

Neighbors

Compute nearest neighbors in data according to reference.

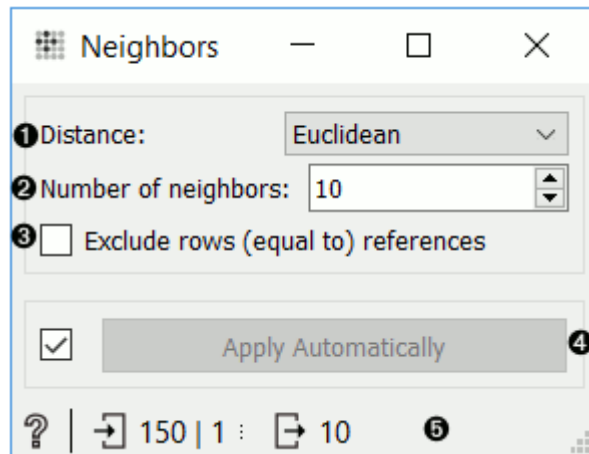
Inputs

- Data: An input data set.
- Reference: A reference data for neighbor computation.

Outputs

- Neighbors: A data table of nearest neighbors according to reference.

The **Neighbors** widget computes nearest neighbors for a given reference and for a given distance measure. The reference can be either one instance or more instances. In the case with one reference widget outputs closest **n** instances from data where **n** is set by the **Number of neighbors** option in the widget. When reference contains more instances widget computes the combined distance for each data instance as a minimum of distances to each reference. Widget outputs **n** data instances with lowest combined distance.



1. Distance measure for computing neighbors. Supported measures are: Euclidean, Manhattan, Mahalanobis, Cosine, Jaccard, Spearman, absolute Spearman, Pearson, absolute Pearson.
2. Number of neighbors on the output.
3. If *Exclude rows (equal to) references* is ticked, data instances that are highly similar to the reference (distance < 1e-5), will be excluded.
4. Click *Apply* to commit the changes. To communicate changes automatically tick *Apply Automatically*.
5. Status bar with access to widget help and information on the input and output data.

Examples

In the first example, we used *iris* data and passed it to **Neighbors** and to **Data Table**. In **Data Table**, we selected an instance of iris, that will serve as our reference, meaning we wish to retrieve 10 closest examples to the select data instance. We connect **Data Table** to **Neighbors** as well.

We can observe the results of neighbor computation in **Data Table (1)**, where we can see 10 closest images to our selected iris flower.

The screenshot displays the Orange Data Mining software interface. The main workflow consists of a **File** widget connected to a **Data** widget, which then connects to a **Neighbors** widget. The **Neighbors** widget is also connected to a **Data Table** widget. The **Data Table** widget is configured to show 10 instances, 4 features, and a discrete class with 3 values. The **Neighbors** widget is configured with a Euclidean distance metric, 10 neighbors, and the option to exclude rows equal to references. The **Data Table (1)** widget shows the results of the neighbor computation, listing 10 instances of *Iris-setosa* with their distances and feature values.

Neighbors Widget Configuration:

- Distance: Euclidean
- Number of neighbors: 10
- ☒ Exclude rows (equal to) references
- ☒ Apply Automatically

Data Table (1) Widget Information:

- Info: 10 instances (no missing values), 4 features (no missing values), Discrete class with 3 values (no missing values), 1 meta attribute (no missing values)
- Variables: ☒ Show variable labels (if present), ☐ Visualize numeric values, ☒ Color by instance classes
- Selection: ☒ Select full rows

Data Table (1) Widget Output:

	iris	distance	sepal length	sepal width
1	Iris-setosa	0.316228	4.7	3.2
2	Iris-setosa	0.264575	4.7	3.2
3	Iris-setosa	0.331662	4.6	3.1
4	Iris-setosa	0.3	4.8	3.4
5	Iris-setosa	0.223607	4.6	3.2
6	Iris-setosa	0.316228	4.4	3.2
7	Iris-setosa	0.424264	5.0	3.3
8	Iris-setosa	0.424264	5.0	3.5
9	Iris-setosa	0.424264	5.0	3.4
10	Iris-setosa	0.424264	4.8	3.1

Data Table Widget Output:

	iris	sepal length	sepal width	petal length
1	Iris-setosa	5.1	3.5	1.4
2	Iris-setosa	4.9	3.0	1.4
3	Iris-setosa	4.7	3.2	1.3
4	Iris-setosa	4.6	3.1	1.3
5	Iris-setosa	5.0	3.6	1.4
6	Iris-setosa	5.4	3.9	1.7
7	Iris-setosa	4.6	3.4	1.4
8	Iris-setosa	5.0	3.4	1.3
9	Iris-setosa	4.4	2.9	1.4
10	Iris-setosa	4.9	3.1	1.3
11	Iris-setosa	5.4	3.7	1.5
12	Iris-setosa	4.8	3.4	1.6
13	Iris-setosa	4.8	3.0	1.4

Now change the selection **Data Table** to multiple examples. As a result, we get instances with closest combined distances to the references. The method computes the combined distance as a minimum of distances to each reference.

The screenshot displays the Orange Data Mining software interface. The main workflow area shows a sequence of widgets: **File** (loading data), **Data Table** (displaying data), **Neighbors** (performing the classification), and **Data Table (1)** (displaying the results). The **Neighbors** widget is currently selected, and its settings panel is open.

Neighbors Widget Settings:

- Distance:** Euclidean
- Number of neighbors:** 10
- ☒ **Exclude rows (equal to) references**
- ☐ **Apply**
- Variables:**
 - ☒ Show variable labels (if present)
 - ☐ Visualize numeric values
 - ☒ Color by instance classes
- Selection:**
 - ☒ Select full rows
- Buttons:** Restore Original Order, Send Automatically

Data Table (1) Info:

- 150 instances (no missing values)
- 4 features (no missing values)
- Discrete class with 3 values (no missing values)
- No meta attributes

Data Table (1) Variables:

- ☒ Show variable labels (if present)
- ☐ Visualize numeric values
- ☒ Color by instance classes

Data Table (1) Data:

	iris	distance	sepal length	sepal width
1	Iris-setosa	0.141421	5.2	3.4
2	Iris-setosa	0.1	5.1	3.5
3	Iris-setosa	0.141421	4.8	3.0
4	Iris-setosa	0.141421	5.1	3.4
5	Iris-setosa	0.141421	5.2	3.5
6	Iris-setosa	0.141421	4.8	3.0
7	Iris-setosa	0.173205	5.0	3.5
8	Iris-setosa	0.173205	5.0	3.4
9	Iris-setosa	0.141421	4.6	3.2
10	Iris-setosa	0.173205	4.9	3.1

Data Table Info:

- 150 instances (no missing values)
- 4 features (no missing values)
- Discrete class with 3 values (no missing values)
- No meta attributes

Data Table Variables:

- ☒ Show variable labels (if present)
- ☐ Visualize numeric values
- ☒ Color by instance classes

Data Table Data:

	iris	sepal length	sepal width	petal length
1	Iris-setosa	5.1	3.5	1.4
2	Iris-setosa	4.9	3.0	1.4
3	Iris-setosa	4.7	3.2	1.3
4	Iris-setosa	4.6	3.1	1.5
5	Iris-setosa	5.0	3.6	1.4
6	Iris-setosa	5.4	3.9	1.7
7	Iris-setosa	4.6	3.4	1.4
8	Iris-setosa	5.0	3.4	1.5
9	Iris-setosa	4.4	2.9	1.4
10	Iris-setosa	4.9	3.1	1.5
11	Iris-setosa	5.4	3.7	1.5
12	Iris-setosa	4.8	3.4	1.6
13	Iris-setosa	4.8	3.0	1.4

Another example requires the installation of Image Analytics add-on. We loaded 15 paintings from famous painters with **Import Images** widget and passed them to **Image Embedding**, where we selected *Painters* embedder.

Then the procedure is the same as above. We passed embedded images to **Image Viewer** and selected a painting from Monet to serve as our reference image. We passed the image to **Neighbors**, where we set the distance measure to *cosine*, ticked off *Exclude reference* and set the neighbors to 2. This allows us to find the actual closest neighbor to a reference painting and observe them side by side in **Image Viewer (1)**.

The screenshot displays the Orange Data Mining software interface. The main workflow area shows a sequence of widgets: **Import Images** (Data → Images), **Image Embedding** (Data → Embeddings), **Neighbors** (Embeddings → Data), and **Image Viewer (1)** (Data → Images). A feedback loop connects the **Neighbors** widget back to the **Image Embedding** widget via the **Selected Images Reference** output.

Two windows are open to show the widget's configuration and output:

- Neighbors Widget Settings:**
 - Distance: Cosine
 - Number of neighbors: 2
 - ☐ Exclude rows (equal to) references
 - ☒ Apply Automatically
 - Info: 15 of 15 images displayed.
 - Image Filename Attribute: image
 - Title Attribute: image name
 - Image Size: (slider)
 - ☒ Send Automatically
- Image Viewer (1) Settings:**
 - Info: 2 of 2 images displayed.
 - Image Filename Attribute: image
 - Title Attribute: image name
 - Image Size: (slider)
 - ☒ Send Automatically

The **Image Viewer (1)** window displays two images side-by-side, identified as **Monet** and **Claude Monet**. The **Neighbors** widget's output area shows a grid of 15 images, with **Monet** highlighted as the selected image.