HCMUT EE MACHINE LEARNING & IOT LAB

# DAY 9

# SUPPORT VECTOR MACHINES

Presentation By: Nguyễn Duy Tân

Ho Chi Minh City – 2025

# Table of Content

# I. Overview

# I. Overview

## Introduction

The distance from a point (vector) $\mathbf{x}_0$ to the hyperplane defined by:

$$\mathbf{w}^{\mathbf{T}}\mathbf{x} + \mathrm{b} = \mathbf{0}$$

is given by

$$\frac{|\mathbf{w}^{\mathbf{T}}\mathbf{x_0} + \mathrm{b}|}{||\mathbf{w}||_{\mathbf{2}}}$$

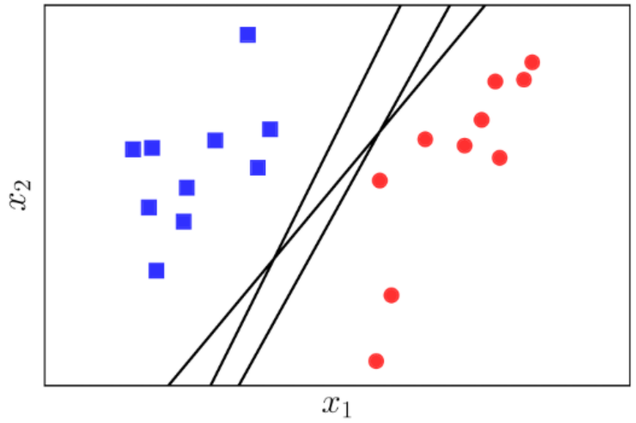
where

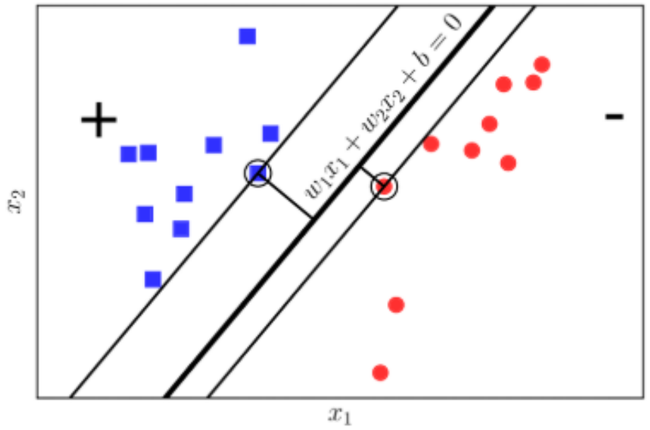
$$||\mathbf{w}||_{\mathbf{2}} = \sqrt{\sum_{i=1}^{d} w_{\boldsymbol{i}}^{\mathbf{2}}}$$

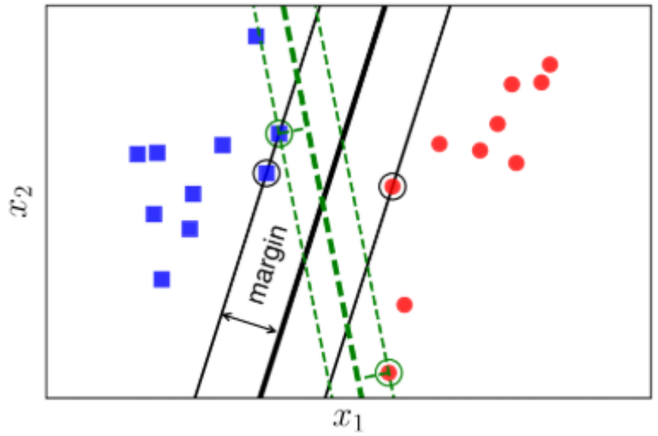
and $d$ is the dimensionality of the space.

# I. Overview

## Perceptron Learning Algorithm




https://machinelearningcoban.com/2017/04/09/smv/

# I. Overview

## Formulate optimization problem for SVM

Assume that the training set consist of **N** labeled pairs

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$$

where each input vector $\mathbf{x}_i \in \mathbb{R}^d$ and its label $y_i$ is either +1 ***(class 1)*** or -1 ***(class 2)***, as in the Perceptron Learning Algorithm (PLA). Here *d* is the data dimensionality and **N** is the number of examples.

Then, we have distance from $\mathbf{x}_\mathrm{n}$ to the hyperplane is

$$\frac{\mathbf{y}_n(\mathbf{w}^\mathrm{T}\mathbf{x}_\mathrm{n} + b)}{\left|\left|\mathbf{w}\right|\right|_\mathbf{2}}$$

# I. Overview

## Formulate optimization problem for SVM

We define the ***margin*** of the classifier as the smallest such distance over all training points

$$margin = \min_{n} \frac{y_n(\mathbf{w}^T\mathbf{x}_n + b)}{\left|\left|\mathbf{w}\right|\right|_2}$$

Thus, the SVM optimization problem is to choose $(\mathbf{w}, b)$ to maximize this margin:

$$(\mathbf{w}, b) = arg \max_{\boldsymbol{w},b} \left\{ \frac{1}{\left|\left|\boldsymbol{w}\right|\right|_2} \min_{n} y_n(\mathbf{w}^T\mathbf{x}_n + b) \right\}$$

So difficult for solving !!!!!!!!!!

# I. Overview

## Formulate optimization problem for SVM

Naturally, we notice that

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, n = 1, \dots, N$$

Then, the margin becomes

$$(\mathbf{w}, b) = arg \max_{\mathbf{w},b} \frac{1}{||\mathbf{w}||_2}$$

By a simple transformation, we can reduce it to the following problem

$$(\mathbf{w}, b) = \arg \min_{\mathbf{w},b} \frac{1}{2} ||\mathbf{w}||_2^2$$

Subject to

$$1 - y_n(\mathbf{w}^T\mathbf{x}_n + b) \leq 0, n = 1, \dots, N$$

# I. Overview

## Formulate optimization problem for SVM

Finally, once $(\boldsymbol{w}, \boldsymbol{b})$ are found, the label of a new point $\boldsymbol{x}$ is simply

$$class(\mathbf{x}) = \text{sgn}(\mathbf{w}^T\mathbf{x} + b) = \begin{cases} 1 \; if \; \mathbf{w}^T\mathbf{x} + b \geq 0 \\ -1 \; others \end{cases}$$

**Question**: Why use $y_n(\mathbf{w}^T\mathbf{x} + b)$and not use $|\mathbf{w}^T\mathbf{x} + b|$ ?

# I. Overview

## Dual problem for SVM

***Lagrange multiplier method***

Let $U$ be an open set in $\mathbb{R}^n$, and suppose that you want to maximize or minimize a smooth function $f: U \to \mathbb{R}$ subject to the constraint $g(x) = c$ for some smooth function $g: U \to \mathbb{R}$ and some constant $\boldsymbol{c}$. If $\boldsymbol{f}$ has a local maximum or local minimum at a point $\boldsymbol{a}$ and if $\nabla g(\boldsymbol{a}) \neq \boldsymbol{0}$, then there is a scalar $\lambda$ such that:

$$\nabla f(\boldsymbol{a}) = \lambda \nabla g(\boldsymbol{a})$$

The scalar $\lambda$ is called a ***Lagrange multiplier.***

Don Shimamoto, *Multivariable Calculus,* Swarthmore College

# I. Overview

## Dual problem for SVM

***General Optimization Problem***

We consider the following general optimization problem:

**1. Objective Function:**

$$\mathbf{x} *= arg\min_{\mathbf{x}} f_0(\mathbf{x}) \quad (1)$$

**2. Constraints:**

$$subject\ to:$$
$$f_i(x) \leq 0, i = 1,2, \dots, m$$
$$h_j(x) = 0, j = 1,2, \dots, p$$

**3. Domain Definition:**

$$D = (\bigcap_{i=1}^{m} domf_i) \cap (\bigcap_{j=1}^{p} domh_i)$$

Understanding Duality and Lagrangians in Optimization | MLDemystified

# I. Overview

## Dual problem for SVM

***The Lagrangian Function***

$$L(\boldsymbol{x}, \lambda, \nu) = f_0(\boldsymbol{x}) + \sum_{j=1}^{m} \lambda_i \, f_i(\boldsymbol{x}) + \sum_{j=1}^{p} \nu_j h_j(\boldsymbol{x})$$

How to optimize it ???, We need to know that, Classical Lagrange just helps us to find the stationary point, not specifically point like the min or max value, so we need to find other methods that can find the optimum value.

So, we have **duality** and some **novelty conditions** that help us to find the minimum value of $F(\boldsymbol{x})$

# I. Overview

## Dual problem for SVM

***Duality***

In mathematical optimization theory, **duality** or the **duality principle** is the principle that optimization problems may be viewed from either of two perspectives, the **primal problem** or the **dual problem**.

***The Lagrangian dual function:***

$$\mathrm{g}(\lambda, \nu) = \inf_{x \in D} L(\boldsymbol{x}, \lambda, \nu) = \inf_{x \in D} \left(f_0(\boldsymbol{x}) + \sum_{j=1}^{m} \lambda_i \boldsymbol{f}_i(\boldsymbol{x}) + \sum_{j=1}^{p} \nu_j \boldsymbol{h_j}(\boldsymbol{x})\right)$$

In other words, for each fixed pair of multipliers $(\lambda, \nu)$, we look for the value of **x** that makes the Lagrangian as small as possible, that minimum becomes the value of $g(\lambda, \nu)$

# I. Overview

## Dual problem for SVM

***Slater's Condition***

The Slater condition says that, if there exists $\boldsymbol{w}, \boldsymbol{b}$ such that:

$$1 - y_n(\mathbf{w}^T\mathbf{x}_n + b) < 0, n = 1, \dots, N$$

then ***strong duality*** holds.

***Lagrangian of the SVM problem***

The Lagrangian of problem is:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda}) = \frac{1}{2}\left|\left|\mathbf{w}\right|\right|_2^2 + \sum_{n=1}^{N} \lambda_n(\,1 - y_n(\mathbf{w}^T\mathbf{x}_n + b))$$

with $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \dots, \lambda_N]$ và $\lambda \geq 0, \forall n = 1, 2, \dots, N$

# I. Overview

## Dual problem for SVM

***Lagrange dual function***

The Lagrange dual function is defined as:

$$g(\boldsymbol{\lambda}) = \min_{\mathbf{w},b} \mathcal{L}(\mathbf{w}, b, \lambda)$$

with $\lambda \succcurlyeq 0$.

And the Lagrange dual function is ***concave function.***

***Lagrange dual problem***

From there, combining the Lagrange dual function and the constraints of $\lambda$, we will obtain the Lagrange dual problem:

$$\boldsymbol{\lambda} = \arg\max_{\boldsymbol{\lambda}} \mathcal{L}(\mathbf{w}, b, \lambda)$$

subject to $\lambda \succcurlyeq 0$ and $\sum_{n=1}^{N} \lambda_n y_n = 0$.

# I. Overview

## Dual problem for SVM

***Karush–Kuhn–Tucker (KKT) conditions***

The Karush-Kuhn-Tucker (KKT) conditions are a generalization of Lagrange multipliers, and give a set of necessary for conditions optimality for systems involving both equality and inequality constraints.

- **Stationarity:** *existence of optimal solution of Lagrange function.*
- **Primal Feasibility:** *optimal solution that satisfies the conditions of the original problem.*
- **Dual Feasibility:** *optimal solution that satisfies the conditions of the dual problem.*
- **Complementary Slackness:** *For each inequality constraint, either its dual coefficient is positive and the constraint is “closed”, or the constraint is “open” and the dual coefficient is 0.*

Adam Rumpf, *Introduction to the Karush-Kuhn-Tucker (KKT) Conditions,* Illinois Institute of Technology - Department of Applied Mathematics

# I. Overview

## Dual problem for SVM

***Karush–Kuhn–Tucker (KKT) conditions for SVM***

*- Stationarity:*

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \lambda)}{\partial \mathbf{w}} = 0 \text{ and } \frac{\partial \mathcal{L}(\mathbf{w}, b, \lambda)}{\partial b} = 0$$

*- Primal Feasibility:* Slater condition.

*- Dual Feasibility:* $\lambda \geqslant 0$

*- Complementary Slackness:* $\lambda_n(1 - y_n(\mathbf{w}^T \mathbf{x}_n + b)) = 0$

***In short, if a convex problem satisfies the Slater criterion, then strong duality is satisfied. And if strong duality is satisfied, then the solution to the problem is the solution of the KKT condition system.***

# I. Overview

## Discussions

***Question:*** *Why is it called “Support Vector Machine”?*

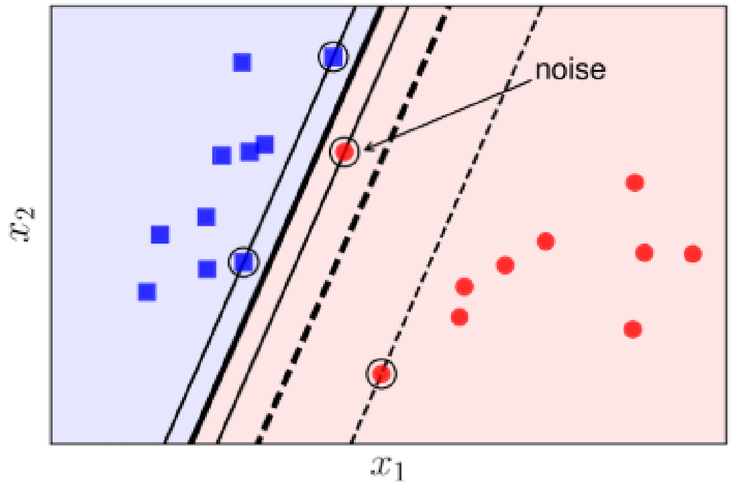
***Question:*** *What problems does SVM help solve in machine learning?*

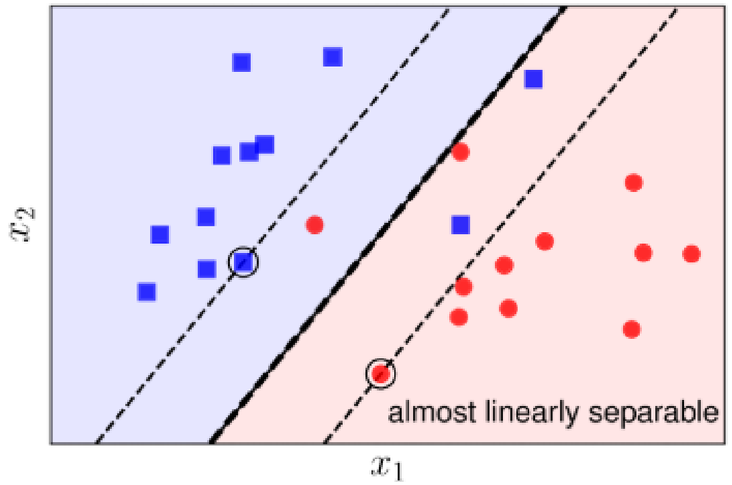
***Question:*** *Why don't we solve the problem in its original form but have to convert it to the KKT system to solve it?*

# II. Soft-margin Support Vector Machine

# II. Soft-margin SVM

## Introduction



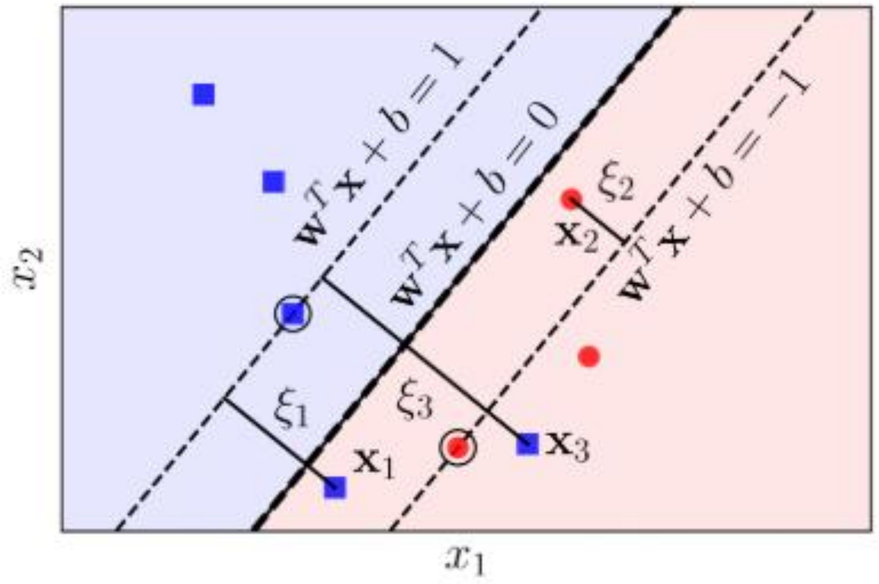




https://machinelearningcoban.com/2017/04/13/softmarginsmv/

# II. Soft-margin SVM

## Mathematical analysis

For points in the safe zone, $\xi_n = 0$. Points in the unsafe zone but still on the right side of the boundary correspond to $0 < \xi_n < 1, \text{e.g. } \mathbf{x_2}$ . Points on the opposite side of their class from the boundary correspond to $\xi_n > 1, \text{e.g. } \mathbf{x_1} \text{ and } \mathbf{x_3}$




https://machinelearningcoban.com/2017/04/13/softmarginsmv/

# II. Soft-margin SVM

## Mathematical analysis

***Slack variable***

For points $\boldsymbol{x_n}$ in the safe region, $\xi_n = 0$. For each point in the unsafe region such as $x_1$, $x_2$ or $x_3$, we have $\xi_i > 0$. Notice that if $y_i = \pm 1$ is the label of $x_i$ in the unsafe region, then $\xi_i = |\mathbf{w}^T \mathbf{x}_i + b - y_i|$

We have the optimization problem in standard form for Soft-margin SVM:

$$(\mathbf{w}, b, \xi) = \arg\min_{\mathbf{w},b,\xi} \frac{1}{2} \left|\left|\mathbf{w}\right|\right|_2^2 + C. \sum_{n=1}^{N} \xi_n$$

Subject to

$$1 - \xi_n - y_n(\mathbf{w}^T \mathbf{x}_n + b) \leq 0, n = 1, \dots, N$$
$$-\xi_n \leq 0, n = 1, \dots, N$$

# II. Soft-margin SVM

## Dual problem for Soft-margin SVM

***Slater's Condition***

For all $n = 1,2, \dots, N$ and for any $(\mathbf{w}, b)$, we can always choose positive scalars $\xi_n, n = 1,2, \dots, N$, large enough so that

$$y_n(\mathbf{w}^T\mathbf{x}_n + b) + \xi_n > 1, \forall n = 1, 2, \dots, N$$

Hence, this problem satisfies Slater's condition.

***Lagrangian of the Soft-margin SVM problem***

The Lagrangian of problem is:

$$\mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu) = \frac{1}{2}||w||_2^2 + C.\sum_{n=1}^{N}\xi_n + \sum_{n=1}^{N}\lambda_n(1 - \xi_n - y_n(\mathbf{w}^T\mathbf{x}_n + b)) - \sum_{n=1}^{N}\mu_n\xi_n$$

with $\lambda = [\lambda_1, \lambda_2, \dots, \lambda_N]$ and $\lambda \geq 0, \forall n = 1, 2, \dots, N$

$\mu = [\mu_1, \mu_2, \dots, \mu_N]$ and $\mu \geq 0, \forall n = 1, 2, \dots, N$

# II. Soft-margin SVM

## Dual problem for Soft-margin SVM

***Lagrange dual function***

The Lagrange dual function is defined as:

$$g(\lambda, \mu) = \min_{\mathbf{w}, b} \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu)$$

with $\lambda \succcurlyeq 0, \mu \succcurlyeq 0$.

***Lagrange dual problem***

From there, combining the Lagrange dual function and the constraints of $\lambda$, we will obtain the Lagrange dual problem:

$$\lambda = \arg\max_{\lambda} \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu)$$

subject to $0 \leq \lambda_n \leq C$ and $\sum_{n=1}^{N} \lambda_n y_n = 0, \forall n = 1, 2, \ldots, N$.

# II. Soft-margin SVM

## Dual problem for Soft-margin SVM

***KKT conditions for Soft-margin SVM***

*- Stationarity:*

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu)}{\partial \mathbf{w}} = 0 \text{ and } \frac{\mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu)}{\partial b} = 0 \text{ and } \frac{\mathcal{L}(\mathbf{w}, b, \xi, \lambda, \mu)}{\partial \xi_n} = 0$$

*- Primal Feasibility:* Slater condition.

*- Dual Feasibility:* $\lambda \geqslant 0$, $\mu \geqslant 0$

*- Complementary Slackness:* $\lambda_n(1 - \xi_n - y_n(\mathbf{w}^T \mathbf{x}_n + b)) = 0$

**Question:** Is there any other way to solve this problem?

# II. Soft-margin SVM

## The unconstrained optimization problem for Soft-margin SVM

***Equivalent unconstrained optimization problem***

We have the first condition

$$1 - \xi_n - y_n(\mathbf{w}^T\mathbf{x}_n + b) \leq 0, n = 1, \dots, N$$
$$\Rightarrow \xi_n \geq 1 - y_n\,(\mathbf{w}^T\mathbf{x}_n + b), n = 1, \dots, N$$

and the second condition is $\xi_n \geq 0$. So, this problem become

$$(\mathbf{w}, b, \xi) = \arg\min_{\mathbf{w},b,\xi} \frac{1}{2}\left|\left|\mathbf{w}\right|\right|_2^2 + C.\sum_{n=1}^{N} \xi_n$$

subject to

$$\xi_n = \max_{(\mathbf{w},b)}\left(0, 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\right)$$

Hence,

$$(\mathbf{w}, b, \xi) = \arg\min_{\mathbf{w},b,\xi} \frac{1}{2}\left|\left|\mathbf{w}\right|\right|_2^2 + C.\sum_{n=1}^{N} \max_{(\mathbf{w},b)}\left(0, 1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\right) \triangleq \arg\min_{(\mathbf{w},b)} J(\mathbf{w}, b)$$

# II. Soft-margin SVM

## The unconstrained optimization problem for Soft-margin SVM

***Hinge Loss***

The hinge loss for a single data point is given by

$$L\big(y, f(\mathbf{x})\big) = \max\big(0, 1 - y.f(\mathbf{x})\big)$$

where

- $y$: Actual label of the data point, either $+1$ or $-1$ (SVMs require binary labels in this format).
- $f(\mathbf{x})$: Predicted score (e.g., the raw output of the model before applying a decision threshold).
- $\max(0, \dots)$: Ensures the loss is non-negative.

What is Hinge loss in Machine Learning?

# II. Soft-margin SVM

## The unconstrained optimization problem for Soft-margin SVM

***Hinge Loss for Soft-margin SVM***

$$L(\mathbf{w}, b) = \sum_{n=1}^{N} \max\left(0, 1 - y_n(\mathbf{w}^T \mathbf{x}_n + b)\right) + \frac{\lambda}{2} ||\boldsymbol{w}||_2^2$$

***Optimize the hinge loss***

- ❖ Gradient Descent
- ❖ Mini-batch Gradient Descent

Gradient descent – Wikipedia

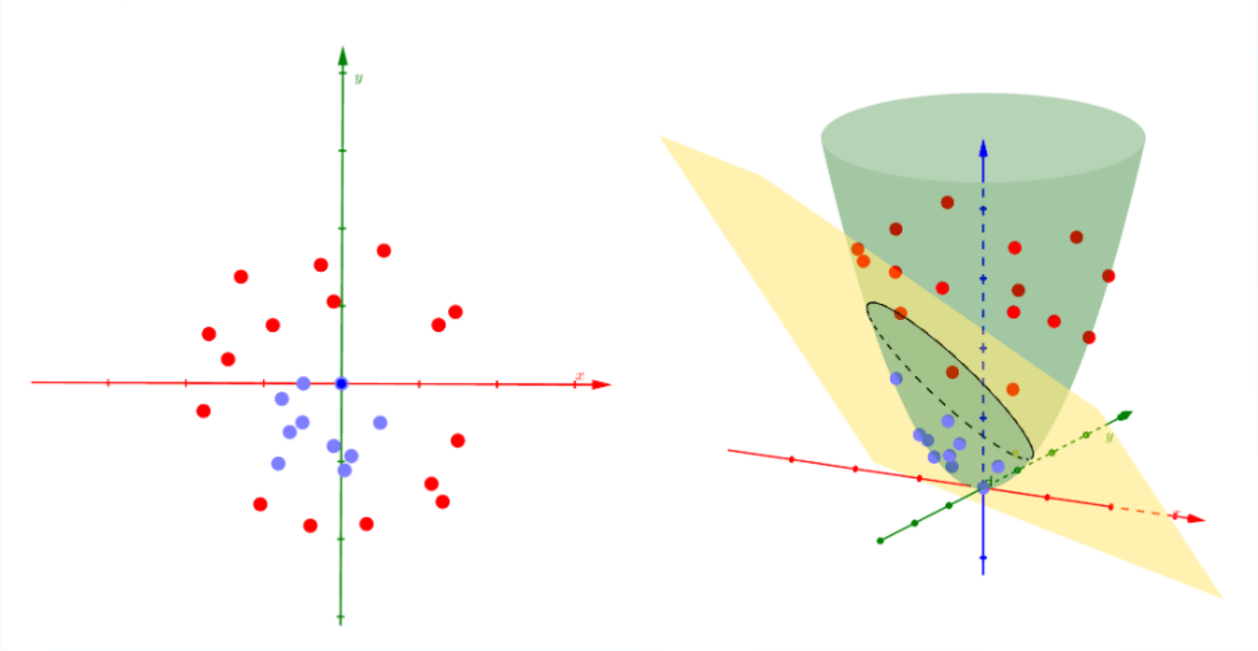
Mini-Batch Gradient Descent in Deep Learning - GeeksforGeeks

# III. Kernel Support Vector Machine

# I. Kernel SVM

## Introduction

The basic idea of Kernel SVM—and kernel methods in general—is to find a transformation that maps the original data, which is not linearly separable, into a new feature space. In this new space, the data becomes linearly separable.



https://machinelearningcoban.com/2017/04/22/kernelsmv/

# I. Kernel SVM

## Foundation of Mathematics

The dual problem for the Soft-Margin SVM with nearly linearly separable data can be written as

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \mathbf{x}_n^T \mathbf{x}_m$$

subject to:

$$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$0 \leq \lambda_n \leq C, \forall n = 1, 2, \dots, N$$

***Problem:*** With real-world data, it is very difficult to obtain data that is nearly linearly separable.

# I. Kernel SVM

## Foundation of Mathematics

Suppose we can find a function $\boldsymbol{\Phi}(.)$ such that, after being transformed into a new space, each data point $\mathbf{x}$ becomes $\boldsymbol{\Phi}(\mathbf{x})$, and in this new space, the data becomes nearly linearly separable.

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \boldsymbol{\Phi}(\mathbf{x}_n^T) \boldsymbol{\Phi}(\mathbf{x}_m)$$

subject to:

$$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$0 \leq \lambda_n \leq C, \forall n = 1, 2, \dots, N$$

$\boldsymbol{\Phi}(\mathbf{x}_n^T) \boldsymbol{\Phi}(\mathbf{x}_m)$ is dot product

# I. Kernel SVM

## Foundation of Mathematics

By defining the kernel function $\mathbf{K}(\mathbf{x}_n, \mathbf{x}_m) = \mathbf{\Phi}(\mathbf{x}_n^T)\mathbf{\Phi}(\mathbf{x}_m)$ we can rewrite problem and expression as follows:

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \mathbf{K}(\mathbf{x}_n, \mathbf{x}_m)$$

subject to:

$$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$0 \leq \lambda_n \leq C, \forall n = 1, 2, \dots, N$$

And

$$\sum_{m \in S} \lambda_m y_m \mathbf{K}(\mathbf{x}_m, \mathbf{x}) + \frac{1}{N_{\mathcal{M}}} \sum_{m \in \mathcal{M}} \left( y_n - \sum_{m \in S} \lambda_m y_m \mathbf{K}(\mathbf{x}_n, \mathbf{x}_m) \right)$$

# I. Kernel SVM

## Kernel function

***Mercer's Condition***

In mathematics, a real-valued function $K(x, y)$ is said to fulfill Mercer's condition if for all square-integrable functions $g(x)$ one has

$$\iint g(x)K(x,y)g(y)dxdy \geq 0$$

**Discrete analog**

This is analogous to the definition of a positive-semidefinite matrix. This is a matrix $K$ of dimension $N$, which satisfies, for all vectors $g$, the property

$$(g, Kg) = g^T.Kg = \sum_{i=1}^{N}\sum_{j=1}^{N} g_i K_{ij} g_j \geq 0$$

Mercer's theorem - Wikipedia

# I. Kernel SVM

## Kernel function

***Properties of kernel functions:***

- *Symmetry*: This is easy to see, since the dot product of two vectors is symmetric.
- A kernel function must satisfy **Mercer's condition**.

***Some commonly used kernel functions:***

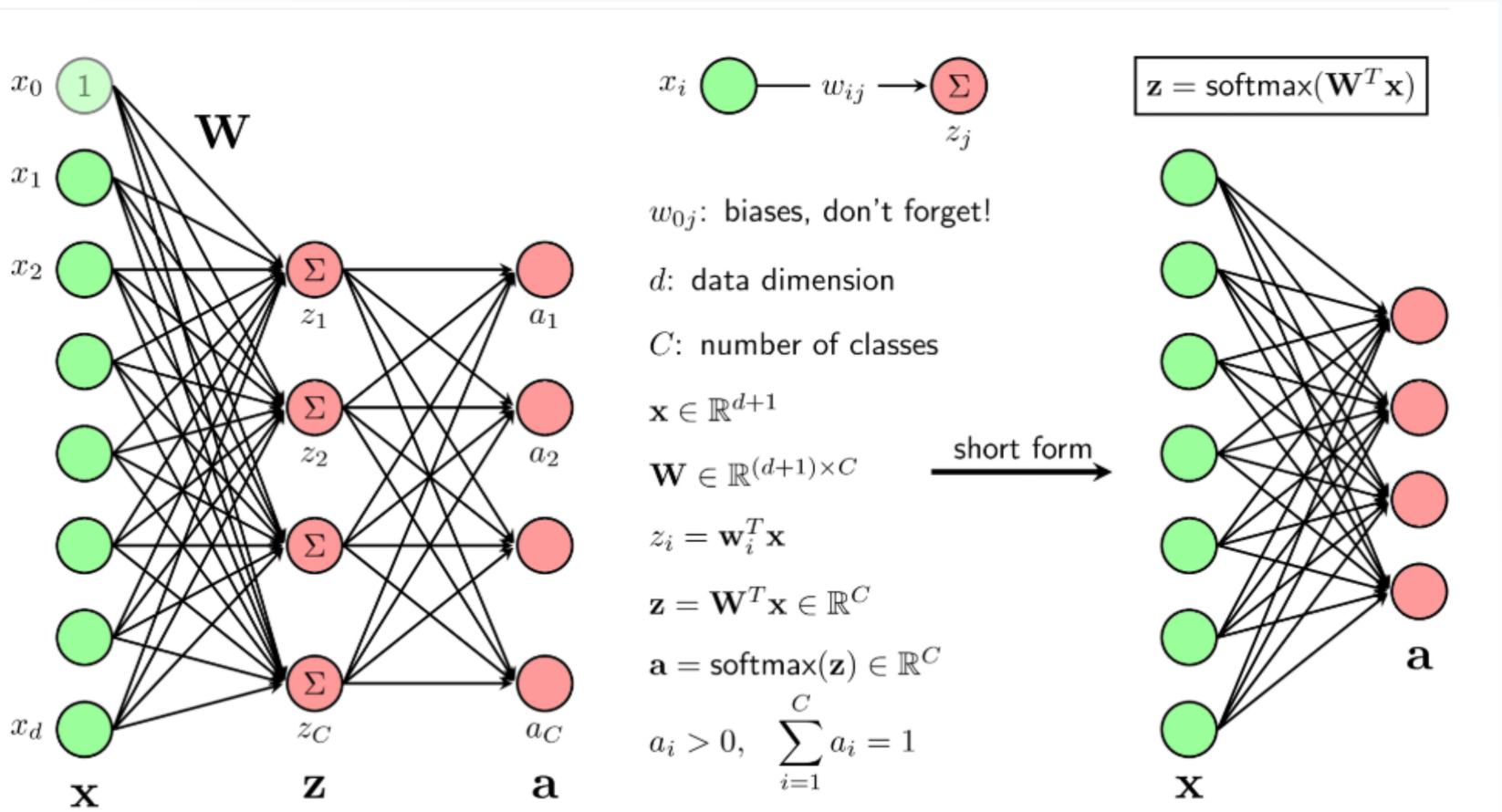
| Name | Formula | Kernel String | Parameter Settings |
|---|---|---|---|
| linear | $x^T z$ | `'linear'` | none |
| polynomial | $(r + \gamma\, x^T z)^d$ | `'poly'` | $d$: degree, $\gamma$: gamma, $r$: coef0 |
| sigmoid | $\tanh(\gamma\, x^T z + r)$ | `'sigmoid'` | $\gamma$: gamma, $r$: coef0 |
| rbf | $\exp\left(-\gamma \lVert x - z \rVert_2^2\right)$ | `'rbf'` | $\gamma > 0$: gamma |

# IV. Multi-class Support Vector Machine

# I. Multi-class SVM

## Introduction

***Question:*** What do you think about combining SVM and Neural Networks?



# I. Multi-class SVM

## Building a loss function for Multi-class SVM

For the entire dataset $\mathbf{X} = [x_1, x_2, \dots, x_N]$, the total loss is

$$L(\mathbf{X}, \mathbf{y}, \mathbf{W}) = \frac{1}{N} \sum_{n=1}^{N} \sum_{j \neq y_n} \max\left(\Delta - z_{y_n}^n + z_j^n\right)$$

where $\mathbf{y} = [y_1, y_2, \dots, y_N]$ is the vector of correct class labels for all training examples. The factor $\frac{1}{N}$ computes the average loss, preventing this expression from becoming excessively large and causing numerical overflow.

***Problem:*** What if **W** gets too large?

# I. Multi-class SVM

## Building a loss function for Multi-class SVM

***Frobenius Norm***

For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the most commonly used norm is the Frobenius norm, denoted $\left|\left|\mathbf{A}\right|\right|_F$, which is the square root of the sum of the squares of all entries of the matrix.

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

***Regularization***

$$L(\mathbf{X}, \mathbf{y}, \mathbf{W}) = \frac{1}{N}\sum_{n=1}^{N}\sum_{j \neq y_n} \max\left(\Delta - z_{y_n}^n + z_j^n\right) + \left|\left|\mathbf{W}\right|\right|_F$$

# I. Multi-class SVM

## Calculating the loss function and its derivative

- Compute the loss function and its derivative in a naive way.
- Calculate the loss function and its derivative by vectorizing.

# Homework

## *“Applying SVM to mushroom classification problem”*

This is the topic of the OAI-HCMC 2025 contest.

ML IoT
HCMUT EE MACHINE LEARNING & IOT LAB

# THANK YOU

## CONTACT US


403.1 H6, BKHCM Campus 2

mlandiotlab@gmail.com

mliotlab.github.io

facebook.com/hcmut.ml.iot.lab

youtube.com/@mliotlab