| **CS 412 — Introduction to Machine Learning (UIC)** | March 04, 2025 |
|---|---|
| Lecture [14] | |
| *Instructor: Aadirupa Saha* | *Scribe(s): (Please Lukau)* |

# Overview

In the last lecture, we covered the following main topics:

1. Kernel & SVM
2. Different Examples of Kernels
3. Properties of a Kernel Matrix

This lecture focuses on:

1. Soft-Margin SVM
2. SVM for Regression
3. The Perceptron Algorithm

# 1 Soft-Margin SVM

## 1.1 Intro to Soft-Margin SVM

As previously seen, SVMs aim to find an optimal separating line (or hyperplane) for classification with a largest margin possible. Real-world data, however, often cannot be perfectly separated by a straight line (or hyperplane). Real datasets are often noisy and not perfectly linearly separable. With Soft-Margin SVM, we allow the variable $\xi_n$ to permit some data points to lie within the margin (weak margin violation) or on the wrong side of the margin (misclassification or strong margin violation) for better generalization and less overfitting (Figure 1).
See Lecture 13, (Sections 1.2) for more about non-linearly separable data.

### 1.1.1 Primal Form

In order to incorporate margin violation of the n-th point, we reformulate the SVM objective function as

$$\text{Let } w \in \mathbb{R}^d,\ b \in \mathbb{R},\ \xi \in \mathbb{R}^N.$$

$$\min \quad \frac{1}{2}\|w\|^2 \ + \ C\sum_{n=1}^{N}\xi_n$$

$$\text{such that:} \quad y_n\Big(w^T x_n + b\Big) \ \geq \ 1 \ - \ \xi_n, \quad \forall\, n = 1, \ldots, N,$$
$$\xi_n \ \geq \ 0, \quad \forall\, n = 1, \ldots, N.$$

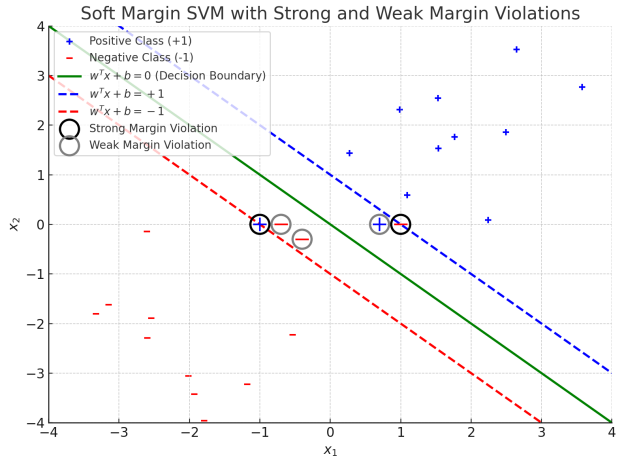


Figure 1: SVM Margin Violation

**Remark 1.** *$C$ is the cost parameter that balances margin size. A large $C$ implies stronger penalty on misclassifications. See ML Lecture slides from McGill University (48-50) to learn more about how to choose $C$.*

*Note :*

- *$\forall n$ such that $\xi_n > 0$ implies a margin violation.*
- *$0 < \xi_n < 1$ implies a weaker margin violation (the point is inside the margin but correctly classified)*
- *$\xi_n \geq 1$ implies a strong margin violation (misclassified points)*

### 1.1.2 Dual Form

For the dual form, you can follow the same procedure as in the hard-margin SVM : form the Lagrangian, derive the corresponding dual problem, apply the KKT conditions, and then solve for w, b, $\xi_n$. The only difference is that the soft-margin problem has extra terms and constraints for the $\xi_n$ values. If you're interested in all the algebraic details of how to set up and solve the dual form, you can find those steps in McGill (slides 50-51).

## 1.2 Hinge Loss : ERM view of Soft-Margin SVM

Recall the Empirical Risk Minimization (ERM) framework. Given training examples $\{(x_n, y_n)\}_{n=1}^N$, we seek a function $f$ from some hypothesis class $\mathcal{F}$ that minimizes the total loss:

$$\arg\min_{f \in \mathcal{F}} \sum_{n=1}^{N} \ell(y_n, \hat{y}_n), \tag{1}$$

where:

- $y_n$ is the true label for instance $n$,

- $\hat{y}_n$ is the predicted label,
- $\ell(y_n, \hat{y}_n)$ is a loss function that penalizes incorrect predictions.

The function class $\mathcal{F}$ consists of all possible functions $f$ mapping the input space $X$ to the reals:

$$\mathcal{F} = \{f : X \to \mathbb{R}\}. \tag{2}$$

For an SVM, the question is: **What is the loss function $\ell$ and hypothesis space $\mathcal{F}$?**

### 1.2.1 Soft-Margin SVM Optimization

SVM optimizes:

$$f(x) = \text{sign}(w^T x + b). \tag{3}$$

The optimal hyperplane is found via a quadratic programming solver using Lagrangian $\alpha^*$:

$$w^* = \sum_{n=1}^{N} \alpha_n^* y_n x_n. \tag{4}$$

The bias term is given by:

$$b^* = \frac{1}{|SV|} \sum_{i:\alpha_i^*>0} \left( \frac{1}{y_i} - w^{*T} x_i \right). \tag{5}$$

### 1.2.2 Prediction Function

Once $w^*$ and $b^*$ are known, the SVM prediction for any new $x \in \mathbb{R}^d$ is:

$$f(x) = \hat{y}(x) = \text{sign}\left(w^{*T} x + b^*\right). \tag{6}$$

with a real-valued score:

$$t(x) = w^{*T} x + b^*. \tag{7}$$

If $t(x) > 0$, the instance is classified as positive $(+1)$.
If $t(x) < 0$, the instance is classified as negative $(-1)$.

### 1.2.3 Hinge Loss

SVM uses the hinge loss function:

$$\ell_{\text{hinge}}(y, f(x)) = \max(0, 1 - y f(x)). \tag{8}$$

This penalizes misclassifications and small-margin correct classifications.
By plugging this $\ell_{\text{hinge}}$ into the ERM framework, we can rewrite the Soft-Margin SVM primal objective

$$\min_{w,b,\{\xi_n\}} \ \frac{1}{2} \|w\|^2 \ + \ C \sum_{n=1}^{N} \xi_n, \quad .$$

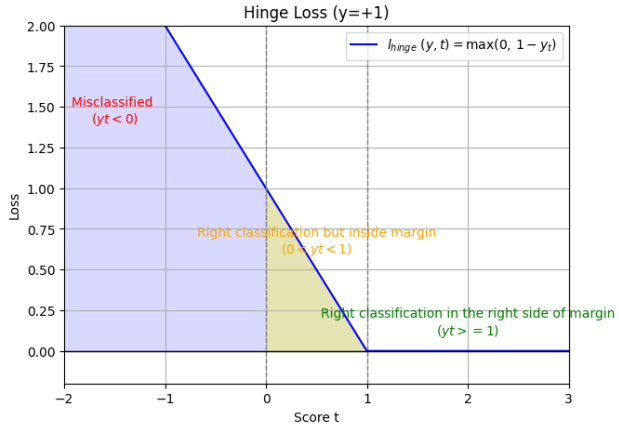


Figure 2: Hinge Loss Visualization

as the complete regularized ERM problem for SVM :

$$\min_{w,b} \sum_{n=1}^{N} \max(0, 1 - y_n(w^T x_n + b)) + \frac{\|w\|^2}{2C}, w \in \mathbb{R}^n, b \in \mathbb{R} \tag{9}$$

where $C$ is a regularization parameter controlling the trade-off between margin maximization and classification error.

**Hypothesis Class** In a Support Vector Machine (SVM), we define the **hypothesis class** as:

$$F_{w,b}^{\text{SVM}} = \{f : X \to \{\pm 1\}\} \tag{10}$$

where the classifier function is given by:

$$f_{(w,b)}(x) = \text{sign}(w^T x + b). \tag{11}$$

This represents a linear decision boundary where:

- $w$ is the weight vector,
- $b$ is the bias term.

**Hinge Loss-Based Empirical Risk Minimization** To optimize the SVM decision boundary, we use the hinge loss function as the underlying loss function,

$$\ell_{\text{hinge}}(y_n, f_{w,b}(x_n)) = \max(0, 1 - y_n f_{w,b}(x_n)). \tag{12}$$

and we rewrite the ERM framework as

$$\arg \min_{f_{(w,b)} \in F_{(w,b)}^{\text{SVM}}} \sum_{n=1}^{N} \text{hinge}(y_n, f_{w,b}(x_n)) + \frac{1}{C}\|w\|^2. \tag{13}$$

This ensures that:

- The loss is 0 when the classification is correct with a large margin.
- The loss increases linearly when the margin constraint is violated.

## 2 SVM for Regression (SVR)

Our goal in regression is, given a dataset $D = \{(x_1, y_1), \ldots, (x_N, y_N)\}$
where:

- $x_i \in \mathbb{R}^d$ represents feature vectors,
- $y_i \in \mathbb{R}$ represents real-valued targets.

to find a function $f(x)$ that approximates $y_i$ well while minimizing complexity.

### 2.1 SVR Optimization Formulation

The objective of Support Vector Regression (SVR) is to find a linear function:

$$f(x) = w^T x + b \tag{14}$$

that predicts $y_n$ while ensuring that deviations smaller than $\varepsilon$ are ignored. To achieve this, we solve:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \tag{15}$$

subject to:

$$|w^T x_n + b - y_n| \leq \varepsilon. \tag{16}$$

This implies that any error below $\varepsilon$ is ignored, which prevents unnecessary penalization.
SVM can be extended for regression by minimizing prediction errors within a margin of tolerance $\varepsilon$, ignoring small deviations.

### 2.2 Equivalence Reformulation

Using the standard property:

$$|a| < \varepsilon \quad \Rightarrow \quad -\varepsilon < a < \varepsilon, \tag{17}$$

we rewrite the constraints as:

$$w^T x_n + b - y_n \leq \varepsilon, \tag{18}$$

$$w^T x_n + b - y_n \geq -\varepsilon. \tag{19}$$

### 2.3 Soft SVR

To allow some violations, we introduce variables $\xi_n^+$ and $\xi_n^-$:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C_1 \sum_{n=1}^{N} \xi_n^+ + C_2 \sum_{n=1}^{N} \xi_n^-. \tag{20}$$

subject to:

$$w^T x_n + b - y_n \leq \varepsilon + \xi_n^+, \tag{21}$$

$$y_n - (w^T x_n + b) \leq \varepsilon + \xi_n^-. \tag{22}$$

### 2.4 Kernelized SVR Formulation

Replacing $w^T x$ with a kernel function $K(x, x')$, the function we optimize becomes:

$$f(x) = \sum_{i=1}^{N} (\alpha_i^+ - \alpha_i^-) K(x_i, x) + b. \tag{23}$$

**Why is this important?**

- It allows SVR to work in higher-dimensional feature spaces.
- It captures nonlinear relationships in the data.
- Common kernel choices include Polynomial, Gaussian RBF, and Sigmoid.

## 3 Online Learning Framework

Unlike traditional batch learning, where the entire dataset is available at the beginning, **online learning** involves receiving data sequentially and updating the model dynamically.

### 3.1 Definition

In the online learning framework:

- At each time step $t = 1, 2, \ldots, T$:
  1. The learner **receives** an instance $x_t \in X \subseteq \mathbb{R}^d$.
  2. The learner **predicts** a label $\hat{y}_t \in \{\pm 1\}$.
  3. The true label $y_t \in \{\pm 1\}$ is revealed.
  4. The learner **updates** the prediction model based on the new information.

This contrasts with batch learning, where all data $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$ is provided beforehand.

### 3.2 Advantages of Online Learning

- **Memory Efficient**: The model does not need to store the full dataset.
- **Adaptability**: The model can adjust to changing data distributions.
- **Scalability**: Online learning is useful for streaming data or large-scale applications.

## 4 Perceptron Algorithm

### 4.1 Problem Setup: Classification Task

Unlike batch learning, where the dataset is given at once, in the perceptron setup, the labeled dataset is not available beforehand. Instead, data arrives sequentially.

- The goal is to find a **linearly separating hyperplane**:

$$w^{*T}x = 0. \tag{24}$$

- A data point $x_n$ is classified based on:

$$\text{if } y_n = +1, \quad w^{*T}x_n > 0, \tag{25}$$

$$\text{if } y_n = -1, \quad w^{*T}x_n < 0. \tag{26}$$

### 4.2 Algorithm

The perceptron is an online learning algorithm that updates the weight vector $w$ based on misclassifications.

1. **Initialize** $w_1 = 0$ $(t = 1)$.
2. For each time step $t = 1, 2, \ldots, T$:
   (a) Receive an example $x_t \in \mathbb{R}^d$.
   (b) Compute the prediction:
   $$\hat{y}_t = \text{sign}(w_t^T x_t). \tag{27}$$
   (c) Receive $y_t$
   (d) If the prediction is incorrect $y_t \cdot (w_t^T x_t) < 0$ , update the weight vector:
   $$w_{t+1} = w_t + x_t \text{ if } y_{t+1} = +1 \tag{28}$$
   $$w_{t+1} = w_t - x_t \text{ if } y_{t+1} = -1 \tag{29}$$

**Remark 2.** *Each update moves the weight vector toward the correctly classified region. The perceptron algorithm converges if the data is linearly separable.*

## 4.3 Intuitive Explanation of the Perceptron Update

**Case 1: $y_t = +1$, Misclassified** If $w_t^T x_t < 0$ (i.e., the point is misclassified), we update:

$$w_{t+1} = w_t + x_t. \tag{30}$$

and compute the score for $x_t$ :

$$w_{t+1}{}^T x_t = w_t{}^T x_t + x_t{}^T x_t > w_t{}^T x_t \tag{31}$$

**Effect on the decision boundary:**

- This moves the decision boundary in the direction of $x_t$, making it more likely to classify $x_t$ correctly in the future.

**Case 2: $y_t = -1$, Misclassified** If $w_t^T x_t > 0$ (i.e., the point is misclassified), we update:

$$w_{t+1} = w_t - x_t. \tag{32}$$

and compute the score for $x_t$ :

$$w_{t+1}{}^T x_t = w_t{}^T x_t - x_t{}^T x_t < w_t{}^T x_t \tag{33}$$

**Effect on the decision boundary:**

- This moves the decision boundary away from $x_t$, improving classification.

—

## 4.4 Geometric Interpretation of the Perceptron Algorithm

- The perceptron algorithm updates $w$ to better separate the positive and negative classes.
- The decision boundary moves closer to misclassified points with each update.
- The updates reduce the classification error over time.

Example with 2D points:

- Suppose at $t = 1$, we initialize $w_1 = 0$.
- The first point $x_1 = (1, 1)$ with $y_1 = +1$ is misclassified.
- The update is:

$$w_2 = x_1 = (1, 1). \tag{34}$$

- At $t = 2$, $x_2 = (-5, -1)$ with $y_2 = -1$ is correctly classified, so no update occurs.
- At $t = 3$, $x_3 = (-1, 5)$ with $y_3 = -1$ is misclassified:

$$w_4 = w_3 - x_3 = (1, 1) - (-1, 5) = (2, -4). \tag{35}$$

- This process continues until the algorithm converges.

### 4.5 Mistake Bound Analysis for the Perceptron

**Theorem:** The number of mistakes the Perceptron algorithm makes on a dataset $D$ is at most:

$$\frac{1}{\gamma^2}, \tag{36}$$

where $\gamma$ is the margin of the dataset distribution:

$$\gamma = \min_{x \in D} \frac{y w^T x}{\|w\|}. \tag{37}$$

**Key Insights:**

- If $\gamma$ is large, fewer updates (mistakes) occur.
- If $\gamma$ is small, the perceptron may take many steps to converge.

---

### Next Lecture

The next lecture will cover the following topics:

- Perceptron Mistake bounds
- Perceptron w/o perfect
- Linear Separator
- Kernel Perceptron

## References:

Kevin P. Murphy - Probabilistic Machine Learning (2021) , Section 17.5
Lecture Slides from McGill University (Slides 45-54)
Scribed notes 13