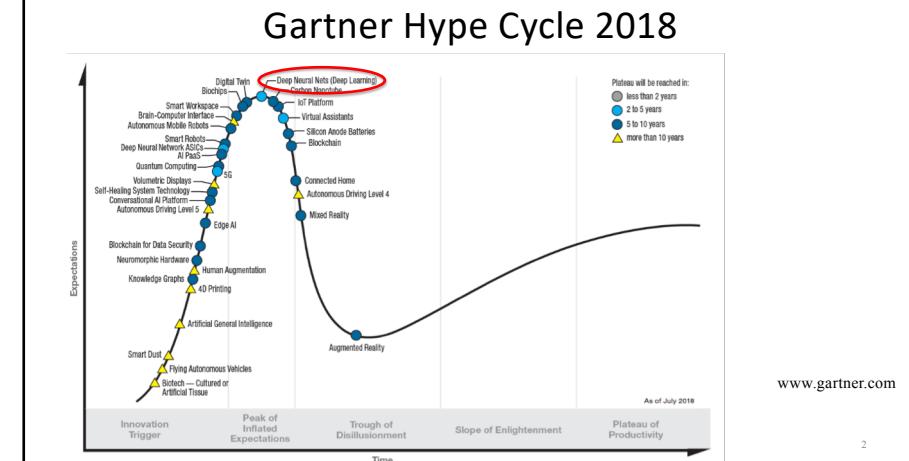


Neural Networks and Learning Systems
TBMI26 / 732A55
2019

Lecture 2
Supervised learning –
Linear classification

Magnus Borga
magnus.borga@liu.se

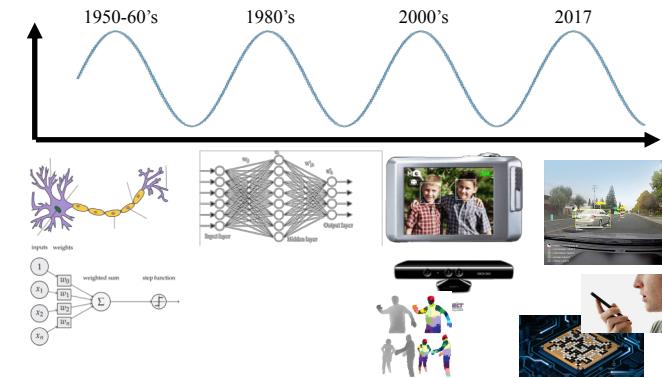


2

What is the hype about?



A history of hypes



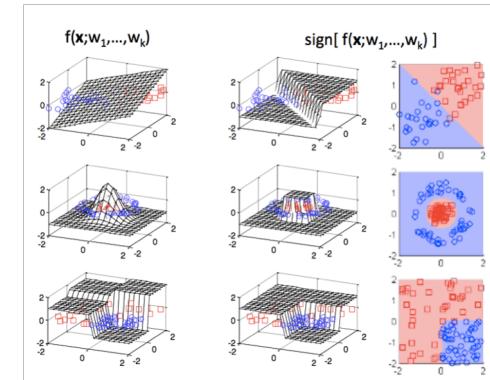
Recap - Supervised learning

- Task:** Learn to predict/classify new data from labelled examples.
- Input:** Training data examples $\{\mathbf{x}_i, y_i\}$ $i=1\dots N$, where \mathbf{x}_i is a feature vector and y_i is a class label in the set Ω . Today we'll assume two classes: $\Omega = \{-1, 1\}$
- Output:** A function $\text{sign}[f(\mathbf{x}; w_1, \dots, w_k)] \rightarrow \Omega$

Find a function f and adjust the parameters w_1, \dots, w_k so that new feature vectors are classified correctly. Generalization!

5

The function $f(\mathbf{x}; w_1, \dots, w_k)$



Advantages of a parametric function

$$f(\mathbf{x}; w_1, \dots, w_k)$$

- Only stores a few parameters (w_0, w_1, \dots, w_n) instead of all the training samples, as in k-NN.
- Fast to evaluate on which side of the line a new sample is on, for example $\mathbf{w}^T \mathbf{x} < 0$ or $\mathbf{w}^T \mathbf{x} > 0$ for a linear function.

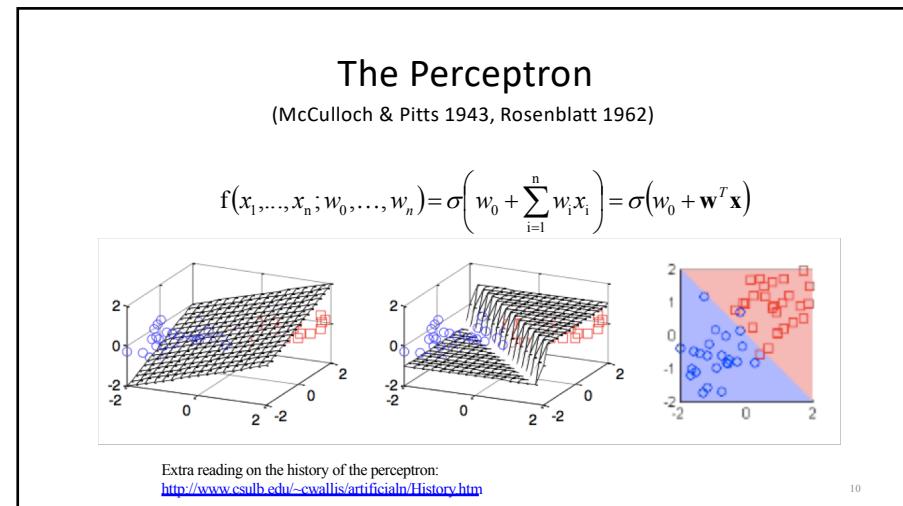
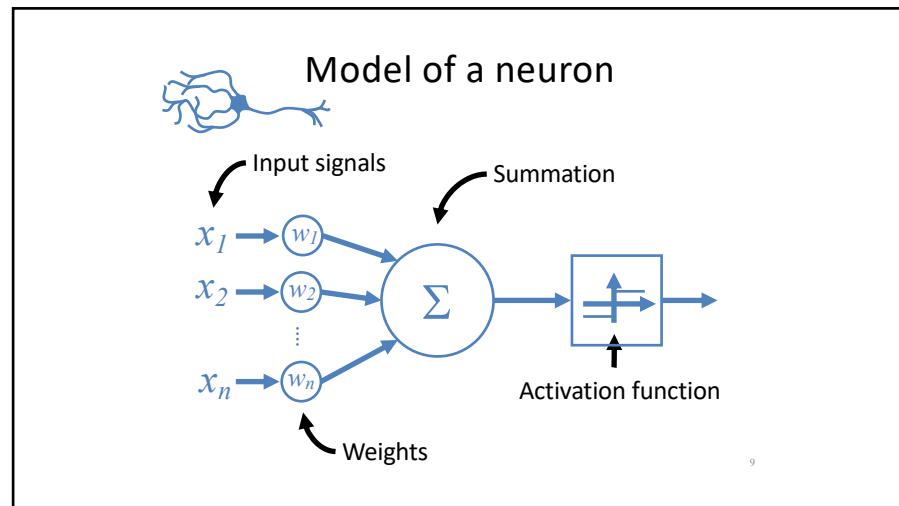
7

How does the brain take decisions?

(on the low level!)

- Basic unit: the neuron
-
- The human brain has approximately 100 billion (10^{11}) neurons.
 - Each neuron connected to about 7000 other neurons.
 - Approx. $10^{14} - 10^{15}$ synapses (connections).

8



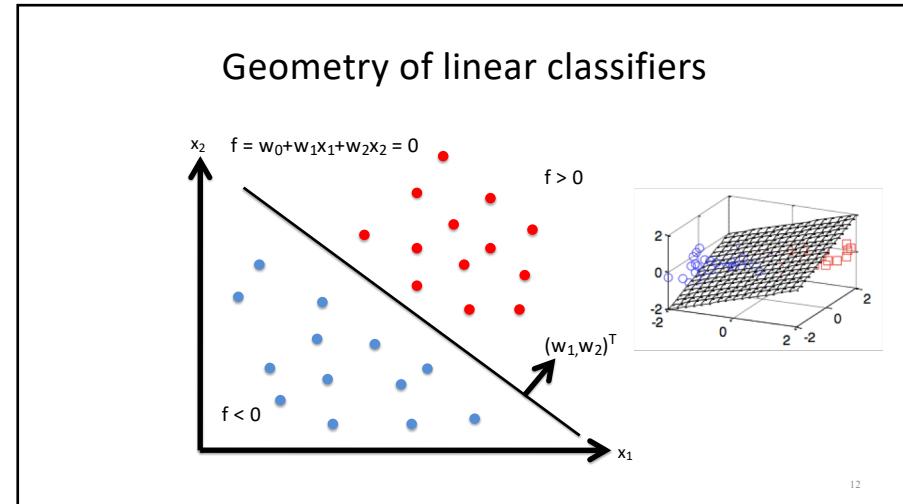
Notational simplification: Bias weight

Add a constant 1 to the feature vector so that we don't have to treat w_0 separately.

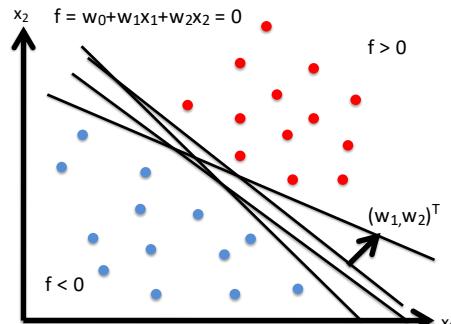
Instead of $\mathbf{x} = [x_1, \dots, x_n]^T$, we have $\mathbf{x} = [1, x_1, \dots, x_n]^T$

$$f(x_1, \dots, x_n; w_0, \dots, w_n) = \sigma\left(\sum_{i=0}^n w_i x_i\right) = \sigma(\mathbf{w}^T \mathbf{x})$$

11



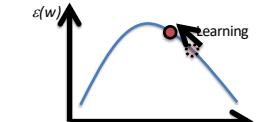
Which linear classifier to choose?



13

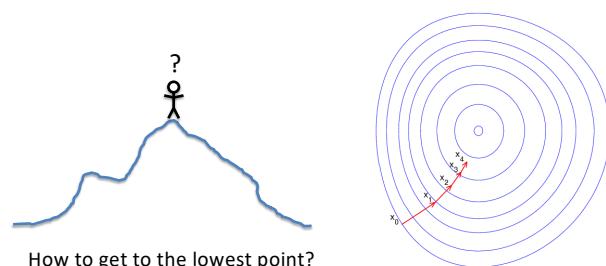
Find the best separator – optimization!

- Min/max of a cost function $\varepsilon(w_0, w_1, \dots, w_n)$ with the weights w_0, w_1, \dots, w_n as parameters.
- Ways to optimize:
 - Algebraic: Set derivative $\frac{\partial \varepsilon}{\partial w_i} = 0$ and solve.
 - Brute force: Try many values systematically and choose the best.
 - Iterative: Follow the gradient direction until minimum/maximum of ε is reached.



14

Gradient descent/ascent



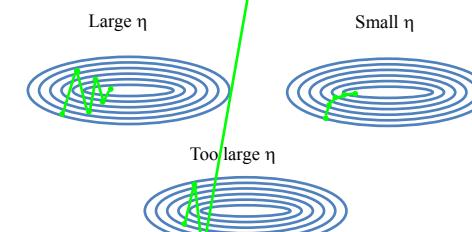
$$\nabla \varepsilon = \frac{\partial \varepsilon}{\partial \mathbf{w}} = \begin{pmatrix} \frac{\partial \varepsilon}{\partial w_1} \\ \frac{\partial \varepsilon}{\partial w_2} \end{pmatrix}$$

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} \pm \eta \frac{\partial \varepsilon}{\partial \mathbf{w}} \Big|_{\mathbf{w}^{(t)}}$$

15

Gradient descent

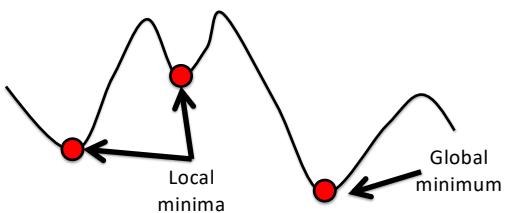
Choosing the step length



16

Local optima

- Gradient descent is not guaranteed to find the global minimum.
- With a sufficiently small step length, the closest local minimum will be found.

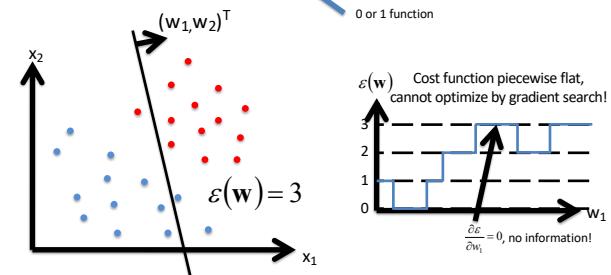


17

0-1 loss function / empirical risk

$$\varepsilon(\mathbf{w}) = \sum_{i=1}^N I(f(\mathbf{x}_i; \mathbf{w}) \neq y_i)$$

"empirical" because it is over the observed training data
0 or 1 function



18

Many different cost functions $\varepsilon(\mathbf{w})$

- 0-1 loss function / empirical risk
- Square error → Neural networks
- Maximum margin → Support Vector Machines

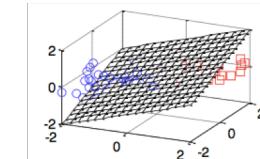
19

Square error cost

Minimize the following cost function

$$\varepsilon(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2$$

$N = \# \text{ training samples}$
 $y_i \in \{-1, 1\} \text{ depending on the class of training sample } i$



20

Minimization algorithm

$$\varepsilon(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2$$

$$\frac{\partial \varepsilon}{\partial \mathbf{w}} = 2 \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i) \mathbf{x}_i \quad \text{Exercise!}$$

Gradient descent:

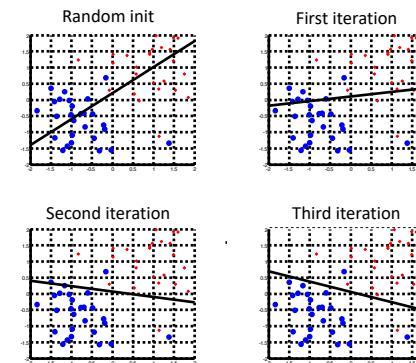
$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \frac{\partial \varepsilon}{\partial \mathbf{w}} = \mathbf{w}_t - \eta \sum_{i=1}^N (\mathbf{w}_t^T \mathbf{x}_i - y_i) \mathbf{x}_i \quad (\text{Eq.1})$$

Algorithm:

1. Start with a random \mathbf{w}
2. Iterate Eq. 1 until convergence

21

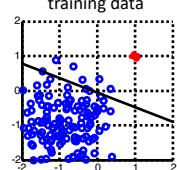
Example



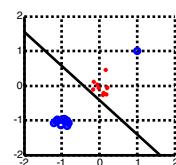
22

More examples

Unevenly distributed training data



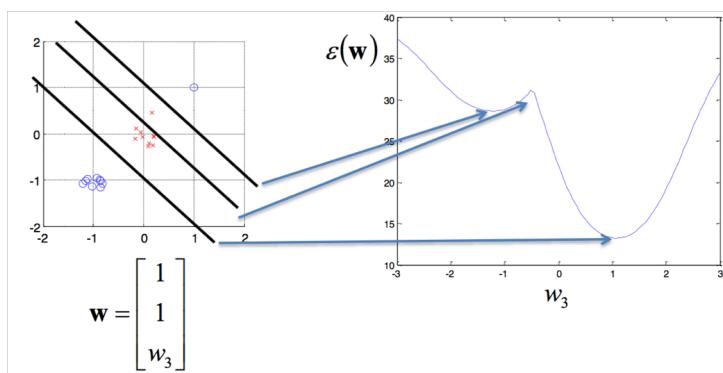
Outlier



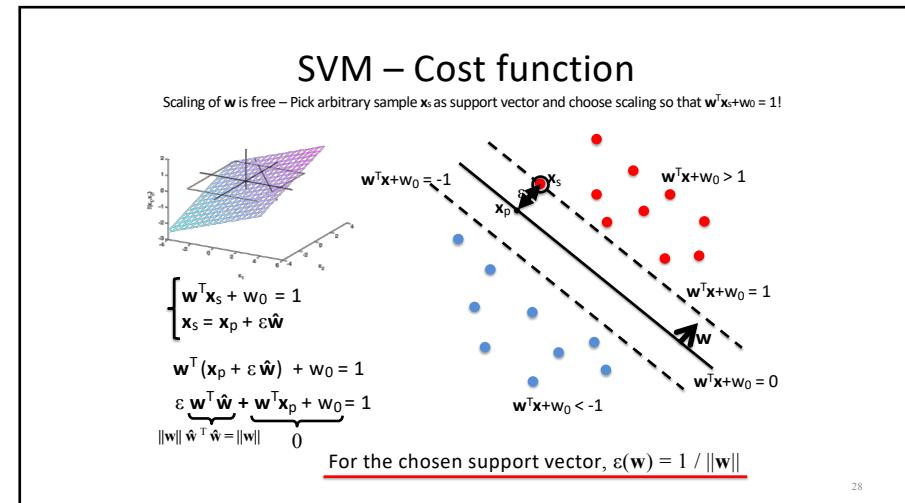
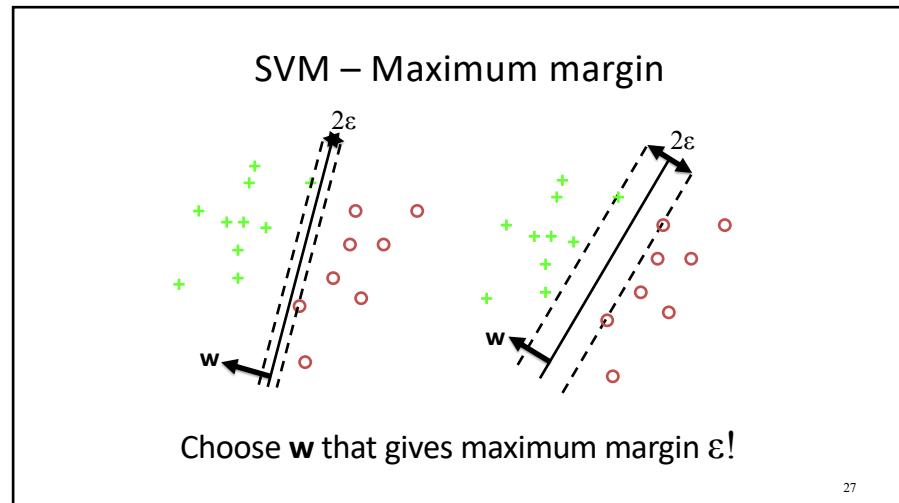
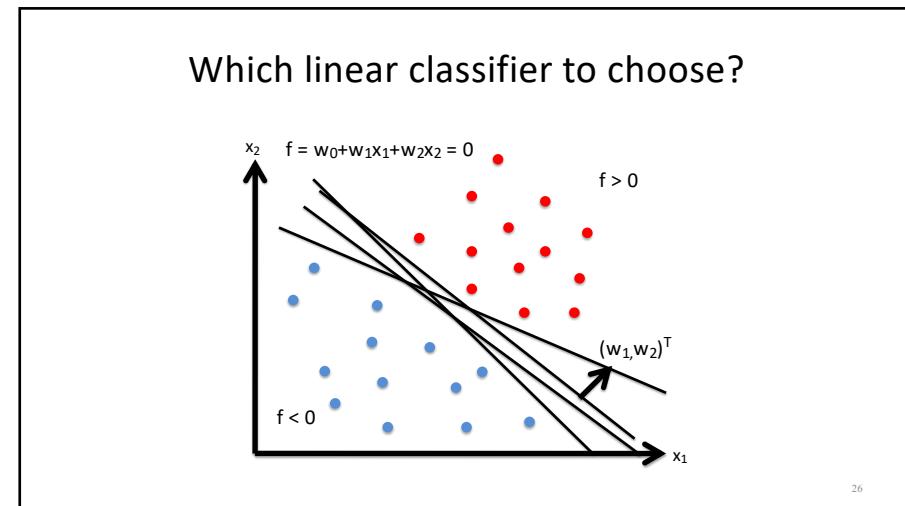
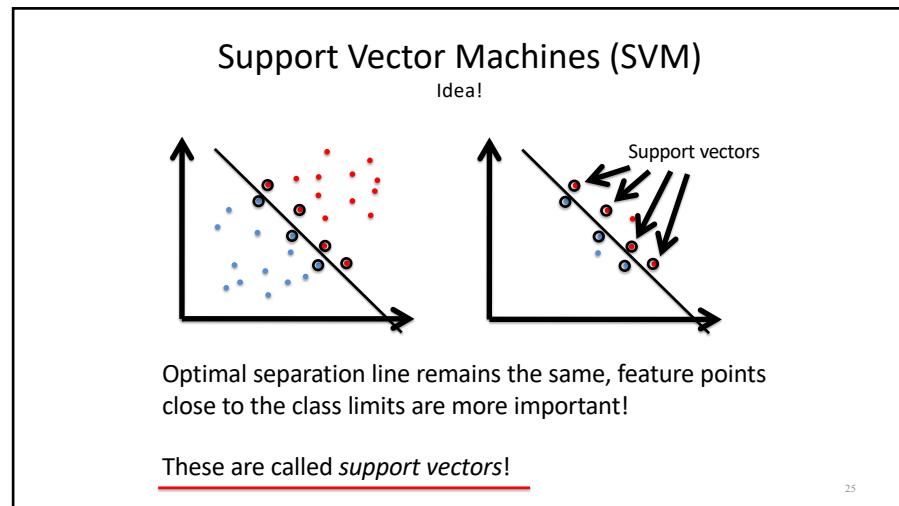
$$\varepsilon(\mathbf{w}) = \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2$$

23

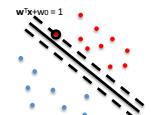
Example of local minimum



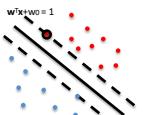
24



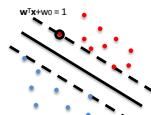
SVM cost function examples



Example 1:
Boundary not in middle →
Large $\|w\|$ (steep function) →
Low margin $\epsilon(w) = 1 / \|w\|$



Example 2:
Boundary more in middle →
Smaller $\|w\|$ (flatter function) →
Larger margin $\epsilon(w) = 1 / \|w\|$



Example 3:
Tilt boundary somewhat →
Smallest possible $\|w\|$ →
Largest margin $\epsilon(w) = 1 / \|w\|$

Choosing another training sample as reference support vector can give an even larger margin!

29

SVM – Cost function, cont.

Maximizing $\epsilon = 1 / \|w\|$ is the same as minimizing $\|w\|^2$!

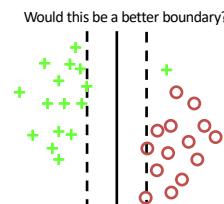
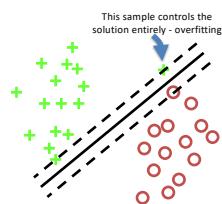
$$\begin{aligned} & \min \|w\|^2 \\ & \text{subject to } y_i(w^T x_i + w_0) \geq 1 \end{aligned}$$

No training samples must reside in the margin region!

Optimization procedure outside the scope of this course...

30

SVM – Soft margin



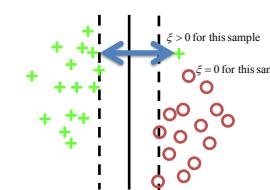
What if the training data is not separable with a line?



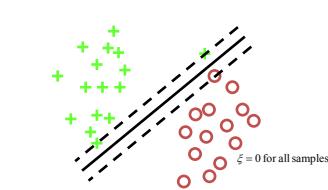
31

SVM – Soft margin, cont.

$$\begin{aligned} & \min \|w\|^2 + C \sum \xi_i \\ & \text{subject to } y_i(w^T x_i + w_0) \geq 1 - \xi_i \end{aligned}$$



Solution for small C



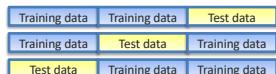
Solution for large C

32

SVM – Choosing C

Solve the optimization problem with different C s and choose the solution with highest accuracy according to cross-validation procedure.

$$C = 2^{-5}, 2^{-3}, \dots, 2^{15}$$



Practical guide:
<http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf>

33

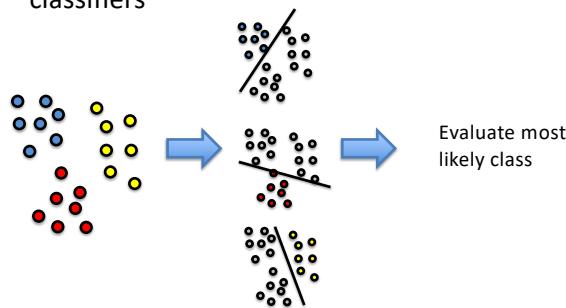
Summary – Linear classifiers

- **Different cost functions give different algorithms**
- **Square error cost**
 - Sensitive to outliers and training data distribution when applied as in this lecture.
 - Improvements possible (lecture 3).
 - Local minima.
- **Support Vector Machines (maximum margin cost)**
 - By many considered as the state-of-the-art classifier.
 - Non-linear extension possible (lecture 9).
 - Many software packages exist on the internet.
 - No local minima.
- **Fisher Linear Discriminant (Lecture 8)**
 - Simple to implement, very useful as a first classifier to try

34

What about more than 2 classes?

- Common solution: Combine several binary classifiers



35