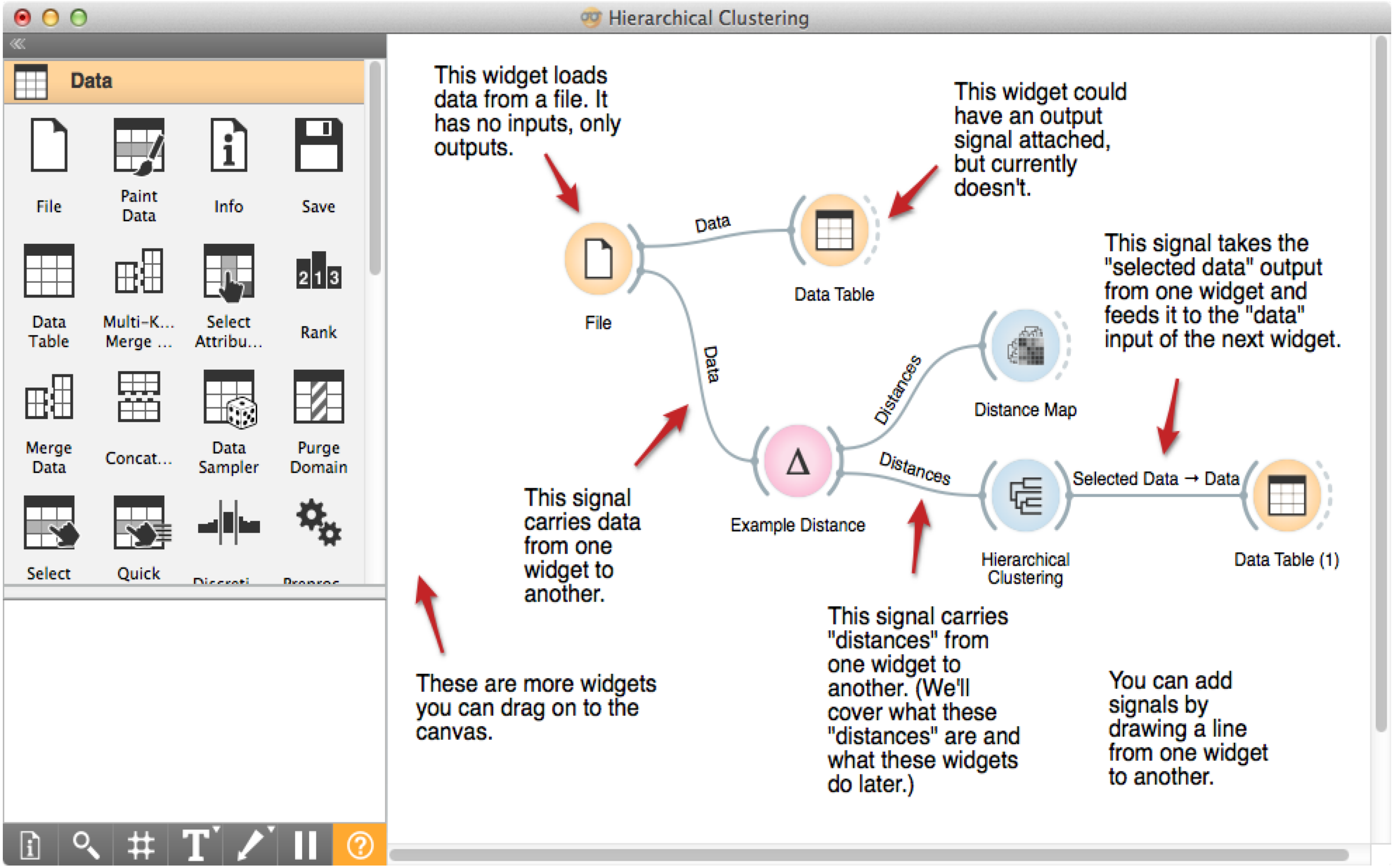
Introduction to Orange

Introduction to Orange

- Orange is a data mining toolkit, so you don’t need to be an expert in any of those subjects
- We will use Orange to:
  - load, manipulate, and save large data sets
  - visualize the relationships between variables
  - discover and quantify patterns in data
  - create rules to predict outcomes based on observed data

Orange: Graphical Programming



This widget loads data from a file. It has no inputs, only outputs.

This widget could have an output signal attached, but currently doesn't.

This signal takes the "selected data" output from one widget and feeds it to the "data" input of the next widget.

This signal carries data from one widget to another.

These are more widgets you can drag on to the canvas.

This signal carries "distances" from one widget to another. (We'll cover what these "distances" are and what these widgets do later.)

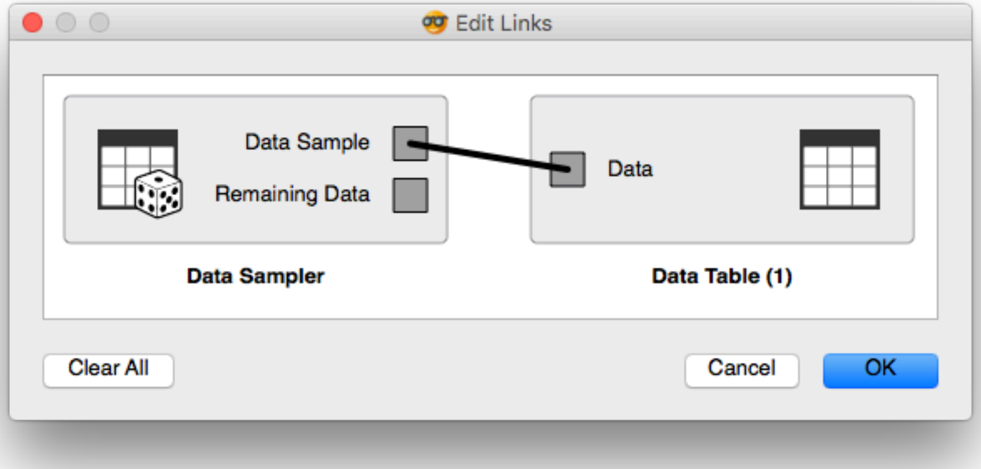
You can add signals by drawing a line from one widget to another.

Using the Orange interface

- To add a widget, drag it onto the canvas from the widget panel, or just click on it in the widget panel
- To add a signal, click on the signal attachment point on a widget and drag from it to the signal attachment point on another widget
    - Input signals come in from the left, output signals go out to the right

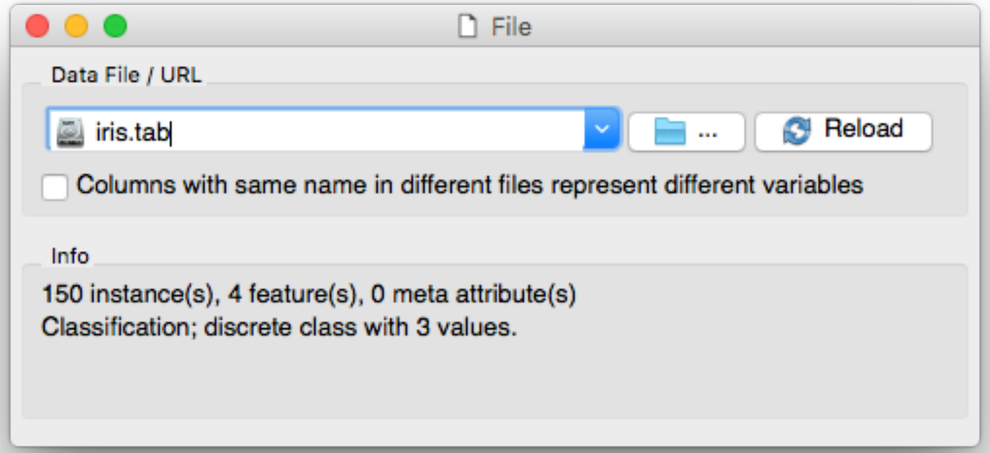
Using the Orange interface

- Some widgets have multiple possible input and output ports
  - Orange tries to guess which one you mean
  - If it guesses wrong, double click on the signal to select which inputs and outputs you are using
  - You can also temporarily disconnect or delete signals by right-clicking on them



File Widget

- Loads data from a file
- Many different file types are supported
  - Recommended: tab-delimited text
- *iris.tab* is an example dataset that comes with Orange, and contains 150 iris flowers from three species



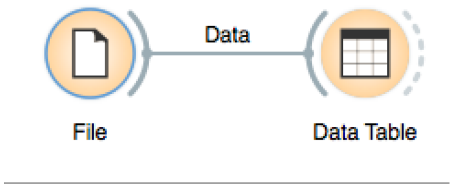
iris.tab

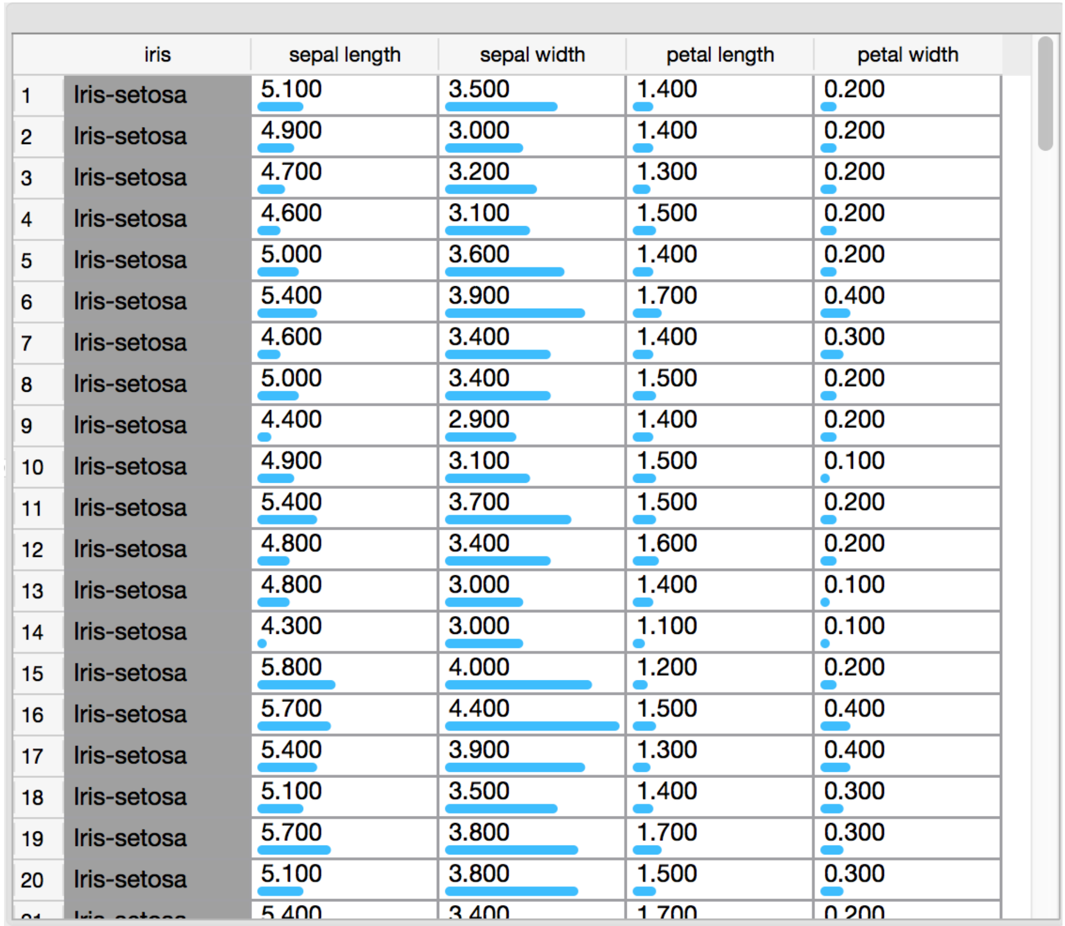
□ Columns with same name in different files represent different variables

150 instance(s), 4 feature(s), 0 meta attribute(s)
Classification; discrete class with 3 values.

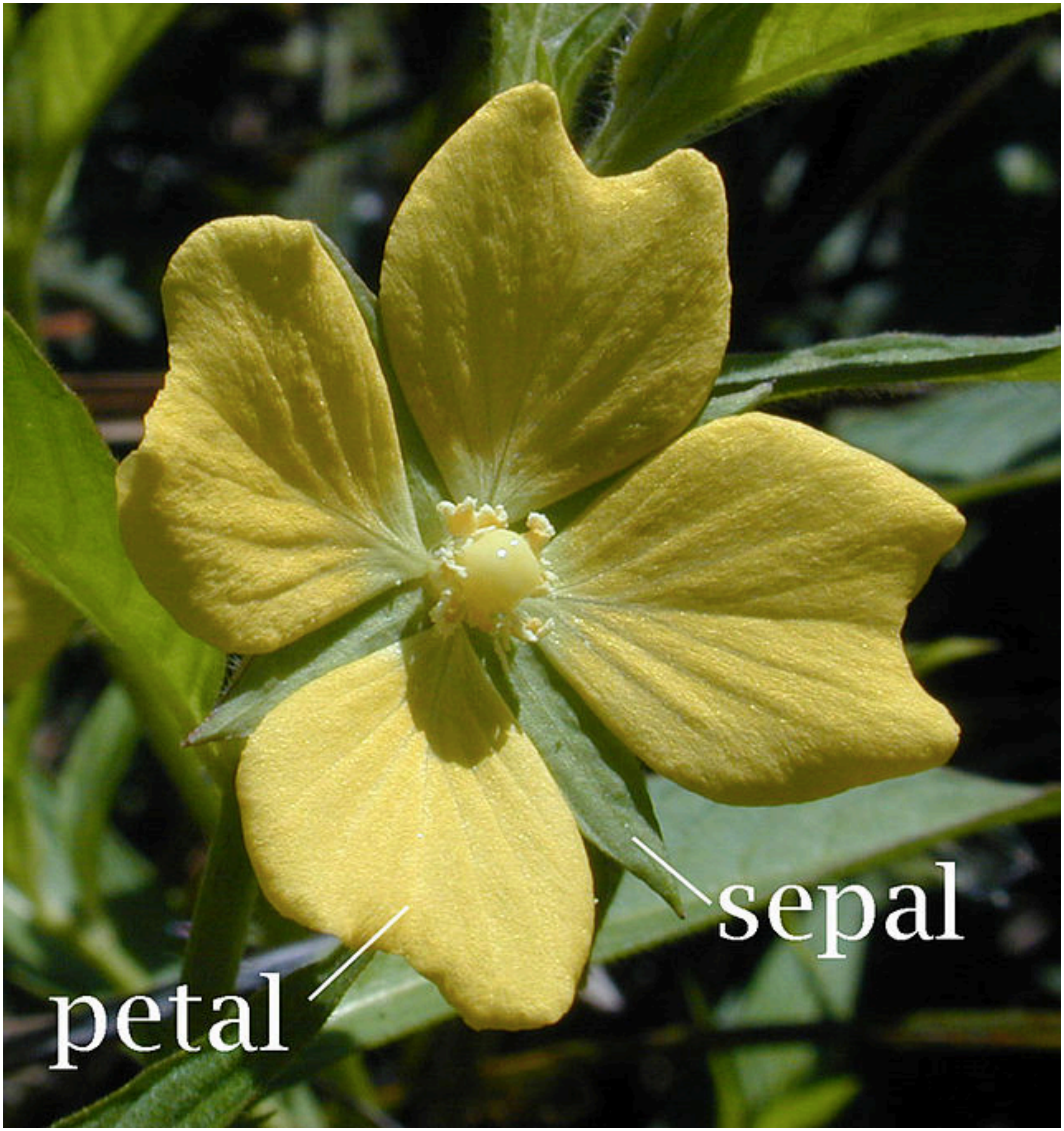
Data Table Widget

- Lists rows in a dataset, sort by clicking on the column heading
- Each value has a bar showing how big it is
- First column is assumed to be a category (in this case, species)





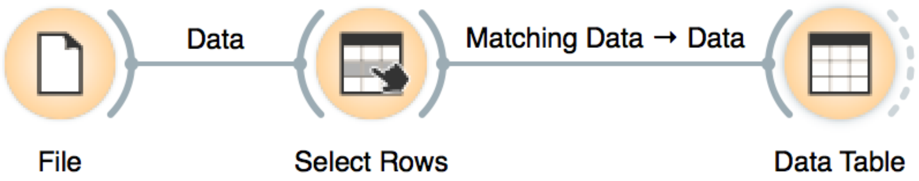
	iris	sepal length	sepal width	petal length	petal width
1	Iris-setosa	5.100	3.500	1.400	0.200
2	Iris-setosa	4.900	3.000	1.400	0.200
3	Iris-setosa	4.700	3.200	1.300	0.200
4	Iris-setosa	4.600	3.100	1.500	0.200
5	Iris-setosa	5.000	3.600	1.400	0.200
6	Iris-setosa	5.400	3.900	1.700	0.400
7	Iris-setosa	4.600	3.400	1.400	0.300
8	Iris-setosa	5.000	3.400	1.500	0.200
9	Iris-setosa	4.400	2.900	1.400	0.200
10	Iris-setosa	4.900	3.100	1.500	0.100
11	Iris-setosa	5.400	3.700	1.500	0.200
12	Iris-setosa	4.800	3.400	1.600	0.200
13	Iris-setosa	4.800	3.000	1.400	0.100
14	Iris-setosa	4.300	3.000	1.100	0.100
15	Iris-setosa	5.800	4.000	1.200	0.200
16	Iris-setosa	5.700	4.400	1.500	0.400
17	Iris-setosa	5.400	3.900	1.300	0.400
18	Iris-setosa	5.100	3.500	1.400	0.300
19	Iris-setosa	5.700	3.800	1.700	0.300
20	Iris-setosa	5.100	3.800	1.500	0.300
21	Iris-setosa	5.400	3.400	1.700	0.200

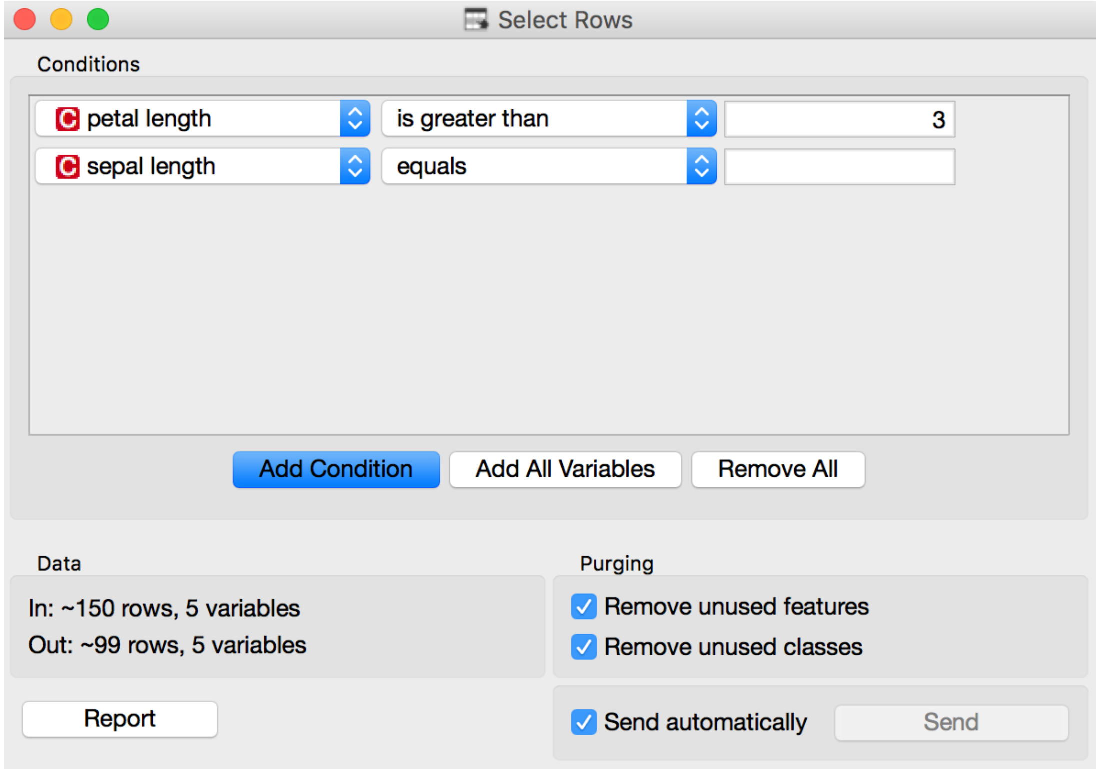


For each of the 150 flowers in the dataset, there is a value for:

- Petal Length
- Petal Width
- Sepal Length
- Sepal Width

Select Rows Widget





Conditions

petal length	is greater than	3
sepal length	equals	

Data

In: ~150 rows, 5 variables

Out: ~99 rows, 5 variables

Purging

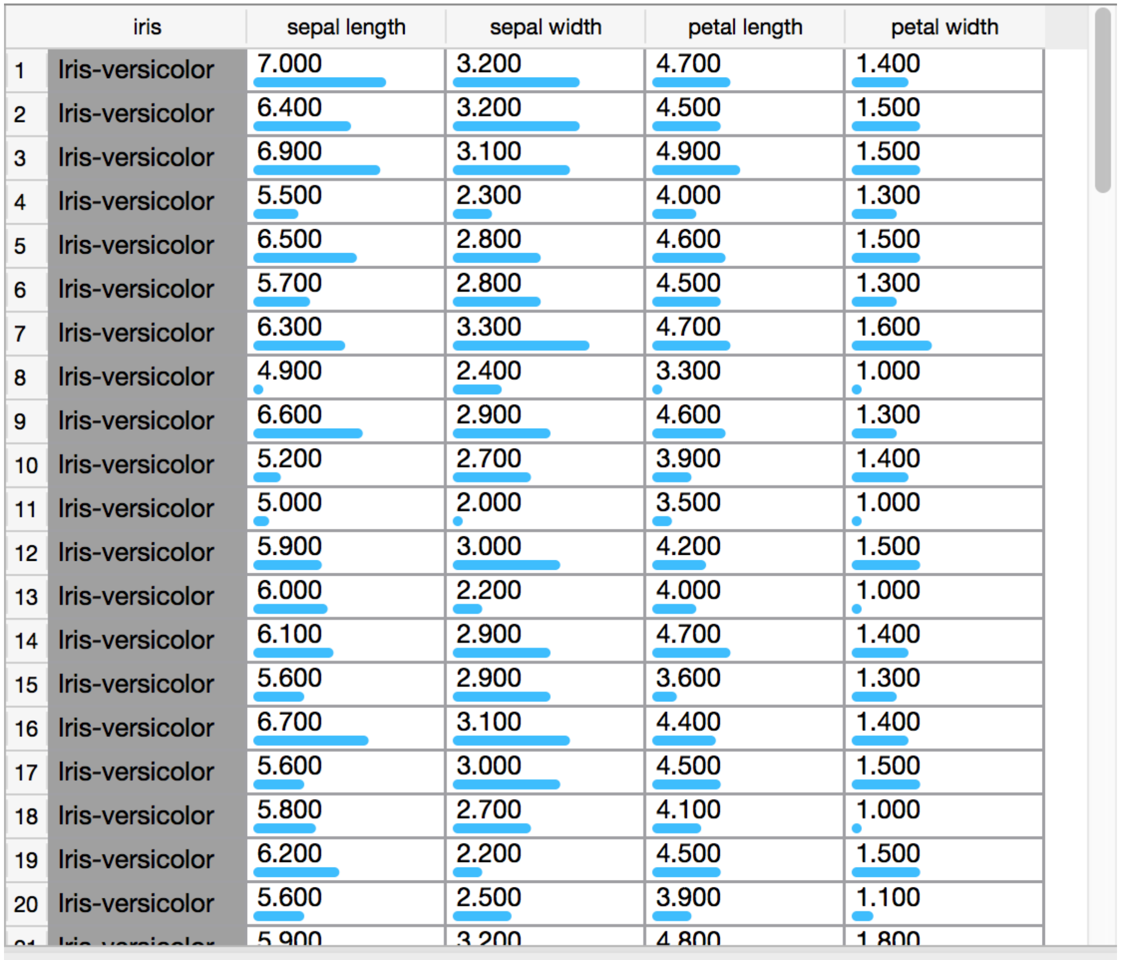
☑ Remove unused features

☑ Remove unused classes

☑ Send automatically

- Filters data according to simple rules
- For example: exclude all irises with short petals
- Select an attribute and a condition and press “Add” to add it to the filter

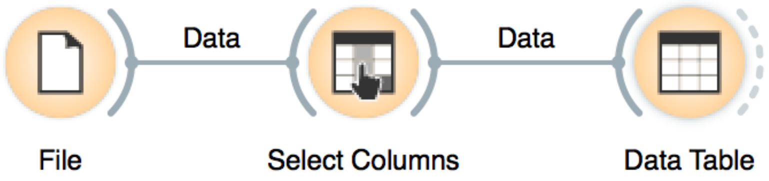
Data Selection Results

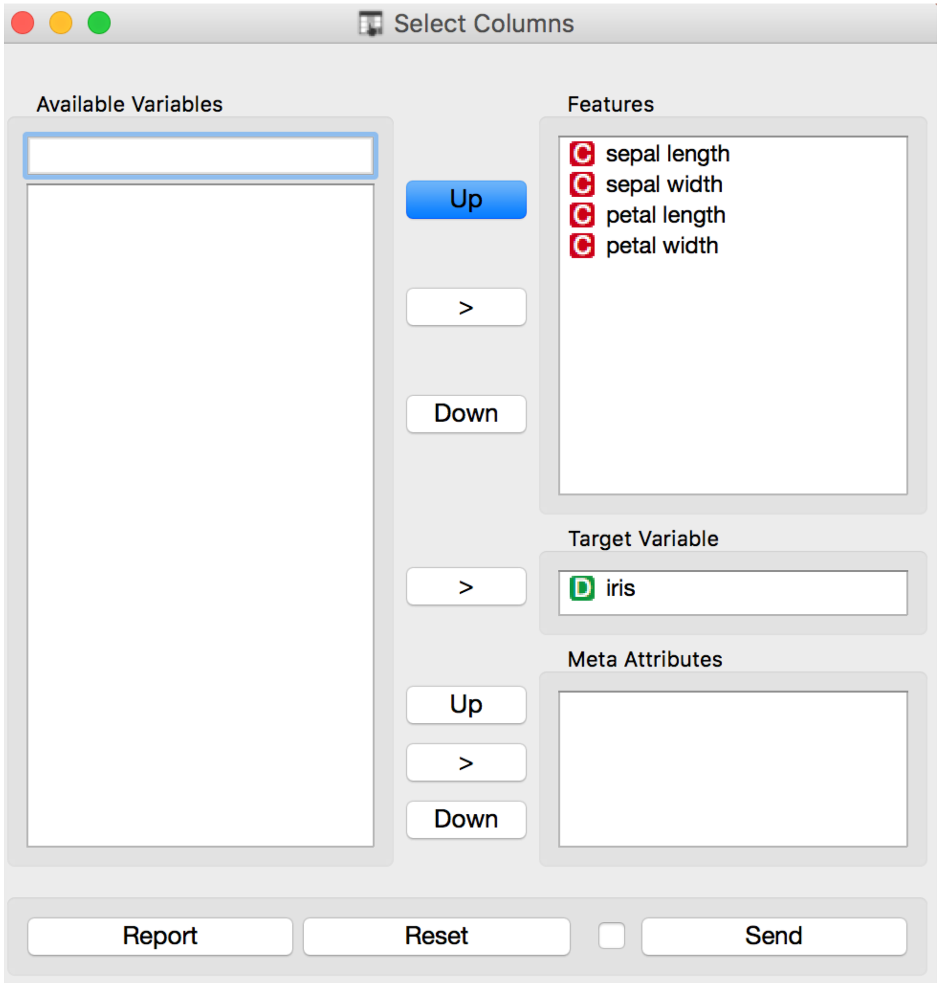


	iris	sepal length	sepal width	petal length	petal width
1	Iris-versicolor	7.000	3.200	4.700	1.400
2	Iris-versicolor	6.400	3.200	4.500	1.500
3	Iris-versicolor	6.900	3.100	4.900	1.500
4	Iris-versicolor	5.500	2.300	4.000	1.300
5	Iris-versicolor	6.500	2.800	4.600	1.500
6	Iris-versicolor	5.700	2.800	4.500	1.300
7	Iris-versicolor	6.300	3.300	4.700	1.600
8	Iris-versicolor	4.900	2.400	3.300	1.000
9	Iris-versicolor	6.600	2.900	4.600	1.300
10	Iris-versicolor	5.200	2.700	3.900	1.400
11	Iris-versicolor	5.000	2.000	3.500	1.000
12	Iris-versicolor	5.900	3.000	4.200	1.500
13	Iris-versicolor	6.000	2.200	4.000	1.000
14	Iris-versicolor	6.100	2.900	4.700	1.400
15	Iris-versicolor	5.600	2.900	3.600	1.300
16	Iris-versicolor	6.700	3.100	4.400	1.400
17	Iris-versicolor	5.600	3.000	4.500	1.500
18	Iris-versicolor	5.800	2.700	4.100	1.000
19	Iris-versicolor	6.200	2.200	4.500	1.500
20	Iris-versicolor	5.600	2.500	3.900	1.100
21	Iris-versicolor	5.900	3.200	4.800	1.800

- The “petal length” column now only contains values longer than 3 cm
- The blue category, iris-setosa, is now completely absent.
- Apparently all iris-setosa flowers have petals shorter than 3 cm.

Select Columns Widget (1)





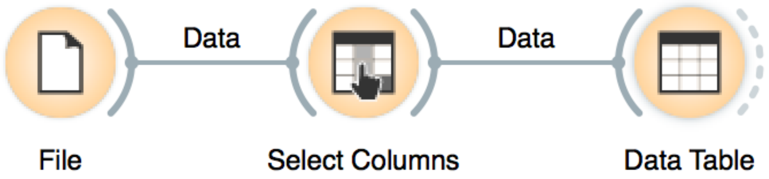
Features
- C sepal length
- C sepal width
- C petal length
- C petal width

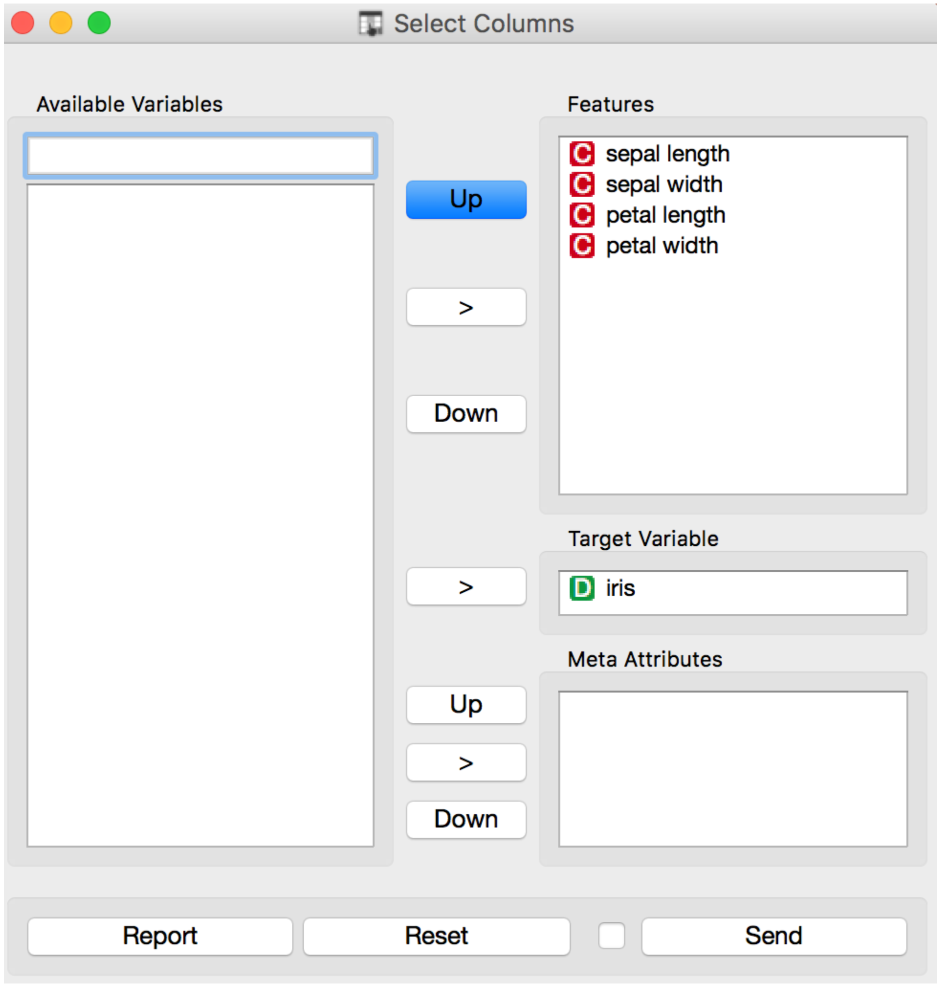
Target Variable
- D iris

Meta Attributes

- Choose which columns go in the dataset
  - "Attributes" are data values to be included in output
  - "Class" is the category of the row
  - "Meta Attributes" are descriptive attributes that are excluded from the analysis (such as a row ID)
  - "Available Attributes" are attributes available to be loaded, but ignored

Select Columns Widget (1)





Features

- C sepal length
- C sepal width
- C petal length
- C petal width

Target Variable

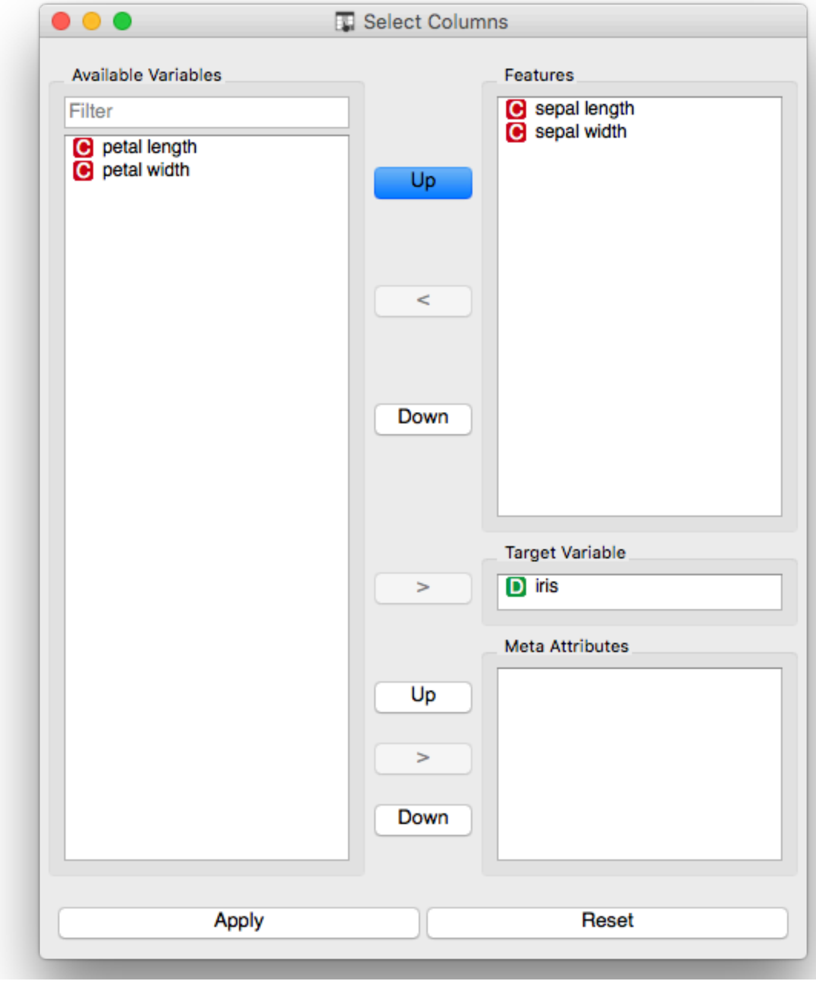
- D iris

Meta Attributes

- Drag or move variables between categories with the “>” and “<“ buttons
- Each variable is marked “C” for continuous (numerical values) or “D” for discrete (categorical values)
- You may need to click “Apply” before any changes you make take effect

Select Columns in action

- Suppose we were only interested in sepals, not petals.



Select Columns

Available Variables

Filter

- C petal length
- C petal width

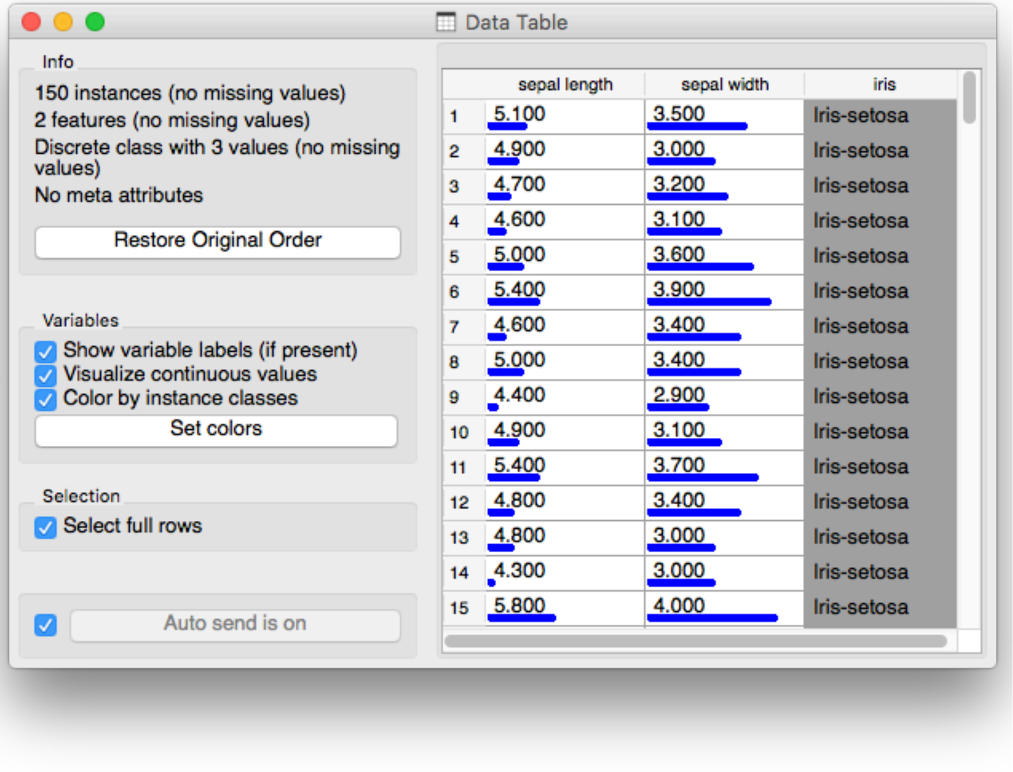
Features

- C sepal length
- C sepal width

Target Variable

- D iris

Meta Attributes



Data Table

Info

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

Variables

☑ Show variable labels (if present)
☑ Visualize continuous values
☑ Color by instance classes

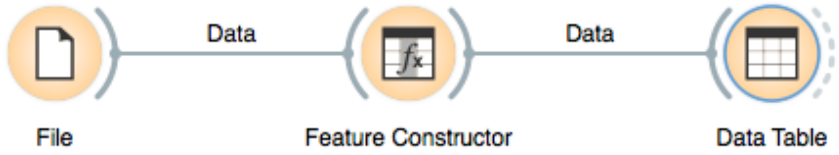
Selection

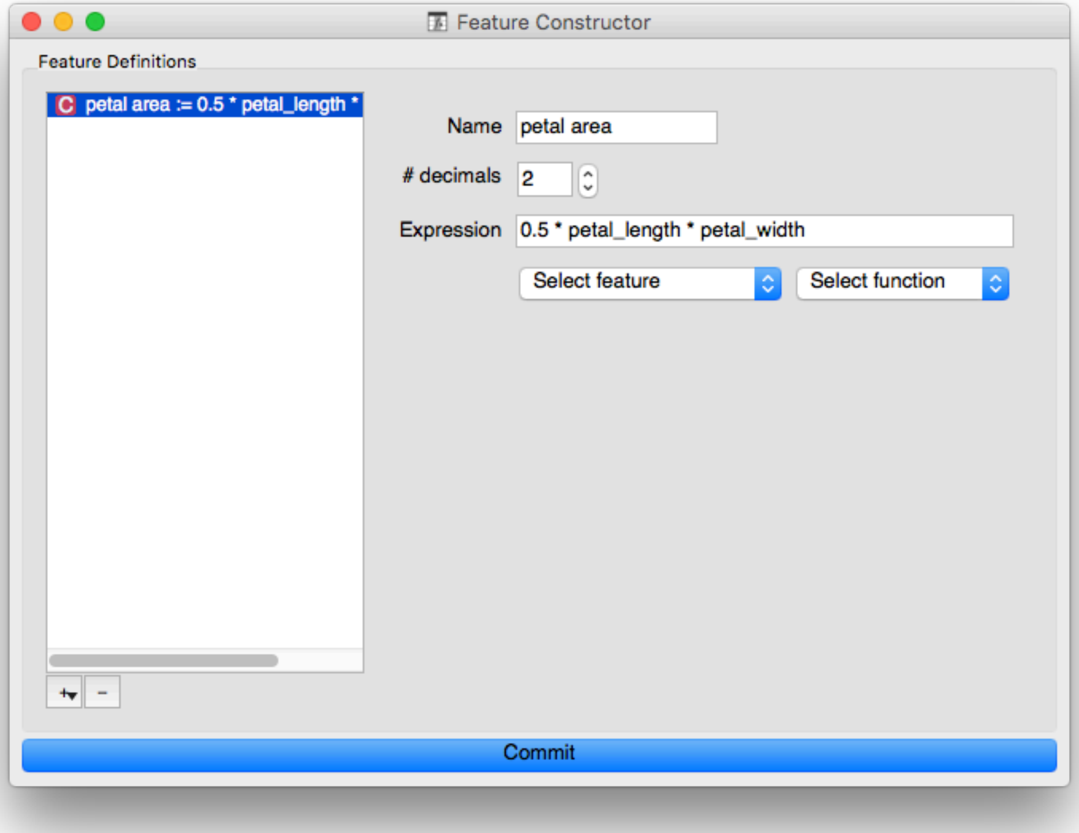
☑ Select full rows

☑ Auto send is on

	sepal length	sepal width	iris
1	5.100	3.500	Iris-setosa
2	4.900	3.000	Iris-setosa
3	4.700	3.200	Iris-setosa
4	4.600	3.100	Iris-setosa
5	5.000	3.600	Iris-setosa
6	5.400	3.900	Iris-setosa
7	4.600	3.400	Iris-setosa
8	5.000	3.400	Iris-setosa
9	4.400	2.900	Iris-setosa
10	4.900	3.100	Iris-setosa
11	5.400	3.700	Iris-setosa
12	4.800	3.400	Iris-setosa
13	4.800	3.000	Iris-setosa
14	4.300	3.000	Iris-setosa
15	5.800	4.000	Iris-setosa

Feature Constructor Widget





C petal area := 0.5 * petal_length *

Name: petal area

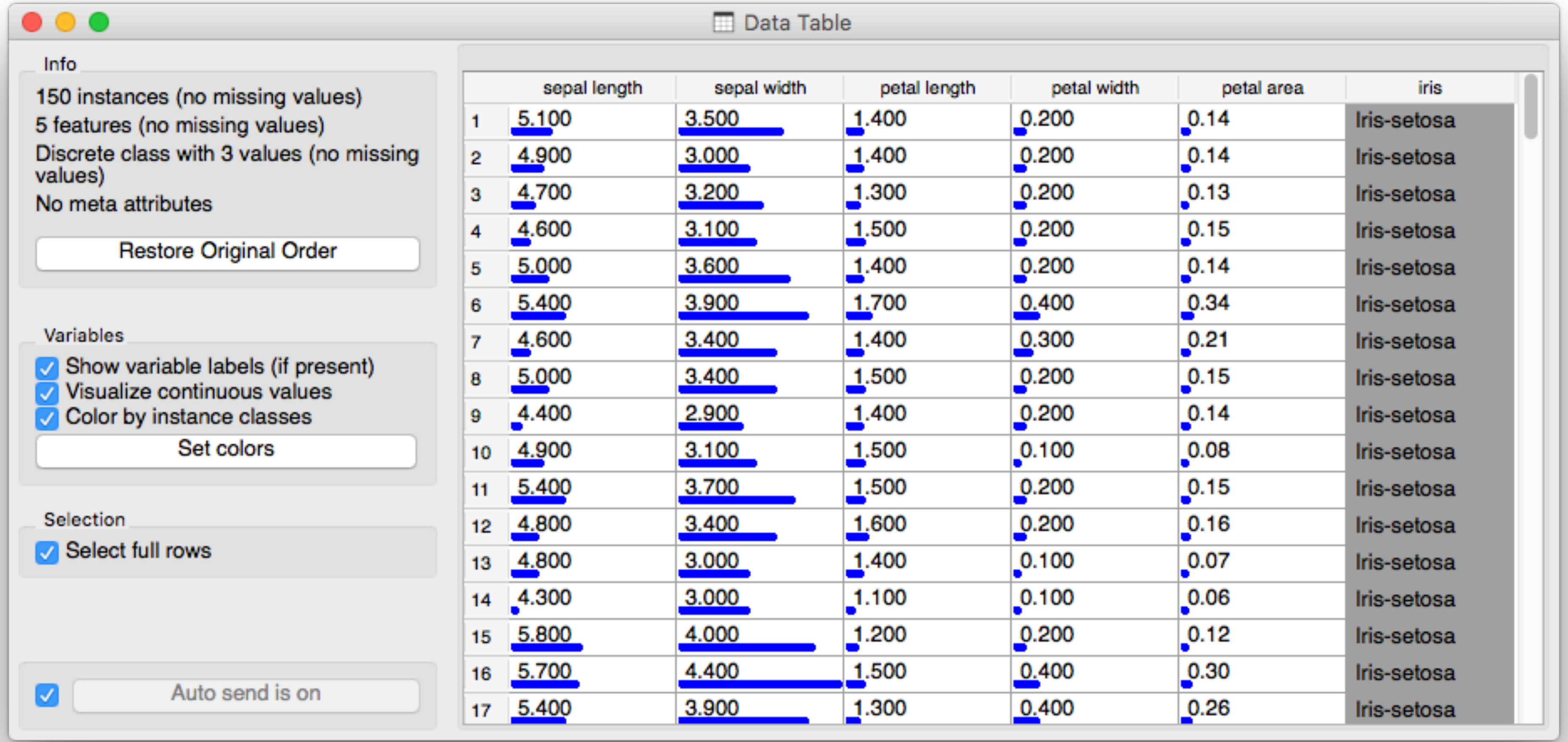
decimals: 2

Expression: 0.5 * petal_length * petal_width

- Defines new attributes (i.e. columns) based on the values of existing attributes
  - Type a formula and click “Add” to add a new feature
  - Select fields using “(all attributes)” and “(all functions)”
- Widget outputs the same data set with new attributes added
- This particular calculation is assuming petals are triangular

Feature Construction Results

- New attribute is added after existing attributes but before class



Info

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

Variables

☑ Show variable labels (if present)
☑ Visualize continuous values
☑ Color by instance classes

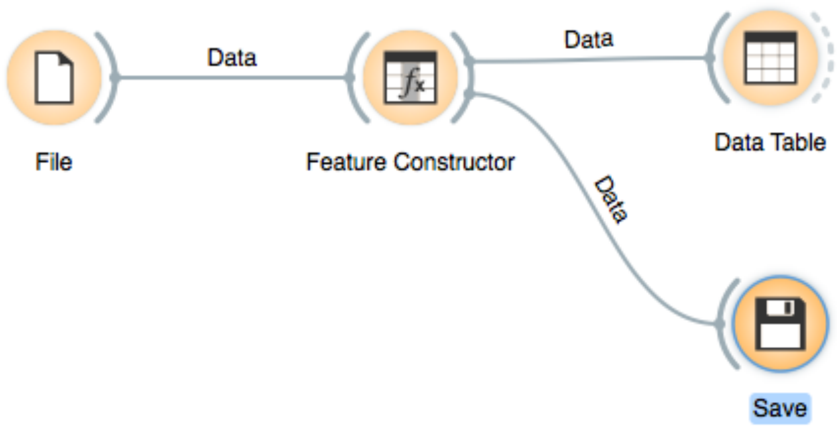
Selection

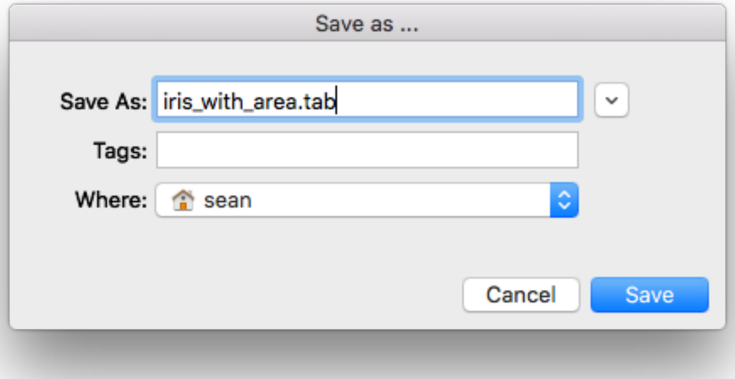
☑ Select full rows

☑ Auto send is on

	sepal length	sepal width	petal length	petal width	petal area	iris
1	5.100	3.500	1.400	0.200	0.14	Iris-setosa
2	4.900	3.000	1.400	0.200	0.14	Iris-setosa
3	4.700	3.200	1.300	0.200	0.13	Iris-setosa
4	4.600	3.100	1.500	0.200	0.15	Iris-setosa
5	5.000	3.600	1.400	0.200	0.14	Iris-setosa
6	5.400	3.900	1.700	0.400	0.34	Iris-setosa
7	4.600	3.400	1.400	0.300	0.21	Iris-setosa
8	5.000	3.400	1.500	0.200	0.15	Iris-setosa
9	4.400	2.900	1.400	0.200	0.14	Iris-setosa
10	4.900	3.100	1.500	0.100	0.08	Iris-setosa
11	5.400	3.700	1.500	0.200	0.15	Iris-setosa
12	4.800	3.400	1.600	0.200	0.16	Iris-setosa
13	4.800	3.000	1.400	0.100	0.07	Iris-setosa
14	4.300	3.000	1.100	0.100	0.06	Iris-setosa
15	5.800	4.000	1.200	0.200	0.12	Iris-setosa
16	5.700	4.400	1.500	0.400	0.30	Iris-setosa
17	5.400	3.900	1.300	0.400	0.26	Iris-setosa

Save Widget





Save As: iris_with_area.tab

Tags:

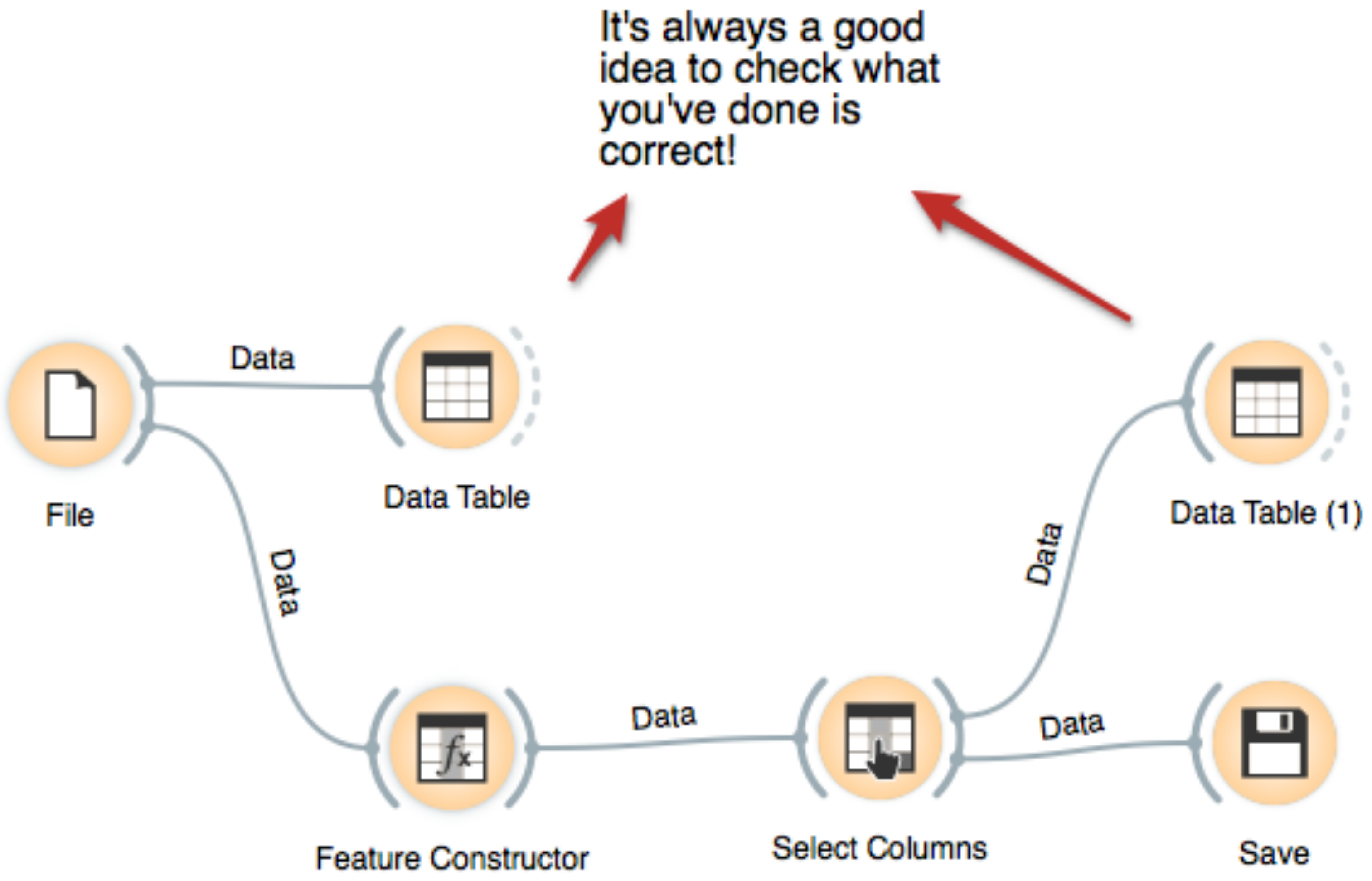
Where: sean

- Save a modified file
- Saves whatever is going to its input
  - If you made changes elsewhere in the scheme, they will not be saved
- Be careful not to accidentally overwrite your input file

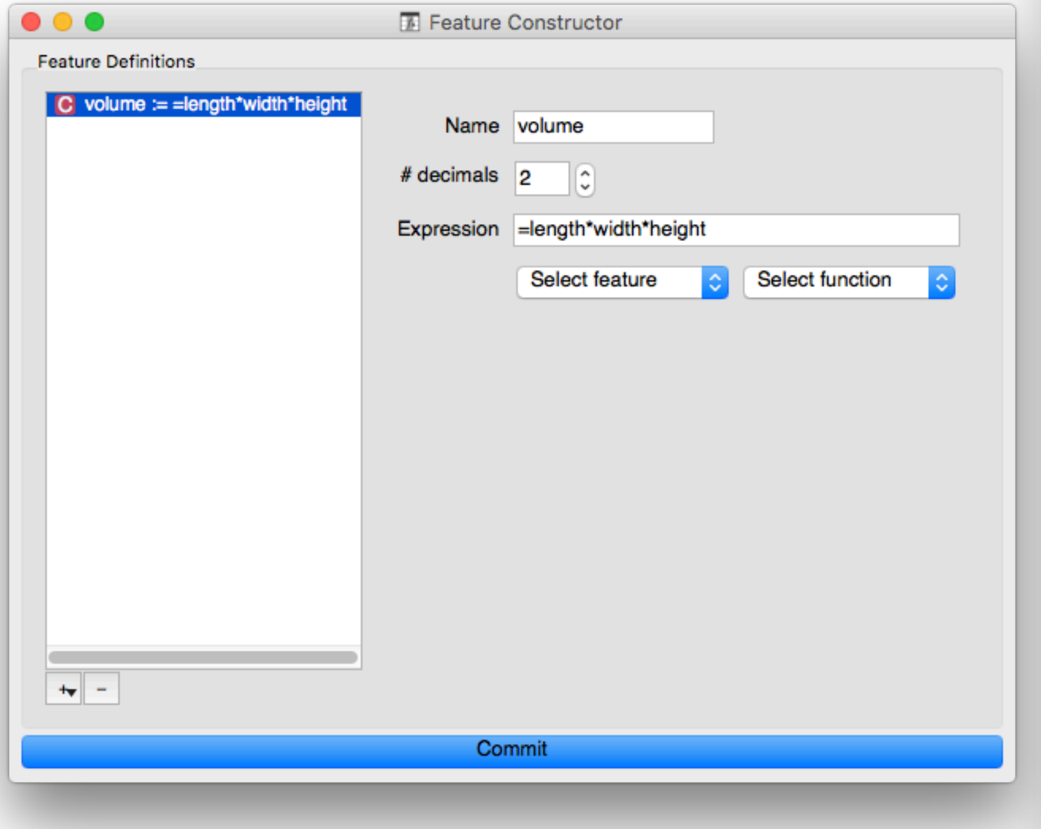
Exercise: First Scheme

- Load and inspect the *imports-85.tab* data file (on course website), which contains information about various imported cars
- Add a “volume” attribute (i.e. length x width x height)
- Remove the original length, width, and height attributes
- Save the dataset using a different filename

Solution


It's always a good idea to check what you've done is correct!
File
Data
Data Table
Data
Data Table (1)
Data
Feature Constructor
Data
Select Columns
Data
Save

Solution, continued

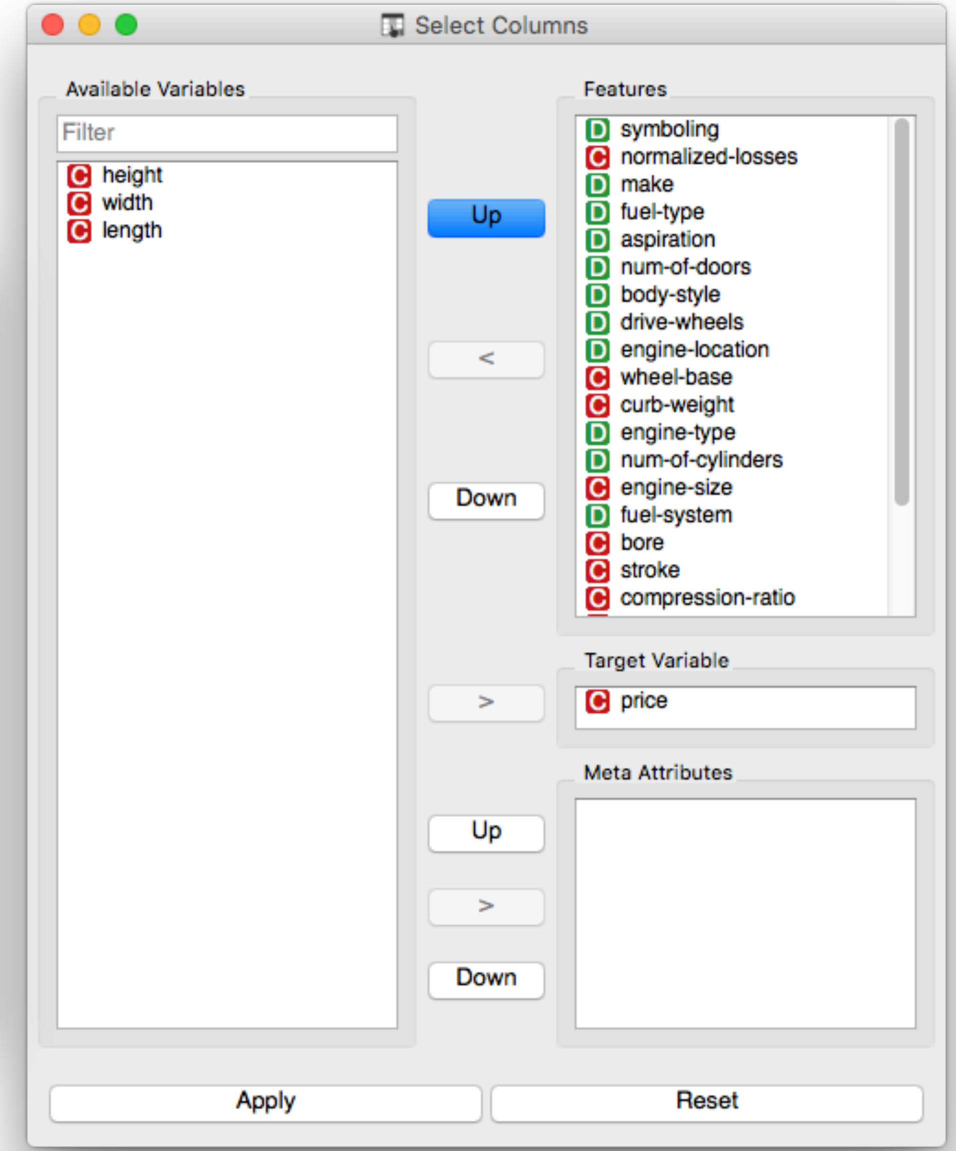


C volume := =length*width*height

Name: volume

decimals: 2

Expression: =length*width*height



Available Variables

- C height
- C width
- C length

Features

- D symboling
- C normalized-losses
- D make
- D fuel-type
- D aspiration
- D num-of-doors
- D body-style
- D drive-wheels
- D engine-location
- C wheel-base
- C curb-weight
- D engine-type
- D num-of-cylinders
- C engine-size
- D fuel-system
- C bore
- C stroke
- C compression-ratio

Target Variable

- C price

Meta Attributes

Remember to click “Apply” after you make changes!