Stochastic Gradient Descent

Minimize an objective function using a stochastic approximation of gradient descent.

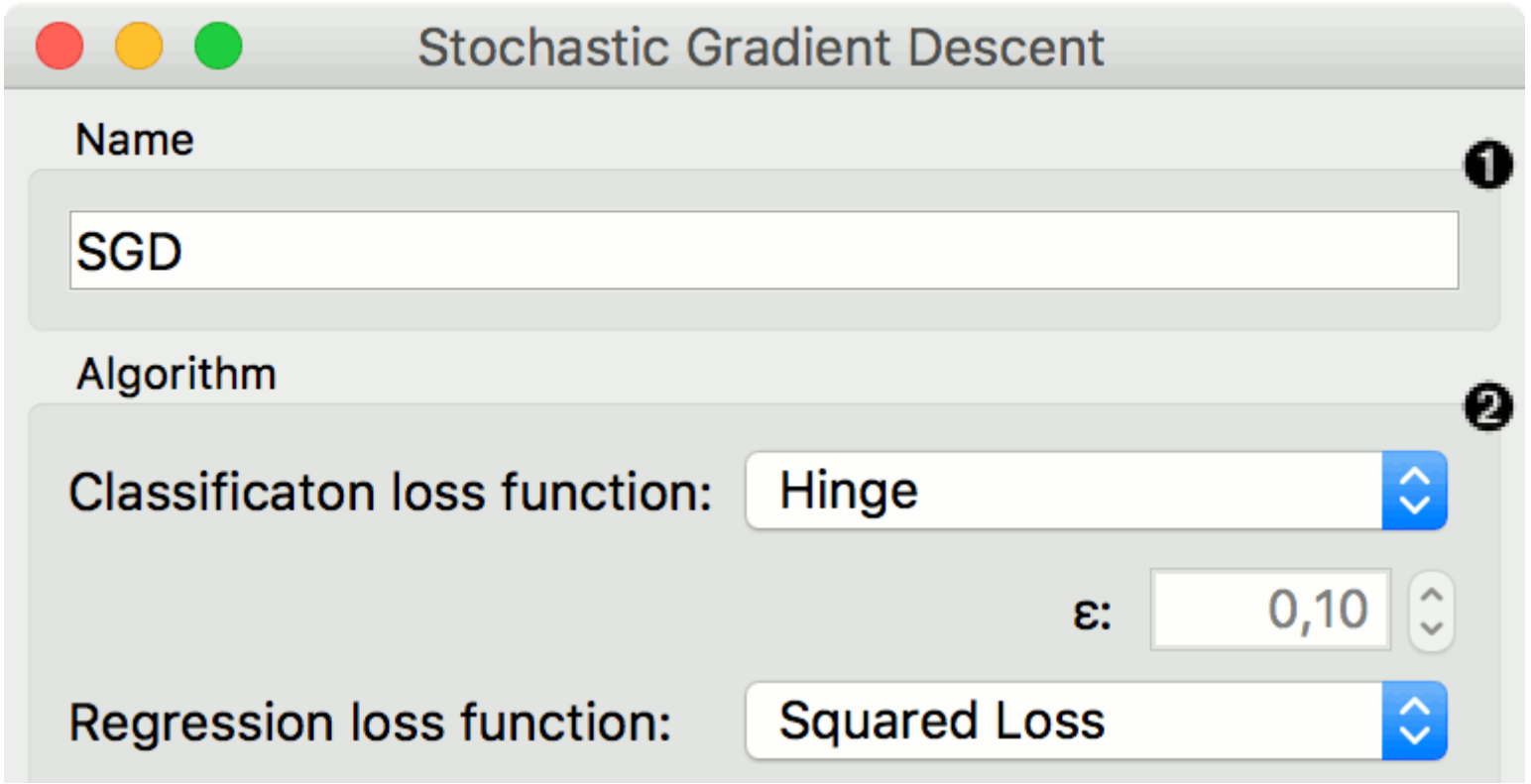
Inputs

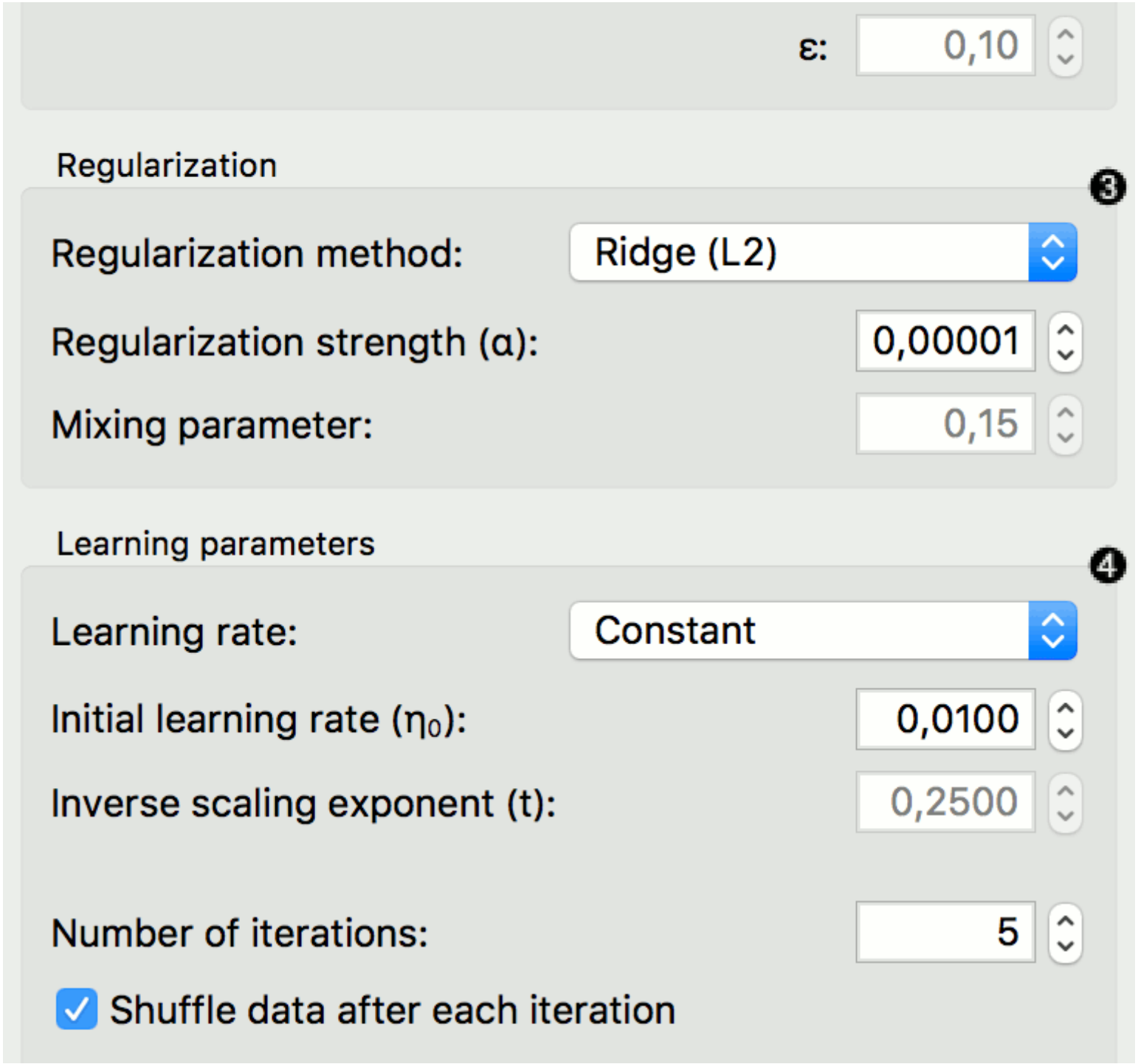
- Data: input dataset
- Preprocessor: preprocessing method(s)

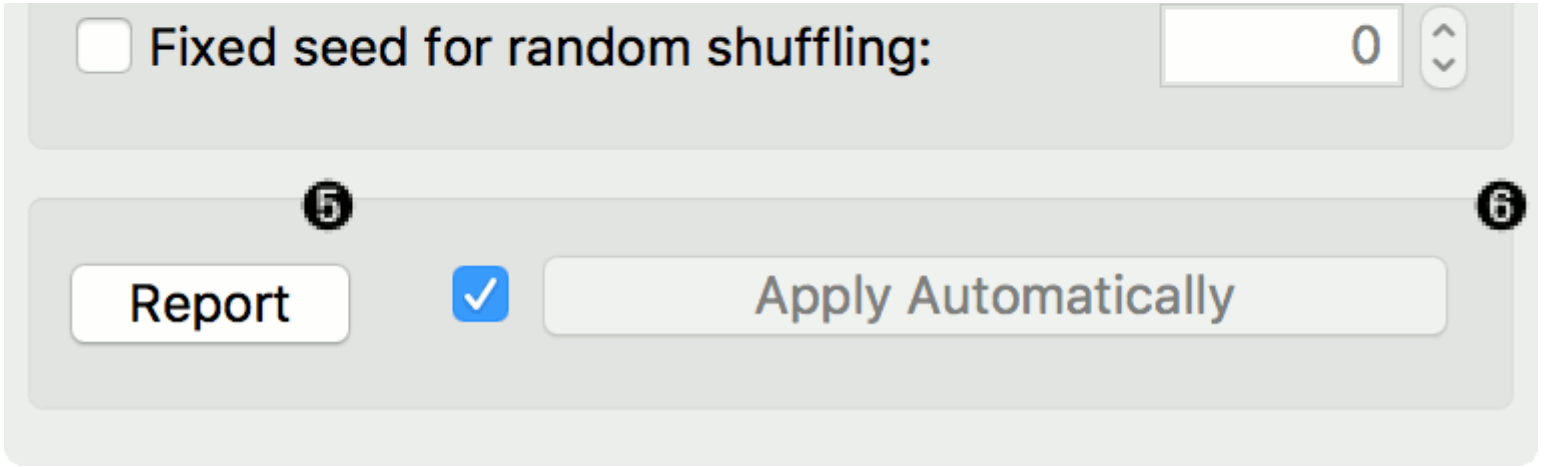
Outputs

- Learner: stochastic gradient descent learning algorithm
- Model: trained model

The **Stochastic Gradient Descent** widget uses stochastic gradient descent that minimizes a chosen loss function with a linear function. The algorithm approximates a true gradient by considering one sample at a time, and simultaneously updates the model based on the gradient of the loss function. For regression, it returns predictors as minimizers of the sum, i.e. M-estimators, and is especially useful for large-scale and sparse datasets.




ε: 0,10
Regularization
3
Regularization method: Ridge (L2)
Regularization strength (α): 0,00001
Mixing parameter: 0,15
Learning parameters
4
Learning rate: Constant
Initial learning rate (η0): 0,0100
Inverse scaling exponent (t): 0,2500
Number of iterations: 5
Shuffle data after each iteration



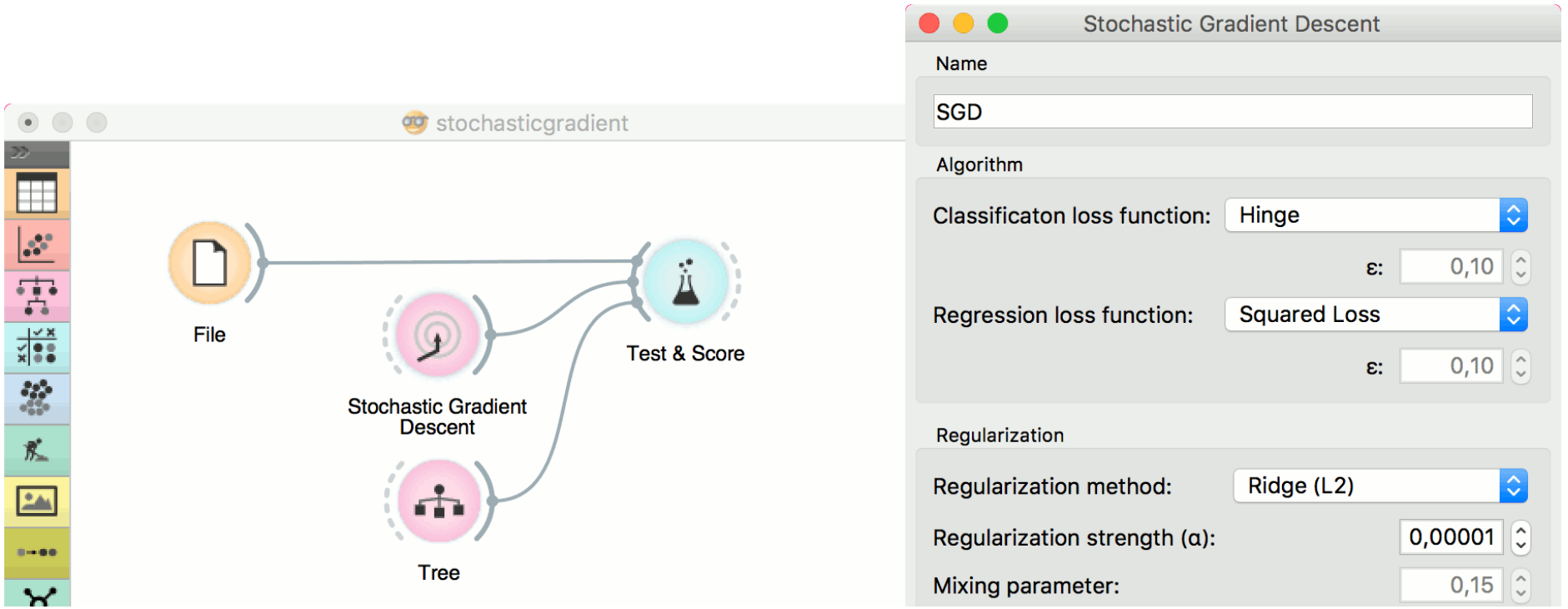
1. Specify the name of the model. The default name is “SGD”.
2. Algorithm parameters:
    - Classification loss function:
        - Hinge (linear SVM)
        - Logistic Regression(logistic regression SGD)
        - Modified Huber (smooth loss that brings tolerance to outliers as well as probability estimates)
        - *Squared Hinge* (quadratically penalized hinge)
        - Perceptron(linear loss used by the perceptron algorithm)
        - Squared Loss (fitted to ordinary least-squares)
        - Huber (switches to linear loss beyond ε)
        - Epsilon insensitive (ignores errors within ε, linear beyond it)
        - *Squared epsilon insensitive* (loss is squared beyond ε-region).
    - Regression loss function:
        - Squared Loss (fitted to ordinary least-squares)
        - Huber (switches to linear loss beyond ε)
        - Epsilon insensitive (ignores errors within ε, linear beyond it)
        - *Squared epsilon insensitive* (loss is squared beyond ε-region).
3. Regularization norms to prevent overfitting:
    - None.
    - Lasso (L1) (L1 leading to sparse solutions)
    - Ridge (L2) (L2, standard regularizer)
    - Elastic net (mixing both penalty norms).

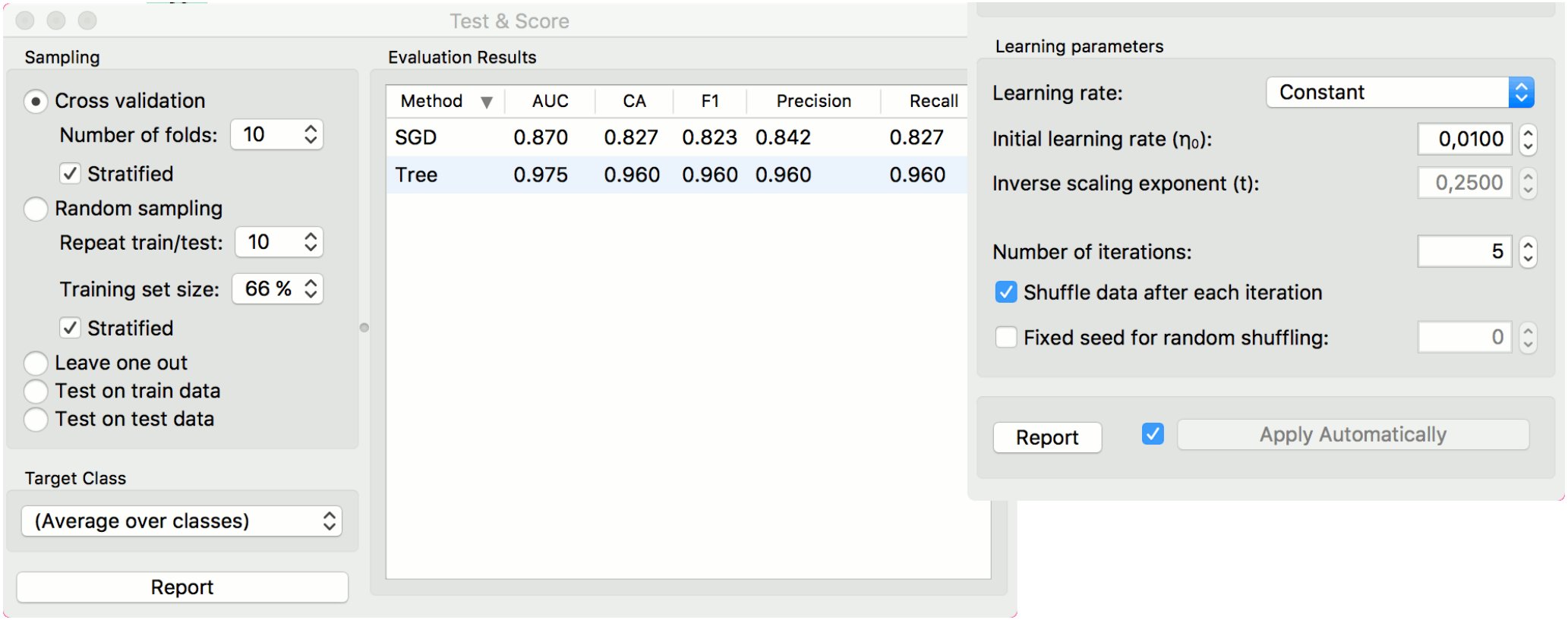
    Regularization strength defines how much regularization will be applied (the less we regularize, the more we allow the model to fit the data) and the mixing parameter what the ratio between L1 and L2 loss will be (if set to 0 then the loss is L2, if set to 1 then it is L1).

4. Learning parameters.
    - Learning rate:
        - *Constant*: learning rate stays the same through all epochs (passes)
        - Optimal: a heuristic proposed by Leon Bottou
        - Inverse scaling: earning rate is inversely related to the number of iterations
    - Initial learning rate.
    - Inverse scaling exponent: learning rate decay.
    - Number of iterations: the number of passes through the training data.
    - If *Shuffle data after each iteration* is on, the order of data instances is mixed after each pass.
    - If *Fixed seed for random shuffling* is on, the algorithm will use a fixed random seed and enable replicating the results.
5. Produce a report.
6. Press *Apply* to commit changes. Alternatively, tick the box on the left side of the *Apply* button and changes will be communicated automatically.

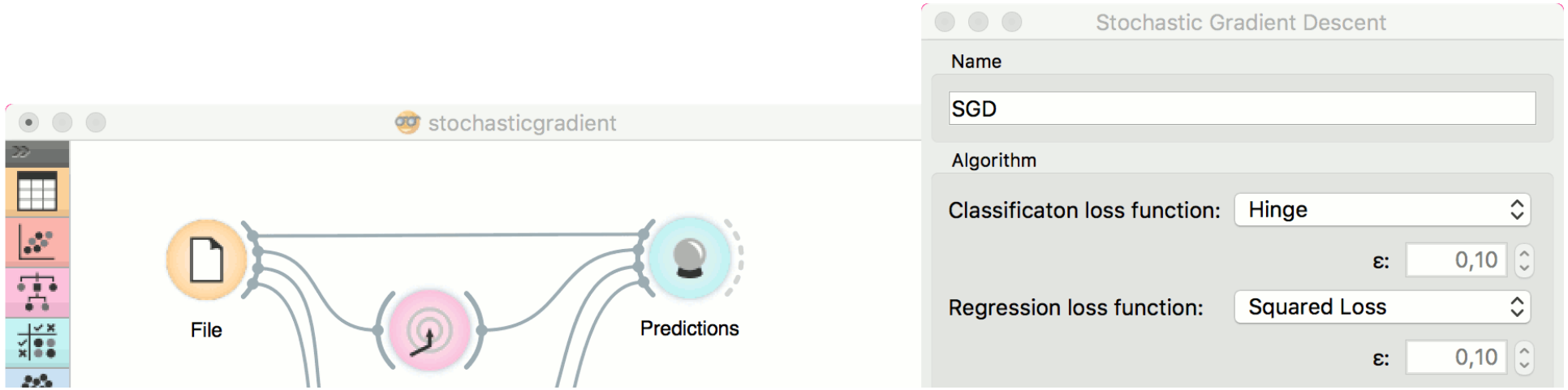
Examples

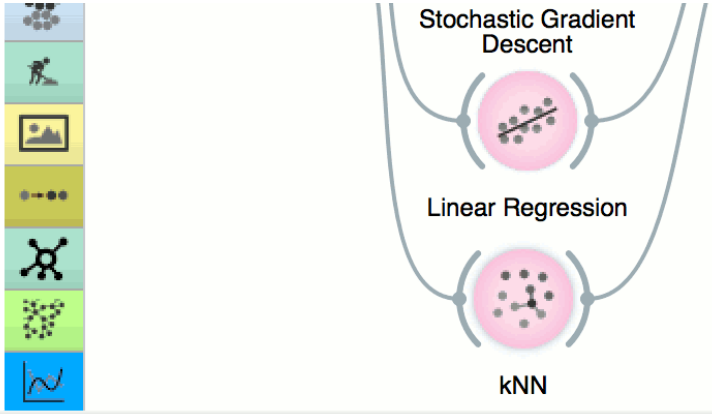
For the classification task, we will use *iris* dataset and test two models on it. We connected Stochastic Gradient Descent and Tree to Test & Score. We also connected File to **Test & Score** and observed model performance in the widget.





For the regression task, we will compare three different models to see which predict what kind of results. For the purpose of this example, the *housing* dataset is used. We connect the File widget to **Stochastic Gradient Descent**, Linear Regression and kNN widget and all four to the Predictions widget.





Regularization

Regularization method: Ridge (L2)

Regularization strength (α): 0,00001

Mixing parameter: 0,15

Learning parameters

Learning rate: Constant

0,0100

0,2500

5

eration

uffling: 0

Apply Automatically

Predictions

Info

Data: 506 instances.
Predictors: 3
Task: Regression

Restore Original Order

Data View

Show full data set

Output

Original data
Predictions
Probabilities

Report

	SGD	Linear Regression	kNN	MEDV	CRIM	ZN
1	28.292	30.004	21.780	24.000	0.006	18.000
2	24.281	25.026	22.900	21.600	0.027	0.000
3	27.426	30.568	25.360	34.700	0.027	0.000
4	25.272	28.607	26.060	33.400	0.032	0.000
5	24.817	27.944	27.100	36.200	0.069	0.000
6	23.460	25.256	27.100	28.700	0.030	0.000
7	23.308	23.002	20.880	22.900	0.088	12.500
8	20.959	19.536	19.100	27.100	0.145	12.500
9	15.658	11.524	18.400	16.500	0.211	12.500
10	20.186	18.920	19.480	18.900	0.170	12.500
11	20.226	18.999	19.280	15.000	0.225	12.500
12	22.214	21.587	22.000	18.900	0.117	12.500
13	21.613	20.907	24.340	21.700	0.094	12.500