeled by the linear predictor, and σ^2 is the variance, which is assumed to be constant across all observations. Here, $\mathcal{N}$ denotes the normal distribution.

The likelihood of observing the data $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$ under this model can be formulated as the product of the probability densities of each observed $y^{(i)}$, assuming independence between the observations:

$$L(\beta, b) = \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y^{(i)} - (\beta^\top \mathbf{x}^{(i)} + b))^2}{2\sigma^2}\right). \tag{3.61}$$

The parameters β and b can be estimated by maximizing the likelihood function. However, it’s often more practical to maximize the log of the likelihood function, which simplifies the computations:

$$\log L(\beta, b) = -\frac{N}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(y^{(i)} - (\beta^\top \mathbf{x}^{(i)} + b))^2. \tag{3.62}$$

Maximizing this log-likelihood function with respect to β and b is equivalent to minimizing the sum of squared residuals (SSR), showing a direct connection between the probabilistic and classical approaches.

3.3.2 Model Validation

Model validation is an essential step in the machine learning workflow, ensuring that a model performs well on unseen data and generalizes beyond the specific examples used during training. This process helps in assessing the model’s practical applicability and in avoiding issues like overfitting, where a model learns the details and noise in the training data to an extent that it negatively impacts the performance on new data.

Holdout Method

The most straightforward method is the holdout method, where the data set is divided into two sets, typically called the training set and the testing set. The training set is used to train the model, while the testing set is used to evaluate its performance. This method is particularly useful when the dataset is large enough to be confidently split into two sets that can represent the variability of the data sufficiently.

Cross-Validation

Cross-validation is a robust statistical method used to evaluate the generalizability of models. It involves partitioning the data into complementary subsets, performing the analysis on one subset (called the training set), and validating the analysis on the other subset (used as the validation set). To reduce variability, multiple rounds of cross-validation are performed using different partitions, and the validation results are averaged over the rounds.

One common method is *k-fold cross-validation*, where the original sample is randomly partitioned into k equal-sized subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining $k - 1$ subsamples are used as training data. The cross-validation process is then repeated k times, with each of the k subsamples used exactly once as the validation data. The k results can then be averaged to produce a single estimation.

3.3.3 Model Evaluation

Model evaluation metrics are crucial for comparing the performance of different machine learning models and choosing the best one for your specific context. These metrics differ depending on the type of model and the specific objectives of the application.

Regression Metrics

For regression tasks, common metrics include:

- **Mean Absolute Error (MAE)**: The average of the absolute differences between predictions and actual observations. It gives an idea of how wrong the predictions were; the lower the MAE, the better.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{3.63}$$

- **Mean Squared Error (MSE)**: Similar to MAE but squares the differences before averaging them. It penalizes larger errors more than MAE, making it more sensitive to outliers.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.64}$$

- **R-squared (Coefficient of Determination)**: Provides an indication of goodness of fit and therefore a measure of how well unseen samples are likely to be predicted by the model, through the proportion of total variation of outcomes explained by the model.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{3.65}$$

Classification Metrics

For classification problems, the choice of metrics might include:

- **Accuracy**: The proportion of true results (both true positives and true negatives) among the total number of cases examined.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{3.66}$$

- **Precision and Recall**: Precision is a measure of result relevancy, while recall is a measure of how many truly relevant results are returned.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.67}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.68}$$

- **F1 Score**: A weighted average of precision and recall. This score takes both false positives and false negatives into account. Intuitively, it is not as easy to understand as accuracy, but it is usually more useful when dealing with uneven class distribution.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.69}$$

Advanced Metrics

In more complex scenarios, advanced metrics like Area Under the Receiver Operating Characteristic Curve (AUC-ROC), Area Under the Precision Recall Curve (AUC-PRC), log-loss, or custom metrics defined for specific business needs might be used. These provide deeper insights into the model's performance, especially in cases involving probabilistic predictions.

By systematically applying these validation and evaluation techniques, practitioners can significantly enhance the reliability and effectiveness of their machine learning models, leading to better decision-making and more impactful outcomes in real-world applications.

Questions

1. TODO