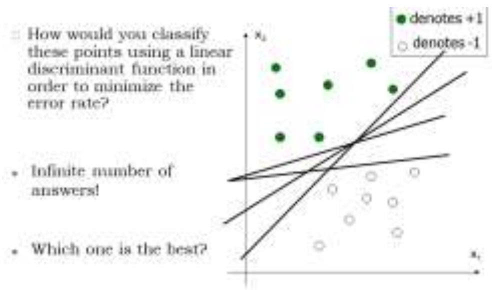
Support Vector Machine

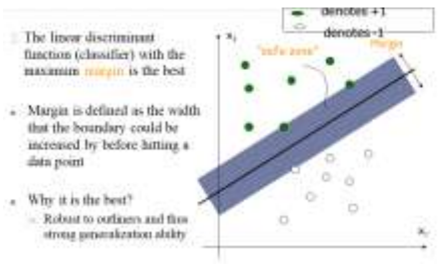
SVM is a kernel-based supervised computational learning method which is based on the statistical learning theory . A SVM kernel maps the input data points of the original input space to the higher dimensional feature space. Here an optimal hyperplane is determined which is then used to define a decision boundary which separates the input data points of different classes and recognizes patterns for classification and regression.

What is a Support Vector Machine (SVM)?

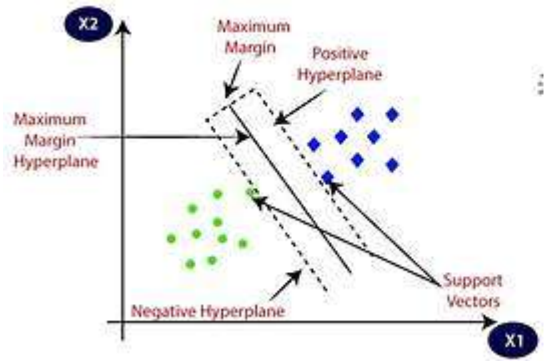
SVM is a **supervised machine learning algorithm** mainly used for **classification** and sometimes regression tasks. It finds the best boundary (called a **hyperplane**) that separates data points of different classes with the **maximum margin**.



- How would you classify these points using a linear discriminant function in order to minimize the error rate?
- Infinite number of answers!
- Which one is the best?



- The linear discriminant function (classifier) with the maximum margin is the best
- Margin is defined as the width that the boundary could be increased by before hitting a data point
- Why it is the best?
  - Robust to outliers and thus strong generalization ability



SVM tries to:

- Find a **hyperplane** that separates classes with the **maximum margin**.
- The **margin** is the distance between the hyperplane and the nearest data points from each class (called **support vectors**).

Terminology

Term	Meaning

Term	Meaning
Hyperplane	A decision boundary that separates data points of different classes.
Support Vectors	Data points closest to the hyperplane; they define the margin.
Margin	Distance between the hyperplane and the support vectors.
Kernel	Function that transforms data into higher dimensions to make it linearly separable.

Key Concepts:

- **Hyperplane:** A decision boundary that separates classes in the feature space. For 2D data, it’s a line; for 3D, a plane; and for higher dimensions, a hyperplane.
- **Margin:** The distance between the hyperplane and the nearest data points of each class. SVM tries to maximize this margin to improve generalization.
- **Support Vectors:** The data points closest to the hyperplane; these are critical in defining the position and orientation of the hyperplane.
- **Kernel Trick:** When data is not linearly separable, SVM uses kernel functions (like polynomial, RBF, sigmoid) to transform data into a higher-dimensional space where it becomes separable.

How SVM works:

1. Find the hyperplane that best separates the classes by maximizing the margin.
2. If data is linearly separable, SVM finds the optimal linear boundary.
3. If data is not linearly separable, use kernels to map data to a higher dimension.
4. The model focuses on support vectors, ignoring points far away from the boundary.

Advantages:

- Effective in high-dimensional spaces.
- Works well when the number of features > number of samples.
- Can model non-linear boundaries with kernels.

Disadvantages:

- Can be slow on very large datasets.
- Choosing the right kernel and parameters can be tricky.
- Less interpretable compared to simpler models.

Types of SVM

1. **Linear SVM (**Linearly Separable Case**)**
2. **Non Linear SVN (**Non-Linearly Separable Data)

1. Linear SVM

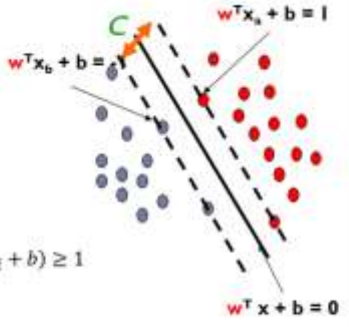
Training Data

$$(x_i, y_i),\ i = 1, 2, \ldots, n$$

Symbol	Meaning
x_i	The **input feature vector** for the i-th data point (e.g., $x_i = [x_{i1}, x_{i2}, \ldots, x_{id}]^T$)
y_i	The **class label** for x_i: usually $+1$ or -1 for binary classification
n	The **total number of training samples**
d	The **number of features** (dimensions) in each input vector

Supervised learning model used for classification and regression analysis

- SVMs maximize the margin around the separating hyperplane.
- Hyperplane equation
  $w^T x + b = 0$
- Extra scaling hyperplane equation
  $\min_{i=1,\ldots,n} |w^T x_i + b| = 1$
- This implies
  $w^T(x_a - x_b) = 2$
  $c = \|x_a - x_b\|_2 = 2/\|w\|_2$
- SVM formulation
  $\min_{w} \frac{1}{2} w^T w \quad Subject\ to\ \ y_i(w^T x_i + b) \geq 1$



The decision boundary (Hyperplane)

$$w^T x + b = 0$$

Symbol	Meaning
w	**Weight vector** (normal to the hyperplane). It determines the orientation of the hyperplane.
x	**Feature vector** (input data point).
b	**Bias (intercept)** term — determines the offset of the hyperplane from the origin.
$w^T x + b$	**Linear combination** of features — used to compute the signed distance from the hyperplane.

✅ If $w^T x + b > 0 \rightarrow$ class $+1$
✅ If $w^T x + b < 0 \rightarrow$ class -1

Given training data:

$$(x_i, y_i),\ i = 1, 2, ..., n$$

where $x_i \in \mathbb{R}^n$ and $y_i \in \{-1, +1\}$.

We want to find a hyperplane:

$$w^T x + b = 0$$

that separates the classes.

4.2 Optimization Objective

We need to **maximize the margin**:

$$\text{Margin} = \frac{2}{||w||}$$

Equivalent to **minimizing** $||w||^2$, subject to correct classification:

$$y_i(w^T x_i + b) \geq 1, \quad \forall i$$

$y_i(w^T x_i + b) \geq 1$	**Constraint** ensuring all points are correctly classified and outside the margin.

Explanation of the constraint:

- If $y_i = +1$: $w^T x_i + b \geq 1$
- If $y_i = -1$: $w^T x_i + b \leq -1$

Both cases ensure that the point lies on the correct side of the margin.

To solve the constrained optimization, we introduce **Lagrange multipliers** α_i:

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_i \alpha_i[y_i(w^T x_i + b) - 1]$$

Symbol	Meaning
L	**Lagrangian function** — combines objective and constraints.
α_i	**Lagrange multiplier** associated with the constraint of the i-th data point.
$\alpha_i \geq 0$	Non-negative constraint (comes from KKT conditions).

Dual Form (after optimization):

$$\max_{\alpha} \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i^T x_j$$

subject to:

$$\sum_i \alpha_i y_i = 0, \quad \alpha_i \geq 0$$

Symbol	Meaning
$x_i^T x_j$	Dot product between training points x_i and x_j.
$y_i y_j$	Product of labels (ensures class consistency).
$\sum_i \alpha_i y_i = 0$	Constraint that maintains balance between classes.

4.4 Decision Function

Once trained:

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

Or in dual form:

$$f(x) = \text{sign}\left(\sum_i \alpha_i y_i (x_i^T x) + b\right)$$

Example:

Problem Setup:

We have a training dataset with n samples:

$$\{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$$

where $\mathbf{x}_i \in \mathbb{R}^d$ (feature vectors), and $y_i \in \{+1, -1\}$ (class labels).

Goal:

Find a **hyperplane** that separates the two classes:

$$\mathbf{w} \cdot \mathbf{x} + b = 0$$

- $\mathbf{w}$ is the weight vector (normal to the hyperplane),
- b is the bias term (offset),
- $\cdot$ denotes the dot product.

Margin:

- For any point $\mathbf{x}$, the **distance to the hyperplane** is:

$$\frac{|\mathbf{w} \cdot \mathbf{x} + b|}{\|\mathbf{w}\|}$$

- For **correct classification**, we want:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \text{for all } i$$

- The **margin** is $\frac{2}{\|\mathbf{w}\|}$, the distance between the support vectors of the two classes.

Objective:

Maximize the margin, equivalently minimize $\|\mathbf{w}\|$:

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

subject to constraints:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, \quad i = 1, \dots, n$$

Using Lagrange Multipliers:

Define the Lagrangian:

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1\right]$$

where $\alpha_i \geq 0$ are Lagrange multipliers.

- The **Lagrangian function** combines the objective (maximizing margin) and constraints (correct classification with margin).
- **αi** serve as weights on how much each constraint influences the solution.
- By solving the dual problem derived from this Lagrangian, we efficiently find the optimal hyperplane and identify support vectors.

The Lagrangian in SVM

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left[y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1\right]$$

- $\mathbf{w}$: weight vector defining the hyperplane,
- b: bias term,
- $\alpha_i \geq 0$: Lagrange multipliers (one per training example),
- y_i: class label of example i (either +1 or -1),
- $\mathbf{x}_i$: feature vector of example i,
- n: total number of training samples.

Significance of Each Term:

1. $\frac{1}{2}\|\mathbf{w}\|^2$:

   This term represents **half the squared norm** of the weight vector $\mathbf{w}$. Minimizing this term corresponds to **maximizing the margin** between the classes. A smaller $\|\mathbf{w}\|$ means a wider margin, which generally leads to better generalization.

2. **Constraints encoded by the sum:**

   The term inside the summation represents the **classification constraints**:

   $$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0$$

   This ensures that each training example is correctly classified **and** lies outside or on the margin boundary (i.e., at least at distance 1 from the hyperplane).

3. **Lagrange multipliers** $\alpha_i \geq 0$:

   These multipliers weigh the importance of each constraint. If the constraint for a particular training point is **active** (meaning the point lies exactly on the margin or violates the margin), the corresponding α_i is non-zero; otherwise, $\alpha_i = 0$.

   The Lagrangian formulation allows us to solve the **constrained optimization problem** by converting it into an unconstrained problem.

Why Use the Lagrangian?

- The primal optimization problem has inequality constraints.
- Using Lagrange multipliers transforms it into a problem where we can find **stationary points** by differentiating $L(\mathbf{w}, b, \boldsymbol{\alpha})$ with respect to $\mathbf{w}$, b, and α_i.
- The dual form derived from this Lagrangian depends only on the dot products between training points, which is key to using the **kernel trick**.
- The Lagrangian framework identifies the **support vectors** naturally — only those data points with non-zero α_i influence the solution.

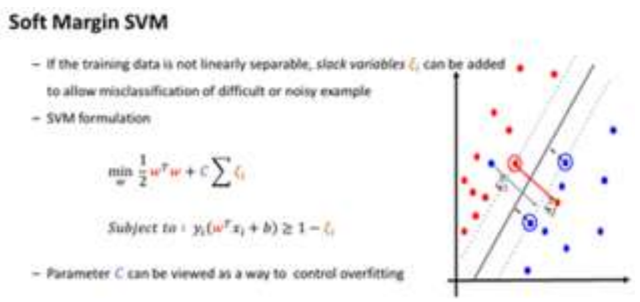
2. Non Linear SVM (Non-Linearly Separable Data)

Real-world data is often not linearly separable. To handle this, SVM introduces two key concepts:

a) Soft Margin

b) Kernel trick

a) Soft Margin



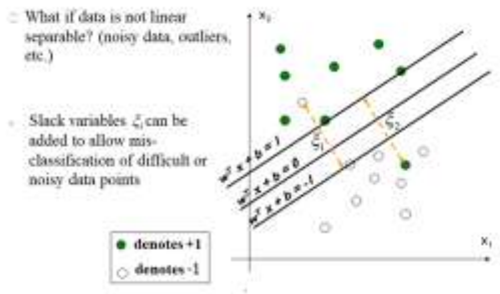
Soft Margin SVM

- If the training data is not linearly separable, *slack variables* ξ_i can be added to allow misclassification of difficult or noisy example
- SVM formulation

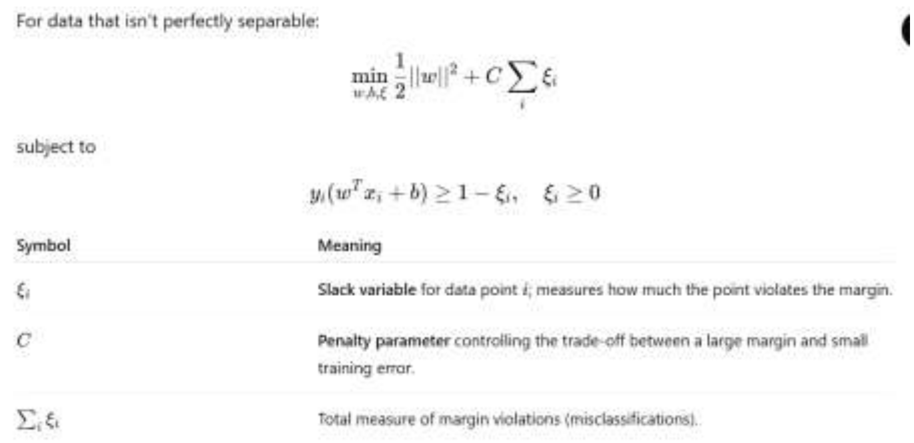
$$\min_{w} \frac{1}{2} w^T w + C \sum \xi_i$$

$$\textit{Subject to}: \; y_i(w^T x_i + b) \geq 1 - \xi_i$$

- Parameter C can be viewed as a way to control overfitting



- What if data is not linear separable? (noisy data, outliers, etc.)
- Slack variables ξ_i can be added to allow mis-classification of difficult or noisy data points



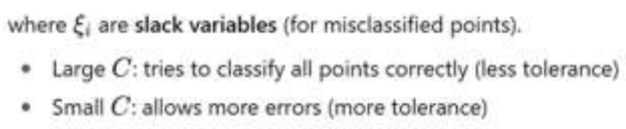
For data that isn't perfectly separable:

$$\min_{w,b,\xi} \frac{1}{2}||w||^2 + C \sum_i \xi_i$$

subject to

$$y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Symbol	Meaning
ξ_i	**Slack variable** for data point i; measures how much the point violates the margin.
C	**Penalty parameter** controlling the trade-off between a large margin and small training error.
$\sum_i \xi_i$	Total measure of margin violations (misclassifications).



where ξ_i are **slack variables** (for misclassified points).

- Large C: tries to classify all points correctly (less tolerance)
- Small C: allows more errors (more tolerance)

b) The Kernel Trick (SVM kernels)

To handle non-linear data, SVM maps inputs into higher-dimensional feature space using a kernel function;

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

$\phi(x)$

Feature mapping function that projects input into higher dimensions.

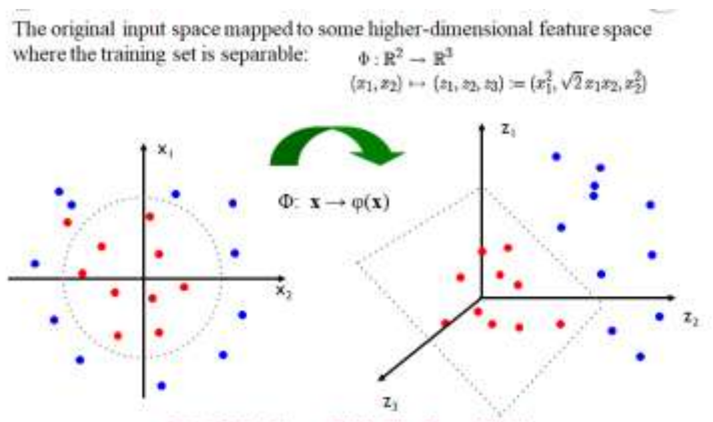
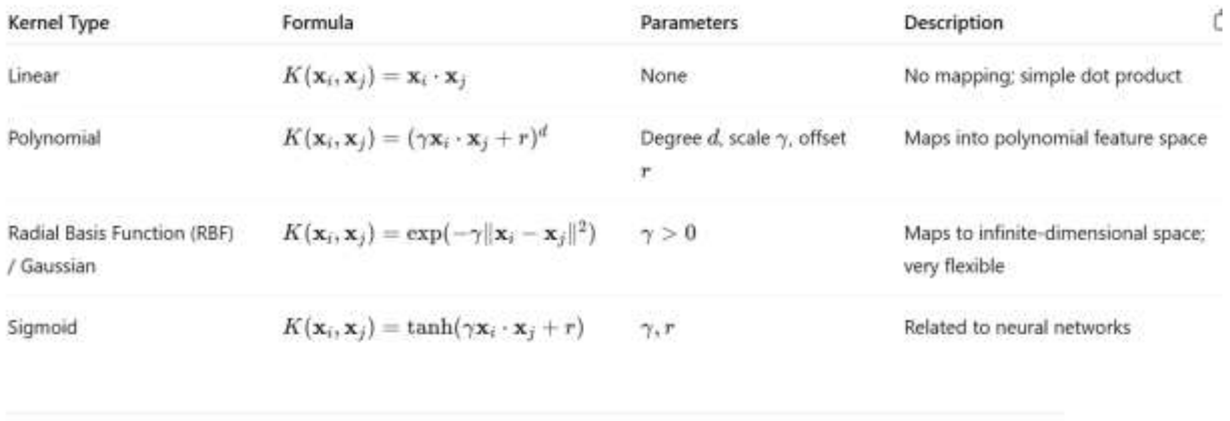

The original input space mapped to some higher-dimensional feature space where the training set is separable:
$\Phi : \mathbb{R}^2 \to \mathbb{R}^3$
$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{2}x_1x_2, x_2^2)$
$\Phi: \mathbf{x} \to \varphi(\mathbf{x})$
x_1
x_2
z_1
z_2
z_3

Fig 4.2: Nonlinear SVMs: The Kernel Trick

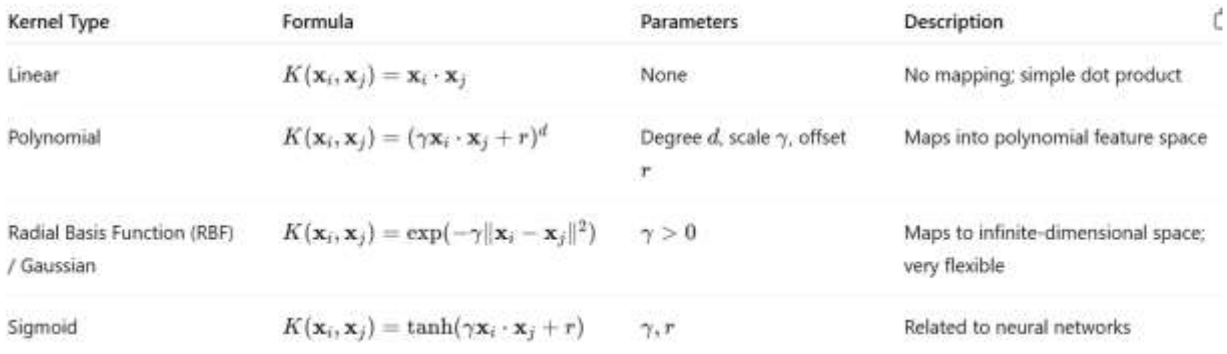
To enhance linear separability, the original input space is mapped into high dimensional dot-product space called the feature space. One of the advantages of SVM is that generalization performance can be improved by proper selection of kernel. s. Here, *γ*, *η*, and *d* are kernel parameters and *x*i and *xj* are the coordinates of input space. In SVM, according to the need of classification, a kernel has to be selected and the values of kernel parameter *γ*, *η*, d , and the margin parameter *C* have to be determined.



Kernel Type	Formula	Parameters	Description
Linear	$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$	None	No mapping; simple dot product
Polynomial	$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\mathbf{x}_i \cdot \mathbf{x}_j + r)^d$	Degree d, scale γ, offset r	Maps into polynomial feature space
Radial Basis Function (RBF) / Gaussian	$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$	$\gamma > 0$	Maps to infinite-dimensional space; very flexible
Sigmoid	$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma\mathbf{x}_i \cdot \mathbf{x}_j + r)$	γ, r	Related to neural networks

Why Kernels Are Powerful:

- They allow SVMs to **fit very complex boundaries** without explicitly handling high-dimensional vectors.
- The **dual optimization problem** depends only on kernels.
- You can customize kernels to encode domain knowledge.



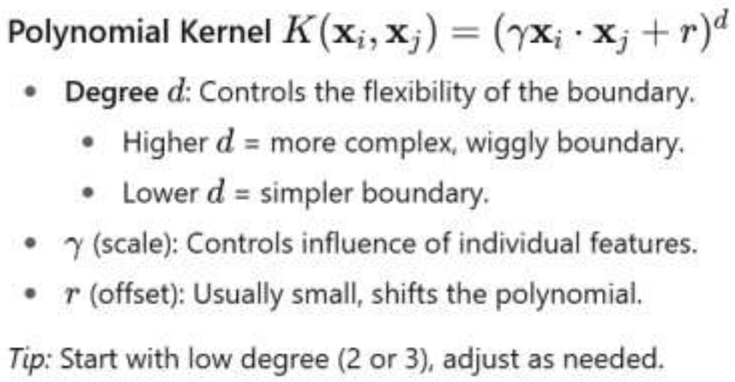
Kernel Type	Formula	Parameters	Description
Linear	$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i \cdot \mathbf{x}_j$	None	No mapping; simple dot product
Polynomial	$K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i \cdot \mathbf{x}_j + r)^d$	Degree d, scale γ, offset r	Maps into polynomial feature space
Radial Basis Function (RBF) / Gaussian	$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$	$\gamma > 0$	Maps to infinite-dimensional space; very flexible
Sigmoid	$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\gamma \mathbf{x}_i \cdot \mathbf{x}_j + r)$	γ, r	Related to neural networks

Choosing Kernel Parameters

Each kernel has parameters that affect the decision boundary and model performance.

Linear Kernel

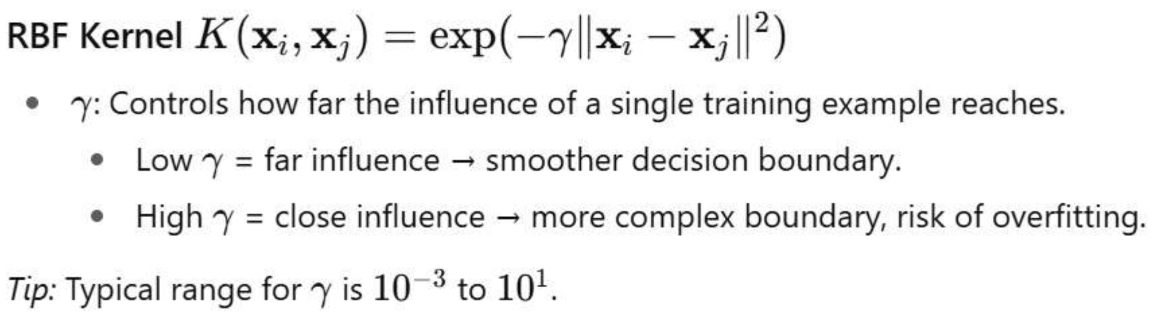
- No parameters to tune.
- Best when data is linearly separable or close to it.



Polynomial Kernel $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma \mathbf{x}_i \cdot \mathbf{x}_j + r)^d$

- **Degree d**: Controls the flexibility of the boundary.
  - Higher d = more complex, wiggly boundary.
  - Lower d = simpler boundary.
- γ (scale): Controls influence of individual features.
- r (offset): Usually small, shifts the polynomial.

Tip: Start with low degree (2 or 3), adjust as needed.



RBF Kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2)$

- γ: Controls how far the influence of a single training example reaches.
  - Low γ = far influence → smoother decision boundary.
  - High γ = close influence → more complex boundary, risk of overfitting.

Tip: Typical range for γ is 10^{-3} to 10^{1}.

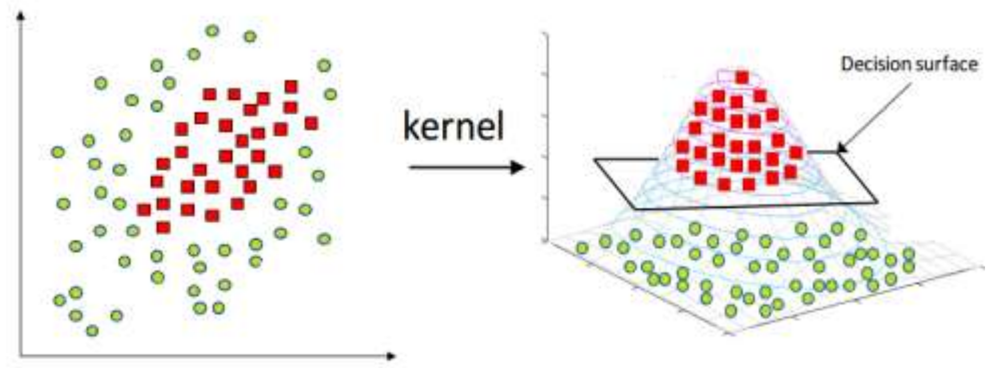
Regularization Parameter CCC

- Controls trade-off between maximizing margin and minimizing classification error.
- High C: Smaller margin, fewer misclassifications → risk of overfitting.
- Low C: Larger margin, more tolerance for misclassifications → risk of underfitting.

Practical Tips for Kernel SVM Tuning

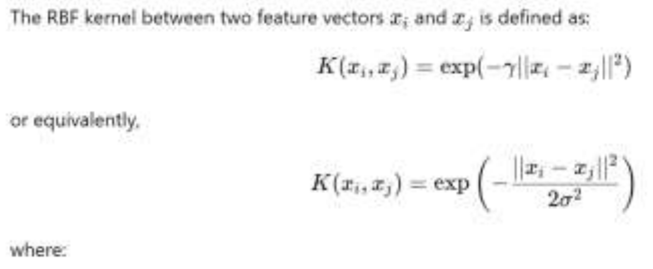
- **Start simple**: Try linear kernel first; if accuracy is low, switch to RBF.
- **Scale your data**: Always standardize features (mean=0, std=1). Kernels depend on distances, so scaling matters a lot.
- **Grid search + cross-validation**: Use tools like Grid SearchCV in scikit-learn to search over γ\gammaγ, C, and degree.
- **Watch out for overfitting**: Very high CCC and γ\gammaγ can fit noise.
- **Check support vectors count**: Too many support vectors may indicate overfitting.
- **Visualize decision boundaries** if data is 2D or 3D to understand model behavior

Radial Basis Function Kernel (RBF)



The **RBF kernel**, also known as the **Gaussian kernel**, is used in SVMs to handle **non-linear relationships** between features and classes. It projects data into an **infinite-dimensional feature space**, allowing the SVM to find a **linear separator in that higher-dimensional space**, even when data is **not linearly separable** in the original input space.

- Measures similarity based on **distance** between points.
- Close points have values near 1; far points tend toward 0.
- Effectively creates a smooth decision boundary that can curve arbitrarily to separate classes.
- The RBF kernel corresponds to an infinite-dimensional space (like an infinite number of features).



The RBF kernel between two feature vectors x_i and x_j is defined as:

$$K(x_i, x_j) = \exp(-\gamma||x_i - x_j||^2)$$

or equivalently,

$$K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)$$

where:

Symbol	**Meaning**
K(xi,xj)	Kernel (similarity measure) between points xi and xj
γ\gammaγ	Kernel parameter that controls the spread of the Gaussian function
σ\sigmaσ	Standard deviation of the Gaussian (related to γ by $\gamma = \frac{1}{2\sigma^2}$)

3. Intuitive Understanding

The RBF kernel measures how **similar** two data points are.

If x_i and x_j are **very close:** $||x_i - x_j||^2 \approx 0$

$$K(x_i, x_j) \approx e^0 = 1$$

⇒ Highly similar.

If x_i and x_j are **far apart:** $||x_i - x_j||^2$ large

$$K(x_i, x_j) \approx e^{-\text{large number}} \approx 0$$

⇒ Almost no similarity.

So, **RBF kernel acts like a similarity function**, assigning high values to close points and low values to distant ones.

---

4. The Role of the Parameter (\gamma)

The parameter (\gamma) (gamma) determines **how much influence a single training**

example has.

(\gamma) Value	**Effect on Model**
Small γ	Wide Gaussian (large σ): each data point influences a large region → **smooth decision boundary**
Large γ	Narrow Gaussian (small σ): each data point influences a small region → **complex decision boundary**, risk of overfitting

Summary:

- **Small (\gamma)** → broader similarity → smoother classifier
- **Large (\gamma)** → sharper similarity → classifier fits training data tightly (may overfit)

---

5. Why RBF Kernel Works So Well

□ **Flexibility:**
Can model very complex, nonlinear relationships.

□ **Few Hyperparameters:**
Only two main parameters to tune: (C) (regularization) and (\gamma) (kernel width).

□ **Generalization Power:**
Good balance between flexibility and regularization if parameters are tuned properly (e.g., via cross-validation).

□ **Mathematically Sound:**
Always satisfies **Mercer's condition** (a requirement for valid kernels in SVM).

---

6. Relationship Between RBF and Linear Kernels

- If (\gamma \to 0):
  (K(x_i, x_j) \to 1) for all pairs → kernel matrix becomes constant → model loses discriminative power.
- If (\gamma) is **very small**, RBF behaves like a **linear kernel**.
- If (\gamma) is **too large**, the model becomes **overly complex** (each point defines its own boundary).

Hence, **choosing the right (\gamma)** is crucial.

Symbol	**Meaning**
(\alpha_i)	Lagrange multiplier (weight of support vector (xi))
(yi)	Label of support vector
(b)	Bias term
(x)	New data point

Thus, the classification depends on how **close** (x) is to each support vector, measured by the RBF kernel.

---

8. Hyperparameter Tuning

You typically tune (C) and (\gamma) together using **grid search with cross-validation**:

Parameter	**Description**	**Typical Range**
(C)	Regularization parameter (penalty for misclassification)	(0.01) – (1000)

Parameter	Description	Typical Range
(\gamma)	Controls influence radius of support vectors	(10^{-4}) – (10^{1})

Goal:

- Find (C) and (\gamma) that minimize validation error.
- Often done using logarithmic grid search and k-fold cross-validation.

---

9. Advantages of RBF Kernel

□ Handles **nonlinear** data very well.
□ Only two parameters → relatively simple to tune.
□ Works effectively in **high-dimensional spaces**.
□ Smooth decision boundaries → good generalization.

---

10. Disadvantages

Sensitive to choice of (\gamma) — poor tuning leads to **underfitting or overfitting**.
Computation can be expensive for **very large datasets** (since it computes pairwise distances).
Results can be hard to **interpret** (since it's nonlinear and high-dimensional).

Example Visualization (Conceptual)

- **Linear SVM:** fails → cannot draw a straight line to separate circles.
- **RBF SVM:** maps data to higher dimensions → finds a **circular boundary** that separates classes perfectly.

So, RBF kernel allows SVM to **draw curved decision boundaries** in the original space.

12. Key points

Concept	**Explanation**
RBF Kernel	Measures similarity between data points using Gaussian function
Equation	(K(xi, xj) = e^{-\gamma
γ (Gamma)	Controls width of Gaussian (model flexibility)
Small γ	Smooth, simple boundary (underfitting risk)
Large γ	Sharp, complex boundary (overfitting risk)
Usage	Default kernel in many SVM implementations (e.g., scikit-learn)

Advantages and Disadvantages of Kernel trick

Advantages

- Works well with **high-dimensional data**.
- Effective even when number of features > number of samples.
- Uses only **support vectors** → efficient model.
- Robust to overfitting (especially with proper kernel and regularization).

Disadvantages

- Computationally expensive for large datasets.
- Choice of **kernel** and **parameters (C, γ)** is critical.
- No direct probability outputs (requires extra calibration).

Applications

- Text classification (spam detection, sentiment analysis)
- Image classification
- Bioinformatics (protein classification, gene expression)
- Handwriting recognition
- Financial time series prediction

Summary

Concept	Description
Goal	Find optimal separating hyperplane
Support Vectors	Critical boundary points
Soft Margin	Allows misclassification
Kernel Trick	Handles nonlinear separability
Hyperparameters	C, \gammaC,γ

Symbol	Meaning
(xi)	Feature vector (input)
(yi)	Label (+1 or -1)
(w)	Weight vector (normal to hyperplane)
(b)	Bias term
(ξi)	Slack variable (margin violation)
(C)	Regularization parameter
(αi)	Lagrange multiplier
(K(xi, xj))	Kernel function
(f(x))	Decision function

Effect of the Margin Parameter and Kernel Parameter

A well chosen parameter is the first step for obtaining a high level of performance of learning machine. The most reliable but time-consuming method of estimating the generalization ability is based on repetitive training of support vector machines. The physical meanings of the parameters margin parameter and kernel parameter are:

1. Effect of Marine Parameter C: C controls the cost of misclassification on the training data. The goal of SVM is to find a hyperplane that would leave the widest possible margin between input points from two classes. The regularization parameter C determines the trade-off between minimizing the training error and maximizing the margin.The value C is called the Error Penalty. A high error penalty will force the SVM training to avoid classification errors.

   A low C makes the decision surface smooth, while a high C aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors. A larger C will result in a larger search space for the optimizer.

2. Effect of kernel Parameter γ: Kernel parameter in the RBF kernel function implicitly defines the non-linear mapping from input space to some high-dimensional feature space. The support vector will be its center and will determine the area of influence this support vector has over the data space. Larger value of kernel parameter will give a smoother decision surface and more regular decision boundary. The (Gaussian) radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. In particular, it is commonly used in support vector machine classification. The RBF kernel on two samples x and x', represented as feature vectors in some input space, is defined as

$$K(x, x') = \exp\left[-\gamma||x - x'||^2\right] \tag{5.10}$$

   $||x - x'||$ may be recognized as the squared Euclidean distance between the two feature vectors whereas γ is a free parameter(radius). Since the value of the RBF kernel decreases with distance and lies between zero (in the limit) and one (when $x = x'$), it has a ready interpretation as a similarity measure. Inverse of the radius of influence of sample selected by the support vector is given by :

$$\gamma = \frac{1}{2\sigma^2} \tag{5.11}$$

Intuitively, the γ parameter defines how far the influence of a single training example reaches, with low values meaning far and high values meaning close. The γ parameters can be seen as the inverse of the radius of influence of samples selected by the model as support vectors.

The behavior of the model is very sensitive to the γ parameter. If γ is too large, the radius of the area of influence of the support vectors only includes the support vector itself and no amount of regularization with C will be able to prevent overfitting. When γ is very small, the model is too constrained and cannot capture the complexity or shape of the data. The region of influence of any selected support vector would include the whole training set. The resulting model will behave similarly to a linear model with a set of hyper planes that separate the centers of high density of any pair of two classes.

The goal is to identify good parameters so that the classifier can accurately predict the unknown target value for each instance. It may not be useful to achieve high training accuracy in a single set of training data since the training and test data may not be consistent all the time. Therefore, a common way is to separate training data into two parts where one part is considered unknown in training the classifier.