Linear Regression and Support Vector Regression

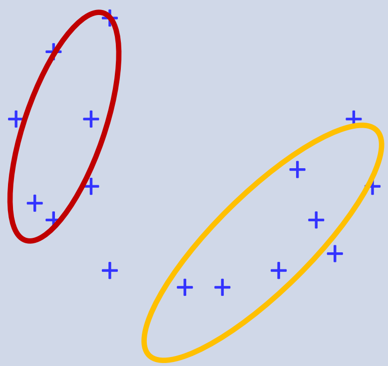
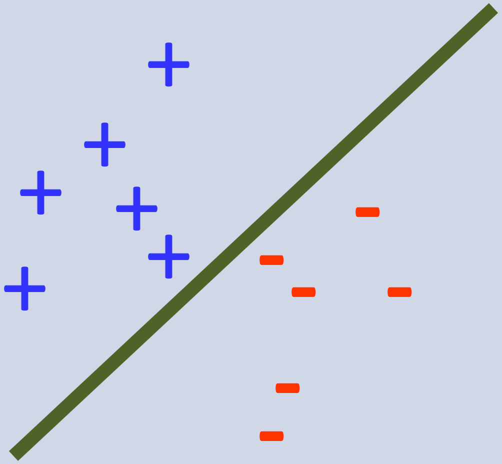
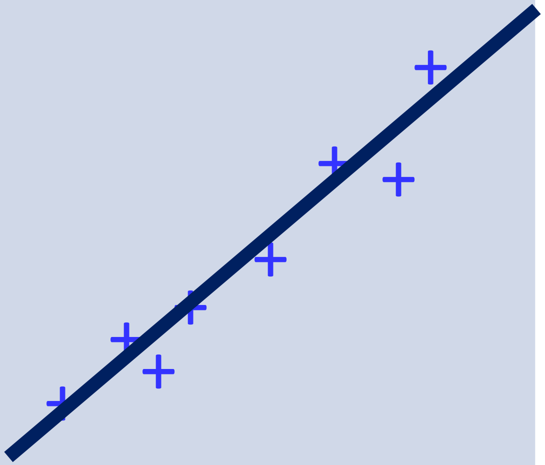
Paul Paisitkriangkrai
paulp@cs.adelaide.edu.au
The University of Adelaide

18 August 2014

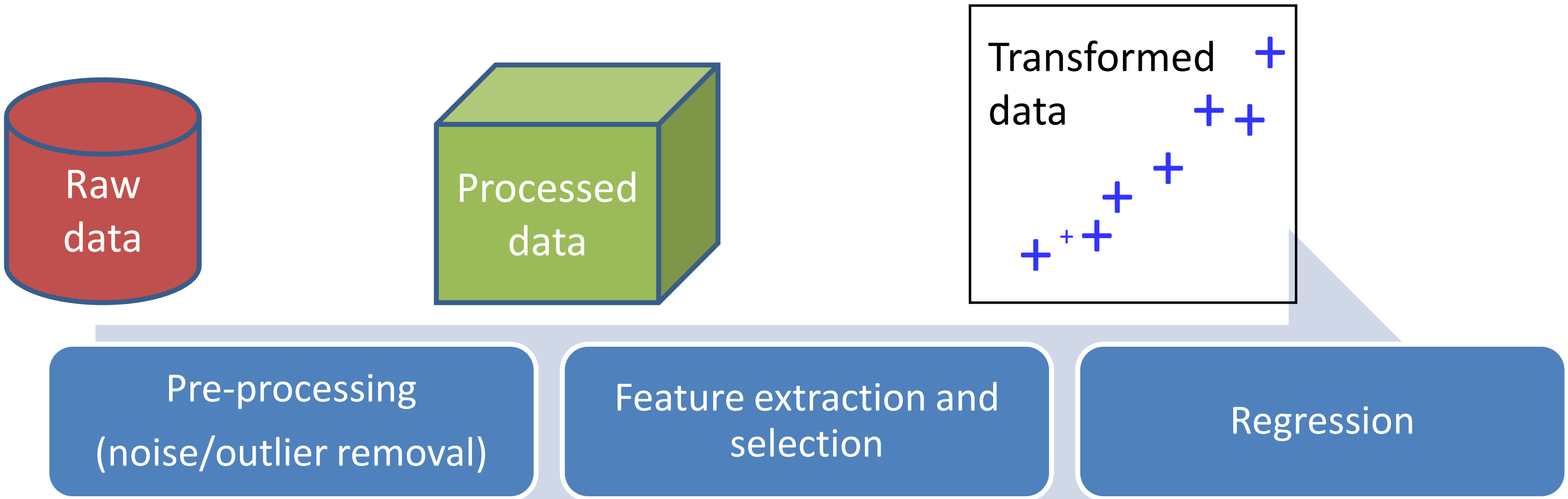
Outlines

- Regression overview
- Linear regression
- Support vector regression
- Machine learning tools available

Regression Overview

CLUSTERING	CLASSIFICATION	REGRESSION (THIS TALK)
		
K-means	• Decision tree • Linear Discriminant Analysis • Neural Networks • Support Vector Machines • Boosting	• Linear Regression • Support Vector Regression
Group data based on their characteristics	Separate data based on their labels	Find a model that can explain the output given the input

Data processing flowchart (Income prediction)



Height and sex seem to be irrelevant.

Sex	Age	Height	Income
M	20	1.7	25,000
F	30	1.6	55,000
M	**125**	1.8	30,000
…			

Mis-entry (should have been 25!!!)

Linear Regression

- Given data with n dimensional variables and 1 target-variable (real number)

  $$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_m, y_m)\}$$

  Where $\mathbf{x} \in \Re^n, y \in \Re$

- The objective: Find a function *f* that returns the best fit. $f : \Re^n \rightarrow \Re$

- Assume that the relationship between X and y is approximately linear. The model can be represented as (w represents coefficients and b is an intercept)

  $$f(w_1, \ldots, w_n, b) = y = \mathbf{w} \cdot \mathbf{x} + b + \varepsilon$$

Linear Regression

- To find the best fit, we minimize the sum of squared errors → Least square estimation

$$\min \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{m} (y_i - (\mathbf{w} \cdot \mathbf{x}_i + b))^2$$

- The solution can be found by solving

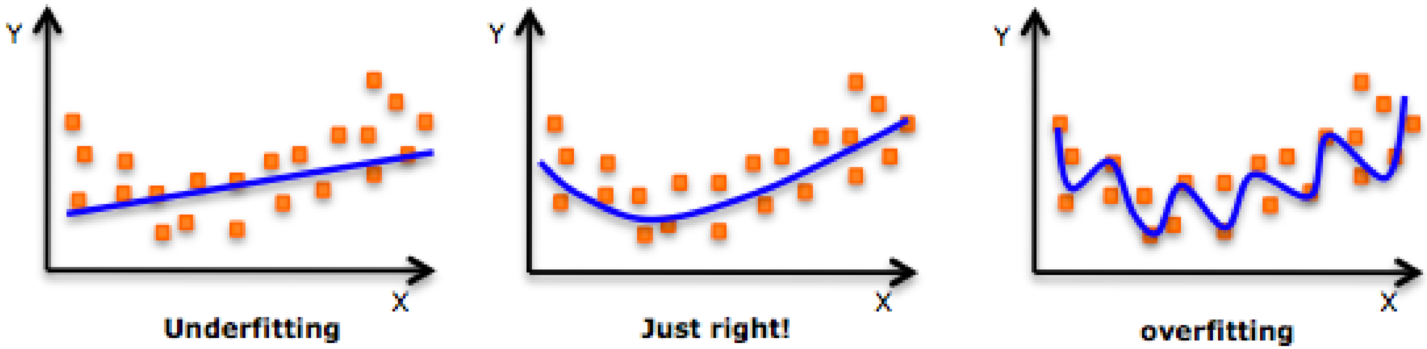
$$\hat{\mathbf{w}} = (X^T X)^{-1} X^T Y$$

(By taking the derivative of the above objective function w.r.t. $\mathbf{w}$ - Proof on the whiteboard)

- In MATLAB, the back-slash operator computes a least square solution.

Linear Regression

$$\min \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{m} (y_i - (\hat{\mathbf{w}} \cdot \mathbf{x}_i + \hat{b}))^2$$



- To ovoid over-fitting, a regularization term can be introduced (minimize a magnitude of w)
  - LASSO: $$\min \sum_{i=1}^{m} (y_i - \mathbf{w} \cdot \mathbf{x}_i - b)^2 + C \sum_{j=1}^{n} | w_j |$$
  - Ridge regression: $$\min \sum_{i=1}^{m} (y_i - \mathbf{w} \cdot \mathbf{x}_i - b)^2 + C \sum_{j=1}^{n} | \mathbf{w}_j^2 |$$

Support Vector Regression

- Find a function, f(x), with at most ε-deviation from the target y

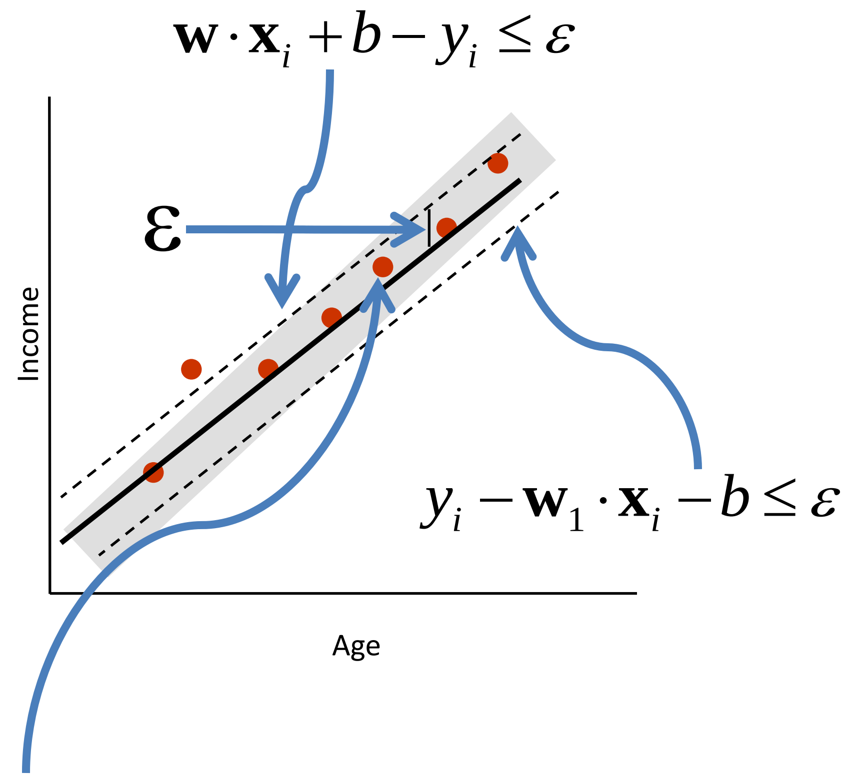
The problem can be written as a convex optimization problem

$$\min \frac{1}{2}\|\mathbf{w}\|^2$$
$$s.t.\ y_i - \mathbf{w}_1 \cdot \mathbf{x}_i - b \le \varepsilon;$$
$$\mathbf{w}_1 \cdot \mathbf{x}_i + b - y_i \le \varepsilon;$$

C: trade off the complexity

What if the problem is not feasible?

We can introduce slack variables
(similar to soft margin loss function).

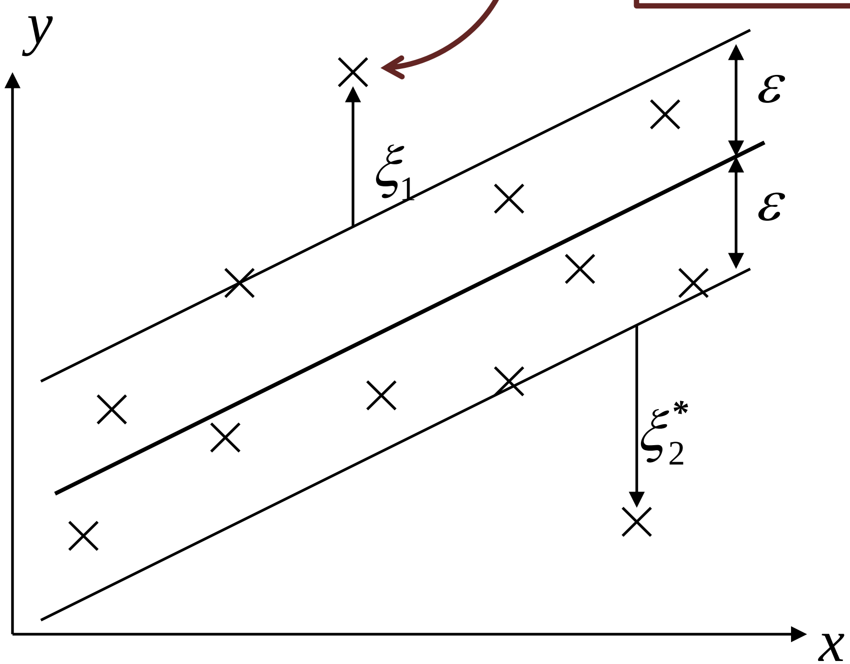


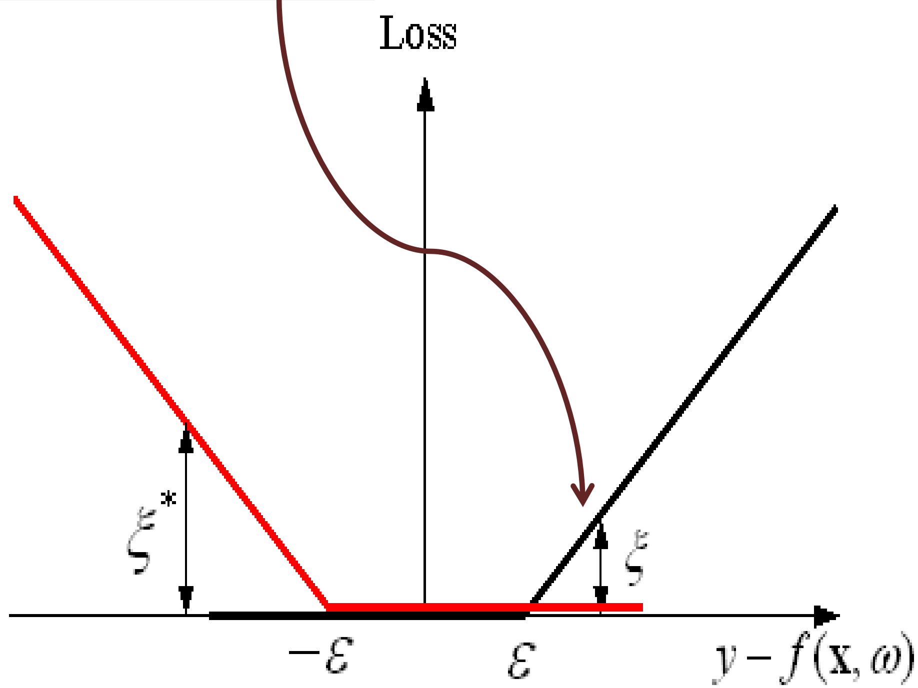
We do not care about errors as long as they are less than ε

Support Vector Regression

Assume linear parameterization $f(\mathbf{x}, \omega) = \mathbf{w} \cdot \mathbf{x} + b$

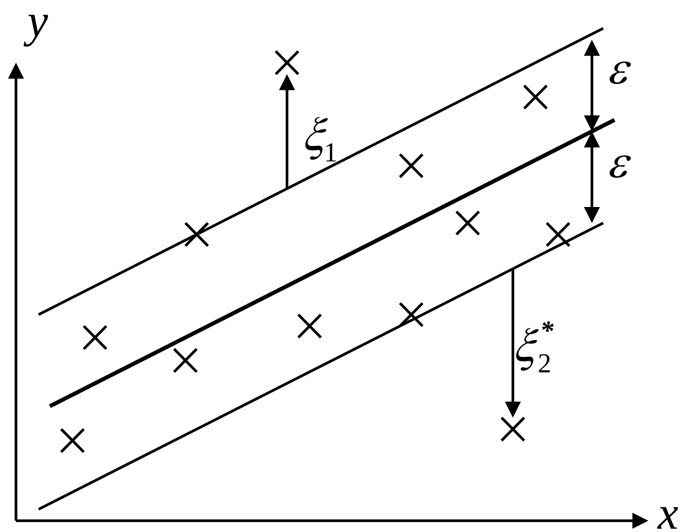
Only the point outside the ε-region contribute to the final cost





$$L_\varepsilon(y, f(\mathbf{x}, \omega)) = \max\left(|\, y - f(\mathbf{x}, \omega)\,| - \varepsilon, 0\right)$$

Soft margin



Given training data

$$(\mathbf{x}_i, y_i) \quad i = 1, \ldots, m$$

Minimize

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*)$$

Under constraints

$$\begin{cases} y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ (\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0, i = 1, \ldots, m \end{cases}$$

Dual problem derivation – whiteboard

- Primal

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}(\xi_i+\xi_i^*)$$

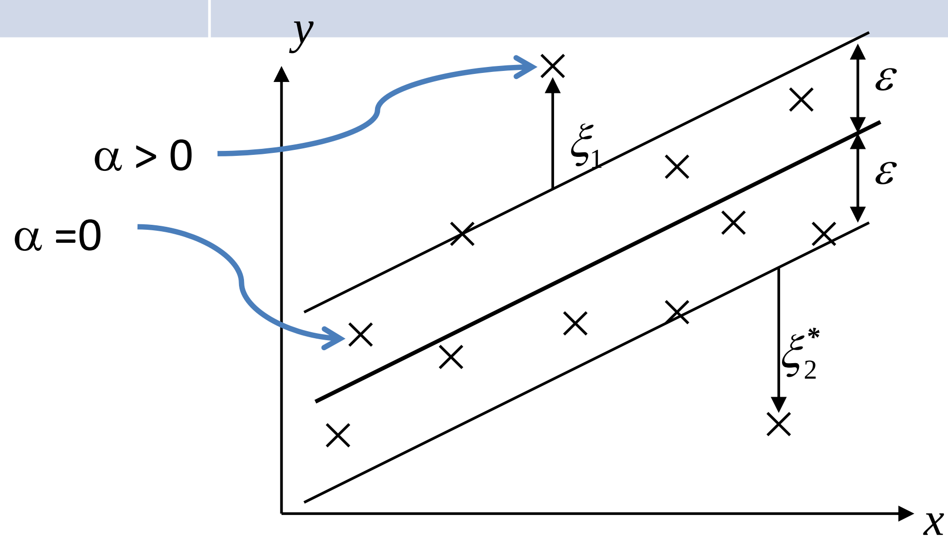
$$s.t.\begin{cases} y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ (\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i\ ,\xi_i^* \ge 0, i = 1,\ldots,m \end{cases}$$

- Dual

$$\max\begin{cases} \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i,x_j\rangle \\ -\varepsilon\sum_{i=1}^{m}(\alpha_i+\alpha_i^*)+\sum_{i=1}^{m}y_i(\alpha_i-\alpha_i^*) \end{cases}$$

$$s.t.\sum_{i=1}^{m}(\alpha_i-\alpha_i^*)=0;\ \ 0\le\alpha_i,\alpha_i^*\le C$$

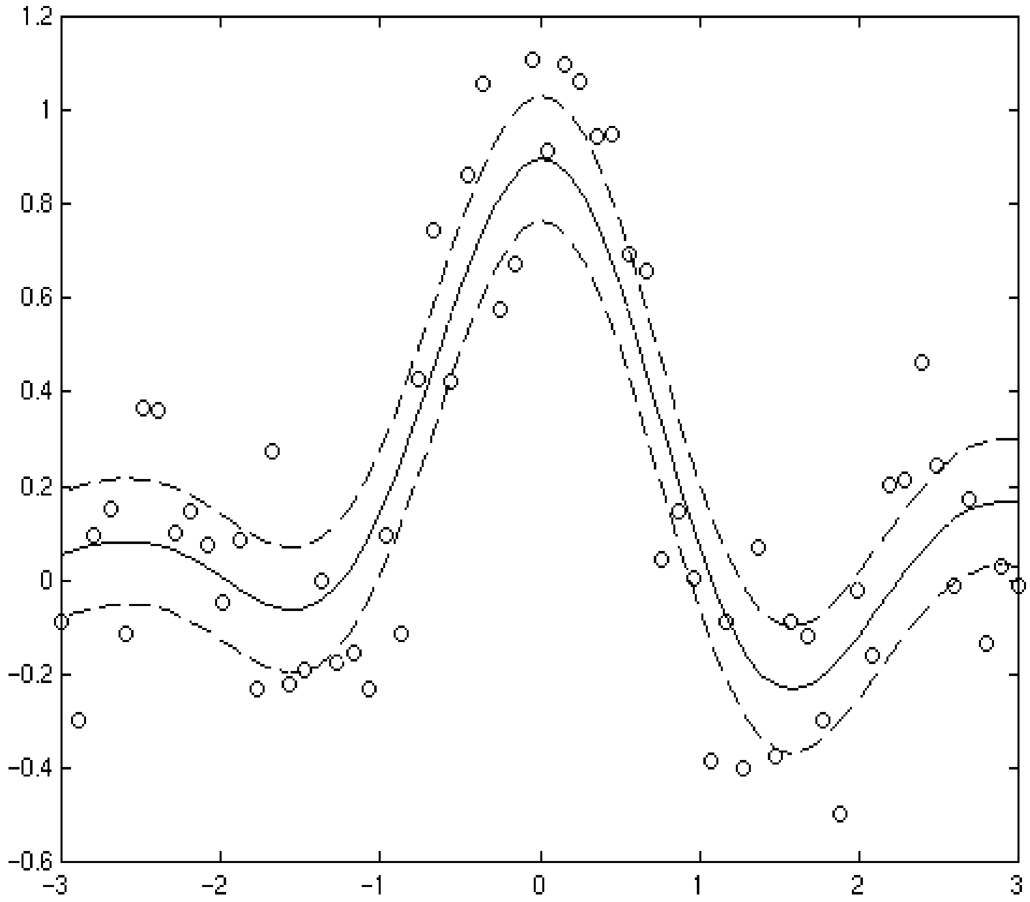
Primal variables: w for each feature dim	Dual variables: α, α* for each data point
Complexity: the dim of the input space	Complexity: Number of support vectors



KKT condition

$$\mathbf{w} = \sum_{i=1}^{m}(\alpha_i-\alpha_i^*)\mathbf{x}_i$$

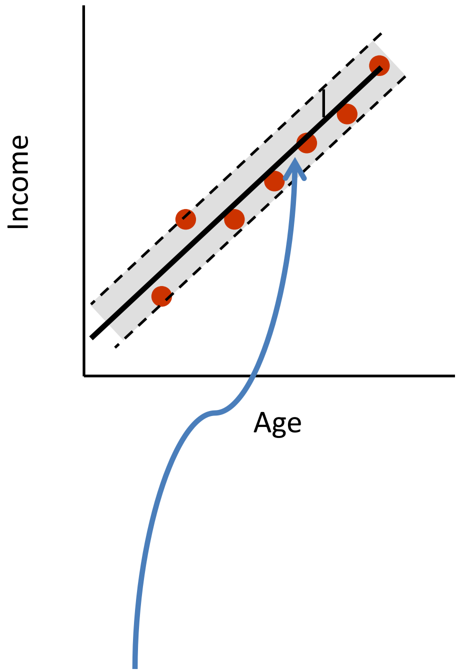
How about a non-linear case?



Linear versus Non-linear SVR

- **Linear case**

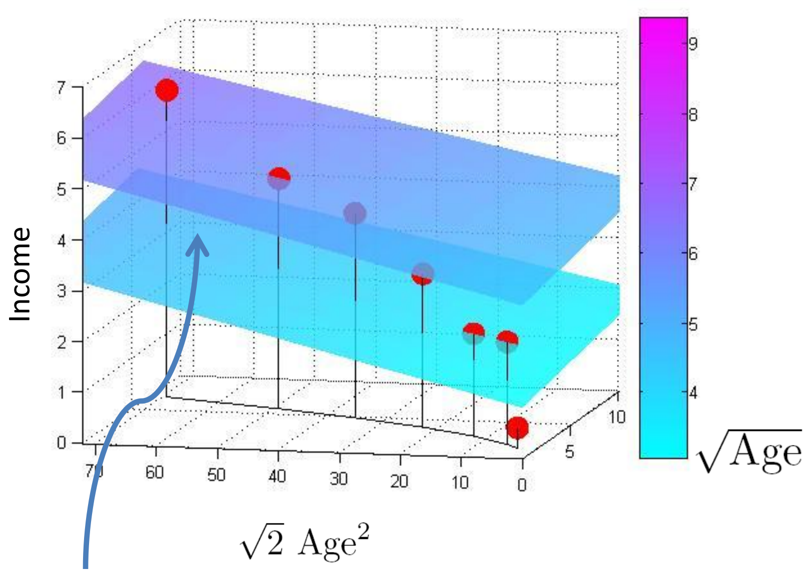
$$f : age \rightarrow income$$



$$y_i = \mathbf{w}_1 \cdot \mathbf{x}_i + b$$

- **Non-linear case**
  - Map data into a higher dimensional space, e.g.,

$$f : (\sqrt{age}, \sqrt{2}age^2) \rightarrow income$$



$$y_i = \mathbf{w}_1 \sqrt{\mathbf{x}_i} + \mathbf{w}_2 \sqrt{2}\mathbf{x}_i^2 + b$$

Dual problem

- Primal

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*)$$

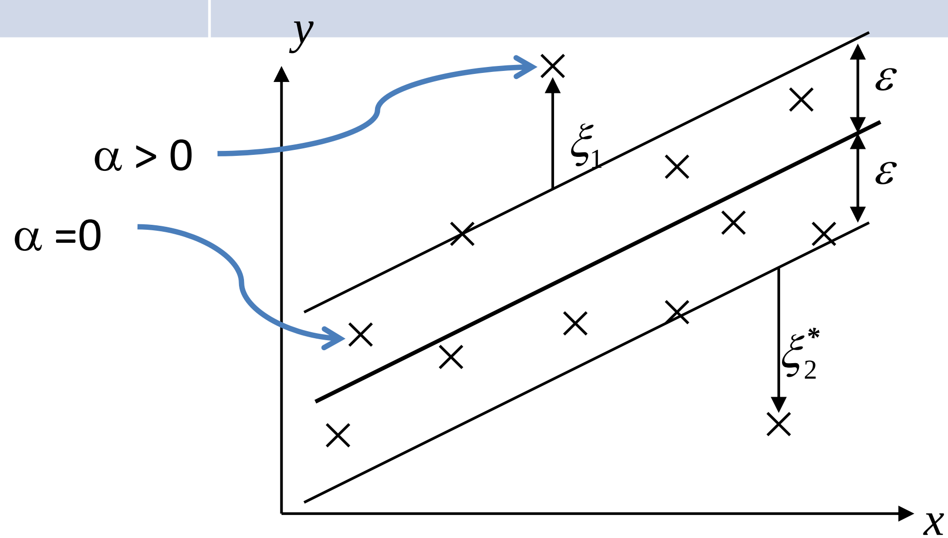
$$s.t.\begin{cases} y_i - (\mathbf{w}\cdot\mathbf{x}_i) - b \le \varepsilon + \xi_i \\ (\mathbf{w}\cdot\mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i\ , \xi_i^* \ge 0, i = 1,\ldots,m \end{cases}$$

- Dual

$$\max\begin{cases} \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j\rangle \\ -\varepsilon\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i(\alpha_i - \alpha_i^*) \end{cases}$$

$$s.t.\sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0;\ \ 0 \le \alpha_i, \alpha_i^* \le C$$

Primal variables: w for each feature dim	Dual variables: α, α* for each data point
Complexity: the dim of the input space	Complexity: Number of support vectors



Kernel trick

- Linear: $\langle x, y \rangle$
- Non-linear: $\langle \varphi(x), \varphi(y) \rangle = K(x, y)$

Note: No need to compute the mapping function, φ(.), explicitly. Instead, we use the kernel function.

Commonly used kernels:

- Polynomial kernels: $K(x, y) = (x^T y + 1)^d$

- Radial basis function (RBF) kernels:

$$K(x, y) = \exp(-\frac{1}{2\sigma^2} \| x - y \|^2)$$

Note: for RBF kernel, dim(φ(.)) is infinite

Dual problem for non-linear case

- Primal

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*)$$

$$s.t.\begin{cases} y_i - (\mathbf{w}\cdot\varphi(\mathbf{x}_i)) - b \le \varepsilon + \xi_i \\ (\mathbf{w}\cdot\varphi(\mathbf{x}_i)) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i \;, \xi_i^* \ge 0, i = 1,\dots, m \end{cases}$$

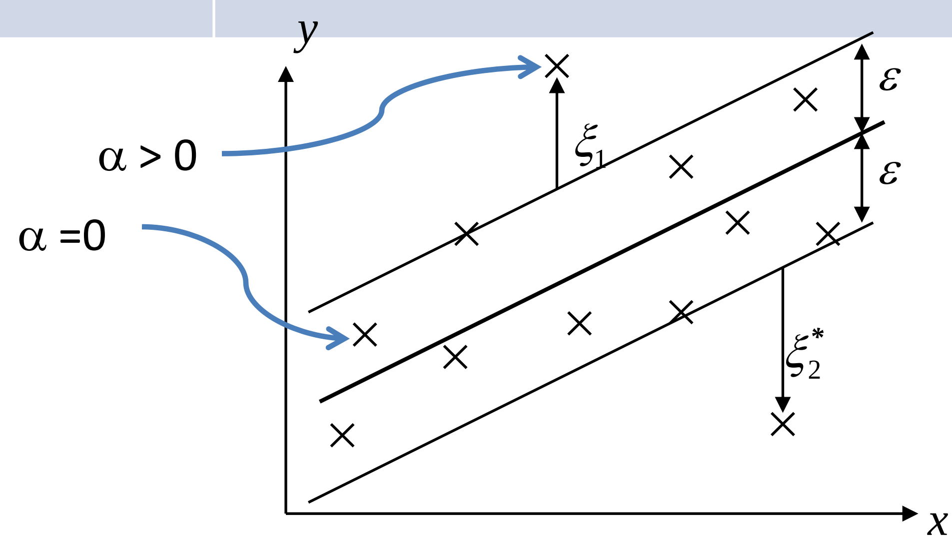
- Dual

K(xi, xj)

$$\max\begin{cases} \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle\varphi(\mathbf{x}_i),\varphi(\mathbf{x}_j)\rangle \\ -\varepsilon\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i(\alpha_i - \alpha_i^*) \end{cases}$$

$$s.t.\sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0;\;\; 0 \le \alpha_i, \alpha_i^* \le C$$

Primal variables: w for each feature dim	Dual variables: α, α* for each data point
Complexity: the dim of the input space	Complexity: Number of support vectors



How to choose SVR parameters?

$$\max \begin{cases} \frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\mathbf{x}_i, \mathbf{x}_j) \\ -\varepsilon\sum_{i=1}^{m}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i(\alpha_i - \alpha_i^*) \end{cases}$$

$$s.t. \sum_{i=1}^{m}(\alpha_i - \alpha_i^*) = 0;\ \ 0 \le \alpha_i, \alpha_i^* \le C$$

Cross-validation

Trade off parameter: C

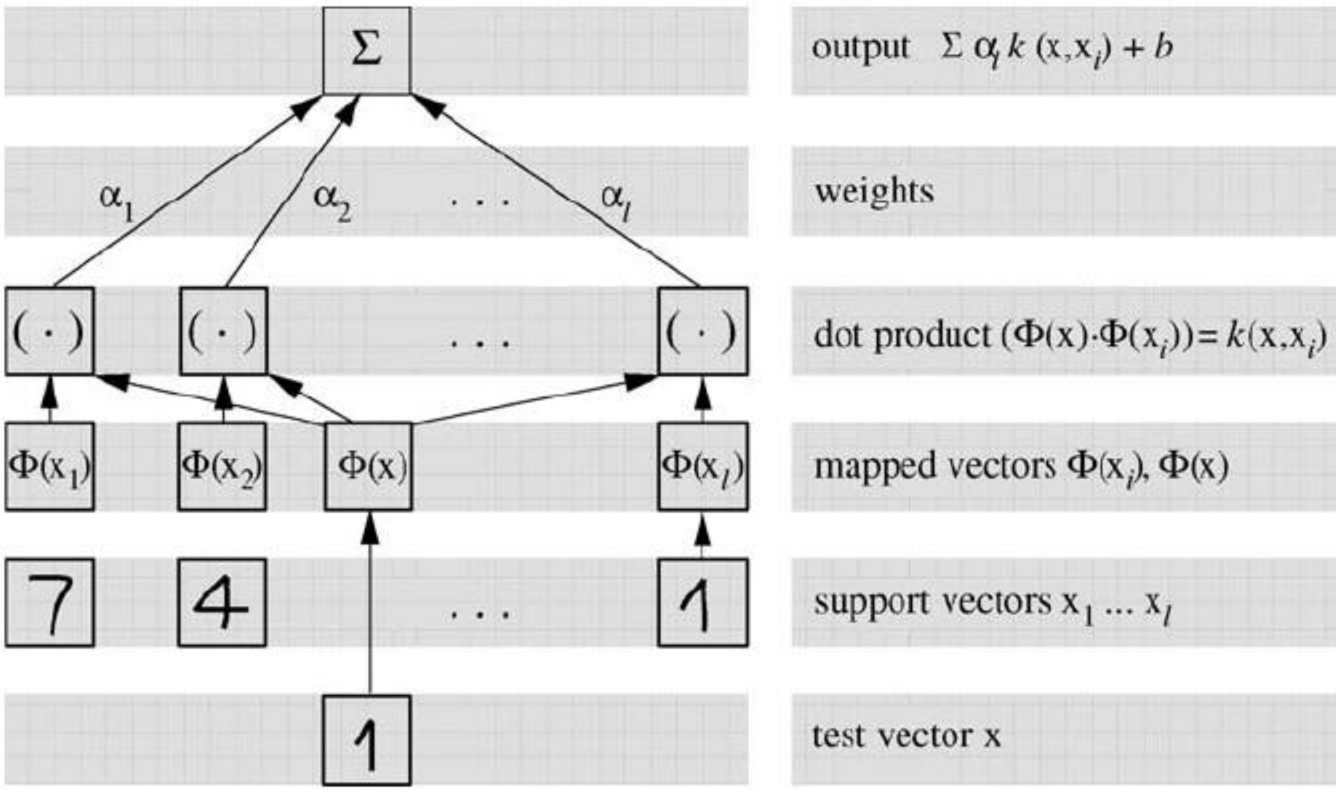
Kernel parameter: d (Polynomial) and σ (RBF)

- Polynomial kernels: $K(x, y) = (x^T y + 1)^d$
- Radial basis function (RBF) kernels:

$$K(x, y) = \exp(-\frac{1}{2\sigma^2}\| x - y \|^2)$$

SVR Applications

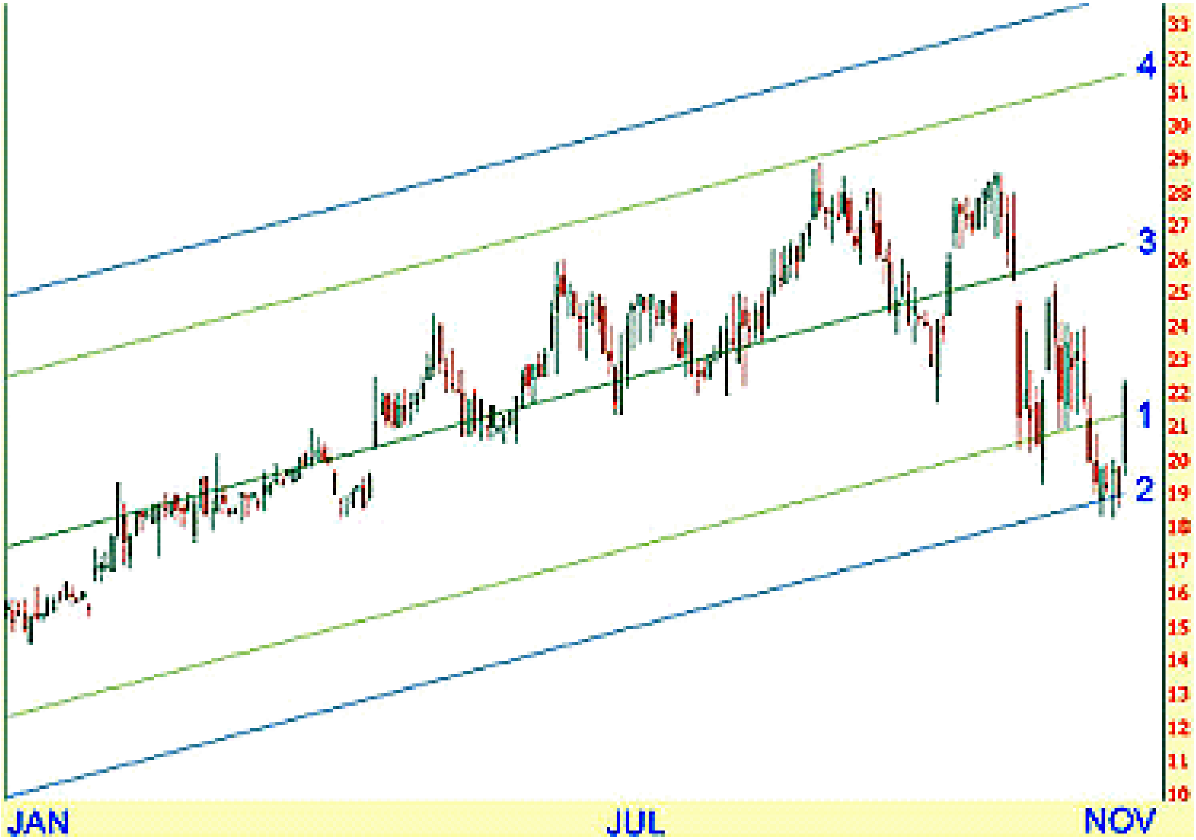
Optical Character Recognition (OCR)



A. J. Smola and B. Scholkopf, A Tutorial on Support Vector Regression, NeuroCOLT Technical Report TR-98-030

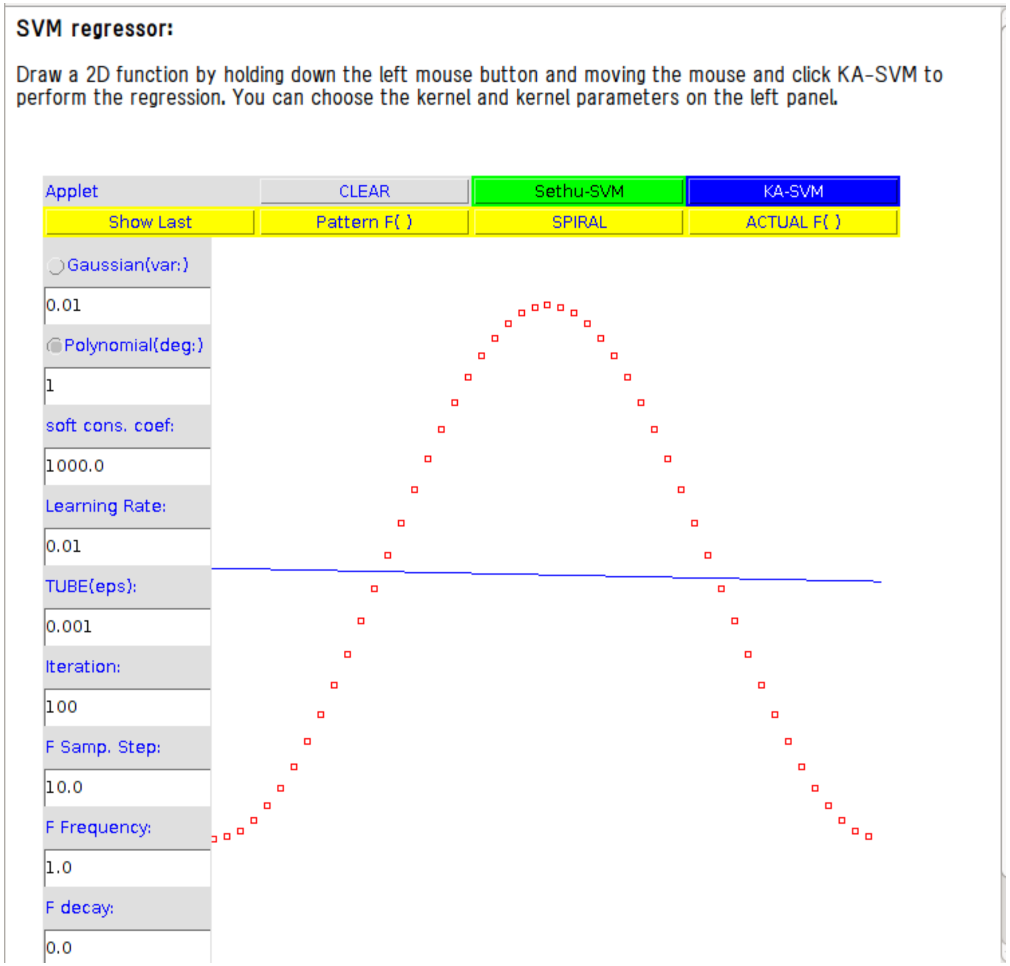
SVR Applications

- Stock price prediction



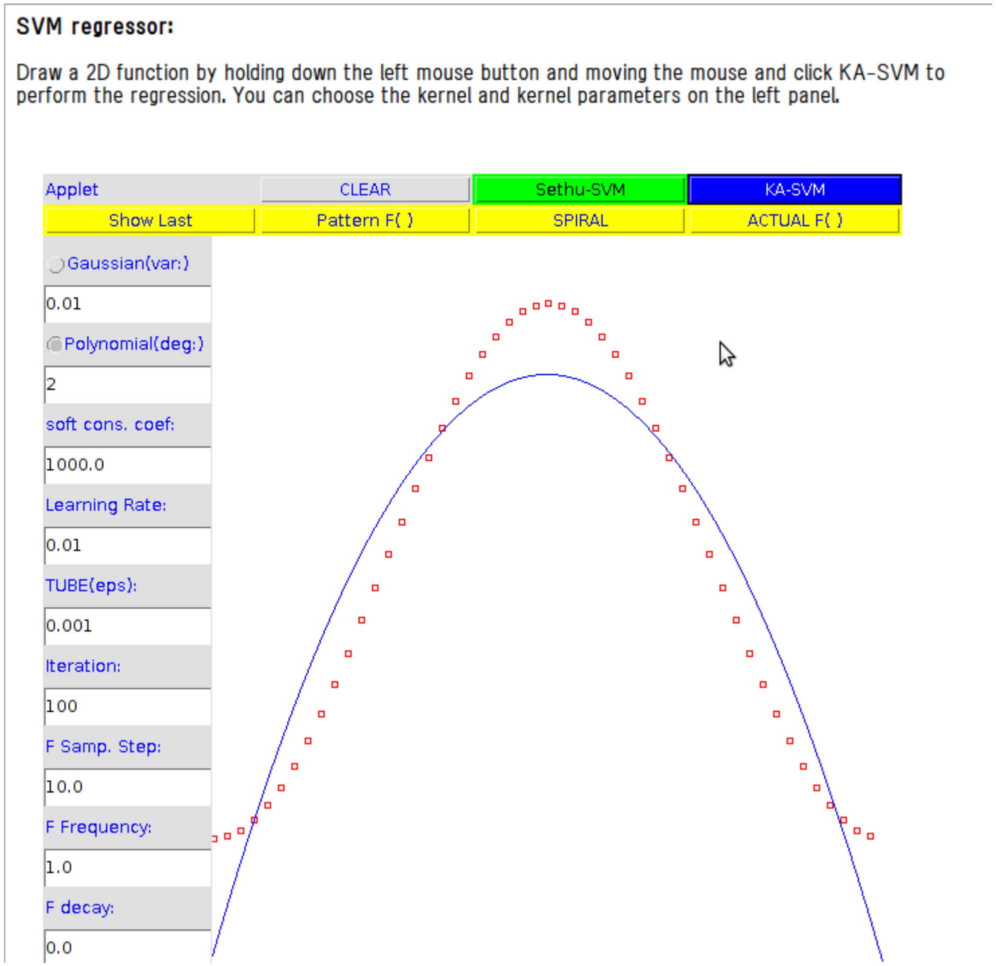
SVR Demo (Linear) :

http://www.cns.atr.jp/~erhan/SVMreg/SVM.html



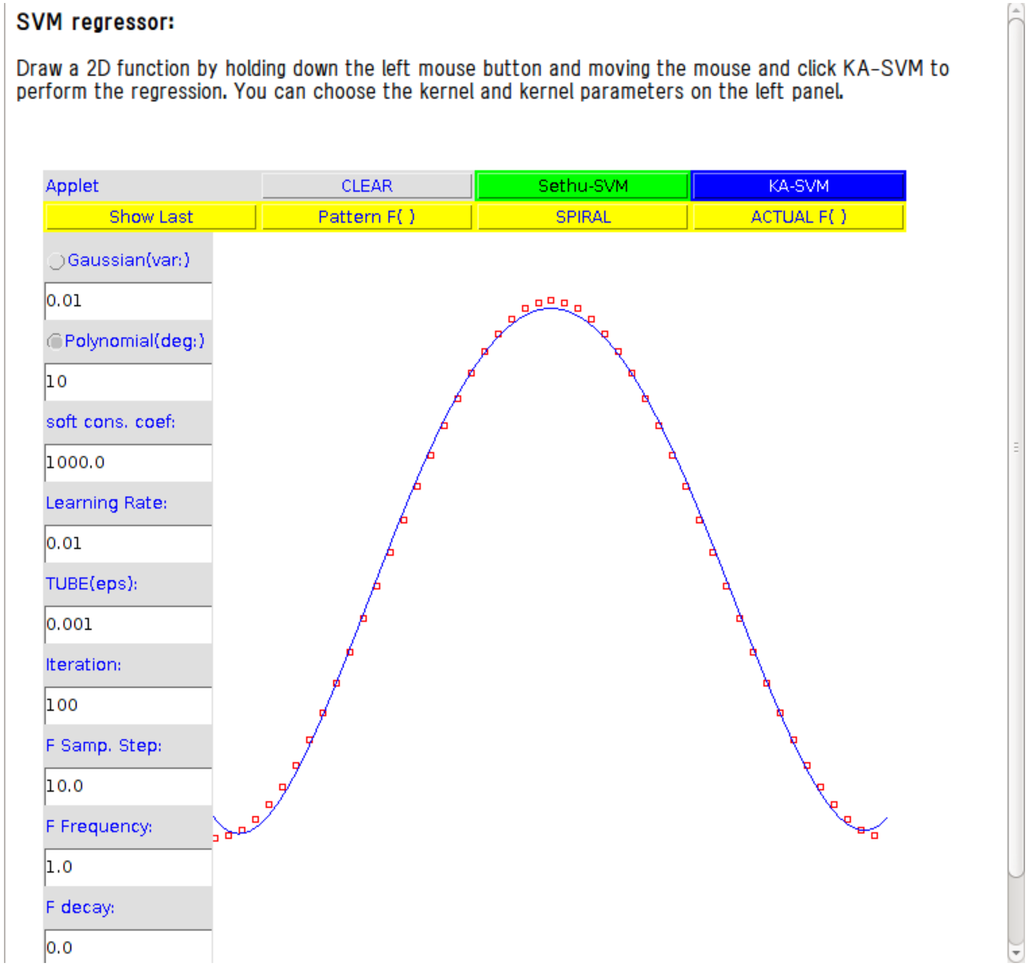
SVR Demo (Polynomial degree 2):

http://www.cns.atr.jp/~erhan/SVMreg/SVM.html



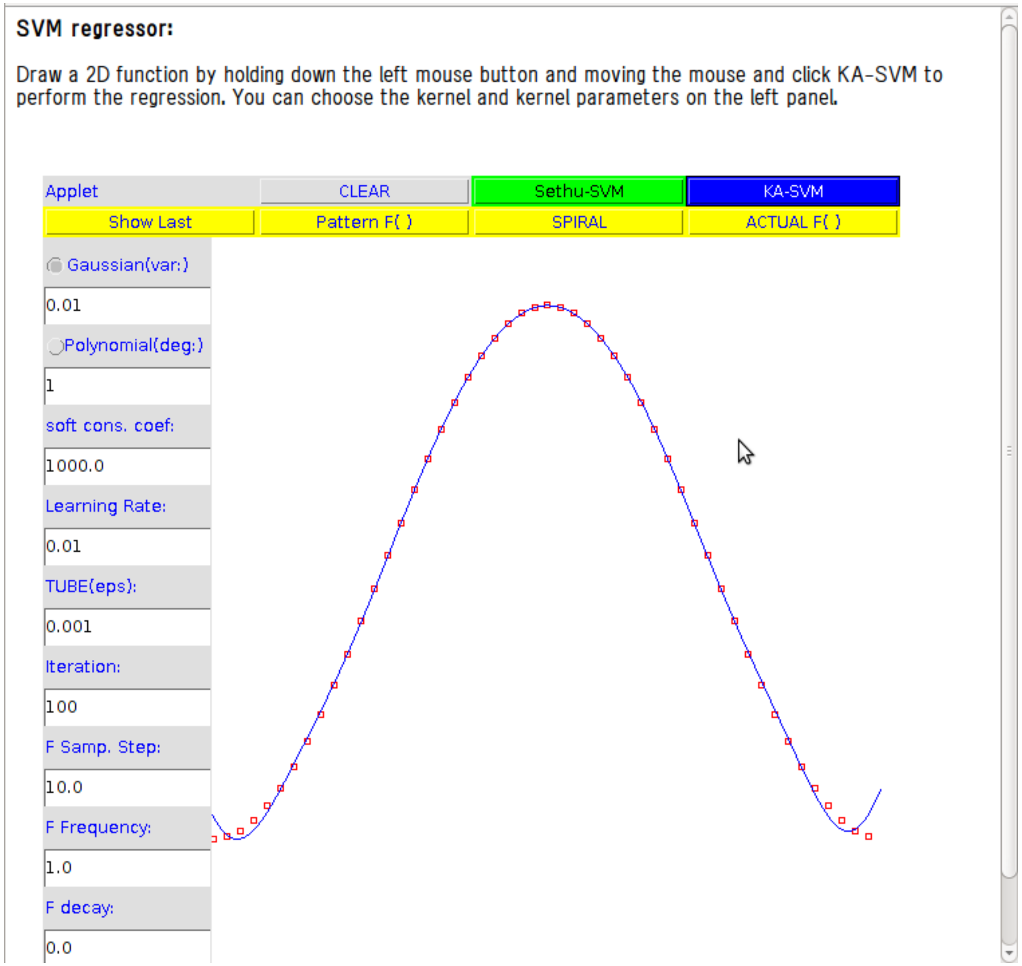
SVR Demo (Polynomial degree 10):

http://www.cns.atr.jp/~erhan/SVMreg/SVM.html



SVR Demo (RBF):

http://www.cns.atr.jp/~erhan/SVMreg/SVM.html



WEKA and linear regression

- Software can be downloaded from http://www.cs.waikato.ac.nz/ml/weka/
- Data set used in this experiment: Computer hardware
- The objective is to predict CPU performance based on these given attributes:
  - Machine cycle time in nanoseconds (MYCT)
  - Minimum main memory in kilobytes (MMIN)
  - Maximum main memory (MMAX)
  - Cache memory in kilobytes (CACH)
  - Minimum channels in units (CHMIN)
  - Maximum channels in units (CHMAX)
- Output is expressed as a linear combination of the attributes. Each attribute has a specific weight.
  - Output = $w_1 a_1 + w_2 a_2 + ... + w_n a_n + b$

Evaluation

- Root mean-square error

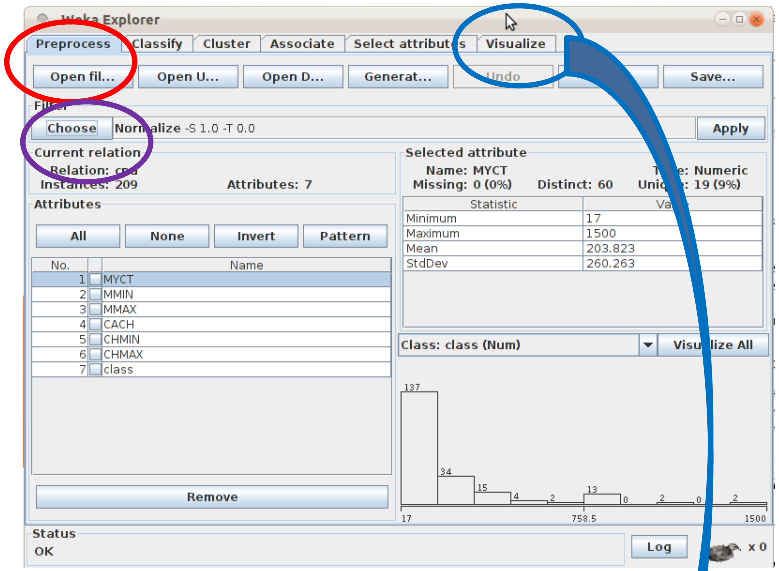
$$\sqrt{\frac{(y_1-\hat{y}_1)^2+(y_2-\hat{y}_2)^2+\ldots+(y_m-\hat{y}_m)^2}{n}}$$

- Mean absolute error

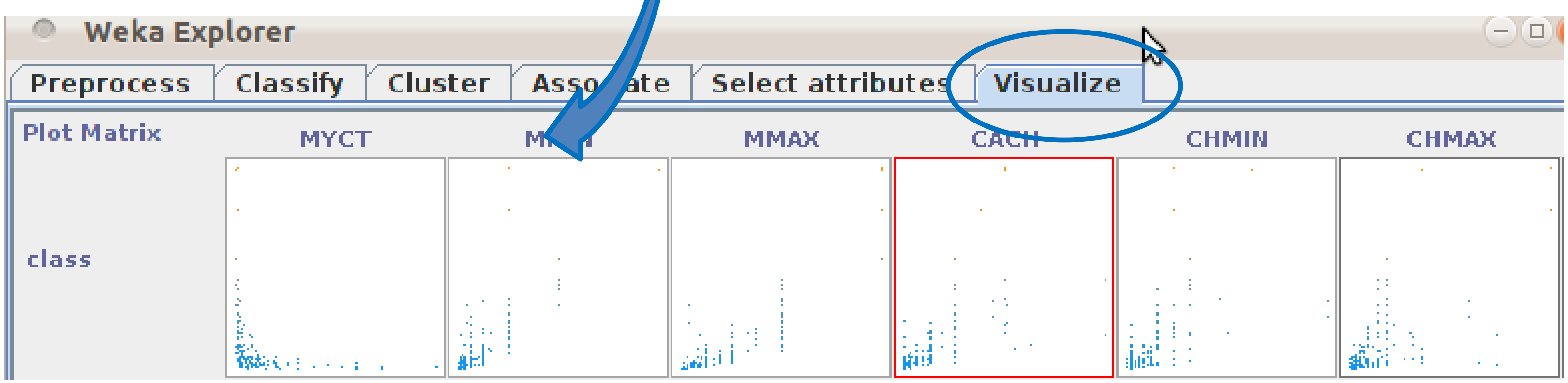
$$\frac{|y_1-\hat{y}_1|+|y_2-\hat{y}_2|+\ldots+|y_m-\hat{y}_m|}{n}$$

WEKA

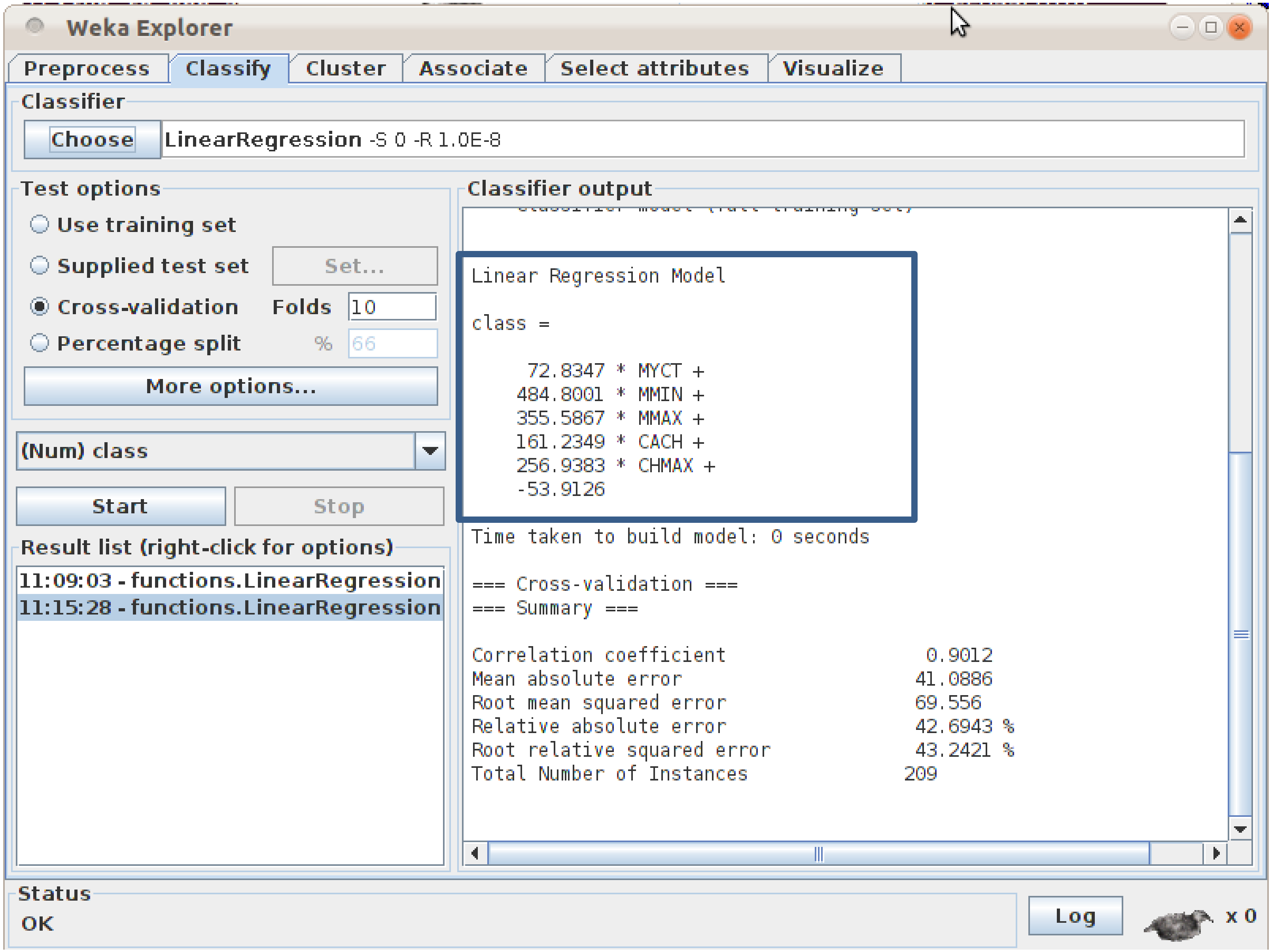
Load data and normalize each attribute to [0, 1]



Data visualization

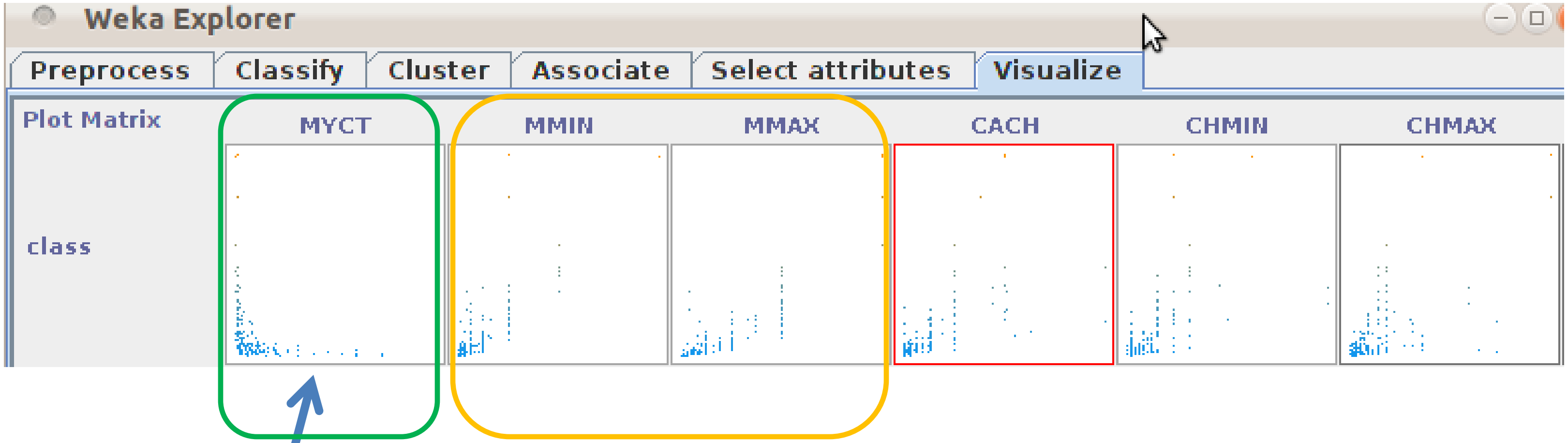


WEKA (Linear regression)



WEKA (Linear Regression)

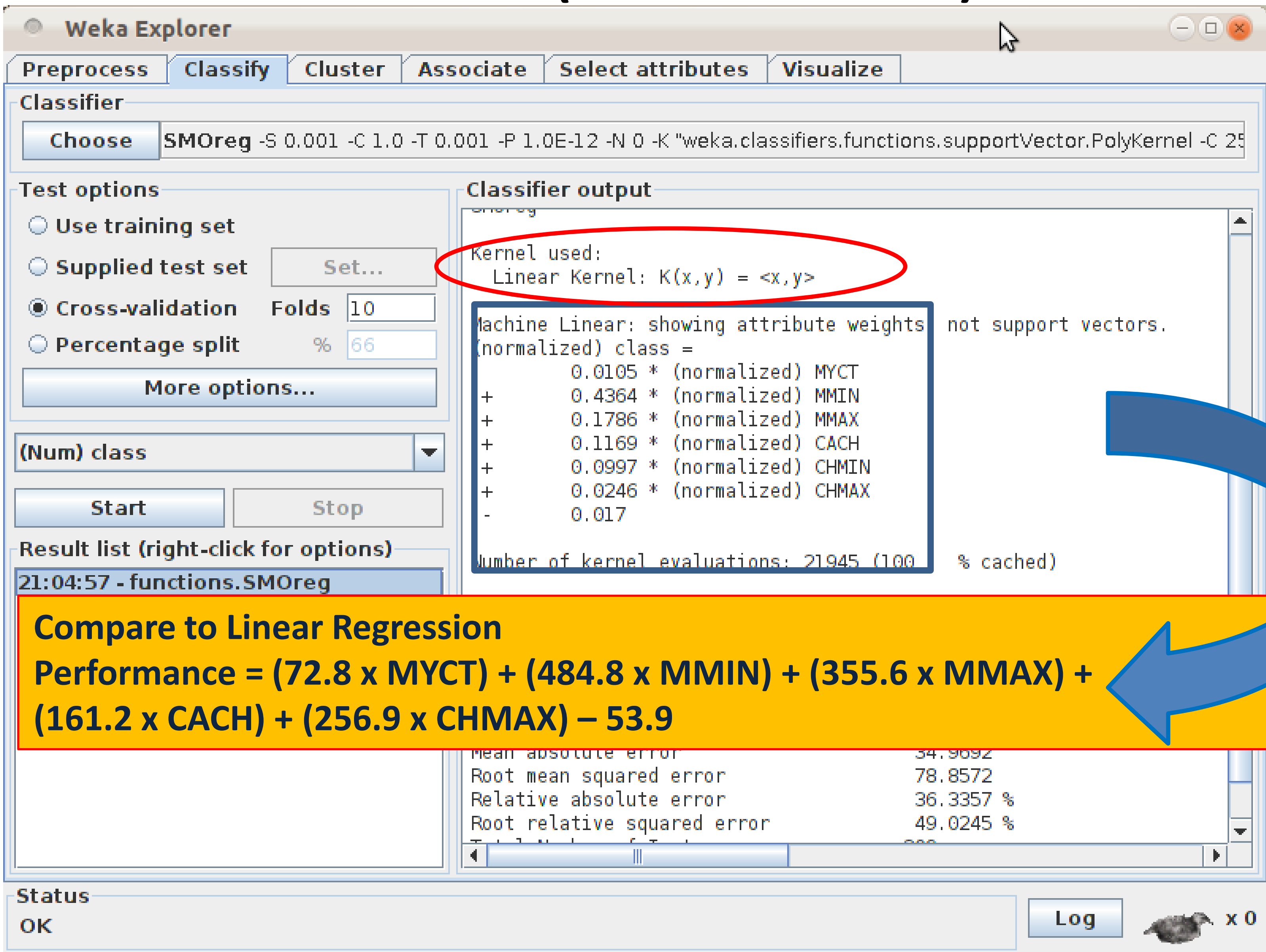
Performance = (72.8 x MYCT) + (484.8 x MMIN) + (355.6 x MMAX) + (161.2 x CACH) + (256.9 x CHMAX) – 53.9



Main memory plays a more important role in the system performance

Large Machine cycle time (MYCT) does not indicate the best performance

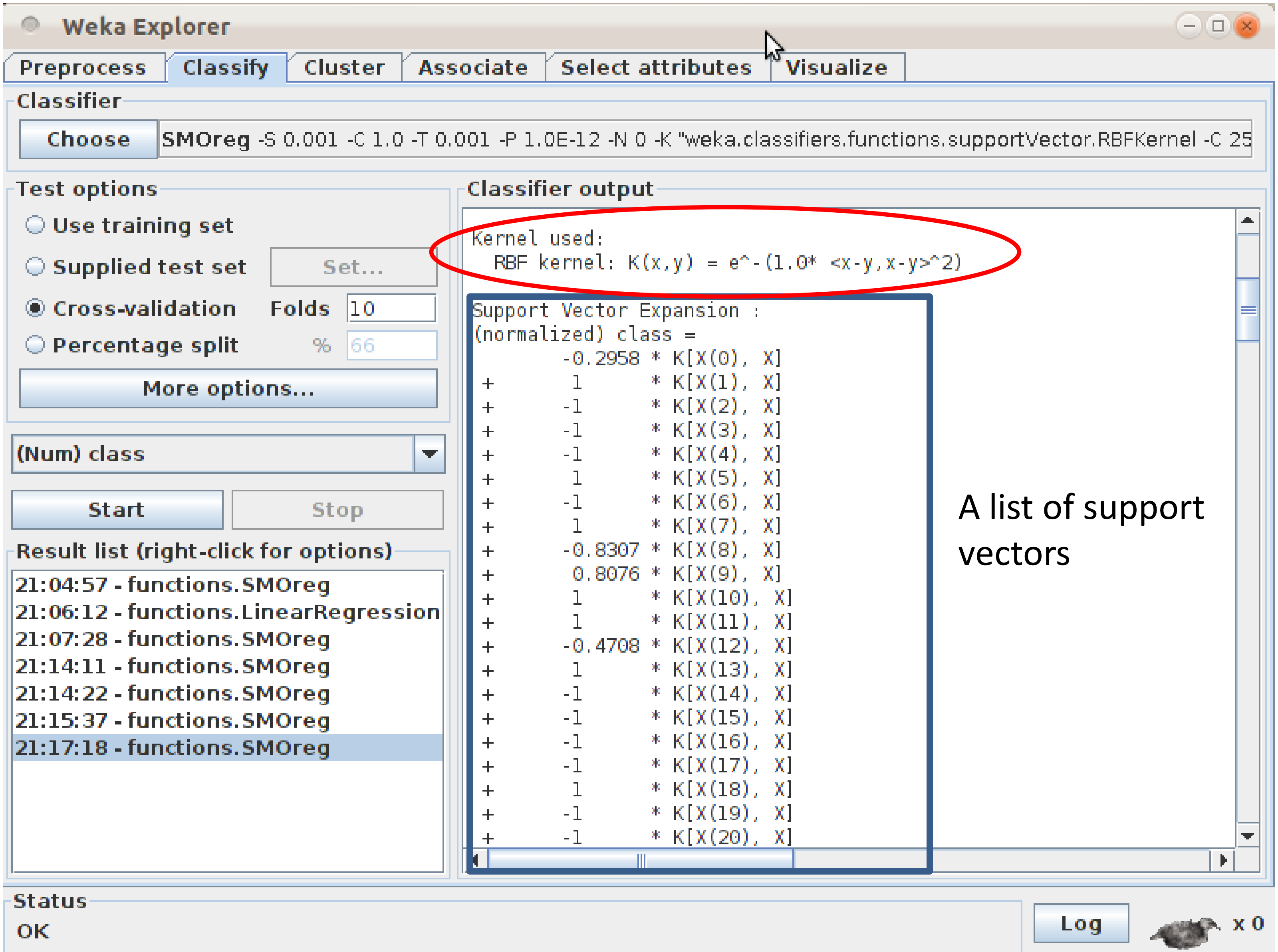
WEKA (linear SVR)



Compare to Linear Regression

Performance = (72.8 x MYCT) + (484.8 x MMIN) + (355.6 x MMAX) + (161.2 x CACH) + (256.9 x CHMAX) – 53.9

WEKA (non-linear SVR)



WEKA (Performance comparison)

Method	Mean absolute error	Root mean squared error
Linear regression	41.1	69.55
SVR (Linear) C = 1.0	35.0	78.8
SVR (RBF) C = 1.0, gamma = 1.0	28.8	66.3

Parameter C (for linear SVR) and <C,γ> (for non-linear SVR) need to be cross-validated for a better performance.

Other Machine Learning tools

- Shogun toolbox (C++)
  - http://www.shogun-toolbox.org/
- Shark Machine Learning library (C++)
  - http://shark-project.sourceforge.net/
- Machine Learning in Python (Python)
  - http://pyml.sourceforge.net/
- Machine Learning in Open CV2
  - http://opencv.willowgarage.com/wiki/
- LibSVM, LibLinear, etc.