Deep Learning

3. General approach to classification. Linear classifiers. Logistic regression. Gradient descent.

Viacheslav Dudar

Taras Shevchenko National University of Kyiv

2018

Classification

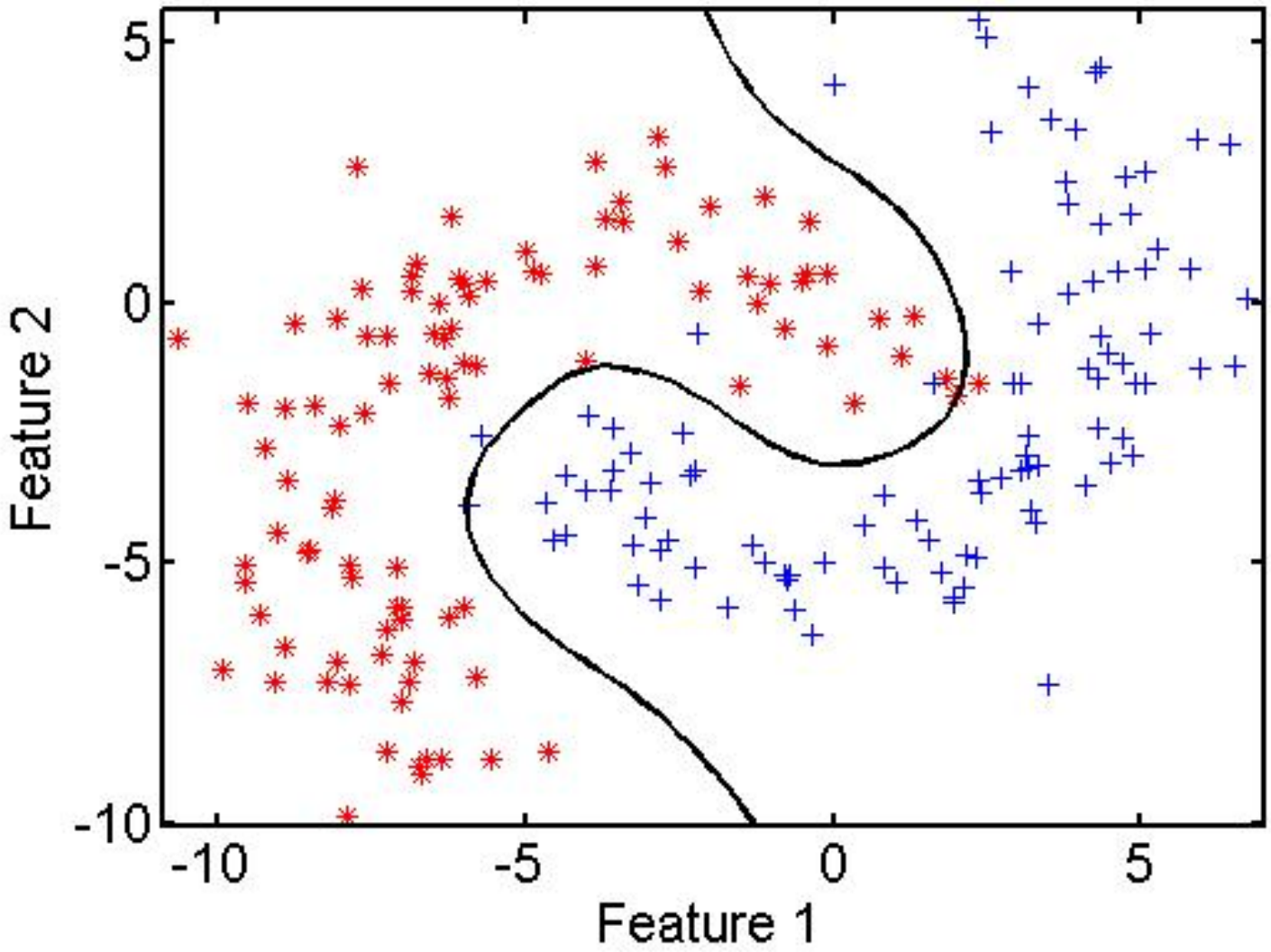
- Goal of classification: assign input vector x to one of K discrete classes C_k
- Input space is thereby divided into **decision regions**
- Boundaries of decision regions are called **decision boundaries** or **decision surfaces**

Types of classification

- **Binary**: number of classes is 2
- **Multiclass**: number of classes is 3 and more

Binary classification: general overview

- Use training set to divide input set into 2 decision regions
- Classify new points using these decision regions



Training binary classifier

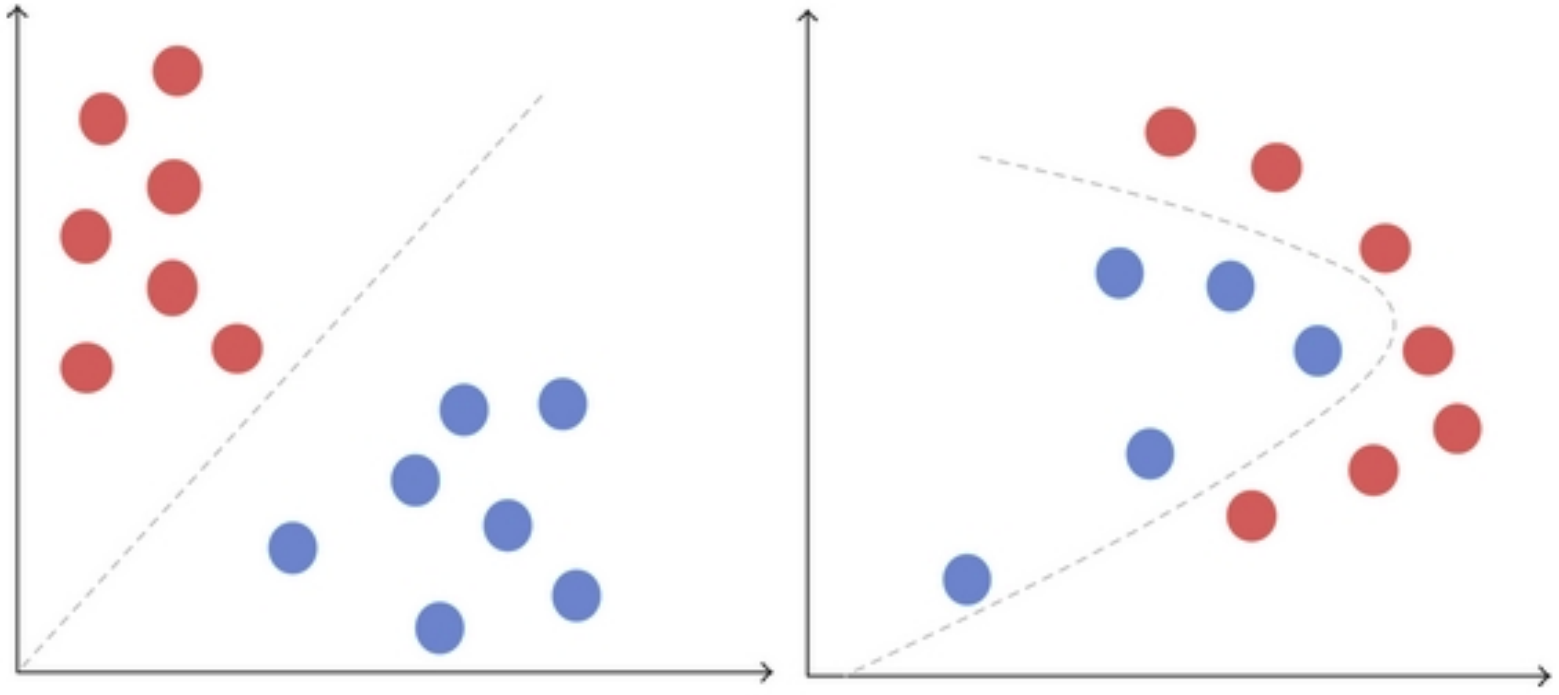
Suppose we have 2 classes: C_{-1} and C_1
Input vector: $x \in R^N$
Aim: construct **discriminant** function $f(x)$ such that:
$f(x) > 0$ if $x \in C_1$
$f(x) < 0$ if $x \in C_{-1}$
Surface $f(x) = 0$ is **decision surface**.

Training binary classifier

We build a parametric function $f(x, w)$: x is input vector, w are unknown coefficients to be determined during training.
Train set: (x_i, t_i), where $x_i \in R^N$, $t_i \in \{-1, 1\}$
Learning stage:
Find parameters w_* such that for most training elements we have:
$f(x_i, w_*) > 0$ if $t_i = 1$
$f(x_i, w_*) < 0$ if $t_i = -1$
Test stage: For new input x_{new}:
if $f(x_{new}, w_*) > 0$ then classify as C_1
if $f(x_{new}, w_*) < 0$ then classify as C_{-1}

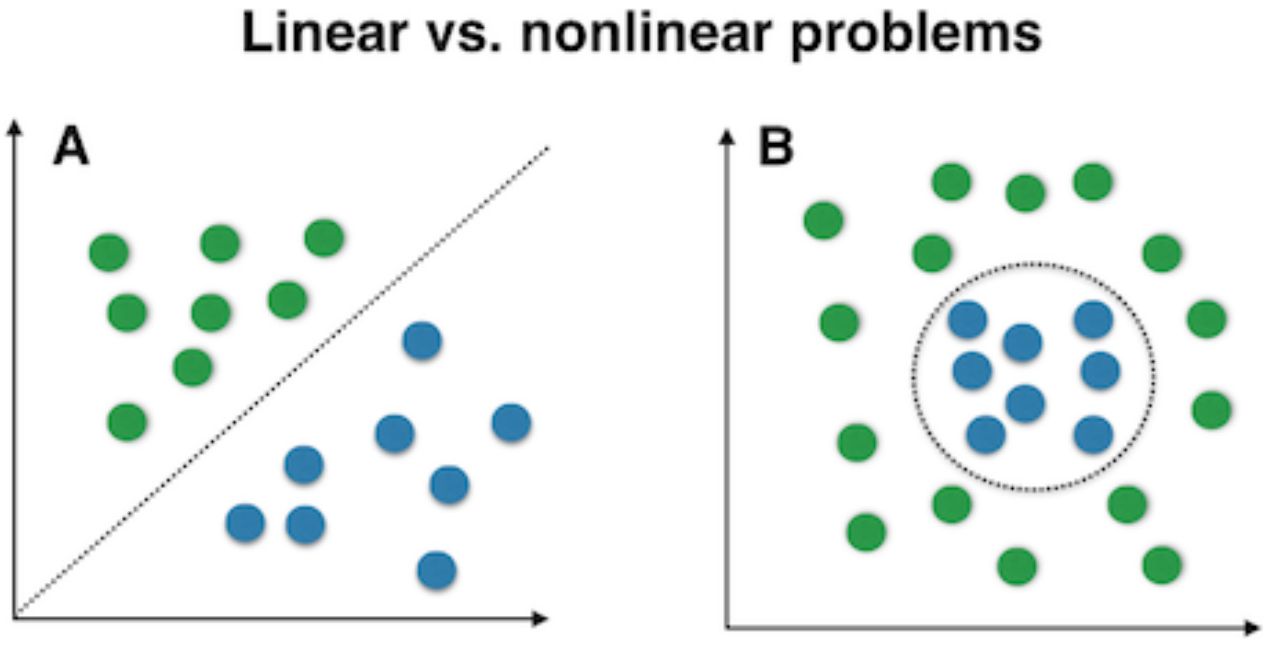
Linear and nonlinear classifiers

Classifier is called **linear** if $f(x, w)$ is linear
Classifier is called **nonlinear** if $f(x, w)$ is nonlinear



Linear separability

Training set is **linearly separable** if there exists hyperplane that perfectly separates data points.



Attempt 1: least squares for classification

$$y(x, w) = w_0 + \sum_{i=1}^{N} w_i x_i = w_0 + w^T x$$

To get rid of w_0 we extend input x with a fixed number 1 to get: $(1, x)$. With new x we have: $y(x, w) = w^T x$
Now we look for w that minimizes such expression:

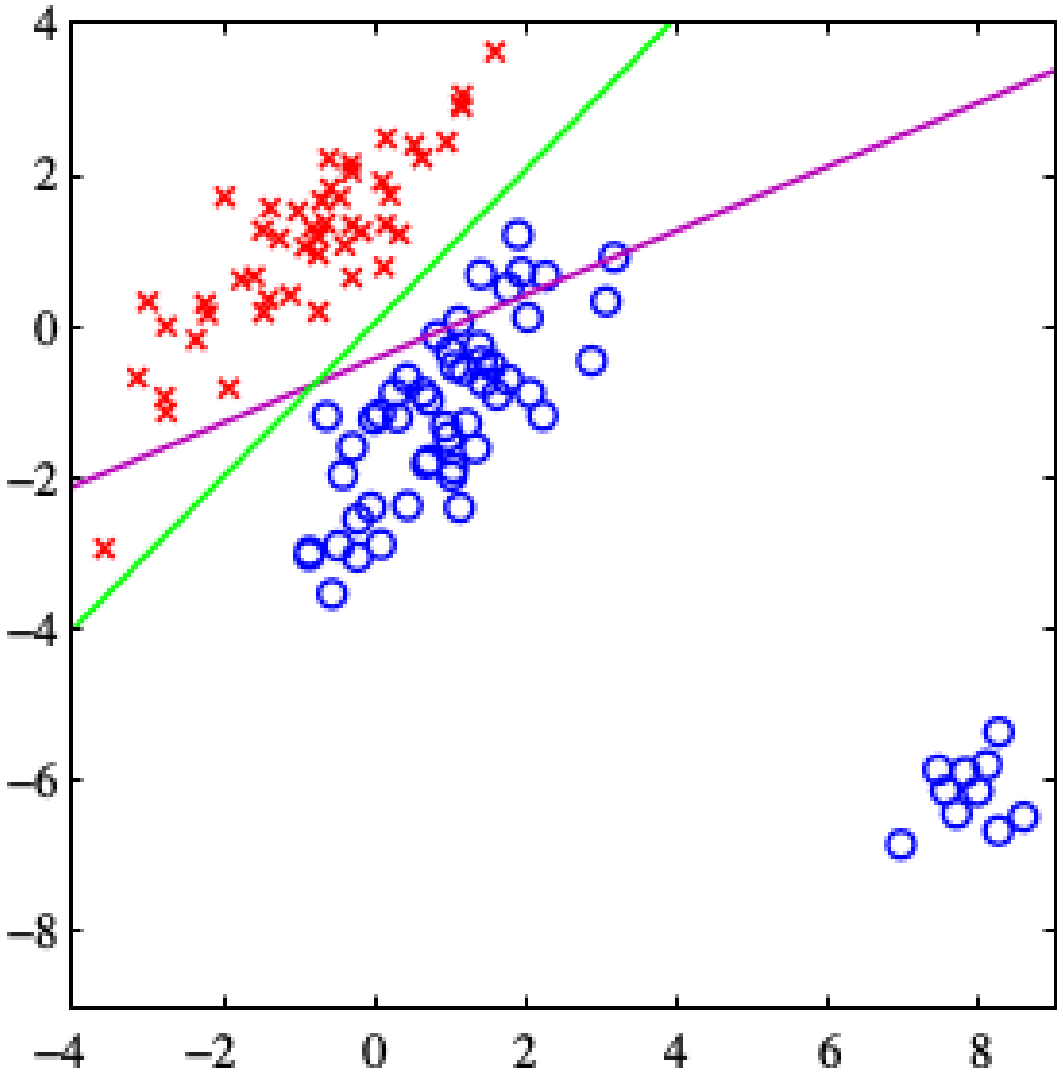
$$F(w) = \sum_{i=1}^{M} (w^T x_i - t_i)^2 \rightarrow min$$

It becomes linear regression problem and has closed-form solution.

$$w_* = (X^T X)^{-1} X^T t$$

Least squares problems

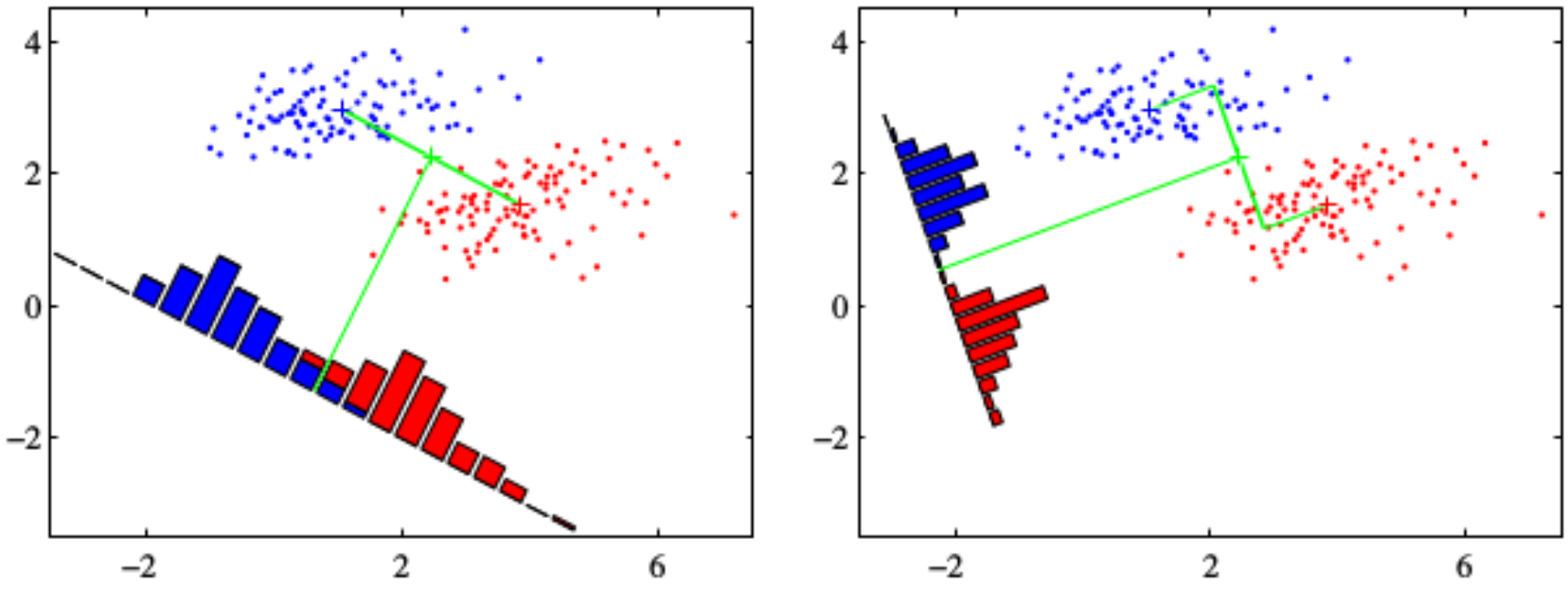
High sensitivity to outliers:



(green - logistic regression, magenta: least squares)

Attempt 2: Fisher linear discriminant

Idea: choose direction of projection in such a way that distance between centers is big, but variance for each class is small.



Fisher linear discriminant

$$m_{-1} = \frac{1}{N_{-1}} \sum_{n \in C_{-1}} x_n; m_1 = \frac{1}{N_1} \sum_{n \in C_1} x_n;$$

$$\hat{m}_{-1} = w^T m_{-1}; \hat{m}_1 = w^T m_1;$$

$$\hat{x_n} = w^T x_n$$

$$s_k^2 = \sum_{n \in C_k} (\hat{x_n} - \hat{m}_k)^2$$

Fisher linear discriminant solution

Aim: find w that maximizes

$$J(w) = \frac{(\hat{m_{-1}} - \hat{m_1})^2}{s_{-1}^2 + s_1^2}$$

$$S_{within} = \sum_{n \in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \in C_{-1}} (x_n - m_{-1})(x_n - m_{-1})^T$$

Optimal w:

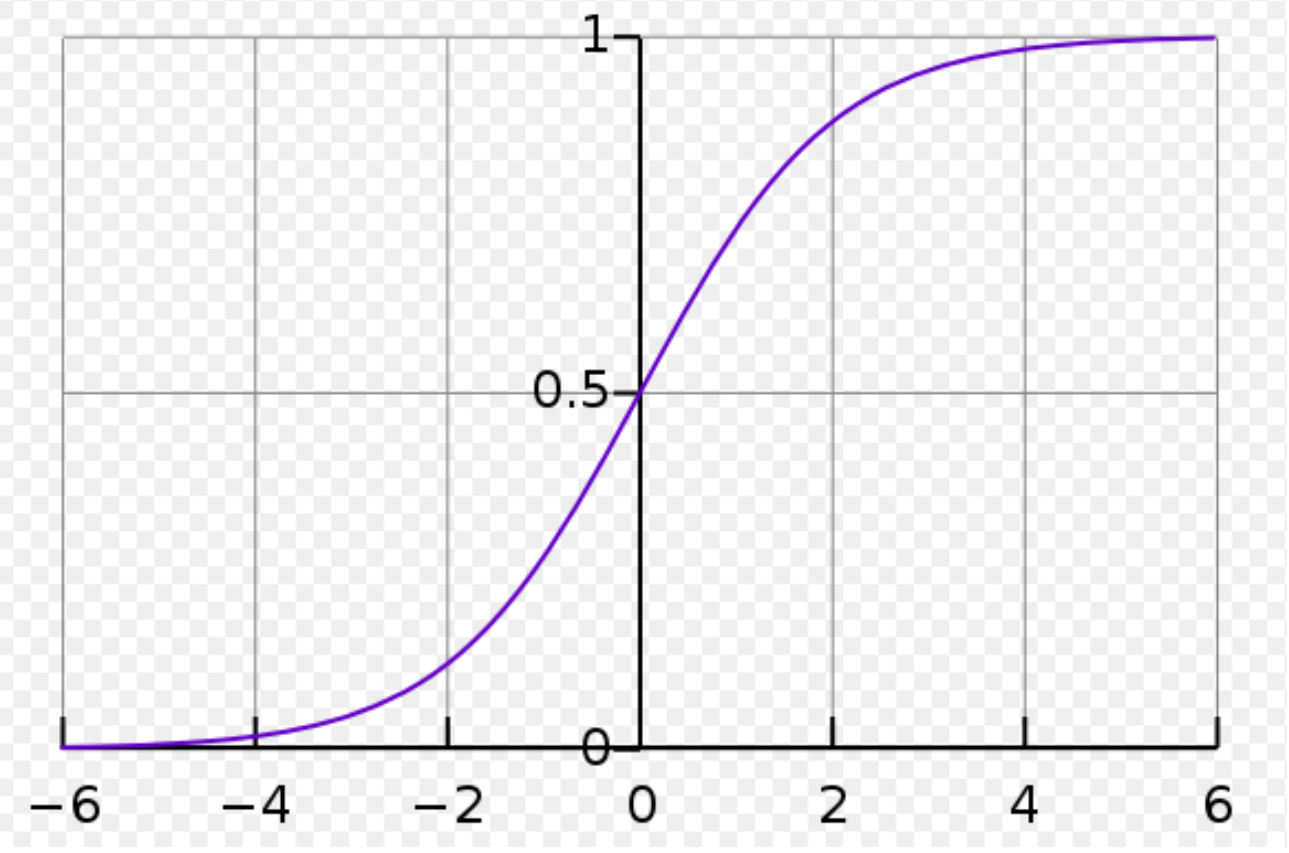
$$w_* = S_w^{-1}(m_1 - m_{-1})$$

Problems with Fisher linear discriminant

Let $N = N_1 + N_{-1}$ be total number of samples.
If we take the targets $\{N/N_1, -N/N_2\}$ and apply least squares for classification, it becomes equivalent to Fisher's discriminant!
So Fisher linear disciminant has the same problem as least squares solution: sensitivity to outliers.

Sigmoid (logistic) function

$$\sigma(x) = \frac{1}{1 + exp(-x)}$$



Properties of sigmoid function

$$\sigma(-x) = 1 - \sigma(x)$$

$$\sigma'(x) = \sigma(x)(1 - \sigma(x))$$

$$\lim_{x \to \infty} \sigma(x) = 1$$

$$\lim_{x \to -\infty} \sigma(x) = 0$$

Binary logistic regression

We want to find w such that:
$w^T x_i > 0$ for $t_i = 1$
$w^T x_i < 0$ for $t_i = -1$
So we want:
$\sigma(w^T x_i) \approx 1$ if $t_i = 1$
$\sigma(w^T x_i) \approx 0$ if $t_i = -1$
Taking all together, we want:
$\sigma(t_i w^T x_i) \to max$
for each element x_i
So we want to find w that maximizes product:

$$F(w) = \prod_{i=1}^{N} \sigma(t_i w^T x_i) \to max$$

$$E(w) = -ln(F(w)) = -\sum_{i=1}^{N} ln(\sigma(t_i w^T x_i)) \to min$$

Probabilistic interpretation of logistic output

Since $\sigma(x) \in (0, 1)$, we can interpret $\sigma(w^T x)$ as probability that $x \in C_1$

$$p(x \in C_1) = \sigma(w^T x)$$

$$p(x \in C_{-1}) = 1 - p(x \in C_1) = 1 - \sigma(w^T x) = \sigma(-w^T x)$$

Then maximization of $F(w)$ corresponds to finding the most likelihood solution under assumption of independence of labels.

Properties of function G(w)

$$G(w) = -ln(F(w)) = -\sum_{i=1}^{N} ln(\sigma(t_i w^T x_i)) \rightarrow min$$

- G(w) does not have closed-form minima
- G(w) is continuous, differentiable.
- G(w) is convex

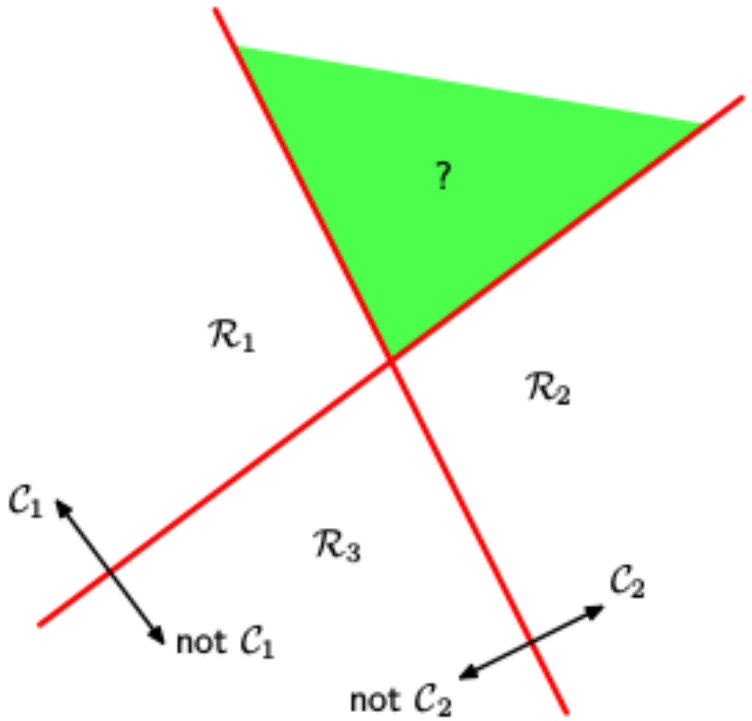
If we add quadratic regularization:

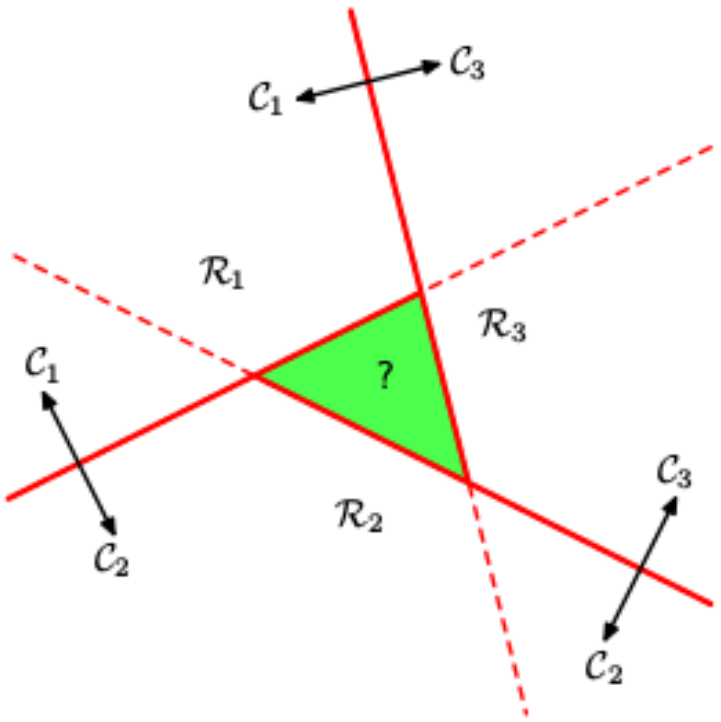
$$E(w) = -ln(F(w)) = -\sum_{i=1}^{N} ln(\sigma(t_i w^T x_i)) + \lambda \sum_{j=0}^{M} w_j^2 \rightarrow min$$

then $E(w)$ becomes strongly convex and has **exactly one minima**. This error function is called **binary cross-entropy error function**.

Multiclass classification

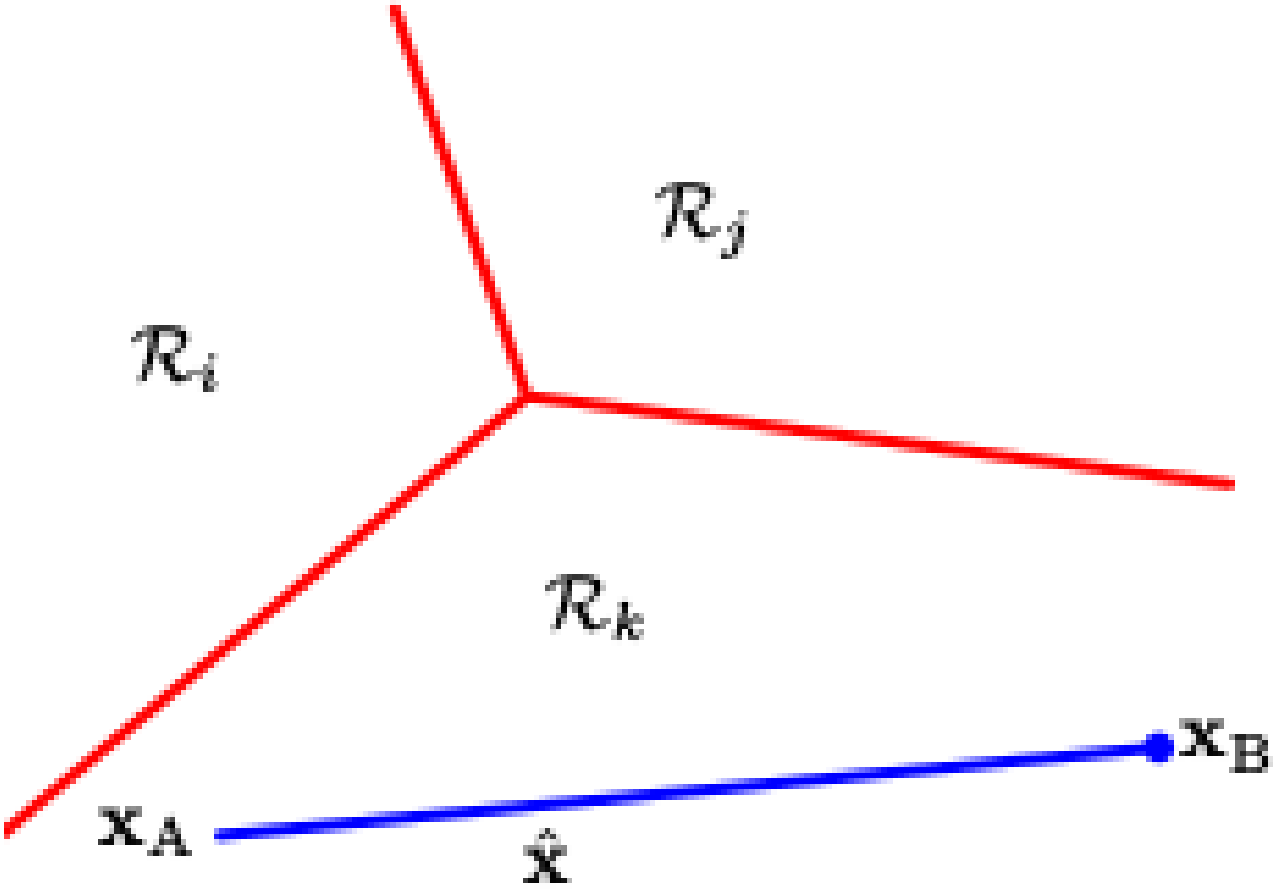
Trying to construct multiclass classifier from binary using: one-versus-all and pairwise classifiers





Better multiclass approach

- Construct separate function $f_k(x, w_k)$ for each class.
- For new input x, for each $l = 1...K$, evaluate $f_k(x, w_k)$
- Assign x to the class for which $k = argmax(f_k(x, w_k))$



General process of multiclass classification

- Construct K parameteric functions $f_k(x, w_k)$
- Find coefficients $w_1 ... w_K$ such that if $x_i \in C_j$ then $f_j(x_i, w_j)$ is maximal among $f_j(x_i, w_j)$ (for each training element)

Softmax

$$softmax_j(x) = \frac{exp(x_j)}{\sum_{j=1}^{M} exp(x_j)}$$

Properties of softmax function:

$$0 < softmax_j(x) < 1$$

$$\sum_{j=1}^{M} softmax_j(x) = 1$$

$$\frac{\partial softmax_k}{\partial x_j} = softmax_k(x)(I_{jk} - softmax_j(x))$$

Multiclass logistic regression

$$f_k(x, w_k) = w_k^T x$$

In matrix notation:

$$f(x, W) = Wx$$

$$y(x) = softmax(f(x, W)) = softmax(Wx)$$

Target vectors:

$$t_i = (0, 0...0, 1, 0...0)$$

Error function to minimize:

$$E(w) = -\sum_{n=1}^{K}\sum_{k=1}^{K} t_{nk} ln(y_{nk}) \rightarrow min$$

Multiclass logistic regression properties

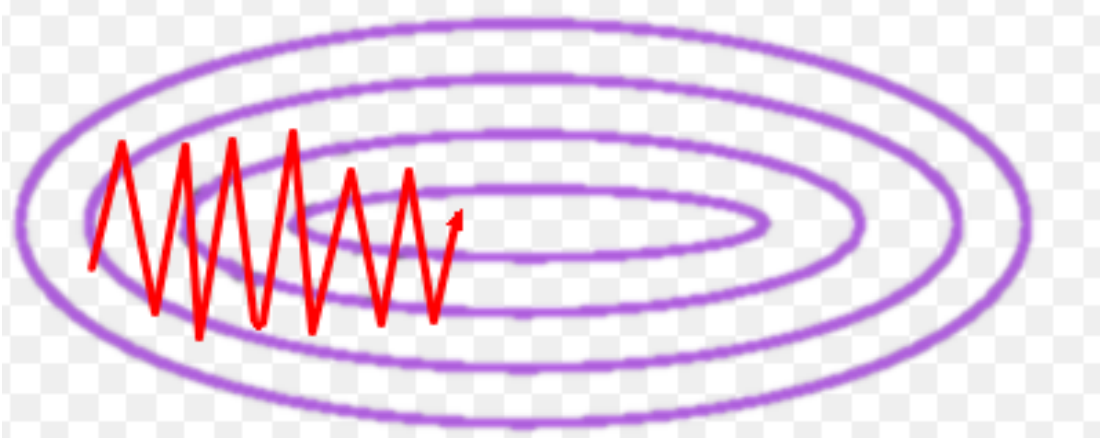
- Error function is called multiclass cross-entropy function
- E(w) is convex
- With quadratic regularization E(w) has exactly one minima
- Output is interpreted as probabilities that x belongs to corresponding classes

How to train models

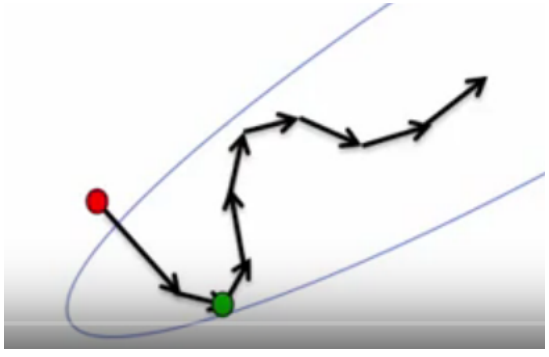
Since we can't find closed-form solutions for w, we need to use iterative schemes.
Aim: find w minimizing $E(w)$:
Gradient descent:

$$w_{k+1} = w_k - \alpha_k \nabla E(w_k)$$

Problems with gradient descent:



Momentum



Momentum term speeds up convergence:

$$w_{k+1} = w_k - \alpha_k \nabla E(w_k) + \mu_k (w_k - w_{k-1})$$

Oscilating directions are supressed.
Start with small μ, slowly increase it to values of order 0.99.

Nesterov momentum

Firstly move in momentum direction, and correct the point with gradient descent step.

$$x_{k+1} = w_k + \mu_k(w_k - w_{k-1})$$

$$w_{k+1} = x_{k+1} - \alpha_k \nabla E(x_{k+1})$$

Explanation: correct mistakes after the guess.
Shows slightly better results than usual momentum.

Gradient for logistic regression

Lin. regression	Binary log. regression	Multiclass log. regression
$t \in R$	$t \in \{-1, 1\}$	$t = (0, ..1...0)$
$y = w^T x$	$y = \sigma(tw^T x)$	$y = softmax(Wx)$
$\nabla w = (y - t)x^T$	$\nabla w = (y - 1)x^T$	$\nabla W = (y - t)x^T$