Data Sampler

Selects a subset of data instances from an input dataset.

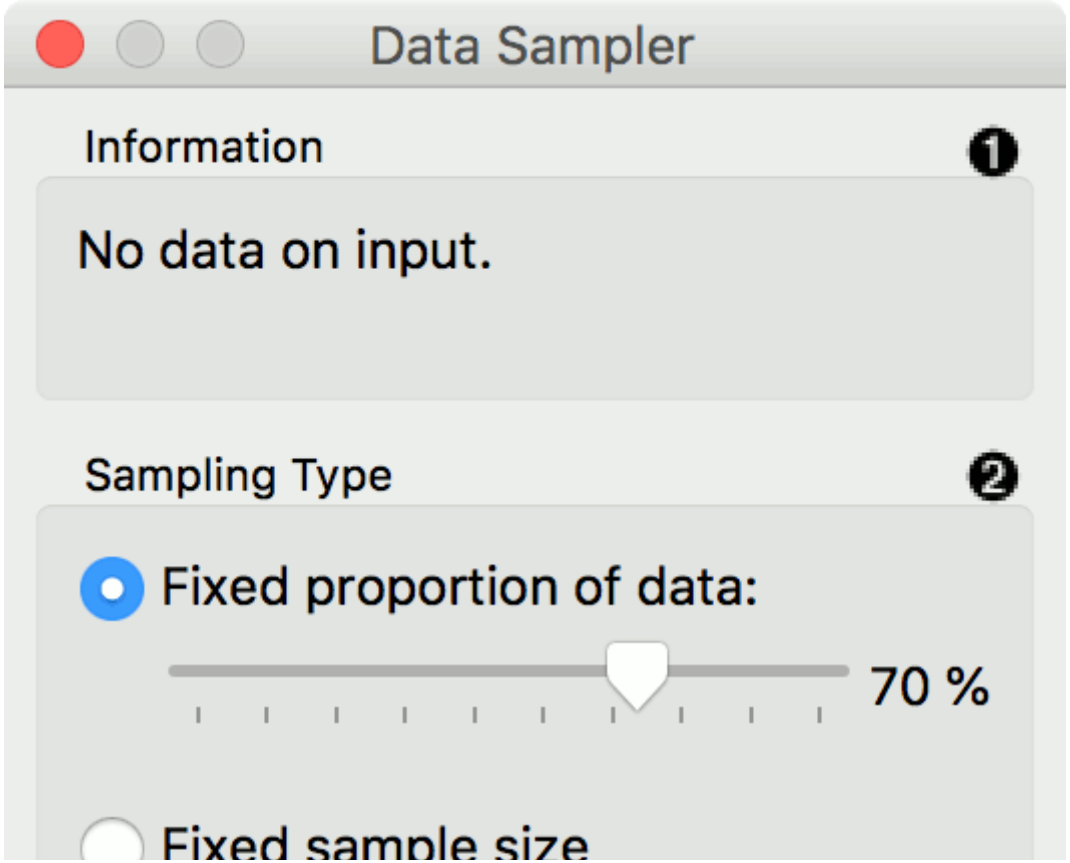
Inputs

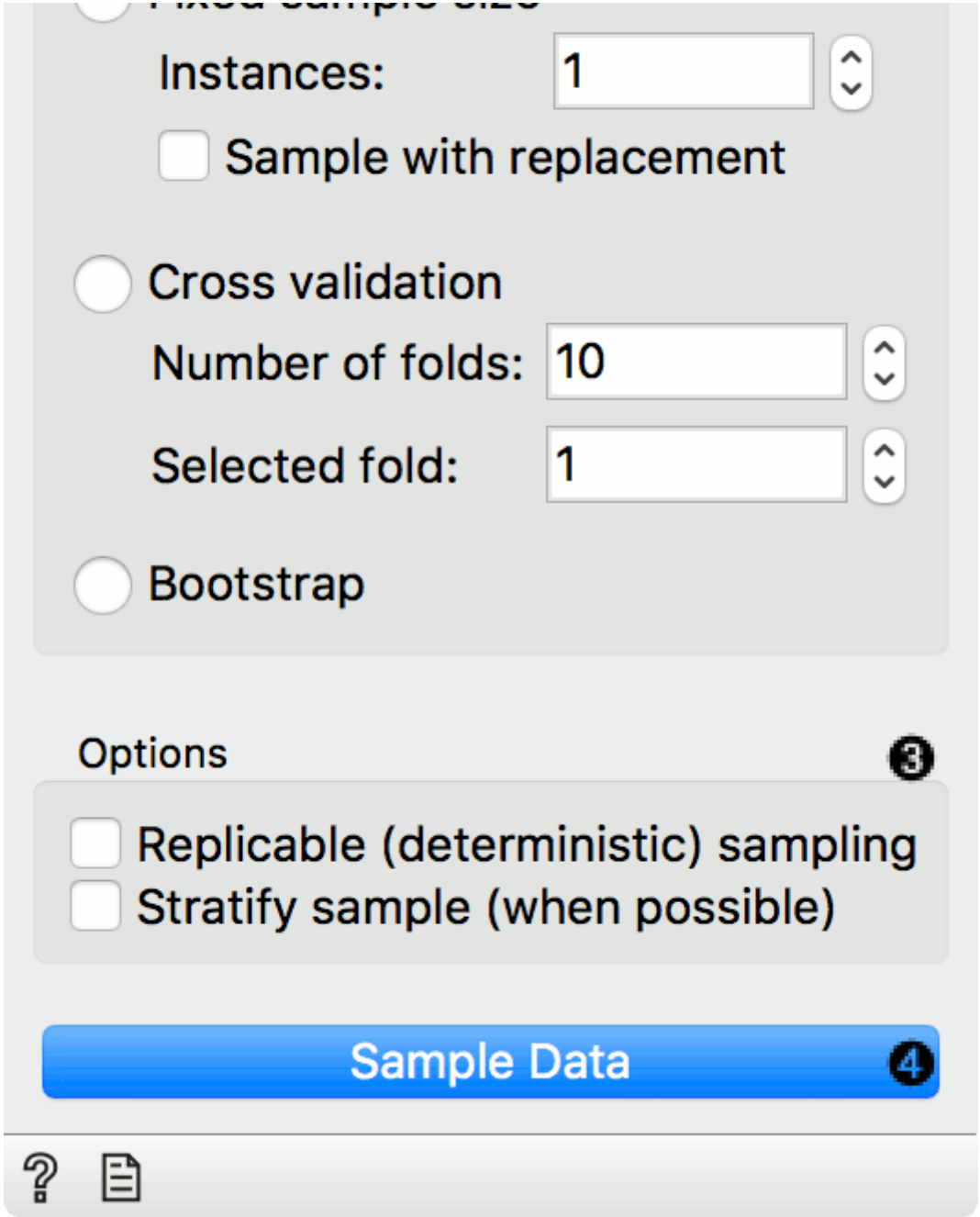
- Data: input dataset

Outputs

- Data Sample: sampled data instances
- Remaining Data: out-of-sample data

The **Data Sampler** widget implements several data sampling methods. It outputs a sampled and a complementary dataset (with instances from the input set that are not included in the sampled dataset). The output is processed after the input dataset is provided and *Sample Data* is pressed.





1. Information on the input and output dataset.
2. The desired sampling method:

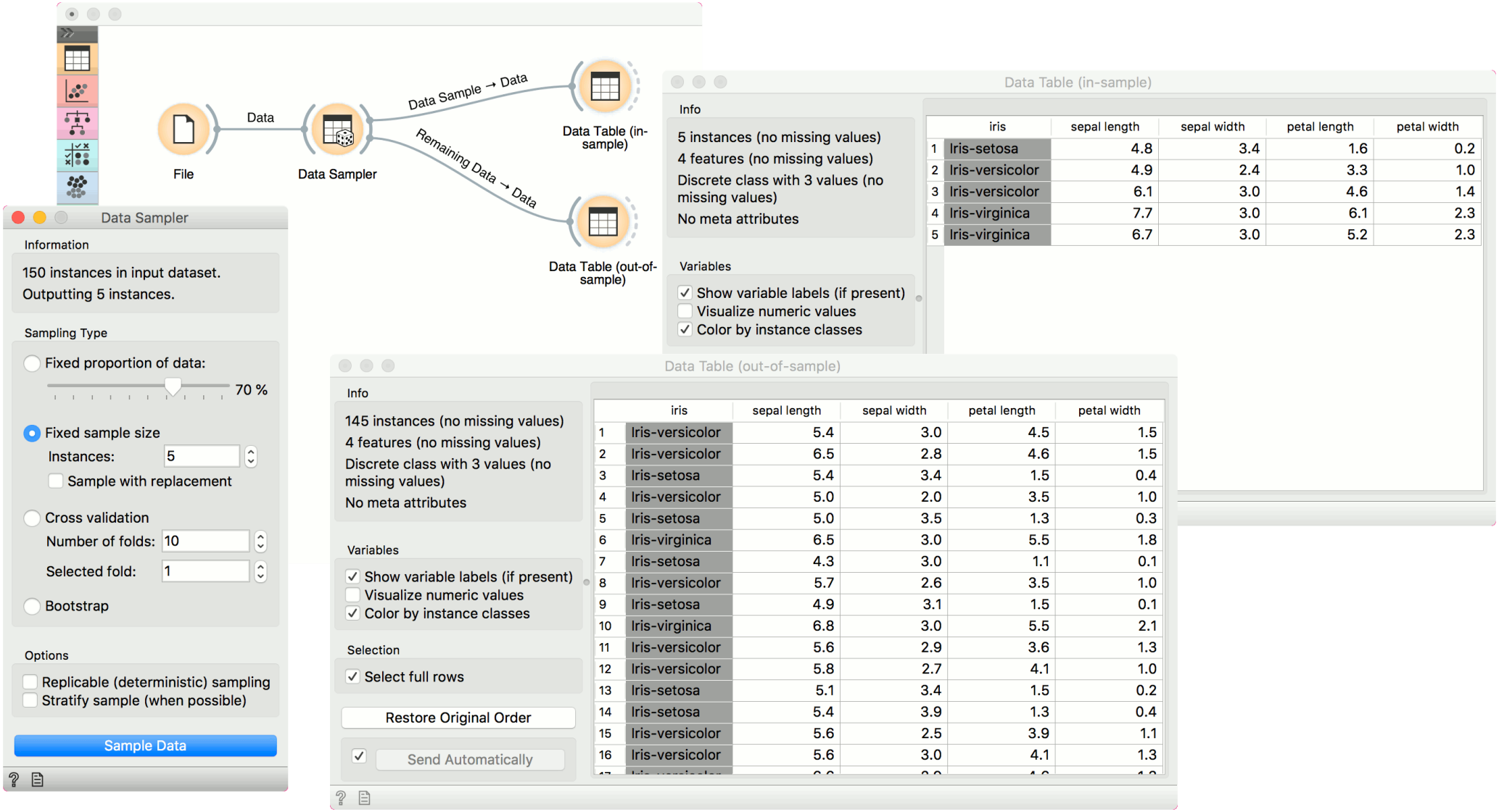
- **Fixed proportion of data** returns a selected percentage of the entire data (e.g. 70% of all the data)
- **Fixed sample size** returns a selected number of data instances with a chance to set *Sample with replacement*, which always samples from the entire dataset (does not subtract instances already in the subset). With replacement, you can generate more instances than available in the input dataset.
- Cross Validation partitions data instances into the specified number of complementary subsets. Following a typical validation schema, all subsets except the one selected by the user are output as Data Sample, and the selected subset goes to Remaining Data. (Note: In older versions, the outputs were swapped. If the widget is loaded from an older workflow, it switches to compatibility mode.)
- Bootstrap infers the sample from the population statistic.

3. *Replicable sampling* maintains sampling patterns that can be carried across users, while *stratify sample* mimics the composition of the input dataset.
4. Press *Sample Data* to output the data sample.

If all data instances are selected (by setting the proportion to 100 % or setting the fixed sample size to the entire data size), output instances are still shuffled.

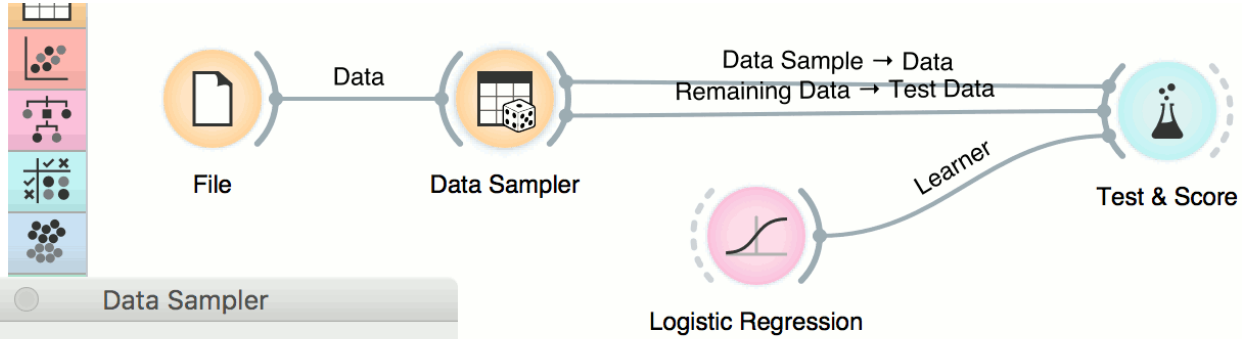
Examples

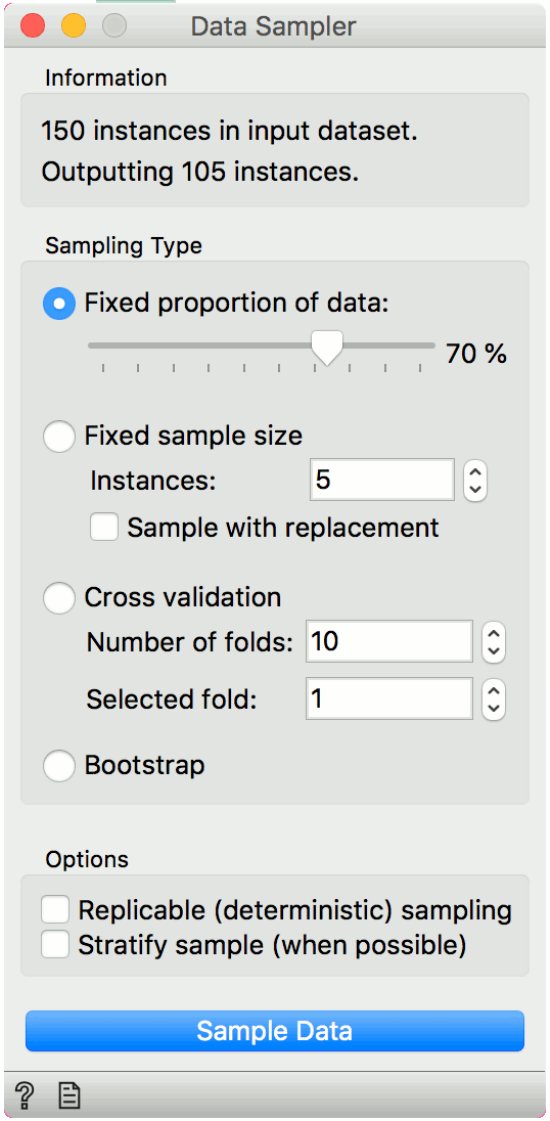
First, let's see how the **Data Sampler** works. We will use the *iris* data from the File widget. We see there are 150 instances in the data. We sampled the data with the **Data Sampler** widget and we chose to go with a fixed sample size of 5 instances for simplicity. We can observe the sampled data in the Data Table widget (Data Table (in-sample)). The second Data Table (Data Table (out-of-sample)) shows the remaining 145 instances that weren't in the sample. To output the out-of-sample data, double-click the connection between the widgets and rewire the output to *Remaining Data –> Data*.

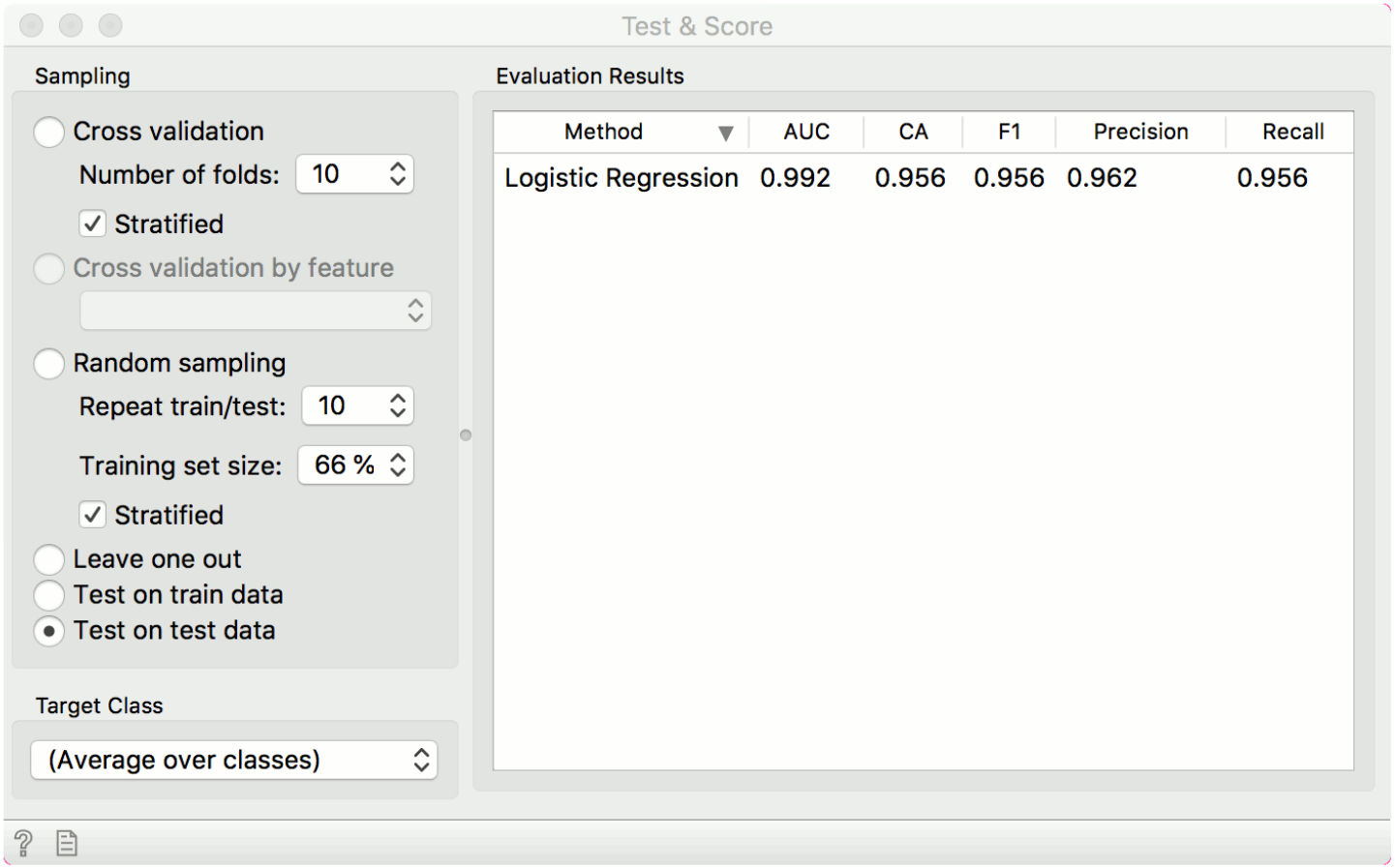


Now, we will use the **Data Sampler** to split the data into training and testing part. We are using the *iris* data, which we loaded with the File widget. In **Data Sampler**, we split the data with *Fixed proportion of data*, keeping 70% of data instances in the sample.

Then we connected two outputs to the Test & Score widget, *Data Sample –> Data* and *Remaining Data –> Test Data*. Finally, we added Logistic Regression as the learner. This runs logistic regression on the Data input and evaluates the results on the Test Data.


Data
File
Data Sampler
Data Sample → Data
Remaining Data → Test Data
Learner
Test & Score
Logistic Regression


Data Sampler
Information
150 instances in input dataset.
Outputting 105 instances.
Sampling Type
Fixed proportion of data:
70 %
Fixed sample size
Instances: 5
Sample with replacement
Cross validation
Number of folds: 10
Selected fold: 1
Bootstrap
Options
Replicable (deterministic) sampling
Stratify sample (when possible)
Sample Data

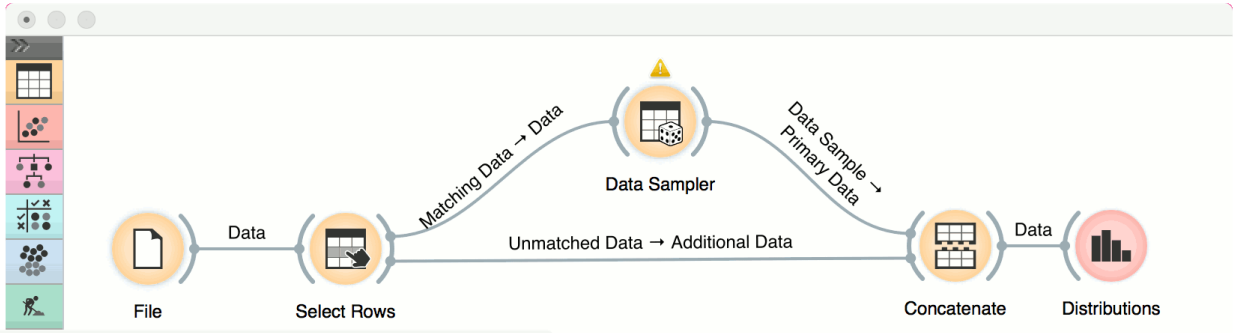

Test & Score
Sampling
Cross validation
Number of folds: 10
Stratified
Cross validation by feature
Random sampling
Repeat train/test: 10
Training set size: 66 %
Stratified
Leave one out
Test on train data
Test on test data
Target Class
(Average over classes)
Evaluation Results
Method AUC CA F1 Precision Recall
Logistic Regression 0.992 0.956 0.956 0.962 0.956

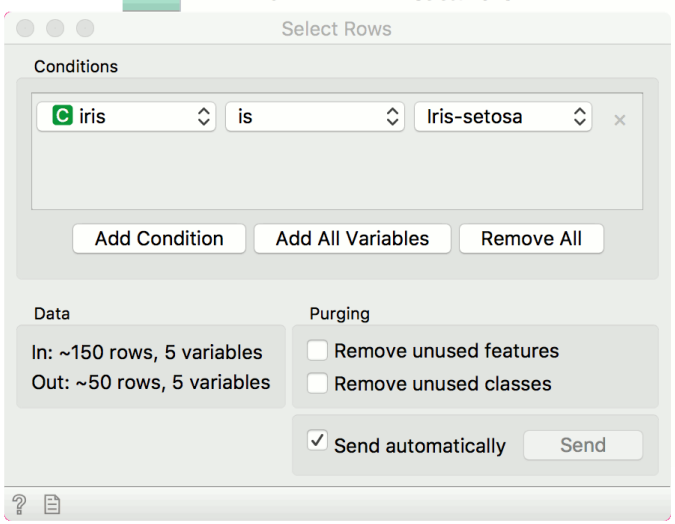
Over/Undersampling

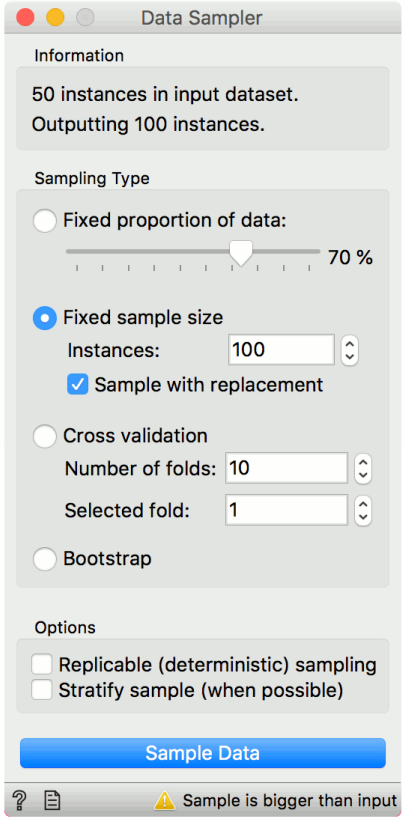
Data Sampler can also be used to oversample a minority class or undersample majority class in the data. Let us show an example for oversampling. First, separate the minority class using a Select Rows widget. We are using the *iris* data from the File widget. The data set has 150 data instances, 50 of each class. Let us oversample, say, *iris-setosa*.

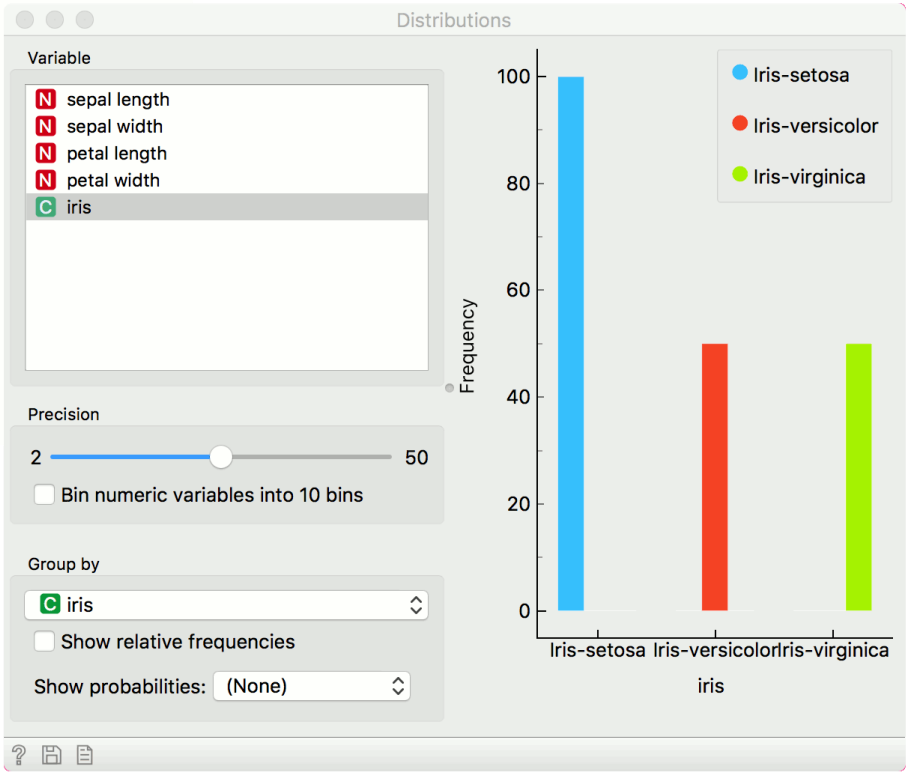
In **Select Rows**, set the condition to *iris is iris-setosa*. This will output 50 instances of the *iris-setosa* class. Now, connect *Matching Data*into the **Data Sampler**, select *Fixed sample size*, set it to, say, 100 and select *Sample with replacement*. Upon pressing *Sample Data*, the widget will output 100 instances of *iris-setosa* class, some of which will be duplicated (because we used *Sample with replacement*).

Finally, use Concatenate to join the oversampled instances and the *Unmatched Data* output of the **Select Rows**widget. This outputs a data set with 200 instances. We can observe the final results in the Distributions.


Data
Matching Data → Data
Data Sampler
Data Sample → Primary Data
Unmatched Data → Additional Data
Data
File
Select Rows
Concatenate
Distributions


Select Rows
Conditions
iris
is
Iris-setosa
Add Condition
Add All Variables
Remove All
Data
In: ~150 rows, 5 variables
Out: ~50 rows, 5 variables
Purging
Remove unused features
Remove unused classes
Send automatically
Send


Data Sampler
Information
50 instances in input dataset.
Outputting 100 instances.
Sampling Type
Fixed proportion of data:
70 %
Fixed sample size
Instances: 100
Sample with replacement
Cross validation
Number of folds: 10
Selected fold: 1
Bootstrap
Options
Replicable (deterministic) sampling
Stratify sample (when possible)
Sample Data
Sample is bigger than input


Distributions
Variable
sepal length
sepal width
petal length
petal width
iris
Precision
2
50
Bin numeric variables into 10 bins
Group by
iris
Show relative frequencies
Show probabilities: (None)
Frequency
100
80
60
40
20
0
Iris-setosa
Iris-versicolor
Iris-virginica
Iris-setosa Iris-versicolor Iris-virginica
iris