# K-nearest neighbours

Victor Kitov
v.v.kitov@yandex.ru

# Table of Contents

# K-nearest neighbours

## Classification using k nearest neighbours

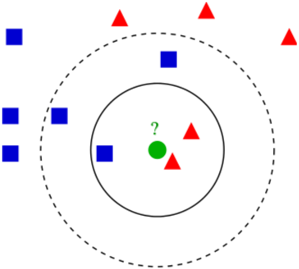
1. Find $k$ closest objects to the predicted object $x$ in the training set.
2. Associate $x$ the most frequent class among its $k$ neighbours.

- Regression case: targets of nearest neighbours are averaged
- $k = 1$: nearest neighbour algorithm[1]
- Base assumption of the method[2]:
  - similar objects yield similar outputs

[1] what will happen for $K = N$?
[2] what is simpler - to train K-NN model or to apply it?

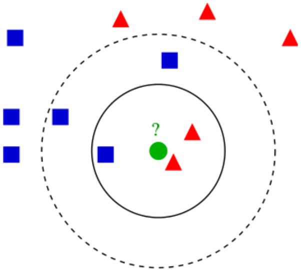
# K-NN illustration

Classification:

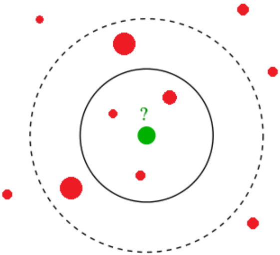


# K-NN illustration

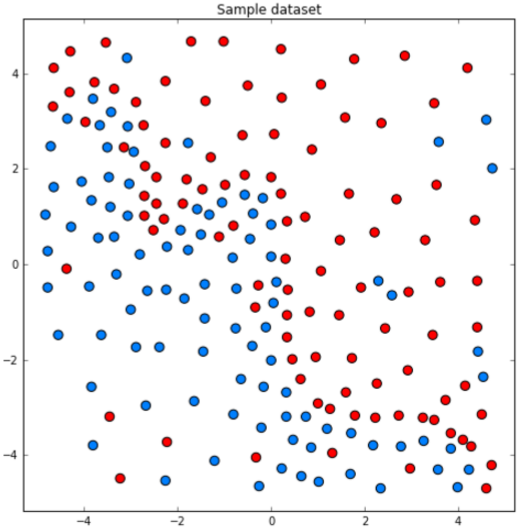
Classification:



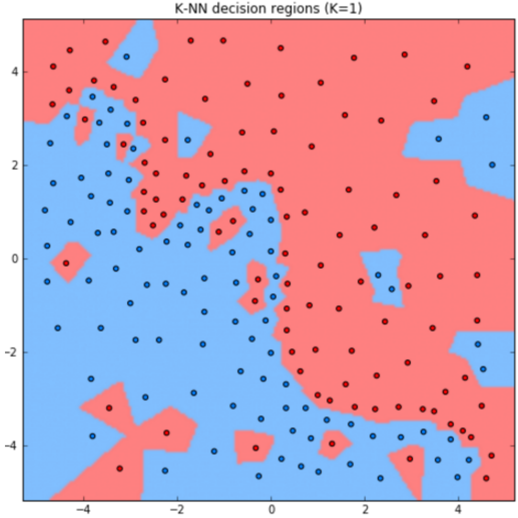

Regression:



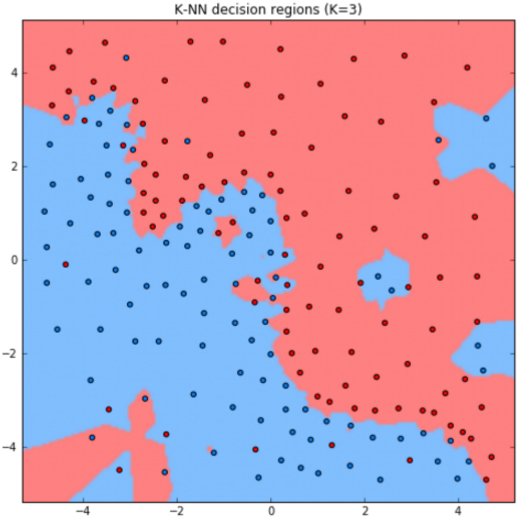
# Sample dataset



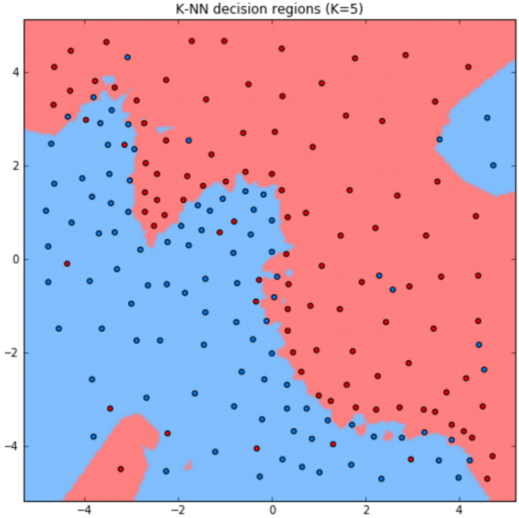
# Example: K-NN classification



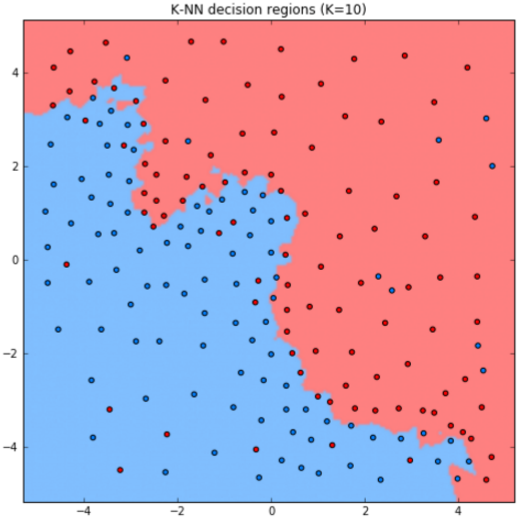
# Example: K-NN classification



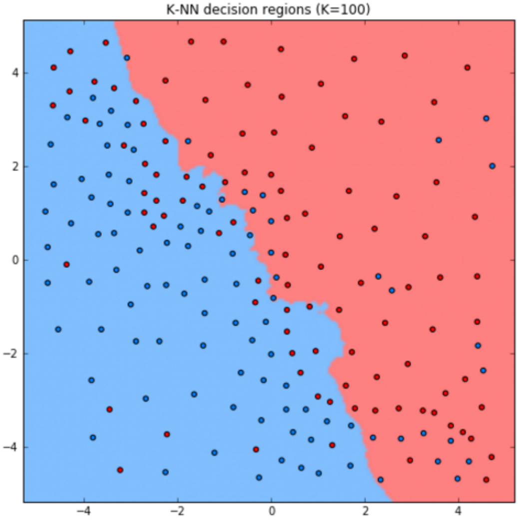
# Example: K-NN classification



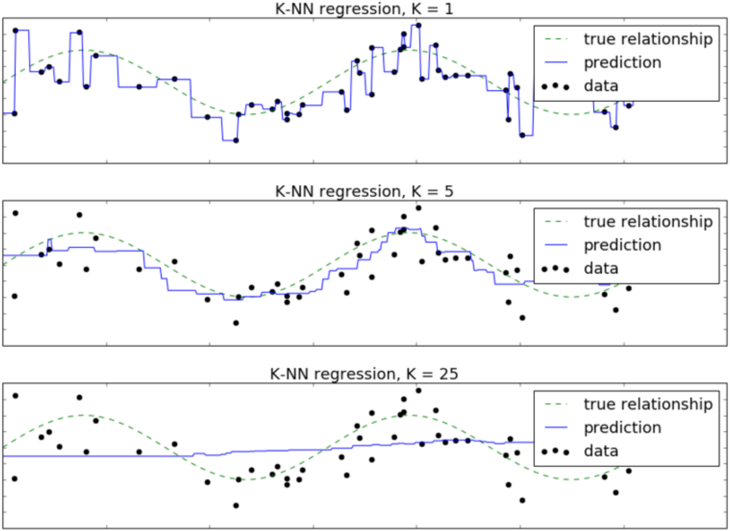
# Example: K-NN classification



# Example: K-NN classification



# Example: K-NN regression



# Parameters of the method

- Parameters:
  - the number of nearest neighbours $K$
  - distance metric $\rho(x, y)$
- Modifications:
  - forecast rejection option[3]
  - variable $K$[4]

[3] Propose a rule, under what conditions to apply rejection in a) classification b) regression

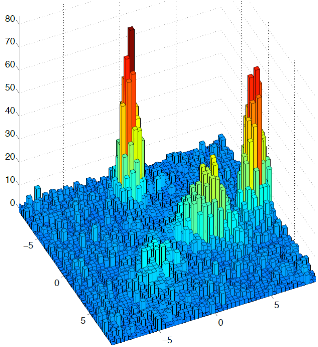
[4] Propose a method of K-NN with adaptive variable K in different parts of the feature space

# Properties

- Advantages:
  - only similarity between objects is needed, not exact feature values.
    - so it may be applied to objects with arbitrary complex feature description
  - simple to implement
  - interpretable (case based reasoning)
  - does not need training
    - may be applied in online scenarios
    - Cross-validation may be replaced with LOO.
- Disadvantages:
  - slow classification with complexity $O(N)$
  - accuracy deteriorates with the increase of feature space dimensionality

# The curse of dimensionality

- The curse of dimensionality: with growing $D$ data distribution becomes sparse and insufficient.
- Example: histogram estimation[5]




[5] At what rate should training size grow with increase of $D$ to compensate curse of dimensionality?

# Curse of dimensionality

- Case of K-nearest neigbours:
  - assumption: objects are distributed uniformly in feature space
  - ball of radius $R$ has volume $V(R) = CR^D$, where $C = \frac{\pi^{D/2}}{\Gamma(D/2+1)}$.
  - ratio of volumes of balls with radius $R-\varepsilon$ and $R$:

$$\frac{V(R-\varepsilon)}{V(R)} = \left(\frac{R-\varepsilon}{R}\right)^D \xrightarrow{D\to\infty} 0$$

  - most of volume concentrates on the border of the ball, so there lie the nearest neighbours.
  - nearest neighbours stop being close by distance
- Good news: in real tasks the true dimensionality of the data is often less than $D$ and objects belong to the manifold with smaller dimensionality.

# Dealing with similar rank

When several classes get the same rank, we can assign to class:

# Dealing with similar rank

When several classes get the same rank, we can assign to class:

- with higher prior probability
- having closest representative
- having closest mean of representatives (among nearest neighbours)
- which is more compact, having nearest most distant representative

# Table of Contents

# Distance metric selection

- Baseline case - Euclidean metric
- Necessary to normalize features.
  - Define $\mu_j$, $\sigma_j$, $L_j$, $U_j$ to be mean value, standard deviation, minimum and maximum value of the $j$-th feature.

| **Name** | **Transformation** | **Properties of resulting feature** |
|---|---|---|
| Autoscaling | $x_j' = \frac{x_j - \mu_j}{\sigma_j}$ | zero mean and unit variance. |
| Range scaling | $x_j' = \frac{x_j - L_j}{U_j - L_j}$ | belongs to $[0, 1]$ interval. |

## Normalization of features

- Non-linear transformations incorporating features with rare large values:
  - $x_i' = \log(x_i)$
  - $x_i' = x^p$, $0 \le p < 1$
- For $F_i(\alpha) = P(x^i \le \alpha)$ transformation $\tilde{x}^i \to F_i(x^i)$ will give feature uniformly distributed on $[0, 1]$[6].

[6] Prove that

# Distance metric selection[7]

| Metric | $d(x,z)$ |
|---|---|
| Euclidean | $\sqrt{\sum_{i=1}^{D}(x^i - z^i)^2}$ |
| $L_p$ | $\sqrt[p]{\sum_{i=1}^{D}(x^i - z^i)^p}$ |
| $L_\infty$ | $\max_{i=1,2,...D} \lvert x^i - z^i \rvert$ |
| $L_1$ | $\sum_{i=1}^{D} \lvert x^i - z^i \rvert$ |
| Canberra | $\frac{1}{D}\sum_{i=1}^{D} \frac{\lvert x^i - z^i \rvert}{\lvert x^i + z^i \rvert}$ |
| Lance-Williams | $\frac{\sum_{i=1}^{D} \lvert x^i - z^i \rvert}{\sum_{i=1}^{D} \lvert x^i + z^i \rvert}$ |

[7] Plot iso-lines for $L_1, L_2, L_\infty$ metrics

# Other frequently used measures

1. Cosine metric[8]

$$s(x,z)=\frac{\langle x,z\rangle}{\|x\|\,\|z\|}=\frac{\sum_{i=1}^{D}x^i z^i}{\sqrt{\sum_{i=1}^{D}(x^i)^2}\sqrt{\sum_{i=1}^{D}(z^i)^2}}$$

2. Jaccard metric[9][10]

$$f(A,B)=\frac{|A\cap B|}{|A\cup B|}$$

[8] Is it a measure of distance or a measure of similarity? Use $\langle x,z\rangle=\|x\|\,\|z\|\,cos(\alpha)$ where $\alpha$ - is the angle between $x$ and $y$.

[9] Is it a measure of distance or a measure of similarity?

[10] Compare qualitively cosine and Jaccard measures for binary encoded sets.

# Whitening transformation

- $x \sim F(\mu, \Sigma)$, $\mu = \mathbb{E}[\mu]$, $\Sigma = cov(x, x)$, $\mu \in \mathbb{R}^D$, $\Sigma \in \mathbb{R}^{DxD}$
- Whitening transformation:

$$z = \Sigma^{-1/2}(x - \mu)$$

- Properties[11]:

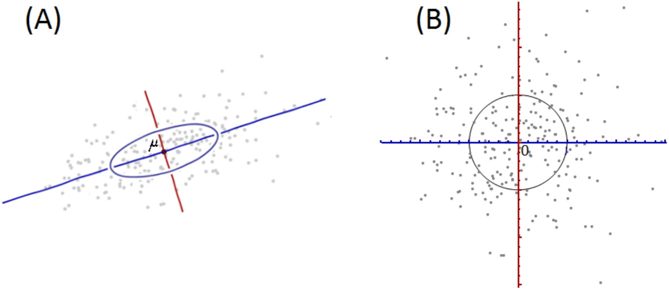
$$Ez = \mathbf{0},\ cov[z, z] = I.$$

[11] Prove them

# Distance between whitened objects (Mahalanobis distance)

(A): object in initial feature space with Mahalonobis sphere $G_\alpha = \{x : \rho_M(x, \mu) = \alpha\}$.

(B): the image of objects and sphere in normalized space $(Im[G_\alpha] = \{z : \rho_E(z, 0) = \alpha\}$.



# Distance between normalized feature vectors

- Distance between normalized $x$ and $x'$ is equal to Euclidean distance between $z = \Sigma^{-1/2}(x - \mu)$ and $z' = \Sigma^{-1/2}(x' - \mu)$:

$$
\begin{aligned}
\rho_M(x, x') &= \rho_E(z, z') = \sqrt{(z - z')^T(z - z')} = \\
&= \sqrt{(x - x')^T \Sigma^{-1/2} \Sigma^{-1/2} (x - x')} \\
&= \sqrt{(x - x')^T \Sigma^{-1} (x - x')}
\end{aligned}
$$

- This is known as *Mahalonobis distance*[12].

12 How will Mahalanobis distance look like when features are uncorrelated? Interpret the result.

# Table of Contents

# Weighted voting

Let training set $x_1, x_2, ...x_N$ be rearranged to $x_{i_1}, x_{i_2}, ...x_{i_N}$ by increasing distance to the test pattern $x$:
$d(x, x_{i_1}) \le d(x, x_{i_2}) \le ... \le d(x, x_{i_N})$.
Define $z_1 = x_{i_1}$, $z_2 = x_{i_2}, ...z_K = x_{i_K}$.
Usual K-NN algorithm can be defined, using $C$ discriminant functions:

$$g_c(x) = \sum_{k=1}^{K} \mathbb{I}[z_k \in \omega_c], \quad c = 1, 2, ...C.$$

# Weighted voting

Let training set $x_1, x_2, ...x_N$ be rearranged to $x_{i_1}, x_{i_2}, ...x_{i_N}$ by increasing distance to the test pattern $x$:
$d(x, x_{i_1}) \le d(x, x_{i_2}) \le ... \le d(x, x_{i_N})$.
Define $z_1 = x_{i_1}$, $z_2 = x_{i_2}, ...z_K = x_{i_K}$.
Usual K-NN algorithm can be defined, using $C$ discriminant functions:

$$g_c(x) = \sum_{k=1}^{K} \mathbb{I}[z_k \in \omega_c], \quad c = 1, 2, ...C.$$

Weighted K-NN algorithm uses weighted voting scheme:

$$g_c(x) = \sum_{k=1}^{K} w(k,\ d(x, z_k))\mathbb{I}[z_k \in \omega_c], \quad c = 1, 2, ...C.$$

# Commonly chosen weights

Index dependent weights:

$$w_k = \alpha^k, \quad \alpha \in (0,1)$$

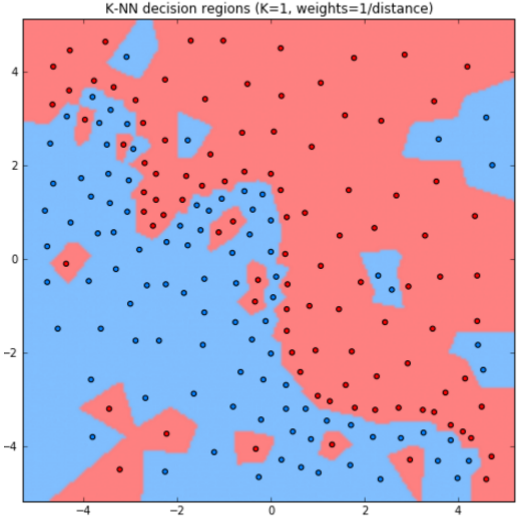
$$w_k = \frac{K+1-k}{K}$$

Distance dependent weights:

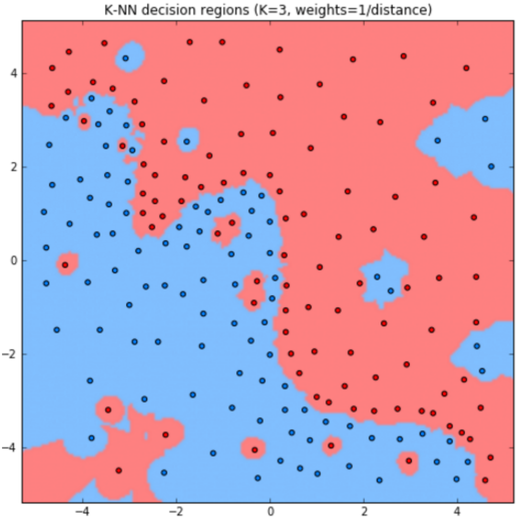
$$w_k = \begin{cases} \frac{d(z_K,x)-d(z_k,x)}{d(z_K,x)-d(z_1,x)}, & d(z_K,x) \neq d(z_1,x) \\ 1 & d(z_K,x) = d(z_1,x) \end{cases}$$

$$w_k = \frac{1}{d(z_k,x)}$$

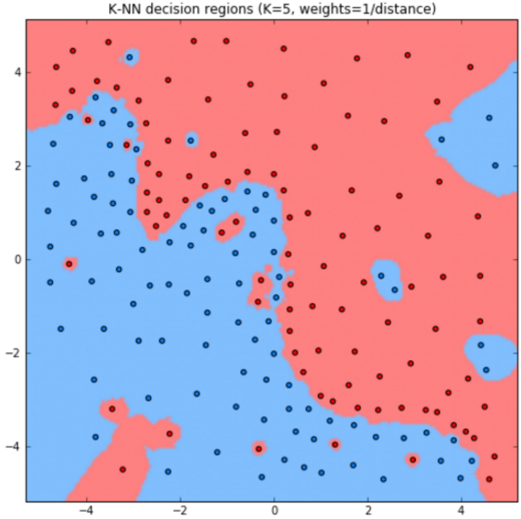
# Example: K-NN classification with weights



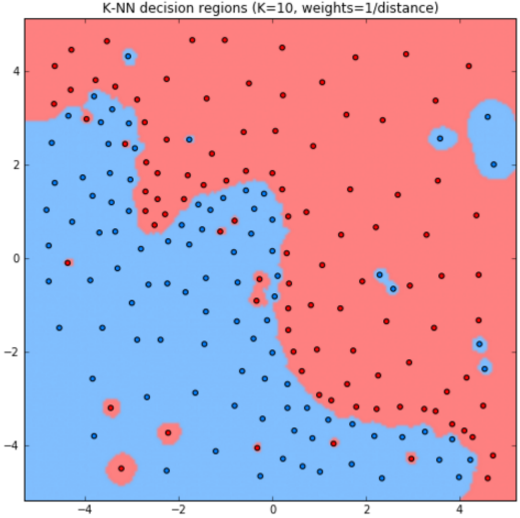
# Example: K-NN classification with weights



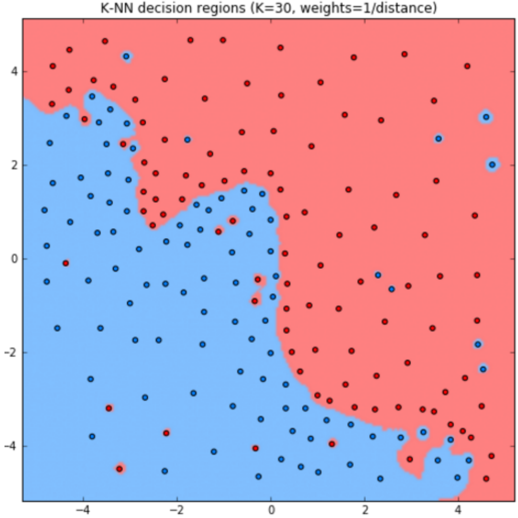
# Example: K-NN classification with weights



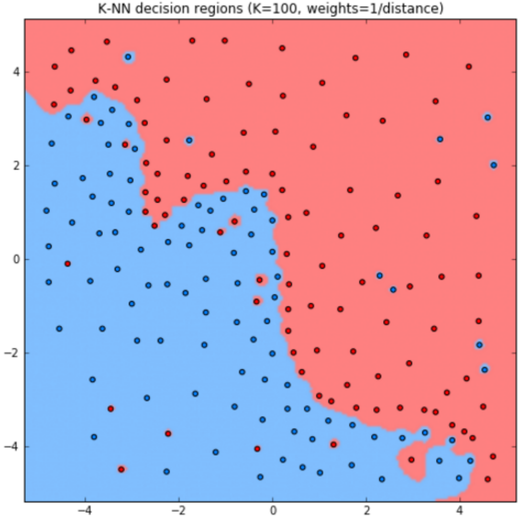
# Example: K-NN classification with weights



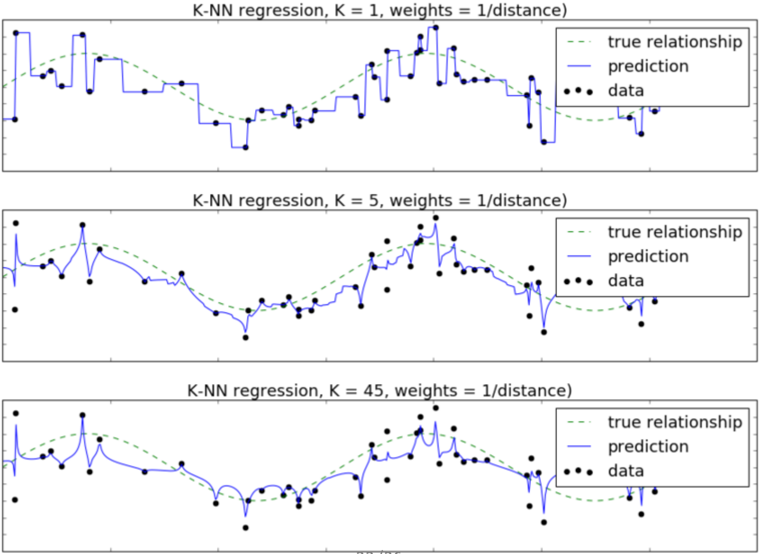
# Example: K-NN classification with weights



# Example: K-NN classification with weights



# Example: K-NN regression with weights



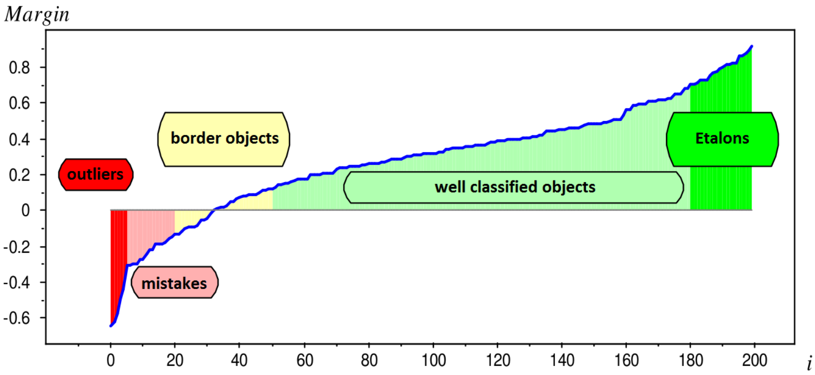
# Margin definition

- Consider the training set: $(x_1, c_1), (x_2, c_2), ...(x_N, c_N)$, where $c_i$ is the correct class for object $x_i$, and $\mathbf{C} = \{1, 2, ...C\}$ - is the set of all classes.
- Define the margin:

$$M(x_i, c_i) = g_{c_i}(x_i) - \max_{c \in \mathbf{C} \setminus \{\mathbf{c_i}\}} g_c(x_i)$$

  - margin is negative <=> object $x_i$ was incorrectly classified
  - the value of margin shows how much the classifier is inclined to vote for class $c_i$

# Categorization of objects based on margin




Good classifier should:

- minimize the number of negative margin region
- classify correctly with high margin

# Alternative to K-NN: Parzen window method[13]

Parzen window method:

$$\widehat{f}(x) = \arg\max_{y \in Y} \sum_{n=1}^{N} \mathbb{I}[y_n = y] K\left(\frac{\rho(x, x_n)}{h(x)}\right)$$

- Selection of $h(x)$:
    - $h(x) = const$
    - $h(x) = \rho(x, z_K)$, where $z_K$ - $K$-th nearest neighbour.
        - better for unequal distribution of objects

[13] Under what selection of K(u) and h(x) will Parzen window reduce to simple K-NN?