Lecture 14 EE 559 Feb 28, 2024

Machine Learning I: Supervised Methods

B. Keith Jenkins

Announcements

- Slido event code: 1761012
- Homework 5 is due Friday.
- Midterm exam is 3/6/2024, 4:00 - 5:50 PM.
  - On paper
  - Closed book and no internet
  - One formula sheet allowed (2 sided)
  - Basic scientific calculator allowed (stand-alone HW based device)

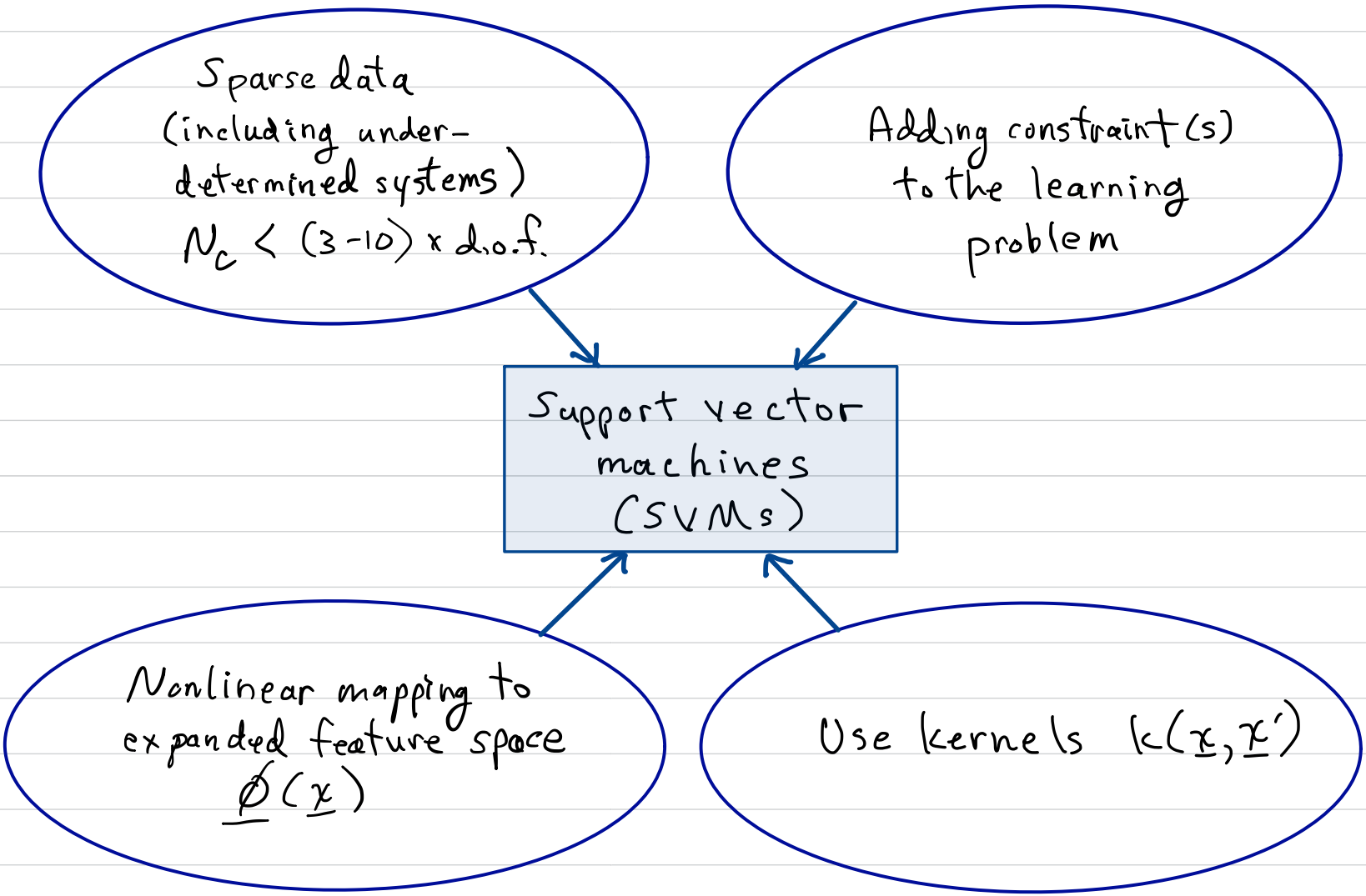
Reading

- Bishop 7.1.4 - Support Vector Regression

Today's lecture

- Support Vector Machines
  - Maximum margin classifier
  - Primal and dual forms
  - Linearly separable data
  - Not linearly separable data
  - Slack variables and hinge loss
- Support vector regression (part 1)
  - Epsilon-insensitive loss

Support Vector Machines (SVMs) [Bishop 7.1]



Concept

→ $C = 2$ classes. → Non-augmented notation, space

(i) Nonlinear mapping: $\underline{x} \xrightarrow{\text{N.L. mapping}} \underline{u} = \underline{\phi}(\underline{x})$

→ Assume data is linearly separable in $\underline{u}$-space (nonseparable case — later)

(ii) Optimized version of using a margin

$$g(\underline{x}) = \underline{w}^T \underline{x} + w_0 \geq b > 0 \quad \text{for } \underline{x} \in S_1$$

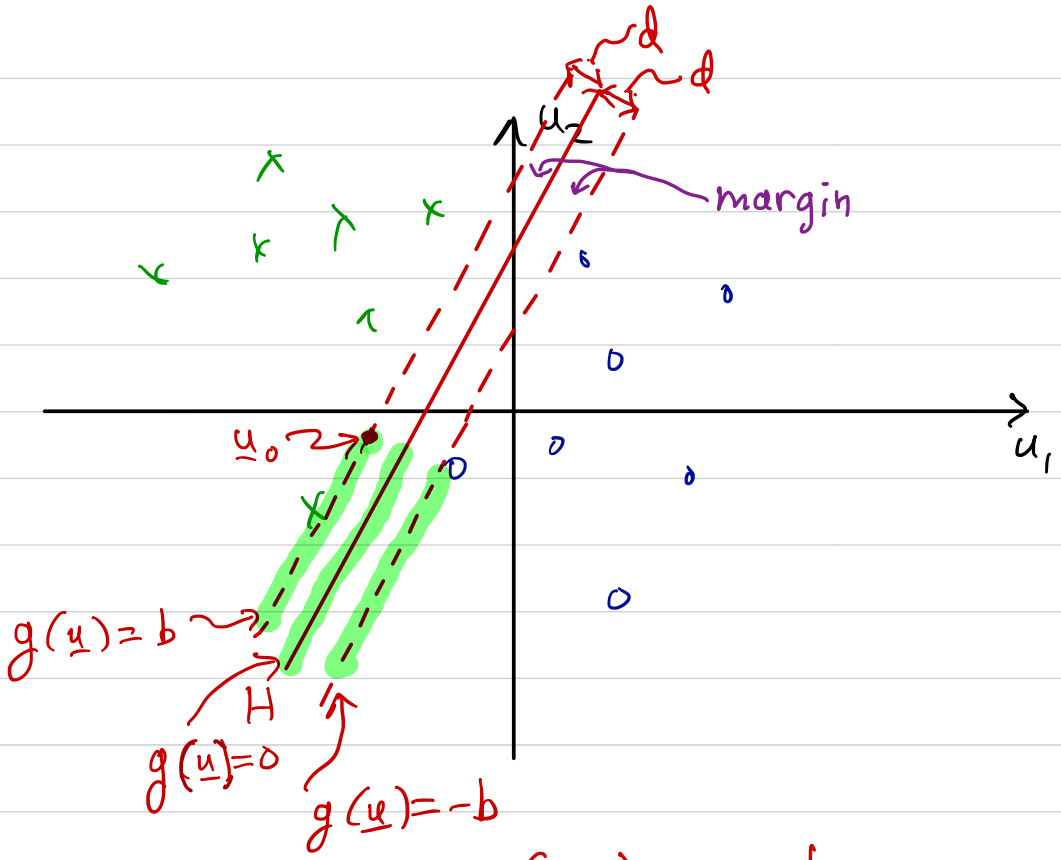
$$g'(\underline{u}) = \underline{w}'^T \underline{u} + w_0' \geq b' > 0 \quad \text{for } \underline{u} \in S_1$$

$\Rightarrow$ $\underline{\phi}(\underline{x})$ or $k(\underline{x}, \underline{x}')$

→ We will drop primes from here on, and work in $\underline{u}$ space.

$$\Rightarrow g(\underline{u}_n) = \begin{cases} \underline{w}^T \underline{u}_n + w_0 \geq b > 0 & \forall\, \underline{u}_n \in S_1 \\ \underline{w}^T \underline{u}_n + w_0 \leq -b < 0 & \forall\, \underline{u}_n \in S_2 \end{cases}$$

↑ for correct classification of all $\underline{u}_n$.



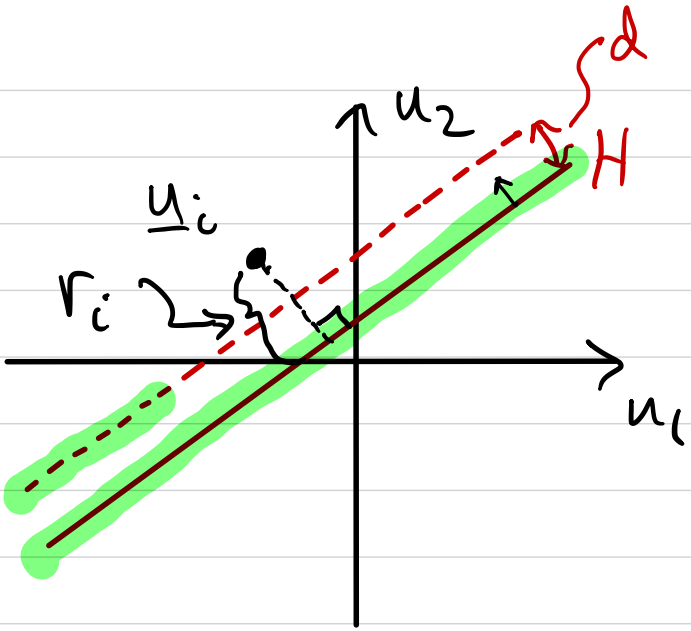
× S_1 (price increase)
o S_2 (price decrease)

$$d = d_S(H, \underline{u}_0) = \frac{g(\underline{u}_0)}{\|\underline{w}\|} = \frac{b}{\|\underline{w}\|}$$

$$g(\underline{u}) = \underline{w}^T \underline{u} + w_0$$

(S) augm. refl. $\Rightarrow g(\underline{u}) = \underline{w}^{(+)T} z_i \underline{u}_i^{(+)} > 0 \Rightarrow \underline{u}_i$ is correctly classified.

$\therefore z_i(\underline{w}^T \underline{u}_i + w_0) > 0 \quad \forall i \Rightarrow$ all data pts correctly classified.



Distance:

$$d_S(H, \underline{u}_i) = r_i = \frac{z_i(\underline{w}^T \underline{u}_i + w_0)}{\|\underline{w}\|} > 0$$

$$\Rightarrow \boxed{\text{Want: } r_i \geq d > 0}$$

$\therefore$ Require:

$$(1) \begin{cases} \dfrac{z_i(\underline{w}^T \underline{u}_i + w_0)}{\|\underline{w}\|} \geq d > 0 \quad \forall i \\ \text{with maximum possible } d. \end{cases}$$

SVM Learning

(Still assuming data is linearly separable in $\underline{u}$ space.)

Note: $\underline{w} \to a\underline{w}$, $w_0 \to aw_0$, ANY $a > 0$

$\Rightarrow$ LHS of (1) doesn't change.

RHS of (1): $d = \dfrac{b}{\|\underline{w}\|}$

max. $d \Rightarrow$ max. b, or min. $\|\underline{w}\|$.

$\Rightarrow$ min. $\|\underline{w}\|$ for a given b.

and fix b by letting $b = 1$.

(S) $\therefore$ (1) $\Rightarrow$ Require $z_i(\underline{w}^T \underline{u}_i + w_0) \geq 1 \quad \forall i$

with minimal $\|\underline{w}\|$.

Use Lagrange Optimization

(2)
$$\begin{cases} \text{Minimize} \quad J(\underline{w}) = \|\underline{w}\|^2 \\ \text{subject to constraints:} \quad z_i(\underline{w}^T \underline{u}_i + w_0) - 1 \geq 0 \quad \forall i \end{cases}$$

Note: use $\underline{\lambda}$ instead of $\underline{\mu}$ for Lagrange multipliers

(2')
$$L(\underline{w}, w_0, \underline{\lambda}) = \frac{1}{2}\|\underline{w}\|^2 - \sum_{i=1}^{N} \lambda_i \left[z_i(\underline{w}^T \underline{u}_i + w_0) - 1 \right]$$

$$\left.\begin{array}{ll} \lambda_i \geq 0 & \forall i \\ \lambda_i \left[z_i(\underline{w}^{*T} \underline{u}_i + w_0^*) - 1 \right] = 0 & \forall i \\ z_i(\underline{w}^T \underline{u}_i + w_0) - 1 \geq 0 & \forall i \end{array}\right\} \text{KKT conditions}$$

↑ Primal form of Lagrangian (linearly separable case)

SVM – Dual Lagrangian for Separable Case

Easier to use if expressed in its dual representation:

Solve:
$$\left\{ \begin{array}{l} \nabla_{\underline{w}} L(\underline{w}, w_0, \underline{\lambda}) = \underline{0} \\ \text{and } \dfrac{\partial}{\partial w_0} L(\underline{w}, w_0, \underline{\lambda}) = 0 \end{array} \right\} \quad \text{for } \underline{w}^*.$$

Substitute into (2') $\Rightarrow$

(2'')
$$L_D(\underline{\lambda}) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j z_i z_j \underline{x}_i^T \underline{x}_j + \sum_{i=1}^{N} \lambda_i$$

with: $\sum_{i=1}^{N} \lambda_i z_i = 0$

$\lambda_i \geq 0$, $\lambda_i [z_i (\underline{w}^{*T} \underline{x}_i + w_0) - 1] = 0 \quad \forall i$

$\underline{w}^* = \sum_{i=1}^{N} \lambda_i z_i \underline{x}_i$

$z_i (\underline{w}^T \underline{x}_i + w_0) - 1 \geq 0 \quad \forall i$

← Dual form of Lagrangian (linearly separable case)

$L_D(\underline{\lambda}) = L(\underline{w}^*, w_0^*, \underline{\lambda}) = L$ optimized w.r.t. $\underline{w}, w_0$.

Comments

1. $L_D(\underline{\lambda}) \Rightarrow$ only need to optimize w.r.t. $\underline{\lambda}$; yields solution $\underline{\lambda}^*$.

2. In $L_D(\underline{\lambda})$, $\underline{u}$ appears in terms of what form?

$$\lambda_i \lambda_j (z_i \underline{u}_i)^T (z_j \underline{u}_j) \rightarrow \text{form of a kernel!}$$

$\Rightarrow$ can define nonlinear transformation by a choice of kernel $k(\underline{x}_i, \underline{x}_j)$.

Implementation

For simple problems, can be solved algebraically.

For more complex problems, a variety of numerical techniques can be used, including:

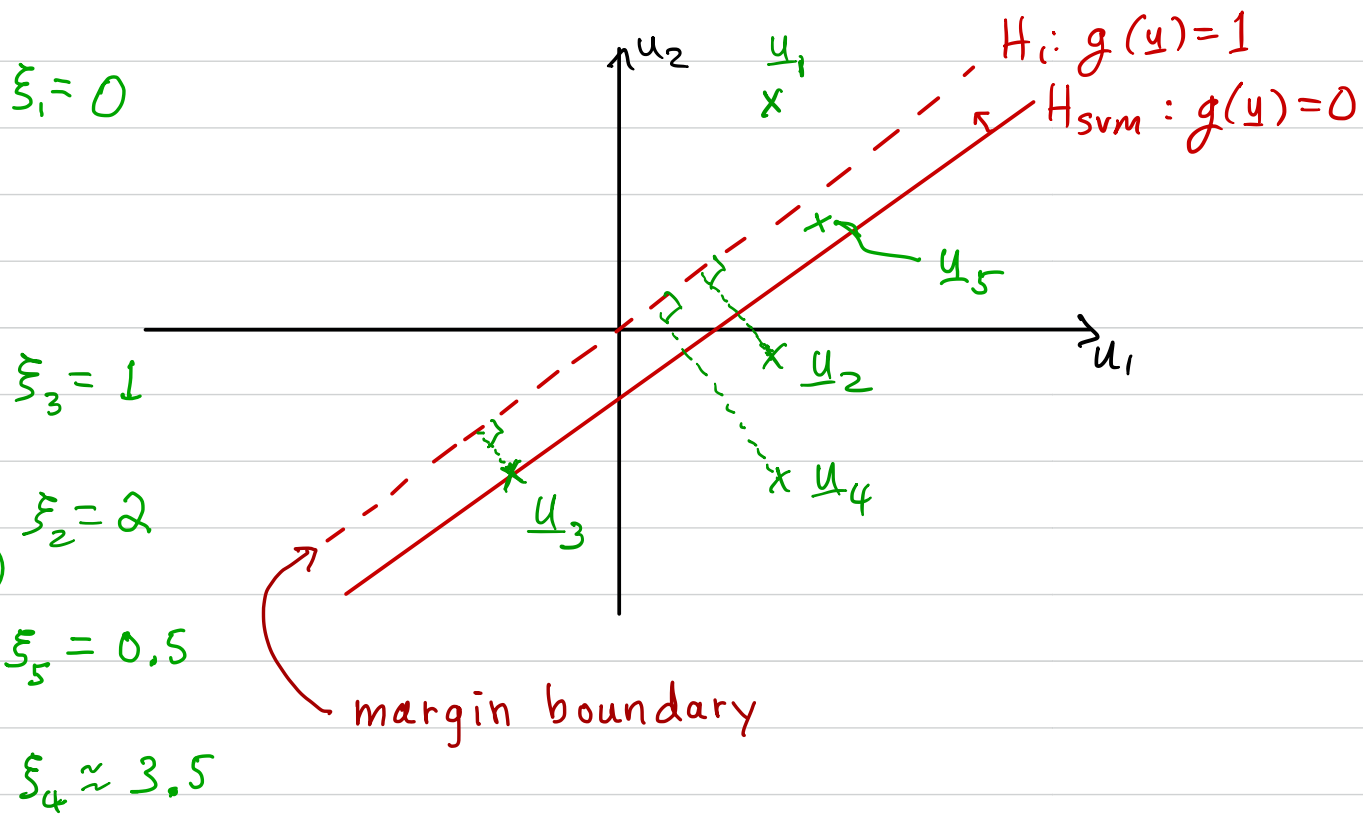
1. Quadratic programming (optimizes quadratic fcn. with constraints)

2. Projected gradient descent / sub-gradient methods (e.g., Pegasos).

SVM Learning without Linearly Separable Assumption

→ Introduce slack variables ξ_i, such that: (ξ is xi)

$\xi_i = 0$ if $\underline{u}_i$ is on correct side of margin boundary.
ξ_i = normalized distance to margin boundary, if on incorrect side of margin boundary.
$\xi_i = 1$ if $\underline{u}_i$ is on decision boundary.



$\xi_1 = 0$

$\xi_3 = 1$

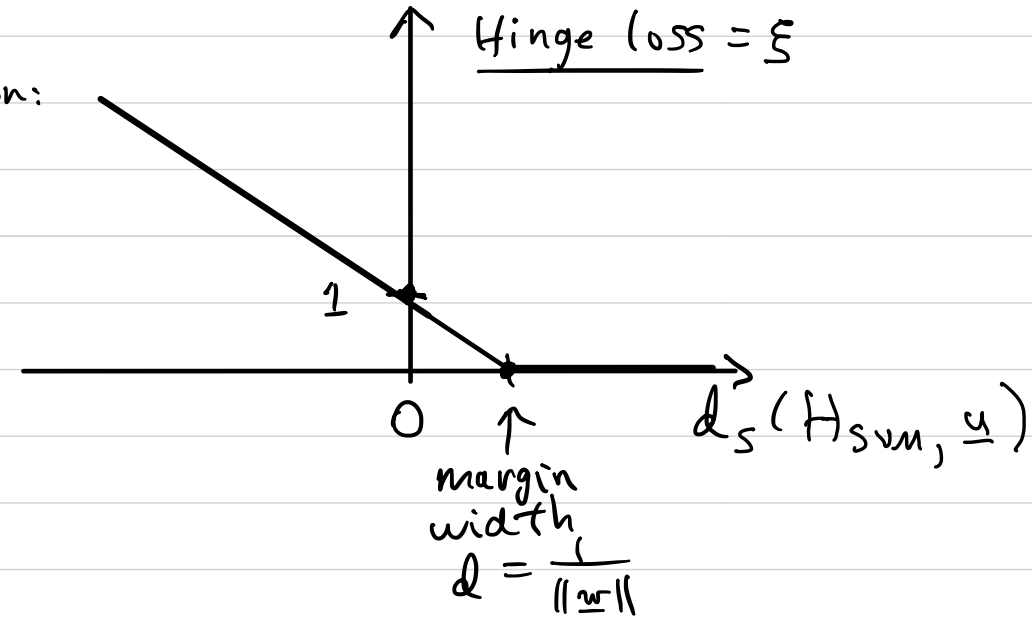
$\xi_2 = 2$

(S)

$\xi_5 = 0.5$

$\xi_4 \approx 3.5$

⇒ Loss function:



Now, (2) becomes:

(3)

Minimize $J = \frac{1}{2}\|\underline{w}\|^2 + C\sum_{i=1}^{N} \xi_i$

new term
slack-variable parameter.

Constraints:

$$z_i(\underline{w}^T \underline{u}_i + w_0) \geq 1 - \xi_i \qquad \forall i$$

new term

Generates slighty different $L(\underline{w}, w_0, \underline{\lambda})$ and $L_D(\underline{\lambda})$, and constraints.

SVM With Slack Variables

L BECOMES:

(3')

$$L(\underline{w}, w_0, \underline{\xi}, \underline{\lambda}, \underline{\mu}) = \frac{1}{2}\|\underline{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\lambda_i\left[z_i(\underline{w}^T\underline{u}_i + w_0) - (1-\xi_i)\right] - \sum_{i=1}^{N}\mu_i\xi_i$$

$$\lambda_i \geq 0, \quad \lambda_i^*\left[z_i(\underline{w}^{*T}\underline{u}_i + w_0) - (1-\xi_i)\right] = 0 \quad \forall i$$

$$\mu_i \geq 0, \quad \mu_i^*\xi_i^* = 0 \quad \forall i$$

$$z_i(\underline{w}^T\underline{u}_i + w_0) - (1-\xi_i) \geq 0, \quad \xi_i \geq 0 \quad \forall i$$

} KKT conditions

↑ Primal L (no linearly sep. assumption)

From L, we can derive L_D:

$\Rightarrow L_D(\underline{\lambda})$ has the same form as L_D for the separable case, except that:

constraint $\lambda_i \geq 0 \quad \forall i$

becomes $0 \leq \lambda_i \leq C \quad \forall i$.

Back to Nonlinear Mapping

$$L_D(\underline{\lambda}) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j z_i z_j \underline{u}_i^T \underline{u}_j$$

$$z_i z_j \underline{\phi}^T(\underline{x}_i) \underline{\phi}(\underline{x}_j)$$

$k(\underline{x}_i, \underline{x}_j) = \underline{\phi}^T(\underline{x}_i) \underline{\phi}(\underline{x}_j)$ is the kernel function

$$L_D(\underline{\lambda}) = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j z_i z_j k(\underline{x}_i, \underline{x}_j)$$

$\Rightarrow$ We can:

1. Pick a kernel function
2. Use kernel substitution

to implement the nonlinear transformation, or to use a similarity function appropriate for the application.

Support Vector Regression (SVR) [Bishop 7.1.4] (non-augmented notation)

SVM adapted to regression problems.

Motivation: to have a regression model suitable to sparse data (e.g., high D, relatively low N).

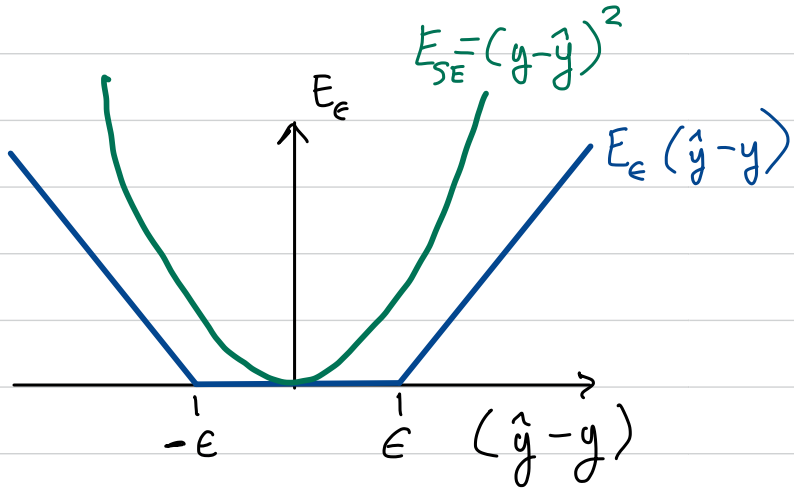
In Ridge Regression, we used a criterion fcn.: MSE + ℓ_2^2 regularizer:

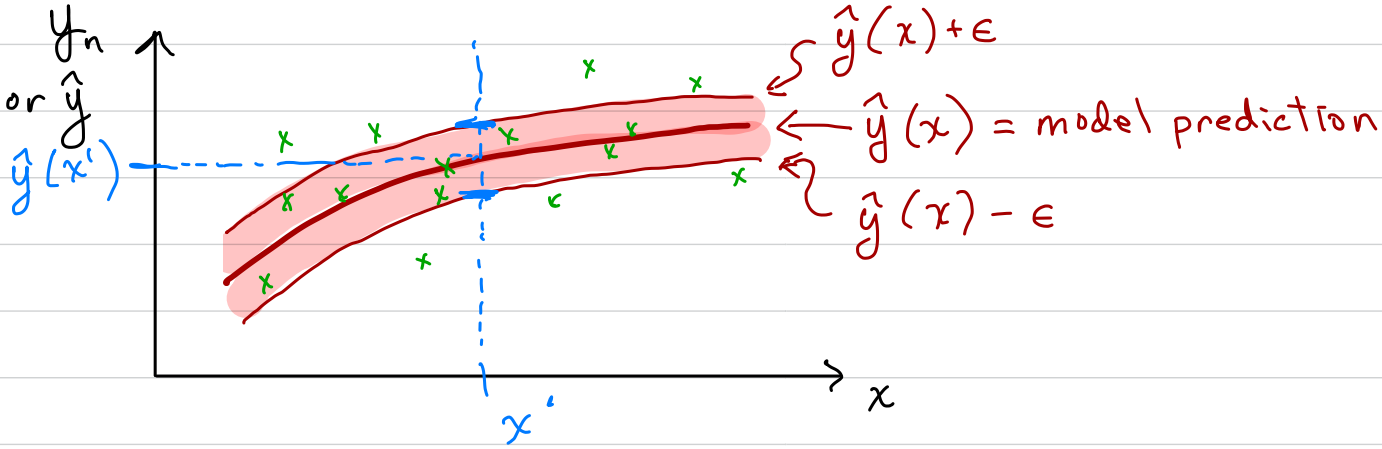
$$J_{RR}(\underline{w}) = \underbrace{\frac{1}{N}\sum_{n=1}^{N}\left[\hat{y}(\underline{x}_n) - y_n\right]^2}_{\text{use "}\epsilon\text{-insensitive" loss}} + \underbrace{\lambda \|\underline{w}\|_2^2}_{\text{keep } \ell_2^2 \text{ regularizer}}, \quad \lambda \geq 0$$

For SVR: instead of squared-error loss

ϵ-insensitive loss E_ϵ:

$$E_\epsilon(\hat{y}-y) = \begin{cases} 0, & \text{if } |\hat{y}-y| < \epsilon \\ |\hat{y}-y| - \epsilon & \text{if } |\hat{y}-y| \geq \epsilon \end{cases}$$





Any data points in the shaded region have $E_\epsilon = 0$.

- 0 error for predictions $\hat{y}(\underline{x}_n)$ within ϵ of known output y_n, means learning algorithm is less likely to overfit to noise or small variations in data.