Image Embedding

Image embedding through deep neural networks.

Inputs

- Images: List of images.

Outputs

- Embeddings: Images represented with a vector of numbers.
- Skipped Images: List of images where embeddings were not calculated.

Image Embedding reads images and uploads them to a remote server or evaluate them locally. Deep learning models are used to calculate a feature vector for each image. It returns an enhanced data table with additional columns (image descriptors).

Images can be imported with Import Images widget or as paths to images in a spreadsheet. In this case the column with images paths needs a three-row header with *type=image* label in the third row.

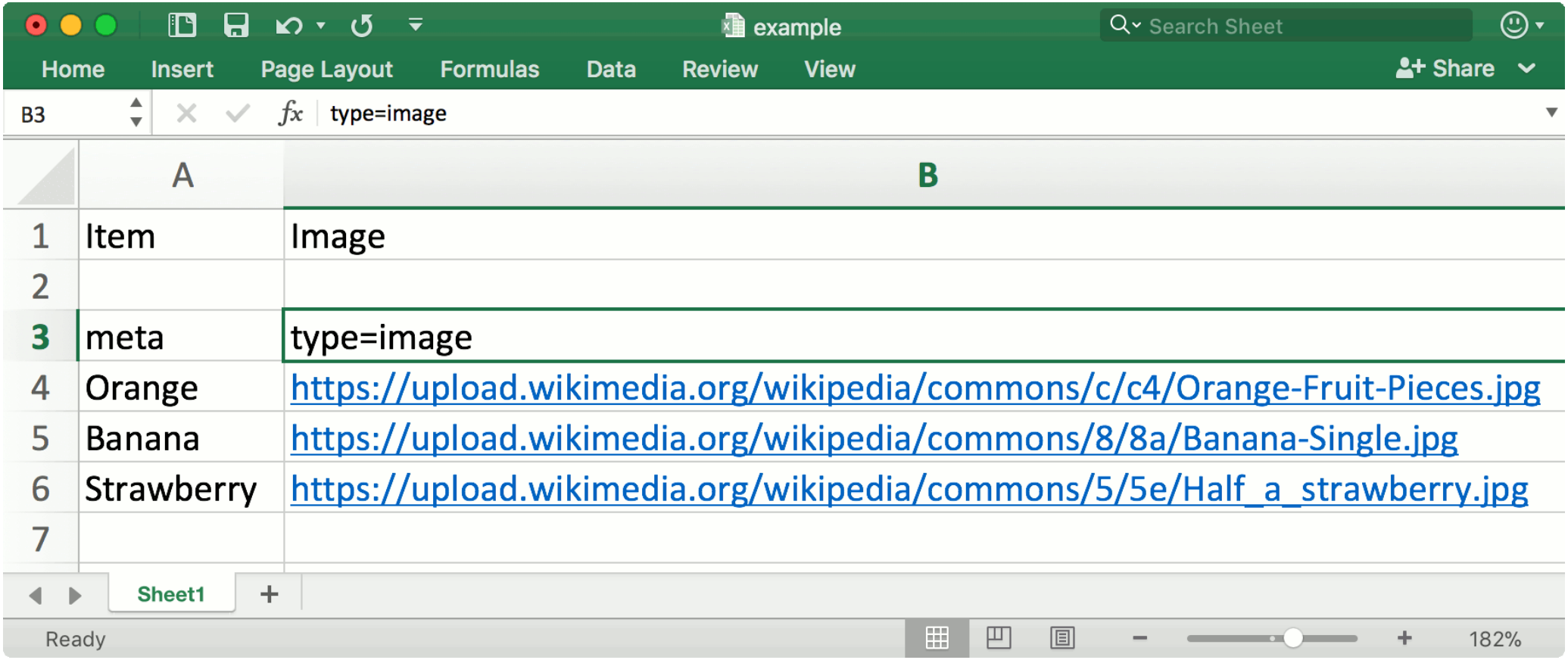
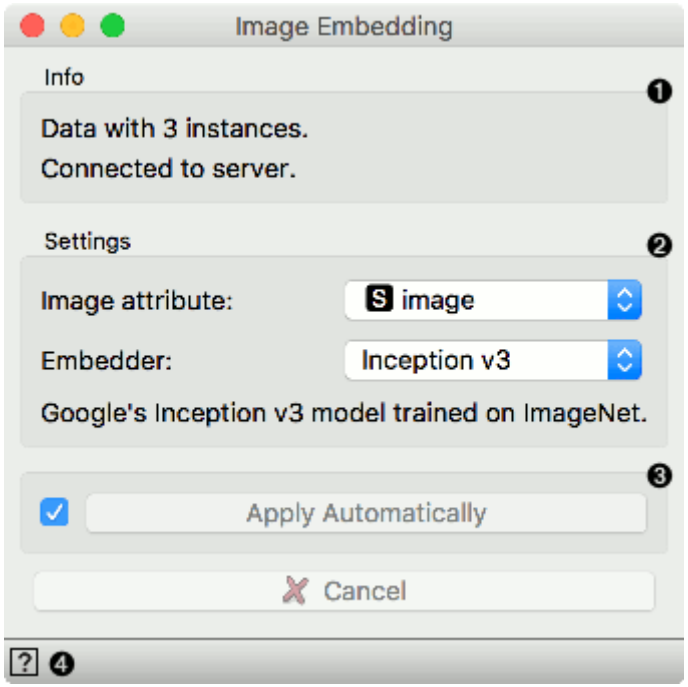


Image Embedding offers several embedders, each trained for a specific task. Images are sent to a server or they are evaluated locally on the user's computer, where vectors representations are computed. SqueezeNet embedder offers a fast evaluation on users computer which does not require an internet connection. If you decide to use other embedders than SqueezeNet, you will need an internet connection. Images sent to the server are not stored anywhere.



1. Information on the number of embedded images and images skipped.
2. Settings:
    - *Image attribute*: attribute containing images you wish to embed
    - *Embedder*:
        - SqueezeNet: Small and fast model for image recognition trained on ImageNet.
        - Inception v3: Google's Inception v3 model trained on ImageNet.
        - VGG-16: 16-layer image recognition model trained on ImageNet.
        - VGG-19: 19-layer image recognition model trained on ImageNet.
        - Painters: A model trained to predict painters from artwork images.
        - DeepLoc: A model trained to analyze yeast cell images.
3. Tick the box on the left to start the embedding automatically. Alternatively, click *Apply*. To cancel the embedding, click *Cancel*.
4. Access help.

Embedders

InceptionV3 is Google's deep neural network for image recognition. It is trained on the ImageNet data set. The model we are using is available here. For the embedding, we use the activations of the penultimate layer of the model, which represents images with vectors.

SqueezeNet is a deep model for image recognition that achieves AlexNet-level accuracy on ImageNet with 50x fewer parameters. The model is trained on the ImageNet dataset. We re-implemented the SqueezeNet by using weights from the author's pretrained model. We use activations from pre-softmax (`flatten10`) layer as an embedding.

VGG16 and **VGG19** are deep neural networks for image recognition proposed by Visual Geometry Group from the University of Oxford. They are trained on the ImageNet data set. We use a community implementation of networks with original weights. As an embedding, we use activations of the penultimate layer - `fc7` .

Image Embedding also includes **Painters**, an embedder that was trained on 79,433 images of paintings by 1,584 painters and won Kaggle's Painter by Numbers competition. Activations of the penultimate layer of the network are used as an embedding.

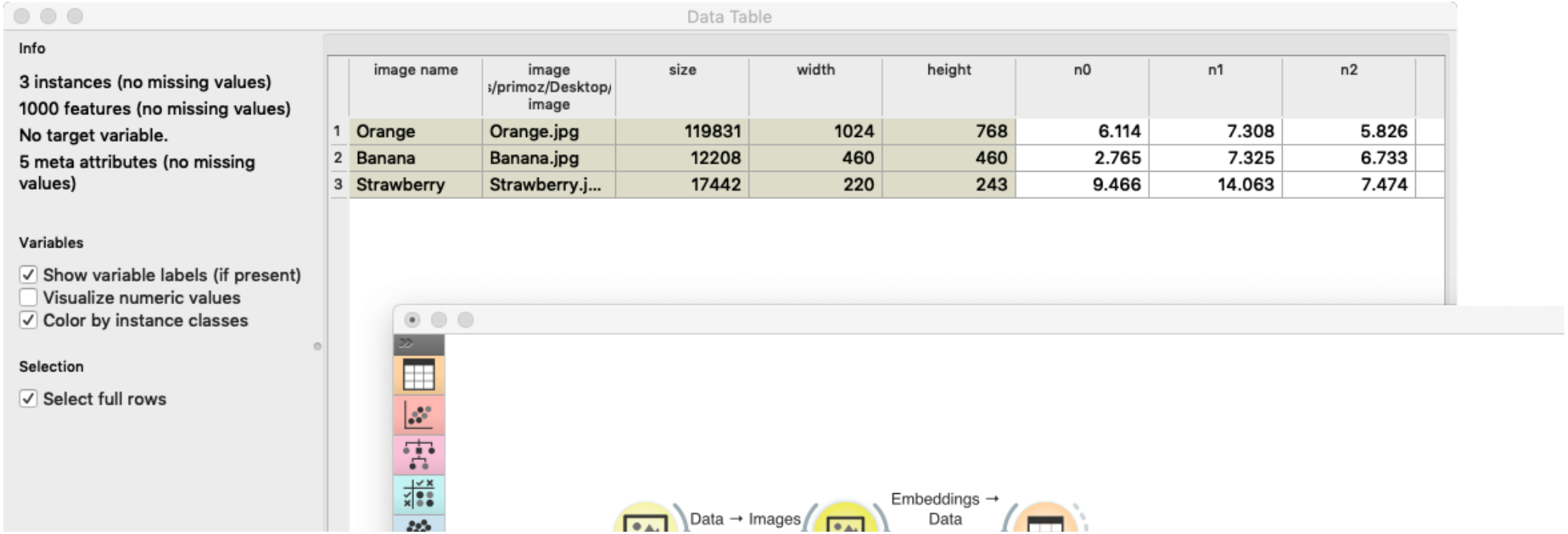
DeepLoc is a convolutional network trained on 21,882 images of single cells that were manually assigned to one of 15 localization compartments. We use the pre-trained network proposed by authors. The embeddings are activations of penultimate layer `fc_2` .

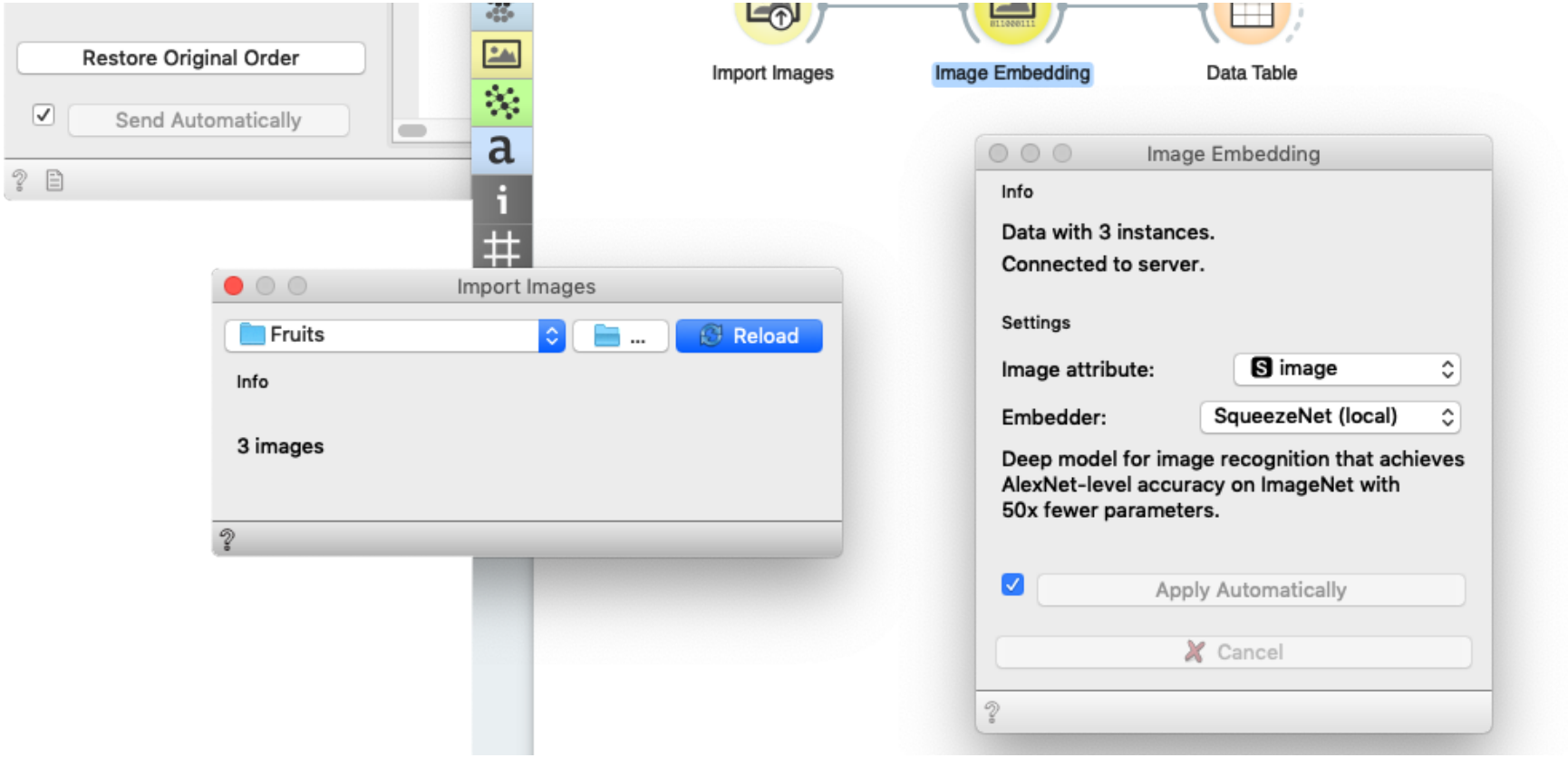
An article by Godec et al. (2019) explains how the embeddings work and how to use it in Orange.

Example

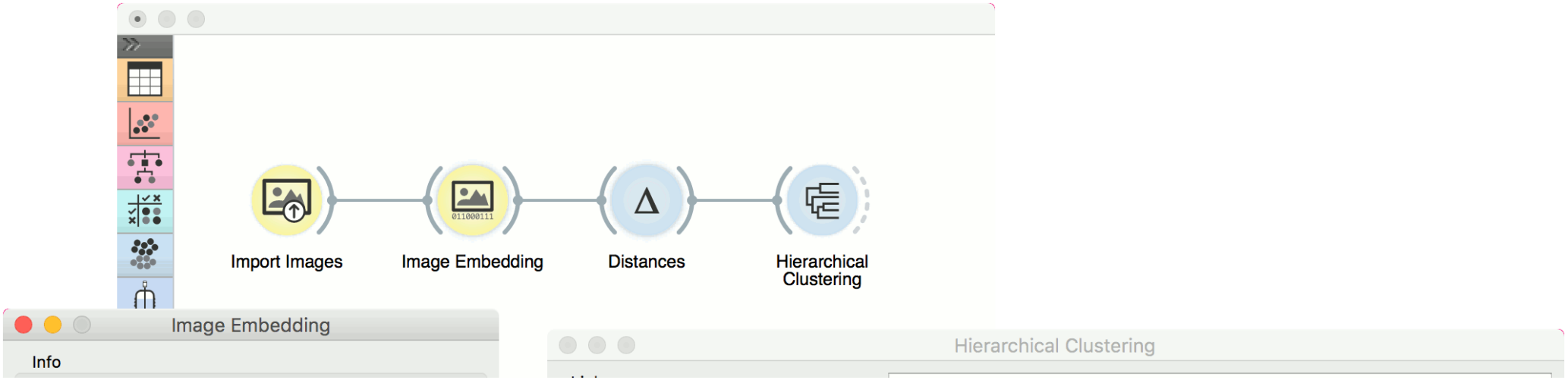
Let us first import images from a folder with Import Images. We have three images of an orange, a banana and a strawberry in a folder called Fruits. From **Import Images** we will send a data table containing a column with image paths to **Image Embedding**.

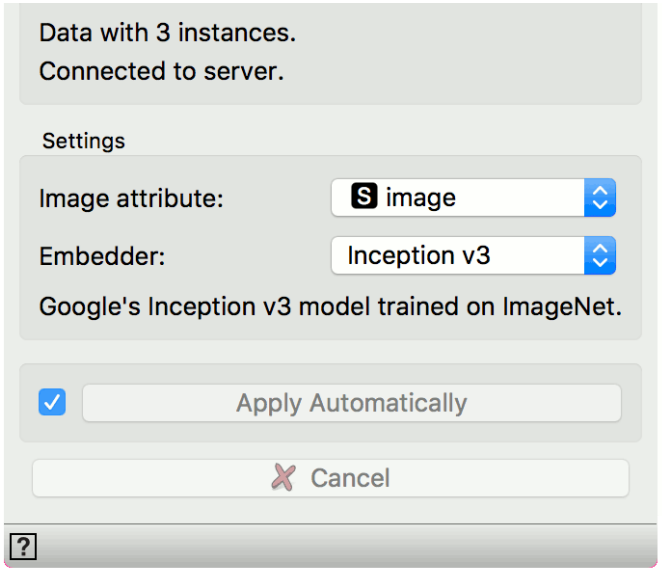
We will use the default embedder *SqueezeNet*. The widget will automatically start retrieving image vectors from the server.

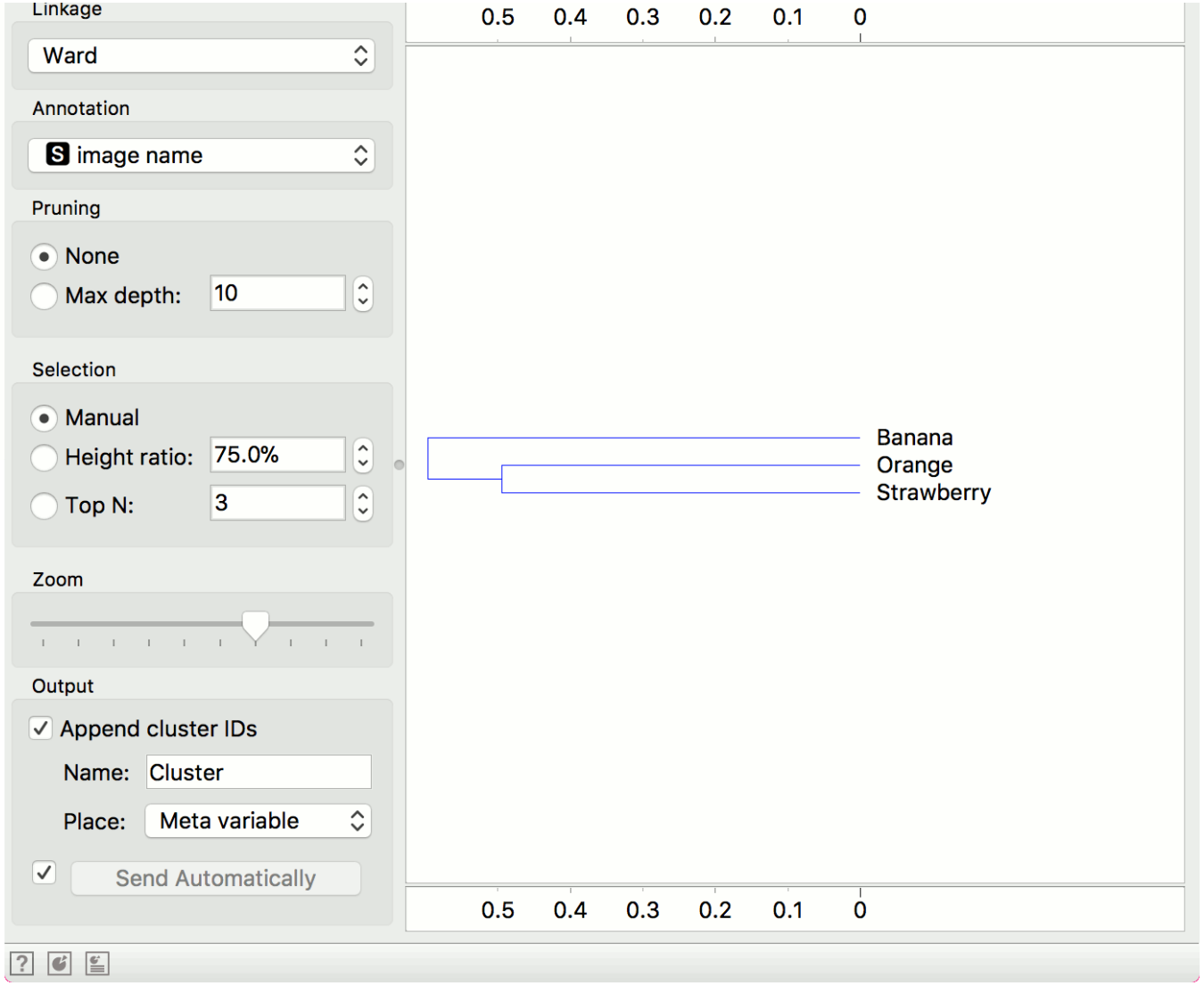




Once the computation is done, you can observe the enhanced data in a **Data Table**. With the retrieved embeddings, you can continue with any machine learning method Orange offers. Below is an example for clustering.




Data with 3 instances.
Connected to server.
Settings
Image attribute:
image
Embedder:
Inception v3
Google's Inception v3 model trained on ImageNet.
Apply Automatically
Cancel


Linkage
Ward
Annotation
image name
Pruning
None
Max depth:
10
Selection
Manual
Height ratio:
75.0%
Top N:
3
Zoom
Output
Append cluster IDs
Name:
Cluster
Place:
Meta variable
Send Automatically
0.5 0.4 0.3 0.2 0.1 0
Banana
Orange
Strawberry
0.5 0.4 0.3 0.2 0.1 0