Support Vector Machines

Máster Universitario en Ciencia de Datos - Métodos Avanzados en Aprendizaje Automático

Carlos María Alaíz Gudín

Escuela Politécnica Superior
Universidad Autónoma de Madrid

Academic Year 2021/22

Universidad Autónoma
de Madrid

Contents

Support Vector Classifiers

Back to the Linear Case: Multiple Hyperplanes

- Support Vector Machines emerge in the framework of **linearly separable classification problems**.
- There are multiple hyperplanes that separate the data perfectly.
  - Some of them will **generalize better** than others.
  - Which one is the best?

- In the case of **logistic regression**, a probabilistic approach selects the best hyperplane.
- There are other geometrical interpretations that can be used.

Notebook

SVC: Multiple Hyperplanes

Margin of a Linear Model

- The geometrical intuition can be formalized with the concept of **margin**.

Definition (Margin)

The **margin** on a linearly-separable binary classification problem is defined as the distance between the hyperplane and the nearest data point:

$$m = \min_{1 \le i \le N} \left\{ \frac{|\mathbf{w}^\intercal \mathbf{x}_i + b|}{\|\mathbf{w}\|_2} \right\}.$$

- Since the problem is linearly separated and assuming $y_i \in \{-1, 1\}$, the margin can also be written as:

$$m = \min_{1 \le i \le N} \left\{ \frac{y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)}{\|\mathbf{w}\|_2} \right\}.$$

Margin of a Linear Model - Exercise

Exercise

Given the 2-dimensional linear classification model $\{b = 0, w_1 = 1, w_2 = 0\}$, and this dataset:

$x_{i,1}$	$x_{i,2}$	y_i
-1	-1	-1
-2	1	-1
1	0	1

1. Is the model separating both classes?
2. Compute the distances between each point and the hyperplane, using $|\mathbf{w}^\intercal \mathbf{x}_i + b| / \|\mathbf{w}\|_2$.
3. Compute the margin of this model.

Solution

1. Yes, since the predictions are:
   - $\mathbf{w}^\intercal \mathbf{x}_1 = -1 \leq 0.$
   - $\mathbf{w}^\intercal \mathbf{x}_2 = -2 \leq 0$
   - $\mathbf{w}^\intercal \mathbf{x}_3 = 1 \geq 0.$
2. Since $\|\mathbf{w}\|_2 = 1$, the distances are the absolute value of the predictions above:
   - $d_1 = 1.$
   - $d_2 = 2.$
   - $d_3 = 1.$
3. The margin is the minimum of the distances, $m = 1.$

Maximum Margin Hyperplane (I)

- The idea is to find the hyperplane that maximizes m.

- The hyperplane defined by $(\mathbf{w}, b)$ is the same as the one defined by $(c\mathbf{w}, cb)$, for $c > 0$.
- Some kind of normalization should be applied.
- Two different approaches:
  1. Fix the norm of $\mathbf{w}$.
  2. Enforce that the closest points belong to the supporting hyperplanes $\mathbf{w}^\intercal \mathbf{x} + b = \pm 1$. $\Leftarrow$

- With the second normalization:

$$y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1,$$
$$m = \frac{1}{\|\mathbf{w}\|_2}.$$

Hard-Margin Support Vector Classifier

- The Hard-Margin Support Vector Classifier is defined as the solution of the problem:

$$\max_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \frac{1}{\|\mathbf{w}\|_2} \right\} \quad \text{s.t.} \begin{cases} y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1, \\ 1 \leq i \leq N, \end{cases} \equiv \boxed{\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \frac{1}{2} \|\mathbf{w}\|_2^2 \right\} \quad \text{s.t.} \begin{cases} y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1, \\ 1 \leq i \leq N. \end{cases}}$$

- This optimization problem is defined for **binary classification problems**.
- The problem has to be **linearly separable** (otherwise, it is not feasible).
- Since the **margin** of the model is maximized, a **good generalization** can be expected.

Notebook

Hard-Margin SVC

Hard-Margin Support Vector Classifier - Optimization (I)

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \frac{1}{2} \|\mathbf{w}\|_2^2 \right\} \text{ s.t. } y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1, 1 \leq i \leq N.$$

- The objective function is **convex** and **differentiable**.
- The problem has linear constraints.
- It can be solved using **Lagrangian duality**.

Hard-Margin Support Vector Classifier - Optimization (II)

- The Lagrangian becomes:

$$\mathcal{L}(\mathbf{w}, b; \boldsymbol{\alpha}) = \frac{1}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} \alpha_i (1 - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)).$$

- The saddle-point problem is:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \max_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^N \\ \boldsymbol{\alpha}\geq\mathbf{0}}} \{\mathcal{L}(\mathbf{w}, b; \boldsymbol{\alpha})\} \right\} \equiv \max_{\substack{\boldsymbol{\alpha}\in\mathbb{R}^N \\ \boldsymbol{\alpha}\geq\mathbf{0}}} \left\{ \min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \{\mathcal{L}(\mathbf{w}, b; \boldsymbol{\alpha})\} \right\}.$$

Hard-Margin Support Vector Classifier - Optimization (III)

- Solving the inner problem (taking derivatives with respect to $\mathbf{w}$ and b) leads to:

$$\frac{\partial}{\partial b}\mathcal{L}(\mathbf{w}, b; \boldsymbol{\alpha}) = -\sum_{i=1}^{N} \alpha_i y_i = 0 \implies \sum_{i=1}^{N} \alpha_i y_i = 0;$$

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, \mathbf{e}; \boldsymbol{\alpha}) = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = \mathbf{0} \implies \boxed{\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i}.$$

- Substituting back leads to the dual function:

$$\mathcal{D}(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j + \sum_{i=1}^{N} \alpha_i - \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j - b \underbrace{\sum_{i=1}^{N} \alpha_i y_i}_{0}$$

$$= -\frac{1}{2}\boldsymbol{\alpha}^\intercal \tilde{\mathbf{X}} \tilde{\mathbf{X}}^\intercal \boldsymbol{\alpha} + \boldsymbol{\alpha}^\intercal \mathbf{1},$$

where $\tilde{\mathbf{X}}$ is the labelled data matrix, in which the i-th row corresponds to $y_i \mathbf{x}_i^\intercal$.

Hard-Margin Support Vector Classifier - Optimization (IV)

- The resultant **dual problem** is hence:

$$\max_{\boldsymbol{\alpha} \in \mathbb{R}^N} \left\{ -\frac{1}{2} \boldsymbol{\alpha}^\intercal \tilde{\mathbf{X}} \tilde{\mathbf{X}}^\intercal \boldsymbol{\alpha} + \boldsymbol{\alpha}^\intercal \mathbf{1} \right\} \quad \text{s.t.} \begin{cases} \boldsymbol{\alpha}^\intercal \mathbf{y} = 0, \\ \boldsymbol{\alpha} \geq \mathbf{0}, \end{cases} \equiv \boxed{\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \left\{ \frac{1}{2} \boldsymbol{\alpha}^\intercal \tilde{\mathbf{X}} \tilde{\mathbf{X}}^\intercal \boldsymbol{\alpha} - \boldsymbol{\alpha}^\intercal \mathbf{1} \right\} \quad \text{s.t.} \begin{cases} \boldsymbol{\alpha}^\intercal \mathbf{y} = 0, \\ \boldsymbol{\alpha} \geq \mathbf{0}. \end{cases}}$$

- It is a **constrained quadratic problem**.
- There are different *ad hoc* algorithms for solving it.
- The data only appear in form of **inner products**.
- As a consequence of the Lagrangian duality, $\alpha_i(1 - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)) = 0$, for $i = 1, \dots, N$.
  - If $\alpha_i > 0$, $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) = 1$ and this point over the supporting hyperplane is a **support vector**.
  - If $\alpha_i = 0$, $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1$ and the point has no impact on the model.
  - The model is **sparse** in terms of the training samples.

Notebook

Hard-Margin SVC: Optimization

Soft-Margin Support Vector Classifiers: Introduction

- Most problems are not linearly separable.
- Even if they are (e.g. because d is large), maybe it is not convenient to perfectly classify the data.
  - This can lead to **over-fitting**.

- Soft-margin Support Vector Classifiers allow for training errors introducing **slack variables**.
- These variables quantify the margin violation of each pattern.

- The constraints are modified to $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1 - \xi_i$, with $\xi_i \geq 0$ the distance to the corresponding supporting hyperplane.
- The slack variables are penalized to be as small as possible.

Soft-Margin Support Vector Classifier (I)

- The Soft-Margin Support Vector Classifier is defined as the solution of the problem:

$$\boxed{\begin{aligned} &\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi}\in\mathbb{R}^N}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i \right\} \\ &\text{s.t.} \left\{ \begin{array}{l} y_i(\mathbf{w}^\intercal\mathbf{x}_i + b) \ge 1 - \xi_i, \\ \xi_i \ge 0, \\ 1 \le i \le N. \end{array} \right. \end{aligned}}$$

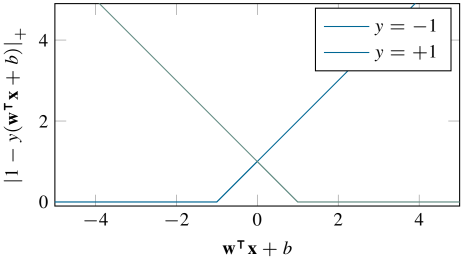
- This problem is defined for **binary classification problems**.
- The problem does **not** need to be **linearly separable**.
- The hyper-parameter C controls the balance between accuracy and complexity.

Soft-Margin Support Vector Classifier (II)

- Equivalently, the problem can be written without constraints as follows:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N} \left|1 - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)\right|_+ \right\},$$

where $|x|_+$ denotes the positive part. The resultant measure is known as the **hinge loss function**.



Hinge Loss - Exercise

Exercise

Given the 2-dimensional linear classification model $\{b = 0, w_1 = 0.25, w_2 = -0.5\}$, and this dataset:

$x_{i,1}$	$x_{i,2}$	y_i
-1	-1	-1
-2	1	-1
1	0	1

1. Is the model separating both classes?
2. Compute the hinge loss error for each pattern, using $|1 - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)|_+$.
3. Is the error 0 for any pattern? Is it 0 for all the correctly classified patterns?

Solution

1. No, the first sample is wrongly classified. The predictions are:
   - $\mathbf{w}^\intercal \mathbf{x}_1 = 0.25 \not\leq 0$.
   - $\mathbf{w}^\intercal \mathbf{x}_2 = -1 \leq 0$.
   - $\mathbf{w}^\intercal \mathbf{x}_3 = 0.25 \geq 0$.
2. The corresponding errors are:
   - $e_1 = |1 - 0.25 y_1|_+ = 1.25$.
   - $e_2 = |1 + y_2|_+ = 0$.
   - $e_3 = |1 - 0.25 y_3|_+ = 0.75$.
3. For $\mathbf{x}_1$, wrongly classified, it is larger than 1. For $\mathbf{x}_2$ is 0 since it is respecting the margin. For $\mathbf{x}_3$, not respecting the margin but correctly classified, it is between 0 and 1.

Notebook

Soft-Margin SVC

Soft-Margin Support Vector Classifier - Optimization (I)

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi}\in\mathbb{R}^N}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i \right\} \text{ s.t. } y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0, 1 \leq i \leq N.$$

- The objective function is **convex** and **differentiable**.
- The problem has linear constraints.
- It can be solved using **Lagrangian duality**.

Soft-Margin Support Vector Classifier - Optimization (II)

- The Lagrangian becomes:

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i(1 - \xi_i - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)) + \sum_{i=1}^{N}\beta_i(-\xi_i).$$

- The saddle-point problem is:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi}\in\mathbb{R}^N}} \left\{ \max_{\substack{\boldsymbol{\alpha},\boldsymbol{\beta}\in\mathbb{R}^N \\ \boldsymbol{\alpha},\boldsymbol{\beta}\geq\mathbf{0}}} \{\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta})\} \right\} \equiv \max_{\substack{\boldsymbol{\alpha},\boldsymbol{\beta}\in\mathbb{R}^N \\ \boldsymbol{\alpha},\boldsymbol{\beta}\geq\mathbf{0}}} \left\{ \min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi}\in\mathbb{R}^N}} \{\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta})\} \right\}.$$

Soft-Margin Support Vector Classifier - Optimization (III)

- Solving the inner problem (taking derivatives with respect to **w** and b) leads to:

$$\frac{\partial}{\partial b}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = -\sum_{i=1}^{N} \alpha_i y_i = 0 \implies \sum_{i=1}^{N} \alpha_i y_i = 0;$$

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = \mathbf{w} - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i = \mathbf{0} \implies \boxed{\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i};$$

$$\frac{\partial}{\partial \xi_i}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}; \boldsymbol{\alpha}, \boldsymbol{\beta}) = C - \alpha_i - \beta_i = 0 \implies 0 \leq \alpha_i \leq C.$$

Soft-Margin Support Vector Classifier - Optimization (IV)

- Substituting back leads to the dual function:

$$
\begin{aligned}
\mathcal{D}(\boldsymbol{\alpha}) &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j + C\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i\xi_i \\
&\quad - \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j - \sum_{i=1}^{N}\alpha_i y_i b - \sum_{i=1}^{N}\beta_i\xi_i \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j + \sum_{i=1}^{N}\xi_i\underbrace{(C-\alpha_i-\beta_i)}_{0} + \sum_{i=1}^{N}\alpha_i - b\underbrace{\sum_{i=1}^{N}\alpha_i y_i}_{0} \\
&= -\frac{1}{2}\boldsymbol{\alpha}^\intercal \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\intercal \boldsymbol{\alpha} + \boldsymbol{\alpha}^\intercal \mathbf{1}.
\end{aligned}
$$

Soft-Margin Support Vector Classifier - Optimization (V)

- The resultant **dual problem** is:

$$\boxed{\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \left\{ \frac{1}{2}\boldsymbol{\alpha}^\intercal \tilde{\mathbf{X}}\tilde{\mathbf{X}}^\intercal \boldsymbol{\alpha} - \boldsymbol{\alpha}^\intercal \mathbf{1} \right\} \\ &\text{s.t.} \left\{ \begin{array}{l} \boldsymbol{\alpha}^\intercal \mathbf{y} = 0, \\ \mathbf{0} \leq \boldsymbol{\alpha} \leq C. \end{array} \right. \end{aligned}}$$

- It is again a **constrained quadratic problem**.
- The dual coefficients have an additional upper bound C.
  - If C is larger than a certain value the hard-margin SVC is recovered.
- There are different *ad hoc* algorithms for solving it.
- The data only appear in form of **inner products**.
- As a consequence of the Lagrangian duality, $\alpha_i(1 - \xi_i - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)) = 0$, for $i = 1, \dots, N$.
  - If $\alpha_i = 0$, $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1$ and the point has no impact on the model.
  - If $0 < \alpha_i < C$, $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) = 1$.
  - If $\alpha_i = C$, $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) = 1 - \xi_i \leq 1$.
  - The model is **sparse** in terms of the training samples.

Support Vector Regression

Introduction

- The SVC models have certain desirable properties:
  - They can be trained using a dual problem.
  - They are sparse in terms of the training samples.
  - They control naturally the complexity.
- These properties motivate their extension to a regression setting.

- What is the origin of these good properties?
  1. Maximizing the margin (minimizing the complexity of the model).
  2. Having a sparse-inducing error term.

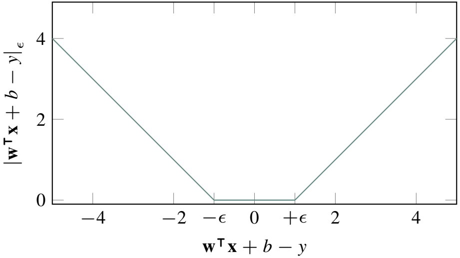
- Can this be extended to a regression framework?
  1. It is partially done in (Kernel) Ridge Regression.
  2. A new **loss function** is needed.

The ϵ-Insensitive Loss

- The **ϵ-insensitive loss** of a linear model $\{\mathbf{w}, b\}$ over a pattern $(\mathbf{x}, y)$ is defined as:

$$|\mathbf{w}^\intercal \mathbf{x} + b - y|_\epsilon = \max\{0, |\mathbf{w}^\intercal \mathbf{x} + b - y| - \epsilon\}.$$

- Errors smaller than ϵ are simply ignored.
- Errors larger than ϵ are penalized linearly.
- It avoids over-fitting by ignoring small errors, but the hyper-parameter ϵ has to be tuned.



The ϵ-Insensitive Loss - Exercise

Exercise

Given the 2-dimensional linear regression model $\{b = 0, w_1 = 1, w_2 = 1\}$, and this dataset:

$x_{i,1}$	$x_{i,2}$	y_i
-1	-1	-1.9
-2	1	-1
1	0	2

1. Compute the prediction for each pattern.
2. Compute the ϵ-insensitive loss for each pattern, using $\max\{0, |\mathbf{w}^\intercal \mathbf{x} + b - y| - \epsilon\}$, with $\epsilon = 0.25$.

Solution

1. The predictions are:
   - $\mathbf{w}^\intercal \mathbf{x}_1 = -2.$
   - $\mathbf{w}^\intercal \mathbf{x}_2 = -1$
   - $\mathbf{w}^\intercal \mathbf{x}_3 = 1.$
2. The corresponding errors are:
   - $e_1 = \max\{0, |-2 - y_1| - 0.25\} = 0.$
   - $e_2 = \max\{0, |-1 - y_2| - 0.25\} = 0.$
   - $e_3 = \max\{0, |1 - y_3| - 0.25\} = 0.75.$

Support Vector Regression

- The Support Vector Regression is defined as the solution of the problem:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R}}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N} \left|\mathbf{w}^\intercal \mathbf{x}_i + b - y_i\right|_\epsilon \right\} \equiv \boxed{\begin{array}{l} \displaystyle\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi},\boldsymbol{\xi}^*\in\mathbb{R}^N}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N} (\xi_i + \xi_i^*) \right\} \\ \text{s.t.} \left\{ \begin{array}{l} \mathbf{w}^\intercal \mathbf{x}_i + b - y_i \le \epsilon + \xi_i, \\ y_i - \mathbf{w}^\intercal \mathbf{x}_i - b \le \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* \ge 0, \\ 1 \le i \le N. \end{array} \right. \end{array}}$$

- The hyper-parameter C controls the balance between accuracy and complexity.

Notebook

SVR

Support Vector Regression - Optimization (I)

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ b\in\mathbb{R} \\ \boldsymbol{\xi},\boldsymbol{\xi}^*\in\mathbb{R}^N}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \right\}$$

$$\text{s.t.} \begin{cases} \mathbf{w}^\intercal \mathbf{x}_i + b - y_i \leq \epsilon + \xi_i, \\ y_i - \mathbf{w}^\intercal \mathbf{x}_i - b \leq \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* \geq 0, \\ 1 \leq i \leq N. \end{cases}$$

- The objective function is **convex** and **differentiable**.
- The problem has linear constraints.
- It can be solved using **Lagrangian duality**.

Support Vector Regression - Optimization (II)

- The Lagrangian becomes:

$$\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right) = \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) + \sum_{i=1}^{N}\alpha_i(\mathbf{w}^\intercal\mathbf{x}_i + b - y_i - \epsilon - \xi_i)$$
$$+ \sum_{i=1}^{N}\alpha_i^*(y_i - \mathbf{w}^\intercal\mathbf{x}_i - b - \epsilon - \xi_i^*) + \sum_{i=1}^{N}\beta_i(-\xi_i) + \sum_{i=1}^{N}\beta_i^*(-\xi_i^*).$$

- The saddle-point problem is:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d\\ b\in\mathbb{R}\\ \boldsymbol{\xi}^{(*)}\in\mathbb{R}^N}} \left\{ \max_{\substack{\boldsymbol{\alpha}^{(*)}\in\mathbb{R}^N\\ \boldsymbol{\beta}^{(*)}\in\mathbb{R}^N\\ \boldsymbol{\alpha}^{(*)}\geq\mathbf{0}\\ \boldsymbol{\beta}^{(*)}\geq\mathbf{0}}} \left\{\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right)\right\} \right\} \equiv \max_{\substack{\boldsymbol{\alpha}^{(*)}\in\mathbb{R}^N\\ \boldsymbol{\beta}^{(*)}\in\mathbb{R}^N\\ \boldsymbol{\alpha}^{(*)}\geq\mathbf{0}\\ \boldsymbol{\beta}^{(*)}\geq\mathbf{0}}} \left\{ \min_{\substack{\mathbf{w}\in\mathbb{R}^d\\ b\in\mathbb{R}\\ \boldsymbol{\xi}^{(*)}\in\mathbb{R}^N}} \left\{\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right)\right\} \right\}.$$

Support Vector Regression - Optimization (III)

- Solving the inner problem (taking derivatives with respect to **w** and b) leads to:

$$\frac{\partial}{\partial b}\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right) = \sum_{i=1}^{N}\alpha_i - \sum_{i=1}^{N}\alpha_i^* = 0 \implies \sum_{i=1}^{N}(\alpha_i^* - \alpha_i) = 0;$$

$$\nabla_{\mathbf{w}}\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right) = \mathbf{w} + \sum_{i=1}^{N}\alpha_i\mathbf{x}_i - \sum_{i=1}^{N}\alpha_i^*\mathbf{x}_i = \mathbf{0} \implies \boxed{\mathbf{w} = \sum_{i=1}^{N}(\alpha_i^* - \alpha_i)\mathbf{x}_i};$$

$$\frac{\partial}{\partial \xi_i^{(*)}}\mathcal{L}\left(\mathbf{w}, b, \boldsymbol{\xi}^{(*)}; \boldsymbol{\alpha}^{(*)}, \boldsymbol{\beta}^{(*)}\right) = C - \alpha_i^{(*)} - \beta_i^{(*)} = 0 \implies 0 \le \alpha_i^{(*)} \le C.$$

Support Vector Regression - Optimization (IV)

- Substituting back leads to the dual function:

$$
\begin{aligned}
\mathcal{D}\left(\boldsymbol{\alpha}^{(*)}\right) &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^* - \alpha_i)\left(\alpha_j^* - \alpha_j\right)\mathbf{x}_i^\intercal\mathbf{x}_j + C\sum_{i=1}^{N}(\xi_i + \xi_i^*) \\
&+ \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\left(\alpha_j^* - \alpha_j\right)\mathbf{x}_i^\intercal\mathbf{x}_j + \sum_{i=1}^{N}\alpha_i b - \sum_{i=1}^{N}\alpha_i y_i - \sum_{i=1}^{N}\alpha_i\epsilon - \sum_{i=1}^{N}\alpha_i\xi_i + \sum_{i=1}^{N}\alpha_i^* y_i \\
&- \sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i^*\left(\alpha_j^* - \alpha_j\right)\mathbf{x}_i^\intercal\mathbf{x}_j - \sum_{i=1}^{N}\alpha_i^* b - \sum_{i=1}^{N}\alpha_i^*\epsilon - \sum_{i=1}^{N}\alpha_i^*\xi_i^* - \sum_{i=1}^{N}\beta_i\xi_i - \sum_{i=1}^{N}\beta_i^*\xi_i^* \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^* - \alpha_i)\left(\alpha_j^* - \alpha_j\right)\mathbf{x}_i^\intercal\mathbf{x}_j - \sum_{i=1}^{N}(\alpha_i - \alpha_i^*)y_i - \epsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) \\
&+ \underbrace{b\sum_{i=1}^{N}(\alpha_i - \alpha_i^*)}_{0} + \sum_{i=1}^{N}\xi_i\underbrace{(C - \alpha_i - \beta_i)}_{0} + \sum_{i=1}^{N}\xi_i^*\underbrace{(C - \alpha_i^* - \beta_i^*)}_{0}
\end{aligned}
$$

Support Vector Regression - Optimization (V)

- In matrix notation, the dual function becomes:

$$\mathcal{D}\left(\boldsymbol{\alpha}^{(*)}\right) = -\frac{1}{2}(\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{X}\mathbf{X}^\intercal(\boldsymbol{\alpha}^* - \boldsymbol{\alpha}) - \epsilon(\boldsymbol{\alpha}^* + \boldsymbol{\alpha})^\intercal \mathbf{1} + (\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{y}.$$

- The resultant **dual problem** is:

$$\boxed{\begin{array}{l} \min\limits_{\boldsymbol{\alpha},\boldsymbol{\alpha}^* \in \mathbb{R}^N} \left\{ \frac{1}{2}(\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{X}\mathbf{X}^\intercal(\boldsymbol{\alpha}^* - \boldsymbol{\alpha}) + \epsilon(\boldsymbol{\alpha}^* + \boldsymbol{\alpha})^\intercal \mathbf{1} - (\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{y} \right\} \\ \text{s.t.} \begin{cases} (\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{1} = 0, \\ \mathbf{0} \le \boldsymbol{\alpha}, \boldsymbol{\alpha}^* \le C. \end{cases} \end{array}}$$

- It is again a **constrained quadratic problem**.
- There are different *ad hoc* algorithms for solving it.
- The data only appear in form of **inner products**.
- As a consequence of the Lagrangian duality:
  - If $\alpha_i - \alpha_i^* = 0$, the point lies inside the ϵ-insensitive tube and it has no impact on the model.
  - Otherwise, the point lies outside the tube (or over the border) and it is a support vector.

Notebook

SVR: Optimization

One-Class Support Vector Machine

Novelty Detection

Outlier Detection

- In many cases, the training data contains **outliers** (data points generated by a different distribution).
- In practice, **outlier detection estimators** try to find the regions where the data is more concentrated, ignoring the data points far away from the mean.

Novelty Detection

- The training data in this case has no outliers.
- The goal is instead to detect **anomalies** in new observations.

One-Class SVM

- **One-Class SVM** is an unsupervised learning algorithm used for novelty detection.
  - Given a set of samples, it will define a soft boundary around the regions with high density of points.
  - It will provide good results also in outlier problems.

One-Class Support Vector Machine

- The One-Class Support Vector Machine is defined as the solution of the problem:

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ \rho\in\mathbb{R}}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 - \rho + \frac{1}{\nu N}\sum_{i=1}^{N} |\rho - \mathbf{w}^\intercal \mathbf{x}_i|_+ \right\}.$$

- Similar idea than a classical SVC (with ν-SVM formulation), but separating "data" from "no data".
- It is defined in terms of a hinge loss function.
- The hyper-parameter $\nu \in (0, 1]$ controls the anomaly detection sensitivity.
  - It is an upper-bound of the fraction of errors allowed.
  - It is a lower-bound of the number of support vectors.

Notebook

OC-SVM

One-Class Support Vector Machine - Optimization (I)

$$\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ \rho\in\mathbb{R}}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 - \rho + \frac{1}{\nu N}\sum_{i=1}^{N} |\rho - \mathbf{w}^\intercal \mathbf{x}_i|_+ \right\} \equiv \boxed{\begin{array}{l} \displaystyle\min_{\substack{\mathbf{w}\in\mathbb{R}^d \\ \rho\in\mathbb{R} \\ \boldsymbol{\xi}\in\mathbb{R}^N}} \left\{ \frac{1}{2}\|\mathbf{w}\|_2^2 - \rho + \frac{1}{\nu N}\sum_{i=1}^{N} \xi_i \right\} \\ \text{s.t.} \left\{ \begin{array}{l} \mathbf{w}^\intercal \mathbf{x}_i \geq \rho - \xi_i, \\ \xi_i \geq 0, \\ 1 \leq i \leq N. \end{array} \right. \end{array}}$$

- The objective function is **convex** and **differentiable**.
- The problem has linear constraints.
- It can be solved using **Lagrangian duality**.

One-Class Support Vector Machine - Optimization (II)

- After the corresponding derivations, the resultant **dual problem** is:

$$\boxed{\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \left\{\frac{1}{2}\boldsymbol{\alpha}^\intercal \mathbf{X}\mathbf{X}^\intercal \boldsymbol{\alpha}\right\} \\ &\text{s.t.} \left\{\begin{array}{l} \boldsymbol{\alpha}^\intercal \mathbf{1} = 1, \\ \mathbf{0} \le \boldsymbol{\alpha} \le \frac{1}{\nu N}. \end{array}\right. \end{aligned}}$$

- The primal hyperplane is recovered as:

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i \mathbf{x}_i .$$

- It is again a **constrained quadratic problem**.
- There are different *ad hoc* algorithms for solving it.
- The data only appear in form of **inner products**.
- As a consequence of the Lagrangian duality:
  - If $\alpha_i = 0$, the point lies on the correct side and it has no impact on the model.
  - Otherwise, the point is a support vector.

Notebook

OC-SVM: Optimization

The Kernel Trick

The Kernel Trick

- Linear models are not enough in many problems.

- In the optimization problems for training SVMs, the data only appear as inner products.
- Moreover, the prediction for a new data point, $\mathbf{w}^\intercal \mathbf{x} + b$, can also be computed using only inner products.

- The SVMs can be extended to a non-linear framework using a mapping $\phi : \mathbb{R}^d \to \mathbb{R}^D$.
- Thanks to the **kernel trick**, instead of defining explicitly ϕ, a kernel function $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is used.
  - The selection of the kernel, and its hyper-parameters, is crucial.
  - One of the most common choices is the **RBF** kernel $\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \|\mathbf{x} - \mathbf{x}'\|_2^2\right)$.

- The samples $\mathbf{x}_i$ are substituted by $\phi(\mathbf{x}_i)$.
- The matrix $\mathbf{X}\mathbf{X}^\intercal$ is substituted by the kernel matrix $\mathbf{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^\intercal$.

Non-Linear SVMs (I)

SVC: Training

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \left\{ \frac{1}{2}\boldsymbol{\alpha}^\intercal \tilde{\mathbf{K}} \boldsymbol{\alpha} - \boldsymbol{\alpha}^\intercal \mathbf{1} \right\} \text{ s.t. } \left\{ \begin{array}{r} \boldsymbol{\alpha}^\intercal \mathbf{y} = 0, \\ \mathbf{0} \le \boldsymbol{\alpha} \le C. \end{array} \right.$$

SVR: Training

$$\min_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*\in\mathbb{R}^N} \left\{ \frac{1}{2}(\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{K} (\boldsymbol{\alpha}^* - \boldsymbol{\alpha}) + \epsilon(\boldsymbol{\alpha}^* + \boldsymbol{\alpha})^\intercal \mathbf{1} - (\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{y} \right\} \text{ s.t. } \left\{ \begin{array}{r} (\boldsymbol{\alpha}^* - \boldsymbol{\alpha})^\intercal \mathbf{1} = 0, \\ \mathbf{0} \le \boldsymbol{\alpha}, \boldsymbol{\alpha}^* \le C. \end{array} \right.$$

OC-SVM: Training

$$\min_{\boldsymbol{\alpha}\in\mathbb{R}^N} \left\{ \frac{1}{2}\boldsymbol{\alpha}^\intercal \mathbf{K} \boldsymbol{\alpha} \right\} \text{ s.t. } \left\{ \begin{array}{r} \boldsymbol{\alpha}^\intercal \mathbf{1} = 1, \\ \mathbf{0} \le \boldsymbol{\alpha} \le \frac{1}{\nu N}. \end{array} \right.$$

Non-Linear SVMs (II)

SVC: Prediction

$$f(\mathbf{x}) = \sum_{i=1}^{N} y_i \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b.$$

SVR: Prediction

$$f(\mathbf{x}) = \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) \mathcal{K}(\mathbf{x}_i, \mathbf{x}) + b.$$

OC-SVM: Prediction

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \mathcal{K}(\mathbf{x}_i, \mathbf{x}) - \rho.$$

Notebook

Non-Linear SVC
Non-Linear SVR
Non-Linear OC-SVM

Support Vector Machines

Carlos María Alaíz Gudín