Metric methods of machine learning

Victor Kitov

v.v.kitov@yandex.ru

Yandex School of Data Analysis

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.

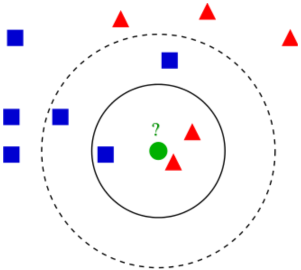
- Regression case: targets of nearest neighbours are averaged
- $k = 1$: nearest neighbour algorithm[1]
- Base assumption of the method[2]:
  - similar objects yield similar outputs

[1] what will happen for $K = N$? Majority vote

[2] what is simpler - to train K-NN model or to apply it? there is no training (only parameter fine tuning)

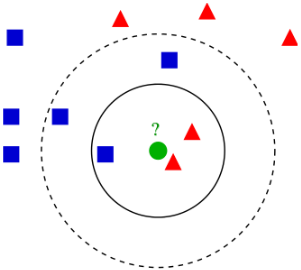
K-NN illustration

Classification:

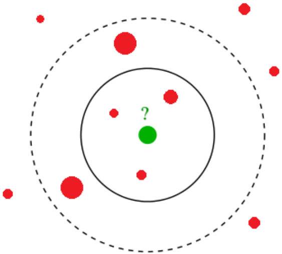


K-NN illustration

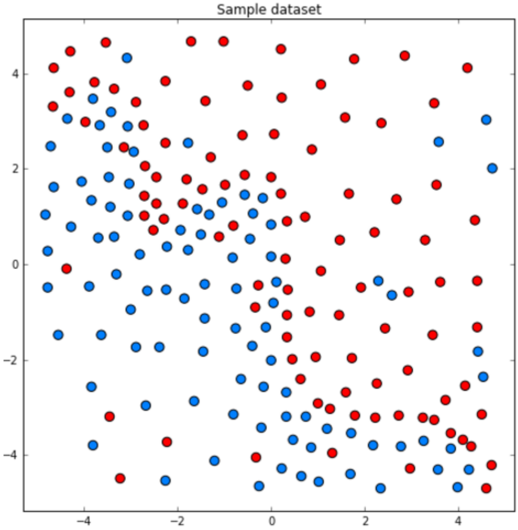
Classification:



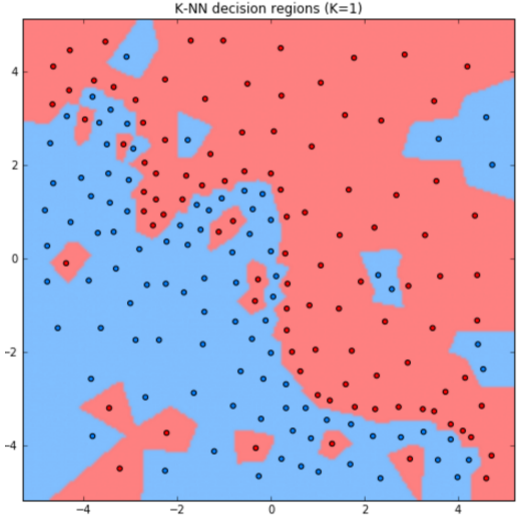
Regression:



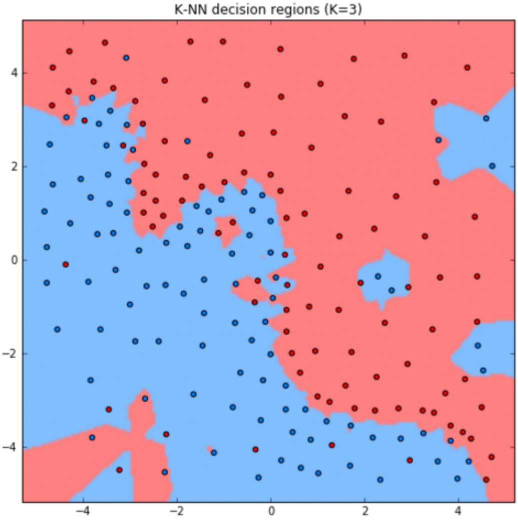
Sample dataset



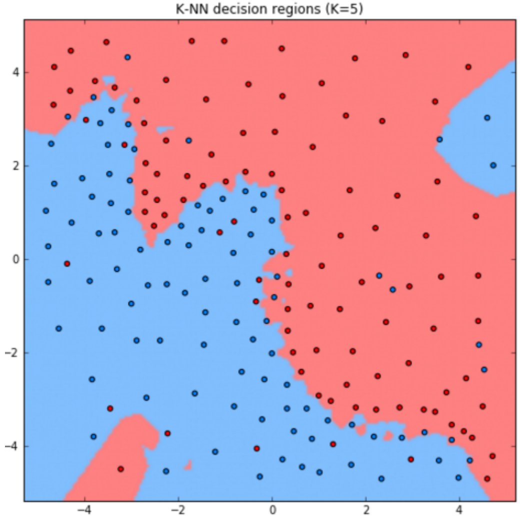
Example: K-NN classification



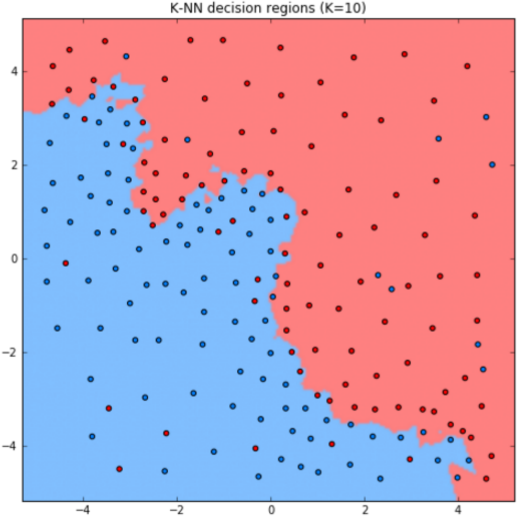
Example: K-NN classification



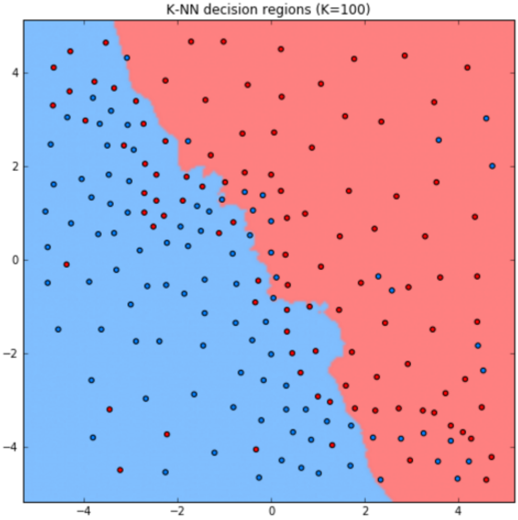
Example: K-NN classification



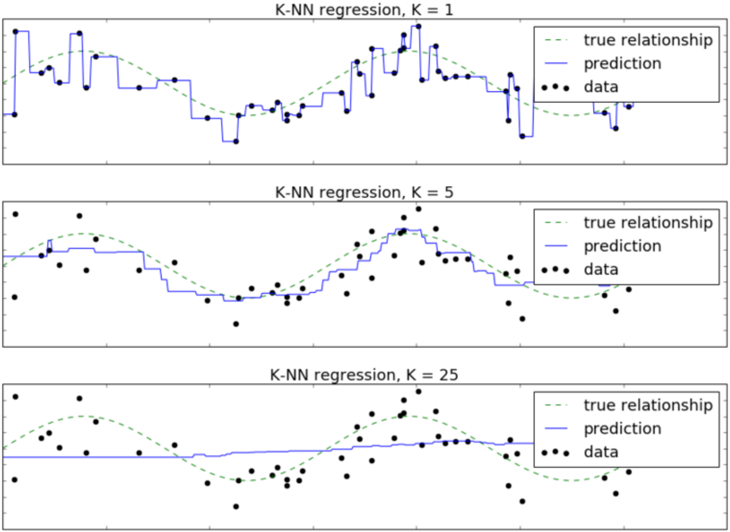
Example: K-NN classification



Example: K-NN classification



Example: K-NN regression



Parameters of the method

- Parameters:
  - the number of nearest neighbours K
  - distance metric $\rho(x, y)$

how do we define distance (Minkowsky), weighted distance

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

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 μ_j, σ_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 $[0,1]$[3].

[3] Prove that

Distance metric selection[4]

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_∞	$\max_{i=1,2,...D} \lvert x^i - z^i \rvert$
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	$\frac{\sum_{i=1}^{D} \lvert x^i - z^i \rvert}{\sum_{i=1}^{D} \lvert x^i + z^i \rvert}$

[4] Plot iso-lines for L_1, L_2, L_∞ metrics.

Other frequently used measures

1. Cosine metric[5]

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

2. Jaccard metric[6][7]

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

[5] Is it a measure of distance or a measure of similarity? Use $\langle x,z\rangle=\|x\|\,\|z\|\,cos(\alpha)$ where α - is the angle between x and y.
[6] Is it a measure of distance or a measure of similarity?
[7] Compare qualitatively cosine and Jaccard measures for binary encoded sets.

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:
$d(x, x_{i_1}) \leq d(x, x_{i_2}) \leq ... \leq d(x, x_{i_N})$.
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:

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

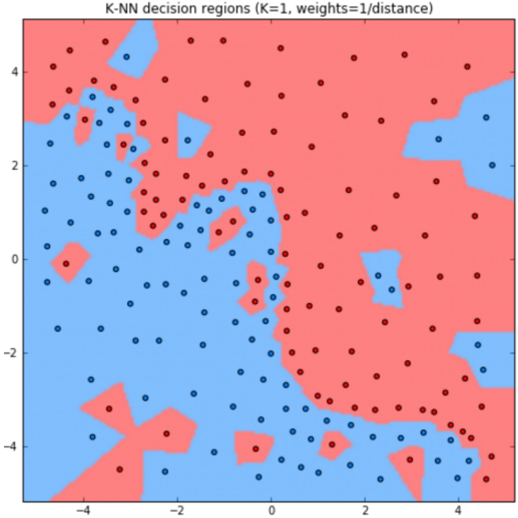
$$w_k = \frac{K+1-k}{K}$$

Distance dependent weights:

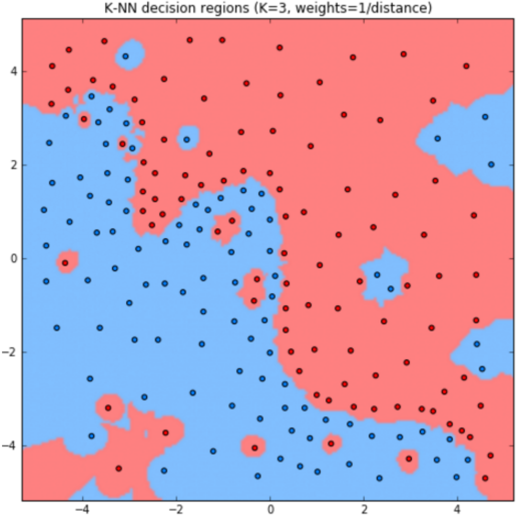
$$w_k = \begin{cases} \frac{d(z_K,x)-d(z_k,x)}{d(z_K,x)-d(z_1,x)}, & d(z_K,x) \neq d(z_1,x) \\ 1 & d(z_K,x) = d(z_1,x) \end{cases}$$

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

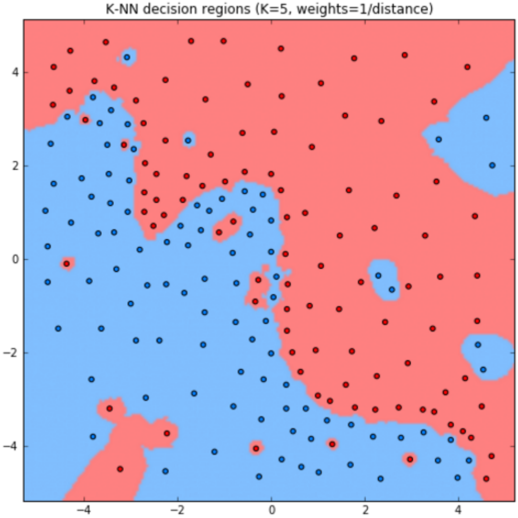
Example: K-NN classification with weights



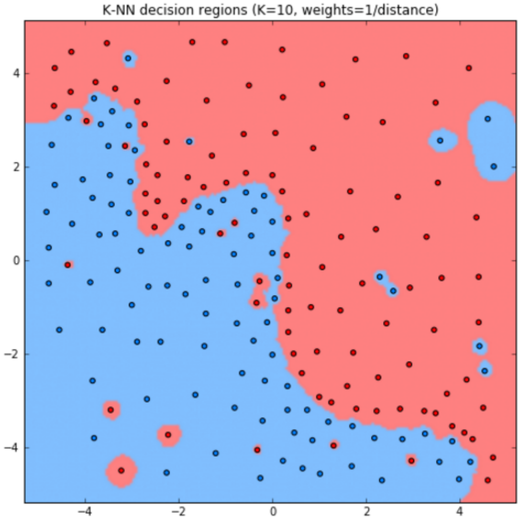
Example: K-NN classification with weights



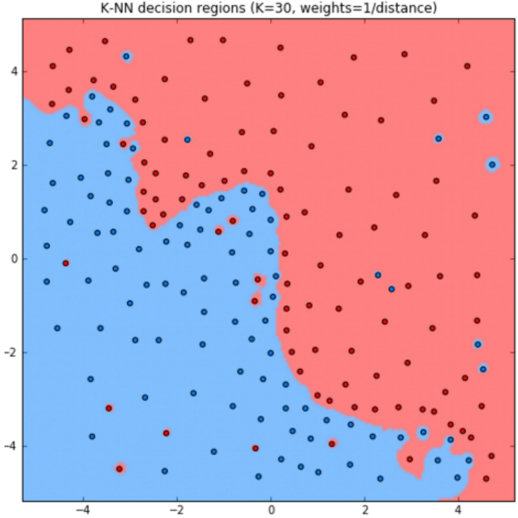
Example: K-NN classification with weights



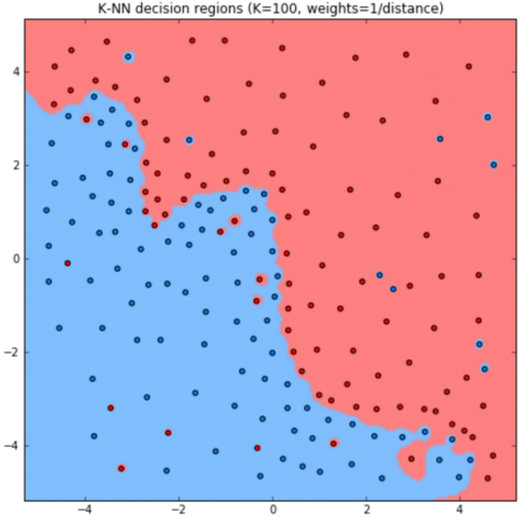
Example: K-NN classification with weights



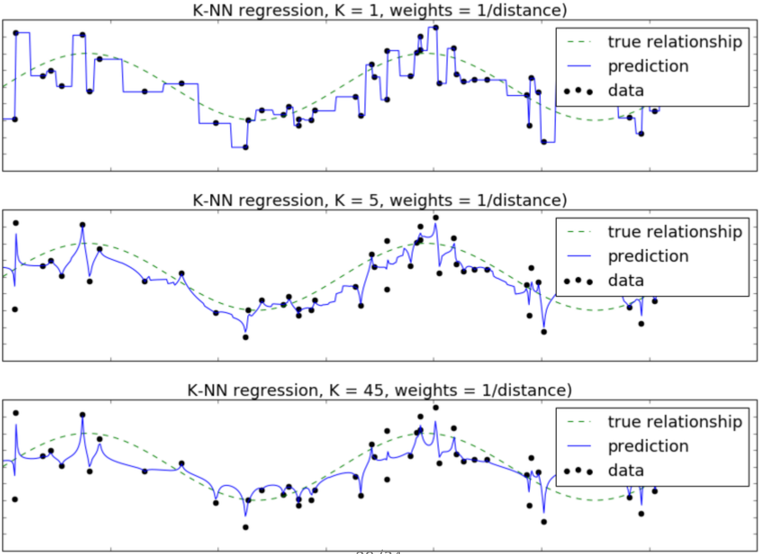
Example: K-NN classification with weights



Example: K-NN classification with weights



Example: K-NN regression with weights



Alternative to K-NN classification: Parzen window method[8]

. Parzen windows use neighbourhoods of constant size (which can contain more or less than k training examples). k-NN expands or shrinks the neighbourhood to always contain exactly k training examples

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

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

Table of Contents

Nadaraya-Watson regression

- Names: local constant regression, kernel regression
- For each x assume $f(x) = const = \alpha$, $\alpha \in \mathbb{R}$.

$$Q(\alpha, X_{training}) = \sum_{i=1}^{N} w_i(x)(\alpha - y_i)^2 \to \min_{\alpha \in \mathbb{R}}$$

- Weights depend on the proximity of training objects to the predicted object:

$$w_i(x) = K\left(\frac{\rho(x, x_i)}{h}\right)$$

- From stationarity condition $\frac{\partial Q}{\partial \alpha} = 0$ obtain optimal $\widehat{\alpha}(x)$:

$$f(x, \alpha) = \widehat{\alpha}(x) = \frac{\sum_{i=1}^{N} y_i w_i(x)}{\sum_{i=1}^{N} w_i(x)} = \frac{\sum_{i=1}^{N} y_i K\left(\frac{\rho(x,x_i)}{h}\right)}{\sum_{i=1}^{N} K\left(\frac{\rho(x,x_i)}{h}\right)}$$

Comments

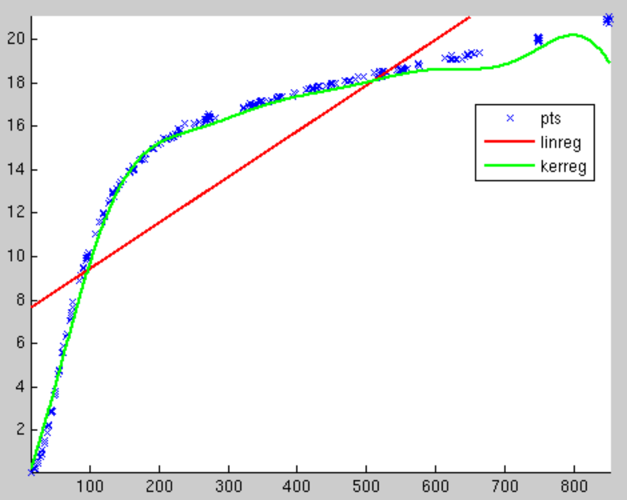
Under certain regularity conditions $g(x,\alpha) \xrightarrow{P} E[y|x]$
Typically used kernel functions[9]:

$$\begin{aligned} K_G(r) &= e^{-\frac{1}{2}r^2} - \text{Gaussian kernel} \\ K_P(r) &= (1-r^2)^2\mathbb{I}[|r|<1] - \text{quartic kernel} \end{aligned}$$

- The specific form of the kernel function does not affect the accuracy much
- h controls the adaptability of the model to local changes in data
    - *how h affects under/overfitting?*
    - h can be constant or depend on x (if concentration of objects changes significantly)

[9] Compare them in terms of required computation.

Example


20
18
16
14
12
10
8
6
4
2
100
200
300
400
500
600
700
800
pts
linreg
kerreg