UNIVERSIDADE CATÓLICA PORTUGUESA

BRAGA

Machine Learning

Session 16 - T

Support Vector Machines – Part 3

Degree in Applied Data Science

2024/2025

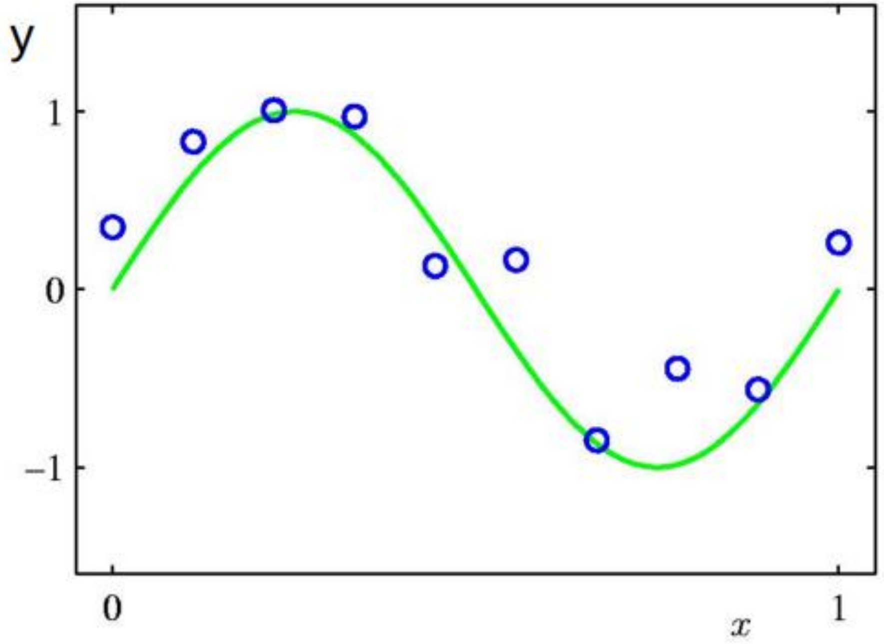
SVMs - Regression

- Suppose we are given a training set of N observations

$$((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)) \text{ with } \mathbf{x}_i \in \mathbb{R}^d, y_i \in \mathbb{R}$$

- The **regression** problem is to estimate f(x) from the data such that

$$y_i = f(\mathbf{x}_i)$$



SVMs - Regression

UNIVERSIDADE CATÓLICA PORTUGUESA
BRAGA

- As for classification, learning a regressor can be formulated as an optimization problem:

- Minimize $$\sum_{i=1}^{N} \underbrace{l\left(f(\mathbf{x}_i), y_i\right)}_{\text{loss function}} + \underbrace{\lambda R\left(f\right)}_{\text{regularization}}$$

- There is a choice of both **loss function** and **regularization**
  - e.g. squared loss, "Hinge-like" loss
  - ridge, lasso regularization

SVMs - Choice of Regression Function

- Function for regression y(x, w) is a non-linear function of x, but linear in w:

$$f(\mathbf{x}, \mathbf{w}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \ldots + w_M\phi_M(\mathbf{x}) = \mathbf{w}^\top \Phi(\mathbf{x})$$

- For example, for $x \in \mathbb{R}$, polynimial regression with $\phi_j(x) = x^j$:

$$f(x, \mathbf{w}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \ldots + w_M\phi_M(\mathbf{x}) = \sum_{j=0}^{M} w_j x^j$$

e.g. for $M = 3$,

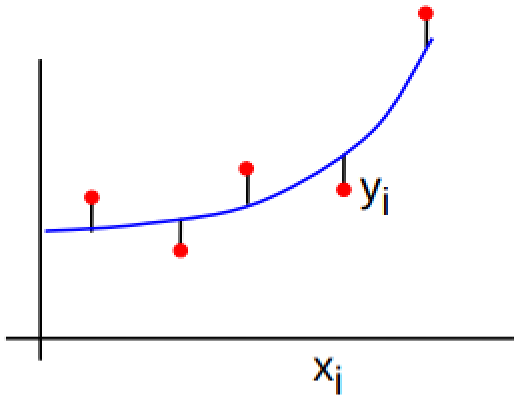
$$f(x, \mathbf{w}) = (w_0, w_1, w_2, w_3) \begin{pmatrix} 1 \\ x \\ x^2 \\ x^3 \end{pmatrix} = \mathbf{w}^\top \Phi(x)$$

$$\Phi : x \to \Phi(x) \quad \mathbb{R}^1 \to \mathbb{R}^4$$

SVMs - Least Squares Ridge Regression

- Cost function – squared loss:

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N} \underbrace{\{f(x_i,\mathbf{w}) - \overset{\text{target value}}{y_i}\}^2}_{\text{loss function}} + \underbrace{\frac{\lambda}{2}\|\mathbf{w}\|^2}_{\text{regularization}}$$



- Regression function for x:

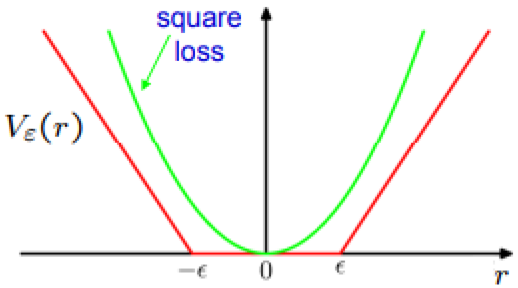
$$f(\mathbf{x},\mathbf{w}) = w_0 + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \ldots + w_M\phi_M(\mathbf{x}) = \mathbf{w}^\top\Phi(\mathbf{x})$$

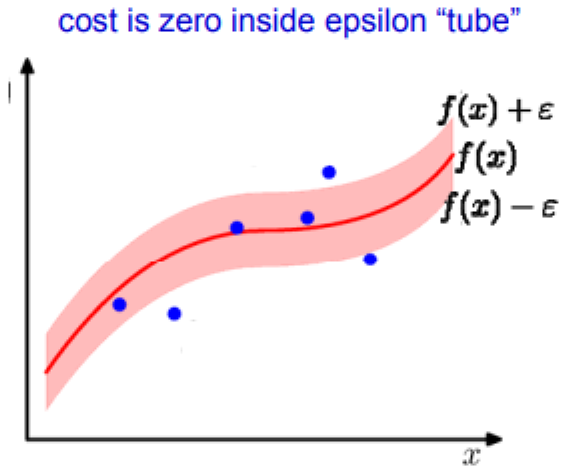
SVMs - Loss Function for Regression

- To allow for misclassification in SVM regression, we can use the **ε–insensitive loss**:

$$J_\epsilon = \sum_{i=1}^{m} J_\epsilon(\mathbf{x_i}), \text{ where}$$

$$J_\epsilon(\mathbf{x_i}) = \begin{cases} 0 & \text{if } |y_i - (\mathbf{w} \cdot \mathbf{x_i} + w_0)| \leq \epsilon \\ |y_i - (\mathbf{w} \cdot \mathbf{x_i} + w_0)| - \epsilon & \text{otherwise} \end{cases}$$





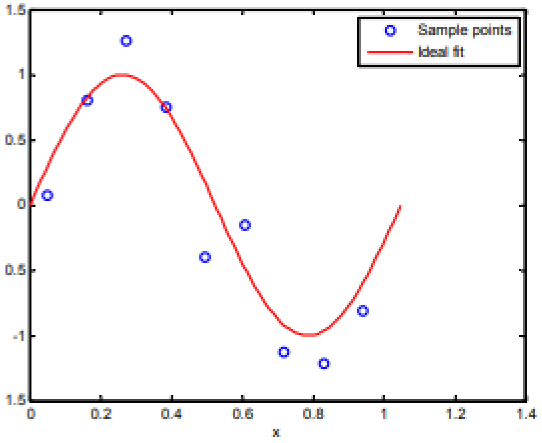
SVMs - The Optimization Problem

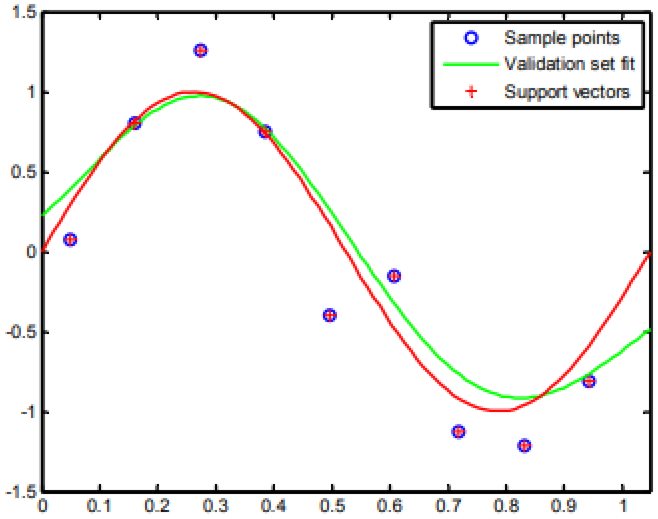
$$
\begin{array}{rl}
\text{min} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i(\xi_i^+ + \xi_i^-) \\
\text{w.r.t.} & \mathbf{w}, w_0, \xi_i^+, \xi_i^- \\
\text{s.t.} & y_i - (\mathbf{w}\cdot\mathbf{x}_i + w_0) \leq \epsilon + \xi_i^+ \\
& y_i - (\mathbf{w}\cdot\mathbf{x}_i + w_0) \geq -\epsilon - \xi_i^- \\
& \xi_i^+, \xi_i^- \geq 0
\end{array}
$$

- As before, Kernels can be used to get non-linear functions

SVMs - Effect of ε

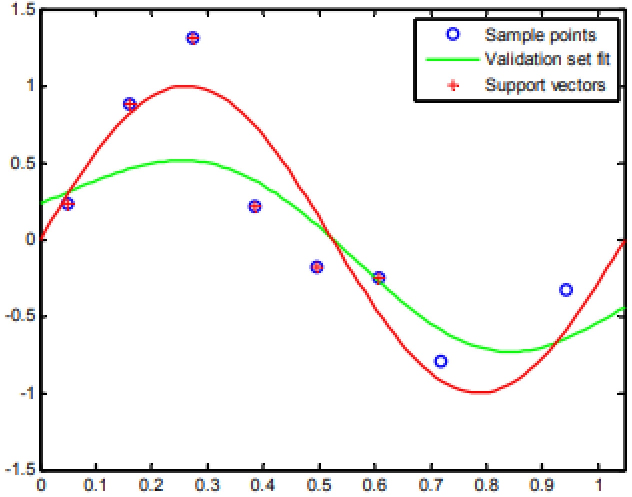
- As **ε increases**, the function is allowed to **move away from the data points**, the **number of support vectors decreases** and the **fit gets worse**.





epsilon = 0.01

epsilon = 0.8



Resources

- https://www.youtube.com/watch?v=kPw1IGUAoY8
- https://www.researchgate.net/publication/228537532_Support_Vector_Regression