Lecture 3

Linear classifiers

Intellectual system
(Machine Learning)

Andrey Filchenkov

26.09.2017

Lecture plan

- Linear classification problem
- Gradient descent
- Heuristics for gradient descent
- Classifier performance measures

- The presentation is prepared with materials of the K.V. Vorontsov's course "Machine Leaning".
- Slides are available online: **goo.gl/fDBgMq**

Lecture plan

- Linear classification problem
- Gradient descent
- Heuristics for gradient descent
- Classifier performance measures

Problem formulation

Constraint: $Y = \{-1, +1\}$

$T^\ell = \{(x_i\ , y_i)\}_{i=1}^{\ell}$ is given

Find classifier $a_w(x, T^\ell) = \text{sign}\big(f(x, w)\big)$.

$f(x, w)$ is a discernment function,

w is a parameter vector.

Key hypothesis: objects are (well-)separable.

Main idea: search among separating surfaces described with $f(x, w) = 0$.

Margin

Margin of object x_i:

$$M_i(w) = y_i f(x_i, w),$$

$M_i(w) < 0$ is an evidence of misclassification.

Margin

Margin of object x_i:

$$M_i(w) = y_i f(x_i, w),$$

$M_i(w) < 0$ is an evidence of misclassification.

We have already defined **margin** of object x_i as

$$M(x_i) = C_{y_i}(x_i) - \max_{y \in Y \setminus \{y_i\}} C_y(x_i),$$

where $C_y(u) = \sum_{i=1}^{\ell} [y(u,i) = y] w(i,u)$, $w(i,u)$ is function of u's ith neighbor importance.

What is their relation?

Empirical risk

Empirical risk:

$$Q\left(a_w, T^\ell\right) = Q(w) = \sum_i^\ell [M_i(w) < 0],$$

it is just the number of errors.

The function is not smooth, so it is hard to find optima.

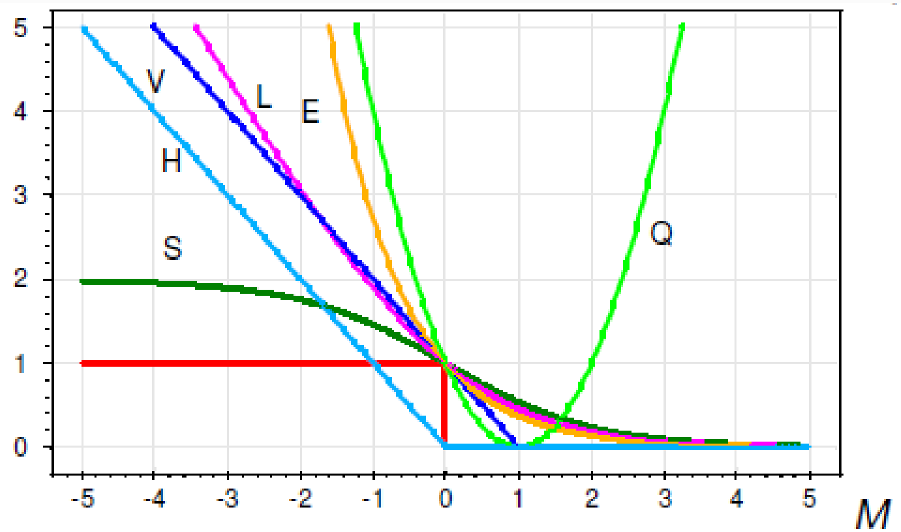
Approximation:

$$\tilde{Q}(w) = \sum_i^\ell L(M_i(w)),$$

where $L(M_i(w)) = L(a_w\left(x_i, T^\ell\right), x_i)$ is a loss function.

Loss function

We want L to be non-negative, non-increasing, and smooth:



$H(M) = (-M)_+$ — piecewise linear (Hebb's rule);

$V(M) = (1 - M)_+$ — piecewise linear (SVM);

$L(M) = \log_2(1 + e^{-M})$ — logarithmic (LR);

$Q(M) = (1 - M)^2$ — square (LDA);

$S(M) = 2(1 + e^{M})^{-1}$ — sigmoid (ANN);

$E(M) = e^{-M}$ — exponential (AdaBoost).

Linear classifier

$f_j: X \to \mathbb{R}, j = 1, \ldots, n$ are numeric features.

Linear classifier:

$$a_w(x, T^\ell) = \text{sign}\left(\sum_{i=1}^{n} w_i f_i(x) - w_0\right).$$

$w_1, \ldots w_n \in \mathbb{R}$ are feature **weights**.

Equivalent notation:

$$a_w(x, T^\ell) = \text{sign}(\langle w, x \rangle),$$

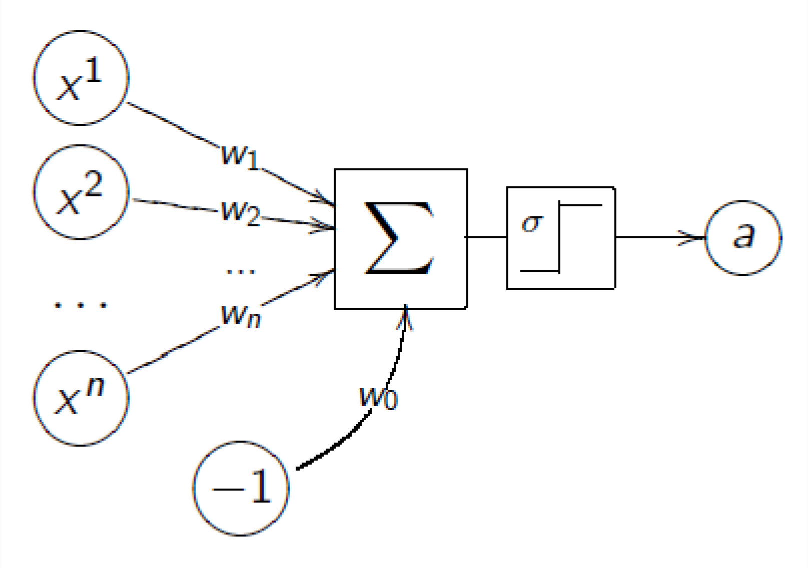
if a feature $f_0(x) = -1$ is added.

Neuron

McCulloch-Pitts neuron:

$$a_w(x, T^\ell) = \sigma\left(\sum_{i=1}^{n} w_i f_i(x) - w_0\right),$$

where σ is an activation function.



Lecture plan

- Linear classification problem
- Gradient descent
- Heuristics for gradient descent
- Classifier performance measures

Gradient descent

Empirical risk minimization problem

$$\tilde{Q}(w) = \sum_{i}^{\ell} L(M_i(w)) = \sum_{i}^{\ell} L(\langle w, x_i \rangle y_i) \to \min_w .$$

Gradient descent:

$w^{[0]} =$ **an initial guess value**;

$w^{[k+1]} = w^{[k]} - \mu \nabla Q\left(w^{[k]}\right)$,

where μ is a **gradient step**.

$w^{[k+1]} = w^{[k]} - \mu \sum_i^{\ell} L'(\langle w, x_i \rangle y_i)\, x_i y_i .$

Stochastic gradient descent

Problem is that there are too many objects, which should be estimated on each step.

Stochastic gradient descent:

$w^{[0]}$ is **an initial guess values**;

$x_{(1)}, \dots, x_{(\ell)}$ is **an objects order**;

$w^{[k+1]} = w^{[k]} - \mu L'\big(\langle w^{[k]}, x_{(k)}\rangle y_{(k)}\big) x_{(k)} y_{(k)},$

$Q^{[k+1]} = (1-\alpha) Q^{[k]} + \alpha L\big(\langle w^{[k]}, x_{(k)}\rangle y_{(k)}\big).$

Stop when values of Q and/or w do not change much.

Hebb's rule

Important special case

$$L(a_w, x) = (-\langle w, x\rangle y)_+,$$

where $(s)_+ = s \cdot [s < 0]$.

Hebb's rule (delta rule):

gradient descent step is

if $-\langle w^{[k]}, x_i\rangle y_i > 0$, then $w^{[k]} = w^{[k]} + \mu x_i y_i$.

Rosenblatt perceptron:

$$w^{[k]} = w^{[k]} + \mu(\mathrm{sign}(\langle w, x_i\rangle) - y_i)x_i$$

(the same, when $Y = \{0,1\}$).

Novikov's theorem

Theorem (Novikov)

Let sample T^{ℓ} be linearly separable: $\exists \widetilde{w}, \exists \delta > 0$:

$\langle \widetilde{w}, x_i \rangle y_i > \delta$ for all $i = 1, \ldots, \ell$.

Them the stochastic gradient descent with Hebb's rule will find weight vector w, which:

- splits sample without error;
- with any initial guess $w^{[0]}$;
- with any learning rate $\mu > 0$;
- independently on objects ordering $x_{(i)}$;
- with finite numbers of changing vector w;
- if $w^{[0]} = 0$, then the number of changes in vector w is

$t_{\max} \leq \frac{1}{\delta^2} \max ||x_j||.$

Lecture plan

- Linear classification problem
- Gradient descent
- Heuristics for gradient descent
- Classifier performance measures

Heuristics for initial guesses

- $w_j = 0$ for all $j = 0, \ldots, n$;
- small random values: $w_j \in \left[-\frac{1}{2n}, \frac{1}{2n}\right]$;
- $w_j = \frac{\langle y, f_j \rangle}{\langle f_j, f_j \rangle}$;
- learn it with a small random subsample;
- multiply runs with different initial guesses.

Heuristics for object ordering

- take objects from different classes by turns;
- take misclassified objects more frequently;
- do not take “good” object, such that $M_i > \kappa_+$;
- do not take noisy objects, such that $M_i < \kappa_-$.

Heuristics for gradient descent step

- Convergence is achieved for convex functions when

$$\mu^{[k]} \to 0, \sum \mu^{[k]} = \infty, \sum \left(\mu^{[k]}\right)^2 < \infty.$$

- **Steepest gradient descent**:

$$Q\left(w^{[k]} - \mu^{[k]} \nabla Q\left(w^{[k]}\right)\right) \to \min_{\mu^{[k]}}.$$

- Steps for “jog of” local minima.

SG algorithm discussion

Advantages:

- it is easy to implement;
- it is easy to generalize for any f and L;
- dynamical learning;
- can handle small samples.

Disadvantages:

- slow convergence or even divergence is possible;
- can stuck in local minima;
- proper heuristic choice is very important;
- overfitting.

Regularization

Key hypothesis: w "swings" during overfitting

Main idea: clip w norm.

Add regularization penalty for weights norm:

$$Q_\tau(a_w, T^\ell) = Q(a_w, T^\ell) + \frac{\tau}{2}\|w\|^2 \to \min_w.$$

For gradient:

$$\nabla Q_\tau(w) = \nabla Q(w) + \tau w,$$
$$w^{[k+1]} = w^{[k]}(1 - \mu\tau) - \mu\nabla Q(w).$$

Lecture plan

- Linear classification problem
- Gradient descent
- Heuristics for gradient descent
- Classifier performance measures

Contingency table

	Positive	Negative
Classified as positive	**TP** = True Positive	**FP** = False Positive
Classified as negative	**FN** = False Negative	**TN** = True Negative

FN in math. stat. – I type error

FP in math. stat. – II type error

P = **TP** + **FN** – number of positive examples

N = **FP** + **TN** – number of negative examples

Some definitions

Sensitivity or **Recall**:

$$\text{Recall} = \text{TPR} = \frac{\text{TP}}{\text{P}}$$

Specificity:

$$\text{SPC} = \frac{\text{TN}}{\text{N}}$$

Precision:

$$\text{Precision} = \text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Accuracy:

$$\text{Accuracy} = \text{ACC} = \frac{\text{TP} + \text{TN}}{\text{P} + \text{N}}$$

F-measure

We will not lose much in accuracy performing badly on small classes.

F_β-measure

$$F = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

F_1-measure:

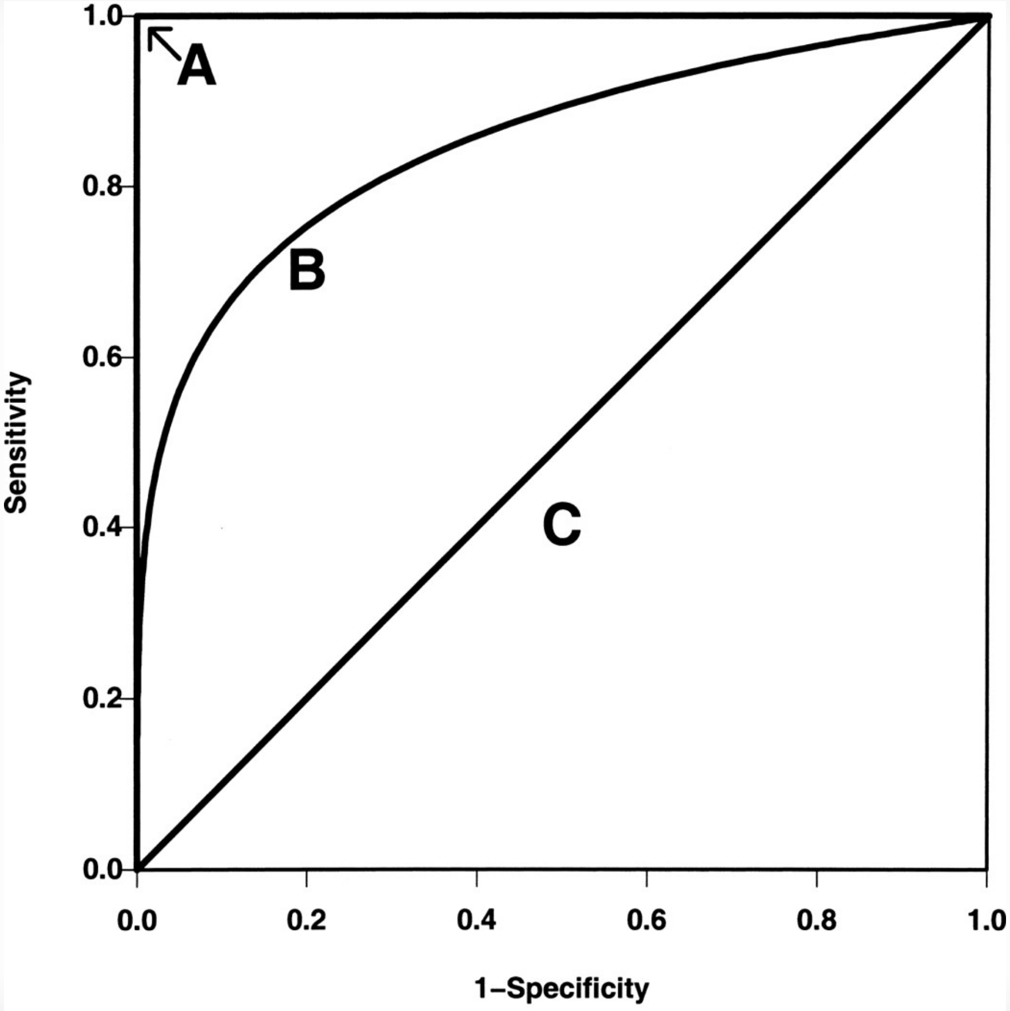
$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

ROC-curve

A is the best algorithm

B is a typical algorithm

C is the worst algorithm



AUC

Area under the curve (AUC) is area under the ROC-curve.

Connected with Mann-Whitney U.

Out of date measure.

Multiclass case

- One vs one classification
- One vs all (one vs rest) classification
- Hierarchical classification
- Confusion matrix