Linear Models for Classification

Daniel Sadoc Menasché

(Textbook Material: C. Bishop, Chapter 4)

The Classification Problem

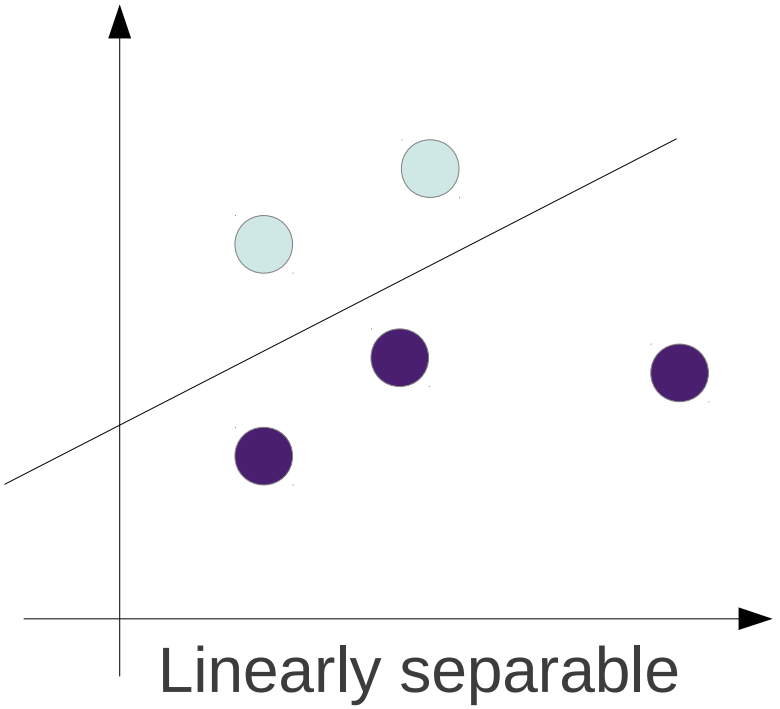
- **Supervised learning**
- Given
  - K classes
  - n points in the data set
  - Training set
    - (x1, x2, …, xn): n data points
    - (t1, t2, …, tn): classes of n data points
- Classify new data x' into its most likely class

How to Encode Target Values

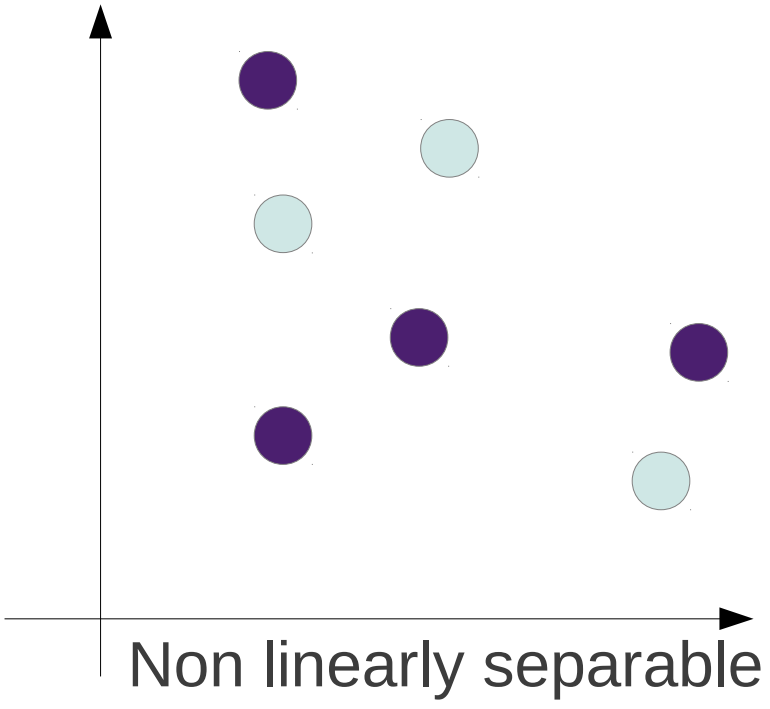
- 1-of-K coding
  - Class 1, t=(1,0,0)
  - Class 2, t=(0,1,0)
  - Class 3, t=(0,0,1)
- In some cases, can interpret t as probability of x being in each class

Decision Boundary

- **Decision boundaries** divide data points into **decision regions**
- Linear classification = decision boundary linear function of input



Linearly separable



Non linearly separable

Three Approaches for Classification

- Given
  - x, data
  - C, classes

1)Discriminant function

2)Inference through discriminative model: P(C|x)

3)Inference through generative model: P(x|C)

Three Approaches for Classification

- Given
  - x, data
  - C, classes

1)Discriminant function

2)Inference through discriminative model: P(C|x)

3)Inference through generative model: P(x|C)

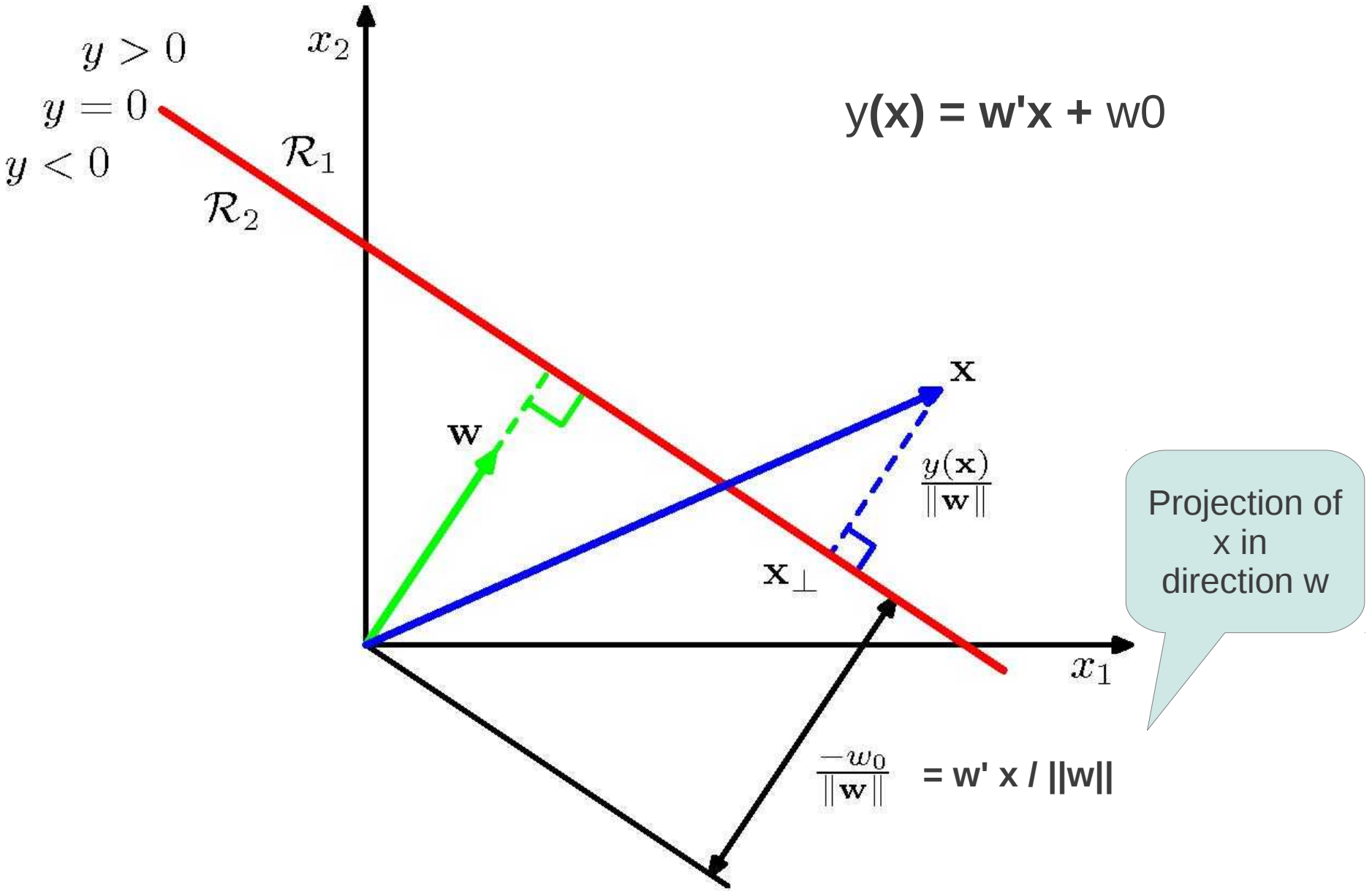
Difficult to update
in face of novel data

Unafected by epidemics,
robust

Activation Function

- Learn weights **w**
- Classify **x** based on value of
  - **w' x** + w0
  - e.g., if (**w' x** + w0) > 0, class 1, otherwise class 2
- Generalization
  - **Activation function f**
  - Classify **x** based on f(**w' x** + w0)
  - e.g., if f(**w' x** + w0) > 0, class 1, otherwise class 2

Discriminant Functions



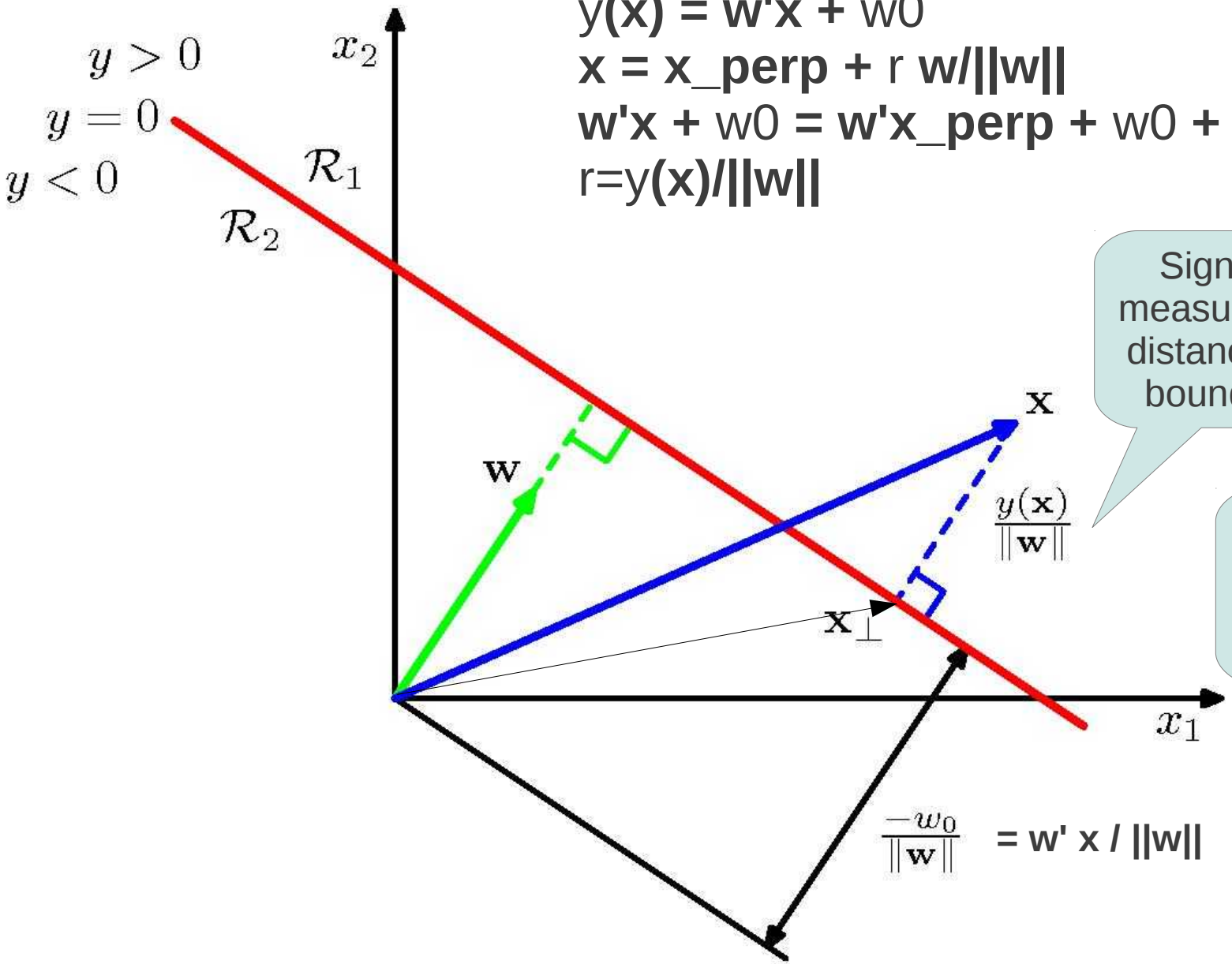
y**(x) = w'x +** w0

Projection of x in direction w

= w' x / ||w||

Discriminant Functions

y**(x) = w'x +** w0
x = x_perp + r **w/||w||**
w'x + w0 **= w'x_perp +** w0 **+ w'** r **w/||w||**
r=y**(x)/||w||**

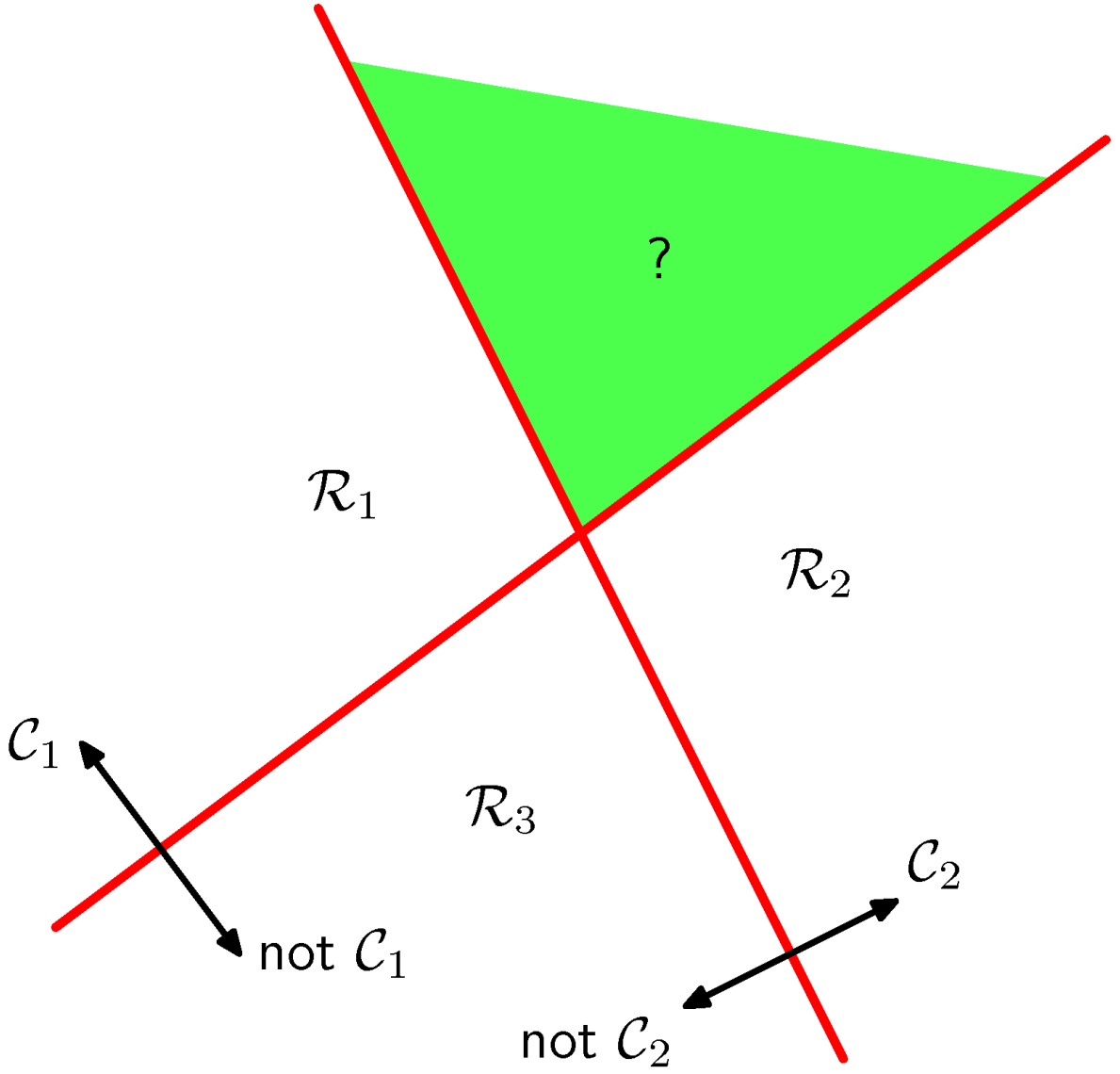


Signed measured of distance to boundary

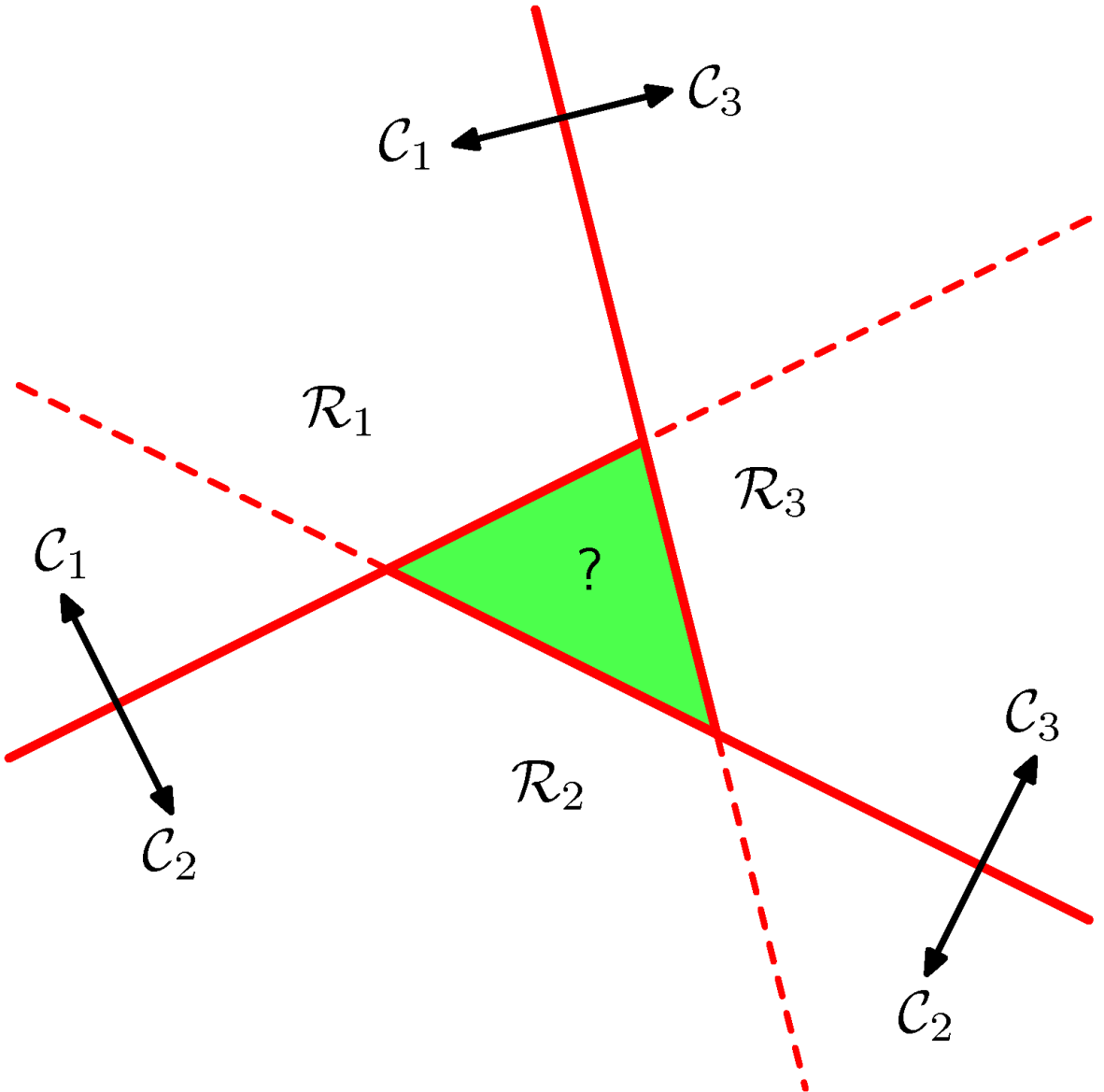
Projection of x in direction w

= w' x / ||w||

Multiple Classes: One-Versus-Rest



Multiple Classes: One-Versus-One

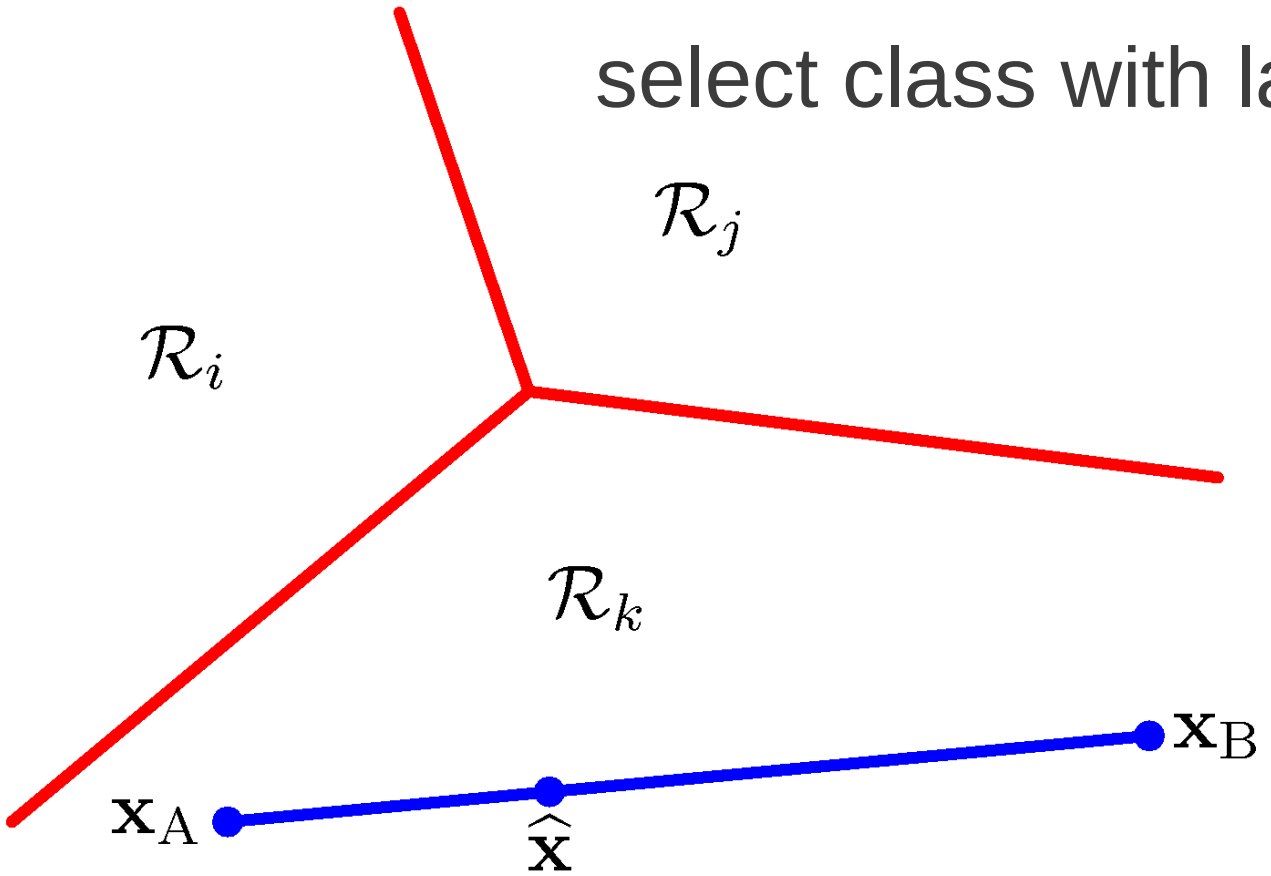


Multiple Classes: Solution

For each x, compute K values,

$$y_k(x) = w_k^T x + w_{k0}$$

select class with largest y_k



How to Compute Weights?

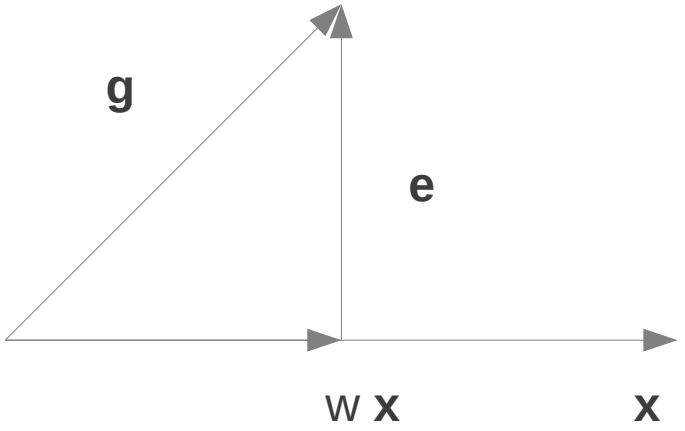
- Three possible ways
  - “The good, old” least squares
  - Fisher discriminant
  - Perceptron (single neuron neural network)

Least Squares

- Recall from linear regression...

Approximating a Vector

- **e = g -** w **x**
- **g' x** = **|g| |x|** cos φ
- **|x|^2 = x' x**
- cos φ = w **|x|** / **|g|**



- **Major result:** w **= g' x (x' x)^(-1)**

Coefficient to compute
1 x 1

Given target point
N x 1

Given basis matrix
N x 1

Least Square & Regression

- **e = g - x w**
- **g' x = |g| |x|** cos φ
- **|x|^2 = x' x**
- cos φ = w **|x|** / **|g|**

w=vector of computed coefficients
g=vector of target points
x=basis matrix
- each column is a basis elem.
- each column is a polynomial evaluated at desired points

- **Major result: w' = g' x ((x' x)^(-1))'**

Coefficient to compute
D+1 x 1

Given target point
N x 1

Given basis matrix
N x D+1

Least Square & Classification

- **e = g - x w**
- **|x|^2 = x' x**

w=matrix of coefficients (one class per column)
g=matrix of target points (one per row)
x=basis matrix
- each column is a basis elem.
- each column is one of the dimensions of the given points

We now have an error matrix

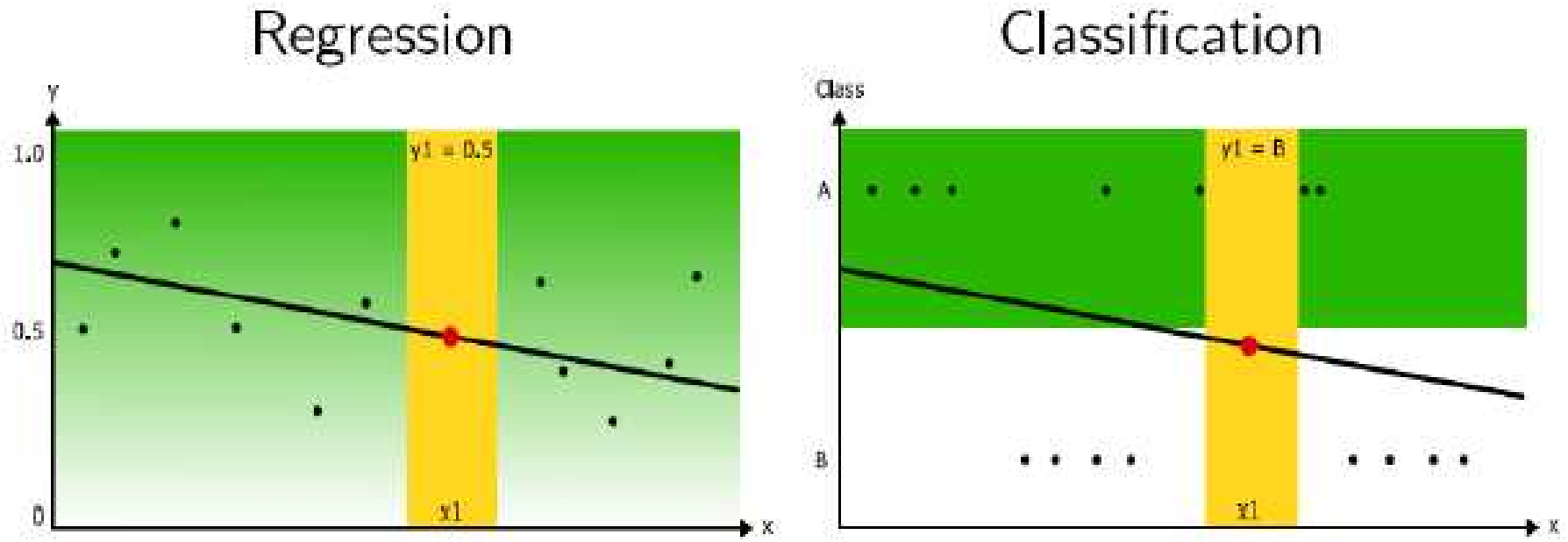
- **Major result: w' = g' x ((x' x)^(-1))'**

Coefficient to compute
(D+1) x K

Given target point
(N) x (K)

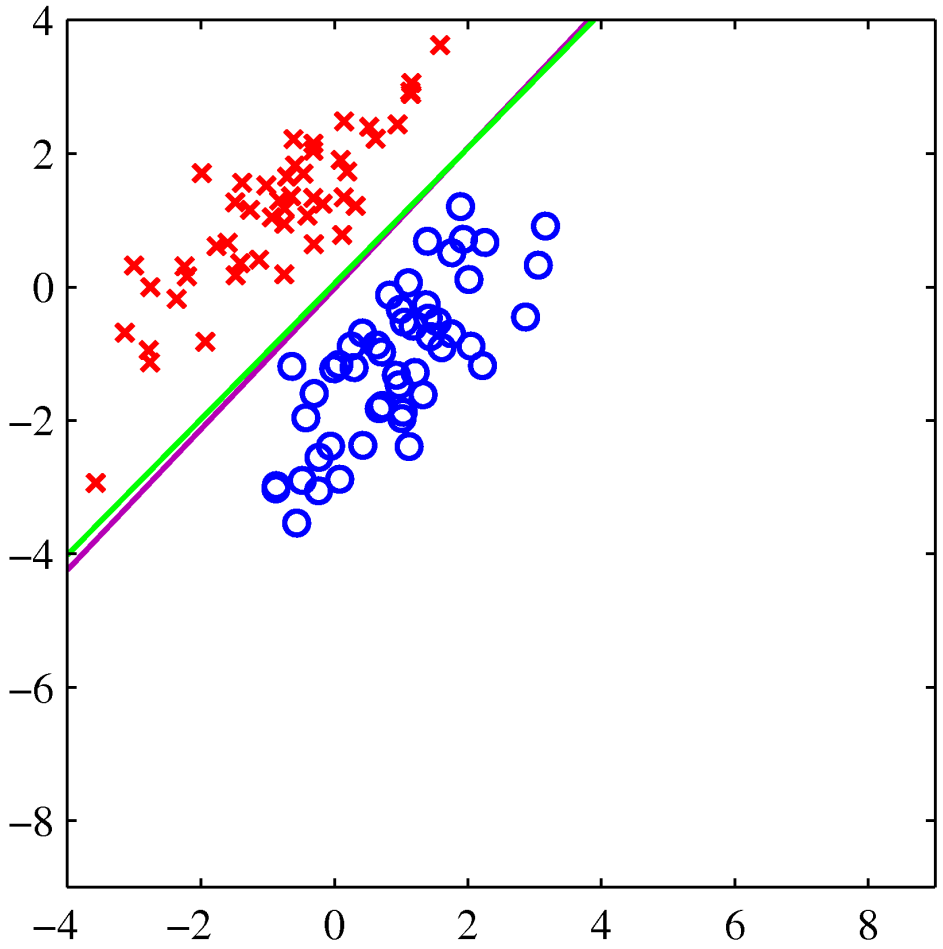
Given basis matrix
(N) x (D+1)

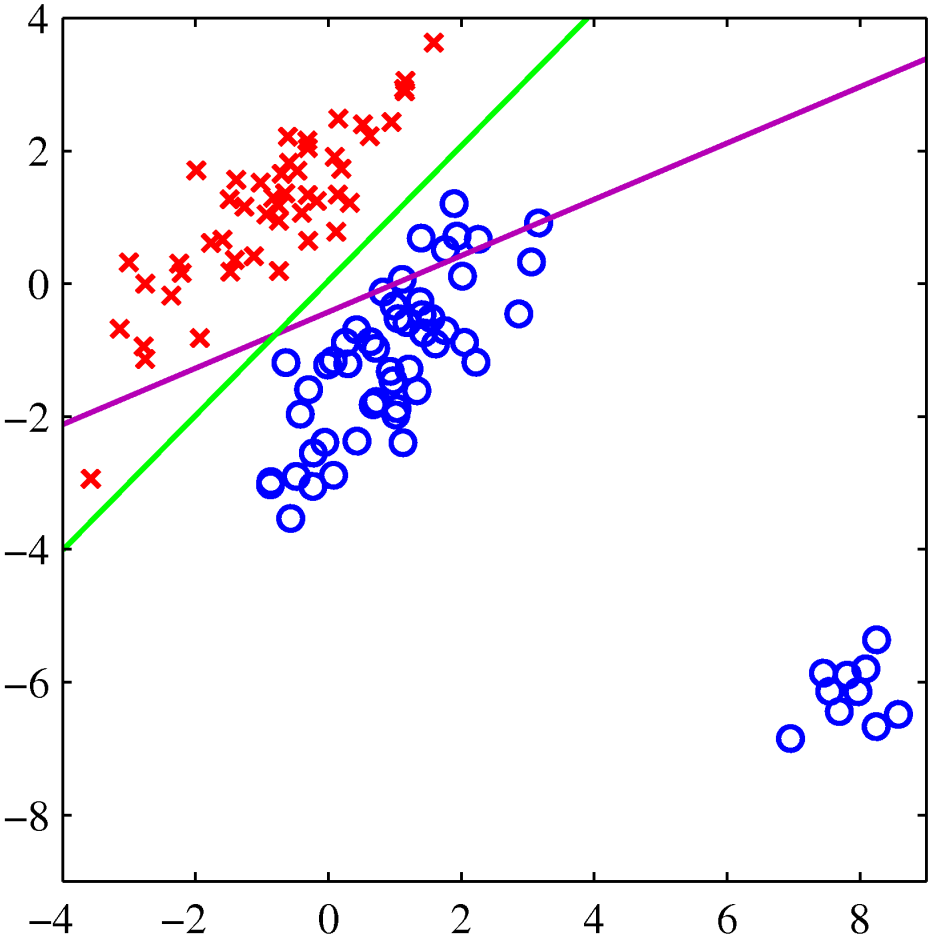
Regression versus Classification



http://informatik.unibas.ch/lehre/hs09/cs253/slides/LS2_09.pdf

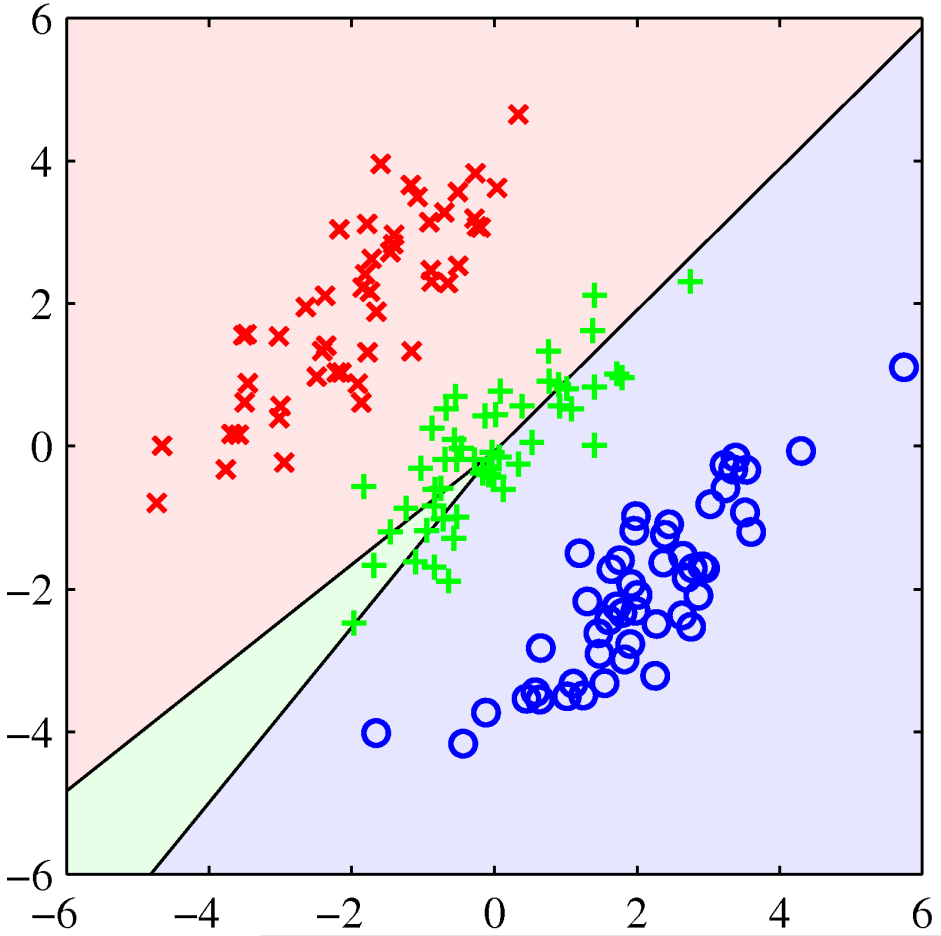
Problem With Least Squares I

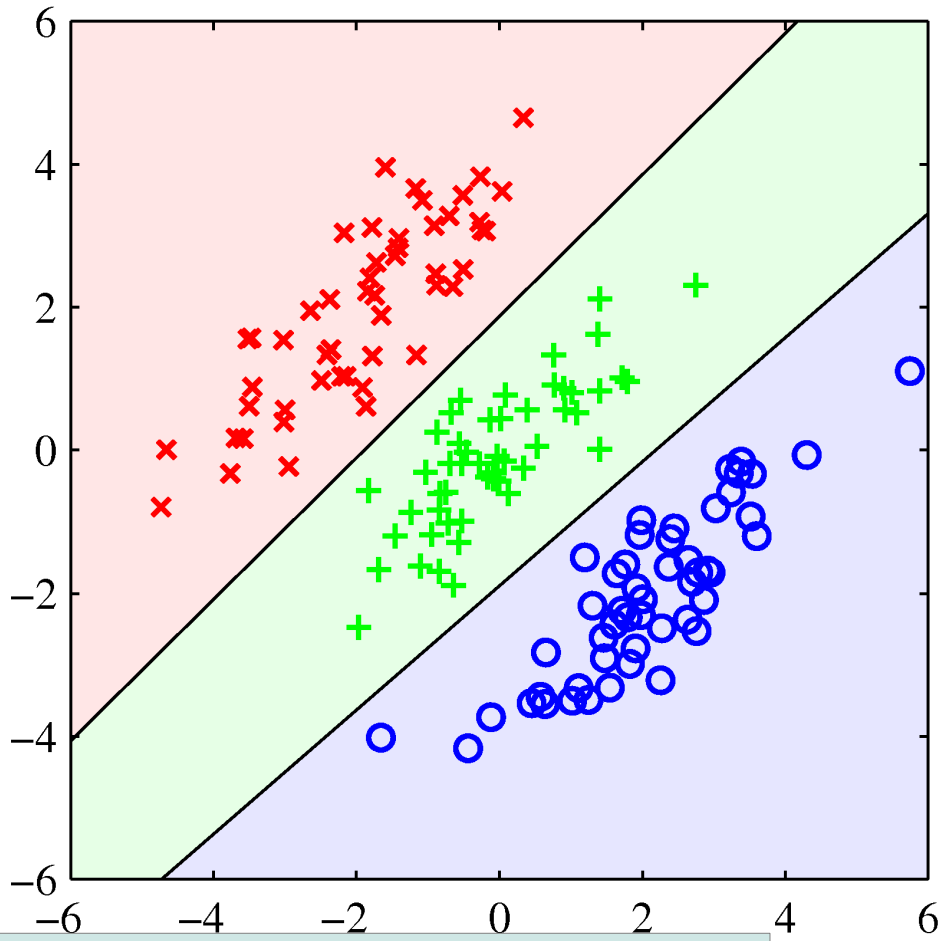




Least squares highly sensitive to outliers

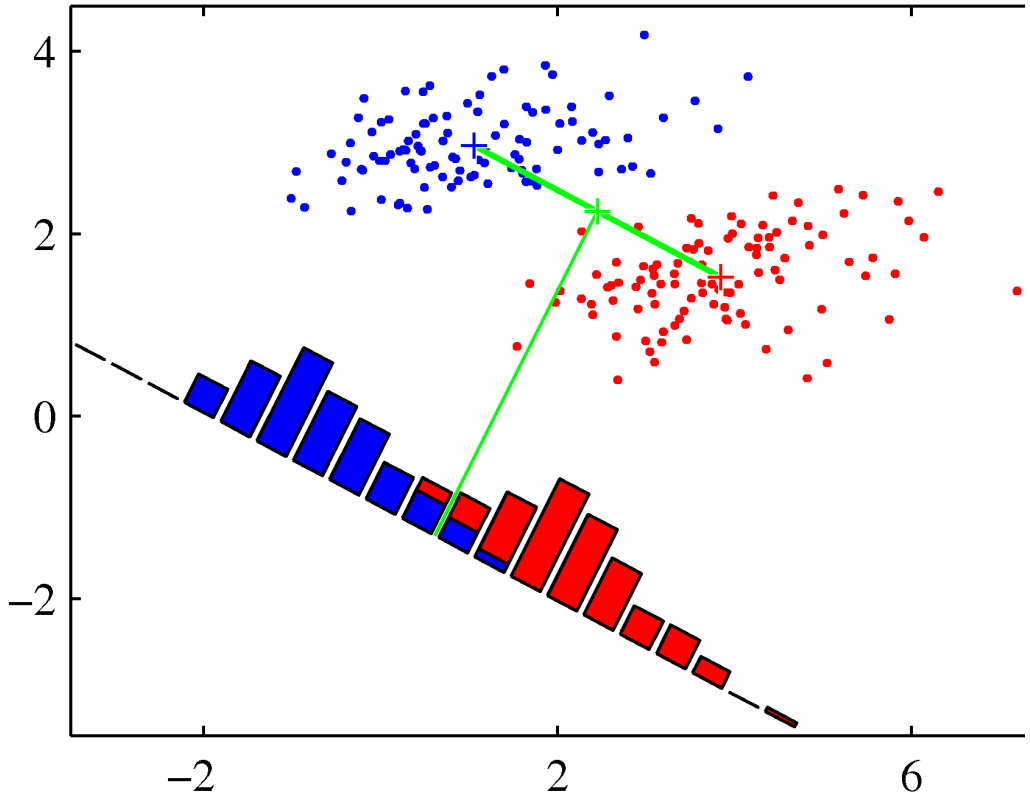
Problem With Least Squares II

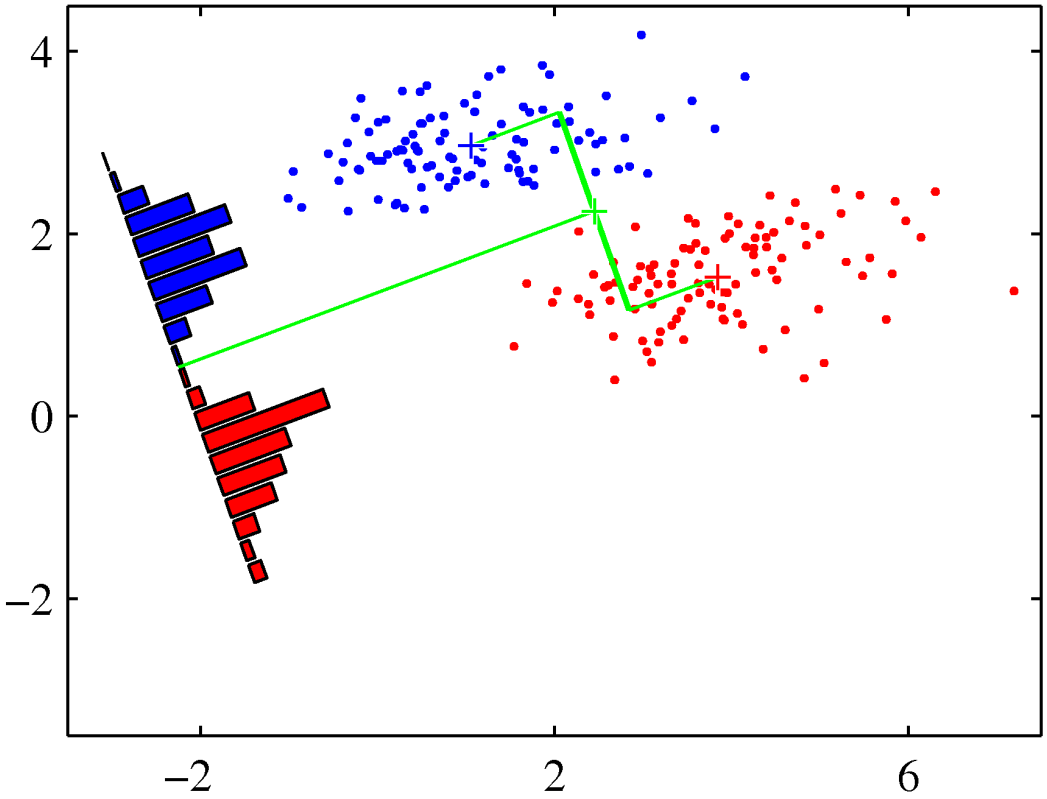




Least squares worse than logistic regression

Fisher Discriminant: Dimensionality Reduction





Weights 1) max distance btw means 2) small variance in class

Perceptron I

- Neural network with single neuron (Rosenblatt)
- y(x) = f(w' x)
- f(a)
  - +1, a >= 0,
  - -1, a <= 0

t(n) := +1 if sample of class
-1 otherwise

- **Perceptron algorithm**
  - **Cycle through training data, for each data x_n, if f(w(t)' x_n) does not lead to right classification**
    - **w(u+1) = w(u) + s x_n t(n)**
    - **u := u + 1**

Perceptron I

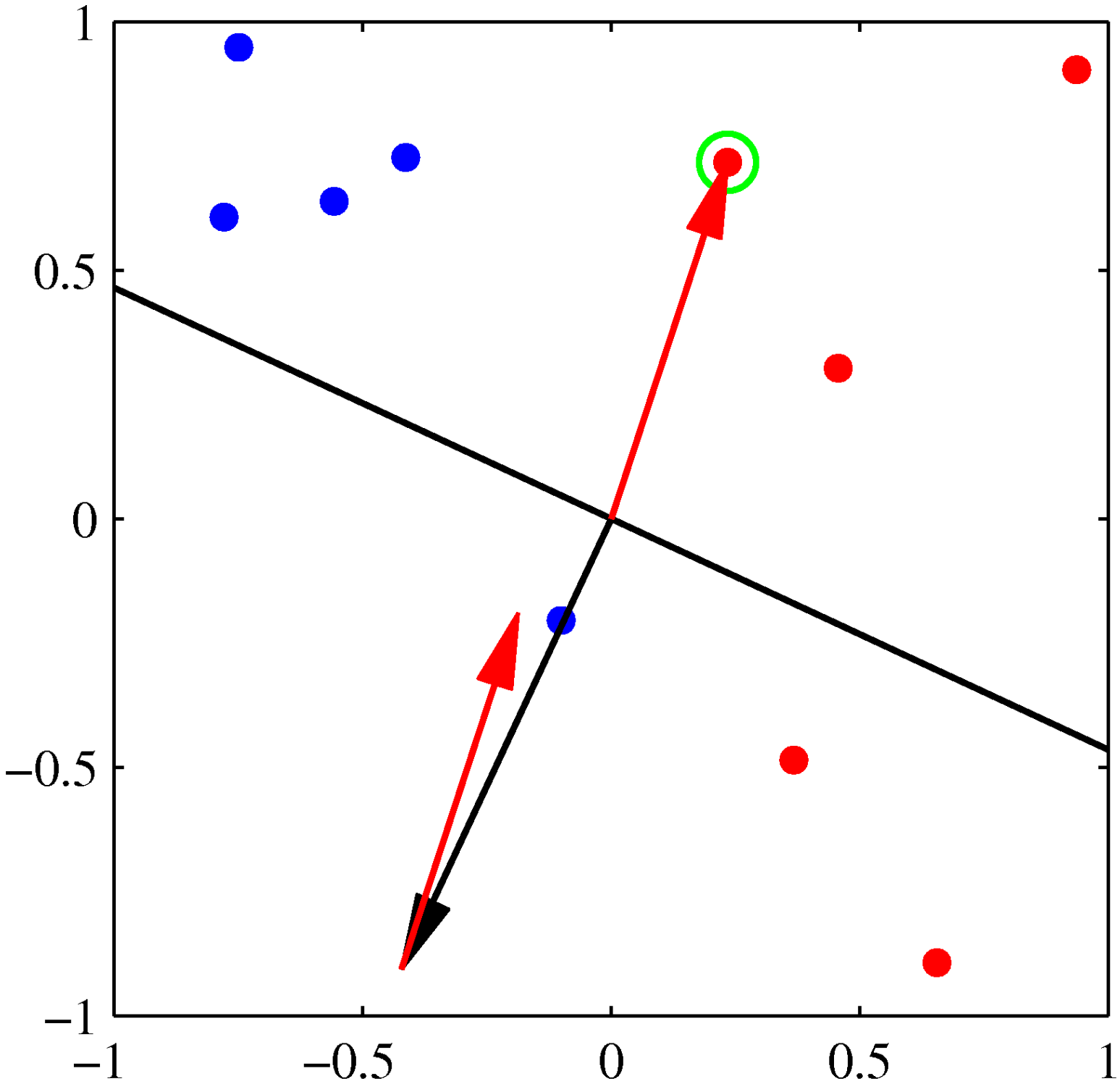
- Neural network with single neuron (Rosenblatt)
- y(x) = f(w' x)
- f(a)
  - +1, a >= 0,
  - -1, a <= 0

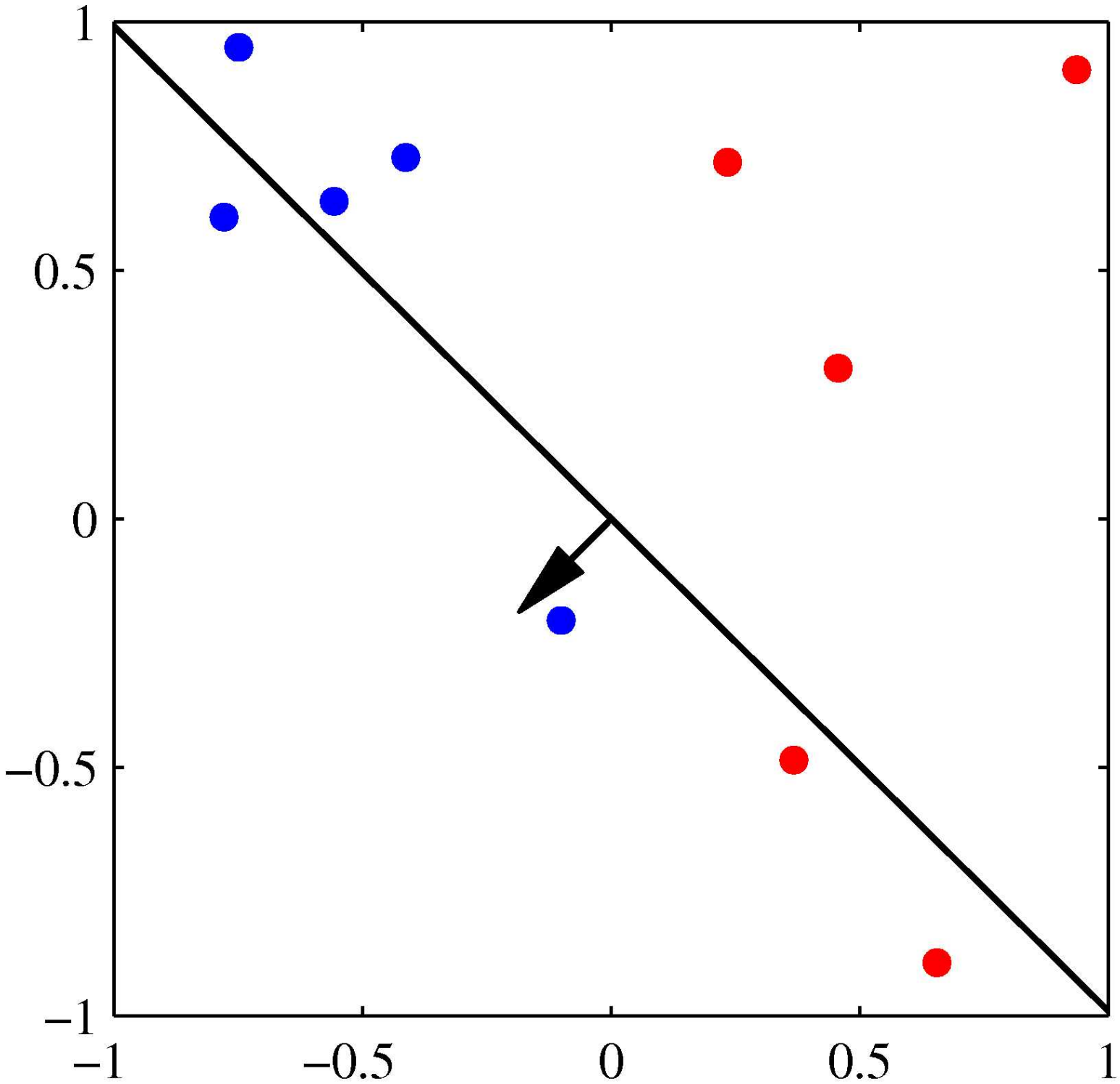
t(n) := +1 if sample of class A
-1 otherwise

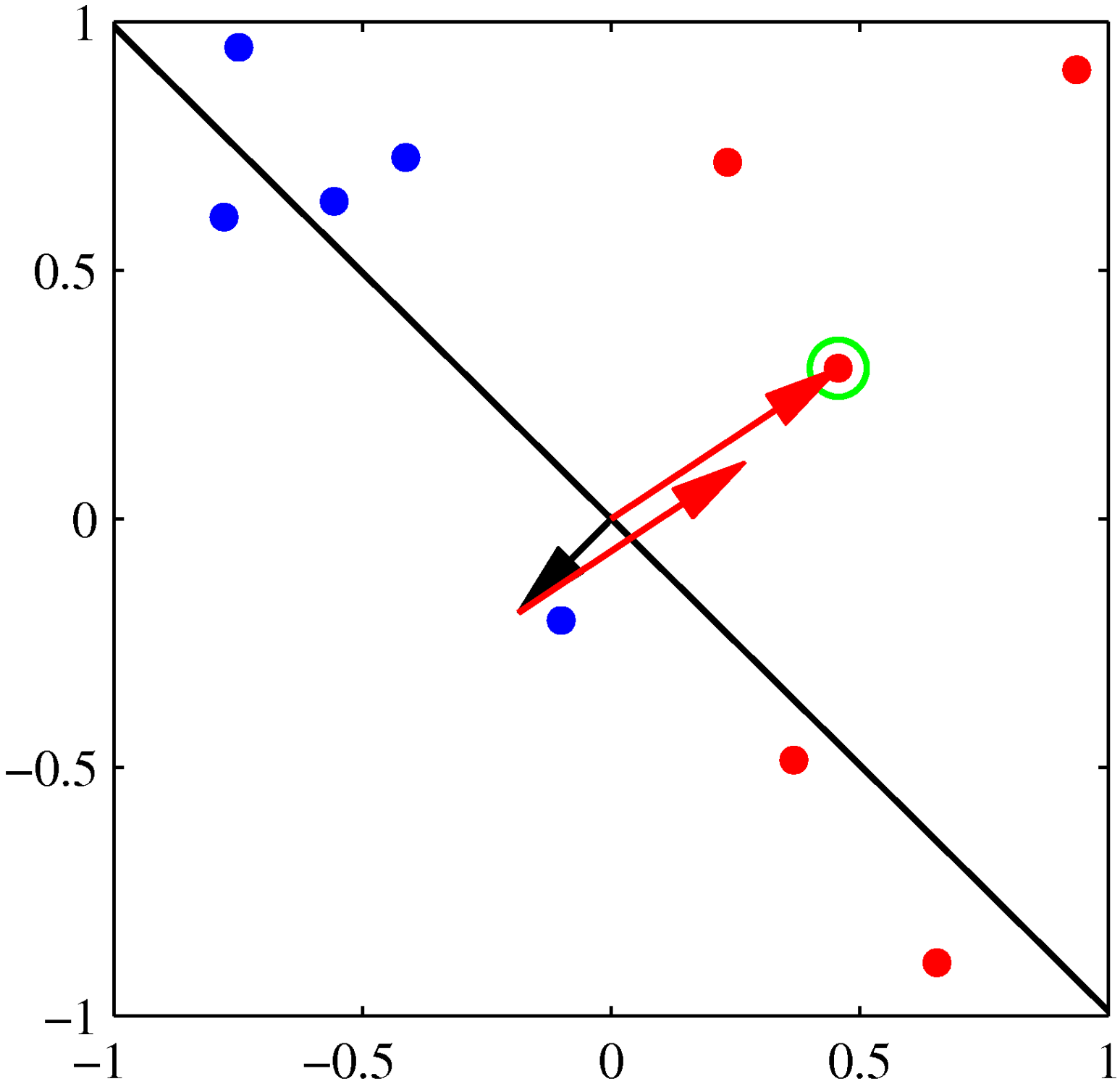
- **Perceptron algorithm**
  - **Cycle through training data, for each data x_n,**
    if f(w(u)' x_n) does not lead to right classification
    - **ERROR(n) = t(n) – t'(n)**
    - **w(u+1) = w(u) + s x_n ERROR(n)**
    - **u := u + 1**

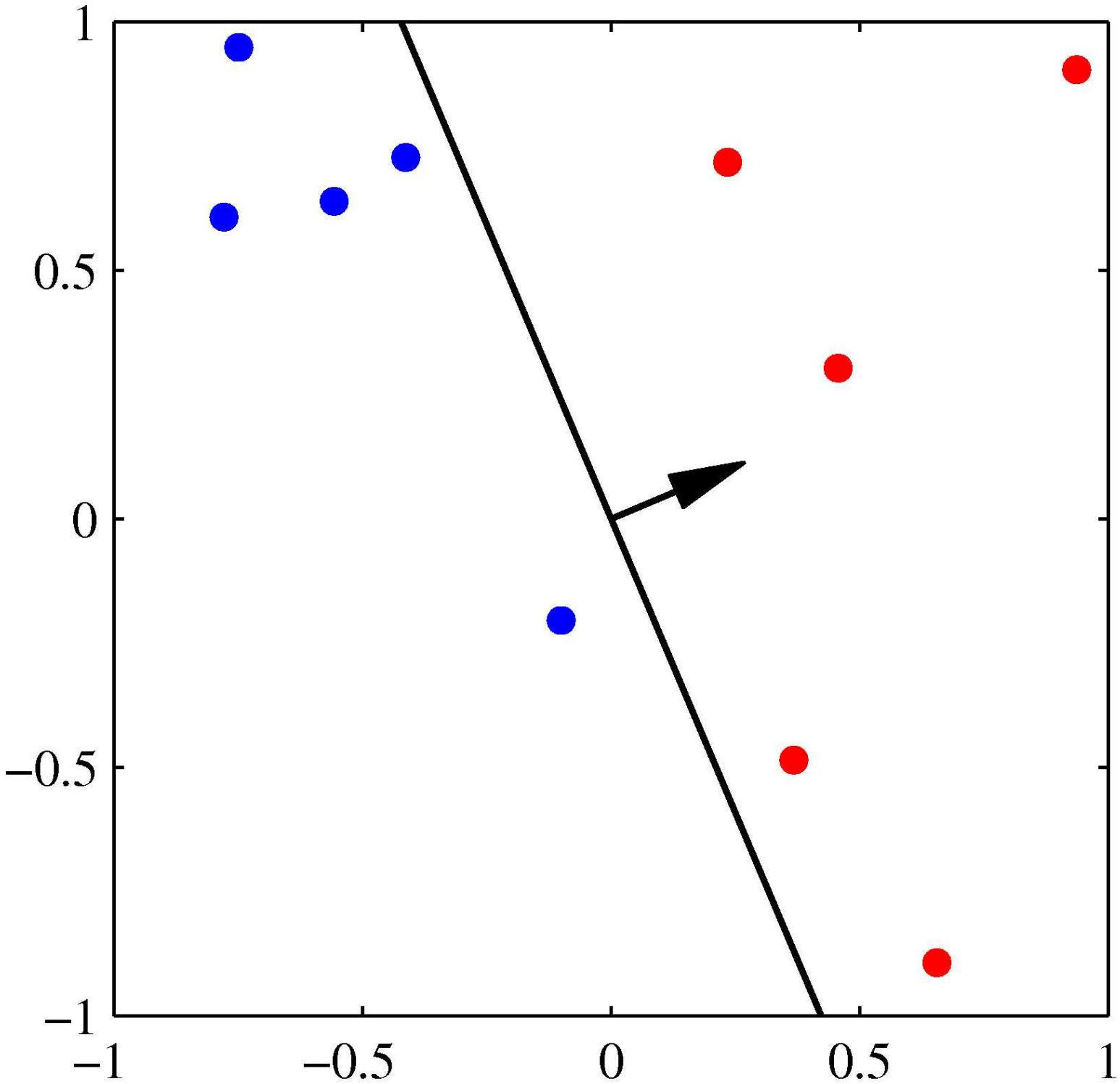
Perceptron II

- **Perceptron convergence theorem**: If the input set is linearly separable, the perceptron algorithm converges
- Time to convergence?
  - Unknown


1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1


1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1


1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1


1
0.5
0
−0.5
−1
−1
−0.5
0
0.5
1