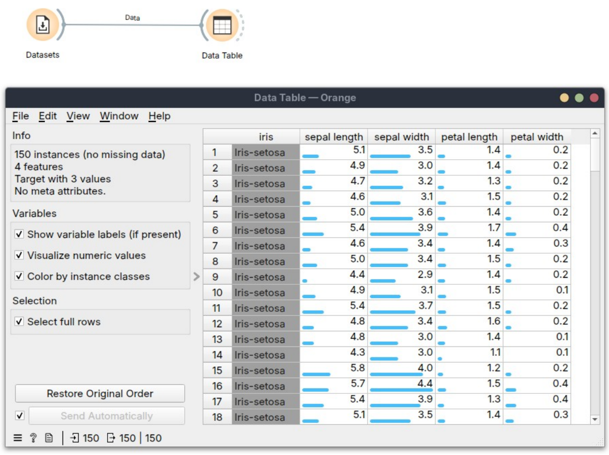
No code lab

Unsupervised learning

Before we begin, take a moment and learn about the k-Means algorithm https://www.naftaliharris.com/blog/visualizing-k-means-clustering/. Follow with https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/ to learn about the DBSCAN algorithm.

1. Use **Datasets** operator to load the *Iris* dataset. Send the data to the **Data Table** operator to get a first look at the data. The dataset is a famous collection created by R. Fisher, it contains data on 150 iris flowers belonging to three species. You can find more about this dataset in the Wikipedia article. The attributes are all numerical and represent the sizes of flowers' petals and sepals. When displaying the data in the Data Table operator, you can check the "Visualize numerical values" checkbox.



Info

150 instances (no missing data)
4 features
Target with 3 values
No meta attributes.

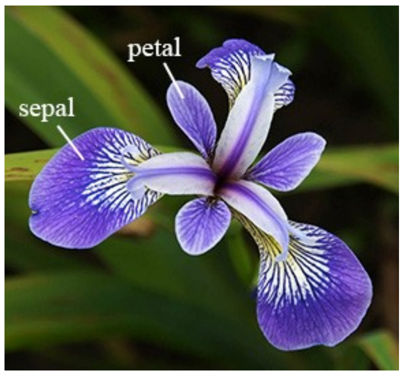
Variables

✓ Show variable labels (if present)
✓ Visualize numeric values
✓ Color by instance classes

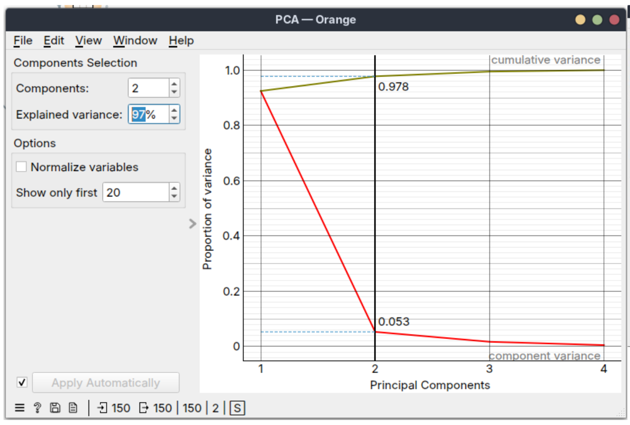
Selection

✓ Select full rows

	iris	sepal length	sepal width	petal length	petal width
1	Iris-setosa	5.1	3.5	1.4	0.2
2	Iris-setosa	4.9	3.0	1.4	0.2
3	Iris-setosa	4.7	3.2	1.3	0.2
4	Iris-setosa	4.6	3.1	1.5	0.2
5	Iris-setosa	5.0	3.6	1.4	0.2
6	Iris-setosa	5.4	3.9	1.7	0.4
7	Iris-setosa	4.6	3.4	1.4	0.3
8	Iris-setosa	5.0	3.4	1.5	0.2
9	Iris-setosa	4.4	2.9	1.4	0.2
10	Iris-setosa	4.9	3.1	1.5	0.1
11	Iris-setosa	5.4	3.7	1.5	0.2
12	Iris-setosa	4.8	3.4	1.6	0.2
13	Iris-setosa	4.8	3.0	1.4	0.1
14	Iris-setosa	4.3	3.0	1.1	0.1
15	Iris-setosa	5.8	4.0	1.2	0.2
16	Iris-setosa	5.7	4.4	1.5	0.4
17	Iris-setosa	5.4	3.9	1.3	0.4
18	Iris-setosa	5.1	3.5	1.4	0.3



2. The first operator that we will use performs Principal Component Analysis (PCA). This technique allows you to reduce the number of attributes while still maintaining the overall "shape" of the data. Drag the line from **Datasets** to the **PCA** operator from the [Unsupervised] section. Double-click at the operator. The red line shows, how much original information is contained in new attributes (called principal components). What we see is that the first new attribute (PC1) contains 92% of the original variance. You may think of it this way: *if I keep only the first principal component (instead of original 4 attributes), I will keep 92% of information from the original dataset*. The second principal component keeps 5% of the original variance, the third and fourth components hold almost no variance. The green-ish line shows the same way in the cumulative fashion (first component keeps 92% of variance, first two components hold 97% of variance, ...)
3. Drag the output of the **PCA** operator to the **Scatter Plot** operator and display the flowers using the new attributes. Use *PC1* and *PC2* as the X-axis and Y-axis, respectively, and use *Iris* as the color attribute. Can you distinguish between iris species using new attributes? Can you distinguis between iris species using only one attribute?


PCA — Orange

File Edit View Window Help

Components Selection

Components: 2

Explained variance: 97%

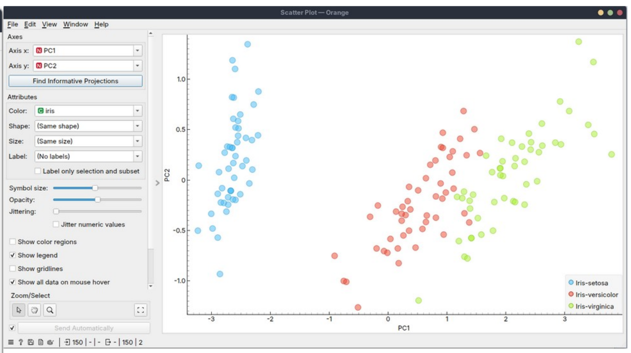
Options

☐ Normalize variables

Show only first 20

☑ Apply Automatically

150 | 150 | 150 | 2


Scatter Plot — Orange

File Edit View Window Help

Axes

Axis x: PC1

Axis y: PC2

Find Informative Projections

Attributes

Color: iris

Shape: (Same shape)

Size: (Same size)

Label: (No labels)

☐ Label only selection and subset

Symbol size:

Opacity:

Jittering:

☐ Jitter numeric values

☐ Show color regions

☑ Show legend

☐ Show gridlines

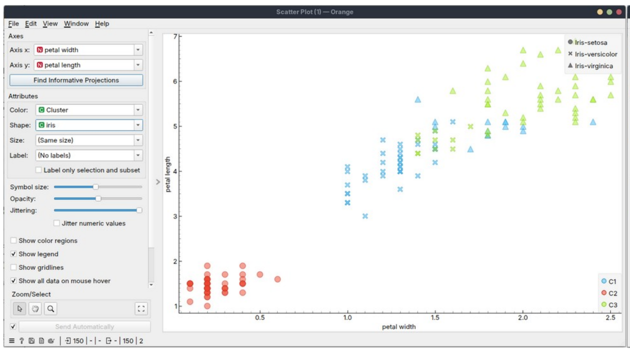
☑ Show all data on mouse hover

Zoom/Select

☑ Send Automatically

150 | - | - | - | 150 | 2

4. Let us now assume, that we don’t know how many species of iris there are. We will use the k-Means clustering algorithm to discover the groups of similar flowers. Drag the line from the **Datasets** to the **k-Means** operator (you will find it in the [Unsupervised] section). Double-click on the operator and set the number of clusters to k=3. Send the output of the **k-Means** operator to the Scatter Plot operator and draw the results. The best visualization can be obtained when you plot *petal width* and *petal length* on XY-axis. Use *Cluster* attribute to color the points, and use *Iris* attribute to set the shape of the points. If clustering is correct, all points of a given color should have the same shape (this would mean that the k-means algorithm assigned all flowers of a given species to a single cluster). Can you spot flowers for which the k-means algorithm makes an error? To better understand the result, go back to the **k-Means** operator and set the number of clusters to k=5), and repeat the visualization.


Scatter Plot (1) — Orange

File Edit View Window Help

Axes

Axis x: petal width

Axis y: petal length

Find Informative Projections

Attributes

Color: Cluster

Shape: iris

Size: (Same size)

Label: (No labels)

☐ Label only selection and subset

Symbol size:

Opacity:

Jittering:

☐ Jitter numeric values

☐ Show color regions

☑ Show legend

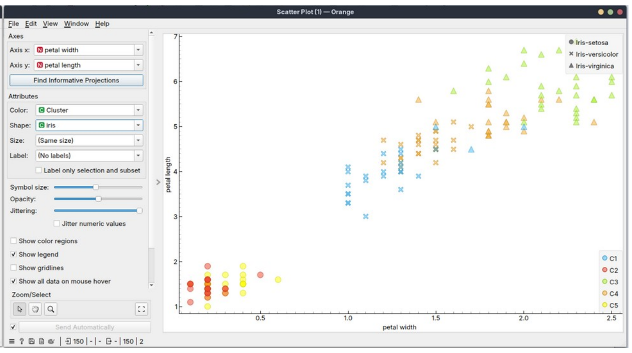
☐ Show gridlines

☑ Show all data on mouse hover

Zoom/Select

☑ Send Automatically

150 | - | - | - | 150 | 2


Scatter Plot (1) — Orange

File Edit View Window Help

Axes

Axis x: petal width

Axis y: petal length

Find Informative Projections

Attributes

Color: Cluster

Shape: iris

Size: (Same size)

Label: (No labels)

☐ Label only selection and subset

Symbol size:

Opacity:

Jittering:

☐ Jitter numeric values

☐ Show color regions

☑ Show legend

☐ Show gridlines

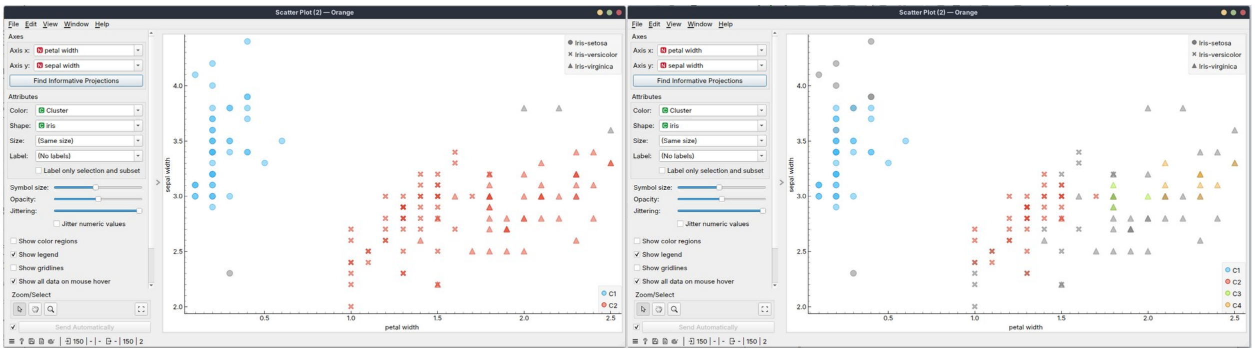
☑ Show all data on mouse hover

Zoom/Select

☑ Send Automatically

150 | - | - | - | 150 | 2

5. Finally, go back to the k-Means operator and select the radiobutton “Number of Clusters” to check multiple values of k (from 2 to 10). What is the “best” number of clusters according to the operator?
6. Compare the clustering produced by the **k-Means** operator with the clustering produced by the **DBSCAN** operator (you will find this operator in the [Visualization] section). First, simply drag the output of the DBSCAN operator to the Scatter Plot and repeat the previous visualization. What is the number of clusters found by the algorithm? Are there flowers that were not assigned to any cluster (i.e. outliers)? Double-click on the **DBSCAN** operator and set the neighborhood distance to 0.4. How many clusters are found now?

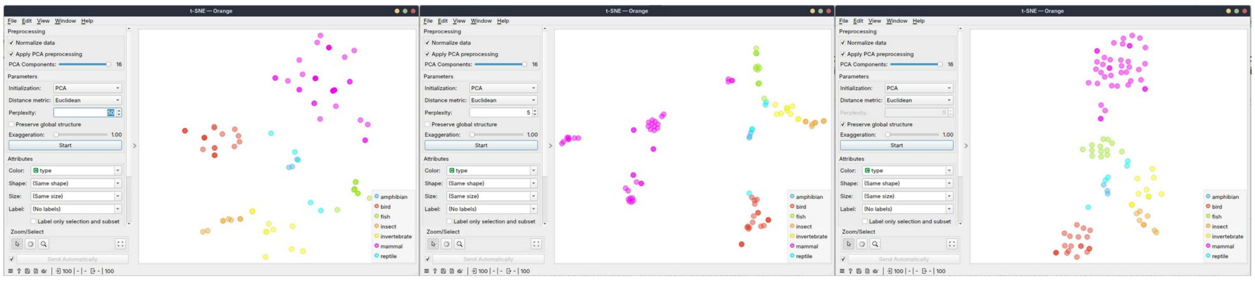

Scatter Plot (2) — Orange

Axis x: petal width
Axis y: sepal width

Color: Cluster
Shape: iris
Size: (Same size)
Label: (No labels)

☐ Show color regions
✓ Show legend
☐ Show gridlines
✓ Show all data on mouse hover

7. Use **Datasets** operator to load the *Zoo* dataset. This dataset represents over 100 different species described by many attributes such as number of legs, being aquatic, having feathers, being venomous, having fins, etc. A special *Type* attribute groups animals into mammals, fish, birds, reptiles, etc. Send the data to the **Data Table** operator to get a first look at the data.
8. From [Unsupervised] section drag the **t-SNE** operator and send the data to the operator. Open the operator to see the data projection. Make sure that the color of each data point is mapped to the *Type* attribute. See what happens when you lower perplexity to small value (1-5). Check the *Preserve global structure* checkbox and run the mapping again. As you can see, the **t-SNE** operator works similarly to **PCA**, but it always returns only two new attributes.

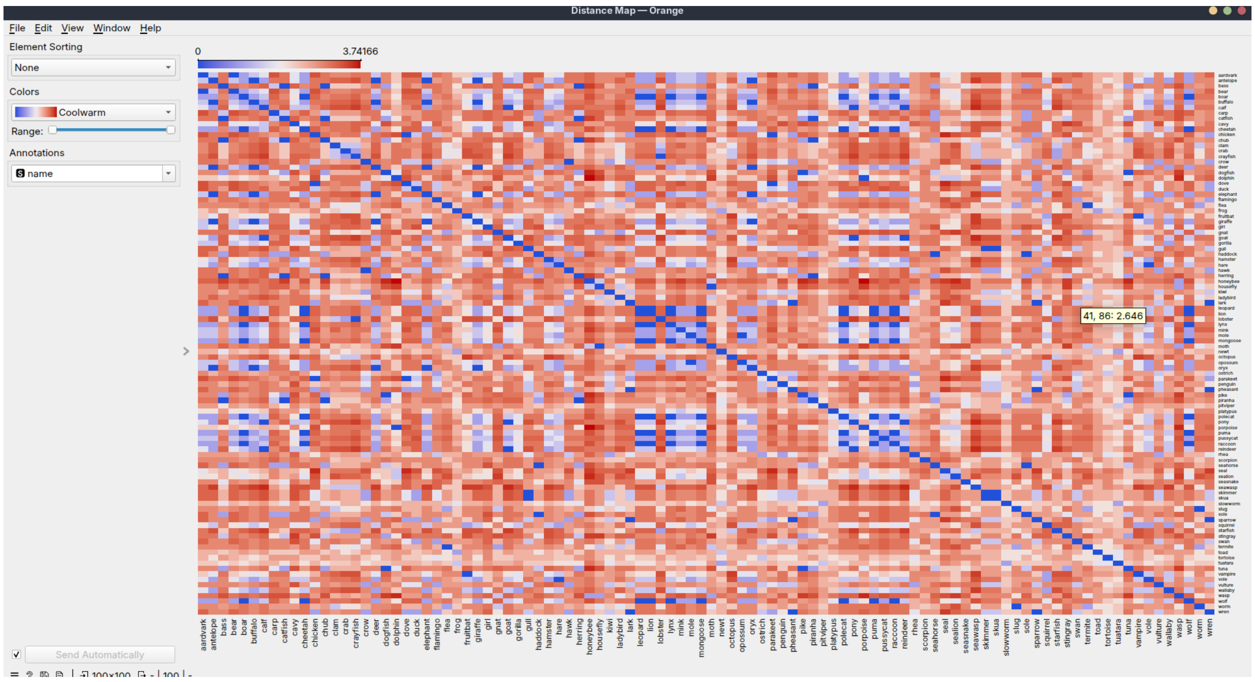

t-SNE — Orange

✓ Normalize data
✓ Apply PCA preprocessing
PCA Components: 16

Initialization: PCA
Distance metric: Euclidean
Perplexity: 50
☐ Preserve global structure
Exaggeration: 1.00

Color: type
Shape: (Same shape)
Size: (Same size)
Label: (No labels)

9. Send the *Zoo* dataset to the **Distances** operator from the [Unsupervised] section. The operator computes the distance between data points using a given distance metric. Visualize the distances using the **Distance Map** operator, simply connect the output of the **Distances** operator with the input of the **Distance Map**. Use *name* attribute to annotate the map. The default color palette is not very helpful, use “colors” dropdown listto select something more readable. Look for the leopard. Which animals are most similar (have the lowest distance) from the leopard?


Distance Map — Orange

File Edit View Window Help

Element Sorting: None

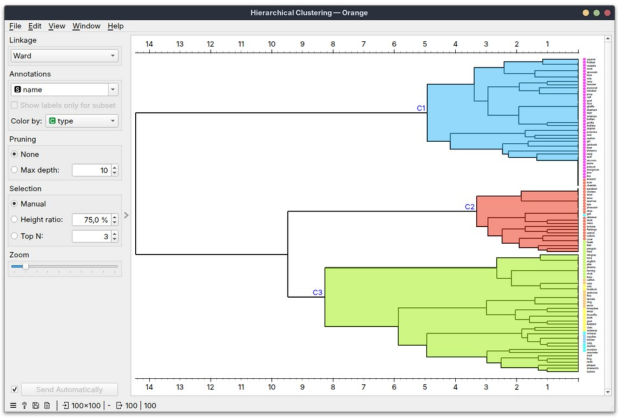
Colors: Coolwarm

Range:

Annotations: name

Send Automatically

10. Send the output of the **Distances** operator to the **Hierarchical Clustering** operator (you will find it in the [Unsupervised] section). This algorithm starts by joining pairs of animals that are most similar, and then continues by joining groups of similar animals. There are many ways in which the similarity between groups of objects can be defined, for the purpose of our experiment select *Ward* as the linkage mechanism. The result is a tree, where branches represent groups or clusters. Annotate data points using the *name* attribute, and use *type* attribute for coloring data points. Observe what happens when you increase the “Top N” parameter of cluster selection. Start with 3 clusters and try to guess the “correct” number of clusters. Use “Zoom” option to see either the entire tree or parts of it.


Hierarchical Clustering — Orange

File Edit View Window Help

Linkage: Ward

Annotations: name

Show labels only for subset

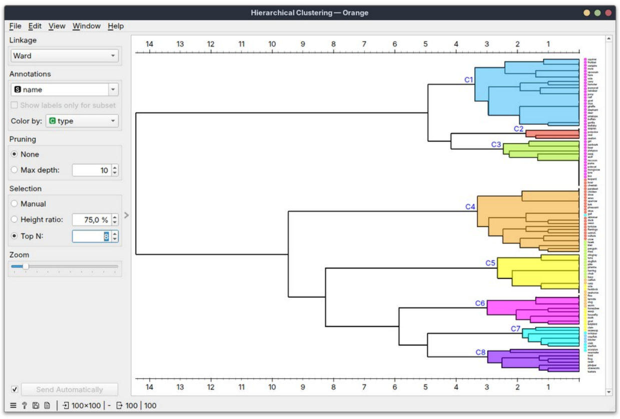
Color by: type

Pruning: None; Max depth: 10

Selection: Manual; Height ratio: 75,0 %; Top N: 3

Zoom

Send Automatically


Hierarchical Clustering — Orange

File Edit View Window Help

Linkage: Ward

Annotations: name

Show labels only for subset

Color by: type

Pruning: None; Max depth: 10

Selection: Manual; Height ratio: 75,0 %; Top N

Zoom

Send Automatically

Assignment

Use the **Datasets** operator to download the *Congressional Voting Records* dataset. This data set includes votes for each of the U.S. House of Representatives Congresspeople on the 16 key votes. Voting is binary (either yea or nay), and the party affiliation of each representative (either democrat or republican) is known. Using unsupervised learning tools try to do the following:

- Visualize the American political class using the t-SNE projection.
- Try to cluster Congress members based on their voting record. See if you can correctly predict political affiliation of Congress members from the clustering model. Experiment with both k-Means and DBSCAN algorithms.
- Construct a hierarchical clustering model using the Manhattan distance. Can you find the number of clusters that produces homogeneous groups of Congress members who share the same political agenda?