Support Vector Regression

Mingmin Chi
School of Computer Science
Fudan University

- Support Vector Classifier: Revisit
- Support Vector Regression

History of SVM

- SVM was first introduced in 1992 [1]
- SVM is related to statistical learning theory [2]
- SVM becomes popular because of its success in handwritten digit recognition
  - 1.1% test error rate for SVM. This is the same as the error rates of a carefully constructed neural network, LeNet 4.
- SVM is now regarded as an important example of “kernel methods”, one of the key area in machine learning
  - Note: the meaning of “kernel” is different from the “kernel” function for Parzen windows

[1] B.E. Boser *et al*. A Training Algorithm for Optimal Margin Classifiers. Proceedings of the Fifth Annual Workshop on Computational Learning Theory 5 144-152, Pittsburgh, 1992.

[2] [3] V. Vapnik. The Nature of Statistical Learning Theory. 2nd edition, Springer, 1999.

Example

- Suppose we have 5 1D data points
  - $x_1=1$, $x_2=2$, $x_3=4$, $x_4=5$, $x_5=6$, with 1, 2, 6 as class 1 and 4, 5 as class 2 $\Rightarrow$ $y_1=1$, $y_2=1$, $y_3=-1$, $y_4=-1$, $y_5=1$
- We use the polynomial kernel of degree 2
  - $K(x,y) = (xy+1)^2$
  - C is set to 100
- We first find a_i ($i=1, \ldots, 5$) by

$$\text{max.} \quad \sum_{i=1}^{5} \alpha_i - \frac{1}{2} \sum_{i=1}^{5} \sum_{j=1}^{5} \alpha_i \alpha_j y_i y_j (x_i x_j + 1)^2$$

$$\text{subject to } 100 \geq \alpha_i \geq 0, \sum_{i=1}^{5} \alpha_i y_i = 0$$

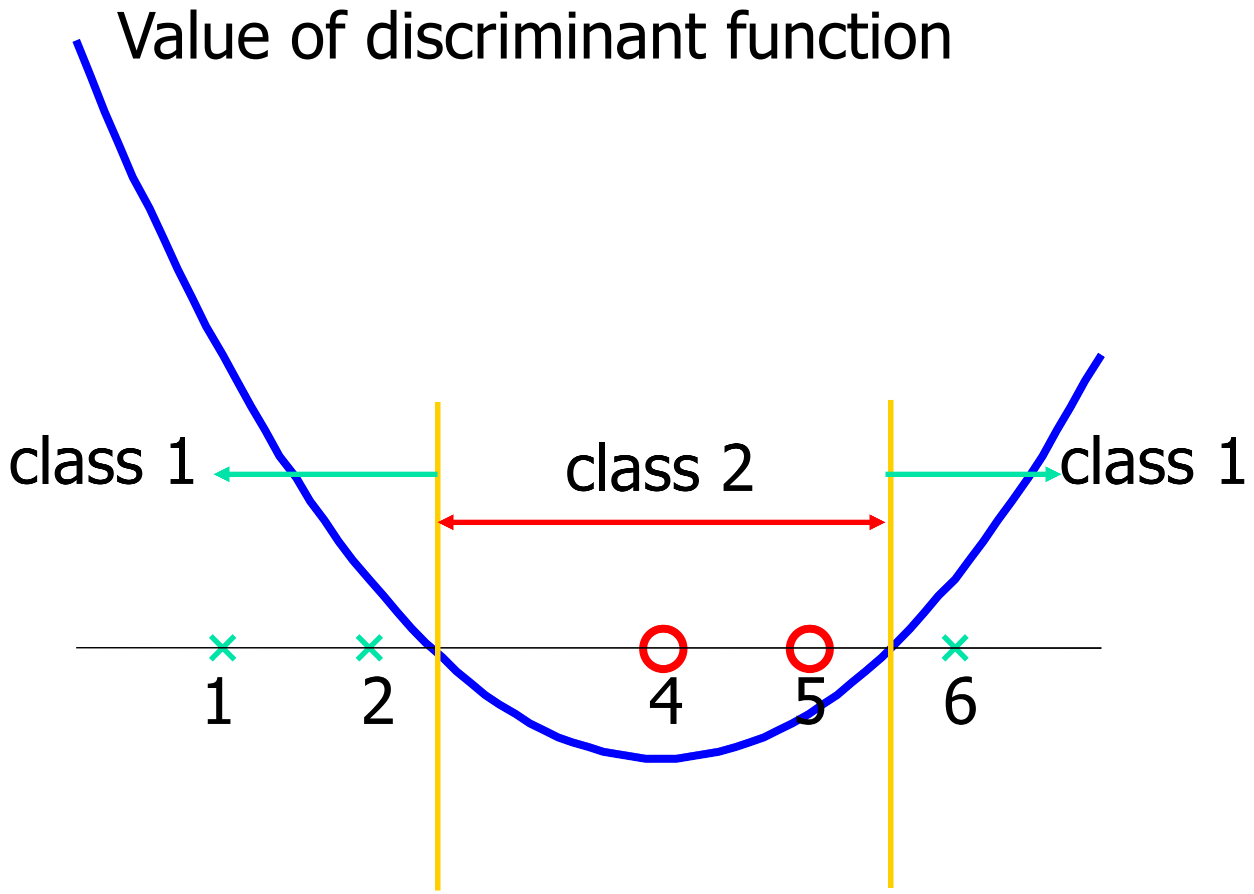
Example

- By using a QP solver, we get
  - a_1=0, a_2=2.5, a_3=0, a_4=7.333, a_5=4.833
  - Note that the constraints are indeed satisfied
  - The support vectors are {x_2=2, x_4=5, x_5=6}
- The discriminant function is

$$
\begin{aligned}
& f(z) \\
&= 2.5(1)(2z+1)^2 + 7.333(-1)(5z+1)^2 + \overbrace{4.833}^{\alpha_5}\overbrace{(1)}^{y_5}\overbrace{(6z+1)^2}^{K(z,x_5)} + b \\
&= 0.6667z^2 - 5.333z + b
\end{aligned}
$$

- *b* is recovered by solving f(2)=1 or by f(5)=-1 or by f(6)=1, as x_2 and x_5 lie on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = 1$and x_4 lies on the line $\phi(\mathbf{w})^T\phi(\mathbf{x}) + b = -1$
- All three give b=9 $\Longrightarrow$ $f(z) = 0.6667z^2 - 5.333z + 9$

Example


Value of discriminant function
class 1
class 2
class 1
1
2
4
5
6

Justification of SVM

- Large margin classifier
- Structural Risk Minimization (SRM) vs ERM
- Ridge regression: the term $\frac{1}{2}||w||^2$ "shrinks" the parameters towards zero to avoid overfitting
- The term $\frac{1}{2}||w||^2$ can also be viewed as imposing a weight-decay prior on the weight vector, and we find the MAP estimate

Choosing the Kernel Function

- Kernel function describes the correlation or similarity between two data points
- Probably the most tricky part of using SVM
- The kernel function is important because it creates the kernel matrix, which summarizes all the data
- In practice, a low degree polynomial kernel or RBF kernel with a reasonable width is a good initial try
- Note that SVM with RBF kernel is closely related to RBF neural networks, with the centers of the radial basis functions automatically chosen for SVM

Summary: Steps for Classification

- Prepare the pattern matrix
- Select the kernel function to use
- Select the parameter of the kernel function and the value of *C*
  - You can use the values suggested by the SVM software, or you can set apart a validation set to determine the values of the parameter
- Execute the training algorithm and obtain the a_i
- Unseen data can be classified using the a_i and the support vectors

Strengths and Weaknesses of SVM

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
  - Non-traditional data like strings and trees can be used as input to SVM, instead of feature vectors
- Weaknesses
  - Need to choose a “good” kernel function
  - Suffer from out-of memory probem with huge amount of training dataset

Other Types of Kernel Methods

- A lesson learnt in SVM: a linear algorithm in the feature space is equivalent to a non-linear algorithm in the input space
- Standard linear algorithms can be generalized to its non-linear version by going to the feature space
  - Kernel principal component analysis
  - kernel independent component analysis
  - kernel canonical correlation analysis
  - kernel k-means
  - ...

Conclusion

- SVM is a useful alternative to multi-layer perceptron neural networks
- Two key concepts of SVM:
  - maximize the margin and
  - the kernel trick

SUPPORT VECTOR REGRESSION

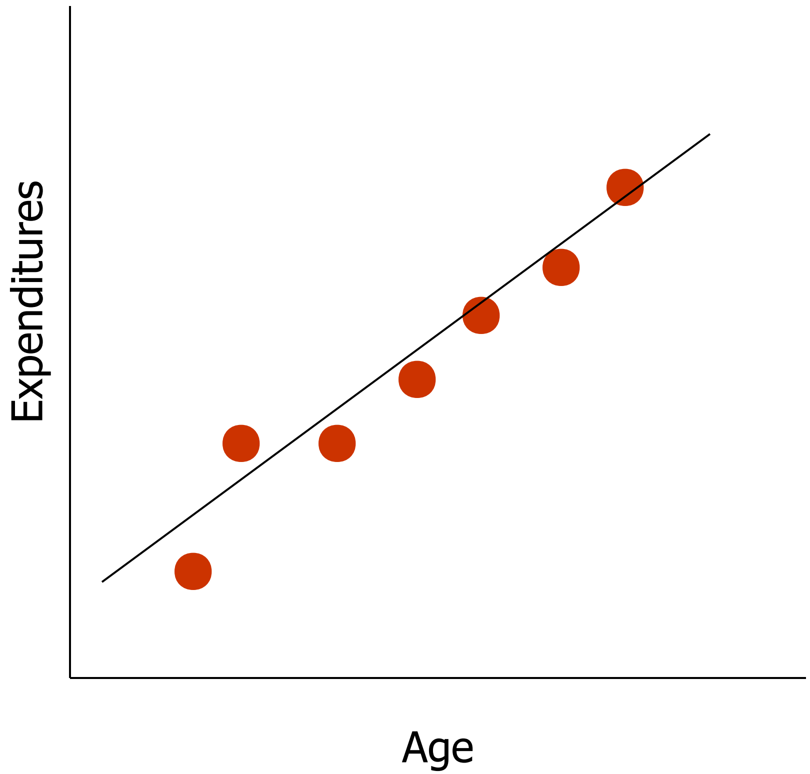
The Regression Task

- Given training data: $\{ (x_1, y_1), \ldots , (x_n , y_n) \} \in \Re^d$
- Find function: $f : \Re^d \rightarrow \Re$

"best function" = the expected error on *unseen* data $(x_{n+1}, y_{n+1}), \ldots , (x_{n+k} , y_{n+k})$ is minimal

- Existing techniques to solve the classification task:
  - Classical (Linear) Regression
  - Ridge Regression
  - NN

Linear Support Vector Regression



- Marketing Problem

Given variables:

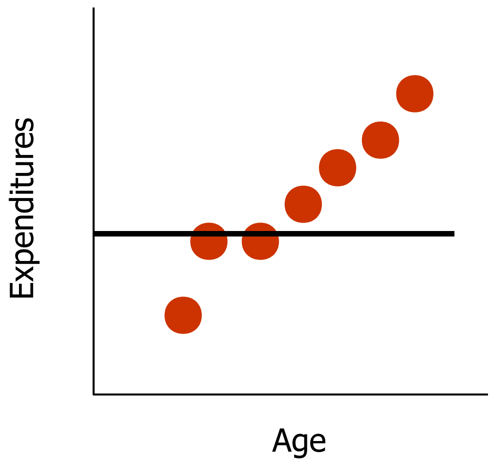
- person’s age
- income group
- season
- holiday duration
- location
- number of children
- etc. (12 variables)

Predict:

- the level of holiday Expenditures

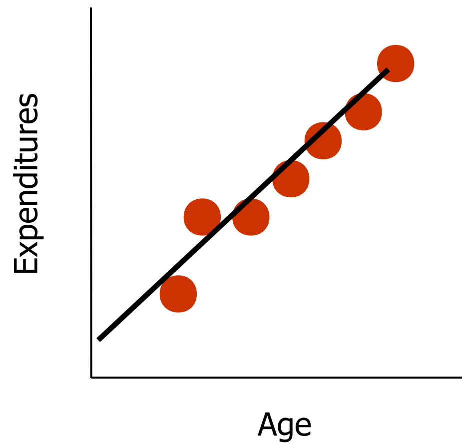
Data collected by Erasmus University Rotterdam in 2003

Linear Support Vector Regression



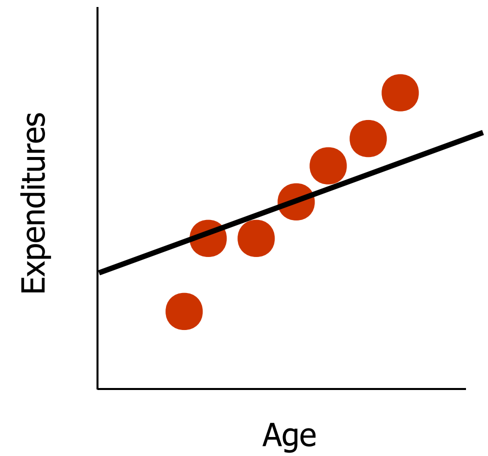
“Lazy case”

(underfitting)



“Suspiciously smart case”

(overfitting)

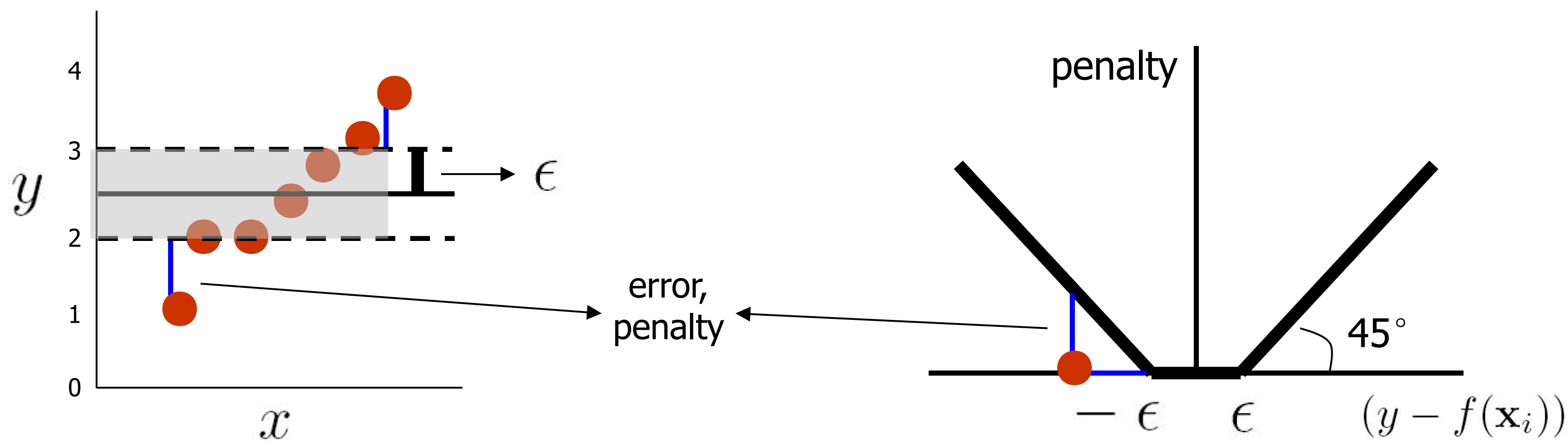


“Compromise case”, SVR

(good generalizability)

Linear Support Vector Regression

- The *epsilon*-insensitive loss function



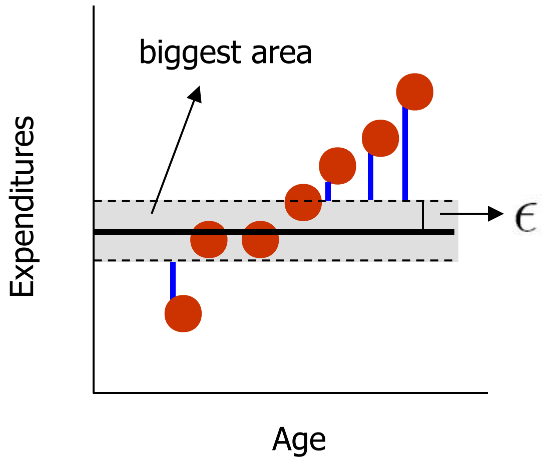
$$y = w_1 x + b$$

$$w_1 = 0$$

$$b = 2.5$$

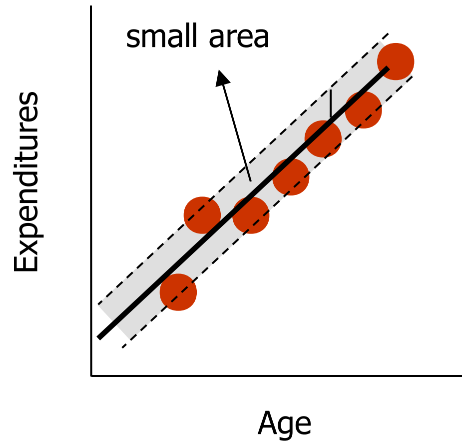
$$|y_i - f(\mathbf{x}_i)|_\epsilon \equiv \max\{0, |y_i - f(\mathbf{x}_i)| - \epsilon\}$$

Linear Support Vector Regression



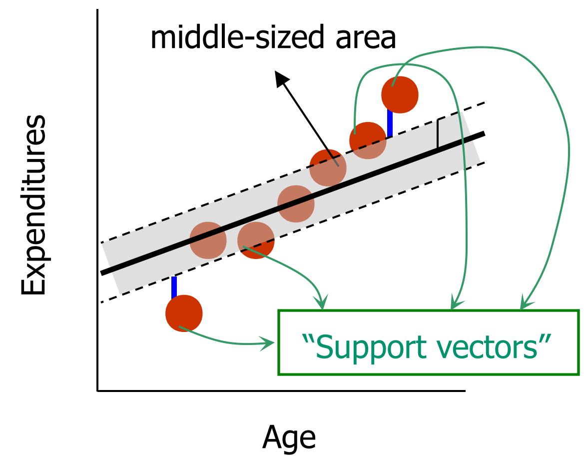
"Lazy case"

(underfitting)



"Suspiciously smart case"

(overfitting)



"Compromise case", SVR

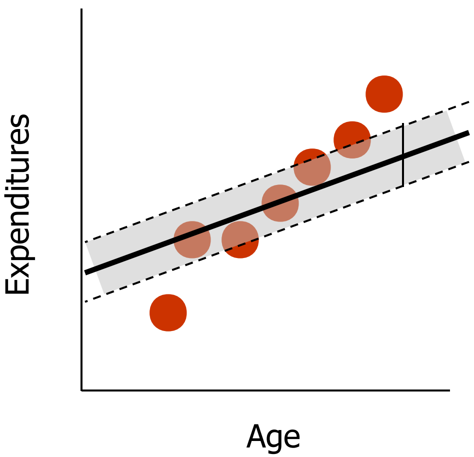
(good generalizability)

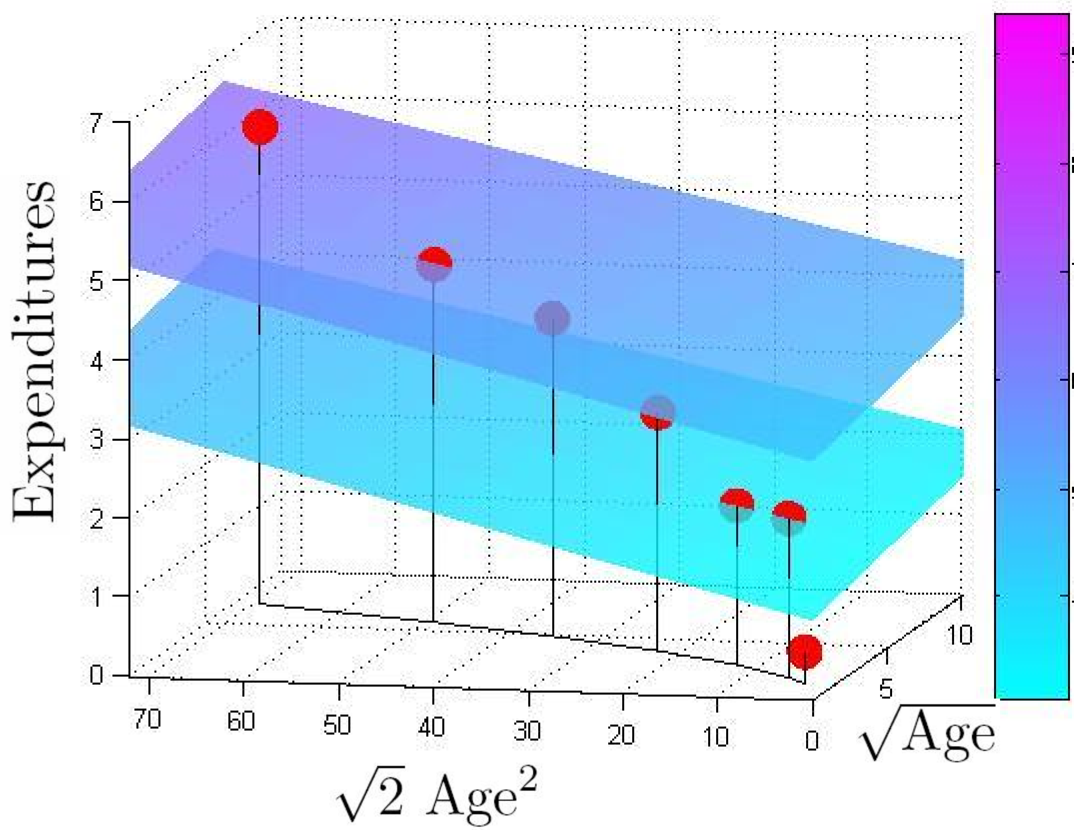
- **The thinner the "tube", the more complex the model**

Non-linear Support Vector Regression

- Map the data into a *higher*-dimensional space:

$$x \rightarrow \Phi(x) = (\sqrt{x}, \sqrt{2}x^2)$$
$$\text{Age} \rightarrow \Phi(\text{Age})$$
$$\text{Age} \rightarrow (\sqrt{\text{Age}}, \sqrt{2}\ \text{Age}^2)$$





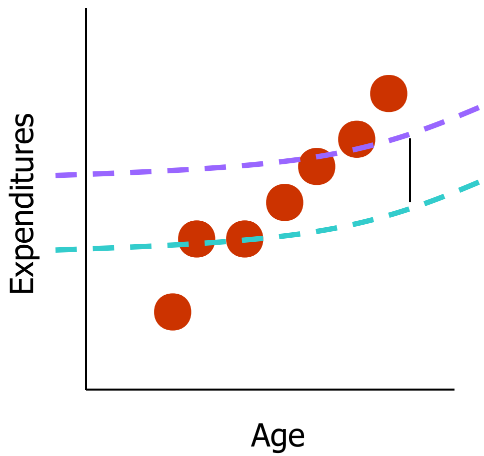
$$y = w_1 x + b$$

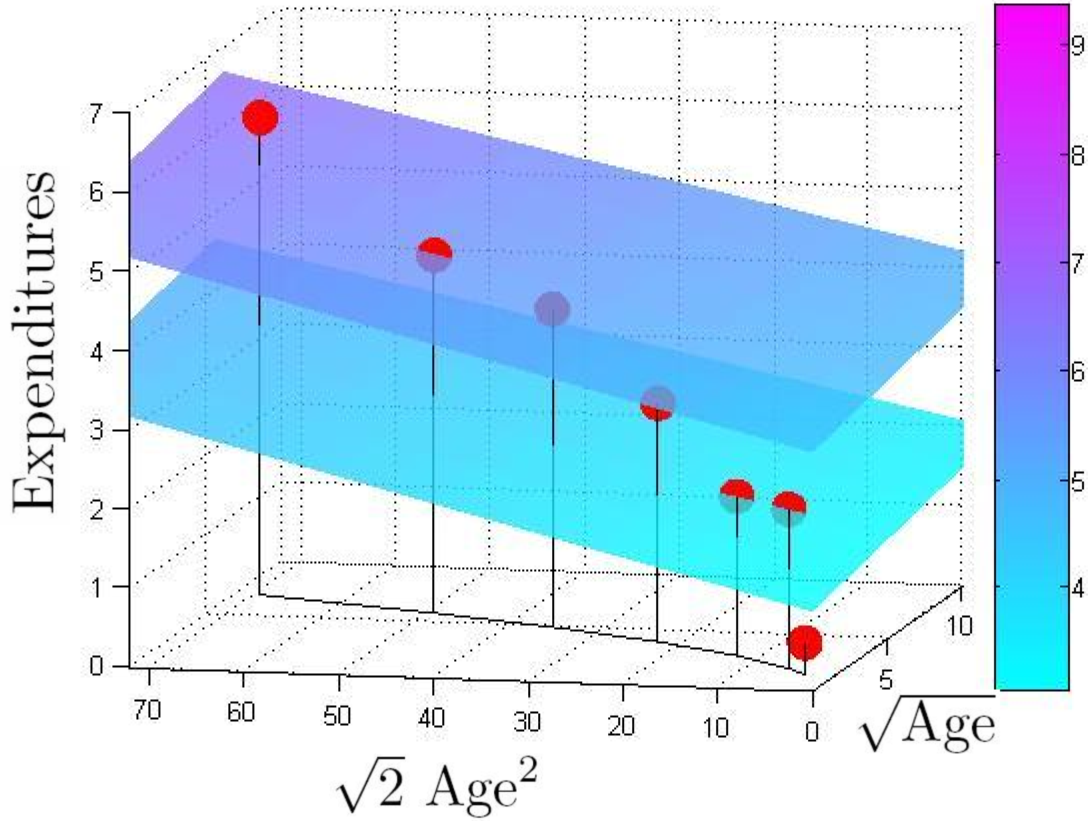
$$y = w_1\sqrt{x} + w_2\sqrt{2}x^2 + b$$

Non-linear Support Vector Regression

- Map the data into a *higher*-dimensional space:

$$x \to \Phi(x) = (\sqrt{x}, \sqrt{2}x^2)$$
$$\text{Age} \to \Phi(\text{Age})$$
$$\text{Age} \to (\sqrt{\text{Age}}, \sqrt{2}\ \text{Age}^2)$$





$$y = \mathbf{w}'\Phi(x) + b$$

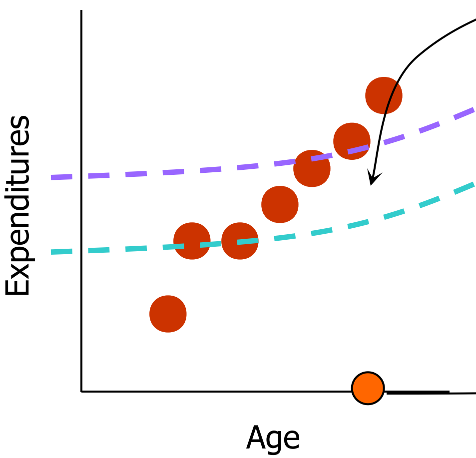
$$y = w_1\sqrt{x} + w_2\sqrt{2}x^2 + b$$

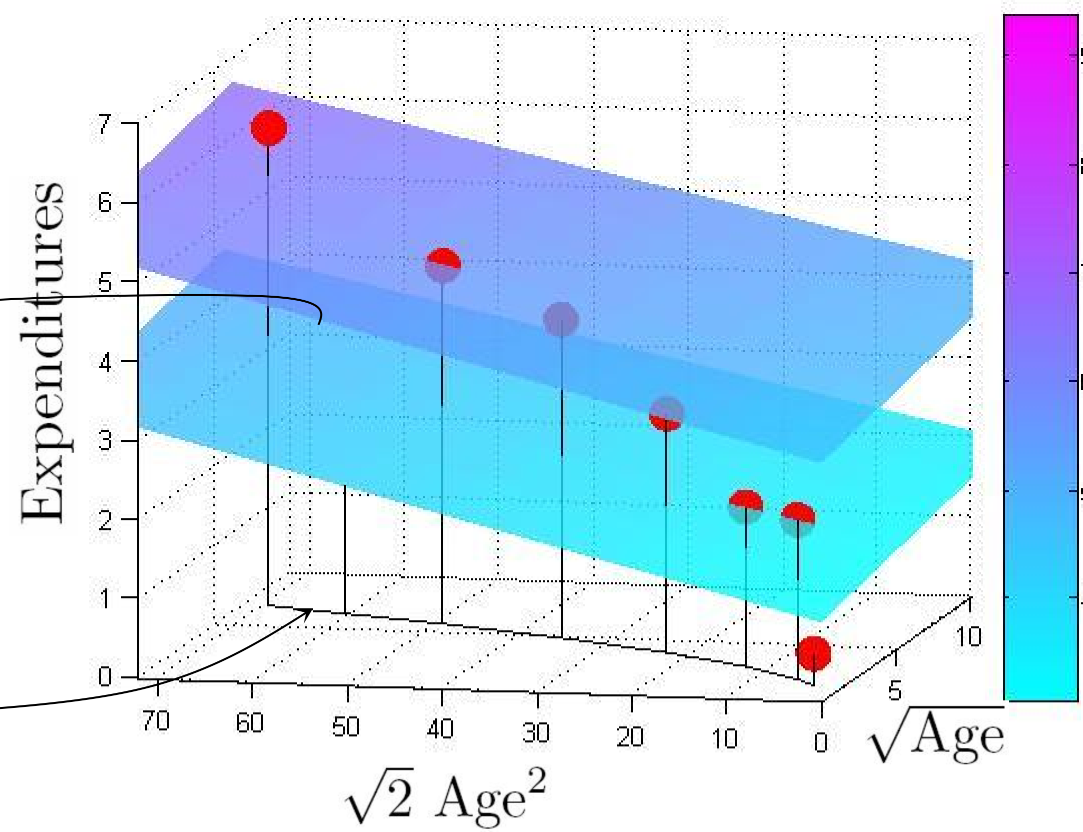
Non-linear Support Vector Regression

- Finding the value of a new point:

$$z_1 \rightarrow (\sqrt{z_1}, \sqrt{2}z_1^2)$$

$$f(z_1) = w_1\sqrt{z_1} + w_2\sqrt{2}z_1^2 + b$$





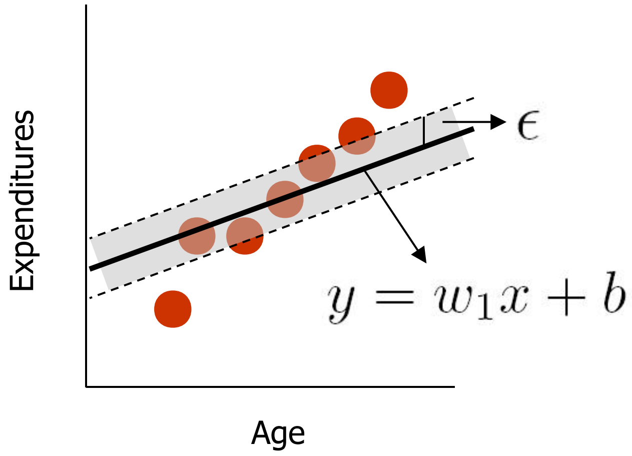
$$y = \mathbf{w}'\Phi(x) + b$$

$$y = w_1\sqrt{x} + w_2\sqrt{2}x^2 + b \pm \epsilon$$

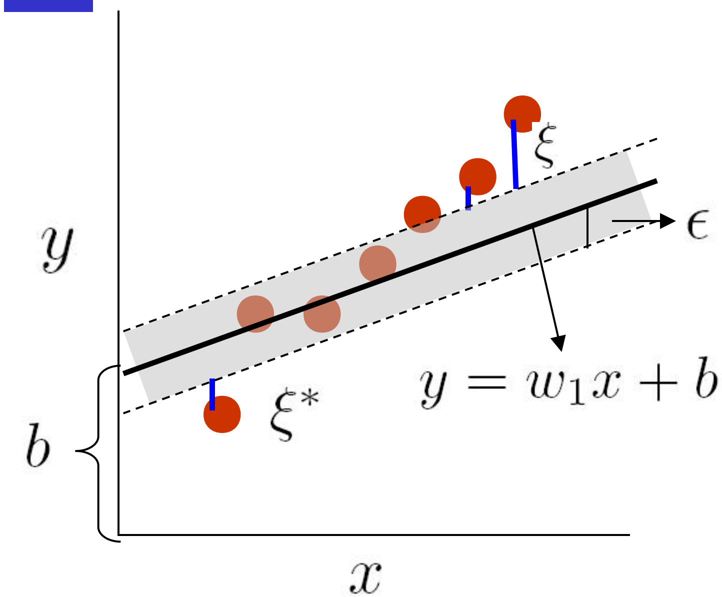
Linear SVR: derivation

- Given training data $\{x_i, y_i\}_{i=1}^{n}$
- Find: w_1 , b

  such that $y = w_1 x + b$ optimally describes the data:



Linear SVR: derivation



$| \, w_1 \, |$ **vs.** $\sum_i (\xi_i + \xi_i^*)$

Complexity — Sum of errors

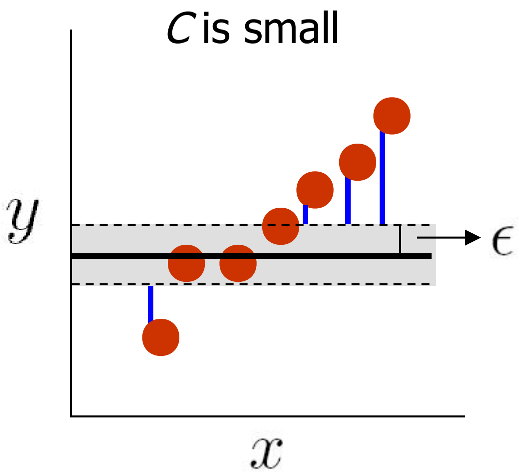
$$\min_{w_1, b, \xi_i, \xi_i^*} \quad \frac{1}{2} w_1^2 + C \sum_i (\xi_i + \xi_i^*)$$

Case I: $w_1 \downarrow$ ➞ "tube"$\uparrow$ ➞ complexity$\downarrow$ ➞ $\sum_i (\xi_i + \xi_i^*) \uparrow$

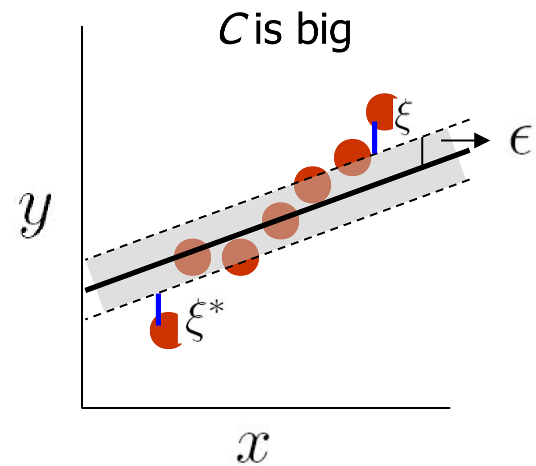
Case II: $w_1 \uparrow$ ➞ "tube"$\downarrow$ ➞ complexity$\uparrow$ ➞ $\sum_i (\xi_i + \xi_i^*) \downarrow$

Linear SVR: derivation

$$\min_{w_1, b, \xi_i, \xi_i^*} \quad \frac{1}{2} w_1^2 + C \sum_i (\xi_i + \xi_i^*)$$



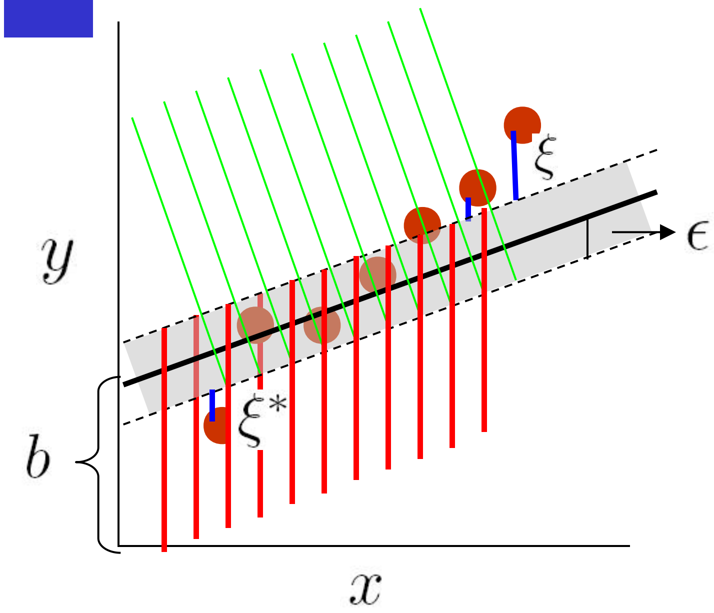
- The role of *C*



Case I: $w_1 \downarrow$ ➡ "tube"$\uparrow$ ➡ complexity$\downarrow$ ➡ $\sum_i (\xi_i + \xi_i^*) \uparrow$

Case II: $w_1 \uparrow$ ➡ "tube"$\downarrow$ ➡ complexity$\uparrow$ ➡ $\sum_i (\xi_i + \xi_i^*) \downarrow$

Linear SVR: derivation



$$y = w_1 x + b$$

$$\min_{w_1, b, \xi_i, \xi_i^*} \quad \frac{1}{2} w_1^2 + C \sum_i (\xi_i + \xi_i^*)$$

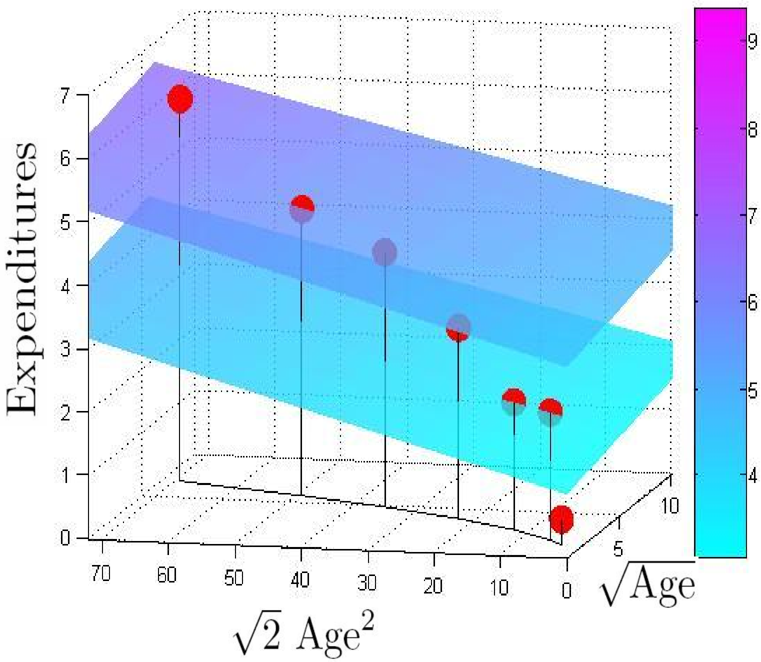
Subject to:

$$y_i - (w_1 x_{i1}) - b \leq \epsilon + \xi_i$$

$$(w_1 x_{i1}) + b - y_i \leq \epsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0 \quad i = 1, 2, \ldots, n$$

Non-linear SVR: derivation



$$\min_{w_1, b, \xi_i, \xi_i^*} \quad \frac{w_1^2 + w_2^2}{2} + C \sum_i (\xi_i + \xi_i^*)$$

Subject to:

$$y_i - (\mathbf{w}'\phi(x_{i1})) - b \leq \epsilon + \xi_i$$
$$(\mathbf{w}'\phi(x_{i1})) + b - y_i \leq \epsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \geq 0 \qquad i = 1, 2, \ldots, n$$

$$y = \mathbf{w}'\Phi(x) + b$$

Non-linear SVR: derivation

$$\min_{\mathbf{w},b,\xi_i,\xi_i^*} \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_i (\xi_i + \xi_i^*)$$

Subject to:

$$y_i - (\mathbf{w}'\phi(\mathbf{x}_i)) - b \leq \epsilon + \xi_i$$
$$(\mathbf{w}'\phi(\mathbf{x}_i))) + b - y_i \leq \epsilon + \xi_i^*$$
$$\xi_i, \xi_i^* \geq 0 \qquad i = 1, 2, \ldots, n$$

$$L := \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_i (\xi_i + \xi_i^*) - \sum_i (\eta_i \xi_i + \eta_i^* \xi_i^*)$$
$$- \sum_i \alpha_i (\epsilon + \xi_i - y_i + \mathbf{w}'\phi(\mathbf{x}_i) + b) - \sum_i \alpha_i^* (\epsilon + \xi_i^* + y_i - \mathbf{w}'\phi(\mathbf{x}_i)) - b)$$

Saddle point of *L* has to be found:

min with respect to $\mathbf{w}, b, \xi_i, \xi_i^*$

max with respect to $\alpha_i, \alpha_i^*, \eta_i, \eta_i^*$

Non-linear SVR: derivation

$$L := \frac{1}{2} \parallel \mathbf{w} \parallel^2 + C \sum_i (\xi_i + \xi_i^*) - \sum_i (\eta_i \xi_i + \eta_i^* \xi_i^*)$$
$$- \sum_i \alpha_i (\epsilon + \xi_i - y_i + \mathbf{w}' \phi(\mathbf{x}_i) + b) - \sum_i \alpha_i^* (\epsilon + \xi_i^* + y_i - \mathbf{w}' \phi(\mathbf{x}_i)) - b)$$

$$\frac{\partial L}{\partial \mathbf{w}} = \mathbf{w} - \sum_i (\alpha_i - \alpha_i^*) \phi(\mathbf{x}_i) = 0$$

...

$$f(\mathbf{x}) = \mathbf{w}' \phi(\mathbf{x}) + b$$

$$f(\mathbf{x}) = \sum_i (\alpha_i - \alpha_i^*)(\phi(\mathbf{x}_i)' \phi(\mathbf{x})) + b$$

$$f(\mathbf{x}) = \sum_i (\alpha_i - \alpha_i^*) k(\mathbf{x}_i, \mathbf{x}) + b$$

Strengths and Weaknesses of SVR

- Strengths of SVR:
  - No local minima
  - It scales relatively well to high dimensional data
  - Trade-off between classifier complexity and error can be controlled explicitly via *C* and *epsilon*
  - Overfitting is avoided (for any fixed *C* and *epsilon*)
  - Robustness of the results
  - The “curse of dimensionality” is avoided

- Weaknesses of SVR:

  What is the best trade-off parameter *C* and best *epsilon*?
  - What is a *good* transformation of the original space

The end!