ITMO

# Classification

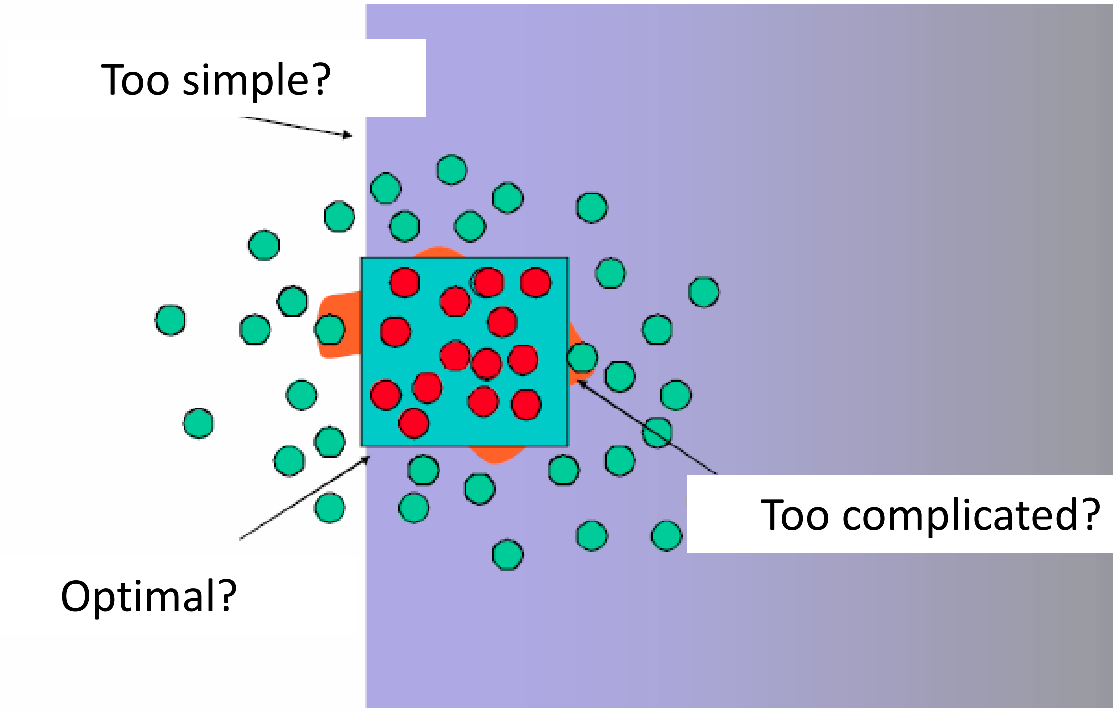
## Computer Vision

## Algorithm Generalization Ability

## Estimation of Generalization Ability

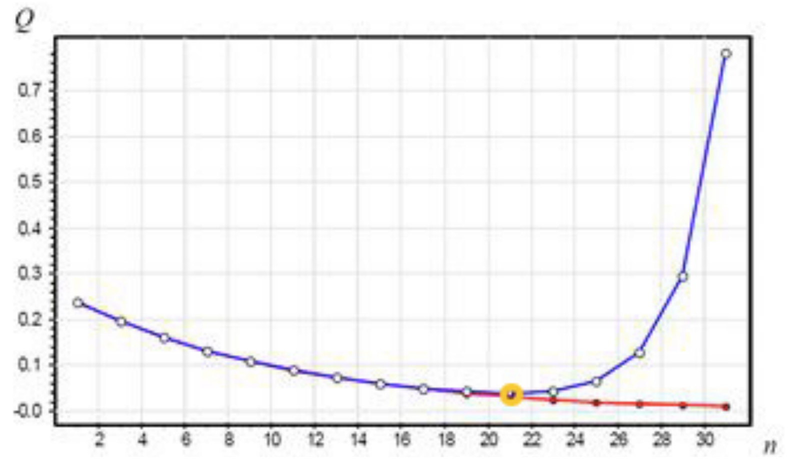
- **Vapnik–Chervonenkis theory (VC)**
  - Estimate of the complexity of a parametric family of functions;
  - Assessing the quality of an algorithm through empirical risk and model complexity.
- **Main idea:** choosing the simplest model among sufficiently accurate ones.
  - Let there be a sequence of nested parametric families of increasing complexity: $F_1 \subset F_2 \subset \cdots \subset F_h = F$.
  - It is necessary to choose a family with minimal complexity that provides the desired accuracy.

## Illustration



# Practical Conclusion

- A balance is required between the complexity of the model, which provides low empirical risk, and the simplicity, which provides the ability to generalize.



The red line is a training error,
blue line is a validation error,
marked yellow dot is an optimum of difficulty.

## Total Risk Estimation

- Minimizing the *total risk* is the main goal.
- However, it cannot be computed because computations are required on an unbounded set.

$$R(f, X) = P_{X_m}(f(x) \neq y) = \int_x P(x)[f(x) \neq y]dx \,.$$

## Holding

- Let us estimate the total risk of an error on some finite subset $X^c$ that does not intersect with the training sample:

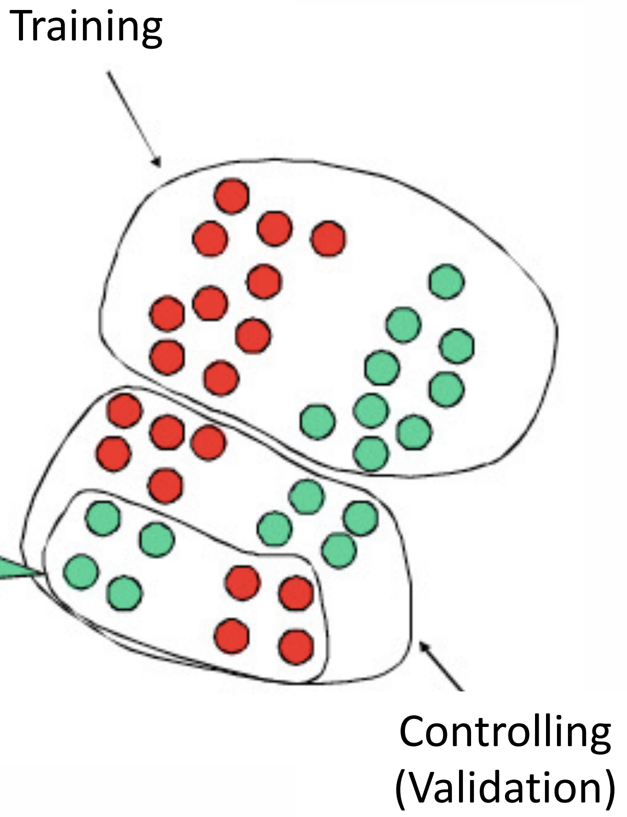
$$R(f, X) \sim P(f(x) \neq y | X^c) = \frac{1}{c} \sum_{j=1}^{c} [f(x_j) \neq y_j]$$

- Let there be a data set $X^k = \{x_1, \dots, x_k\}$ with known answers.
- Let's union $X^l \cup X^c = X^k$: $X^l \cap X^c = \emptyset$.
- We will use for training $X^l$, and for control $X^c$:

$$P(f(x) \neq y) \approx P(f(x) \neq y | X^c).$$

## Holding

- Characteristics:
  1. Quick and easy calculation.
  2. Some complex cases may completely fall into only one of the samples, and then the error estimate will be biased.



The error will occur not through the fault of the classifier, but due to splitting

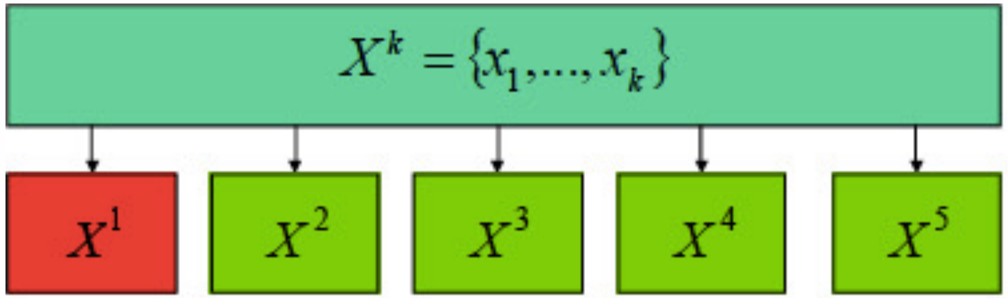
## Sliding Control

- Divide the sample into $d$ non-overlapping parts and alternately use one of them for control, and the rest for training.
- Breaking down: $\{X^i\}^d \colon X^i \cap X^j = \emptyset, i \neq j, \bigcup_{i=1}^{d} X^i = X^k$.
- Calculate the approximate risk:

$$P(f\left(X^k\right) = y^*) \approx \frac{1}{d} \sum_{i=1}^{d} P(f(X^i) \neq y^* | \bigcup_{i \neq j} X^i).$$

## Sliding Control

- The result is considered as the average error over all iterations.



🟥 Control 🟩 Training

## Sliding Control

- Properties:
  - In the limit, the approximate risk will be equal to the total risk.
  - Each case will be present in the control sample.
  - The training samples will overlap heavily (the more segments, the more overlap).
  - If one group of “complex precedents” falls completely into one segment, then the estimation will be biased.

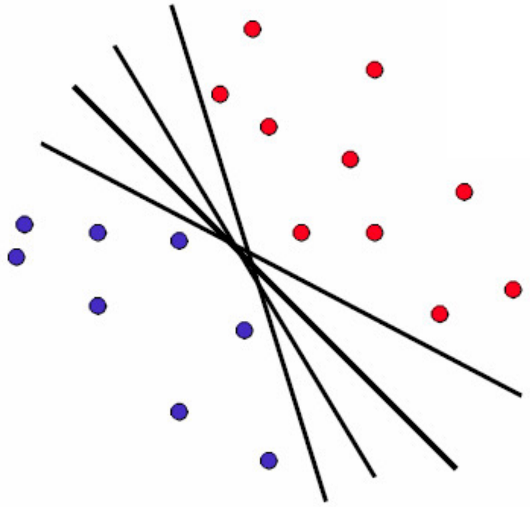
## Cross Sliding Control

- **CV (cross validation)**
- 5-2 cross-validation:
  - Divide the sample randomly in half.
  - We will train the algorithm on one half, test on the other and vice versa.
  - Repeat this experiment five times and average the result.
- **Property:** each of the use cases will participate in control (validation) samples in each of the 5 stages.

## Classification

## Linear Classifier

- Find a linear function (hyperplane) and separate the positive $\{y = +1\}$ and negative $\{y = -1\}$ examples:
  - $x_i$ positive : $x_i \cdot w + b \geq 0$,
  - $x_i$ negative: $x_i \cdot w + b < 0$.



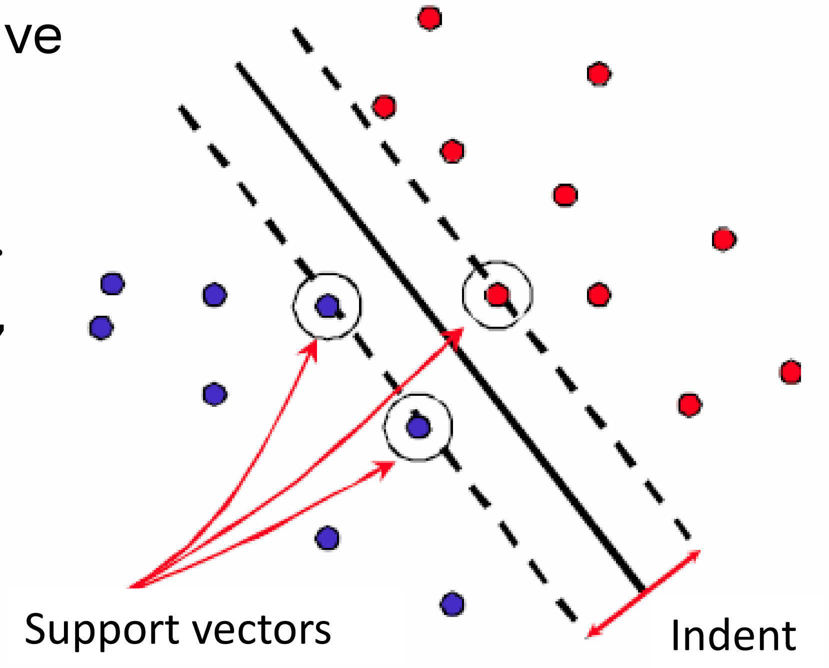
What is the best hyperplane?

## Support Vector Machine (SVM)

- Based on VC theory.
- An optimal separating hyperplane is a hyperplane such that the distance from it to the nearest point from the training sample (no matter from which class) is maximum.
- The optimal separating hyperplane maximizes the margin (indent):
  - the distance from it to the points from the training sample.

# Support Vector Machine (SVM)

- Find a hyperplane that maximizes the indent between positive and negative examples:
  - $x_i$ positive ($y_i = 1$): $x_i \cdot w + b \geq 1$,
  - $x_i$ negative ($y_i = -1$): $x_i \cdot w + b \leq -1$.
- For support vectors: $x_i \cdot w + b = \pm 1$,
- The distance from a point to a hyperplane is: $\frac{|x_i \cdot w + b|}{\|w\|}$,
- The indent is: $\frac{2}{\|w\|}$.



## Support Vector Machine (SVM)

- Finding a hyperplane:
  1. Maximizing $\frac{2}{\|w\|}$.
  2. Classify all data:
     - $x_i$ positive ($y_i = 1$): $x_i \cdot w + b \geq 1$,
     - $x_i$ negative ($y_i = -1$): $x_i \cdot w + b \leq -1$,
  3. Let's solve the quadratic optimization problem:
     - Minimize $\frac{1}{2} w^{\mathrm{T}} w$ on condition $y_i(w \cdot x_i + b) \geq 1$.

## Support Vector Machine (SVM)

- The problem is solved using the method of Lagrange multipliers:

$$w = \sum_i \alpha_i y_i x_i,$$

where $\alpha_i$ – trained weights (Lagrange multipliers), $x_i$ – support vectors.

- For most vectors, the weight is zero.
- All vectors for which the weight is greater than zero $w > 0$ are called support vectors.
- The solution will depend only on the support vectors.

## Support Vector Machine (SVM)

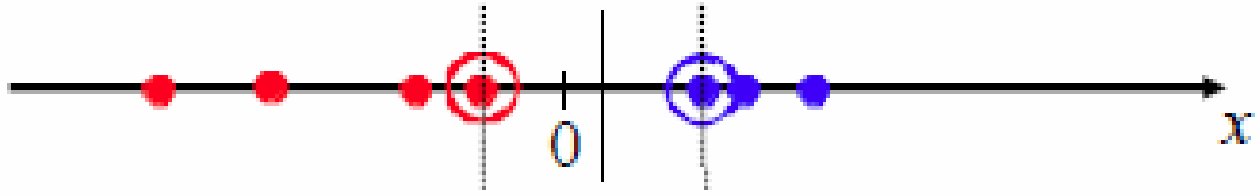
- The decision function takes the form:

$$w \cdot x + b = \sum_i \alpha_i y_i x_i \cdot x + b.$$

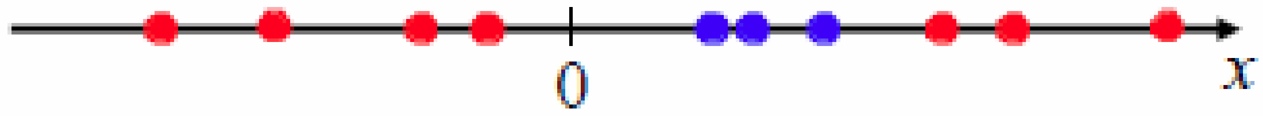
- The decision function depends on the inner products of the test vector $x$ and the support vectors $x_i$.
- The solution of the optimization problem requires the calculation of scalar products $x_i \cdot x_j$ between all pairs of vectors from the training set.

# SVM Examples

- On linearly separable data, the support vector machine works fine:



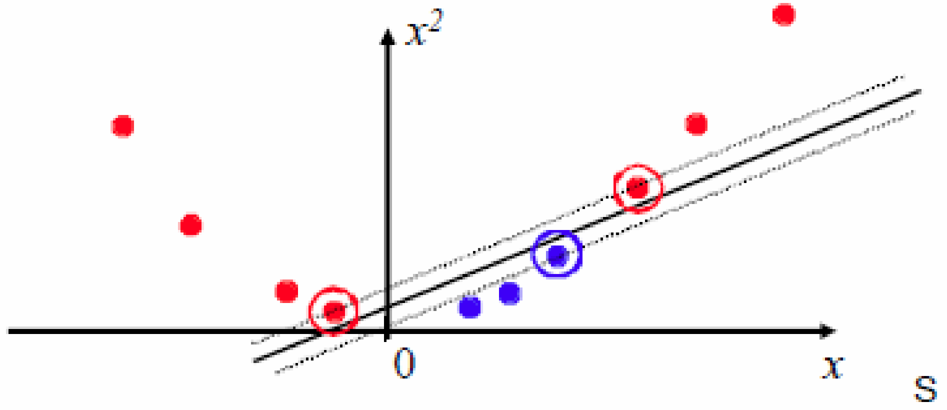
- On more complex data, the method does not work very well:



An example of linearly inseparable data

# Nonlinear SVM

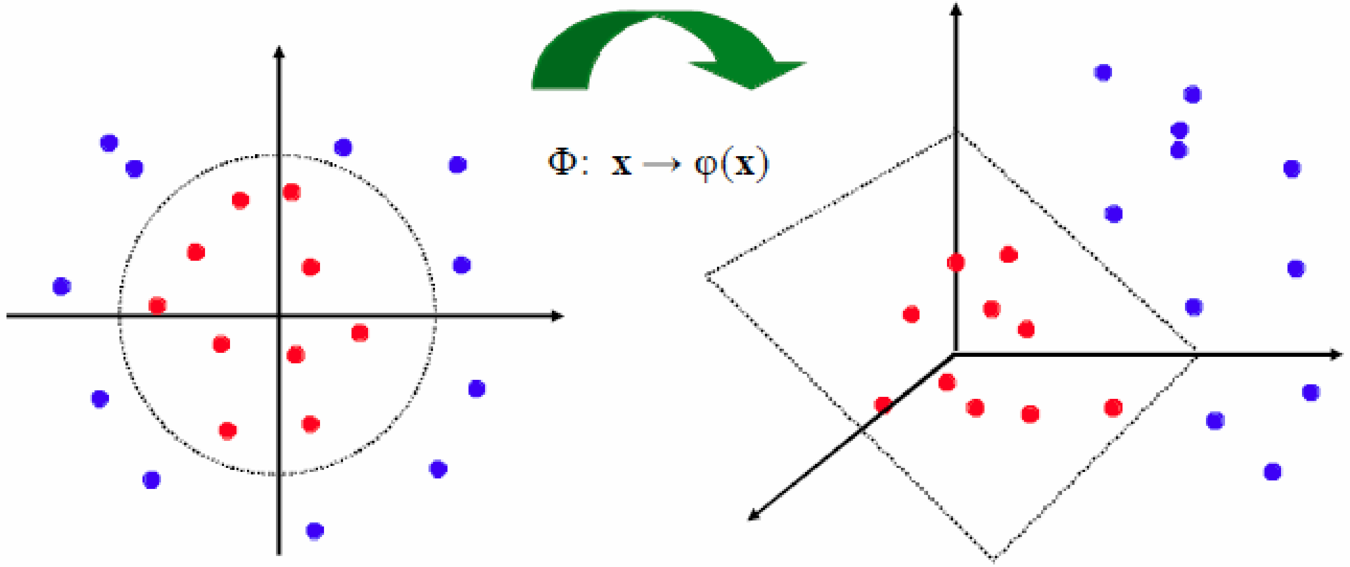
- Basic approach: mapping data to another space of higher dimension and in it their linear separation.



Data in another space

# Nonlinear SVM

- **Idea:** original parameter space mapping into some multidimensional feature space, where the training sample is linearly separable:



## Nonlinear SVM

- **Problem:** Calculating dot products in high-dimensional space is computationally difficult.
- To avoid this, there is an approach called the *kernel trick*, which was designed in 1995, in which, instead of directly calculating the transformation $\varphi(x)$, the kernel function $K$ is defined:

$$K\left(x_i, x_j\right) = \varphi(x_i) \cdot \varphi\left(x_j\right),$$

where is the matrix $K\left(x_i, x_j\right)$ – non-negative definite.

- Using this kernel, a nonlinear decision function is designed in the original space:

$$\sum_i \alpha_i y_i K(x_i, x) + b.$$

## Nonlinear SVM: an Example of Kernels

- Polynomial kernel example:

$$K(x, y) = ((x, y) + c)^d.$$

- Let be $d = 2, x = (x_1, x_2)$:

$$K(x, y) = \left((x, y) + c\right)^2 = (x_1 y_1 + x_2 y_2 + c)^2 = \varphi(x) \cdot \varphi(y),$$

where:

$$\varphi(x) = \left(x_1^2, x_2^2, \sqrt{2} x_1 x_2, \sqrt{2} c x_1, \sqrt{2} c x_2, c\right),$$

those mapping from two-dimensional space to six-dimensional.

## Nonlinear SVM Usage Algorithm

1. Create a training sample.
2. Select kernel.
3. Calculate the matrix of kernel values for each pair of examples from the training set.
4. Load matrix into SVM library to get weights and support vectors.
5. During inference, calculate the kernel values for the test sample and each support vector and take the weighted sum to obtain the value of the decision function.

## Multiclass SVM

- There is no special formulation of SVM for the case of many classes.
- In practice, the SVM for several classes is obtained by combining several two-class SVMs.
- One against all:
  - Training: training SVM for each class against all others.
  - Validation: apply all SVMs to the sample and assign the class for which the SVM gave the most reliable solution.
- One against one:
  - Training: train SVM for each pair of classes.
  - Validation: each SVM votes for classes, choose the class with the most votes.

## Properties of SVM

- **Advantages of SVM:**
  1. Lots of libraries available: http://www.kernel-machines.org/software.
  2. Powerful and flexible kernel-based approach.
  3. Works very well in practice, even for small training samples.
- **Disadvantages of SVM:**
  1. No direct multiclass methods, two-class SVMs need to be combined.
  2. Resource intensity:
     - When learning, you need to design a complete kernel matrix for all examples.
     - Training takes a long time for big tasks.

## Error Types

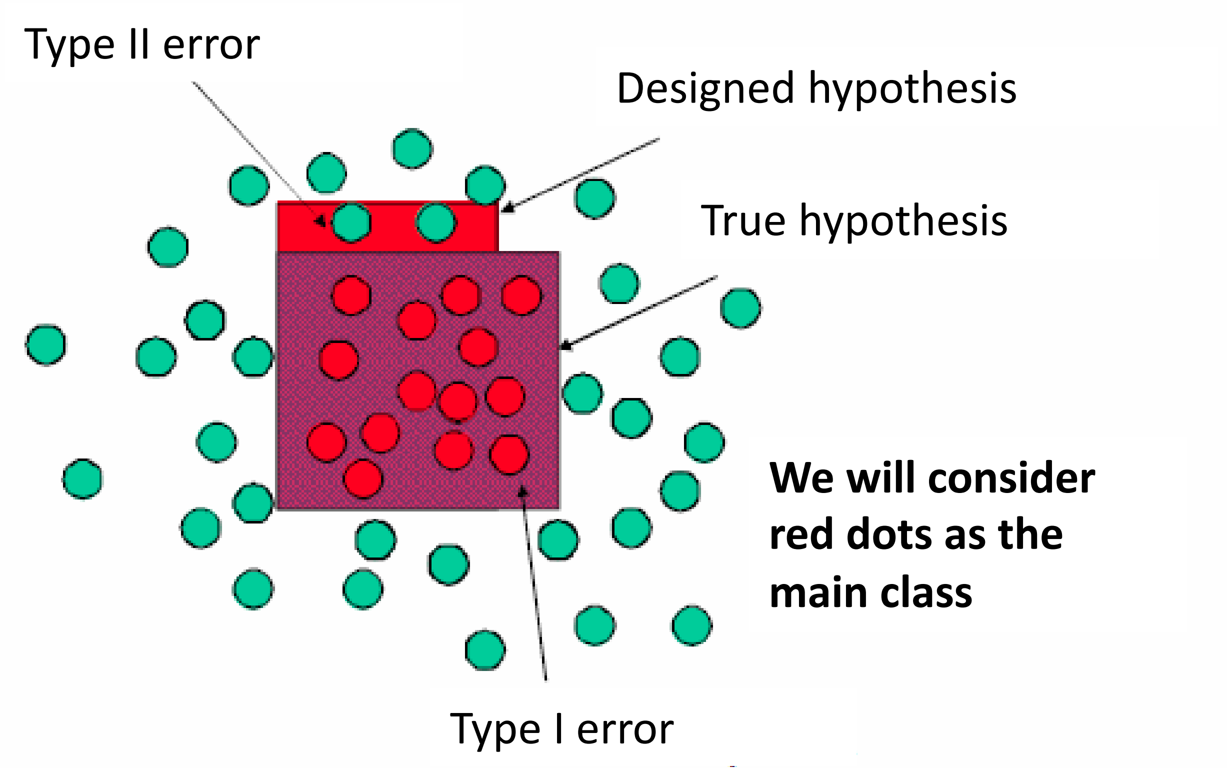
## Error Types

- If we consider the error as the probability of giving an incorrect answer, then such a measure is not fully sufficient.
    - For example, a 15% error in diagnosis may mean that 15% of patients will be considered healthy (and possibly die from lack of treatment),
    - and that 15% of healthy patients will be ill (and money for treatment will be wasted).
- In such cases, the concept of errors of the first and second type is introduced, and then they are evaluated separately.

## Error Types

- **Let there be some main class.**
  - As a rule, the main class is the class, upon detection of which some action is taken.
  - In our example, when making a diagnosis, the main class will be “sick”, and the secondary one will be “healthy”.
- **Type I error** is the probability of mistaking the primary class for a secondary one.
  - The probability of missing the desired object.
- **Type II error** is the probability of mistaking a secondary class for a primary one.
  - The probability when the background will be taken for the desired object.

## Error Types



**We will consider red dots as the main class**

## Error Types

- It is especially important to evaluate errors separately in case of strong imbalance of classes.
- For example, $P(y = +1) = 0{,}01$, $P(y = -1) = 0{,}99$.
- Then for an error of the first kind $P(f(x) = -1 | y = +1) = 0{,}5$ and a zero error of the second kind, the total error will be just:

$$P(a(x) \neq y) = 0{,}005.$$

## Error Types

- Based on the errors, the following two parameters can be distinguished:
- ***Sensitivity*** – the probability of giving the correct answer to the example of the main class:

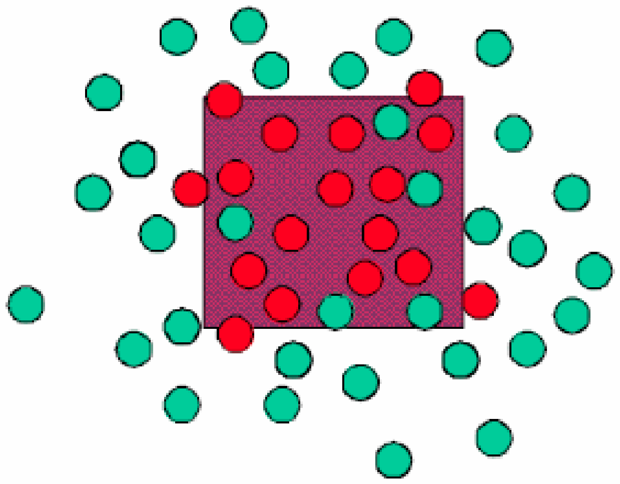
$$sensitivity = P(f(x) = y|y = +1).$$

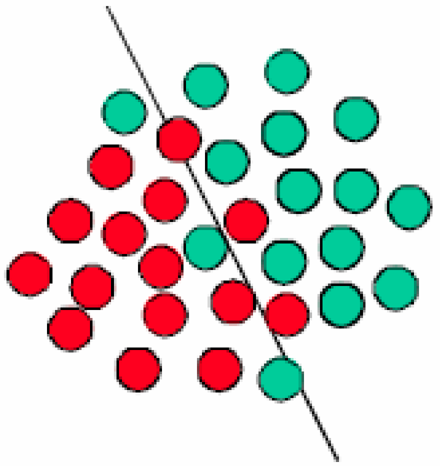
- ***Specificity*** – the probability of giving the correct answer to an example of a secondary class:

$$specificity = P(f(x) = y|y = -1).$$

## Balance Setting

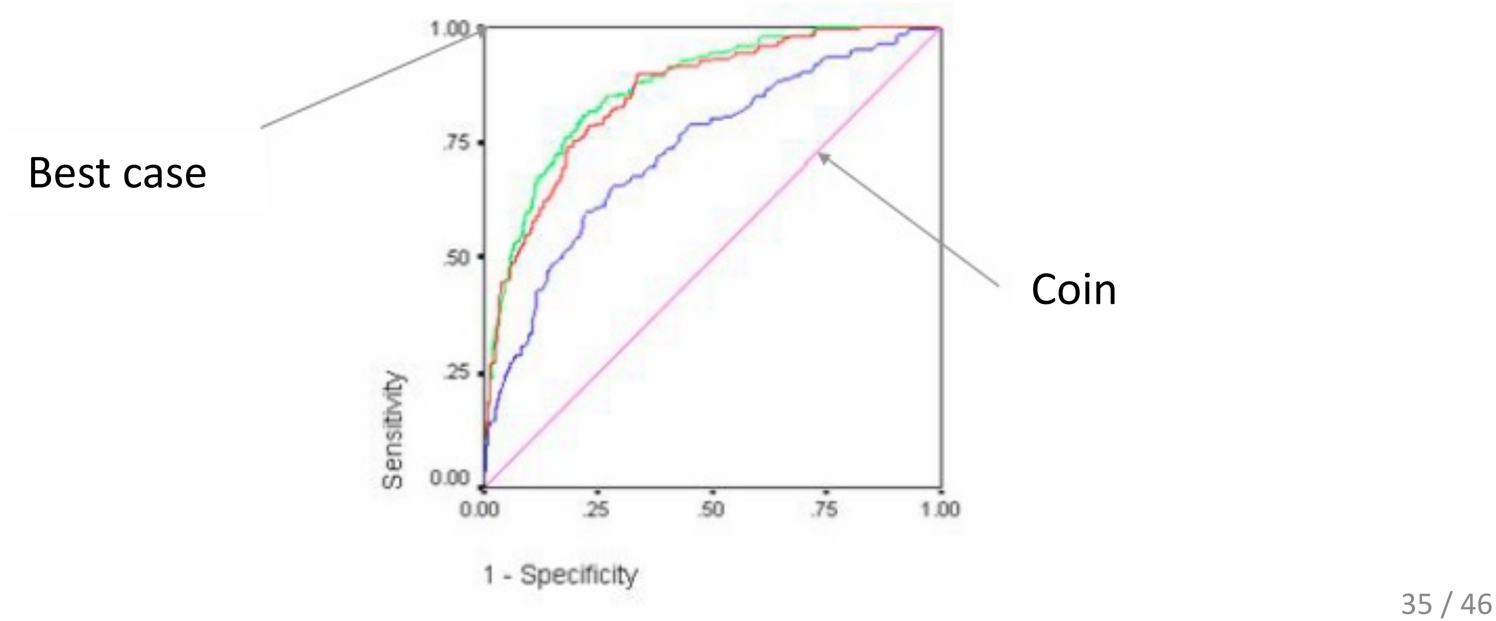
- Almost all classification algorithms have parameters, varying which you can get different levels of errors of the 1st and 2nd type.





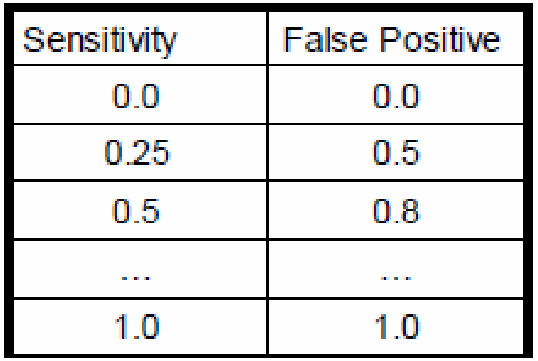
## ROC Curve

- To estimate the dependence of sensitivity on specificity, you can design an ROC curve (Receiver Operating Characteristic curve).

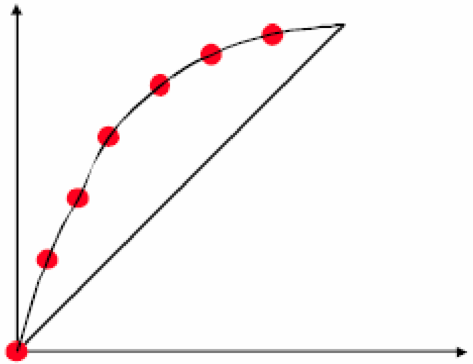


## ROC Curve

- For various values of the variable setting parameter, an error table is built.
- The parameter itself does not appear in the table.
- The classifier is built and evaluated constantly on different samples.
- Each row of the table is a point on which the curve is built.



| Sensitivity | False Positive |
|---|---|
| 0.0 | 0.0 |
| 0.25 | 0.5 |
| 0.5 | 0.8 |
| ... | ... |
| 1.0 | 1.0 |



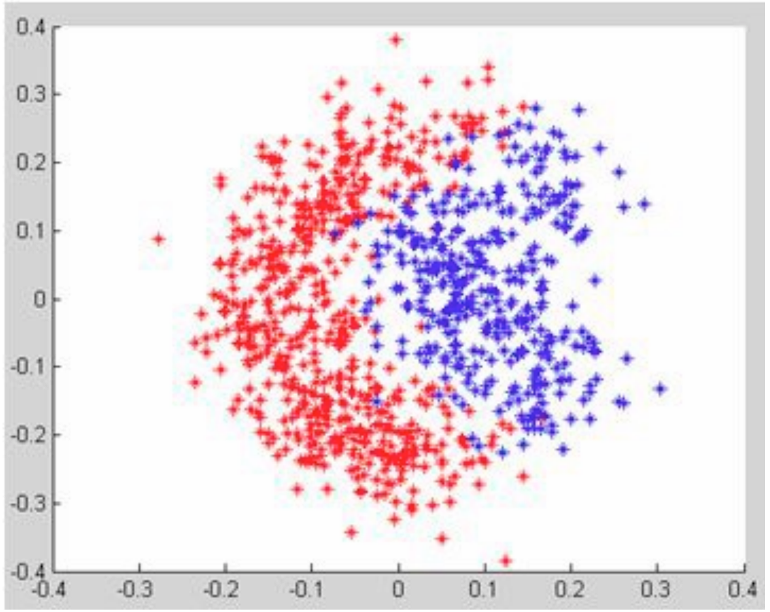
## ROC Curve

- The area under the curve gives some objective indicator of the quality of the classifier and allows you to compare different curves.
- Often, for a particular task, there is a framework for a certain type of error. Using ROC, you can analyze the performance of the current solution for compliance with the requirements.

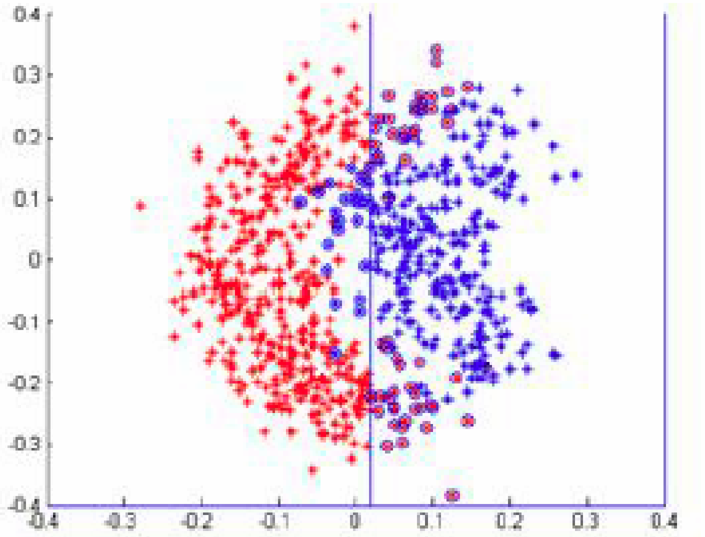
## Example

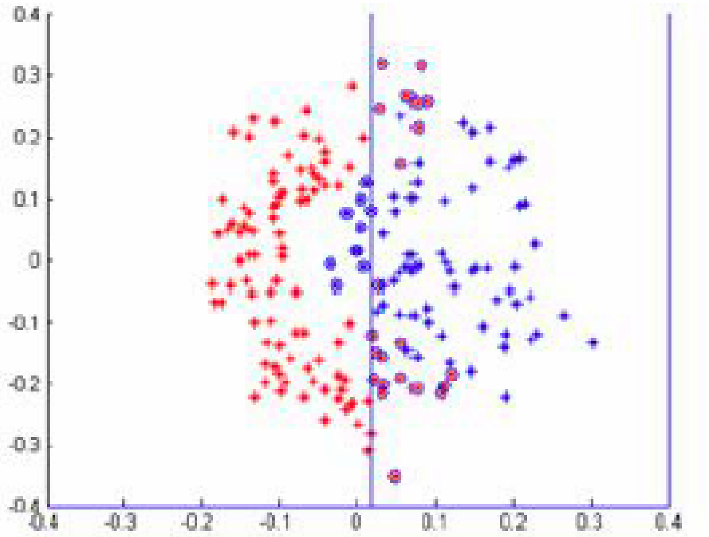
- Given some points on the plane.
- The parametric family is the threshold along the axis $x$:

$$a(x^1, x^2) = \begin{cases} +1, x_1 > \Theta \\ -1, x_2 \leq \Theta \end{cases}.$$



## Holding



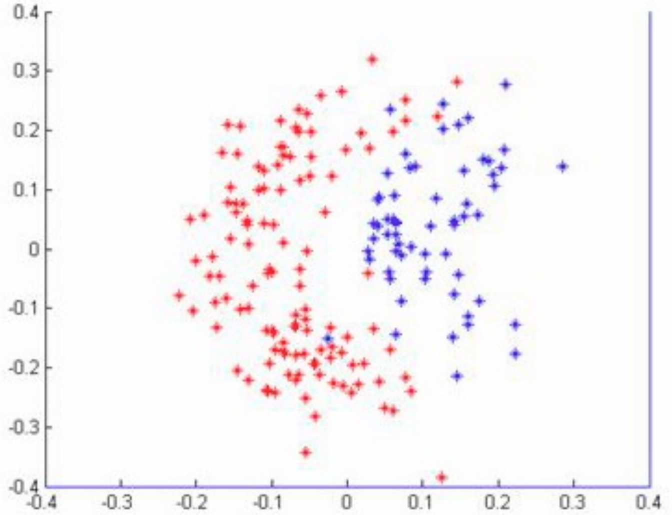


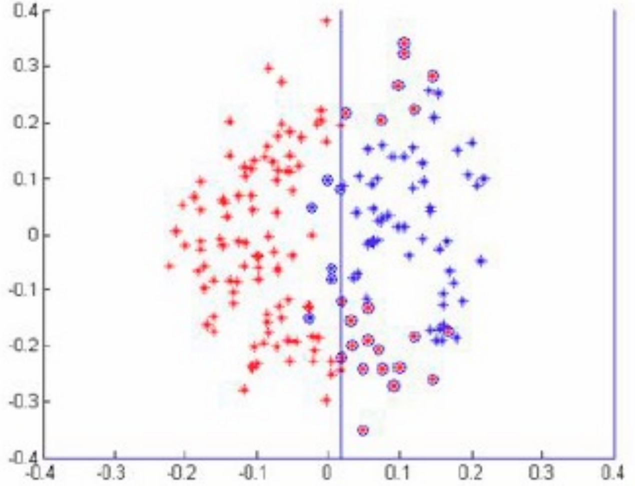
Left – training sample, error 0,1133;
Right – control (validation) sample, error 0,1433.

## Re-Holding

- Training error:
  - $\{0{,}1097; 0{,}1236; 0{,}1209; 0{,}1250; 0{,}1250\}$,
  - the average is $0{,}1208$.
- Error in control:
  - $\{0{,}1833; 0{,}1222; 0{,}1333; 0{,}1222; 0{,}1167\}$,
  - the average is $0{,}1356$.

## Sliding Control





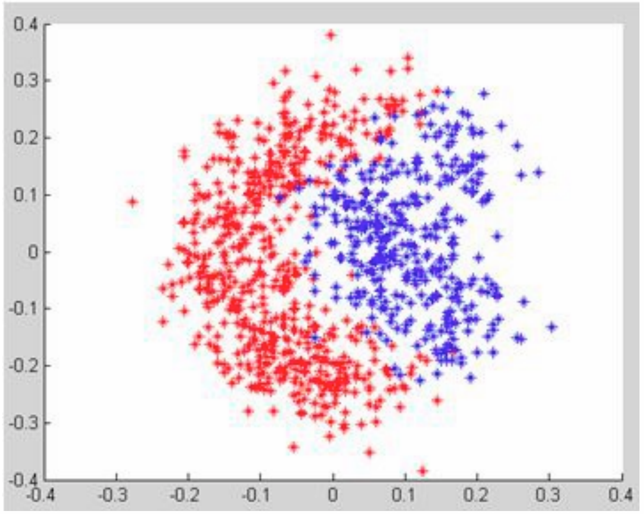
Left – training sample, error 0,1219;
Right – control sample, error 0,1367

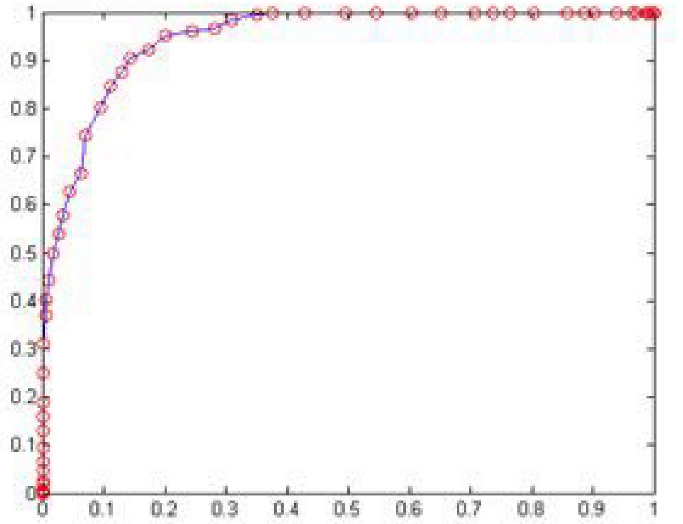
## Sliding Control

- Training error:
  - $\{0{,}1236; 0{,}1208; 0{,}1250; 0{,}1097; 0{,}1306\}$,
  - the average is $0{,}1219$.
- Error in control:
  - $\{0{,}1500; 0{,}1333; 0{,}1222; 0{,}1778; 0{,}1000\}$,
  - the average is $0{,}1367$.

## ROC Curve

- We will change the threshold, estimate the error and build a ROC table, build a curve using the table:





## ROC Curve

- General scheme for selecting classifier parameters:
  1. We take a training sample.
  2. We take different classifier parameters.
     a) For each parameter set:
        - Using 5-2 Cross Validation.
        - Train a classifier with these parameters.
        - Estimating errors.

     b) Building a ROC curve.
  3. Choosing the optimal parameters and training the classifier with them on the entire sample.

## Test on Lectures 7-8

CS3 Test

AT3 Test

THANK YOU
FOR YOUR TIME!

ITsMOre than a UNIVERSITY

s.shavetov@itmo.ru