

Linear Regression and Support Vector Regression

Paul Paisitkriangkrai

paulp@cs.adelaide.edu.au

The University of Adelaide

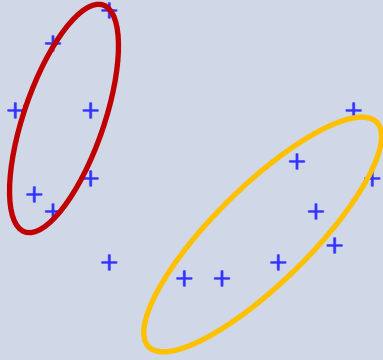
24 October 2012

Outlines

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

Regression Overview

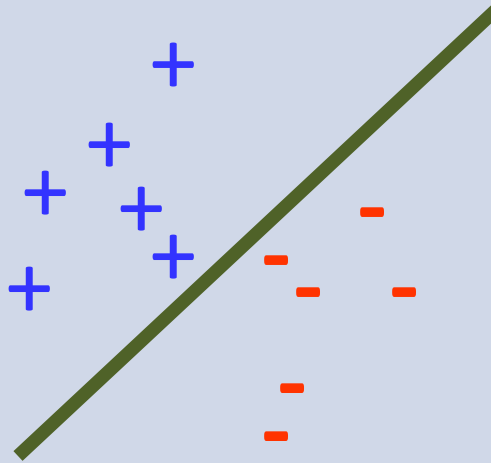
CLUSTERING



K-means

Group data based on their characteristics

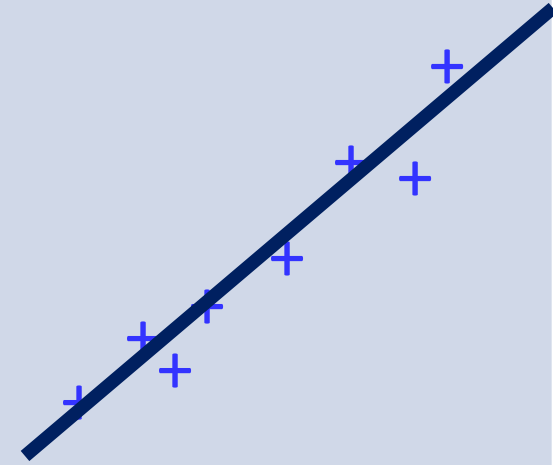
CLASSIFICATION



- Decision tree
- Linear Discriminant Analysis
- Neural Networks
- Support Vector Machines
- Boosting

Separate data based on their labels

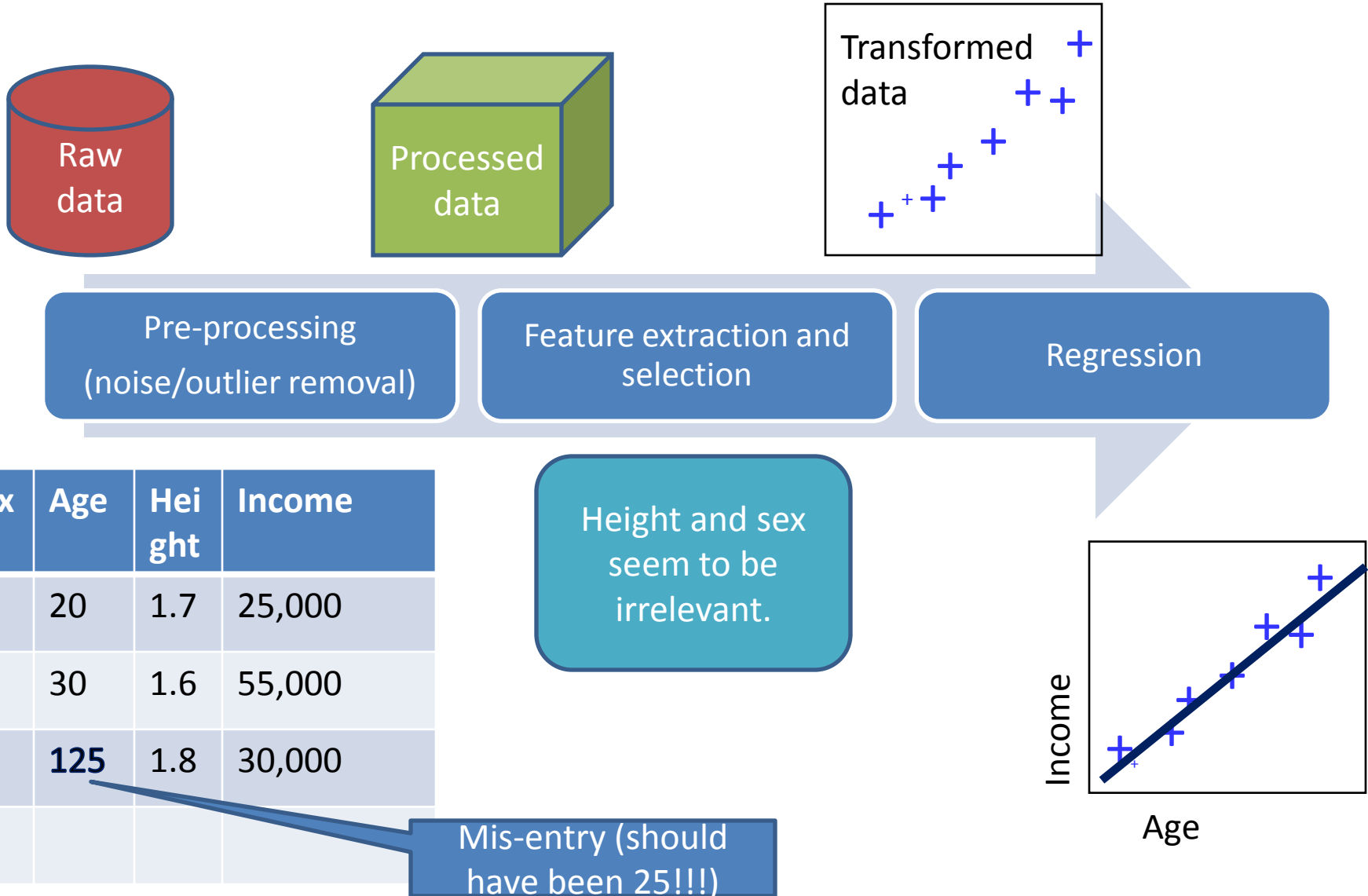
REGRESSION (THIS TALK)



- Linear Regression
- Support Vector Regression

Find a model that can explain the output given the input

Data processing flowchart (Income prediction)



Linear Regression

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

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

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

- The objective: Find a function f that returns the best fit. $f : \mathbb{R}^n \rightarrow \mathbb{R}$
- 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)

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

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

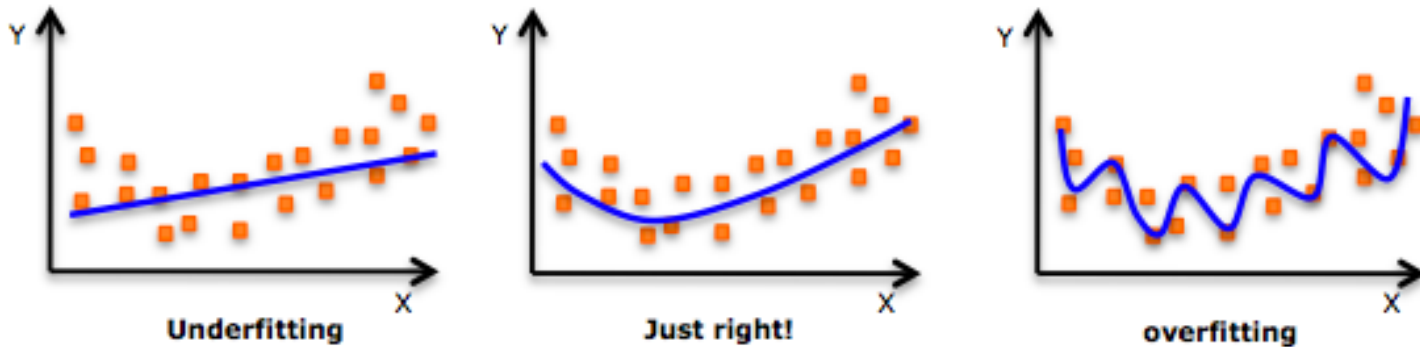
$$\hat{\mathbf{w}} = (X^T X)^{-1} X^T Y$$

(By taking the derivative of the above objective function w.r.t. \mathbf{w})

- 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 avoid 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

$$\min \frac{1}{2} \|\mathbf{w}\|^2$$

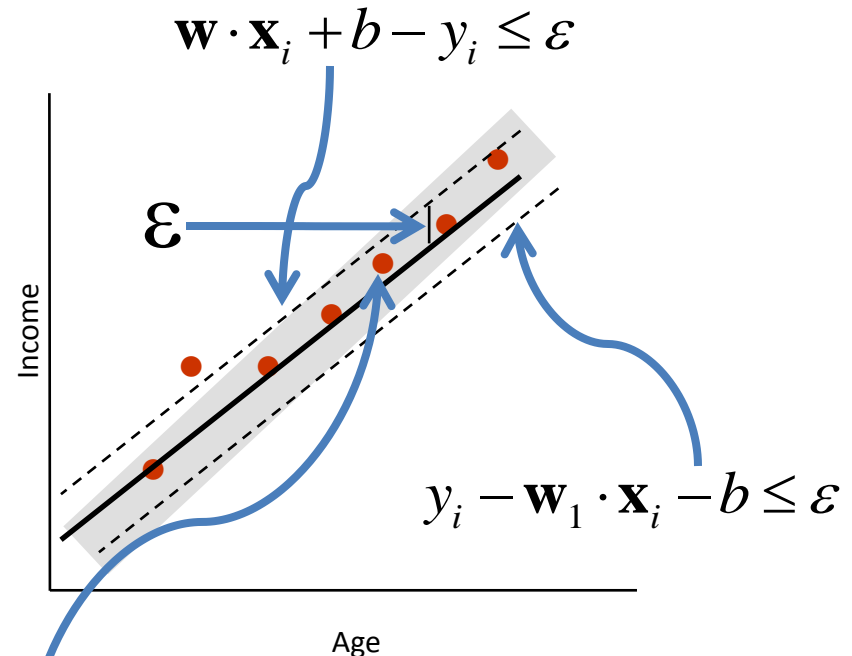
$$s.t. \ y_i - \mathbf{w}_1 \cdot \mathbf{x}_i - b \leq \varepsilon;$$

$$\mathbf{w}_1 \cdot \mathbf{x}_i + b - y_i \leq \varepsilon;$$

C: trade off the complexity

What if the problem is not feasible?

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

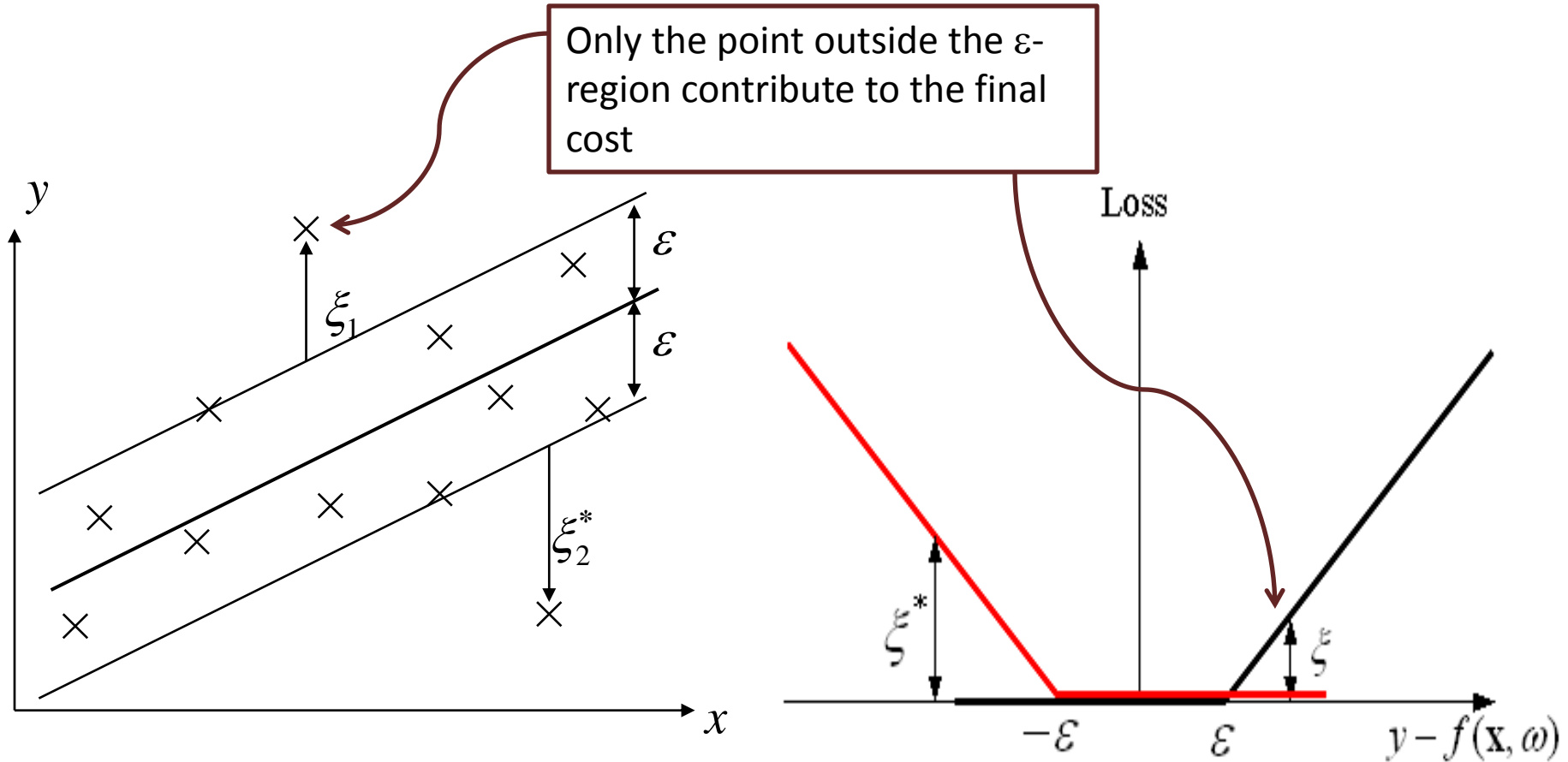


We do not care about errors as long as they are less than ε

Support Vector Regression

Assume linear parameterization

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



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

Soft margin

Given training data

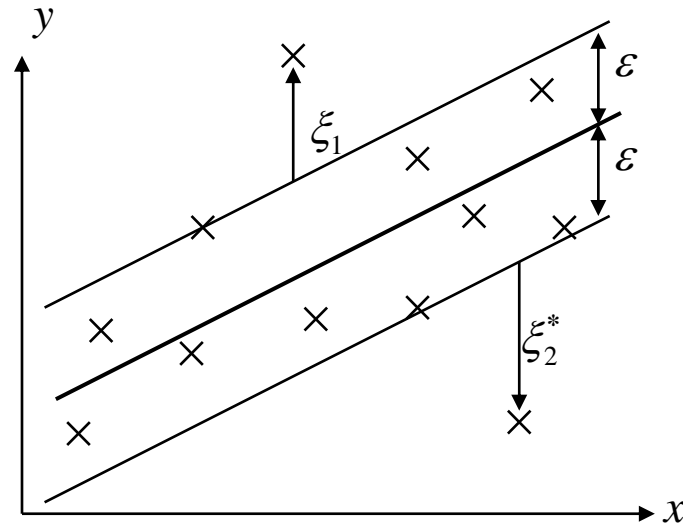
$$(\mathbf{x}_i, y_i) \quad i = 1, \dots, m$$

Minimize

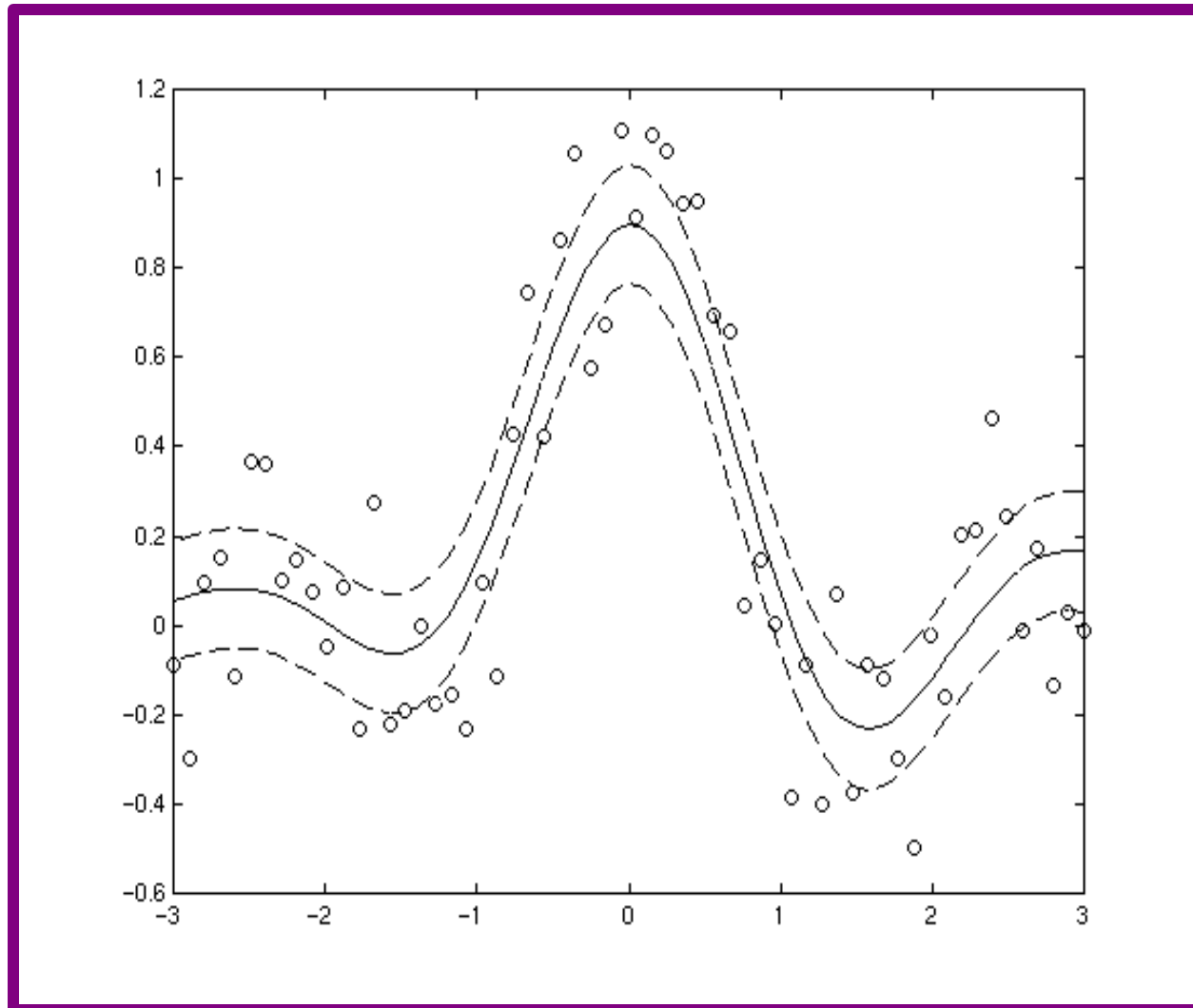
$$\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m (\xi_i + \xi_i^*)$$

Under constraints

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



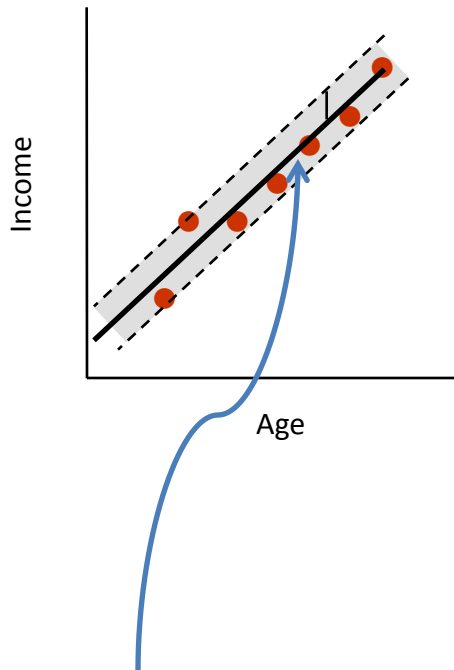
How about a non-linear case?



Linear versus Non-linear SVR

- Linear case

$$f : \text{age} \rightarrow \text{income}$$

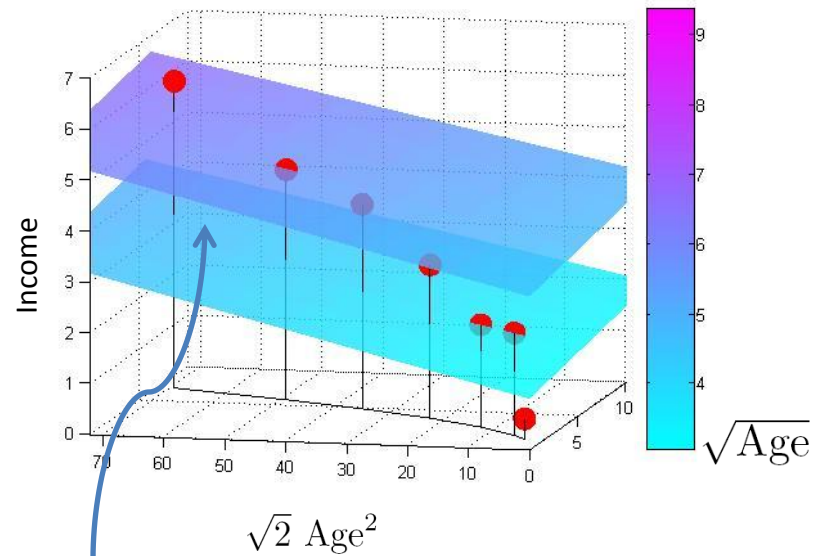


$$y_i = \mathbf{w}_1 \cdot \mathbf{x}_i + b$$

- Non-linear case

- Map data into a higher dimensional space, e.g.,

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



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

Dual problem

- Primal

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

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

- Dual

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

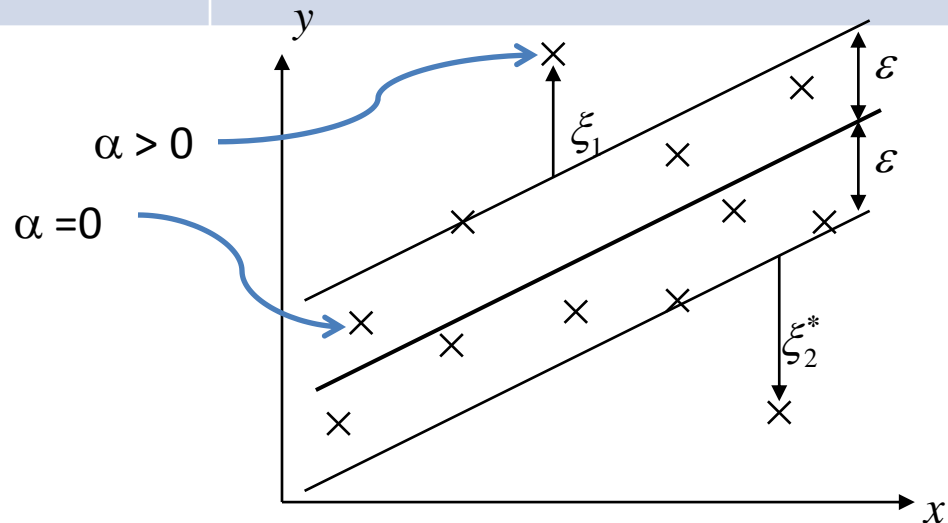
$$s.t. \sum_{i=1}^m (\alpha_i - \alpha_i^*) = 0; \quad 0 \leq \alpha_i, \alpha_i^* \leq C$$

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, $\varphi(\cdot)$, 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:

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

Note: for RBF kernel, $\dim(\varphi(\cdot))$ is infinite

Dual problem for non-linear case

- Primal

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

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

- Dual

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

$K(\mathbf{x}_i, \mathbf{x}_j)$

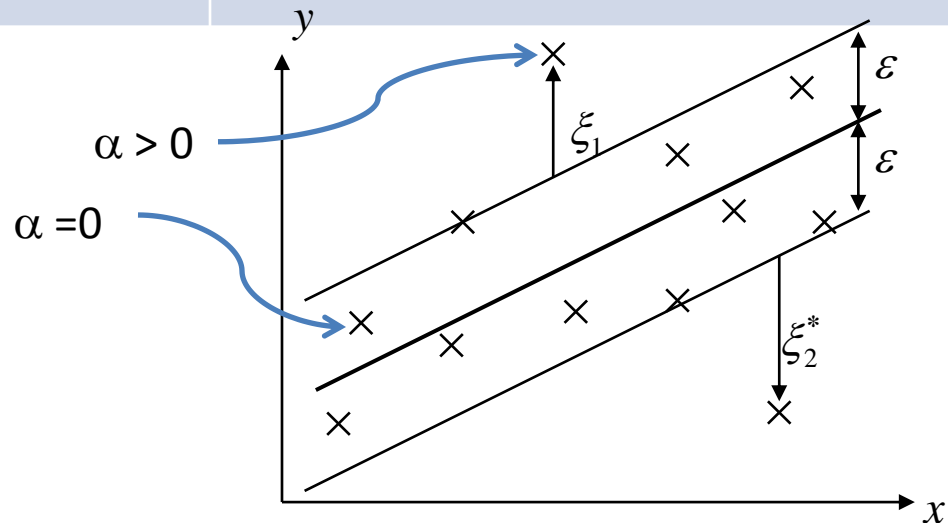
$$s.t. \sum_{i=1}^m (\alpha_i - \alpha_i^*) = 0; 0 \leq \alpha_i, \alpha_i^* \leq C$$

Primal variables: \mathbf{w} for each feature dim

Dual variables: α, α^* for each data point

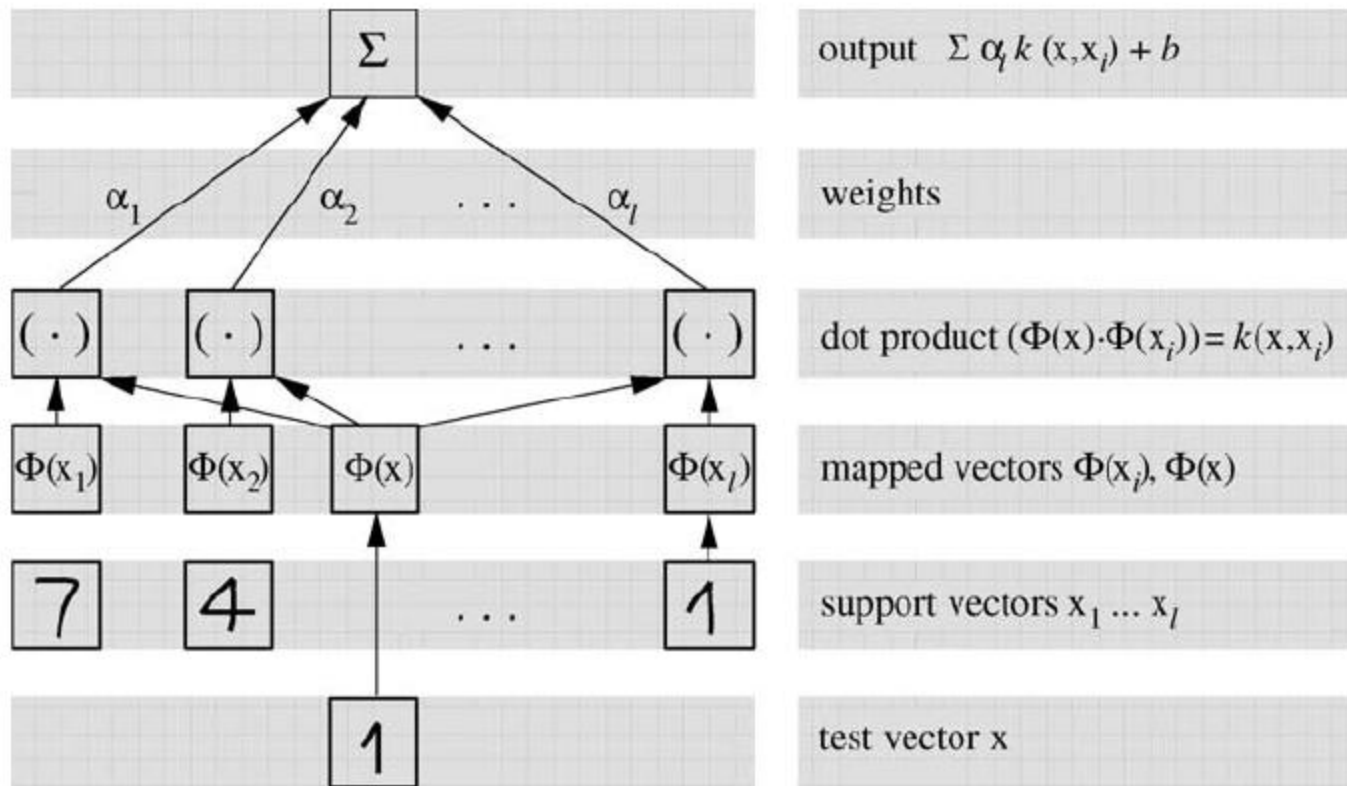
Complexity: the dim of the input space

Complexity: Number of support vectors



SVR Applications

Optical Character Recognition (OCR)



SVR Applications

- Stock price prediction



SVR Demo

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 = $w_1 a_1 + w_2 a_2 + \dots + w_n a_n + b$

Evaluation

- Root mean-square error

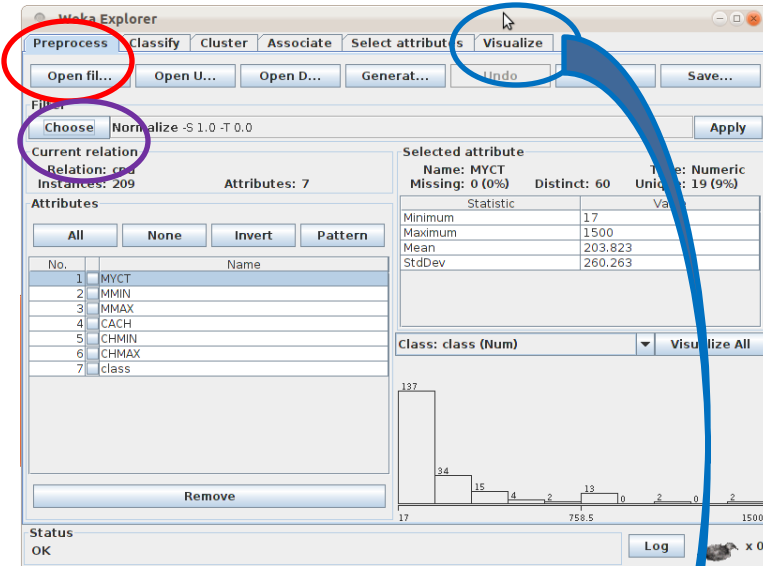
$$\sqrt{\frac{(y_1 - \hat{y}_1)^2 + (y_2 - \hat{y}_2)^2 + \dots + (y_m - \hat{y}_m)^2}{n}}$$

- Mean absolute error

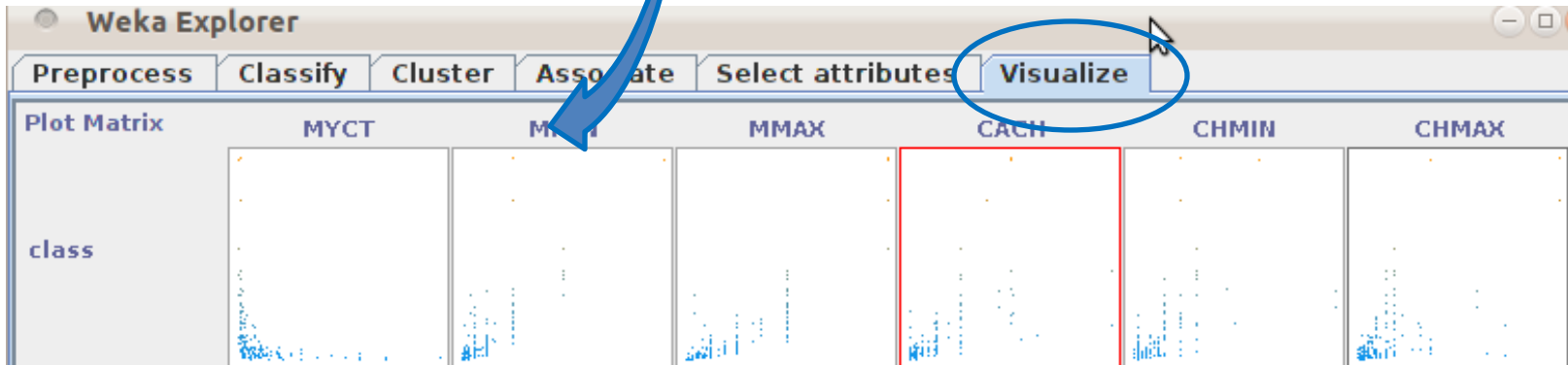
$$\frac{|y_1 - \hat{y}_1| + |y_2 - \hat{y}_2| + \dots + |y_m - \hat{y}_m|}{n}$$

WEKA

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



Data visualization



WEKA (Linear regression)

The screenshot shows the Weka Explorer application window. The 'Classify' tab is selected. The classifier is set to 'LinearRegression' with parameters '-S 0 -R 1.0E-8'. The 'Test options' section shows 'Cross-validation' selected with 10 folds and 66% split. The 'Classifier output' pane displays the following text:

```
Linear Regression Model
class =
    72.8347 * MYCT +
    484.8001 * MMIN +
    355.5867 * MMAX +
    161.2349 * CACH +
    256.9383 * CHMAX +
    -53.9126
```

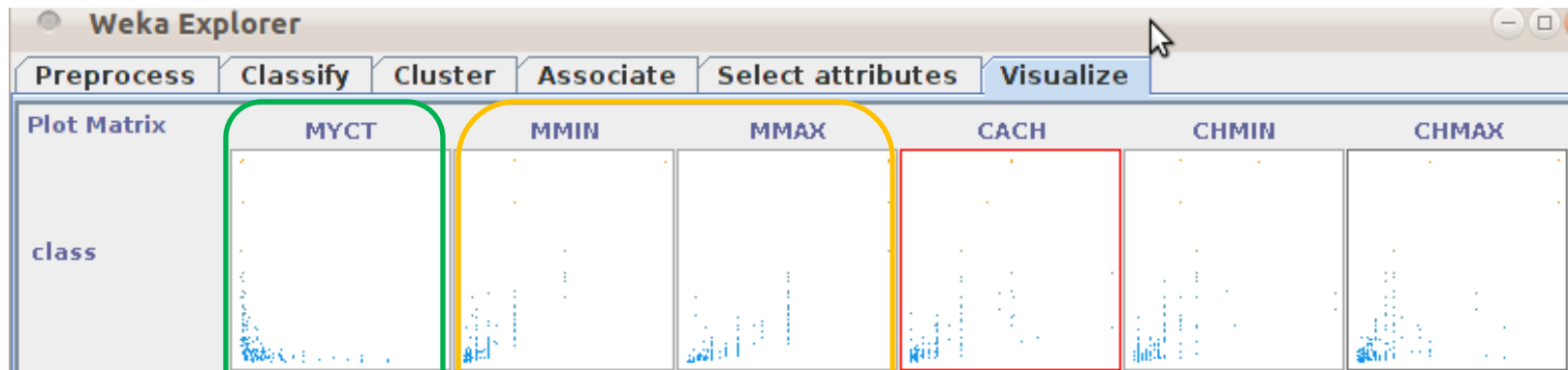
Below the model equation, the output shows 'Time taken to build model: 0 seconds', '=== Cross-validation ===', and '=== Summary ==='. The summary table is as follows:

Correlation coefficient	0.9012
Mean absolute error	41.0886
Root mean squared error	69.556
Relative absolute error	42.6943 %
Root relative squared error	43.2421 %
Total Number of Instances	209

The 'Result list' shows two entries: '11:09:03 - functions.LinearRegression' and '11:15:28 - functions.LinearRegression'. The status bar at the bottom indicates 'Status OK' and a 'Log' button.

WEKA (Linear Regression)

$$\text{Performance} = (72.8 \times \text{MYCT}) + (484.8 \times \text{MMIN}) + (355.6 \times \text{MMAX}) + (161.2 \times \text{CACH}) + (256.9 \times \text{CHMAX}) - 53.9$$



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)

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier is set to 'SMOreg' with the following command: `-S 0.001 -C 1.0 -T 0.001 -P 1.0E-12 -N 0 -K "weka.classifiers.functions.supportVector.PolyKernel -C 25"`. The 'Test options' section has 'Cross-validation' selected with 'Folds' set to 10 and 'Percentage split' at 66%. The 'Classifier output' window shows the kernel used: 'Linear Kernel: $K(x,y) = \langle x,y \rangle$ '. Below this, the machine linear model is displayed as a list of normalized weights for classes MYCT, MMIN, MMAX, CACH, CHMIN, and CHMAX. A blue box highlights this list, and a blue arrow points from it to a yellow box containing a regression equation. The 'Result list' shows the current run as '21:04:57 - functions.SMOreg'. The status bar at the bottom indicates 'OK'.

Kernel used:
Linear Kernel: $K(x,y) = \langle x,y \rangle$

Machine Linear: showing attribute weights (not support vectors).
(normalized) class =

- 0.0105 * (normalized) MYCT
- + 0.4364 * (normalized) MMIN
- + 0.1786 * (normalized) MMAX
- + 0.1169 * (normalized) CACH
- + 0.0997 * (normalized) CHMIN
- + 0.0246 * (normalized) CHMAX
- 0.017

Number of kernel evaluations: 21945 (100 % cached)

Compare to Linear Regression
Performance = $(72.8 \times \text{MYCT}) + (484.8 \times \text{MMIN}) + (355.6 \times \text{MMAX}) + (161.2 \times \text{CACH}) + (256.9 \times \text{CHMAX}) - 53.9$

Mean absolute error: 34.9692
Root mean squared error: 78.8572
Relative absolute error: 36.3357 %
Root relative squared error: 49.0245 %

Status: OK

WEKA (non-linear SVR)

The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. The classifier is 'SMOreg' with the following command: `SMOreg -S 0.001 -C 1.0 -T 0.001 -P 1.0E-12 -N 0 -K "weka.classifiers.functions.supportVector.RBFKernel -C 25"`. The 'Test options' section has 'Cross-validation' selected with 10 folds and 66% split. The 'Classifier output' pane shows the kernel used: $K(x,y) = e^{-(1.0 * \langle x-y, x-y \rangle^2)}$ and a list of support vectors for a normalized class. The status bar shows 'OK'.

Classifier output

Kernel used:
RBF kernel: $K(x,y) = e^{-(1.0 * \langle x-y, x-y \rangle^2)}$

Support Vector Expansion :
(normalized) class =

-0.2958	*	K[X(0), X]
+	1	* K[X(1), X]
+	-1	* K[X(2), X]
+	-1	* K[X(3), X]
+	-1	* K[X(4), X]
+	1	* K[X(5), X]
+	-1	* K[X(6), X]
+	1	* K[X(7), X]
+	-0.8307	* K[X(8), X]
+	0.8076	* K[X(9), X]
+	1	* K[X(10), X]
+	1	* K[X(11), X]
+	-0.4708	* K[X(12), X]
+	1	* K[X(13), X]
+	-1	* K[X(14), X]
+	-1	* K[X(15), X]
+	-1	* K[X(16), X]
+	-1	* K[X(17), X]
+	1	* K[X(18), X]
+	-1	* K[X(19), X]
+	-1	* K[X(20), X]

A list of support vectors

Status: OK

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 $\langle C, \gamma \rangle$ (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.