

K-nearest neighbours

Victor Kitov
v.v.kitov@yandex.ru

Table of Contents

- 1 Basic variant of K-NN
- 2 Distance metric selection
- 3 Weighted voting
- 4 Nadaraya-Watson regression

K-nearest neighbours

Classification using k nearest neighbours

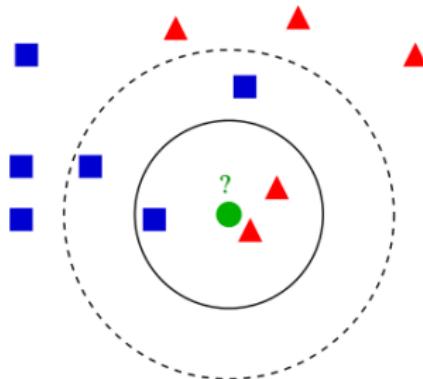
- ➊ Find k closest objects to the predicted object x in the training set.
 - ➋ Associate x the most frequent class among its k neighbours.
-
- Regression case: targets of nearest neighbours are averaged
 - $k = 1$: nearest neighbour algorithm¹
 - Base assumption of the method²:
 - similar objects yield similar outputs

¹what will happen for $K = N$?

²what is simpler - to train K-NN model or to apply it?

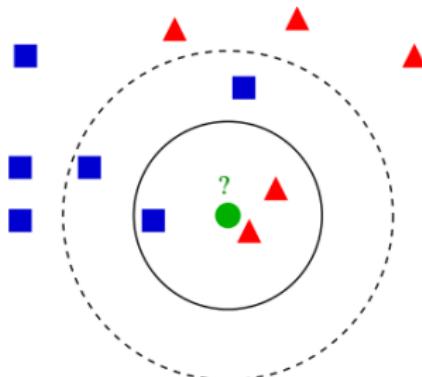
K-NN illustration

Classification:

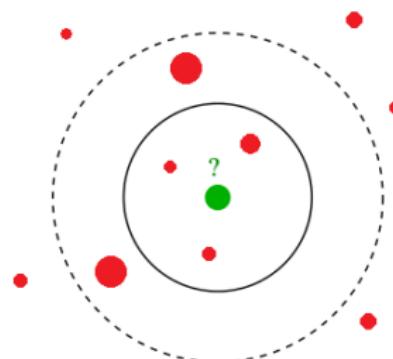


K-NN illustration

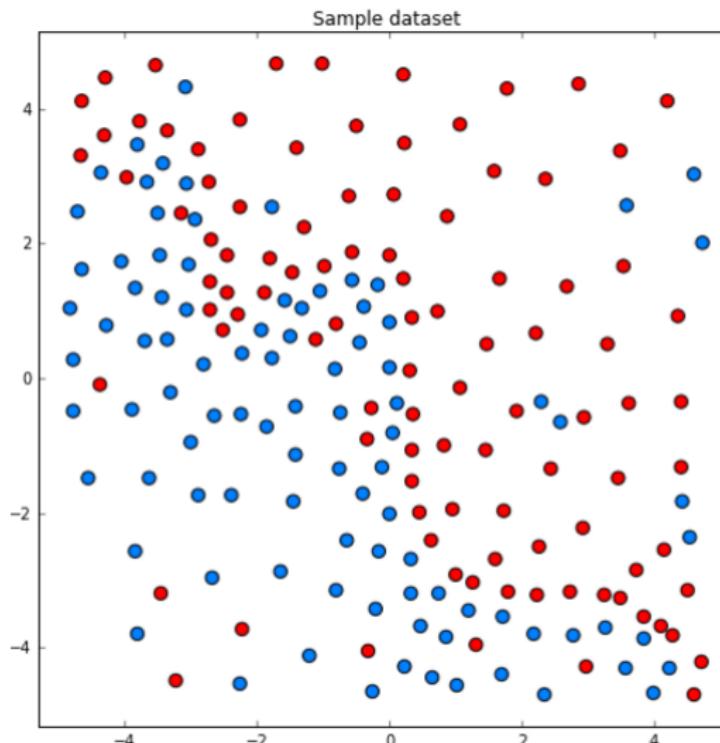
Classification:



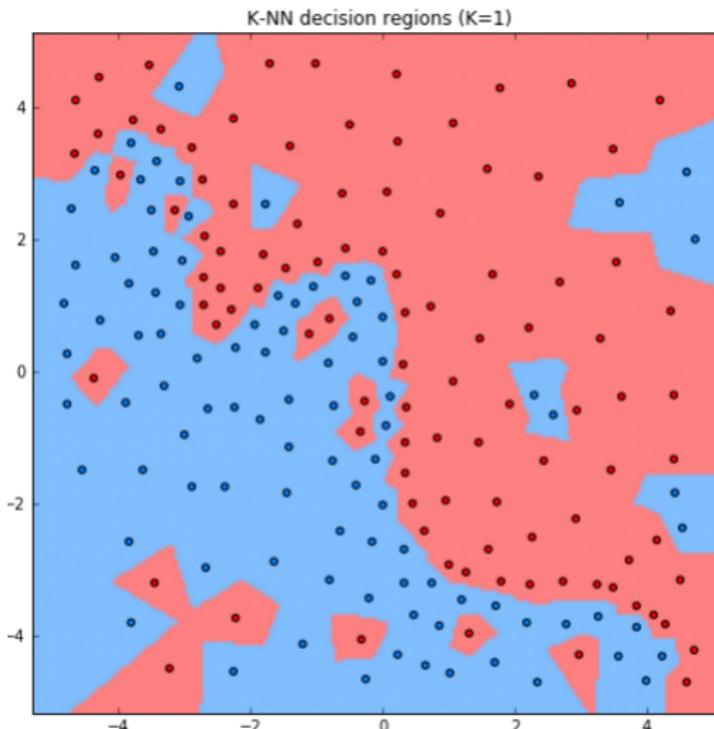
Regression:



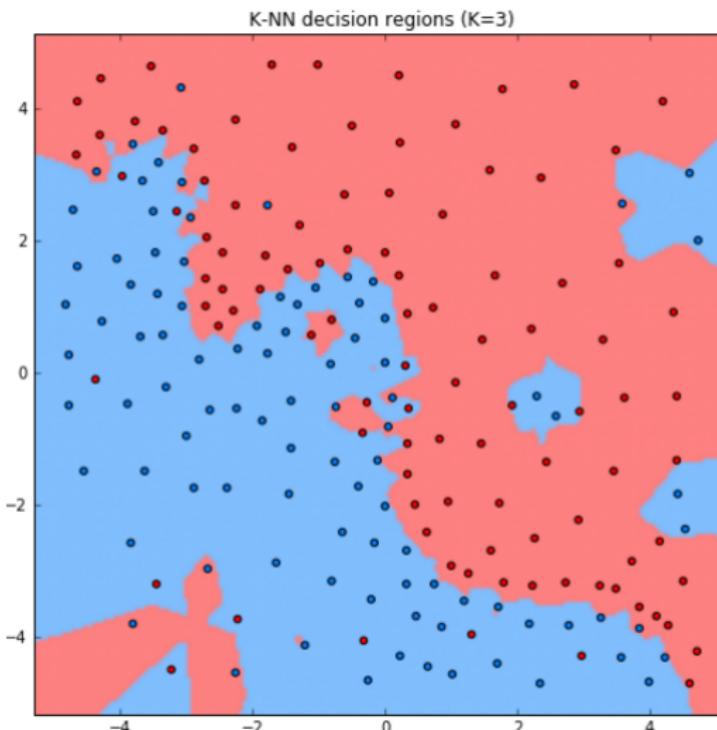
Sample dataset



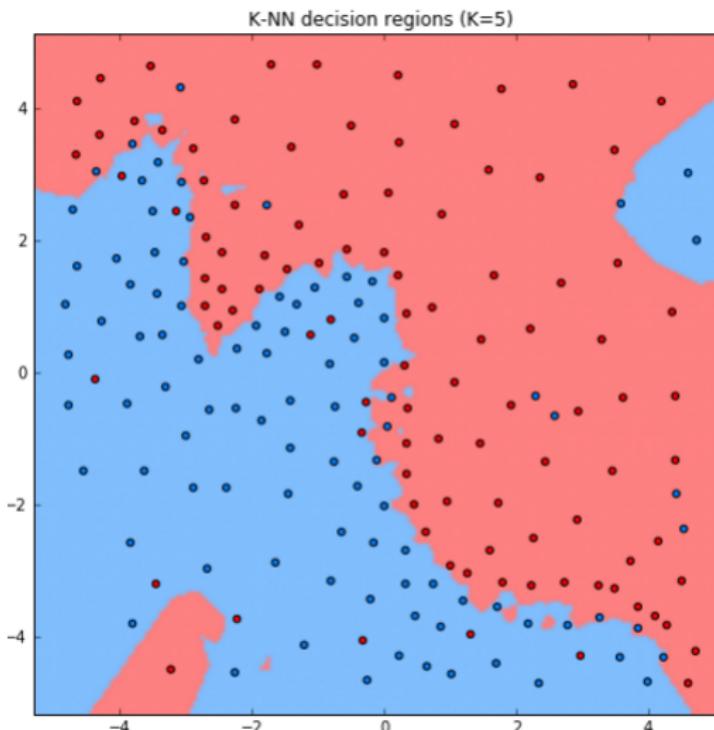
Example: K-NN classification



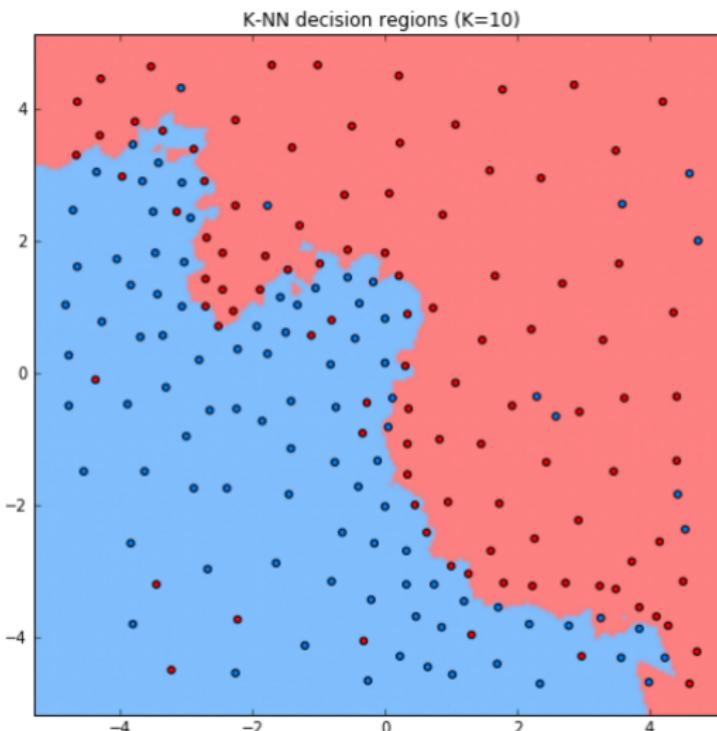
Example: K-NN classification



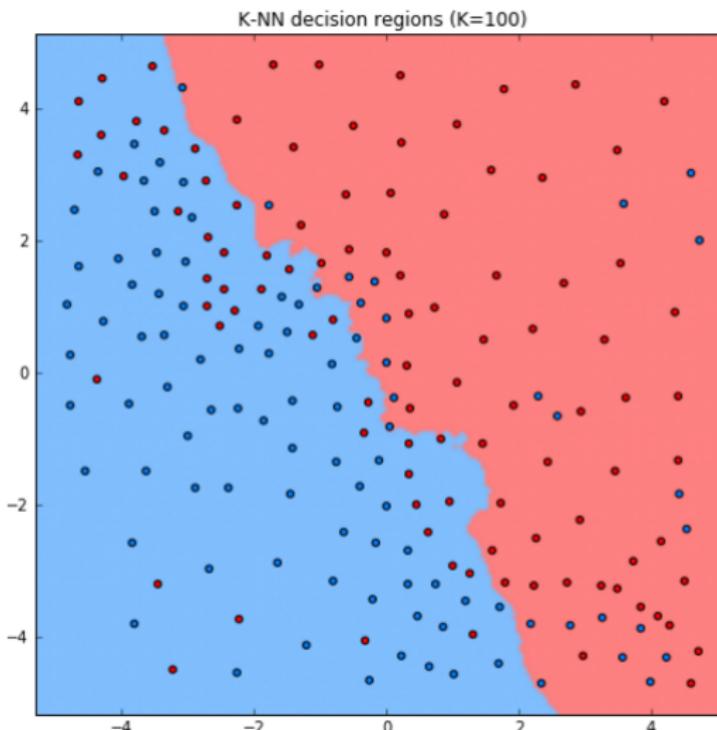
Example: K-NN classification



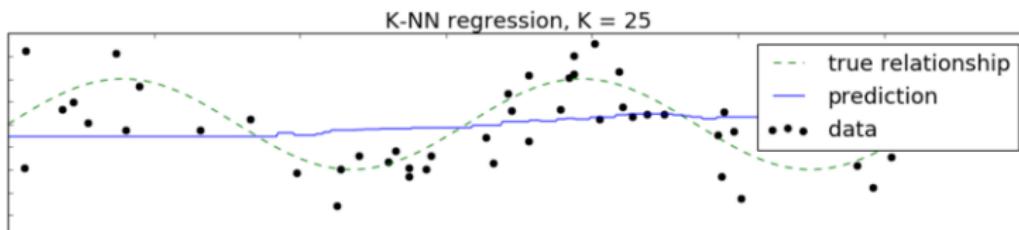
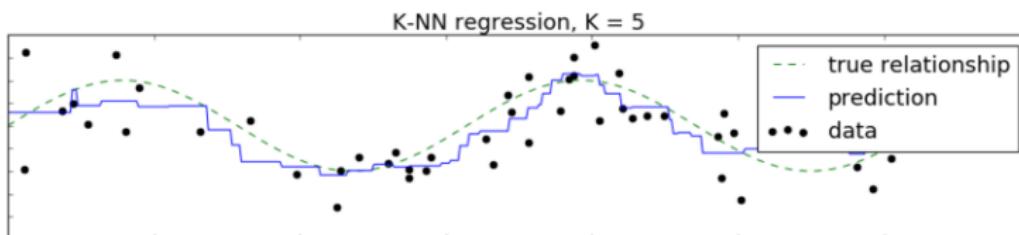
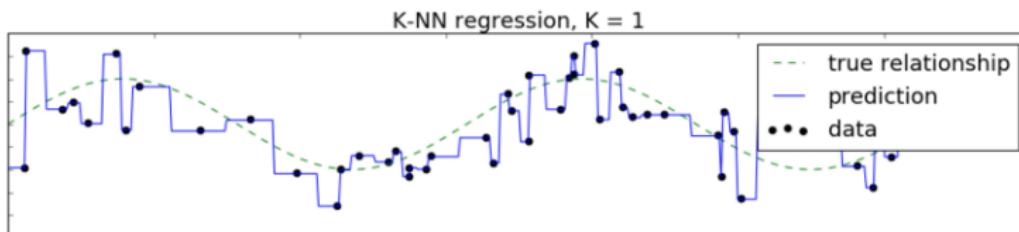
Example: K-NN classification



Example: K-NN classification



Example: K-NN regression



Parameters of the method

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

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 neighbours:
 - 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 :

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

- 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

- 1 Basic variant of K-NN
- 2 Distance metric selection
- 3 Weighted voting
- 4 Nadaraya-Watson regression

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:
 - $x'_i = \log(x_i)$
 - $x'_i = x^p, 0 \leq p < 1$
- For $F_i(\alpha) = P(x^i \leq \alpha)$ transformation $\tilde{x}^i \rightarrow F_i(x^i)$ will give feature uniformly distributed on $[0, 1]^3$.

³Prove that

Distance metric selection⁴

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,\dots,D} x^i - z^i $
L_1	$\sum_{i=1}^D x^i - z^i $
Canberra	$\frac{1}{D} \sum_{i=1}^D \frac{ x^i - z^i }{ x^i + z^i }$
Lance-Williams	$\frac{\sum_{i=1}^D x^i - z^i }{\sum_{i=1}^D x^i + z^i }$

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

Other frequently used measures

① Cosine metric⁵

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

② Jaccard metric⁶⁷

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

⁵Is 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 .

⁶Is it a measure of distance or a measure of similarity?

⁷Compare qualitatively cosine and Jaccard measures for binary encoded sets.

Table of Contents

- 1 Basic variant of K-NN
- 2 Distance metric selection
- 3 Weighted voting
- 4 Nadaraya-Watson regression

Weighted voting

Let training set x_1, x_2, \dots, x_N be rearranged to $x_{i_1}, x_{i_2}, \dots, x_{i_N}$ by increasing distance to the test pattern x :

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

Define $z_1 = x_{i_1}$, $z_2 = x_{i_2}$, $\dots, z_K = x_{i_K}$.

Usual K-NN algorithm can be defined, using C discriminant functions:

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

Weighted voting

Let training set x_1, x_2, \dots, x_N be rearranged to $x_{i_1}, x_{i_2}, \dots, x_{i_N}$ by increasing distance to the test pattern x :

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

Define $z_1 = x_{i_1}$, $z_2 = x_{i_2}$, $\dots, z_K = x_{i_K}$.

Usual K-NN algorithm can be defined, using C discriminant functions:

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

Weighted K-NN algorithm uses weighted voting scheme:

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

Commonly chosen weights

Index dependent weights:

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

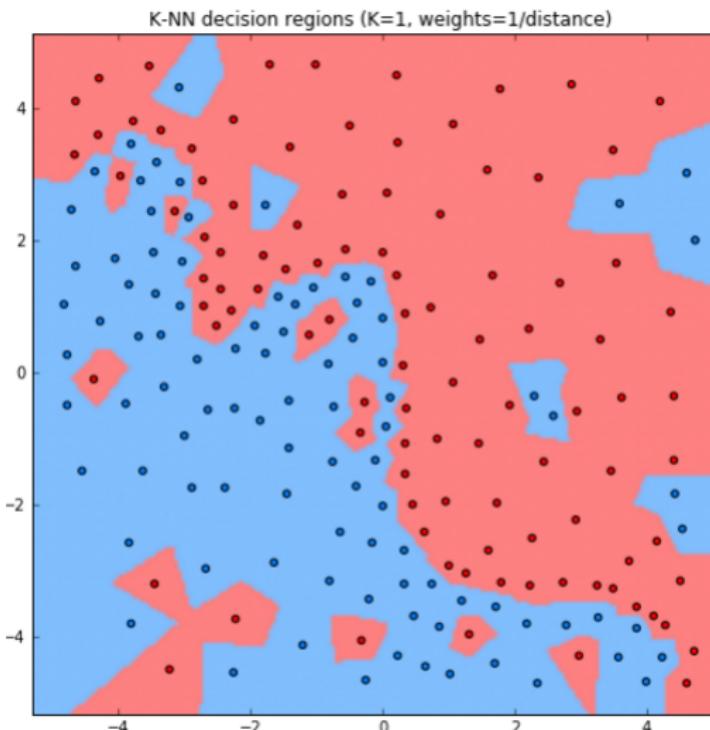
$$w_k = \frac{K+1-k}{K}$$

Distance dependent weights:

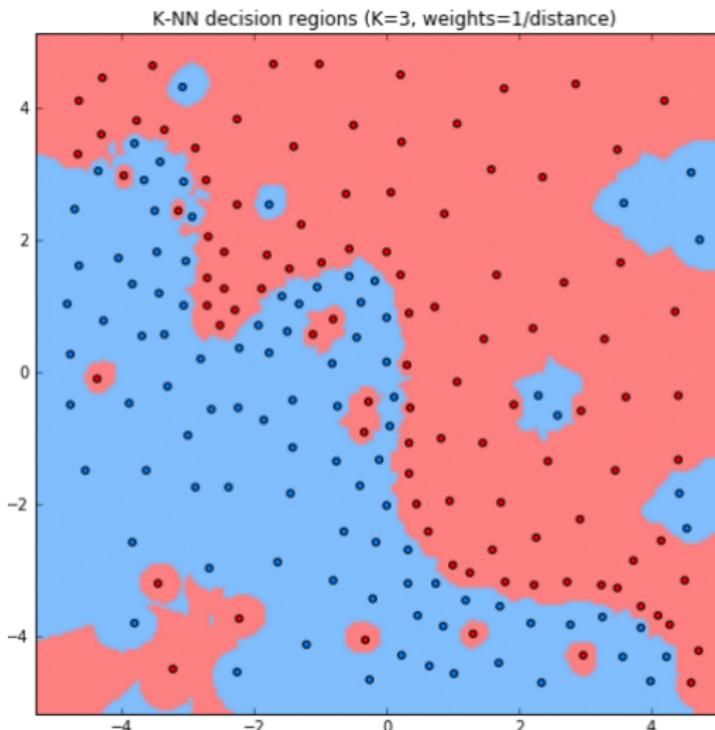
$$w_k = \begin{cases} \frac{d(z_K, x) - d(z_k, x)}{d(z_K, x) - d(z_1, x)}, & d(z_K, x) \neq d(z_1, x) \\ 1 & d(z_K, x) = d(z_1, x) \end{cases}$$

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

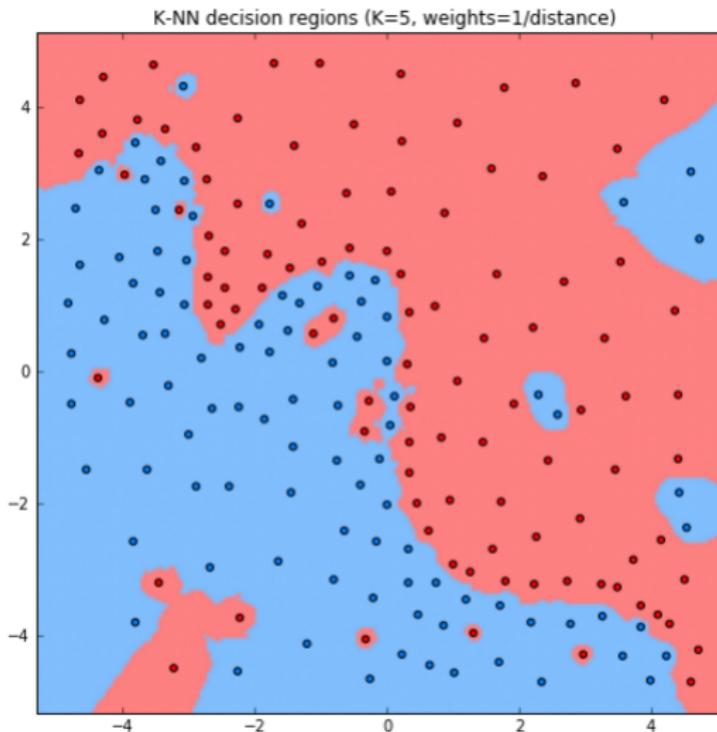
Example: K-NN classification with weights



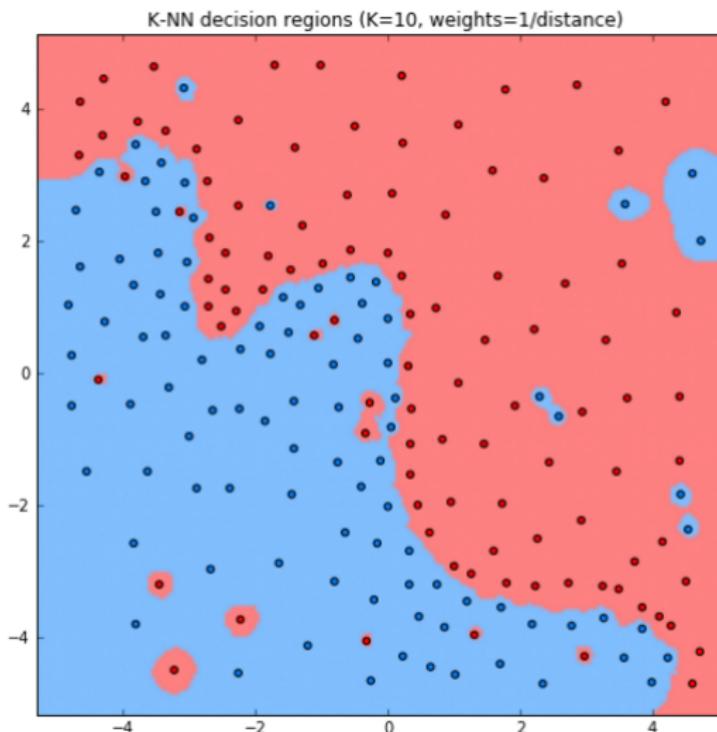
Example: K-NN classification with weights



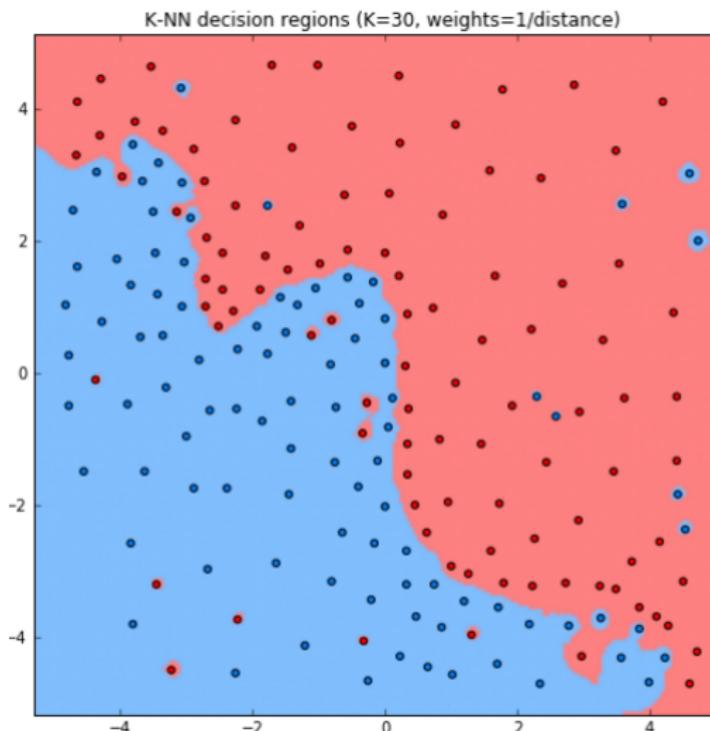
Example: K-NN classification with weights



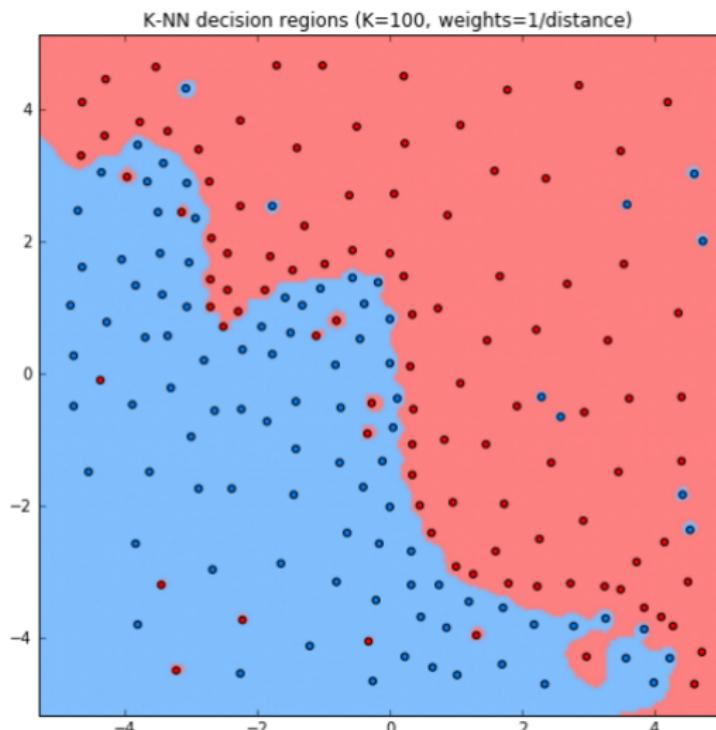
Example: K-NN classification with weights



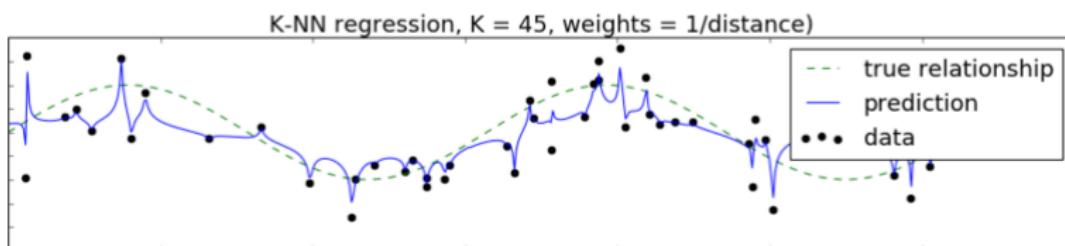
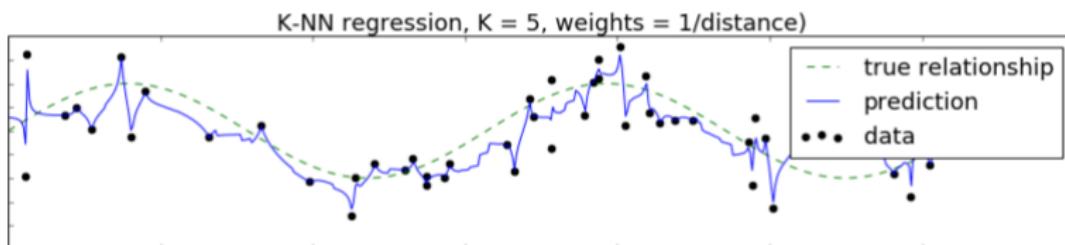
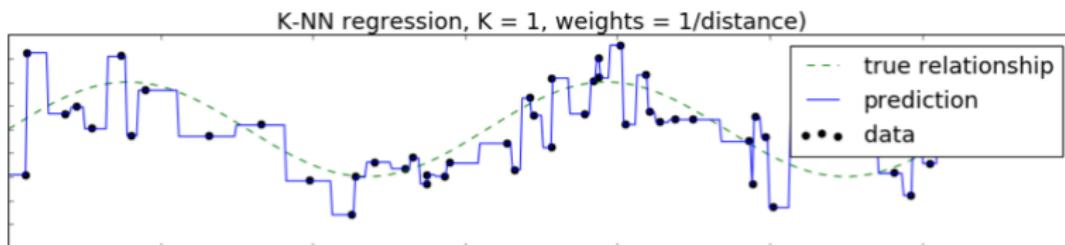
Example: K-NN classification with weights



Example: K-NN classification with weights



Example: K-NN regression with weights



Alternative to K-NN classification: Parzen window method⁸

Parzen window method:

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

- Selection of $h(x)$:

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

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

Table of Contents

- 1 Basic variant of K-NN
- 2 Distance metric selection
- 3 Weighted voting
- 4 Nadaraya-Watson regression

Nadaraya-Watson regression

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

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

- 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 $\hat{\alpha}(x)$:

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

Comments

Under certain regularity conditions $g(x, \alpha) \xrightarrow{P} E[y|x]$

Typically used kernel functions⁹:

$$K_G(r) = e^{-\frac{1}{2}r^2} \text{ -- Gaussian kernel}$$

$$K_P(r) = (1 - r^2)^2 \mathbb{I}[|r| < 1] \text{ -- quartic kernel}$$

- 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)

⁹Compare them in terms of required computation.

Example

