1 Vector and Basis Review

1. Vector: n * 1

   Vector with arrow: $\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ Adding and Scaling: $c\vec{x} + \vec{y} = \begin{bmatrix} cx_1 + y_1 \\ cx_2 + y_2 \\ cx_3 + y_3 \end{bmatrix}$

2. Linear Combinations:

$$\vec{x} = a_1\vec{v_1} + ... + a_n\vec{v_n} = \sum_{i=1}^{N} \vec{v_i}$$

   A line formed when the sum equals 1.

   When $\sum_{i=1}^{N} \mathbf{a_i} = \mathbf{1}$ is **Affine Combination**, x is on the line

   When $\sum_{i=1}^{N} \mathbf{a_i} < \mathbf{1}$ x is under the line

   When $\sum_{i=1}^{N} \mathbf{a_i} > \mathbf{1}$ x is outside the line

   When $\sum_{i=1}^{N} \mathbf{a_i} = \mathbf{1}, a_i \geq 0$ is **Convex Combination**

   If $a_i \geq 0$, lies in the convex hull of the vectors. If $a_i < 0$, lies outside the convex hull of the vectors.

   For example, vectors (S = (1,1),(2,3),(3,1),(2,2)) in 2D, and it has a triangle convex hull (1,1),(2,3),(3,1). If equal or larger than 0, it will always lie in this triangle.

3. Span : any vector can use $\alpha\vec{v} + \beta\vec{u}$

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n) = \{\mathbf{w} \mid \mathbf{w} = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_n\mathbf{v}_n,\ c_i \in \mathbb{R}\}$$

   Linearly Independent: When $\sum_{i=1}^{N} \alpha_i\vec{v_i} = \vec{0} \Rightarrow a_i, \cdots, a_n = 0$

   Linearly Dependent: When $\sum_{i=1}^{N} \alpha_i\vec{v_i} = \vec{0} \Rightarrow a_i, \cdots, a_n \neq 0$

4. Inner Product

   The inner product of two vectors $\mathbf{u} = [u_1, u_2, \ldots, u_n]$ and $\mathbf{v} = [v_1, v_2, \ldots, v_n]$ in $\mathbb{R}^n$ is defined as:

$$\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^{n} u_i v_i = \vec{u}^T \vec{v}$$

   The inner product satisfies:

   1. Bilinearity: $\langle c\mathbf{u} + b\mathbf{z}, \mathbf{v} \rangle = c\langle \mathbf{u}, \mathbf{v} \rangle + b\langle \mathbf{z}, \mathbf{v} \rangle$
   2. Symmetry: $\langle \mathbf{u}, \mathbf{v} \rangle = \langle \mathbf{v}, \mathbf{u} \rangle$.
   3. Positivity(Nonnegativity): $\langle \mathbf{u}, \mathbf{u} \rangle \geq 0, \quad \sum_{i=1}^{N} u_i^2 \geq 0, \quad \langle \mathbf{u}, \mathbf{u} \rangle = 0$ if and only if $\mathbf{u} = \mathbf{0}$.

   The relationship with the angle θ between $\mathbf{u}$ and $\mathbf{v}$ is given by:

$$\langle \mathbf{u}, \mathbf{v} \rangle = \|\mathbf{u}\| \|\mathbf{v}\| \cos\theta,$$

   where $\|\mathbf{u}\|_2 = \sqrt{\langle \mathbf{u}, \mathbf{u} \rangle} = \sqrt{\sum_{i=1}^{n} u_i^2}$ is the norm of $\mathbf{u}$.

   Unit Vector: Vector with norm 1. (Unit norm).

   Normalizing a vector $\tilde{\mathbf{x}}$: $\frac{\tilde{\mathbf{x}}}{\|\tilde{\mathbf{x}}\|_2}$

Projecting a vector $\vec{x}$ onto $\vec{y}$: $\langle \vec{x}, \frac{\vec{y}}{\|\vec{y}\|_2} \rangle \frac{\vec{y}}{\|\vec{y}\|_2} = \frac{\vec{x} \cdot \vec{y}}{\|\vec{y}\|_2^2} \vec{y}$

Sign of $\langle \vec{x}, \vec{y} \rangle$ is positive if angle is less than 90 degrees, and negative otherwise.

4. Linear Subspace: A lower-dimensional slice of the vector space that passes through the origin. Like 2D plane or 1D line in 3D Euclidean space.

5. Basis:

a. Linear Independent.

$$c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + \cdots + c_n \mathbf{v}_n = \mathbf{0}.$$

The vectors are linearly independent if:

$$c_1 = c_2 = \cdots = c_n = 0.$$

b. Span the vector Space (span the full or sub space).

$$\text{Span}(\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n) = \{\mathbf{w} \mid \mathbf{w} = c_1 \mathbf{v}_1 + c_2 \mathbf{v}_2 + \cdots + c_n \mathbf{v}_n,\, c_i \in \mathbb{R}\}$$

6. Orthogonal Basis: An orthogonal basis is a basis whose vectors are orthogonal to one another. (easy to compute the coordinates with respect to the basis)

Two vector $\vec{x}, \vec{y}$ are orthogonal if and only if $\langle \vec{x}, \vec{y} \rangle = 0$

$$\vec{x} = \alpha_1 \vec{v_1} + \cdots + \alpha_n \vec{v_n}$$

$$\langle \vec{x}, \vec{v_j} \rangle = \langle \sum_{i=1}^{N} \alpha_i \vec{v_i}, \vec{v_j} \rangle = \alpha_j \langle \vec{v_j}, \vec{v_j} \rangle$$

$$a_j = \frac{\langle \vec{x}, \vec{v_j} \rangle}{\|\vec{v_j}\|_2^2}$$

7. Orthonormal Basis; Orthogonal basis with unit vectors.

$$\langle \vec{x}, \vec{v_j} \rangle = \langle \sum_{i=1}^{N} \alpha_i \vec{v_i}, \vec{v_j} \rangle = \alpha_j \langle \vec{v_j}, \vec{v_j} \rangle$$

$$a_j = \frac{\langle \vec{x}, \vec{v_j} \rangle}{\|\vec{v_j}\|_2^2}, \text{ where} \|\vec{v_j}\|_2 = 1$$

If not equal to one, need to do normalization: $\frac{\vec{v_i}}{\|\vec{v_i}\|_2}$

8. Standard Basis: is an orthonormal basis with 1 on diagonal matrix. For example:

$$I = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Summary:

Basis ⊃ Orthogonal Basis ⊃ Orthonormal Basis ⊃ Standard Basis.

2 Matrices Review

1. Matrix:

A matrix is a rectangular array of numbers arranged in rows and columns:

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{bmatrix}.$$

2. Matrix Type:

a. Row Matrix:

$$A = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}.$$

b. Column Matrix:

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}.$$

c. Square Matrix:

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}.$$

d. Diagonal Matrix:

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}.$$

e. Identity Matrix:

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

f. Symmetric Matrix:

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}.$$

g. Orthogonal Matrices:

$$A^T A = AA^T = I$$

$$A = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} .\text{where determinant is 1}$$

$$A = \begin{bmatrix} cos\theta & sin\theta \\ sin\theta & -cos\theta \end{bmatrix} .\text{where determinant is -1}$$

3. Matrix Operations

a.. Addition:

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

b. Scalar Multiplication:

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

c. Transpose (row and column change):

$$A^T = \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{bmatrix}.$$

d. Determinant of a 2×2 Matrix

$$\det(A) = \begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc.$$

e. Inverse of a 2×2 Matrix

$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

When $AA^{-1} = I$:

1. Only square and full rank matrix can do this
2. If inverse exist, it is unique, full rank, linearly independent, a span of R^n, basis of R^n)
3. $(A^{-1})^{-1} = A$
4. $(A^{-1})^{-1} = (A^{-1})^T$

f. Rank

The rank of a matrix is the maximum number of linearly independent rows or columns.
Like Row Reduction, do some calculation to make the matrix like this

$$\begin{bmatrix} 1 & a_2 & a_3 \\ 0 & b_2 & b_3 \\ 0 & 0 & c_2 \end{bmatrix}.$$

Count the nonzero rows:

$$\text{Rank}(A) = 2.$$

FULL Rank: full rank if the rank of the matrix equals the smallest of its dimensions (**min(m,n)**, where m is the number of rows and n is the number of columns)

Rank Deficiency: rank(A) < min(m,n)

g. Inner Product(dot product - scalar)
The **inner product** of two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ is defined as:

$$\mathbf{u} \cdot \mathbf{v} = \sum_{i=1}^{n} u_i v_i = \mathbf{u}^T \mathbf{v}.$$

h. Outer Product
The **outer product** of two vectors $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$ is defined as:

$$\mathbf{u} \otimes \mathbf{v} = \mathbf{u}\mathbf{v}^T.$$

3 Singular Value Decomposition (SVD)

The Singular Value Decomposition (SVD) of an $m \times n$ matrix A is given by:

$$A = U\Sigma V^T,$$

where:

- U (left-singular vector) is an $m \times m$ orthogonal matrix (rotation/reflection),
- Σ is an $m \times n$ diagonal matrix containing the singular values with real non-negative entries (scaling along axes),
- V^T (right-singular vector) is the transpose of an $n \times n$ orthogonal matrix (rotation/reflection).

The steps to compute SVD are as follows:

1. **Compute A^TA and AA^T:**
   - Calculate A^TA (an $n \times n$ matrix).
   - The eigenvectors of A^TA form the columns of V.
   - Calculate AA^T (an $m \times m$ matrix).
   - The eigenvectors of AA^T form the columns of U.
2. **Compute Singular Values:**
   - The singular values are the square roots of the eigenvalues ($\sigma_i = \sqrt{\lambda}$) of A^TA (or equivalently AA^T). The number of λ would be the rank for A.
   - Arrange the singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ in descending order.
   - Place these singular values along the diagonal of Σ, with the rest of the entries being zero.
3. **Construct V:**
   - Solve $(A^TA)\mathbf{v}_i = \lambda_i \mathbf{v}_i$ to compute the eigenvectors of A^TA.
   - The eigenvectors form the columns of V.
4. **Construct U:**
   - Solve $(AA^T)\mathbf{u}_i = \lambda_i \mathbf{u}_i$ to compute the eigenvectors of AA^T.
   - The eigenvectors form the columns of U.
5. **Verify the Decomposition:**
   - Multiply $U\Sigma V^T$ to confirm that it reconstructs A, ensuring the correctness of the decomposition.

4 Eigendecomposition

Eigendecomposition is a matrix factorization technique where a **square symmetric** matrix A is decomposed as:

$$A = P\Lambda P^{-1} = P\Lambda P^T,$$

where:

- P: A matrix whose columns are the eigenvectors of A (Rotation),
- Λ: A diagonal matrix whose diagonal entries are the eigenvalues of A. It can be be negative (Scaling/reflection along axes)
- P^{-1}: The inverse of P (Reverse rotation).

For symmetric matrices, the singular values are the absolute values of eigenvalues and the left- and right-singular vectors are positive or negative the eigenvectors.
Same way to calculate as SVD but just need to calculate P. But the $\Lambda = \lambda$. **No need to calculate the square root** of it as SVD.

2. Symmetric Matrices and Transformations
   All symmetric matrices can be decomposed into a sequence of three transformations:
   1. **Rotation:**
      The eigenvectors of A define the new coordinate axes after rotation.
   2. **Scaling or Reflection Along Axes:**
      The eigenvalues in Λ determine how much to scale or reflect along these new axes.
   3. **Reverse Rotation:**
      After scaling or reflection, the original coordinate system is restored via a reverse rotation.

- A symmetric matrix A performs a combination of scaling or reflection along directions that align with its eigenvectors.
- The eigenvalues of A determine the scaling factor (magnitude and direction):
  - **Positive eigenvalues indicate scaling in the same direction as the eigenvector**.
  - **Negative eigenvalues indicate reflection and scaling in the opposite direction**.
- Directions along which the scaling occurs are the eigenvectors.

5 Norms

1. p-Norms for Vectors

The p-norm of a vector $\mathbf{x} \in \mathbb{R}^n$ is defined as:

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}, \quad p \geq 1.$$

Common Vector Norms:

- **1-Norm (Manhattan Norm):**

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{n} |x_i|.$$

- **2-Norm (Euclidean Norm):**

$$\|\mathbf{x}\|_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{1/2}.$$

- **Infinity Norm (∞-Norm, Maximum Norm):**

$$\|\mathbf{x}\|_\infty = \max_{1 \leq i \leq n} |x_i|.$$

2. Matrix Norms

A matrix norm $\|A\|$ measures the size of a matrix A. Common matrix norms include:

- **Frobenius Norm:**
$$\|A\|_F = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2}.$$

- **Induced/operator norms**
$$\|A\|_p = sup_{\|\vec{x}_p=1\|}\{\|A\vec{x}\|_p\},$$

- **2-Norm (Spectral Norm):**
$$\|A\|_2 = sup_{\|\vec{x}_2=1\|}\{\|A\vec{x}\|_2\} = \sigma_{1,1}(A) = \sqrt{\lambda_{\max}(A^T A)},$$
where $\sigma_{1,1}(A)$ is the largest singular of A.

6 Positive/Negative (Semi-)Definite Matrices

Let $A \in \mathbb{R}^{n \times n}$ be a **symmetric matrix** ($A = A^T$). The matrix A is classified as follows:

1.Using Eigenvalues

The definiteness of A can be determined by its eigenvalues:

- Positive Definite: $\lambda_i > 0 \, \forall i$,
- Positive Semi-Definite: $\lambda_i \geq 0 \, \forall i$,
- Negative Definite: $\lambda_i < 0 \, \forall i$,
- Negative Semi-Definite: $\lambda_i \leq 0 \, \forall i$,
- Indefinite: Mixed signs ($\exists \lambda_i > 0$, $\exists \lambda_j < 0$).

2. Quadratic Forms

When **A is symmetric Matrix**

- **Positive Definite (Strictly Convex Function):**
A is positive definite if:
$$\mathbf{x}^T A \mathbf{x} > 0 \quad \forall \, \mathbf{x} \neq 0.$$
$$\alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}) > f\big(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}\big)$$

- **Positive Semi-Definite (Convex Function):**
A is positive semi-definite if:
$$\mathbf{x}^T A \mathbf{x} \geq 0 \quad \forall \, \mathbf{x}.$$
$$\alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}) \geq f\big(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}\big)$$

- **Negative Definite (Strictly Concave Function):**
A is negative definite if:
$$\mathbf{x}^T A \mathbf{x} < 0 \quad \forall \, \mathbf{x} \neq 0.$$
$$\alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}) < f\big(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}\big)$$

- **Negative Semi-Definite(Concave Function):**
A is negative semi-definite if:
$$\mathbf{x}^T A \mathbf{x} \leq 0 \quad \forall \, \mathbf{x}.$$
$$\alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}) \leq f\big(\alpha \mathbf{x} + (1-\alpha)\mathbf{y}\big)$$

- **Indefinite:**
  A is indefinite if:
$$\mathbf{x}^T A\mathbf{x} > 0 \quad \text{for some } \mathbf{x}, \quad \mathbf{x}^T A\mathbf{x} < 0 \quad \text{for other } \mathbf{x}.$$

It's $x^T Ax = \langle x, Ax \rangle$

When **A is non-symmetric Matrix**
A quadratic form is given by:
$$Q(\mathbf{x}) = \mathbf{x}^T A\mathbf{x},$$
where $\mathbf{x} \in \mathbb{R}^n$ and A is an $n \times n$ matrix.

Non-Symmetric Matrices

If A is not symmetric, it can be decomposed into:
$$A = S + N,$$
where:

- $S = \frac{A+A^T}{2}$ is the **symmetric part** of A,
- $N = \frac{A-A^T}{2}$ is the **skew-symmetric part** of A, satisfying $N^T = -N$.

Effect on Quadratic Forms

The quadratic form can be rewritten as:
$$Q(\mathbf{x}) = \mathbf{x}^T S\mathbf{x} + \mathbf{x}^T N\mathbf{x}.$$

Skew-Symmetric Part Contribution

For the skew-symmetric part N, we have:
$$\mathbf{x}^T N\mathbf{x} = 0 \quad \text{for all } \mathbf{x},$$
because:
$$\frac{1}{2}\mathbf{x}^T A\mathbf{x} - \frac{1}{2}\mathbf{x}^T A^T\mathbf{x}.$$
$$\frac{1}{2}\mathbf{x}^T A\mathbf{x} - \frac{1}{2}(A\mathbf{x})^T\mathbf{x}.$$
$$\frac{1}{2}\langle \mathbf{x}, A\mathbf{x}\rangle - \frac{1}{2}\langle A\mathbf{x}, \mathbf{x}\rangle.$$
$$\langle \mathbf{x}, A\mathbf{x}\rangle = -\langle A\mathbf{x}, \mathbf{x}\rangle.$$
Therefore, the contribution of the skew-symmetric part to the quadratic form is:
$$\mathbf{x}^T \frac{A - A^T}{2}\mathbf{x} = 0.$$
Thus, the quadratic form reduces to:
$$Q(\mathbf{x}) = \mathbf{x}^T S\mathbf{x},$$
where $S = \frac{A+A^T}{2}$ determines the behavior of the quadratic form.

For any matrix A:
$$A^T A \geq 0 \quad \text{(i.e., } A^T A \text{ is } \textbf{positive semi-definite)}.$$
To show by using Euclidean Norm:
$$\mathbf{x}^T(A^T A)\mathbf{x} = (\mathbf{x}^T A^T)(A\mathbf{x}) = (A\mathbf{x})^T(A\mathbf{x}) = \langle A\mathbf{x}, A\mathbf{x}\rangle = \|A\mathbf{x}\|_2^2 \geq 0 \quad \forall A, \mathbf{x}.$$

Conclusion

For non-symmetric matrices, the quadratic form $Q(\mathbf{x}) = \mathbf{x}^T A\mathbf{x}$ depends only on the symmetric part $S = \frac{A+A^T}{2}$. The skew-symmetric part $N = \frac{A-A^T}{2}$ does not contribute to the quadratic form.

7 Taylor Expansion

1. Definition

For a function $f(x)$ that is infinitely differentiable at a point a, the Taylor expansion of $f(x)$ around a is given by:

$$f(x) = f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \frac{f^{(3)}(a)}{3!}(x-a)^3 + \cdots$$

Or in compact form:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n,$$

where:

- $f^{(n)}(a)$: The n-th derivative of $f(x)$ evaluated at a,
- $n!$: The factorial of n ($n! = 1 \cdot 2 \cdot \cdots \cdot n$).

2. Maclaurin Series (Special Case)

If $a = 0$, the Taylor series becomes:

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(0)}{n!}x^n.$$

8 Jensen's Inequality

By definition of convexity:

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y), \quad \forall\, \lambda \in [0,1].$$

Extend this property to expectations:

$$f\left(\sum_{i=1}^{n} p_i x_i\right) \leq \sum_{i=1}^{n} p_i f(x_i),$$

where:

$$\sum_{i=1}^{n} p_i = 1, p_i \geq 0$$

9 Checking for Local Minimum, Maximum, or Saddle Point

1. First-Order Condition

To find critical points of a function $f(\mathbf{x})$, compute the gradient:

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}.$$

Solve $\nabla f(\mathbf{x}) = 0$ to locate the stationary points.

2. Second-Order Condition

At each stationary point, compute the Hessian matrix:

$$H(f)(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

Key Criteria:

Then Calculate the eigenvalue

- **Positive definite:** Local minimum.
- **Negative definite:** Local maximum.
- **Indefinite:** Saddle point.
- **Positive/negative semi-definite:** Inconclusive.

10 Ordinary Least Squares (OLS)

The method in linear regression used to estimate the coefficients of a linear model by minimizing the sum of squared differences between the observed and predicted values.

$$\mathbf{y} = \tilde{\mathbf{w}}^{\mathbf{T}} X + b$$

The goal is to minimize the sum of squared residuals:

$$L(\vec{w}) = \min_{\tilde{\mathbf{w}}} \|\mathbf{y} - X\tilde{\mathbf{w}}\|_2^2.$$

If X is **full-rank**.

$$\tilde{\mathbf{w}} = (X^T X)^{-1} X^T \mathbf{y}.$$

It always local and global minimum sine

$$\frac{\partial^2 L}{\partial w \partial w^\top} = \frac{\partial}{\partial w}\left(\frac{\partial L}{\partial w}\right).$$

$$\frac{\partial^2 L}{\partial w \partial w^\top} = 2X^\top X \succeq 0.$$

Since the Hessian $2X^\top X$ is always positive semi-definite ($\succeq 0$), the loss function $L(w)$ is **convex**.

- **Training:** The process of finding the optimal parameters w^*.
- **Testing:** The process of computing predictions $\hat{y}$ for new input $\mathbf{x}$ using:

$$\hat{y} = w^{*\top}\mathbf{x} + b.$$

Pseudoinverse

Let $A = U\Sigma V^T$, where:

- $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices,
- $\Sigma \in \mathbb{R}^{m\times n}$ is a diagonal matrix with singular values $\sigma_1, \sigma_2, \ldots$.

The pseudoinverse is computed as:

$$A^+ = V\Sigma^+ U^T,$$

where Σ^+ is the pseudoinverse of Σ, obtained by:

$$\Sigma^+ = \begin{bmatrix} \frac{1}{\sigma_1} & 0 & \cdots \\ 0 & \frac{1}{\sigma_2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}.$$

Summary:

- **When $A^+ = (X^TX)^{-1}X^T$:**
  - X must have full column rank (i.e., the columns of X are linearly independent).
  - X^TX must be invertible (nonsingular).
- **When $A^+ \neq (X^TX)^{-1}X^T$:**
  - X is rank-deficient (i.e., X^TX is singular or not invertible).
  - X is underdetermined ($m < n$), where X has more columns than rows.
  - In these cases, the general pseudoinverse is computed using the Singular Value Decomposition (SVD):

$$A^+ = V\Sigma^+U^T,$$

    where:
    - $X = U\Sigma V^T$ is the SVD of X,
    - Σ^+ is obtained by taking the reciprocal of the nonzero singular values in Σ and transposing.
- **General Pseudoinverse:** The pseudoinverse A^+ is always well-defined regardless of the rank or shape of X, but the formula $(X^TX)^{-1}X^T$ applies only under the specific conditions mentioned above.

Feature	**Multiple-Output Linear Regression**	**Ordinary Least Squares (OLS)**
Target Variable	$Y \in \mathbb{R}^{n\times k}$	$y \in \mathbb{R}^n$
Regression Coefficients	$W \in \mathbb{R}^{p\times k}$	$\vec{w} \in \mathbb{R}^p$
Loss Function	$\|Y - XW\|_F^2$	$\|y - X\vec{w}\|_2^2$
Solution	$W^* = (X^TX)^{-1}X^TY$	$\vec{w}^* = (X^TX)^{-1}X^Ty$
Outputs	Multiple outputs (simultaneous prediction)	Single output
Applications	Multi-task regression, predict multiple outputs	Univariate regression

If singular values X of OLS are small, it will be overfitting since the σ will nearly to be zero, and become rank deficiency. Then, $\lambda_{X^+} = ...\frac{1}{\sigma}$, so $\|X^+\|_2$ large will also cause OLS overffiting.
Polynomial Features allow the model prediction to depend non-linearly on the data without changing the training procedure.(do everything same but change x into non-linear one $\phi(x)$)
The model for the prediction is:

$$\hat{y} = w^\top \phi(\mathbf{x}),$$

where the feature map $\phi(\mathbf{x})$ is defined as:

$$\phi(\mathbf{x}) \begin{bmatrix} \phi_1(\mathbf{x}) \\ \vdots \\ \phi_n(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \log(x_1) \\ \vdots \\ \log(x_n) \end{bmatrix}.$$

11 Overfitting vs Underfitting

Aspect	**Overfitting**	**Underfitting**
Key Issue	**too complex** for the given data.	**too simple** for the given data.
Causes	1. Too many features or parameters. 2. Excessive training time. 3. Insufficient training data relative to complexity.	1. Too few features or parameters. 2. Insufficient training time. 3. Low model complexity.
Training Error	*Low*	*High*
Validation/Test Error	*High*	*High*
Solutions	1. Add (L1) Lasso, (L2) Ridge. regularization to penalize large weight 2. Reduce model complexity. 3. Increase **training data size**. 4. Stop training earlier (early stopping).	1. Increase model complexity. 2. Add more relevant features. 3. Improve data quality or diversity. 4. Train for a longer time.

12 K-Fold Cross Validation

1. Process

- Divide the dataset into K non-overlapping subsets (folds).
- For each fold i (where $i = 1, 2, \ldots, K$):
  - Use the i-th fold as the testing set.
  - Use the remaining $K - 1$ folds as the training set.
  - Train the model on the training set and evaluate on the testing set to compute the metric M_i.
- Compute the overall performance as:

$$\text{Overall Metric} = \frac{1}{K} \sum_{i=1}^{K} M_i.$$

2. Variants of K-Fold Cross Validation

- **Stratified K-Fold:** Ensures class distributions are consistent across folds.
- **Leave-One-Out Cross Validation (LOOCV):** Special case where $K = N$, the number of data points.
- **Repeated K-Fold:** Repeats K-Fold with different splits for robust evaluation.

13 Ridge Regression vs Ordinary Least Squares

Aspect	**Ridge Regression**	**Ordinary Least Squares (OLS)**
Key Feature	**Adds regularization** to reduce overfitting.	**No regularization**; minimizes residuals directly.
Regularization	$\lambda \sum_{j=1}^{p} w_j^2$	None.
Bias-Var Tradeoff	1. Adds **bias** to reduce **variance**. 2. Handles multicollinearity.	1. **Low bias** but higher variance. 2. Struggles with multicollinearity.
Hyperparameter	**Requires tuning** of λ for optimal performance.	No hyperparameters to tune.
Coefficients	Shrunk towards zero based on λ.	Calculated directly from least squares.
Use Cases	1. Suitable for large predictor sets. 2. Preferred with multicollinearity.	1. Effective with small, clean datasets. 2. Works when predictors are uncorrelated.
Equation Solved	$\min \|y - X\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2$	$\min \|y - X\mathbf{w}\|^2$
Effect of $\lambda = 0$	**Reduces to OLS.**	No equivalent adjustment.
result	$\mathbf{w}_{\text{Ridge}} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$	$\mathbf{w}_{\text{OLS}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$

14 Discrete Random Variable Vs Continuous Random Variable

Aspect	Discrete Random Variable	Continuous Random Variable
Probability	**PMF**: $P(X = x)$.	**PD)**: $f_X(x)$.
Key Property	$P(X = x)$ gives exact probabilities for specific values x.	$P(X = x) = 0$; probabilities are calculated over intervals.
CDF	$F_X(x) = P(X \leq x)$, a step function.	$F_X(x) = \int_{-\infty}^{x} f_X(t)dt$, a continuous function.
Graph	a bar graph or histogram.	a smooth curve of the PDF.
Expected Value	$E[X] = \sum_x x \cdot P(X = x)$.	$E[X] = \int_{-\infty}^{\infty} x \cdot f_X(x)dx$.
Applications	Modeling countable events. Dice rolls, defect counts, etc.	Modeling measurements. Temperatures, durations, etc.

15 Common Discrete Distributions

15.1 Bernoulli Distribution

- Random Variable: $X \sim \text{Bernoulli}(p)$
- Probability mass function:

$$p_X(x) = \Pr(X = x) = \begin{cases} p & \text{if } x = 1, \\ 1 - p & \text{if } x = 0. \end{cases}$$

- More mathematically convenient form:

$$p_X(x) = \Pr(X = x) = p^x(1 - p)^{1-x}.$$

- Expected Value:

$$\mathbb{E}[X] = p$$

- Variance:

$$\text{Var}(X) = p(1 - p)$$

15.2 Uniform Distribution

- Random Variable: $X \sim \text{Uniform}(a, b)$
- Probability density function:

$$f_X(x) = \begin{cases} \frac{1}{b-a} & \text{if } x \in [a, b], \\ 0 & \text{if } x \notin [a, b]. \end{cases}$$

- Support:

$$\text{supp}(X) = [a, b]$$

- Plots:
  - The PDF is constant over $[a, b]$.
  - The CDF is a linear function increasing from 0 to 1 in the range $[a, b]$.

$$\mathbb{E}[X] = \frac{a + b}{2}$$

- Variance:

$$\text{Var}(X) = \frac{(b - a)^2}{12}$$

15.3 Categorical Distribution

- Random Variable: $X \sim \text{Categorical}(p_1, p_2, \ldots, p_k)$
- Probability mass function:

$$p_X(x) = \Pr(X = x) = \begin{cases} p_1 & \text{if } x = 1, \\ p_2 & \text{if } x = 2, \\ \vdots & \vdots \\ p_k & \text{if } x = k. \end{cases}$$

- More mathematically convenient form:

$$p_X(x) = \Pr(X = x) = \prod_{i=1}^{k} p_i^{[x=i]}, \quad \text{where } [x = i] = \begin{cases} 1 & \text{if } x = i, \\ 0 & \text{if } x \neq i. \end{cases}$$

- Expected Value (for $X \in \{1, 2, \ldots, k\}$):

$$\mathbb{E}[X] = \sum_{i=1}^{k} i \cdot p_i$$

- Variance:

$$\text{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$$

  where:

$$\mathbb{E}[X^2] = \sum_{i=1}^{k} i^2 \cdot p_i$$

16 Common Continuous Distributions

Normal Distribution

- Random Variable:

$$X \sim \mathcal{N}(\mu, \sigma^2)$$

- Probability Density Function (PDF):

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

  or equivalently:

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

- Support:

$$\text{supp}(X) = \mathbb{R}$$

$$\frac{X - \mu}{\sigma} \sim \mathcal{N}(0, 1)$$

- Expected Value:

$$\mathbb{E}[X] = \mu$$

- Variance:

$$\text{Var}(X) = \sigma^2$$

17 Chain Rule of Probability

Two random variables:

$$p(y \mid x) = \frac{p(x, y)}{p(x)} \implies p(x, y) = p(x)p(y \mid x)$$

$$\text{Independent: } p(x, y) = p(x)p(y)$$

In general:

$$p(x_1, \ldots, x_n) = p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \cdots p(x_n \mid x_1, \ldots, x_{n-1})$$

$$\text{Independent: } p(x_1, \ldots, x_n) = p(x_1) \cdots p(x_n)$$

18 Chain Rule of Probability (Conditional Case)

Two random variables:

$$p(y \mid x, z) = \frac{p(x, y \mid z)}{p(x \mid z)} \implies p(x, y \mid z) = p(x \mid z)p(y \mid x, z)$$

$$\text{Independent: } p(x, y|z) = p(x|z)p(y|z)$$

In general:

$$p(x_1, \ldots, x_n \mid z_1, \ldots, z_l) = p(x_1 \mid z_1, \ldots, z_l)p(x_2 \mid x_1, z_1, \ldots, z_l) \cdots p(x_n \mid x_1, \ldots, x_{n-1}, z_1, \ldots, z_l)$$

$$\text{Independent: } p(x_1, \ldots, x_n \mid z_1, \ldots, z_l) = p(x_1 \mid z_1, \ldots, z_l) \cdots p(x_1 \mid z_1, \ldots, z_l)$$

20. Bayes' Rule

Bayes' Rule, also known as Bayes' Theorem, is a fundamental concept in probability theory. It provides a way to update the probability of a hypothesis H in light of new evidence E. The formula is given as:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

Where:

- $P(H \mid E)$: The posterior probability, or the probability of the hypothesis H given the evidence E.
- $P(E \mid H)$: The likelihood, or the probability of observing the evidence E assuming the hypothesis H is true.
- $P(H)$: The prior probability, which represents the initial belief about H before observing the evidence.
- $P(E)$: The marginal probability of the evidence E, which acts as a normalization constant and is calculated as:

$$P(E) = \sum_H P(E \mid H) \cdot P(H)$$

22. Comparison of Entropy, Joint Entropy, Conditional Entropy, and KL Divergence

Entropy is a fundamental concept in information theory, introduced by Claude Shannon. It measures the uncertainty, randomness, or information in a random variable or system. Entropy quantifies the unpredictability of outcomes.

Uncertainty Measure:

- If a random variable X has high entropy, its outcomes are highly uncertain and unpredictable. For example, a fair coin toss has high uncertainty because both outcomes are equally likely.
- If X has low entropy, its outcomes are more predictable. For example, a biased coin that heavily favors heads has low entropy.

Concept	Definition
Entropy (Regular)	$H(X) = -\sum_{x \in \mathcal{X}} P(X = x) \log P(X = x)$ (discrete) $H(X) = -\int_{-\infty}^{\infty} f_X(x) \log f_X(x)\, dx$ (continuous)
Joint Entropy	$H(X, Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P(X = x, Y = y) \log P(X = x, Y = y)$
Conditional Entropy	$H(X \mid Y) = -\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P(X = x, Y = y) \log P(X = x \mid Y = y)$ $= H(X, Y) - H(Y)$ (chain rule for entropy)
Relative Entropy (KL Divergence)	$D_{\mathrm{KL}}(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log \frac{P(x)}{Q(x)}$ (discrete) $D_{\mathrm{KL}}(P\|Q) = \int_{-\infty}^{\infty} P(x) \log \frac{P(x)}{Q(x)}\, dx$ (continuous)
Mutual Information	$I(X; Y) = H(X) - H(X \mid Y) \quad = H(X) + H(Y) - H(X, Y)$ Measures shared information between X and Y.

Explanation

- **Entropy (H(X))**: Measures the uncertainty or randomness in a random variable X.
- **Joint Entropy (H(X, Y))**: Measures the combined uncertainty of two random variables X and Y.
- **Conditional Entropy (H(X—Y))**: Measures the uncertainty of X given that Y is known.
- **KL Divergence** $D_{KL}(P||Q)$: Measures the difference between two probability distributions P and Q. It is not symmetric, so $D_{\mathrm{KL}}(P||Q) \neq D_{\mathrm{KL}}(Q||P)$.
- **Mutual Information (I(X; Y))**: Quantifies the reduction in uncertainty of X due to knowing Y (or vice versa).

Special Properties

- $H(X) \geq 0$: Entropy is always non-negative.
- $H(X \mid Y) \leq H(X)$: Knowing Y cannot increase the uncertainty of X.
- $D_{\mathrm{KL}}(P\|Q) \geq 0$: KL divergence is always non-negative, with equality if and only if $P = Q$.

23. Mutual Information

Mutual Information measures the amount of information shared between two random variables X and Y. It quantifies how much knowing one variable reduces the uncertainty of the other.

Definition

The mutual information $I(X;Y)$ is defined as:

$$I(X;Y) = H(X) - H(X \mid Y)$$

where:

- $H(X)$: Entropy of X, representing the uncertainty of X.
- $H(X \mid Y)$: Conditional entropy of X given Y, representing the uncertainty of X after observing Y.

It can also be expressed as:

$$I(X;Y) = H(X) + H(Y) - H(X,Y)$$

or equivalently:

$$I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} P(X = x, Y = y) \log \frac{P(X = x, Y = y)}{P(X = x)P(Y = y)}.$$

Key Properties

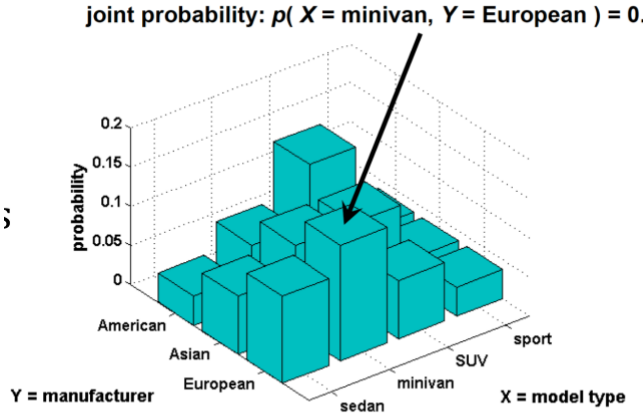
- $I(X;Y) \geq 0$: Mutual information is always non-negative.
- $I(X;Y) = 0$: Mutual information is zero if and only if X and Y are independent.
- Symmetry: $I(X;Y) = I(Y;X)$.

Intuition

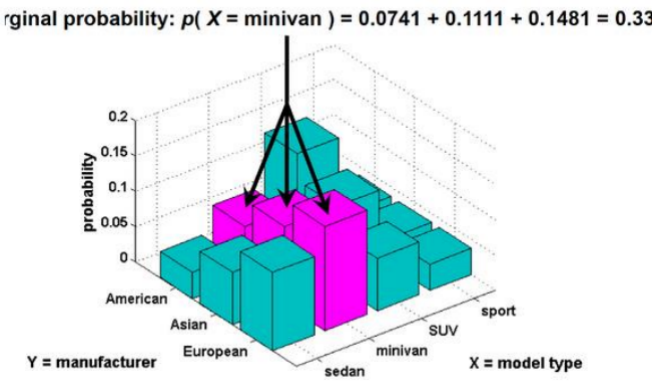
- If $I(X;Y) > 0$, X and Y share information, meaning that knowing one reduces the uncertainty of the other.
- If $I(X;Y) = 0$, X and Y are independent, and knowing one does not provide any information about the other.

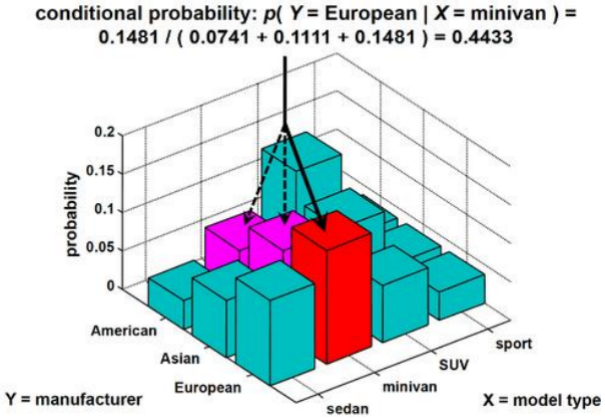
19 Comparison of Probability Distributions

Aspect	Joint Probability Distribution	Marginal Probability Distribution	Conditional Probability Distribution
Definition	For two random variables X and Y, the joint probability distribution specifies the probability that X takes on a specific value x, and Y takes on a specific value y simultaneously.	The probability of one variable, irrespective of the other(s).	The probability of one variable given that another variable has occurred.
Notation	$p(x, y)$ or $f_{X,Y}(x, y)$	$p(x)$ or $f_X(x)$	$p(y \mid x)$ or $f_{Y\|X}(y \mid x)$
Formula	For discrete variables: $P(X = x, Y = y)$ For continuous variables: $p(x, y)$ $= \frac{\partial^2}{\partial x \partial y} F(x, y)$	For discrete: $p(x) = \sum_y p(x, y)$ For continuous: $p(x) = \int p(x, y)\, dy$	$p(y \mid x) = \frac{pX,Y(x,y)}{p(x)}$ provided $f_X(x) > 0$ For discrete: $\frac{p(x,y)}{\sum p(x,y)}$ For continuous: $= \frac{p(x,y)}{\int_\infty^\infty p(x,y)dy}$
In General	Discrete: $P(X_1 = x_1, \& \cdots \& X_n = x_n)$ Continuous: $f_{x_1,\cdots,x_n}(x_1, \cdots, x_n)$ $= \frac{\partial^n}{\partial x_n \cdots \partial x_n} F(x_1, \cdots, x_n)$	Discrete: $p_X(x_1, \cdots, x_m)$ $= \sum_{x_{m+1}} \cdots \sum_{x_n} p_{x_1,\cdots,x_n}$ Continuous: $p_X(x_1, \cdots, x_m)$ $= \int \cdots \int p(x_1, \cdots, x_n), dx_{m+1} \cdots dx_n$	For discrete: $p(x_{m+1}, \cdots, x_n \| x_1, \cdots x_m)$ $\frac{p(x_1,\cdots,x_n)}{\sum \cdots \sum p(x_1,\cdots,x_n)}$ For continuous: $p(x_{m+1}, \cdots, x_n \| x_1, \cdots x_m)$ $= \frac{p(x_1,\cdots,x_n}{\int_\infty^\infty \cdots \int_\infty^\infty p(x_1,\cdots,x_n) dx_{m+1} \cdots dx_n}$



Credit: Jeff Howbert





24. Multivariate Normal Distribution

The **multivariate normal distribution** describes the joint distribution of a vector of random variables that follow a Gaussian distribution.

Probability Density Function (PDF)

The probability density function of a d-dimensional random vector $\mathbf{X} = (X_1, X_2, \ldots, X_d)^T$ is given by:

$$f(\mathbf{X}) = \frac{1}{(2\pi)^{d/2} \det(\mathbf{\Sigma})^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{X} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{X} - \boldsymbol{\mu})\right)$$

$$\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$$

Where:

- $\mathbf{X}$: A d-dimensional random vector.
- $\boldsymbol{\mu}$: The mean vector $(d \times 1)$, $\boldsymbol{\mu} = (\mu_1, \mu_2, \ldots, \mu_d)^T$.
- $\mathbf{\Sigma}$: The covariance matrix $(d \times d)$, which is symmetric and positive definite.
- $\det(\mathbf{\Sigma})$: The determinant of the covariance matrix $\mathbf{\Sigma}$.
- $(\mathbf{X} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{X} - \boldsymbol{\mu})$: The squared Mahalanobis distance between $\mathbf{X}$ and $\boldsymbol{\mu}$.

Key Properties

If

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_A \\ \mathbf{x}_B \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \boldsymbol{\mu}_A \\ \boldsymbol{\mu}_B \end{pmatrix}, \begin{pmatrix} \mathbf{\Sigma}_{AA} & \mathbf{\Sigma}_{AB} \\ \mathbf{\Sigma}_{BA} & \mathbf{\Sigma}_{BB} \end{pmatrix}\right),$$

- **Marginal Distributions**: The marginal distribution of any subset of variables from a multivariate normal distribution is also multivariate normal.

$$\mathbf{x}_A \sim \mathcal{N}(\boldsymbol{\mu}_A, \mathbf{\Sigma}_{AA}),$$

$$\mathbf{x}_B \sim \mathcal{N}(\boldsymbol{\mu}_B, \mathbf{\Sigma}_{BB}).$$

- **Conditional Distributions**: The conditional distribution of a subset of variables, given others, is also multivariate normal.

$$\mathbf{x}_A \mid \mathbf{x}_B \sim \mathcal{N}\left(\boldsymbol{\mu}_A + \mathbf{\Sigma}_{AB}\mathbf{\Sigma}_{BB}^{-1}(\mathbf{x}_B - \boldsymbol{\mu}_B), \mathbf{\Sigma}_{AA} - \mathbf{\Sigma}_{AB}\mathbf{\Sigma}_{BB}^{-1}\mathbf{\Sigma}_{BA}\right),$$
$$\mathbf{x}_B \mid \mathbf{x}_A \sim \mathcal{N}\left(\boldsymbol{\mu}_B + \mathbf{\Sigma}_{BA}\mathbf{\Sigma}_{AA}^{-1}(\mathbf{x}_A - \boldsymbol{\mu}_A), \mathbf{\Sigma}_{BB} - \mathbf{\Sigma}_{BA}\mathbf{\Sigma}_{AA}^{-1}\mathbf{\Sigma}_{AB}\right).$$

- **Independence**: If $\Sigma_{ij} = 0$ (the covariance between X_i and X_j), then X_i and X_j are uncorrelated. For multivariate normals, this also implies independence.

25. Transformations

- If $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$, then $\mathbf{x} + \mathbf{c} \sim \mathcal{N}(\boldsymbol{\mu} + \mathbf{c}, \mathbf{\Sigma})$.
- If $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$, then $A\mathbf{x} \sim \mathcal{N}(A\boldsymbol{\mu}, A\mathbf{\Sigma}A^T)$.
- **Special Case**: If $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$, then $c\mathbf{x} \sim \mathcal{N}(c\boldsymbol{\mu}, c^2\mathbf{\Sigma})$.
- If $\mathbf{x} \perp \mathbf{y}$ ($\mathbf{x}$ and $\mathbf{y}$ are independent):

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_X, \mathbf{\Sigma}_X) \quad \text{and} \quad \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_Y, \mathbf{\Sigma}_Y),$$

  then $\mathbf{x} + \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_X + \boldsymbol{\mu}_Y, \mathbf{\Sigma}_X + \mathbf{\Sigma}_Y)$.

- **Standard Multivariate Normal**:

$$\mathbf{z} \sim \mathcal{N}(\mathbf{0}, I)$$

- $\mathbf{z} + \boldsymbol{\mu} \sim \mathcal{N}(\boldsymbol{\mu}, I)$ and $\sigma\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \sigma^2 I) \implies \boldsymbol{\mu} + \sigma\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 I)$.
- **Compare: Standard (Univariate) Normal**:

$$Z \sim \mathcal{N}(0, 1)$$

$$Z + \mu \sim \mathcal{N}(\mu, 1) \quad \text{and} \quad \sigma Z \sim \mathcal{N}(0, \sigma^2) \implies \mu + \sigma Z \sim \mathcal{N}(\mu, \sigma^2).$$

- **Isotropic Gaussian:** Variance along every direction is the same.
- **Combination of Two Variables**:
  - If X and Y are jointly Gaussian, any linear transformation of their means or covariances will still result in a Gaussian distribution.
  - However, if X and Y are **not jointly Gaussian** and are **not independent**, the combination of X and Y may **not** result in a Gaussian distribution. However, when Covariance is -1 or 1, need to care about that.

27. Comparison of MAP and Full Bayesian Inference

- **Full Bayesian Inference:** Computes the full posterior distribution:

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta)p(\theta)}{p(\mathcal{D})},$$

- **MAP Estimation:** Finds the parameter θ that maximizes the posterior:

$$\hat{\theta}_{\text{MAP}} = \arg\max_{\theta} \; p(\theta \mid \mathcal{D}) = \arg\max_{\theta} \; p(\mathcal{D} \mid \theta)p(\theta).$$

  Here, $p(\mathcal{D})$ is omitted because it is constant with respect to θ.

Summary:

- **Full Bayesian Inference:** Retains the entire posterior distribution, allowing for uncertainty quantification and exploration of all plausible parameter values.
- **MAP Estimation:** Provides a single point estimate corresponding to the mode of the posterior, losing information about uncertainty and multimodality. If the posterior is multimodal, MAP will only find one of the modes (usually the highest one), potentially ignoring other equally or nearly plausible solutions and lose some important information.

27. Bias-Variance Tradeoff

1. Bias-Variance Tradeoff

- Expected Error can be decomposed into three components:

$$\text{Expected Error} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}.$$

- Goal: Strike the right balance between bias and variance to minimize the expected error and avoid both underfitting and overfitting.

Aspect	**Overfitting**	**Underfitting**
Bias	*Low*	*High*
Variance	*High*	*Low*
Model Complexity	Too high	Too low
Solution	Reduce model complexity Add regularization Early stopping	Increase complexity Add features Train longer

21. Comparison of Regular and Conditional Expected Value, Moments, and Covariance

Concept	Regular Case	Conditional Case
Expected Value	$\mathbb{E}[X] = \int_{-\infty}^{\infty} x \cdot f_X(x)\,dx$	$\mathbb{E}[X \mid Y = y] = \int_{-\infty}^{\infty} x \cdot f_{X\mid Y}(x \mid y)\,dx$
Linearity of Expectation	$\mathbb{E}[X+Y] = \mathbb{E}[X] + \mathbb{E}[Y], \quad \mathbb{E}[cX] = c\mathbb{E}[X]$	$\mathbb{E}[X \mid Y]$ preserves linearity : $\mathbb{E}[aX + bY \mid Z] = a\mathbb{E}[X \mid Z] + b\mathbb{E}[Y \mid Z]$
Variance	$\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$	$\mathrm{Var}(X \mid Z = z) = \mathbb{E}[(X - \mathbb{E}[X \mid Z = z])^2 \mid Z = z]$
Covariance	$\mathrm{Cov}(X, Y) = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$	$\mathrm{Cov}(X, Y \mid Z = z) = \mathbb{E}[XY \mid Z = z] - \mathbb{E}[X \mid Z = z]\mathbb{E}[Y \mid Z = z]$
Correlation	$\rho_{X,Y} = \frac{\mathrm{Cov}(X,Y)}{\sqrt{\mathrm{Var}(X)\mathrm{Var}(Y)}}$	$\rho_{X,Y\mid Z=z} = \frac{\mathrm{Cov}(X,Y\mid Z=z)}{\sqrt{\mathrm{Var}(X\mid Z=z)\mathrm{Var}(Y\mid Z=z)}}$
Law of Total Expectation	$\mathbb{E}[X] = \mathbb{E}[\mathbb{E}[X \mid Z]]$	$\mathbb{E}[X \mid Z_1, \ldots, Z_k] =$ nested expectations over Z_i

26. Comparison of MLE and MAP

Aspect	MLE	MAP
Key Formula	$\arg\max_\theta P(D \mid \theta)$	$\arg\max_\theta P(\theta \mid D) = P(\theta, \sigma \mid D) \propto P(D \mid \theta, \sigma)P(\theta, \sigma)$
Loss Function	$L(\vec{\theta}, \sigma; D)$	$L(\theta, \sigma \mid D) + P(\theta, \sigma)$
Prior Involvement	None	Incorporates prior belief ($P(\theta)$), not sure can set $\mu = 0, \sigma = 1$
Objective	Maximize likelihood	Maximize posterior probability
Log Loss Function	$\vec{\theta}_{WLE} = -\arg\max_\theta \log L(\vec{w}, b, \sigma; D) = \arg\min_\theta \sum_i^N (y_i - \vec{\theta}^T \vec{x_i})^2$	$\vec{\theta}_{WLE} = \arg\min_\theta \sum_i^N (y_i - \vec{\theta}^T \vec{x_i})^2 + \lambda \lVert\vec{\theta}\rVert_2^2$
Solution	$\vec{\theta}_{MLE} = (X^T X)^{-1} X^T \vec{y}$	$\vec{\theta}_{MAP} = (X^T X + \lambda I)^{-1} X^T \vec{y}$
Posterior	None	Posterior Covariance: $\Sigma_{\text{post}} = \left(\frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}\right)^{-1}$ Posterior Mean (MAP Estimate): $\mu_{\text{post}} = \Sigma_{\text{post}} \cdot \frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{y}$
Overfitting	More prone	Less prone (regularization effect)
Application	Suitable for data-rich environments	Robust in data-scarce or noisy environments

28. Optimization

Algorithm	Formula	Benefits	Drawbacks	Use Cases	Importants
Gradient	$\theta^t = \vec{\theta}^{t-1} - \gamma_t \frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)})$	Simple and effective for small data.	Slow convergence, sensitive to γ_t.	Small datasets, convex functions. Not good in non-convex Not good in Non-Lipschitz	gradient changes position, can can non-convex but may not global min
Momentum	$\Delta\theta = \alpha\Delta\theta - \gamma_t \frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)})$ $\vec{\theta^t} = \vec{\theta}^{(t-1)} + \Delta\vec{\theta}$	Reduces oscillations faster in some directions.	Requires tuning of α.	Functions with steep valleys Not good in non-convex Not good in Non-Lipschitz	gradient changes velocity, adv in find local min & saddle point
Nesterov(NAG)	$\Delta\theta = \alpha\Delta\theta - \gamma_t \frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)} + \alpha\Delta\vec{\theta})$ $\vec{\theta^t} = \vec{\theta}^{(t-1)} + \Delta\vec{\theta}$	Anticipates future gradients, faster convergence.	Look-ahead adds complexity.	Functions with high curvature or saddle points. Not good in non-convex Not good in Non-Lipschitz	account for the inertia when choose where to compute gradient, improved
AdaGrad	$D = \text{diag}((\sqrt{\sum_{t=0}^{t-1}(\frac{\partial L}{\partial \vec{\theta_i}}(\vec{\theta^{(t)}}))^2})_{i=1}^n)$ $\vec{\theta^t} = \vec{\theta}^{(t-1)} - \gamma_t D^{-1} \frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{t-1})$	Handles sparse data effectively.	Learning rate vanishes.	Sparse data (e.g., NLP, recommender systems). some good in non-convex some good in Non-Lipschitz	use diag preconditioner
Adam	$\vec{m} = \alpha\vec{m} + (1-\alpha)\frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)})$ $D = \beta D +$ $(1-\beta)\text{diag}(((\frac{\partial L}{\partial \vec{\theta_i}}(\vec{\theta^{(t)}}))^2)_{i=1}^n)$ $\vec{\theta^{(t)}} = \theta^{(t-1)} -$ $\gamma_t((\frac{1}{1-\beta^t}D)^{\frac{1}{2}+\epsilon I})^{-1}(\frac{1}{1-\alpha^t}\vec{m})$	Combines Momentum and AdaGrad.	Generalization issues, tuning needed.	Deep learning, non-convex optimization. good in non-convex good in Non-Lipschitz	use diag preconditioner one of best for non-convex and Non-Lipschitz
Newton	$H = \frac{\partial^2 L}{\partial\vec{\theta}\partial\vec{\theta^T}}(\vec{\theta}^{(t-1)})$ $\vec{\theta^t} = \vec{\theta}^{(t-1)} - \gamma_t H^{-1} \frac{\partial L}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)}$	Fast convergence (quadratic near optimum).	Expensive computation of Hessian, high memory.	Small, well-behaved convex problems. some good in non-convex some good in Non-Lipschitz	use non diag preconditioner
SGD	$\theta^t = \vec{\theta}^{(t-1)} - \gamma_t \frac{\partial L_{i'}}{\partial \vec{\theta}}(\vec{\theta}^{(t-1)})$	Fast updates, low memory.	Noisy convergence.	Large datasets, stream data some+ good in non-convex some+ good in Non-Lipschitz	

Lipschitz continuity: $|L(x_1) - L(x_2)| \leq c\|x_1 - x_2\|_2, \|\nabla L(x_1) - \nabla L(x_2)\| \leq c\|x_1 - x_2\|_2, \|\frac{\partial}{\partial \vec{x}} L(\vec{x})\|_2 \leq c$, c is second derivative of L

- **Momentum parameter (in momentum and Adam):** α
  - Typically 0.9 works well. Consider increasing to 0.99 or 0.999 if mini-batch size (number of data points in a mini-batch) is small.
- **Momentum parameter for the preconditioner (in Adam):** β
  - Typically 0.999 works well.

Term	**Description**
(Full Batch) Gradient Descent	Computing the gradient on the entire dataset every iteration.
Mini-Batch Gradient Descent (SGD)	Computing the gradient on a subset of data points (a mini-batch) every iteration. Sometimes referred to as "stochastic gradient descent."
Stochastic Gradient Descent	Computing the gradient on a single data point every iteration.
Epoch Definition	The smallest number of iterations needed to perform one full pass over the dataset: • 1 iteration for full batch gradient descent. • N/M iterations for mini-batch gradient descent, where M is the mini-batch size. • N iterations for stochastic gradient descent.
Best Practice	Mini-batch gradient descent tends to work best in practice.

29. Multi-layer perception (MLP)

One of the simplest models in neural networks

$$\hat{y} = W_L \vec{h}_L, \text{ where } \vec{h}_L = \psi(W_{L-1}\vec{h}_{L-1}) \text{ and } \vec{h}_{L-1} = \psi(W_{L-2}\vec{h}_{L-2}), \ldots, \text{ and } \vec{h}_1 = \psi(W_0\vec{x}).$$

Definitions:

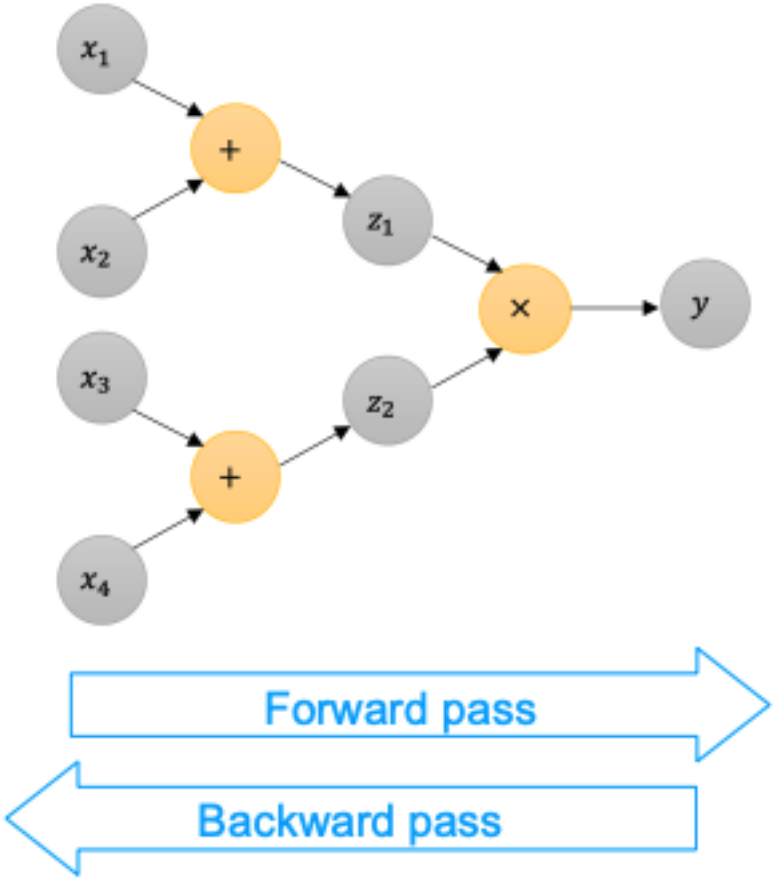
- $\hat{y}$: Output layer / last layer
- $\vec{h}_l$: l-th hidden layer / $(l+1)$-th layer / features / (post-)activations
- $\vec{x}$: Input layer / first layer
- $\vec{z}_l := W_l\vec{h}_l$: Pre-activations

Some activation Functions - h_l:

- Sigmoid: $g(z) = \frac{1}{1+exp(-z)} = \sigma(z)$
- Hyperbolic tangent(Tanh): $g(z) = tanh(z)$
- Softplus: $g(z) = log(1 + exp(z))$
- ReLU: $g(z) = \max(0, z)$

30. Backpropagation

Forward pass: Computing the values of all variables, starting from the children and ending at the ancestor.
Backward pass: Computing the partial derivatives of the ancestor by using **Chain Rule** w.r.t. each variable, starting from the ancestor and ending at the children.



Problems in Backptopagation

Aspect	Vanishing Gradients	Exploding Gradients
Cause	Gradients shrink due to small derivatives.	Gradients grow uncontrollably due to large derivatives.
Symptoms	Slow or no learning in earlier layers.	Instability in weight updates, divergence in training.
Common in	Very deep networks, RNNs (long sequences).	Very deep networks, RNNs (long sequences).
Solutions	ReLU, Weight Initialization, ResNets, Batch Norm, Shallow Networks, Layer Normalization	Gradient Clipping, Weight Initialization, Batch Norm, ResNets, Add Regularization (Weight Decay), adjustment learning rate, Layer Normalization, network shallower
Function Caused	Sigmoid, Tanh	ReLU, Softplus

Weight Initialization:
Xavier initialization (named after Xavier Glorot): Initialize weight matrix W_l with:

$$W_l \sim \mathcal{N}\left(\vec{0}, \frac{2}{n_l + n_{l+1}} I\right)$$

He initialization (named after Kaiming He): Initialize weight matrix W_l with:

$$W_l \sim \mathcal{N}\left(\vec{0}, \frac{4}{n_l + n_{l+1}} I\right)$$

Weight Decay: add squared Frobenius norms of the weight matrices excluding the biases to the loss function to avoid exploding gradients and overfitting.
Layer Normalization: scale the pre-activations of each hidden layer.

Formula

$$\frac{\partial L}{\partial \mathbf{w}_{0,j}^T} = \begin{pmatrix}\mathbf{0} & \cdots & \mathbf{0} & \mathbf{x} & \mathbf{0} & \cdots & \mathbf{0}\end{pmatrix} \prod_{l=1}^{L} \left[\begin{pmatrix} g'(z_{l,1}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g'(z_{l,n_l-1}) \end{pmatrix} W_l^T \right] \frac{\partial L}{\partial \hat{\mathbf{y}}}$$

Vanishing Gradients:

$$\|\frac{\partial L}{\partial \mathbf{w}_{0,j}^T}\|_2 \leq \sigma_{1,1}(\begin{pmatrix}\mathbf{0} & \cdots & \mathbf{0} & \mathbf{x} & \mathbf{0} & \cdots & \mathbf{0}\end{pmatrix}) \prod_{l=1}^{L} \left[\sigma_{1,1}(\begin{pmatrix} g'(z_{l,1}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g'(z_{l,n_l-1}) \end{pmatrix}) \sigma_{1,1}(W_l^T) \right] \|\frac{\partial L}{\partial \hat{\mathbf{y}}}\|_2$$

Exploding Gradients:

$$\|\frac{\partial L}{\partial \mathbf{w}_{0,j}^T}\|_2 \geq \sigma_{n,n}(\begin{pmatrix}\mathbf{0} & \cdots & \mathbf{0} & \mathbf{x} & \mathbf{0} & \cdots & \mathbf{0}\end{pmatrix}) \prod_{l=1}^{L} \left[\sigma_{n,n}(\begin{pmatrix} g'(z_{l,1}) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & g'(z_{l,n_l-1}) \end{pmatrix}) \sigma_{n,n}(W_l^T) \right] \|\frac{\partial L}{\partial \hat{\mathbf{y}}}\|_2$$

0.1 Residual Connection (ResNet)

In general, a **residual connection** can take the pre- or post-activations of an earlier layer and add them to the pre- or post-activations of a later layer.

Recall the Functional Form of an MLP:

$$\hat{y} = W_L \vec{h}_L, \quad \text{where } \vec{h}_L = \psi(\vec{z}_L),\ \vec{z}_L = W_{L-1}\vec{h}_{L-1}, \ldots, \vec{h}_1 = \psi(\vec{z}_1) \text{ and } \vec{z}_1 = W_0 \vec{\tilde{x}}.$$

Residual Connection Modification

In the previous example, we made:

$$\vec{z}_l = \begin{pmatrix} \vec{h}_{l-1} \\ 1 \end{pmatrix} + W_{l-1}\vec{h}_{l-1}.$$

In general, we can make either $\vec{z}_l$ or $\vec{h}_l$ be:

$$\vec{z}_l = W_l\vec{h}'_l + \begin{pmatrix} \vec{h}_{l'} \\ 1 \end{pmatrix} \quad \text{or} \quad \vec{h}_l = W'_l\vec{h}'_l + \begin{pmatrix} \vec{z}_{l'} \\ 1 \end{pmatrix}, \quad \text{where } l' < l.$$

Skip Connection Explanation

A **residual connection** is a special case of a *skip connection*, which in general skips certain operations (e.g., matrix multiplication) and directly *concatenates* (rather than adds) the pre- or post-activations of an earlier layer to the pre- or post-activations of a later layer.

Resulting Model

The resulting model is no longer an MLP and is sometimes known as a *residual network* (though sometimes the term refers to a particular neural network that is in popular use).

30. Support Vector Machines(SVM) - Classifier

Origin Formula:

$$\max_{m,\mathbf{w},b} \quad m$$

subject to

$\mathbf{w}^\top \mathbf{x}_i > b$ for all i such that $y_i = 1$, (For positive examples, should lie on one side of the hyperplane)

$\mathbf{w}^\top \mathbf{x}_i < b$ for all i such that $y_i = -1$, (For negative examples, should lie on the other side of the hyperplane)

$$m = \min_i \left\{ \frac{1}{\|\mathbf{w}\|_2} \left| \mathbf{w}^\top \mathbf{x}_i - b \right| \right\}, \quad \text{(The definition of margin)}$$

Primal SVM - Hard Margin SVM(Linear Seperable) - P

$$\min_{\mathbf{w},b} \quad \frac{1}{2}\|\mathbf{w}\|_2^2$$

subject to:

$$y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1$$

Generalized Lagrangian:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda}) = \frac{1}{2}\|\mathbf{w}\|_2^2 - \sum_{i=1}^{n} \lambda_i \left[y_i(\mathbf{w}^\top \mathbf{x}_i - b) - 1 \right],$$

Dual Problem SVM - D

:

$$\max_{\boldsymbol{\lambda}} \quad \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j$$

Subject to:

$$\sum_{i=1}^{N} \lambda_i y_i = 0, \quad \lambda_i \geq 0, \quad \forall i$$

Relationship of P and D

When $P \geq D$, weak duality holds.
When $P = D$, the following conditions hold:

1. The primal problem is **feasible** (there exists at least one solution satisfying the constraints).
2. The dual problem is **feasible** (there exists at least one solution satisfying the dual constraints).
3. The problem satisfies conditions for **strong duality** (e.g., convexity and Slater's condition for convex optimization problems).
4. Satisfied KKT condition

Slater's Condition

1. Convex objective, f(x) is convex
2. Convex inequality constraints g(x) is convex, Hessian ≥ 0
3. Linear equality constraints h(x) = constant or linear
4. Strictly feasible solution. at least one point satisfies g(x)<0, h(x) = 0

When to use Dual or Primal when they equal:
The primal optimizes over $n + 1$ variables, whereas the dual optimizes over N variables, where n and N denote the dimensionality and the number of data points, respectively.
When $n \ll N$ (dimensionality much less than the number of data points), it is more efficient to solve the primal. Otherwise, it is more efficient to solve the dual.
Later, we will see another reason to solve the dual.

Karush-Kuhn-Tucker (KKT) Conditions

Given the optimization problem:

$$\min_{\mathbf{x}} f(\mathbf{x})$$

Subject to:

$$g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, m \quad \text{(inequality constraints)},$$
$$h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, p \quad \text{(equality constraints)},$$

The KKT conditions are:

1. **Stationarity:**
$$\nabla f(\mathbf{x}^*) + \sum_{i=1}^{m} \lambda_i \nabla g_i(\mathbf{x}^*) + \sum_{j=1}^{p} \mu_j \nabla h_j(\mathbf{x}^*) = 0.$$
2. **Primal Feasibility:**
$$g_i(\mathbf{x}^*) \leq 0, \quad i = 1, \ldots, m,$$
$$h_j(\mathbf{x}^*) = 0, \quad j = 1, \ldots, p.$$
3. **Dual Feasibility:**
$$\lambda_i \geq 0, \quad i = 1, \ldots, m.$$
4. **Complementary Slackness:**
$$\lambda_i g_i(\mathbf{x}^*) = 0, \quad \forall i.$$

Here:

- $\mathbf{x}^*$ is the optimal solution.
- λ_i are the Lagrange multipliers for the inequality constraints.
- μ_j are the Lagrange multipliers for the equality constraints.

Observe that only non-zero dual variables λ_i^* affect $\mathbf{w}^*$, which in turn affects b^*.
Hence, the optimal hyperplane can only be affected by data points on the margin.

- These data points are the **support vectors**.
- By complementary slackness, they must lie on the margin.

Data points outside the margin do not affect the optimal hyperplane.

Kernal Tricks

The **kernel trick** allows algorithms to operate in a high-dimensional feature space without explicitly computing the transformation. It relies on computing the inner product of the transformed features via a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, such that:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j),$$

where $\phi(\mathbf{x})$ is the mapping function to the higher-dimensional space.

Common Kernel Functions

1. **Linear Kernel**:
$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^\top \mathbf{x}_j.$$
This is equivalent to no mapping ($\phi(\mathbf{x}) = \mathbf{x}$).

2. **Polynomial Kernel**:
$$K(\mathbf{x}_i, \mathbf{x}_j) = \left(\mathbf{x}_i^\top \mathbf{x}_j + c\right)^d,$$
where c is a constant and d is the degree of the polynomial.

3. **Radial Basis Function (RBF) or Gaussian Kernel**:
$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\right),$$
where σ controls the spread of the kernel.

4. **Sigmoid Kernel**:
$$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh\left(\alpha \mathbf{x}_i^\top \mathbf{x}_j + c\right),$$
where α and c are parameters.

Advantages of the Kernel Trick

- Avoids explicit computation of the high-dimensional mapping $\phi(\mathbf{x})$, saving computation and memory.
- Enables algorithms like SVMs to solve nonlinear problems by finding a linear hyperplane in the transformed space.
- Flexibility: Different kernels can adapt to various data types and problems.

Key Concepts in Mercer's Theorem

1. **Symmetry**:
$$K(\mathbf{x}, \mathbf{y}) = K(\mathbf{y}, \mathbf{x}),$$
because the kernel must represent an inner product, which is inherently symmetric.

2. **Positive Semi-Definiteness**: For any finite set of points $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$, the kernel matrix $\mathbf{K}$ defined as:
$$\mathbf{K}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j),$$
must be positive semi-definite, meaning:
$$\mathbf{z}^\top \mathbf{K} \mathbf{z} \geq 0,$$
for any vector $\mathbf{z}$.

3. **Feature Space Representation**: Mercer's theorem guarantees that the kernel function $K(\mathbf{x}, \mathbf{y})$ corresponds to an inner product in some (possibly infinite-dimensional) feature space via a mapping ϕ, such that:
$$K(\mathbf{x}, \mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle.$$

If either symmetry or positive semi-definiteness fails, $K(\mathbf{x}, \mathbf{y})$ cannot be considered a valid kernel under Mercer's Theorem.

31. Soft-Margin SVM

The Soft-Margin SVM introduces slack variables $\xi_i \geq 0$ to allow margin violations, making it suitable for non-linearly separable data. The optimization problem is:

$$\min_{\mathbf{w}, b, \boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|_2^2 + C \sum_{i=1}^{n} \xi_i,$$

subject to:

$$y_i(\mathbf{w}^\top \mathbf{x}_i - b) \geq 1 - \xi_i, \quad i = 1, \ldots, n,$$
$$\xi_i \geq 0, \quad i = 1, \ldots, n.$$

Also, same as: (put everything into the formula like $\xi_i = \max(0, 1 - y_i(w^T x_i - b)$, $\frac{1}{2\gamma} = \lambda$)

$$\min_{\mathbf{w}, b} \sum_{i=1}^{N} \max(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i - b)) + \lambda \|\mathbf{w}\|_2^2$$

Slack Variables

- $\xi_i = 0$: The point satisfies the margin constraint.
- $0 < \xi_i \leq 1$: The point lies within the margin but is correctly classified.
- $\xi_i > 1$: The point is misclassified.

Dual Formulation

The dual form of the optimization problem is:

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j),$$

subject to:

$$0 \leq \alpha_i \leq C, \quad \forall i,$$
$$\sum_{i=1}^{n} \alpha_i y_i = 0.$$

Decision Boundary

The decision boundary is:

$$f(\mathbf{x}) = \operatorname{sgn}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b \right).$$

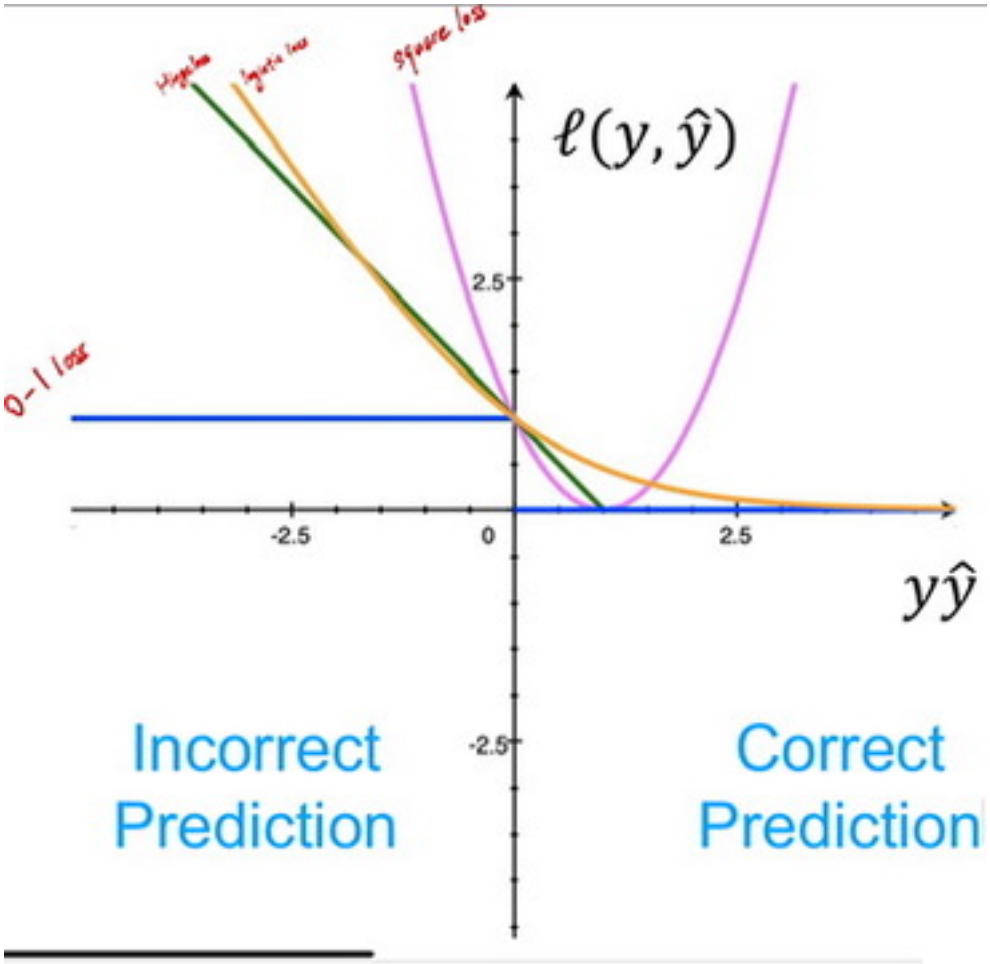
Regularization Parameter C

- **Small** C: Allows more margin violations, resulting in a smoother decision boundary.
- **Large** C: Penalizes margin violations heavily, leading to stricter classification but potentially overfitting.

32.Comparison: Ridge Regression vs. Soft-Margin SVM

Aspect	Logistic Regression	Ridge Regression	Soft-Margin SVM
Objective Function	$\min_{\vec{w}} \sum_{i=1}^{N} l(y_i, f(\vec{x_i} : \vec{w})) + \lambda \|\vec{w}\|_2^2$	$\min_{\vec{w}} \sum_{i=1}^{N} l(y_i, f(\vec{x_i} : \vec{w})) + \lambda \|\vec{w}\|_2^2$	$\min_{\vec{w}} \sum_{i=1}^{N} l(y_i, f(\vec{x_i} : \vec{w})) + \lambda \|\vec{w}\|_2^2$
Loss Function	Logistics loss: $l(y, \hat{y}) = \frac{\log(1+e^{-y\hat{y}})}{\log_2}$	Square loss: $l(y, \hat{y}) = (y_i - \hat{y}_i)^2$	Hinge loss:$l(y, \hat{y}) = \max(0, 1 - y_i \hat{y}_i)$
Regularization L2	$\|\mathbf{w}\|_2^2$	$\|\mathbf{w}\|_2^2$	$\|\mathbf{w}\|_2^2$
Prediction Function	$f(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x} + b), \quad \sigma(z) = \frac{1}{1+e^{-z}}$	$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$	$f(\mathbf{x}) = \mathbf{w}^\top \mathbf{x} + b$
Output Type	Probabilistic (values between 0 and 1)	Continuous output (Regression)	Binary output (Classification)
Target Variable	$Binary : y \in \{0, 1\}$	$Real - valued : y \in \mathbb{R}$	$Binary : y \in \{-1, 1\}$
Goal	Model probabilities for classification.	Minimize squared error and penalize large weights.	Maximize the margin while penalizing margin violations.
Regularization Par	λ : Controls trade-off between fit and regularization.	λ : Controls trade-off between fit and regularization.	λ : Controls trade-off between margin violations and regularization.
Handling Outliers	Moderately sensitive to outliers.	Highly sensitive to outliers due to squared loss.	Robust to outliers due to hinge loss.
Applications	Classification tasks: spam detection, medical diagnosis.	Regression tasks: predict house prices, stock values.	Binary classification: spam detection, image recognition.
Optimization Type	Gradient Descent.	Closed-form solution (Normal equation) or Gradient Descent.	Quadratic Programming (QP) optimization.
Kernelization	Not applicable.	Not applicable.	Supports kernel functions for non-linear classification.

Cross-entropy loss: $-\sum_{i=1}^{N}(\big[y_i = 1\big] \log Pr(\vec{y_i} = 1) + \big[y_i = -1\big] \log Pr(\hat{y}_i = -1)) + \lambda \|\vec{w}\|_2^2$ to get logistic loss



Hinge Loss vs. Logistic Loss

Hinge loss:

- Can be formulated as a **constrained problem**.
- Can derive the dual which only involves inner products between data points. As a result, can apply the kernel trick.
- When formulated as an unconstrained problem, gradient-based methods do not work as well because the derivative is discontinuous.

Logistic loss:

- Can only be formulated as an **unconstrained problem**, so must use iterative gradient-based methods to optimize it.
- Cannot apply the kernel trick, but works well with models that are trained with gradient-based methods anyway, i.e., neural networks.

33. Softmax Classification

Softmax Function

The softmax function maps scores (logits) into probabilities:

$$P(y = k|\mathbf{x}) = \frac{\exp(z_k)}{\sum_{j=1}^{K} \exp(z_j)},$$

where:

$$z_k = \mathbf{w}_k^\top \mathbf{x} + b_k,$$

and $k = 1, \ldots, K$ for K classes.

Prediction

The predicted class is:

$$\hat{y} = \arg\max_k P(y = k|\mathbf{x}).$$

Loss Function

The loss function for Softmax Classification is the cross-entropy loss:

$$\mathcal{L}(\mathbf{w}, \mathbf{b}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{i,k} \log(P(y = k|\mathbf{x}_i)) + \lambda \sum_{k=1}^{K} \|\mathbf{w}_k\|_2^2.$$

Logistic Loss is one specific example in softmax classification when K = 2.

Scenario	Better Choice	Reason
Probabilistic Output Needed	Logistic Loss	Produces probabilities for confidence estimation.
Noisy Data	Logistic Loss	Smooth boundary handles overlapping classes better.
Outlier Robustness	Soft-Margin SVM	Ignores outliers due to hinge loss properties.
Small Dataset	Soft-Margin SVM	Maximizes margin for better generalization.
Kernelization Needed	Soft-Margin SVM	Supports kernels for non-linear separability.
Multi-Class Classification	Logistic Loss (Softmax)	Extends naturally to multi-class via softmax.
Large Dataset with Noise	Logistic Loss	Easier to optimize with gradient descent.