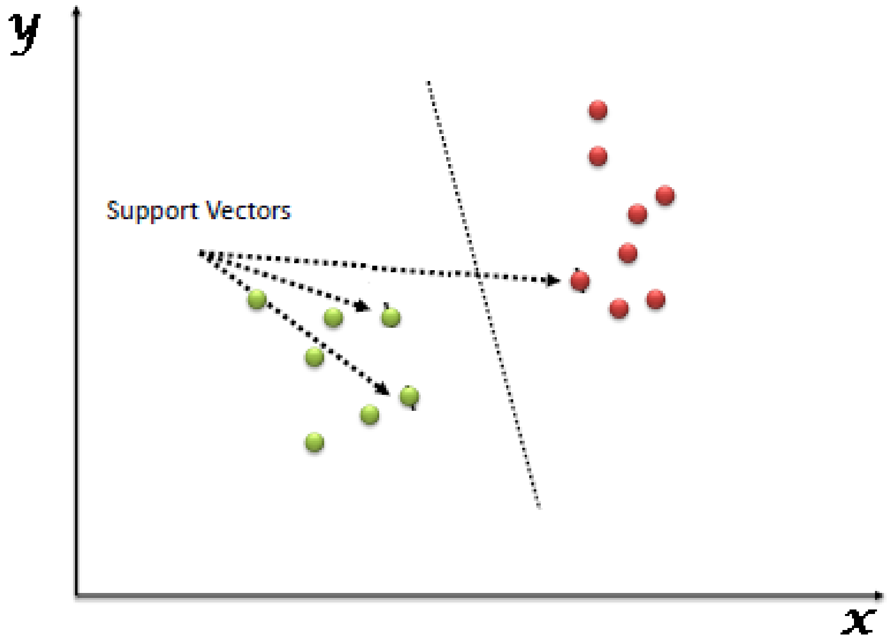
Orange Tool – Part II

Dr. Hemraj S L.

Understanding Support Vector Machine

- As an analogy, think of ‘***Regression***’ is capable of slicing and dicing data efficiently, but incapable of dealing with highly complex data. On the contrary, ‘Support Vector Machines’ works on smaller datasets, but on complex ones, it can be much stronger and powerful in building machine learning models.
- Support Vector Machine (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems.
- In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.



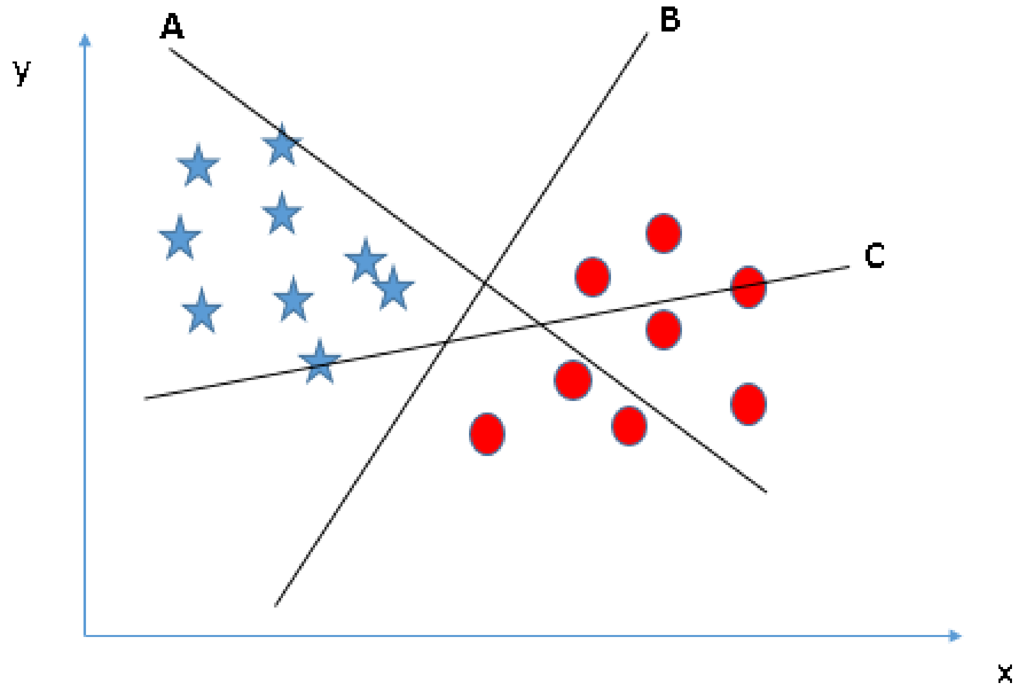
Support Vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).

How does SVM works?

- we got accustomed to the process of segregating the two classes with a hyper-plane. Now the question is "How can we identify the right hyper-plane?".

- **Identify the right hyper-plane (Scenario-1):** Here, we have three hyper-planes (A, B, and C). Now, identify the right hyper-plane to classify stars and circles.

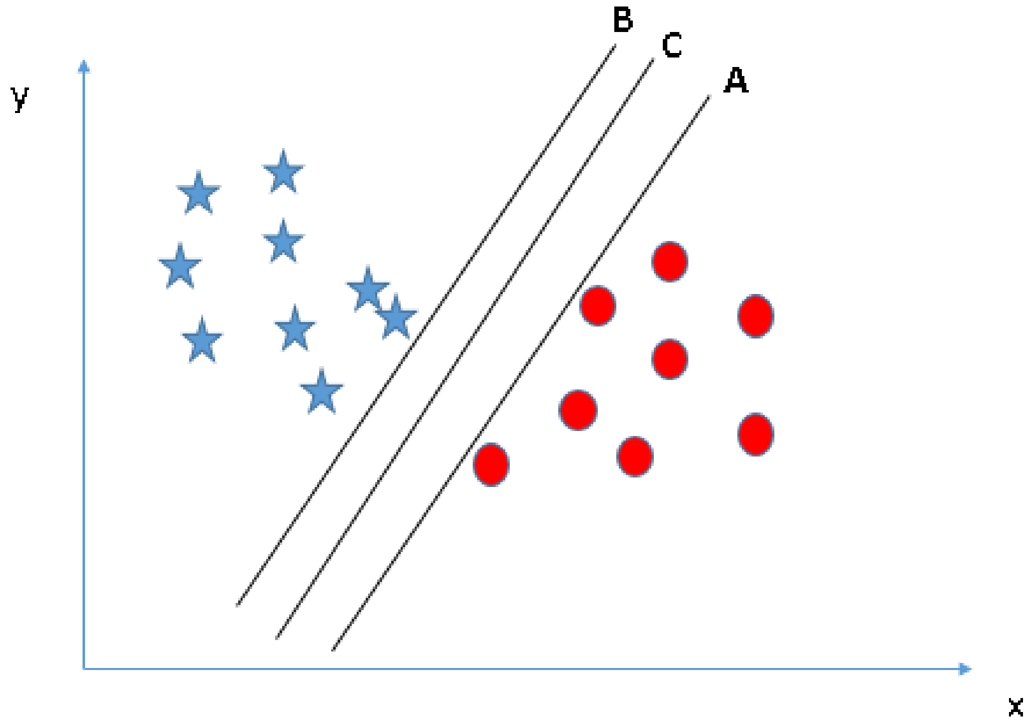
- Now, we have to select the hyper-plane which segregates the two classes better". In this scenario, hyper-plane "B" has excellently performed this job



How does SVM works?

- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B, and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

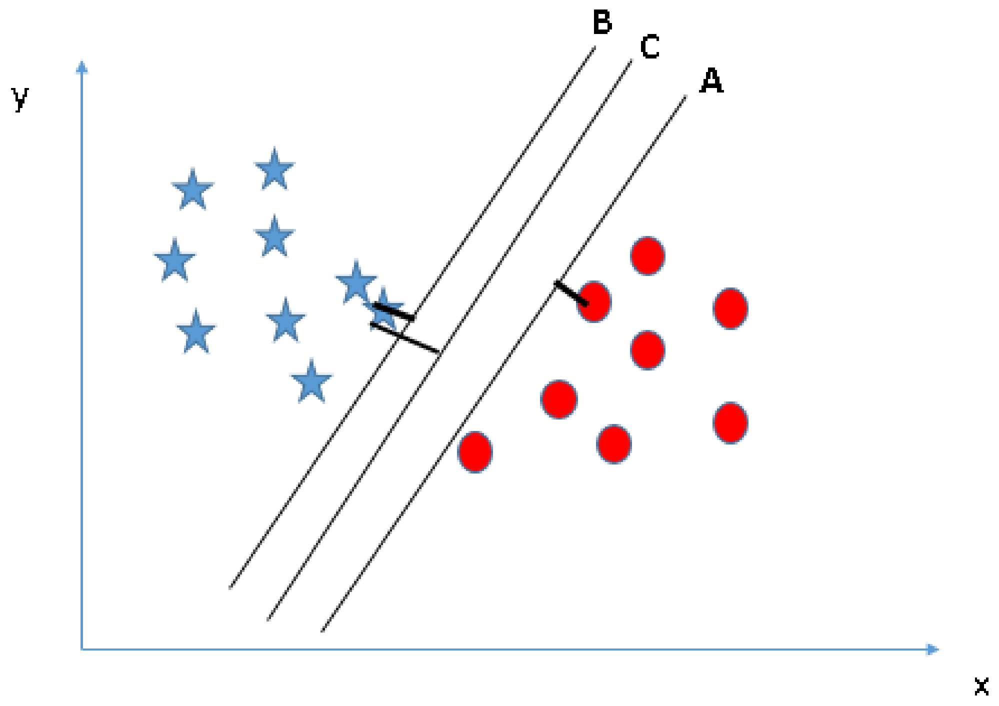
- Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin.



How does SVM works?

- **Identify the right hyper-plane (Scenario-2):** Here, we have three hyper-planes (A, B, and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

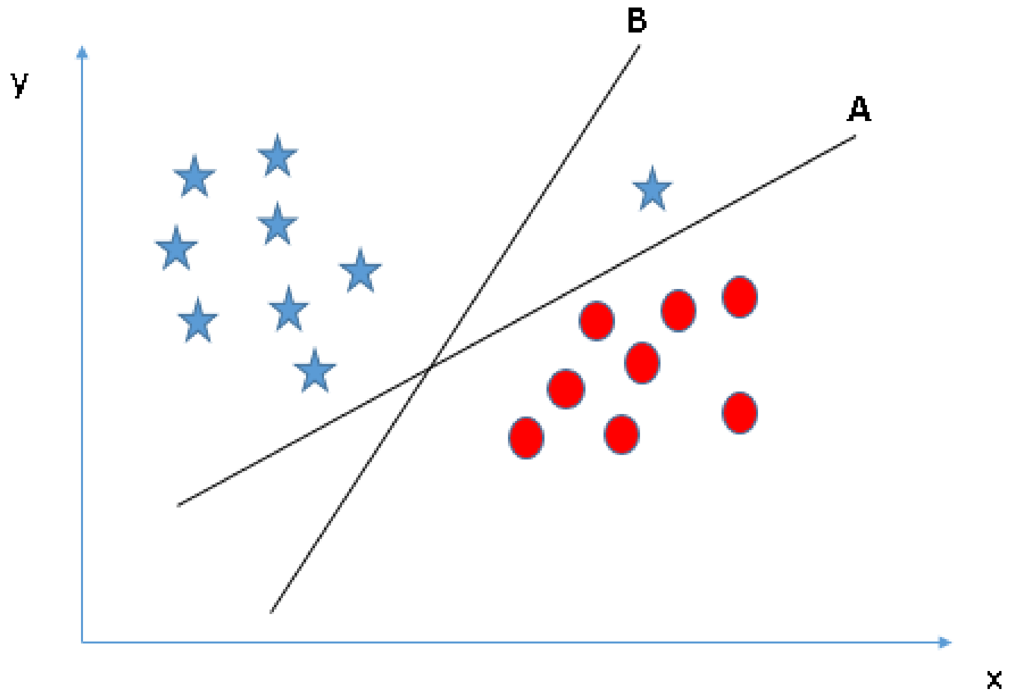
- Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as Margin.
- Here you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.



How does SVM works?

- **Identify the right hyper-plane (Scenario-3)**.

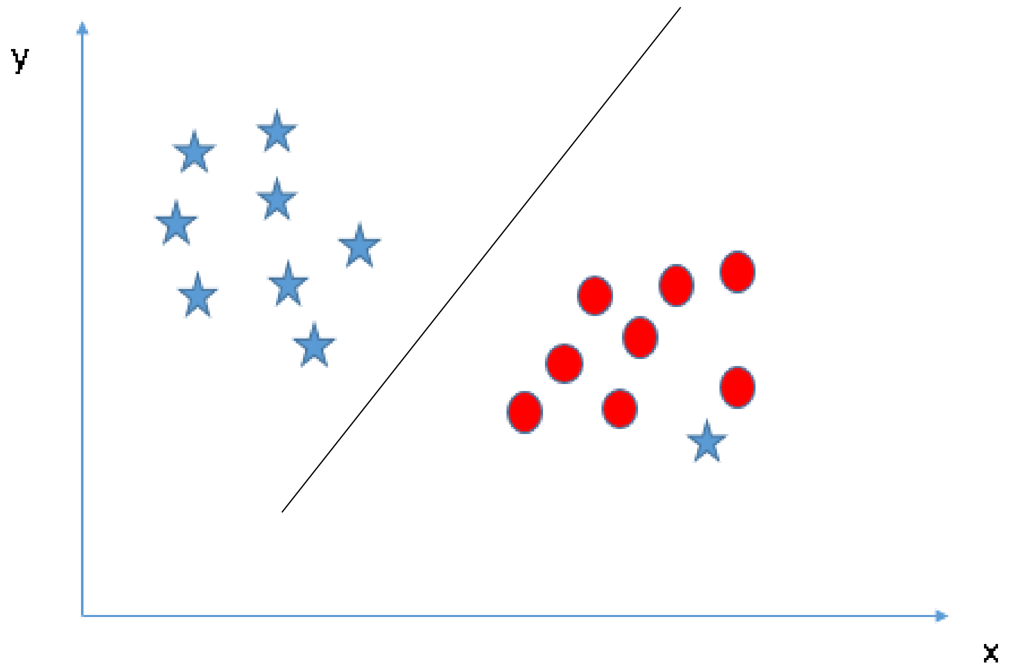
- Some of you may have selected the hyper-plane B as it has higher margin compared to A. But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is A.



How does SVM works?

- **Can we classify two classes (Scenario-4)?:** Below, I am unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.

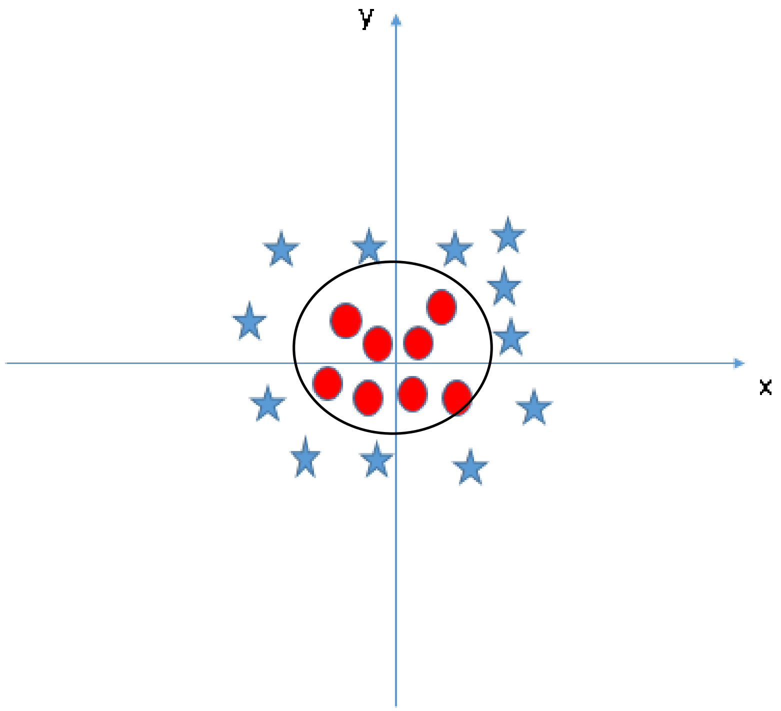
- The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.



How does SVM works?

- **Find the hyper-plane to segregate to classes (Scenario-5):** we can't have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.

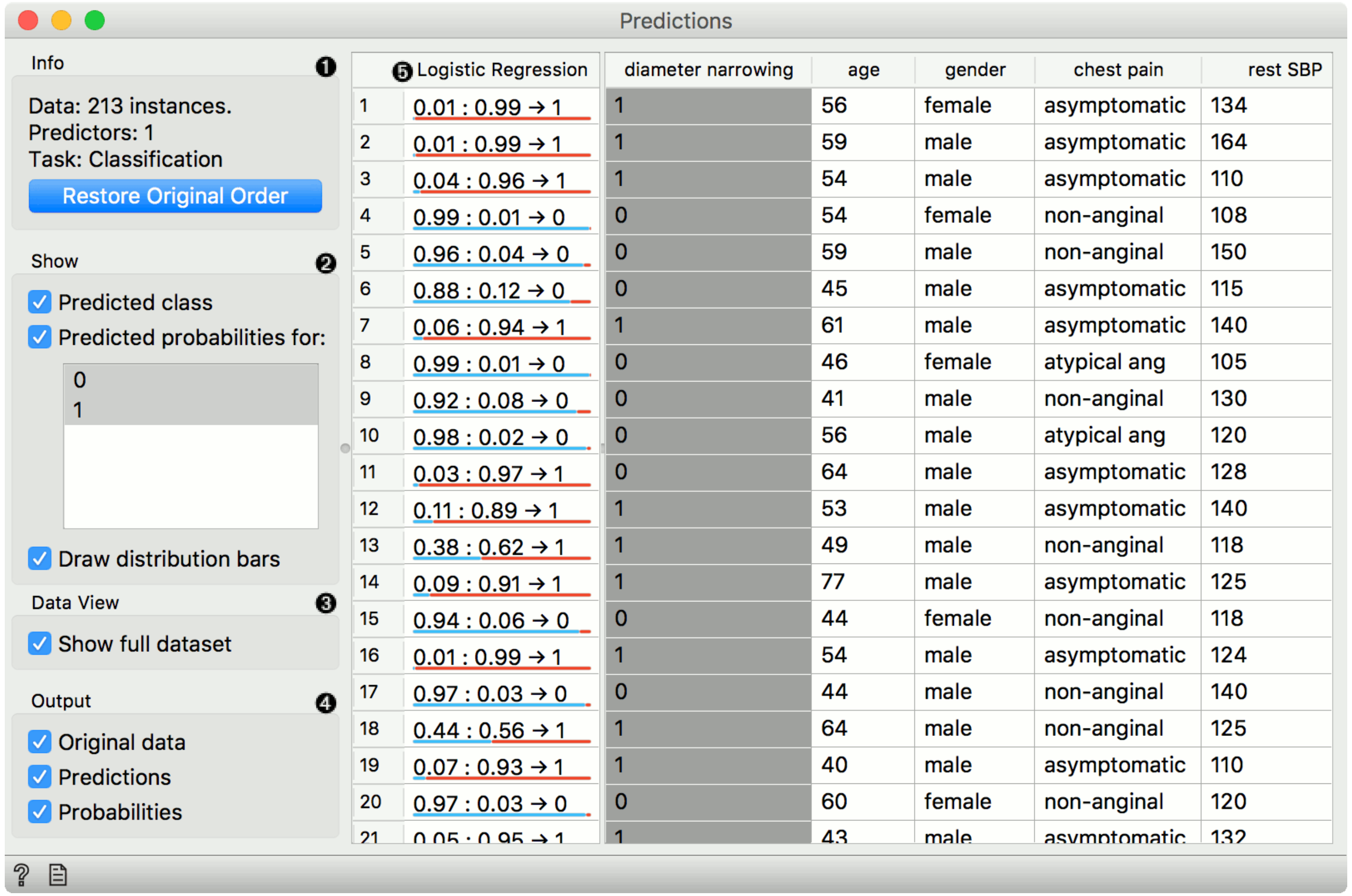
- the SVM algorithm has a technique called the ***kernel trick***. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts *not separable problem* into *separable problem*. It is mostly useful in non-linear separation problem.
- It does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you've defined.



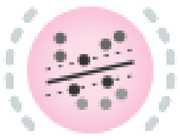
Orange: Predictions Widget

Predictions (1)

- The widget receives a dataset and one or more predictors (predictive models, not learning algorithms - see the example below). It outputs the data and the predictions.

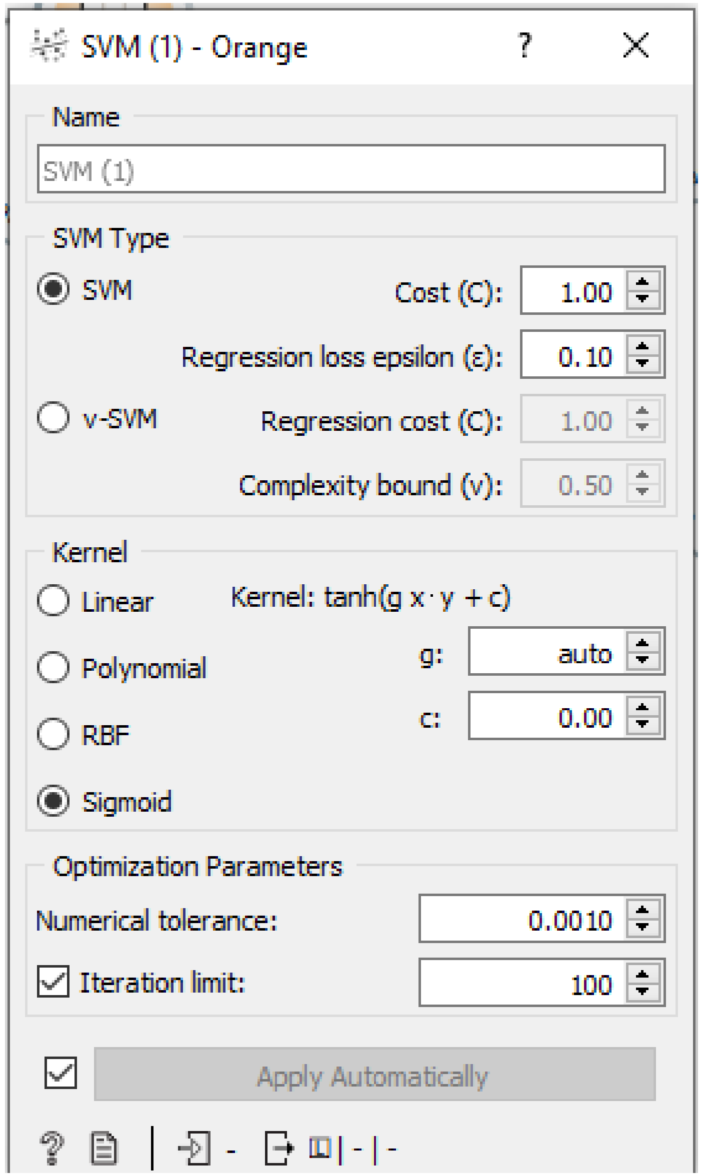


Orange: SVM Widget

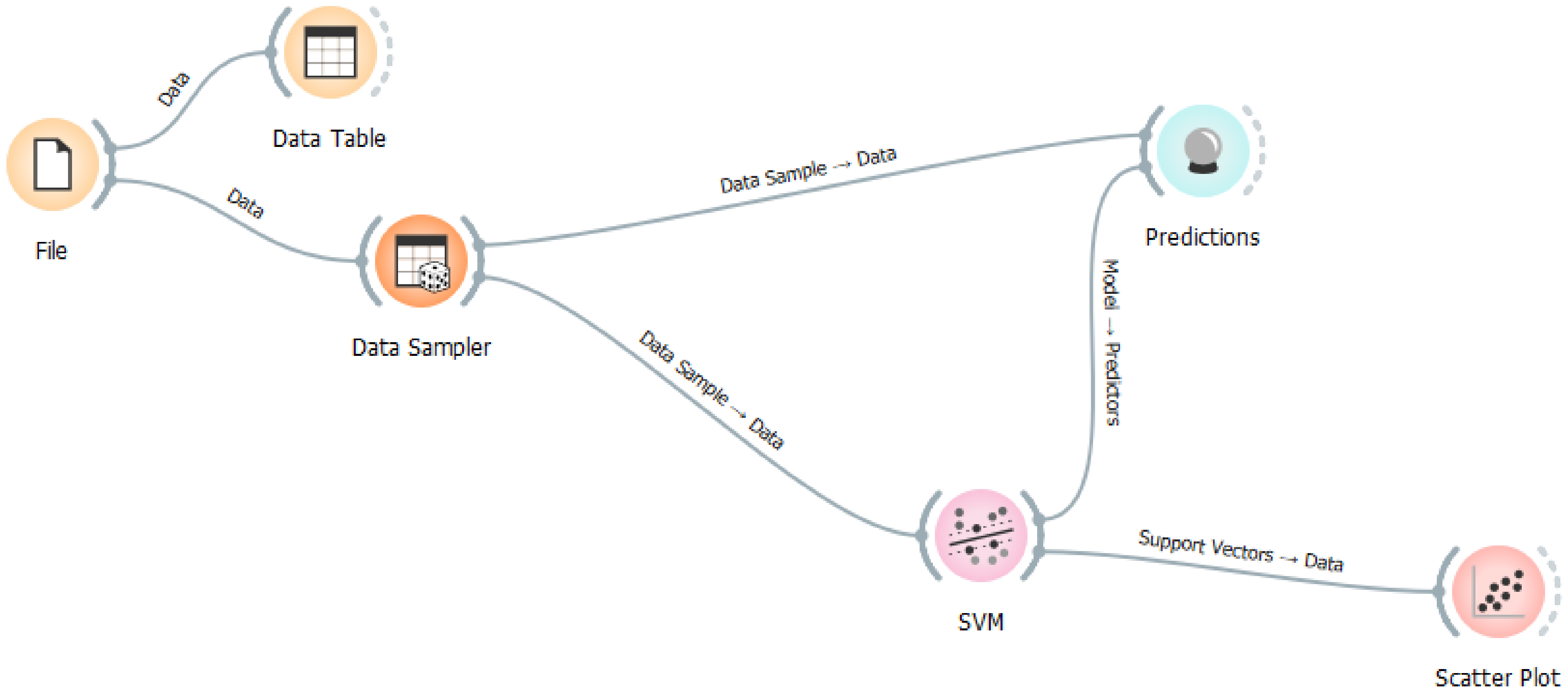


SVM (1)

- Support vector machine (SVM) is a machine learning technique that separates the attribute space with a hyperplane, thus maximizing the margin between the instances of different classes or class values. The technique often yields supreme predictive performance results. Orange embeds a popular implementation of SVM from the LIBSVM package. This widget is its graphical user interface.



Orange: SVM Widget

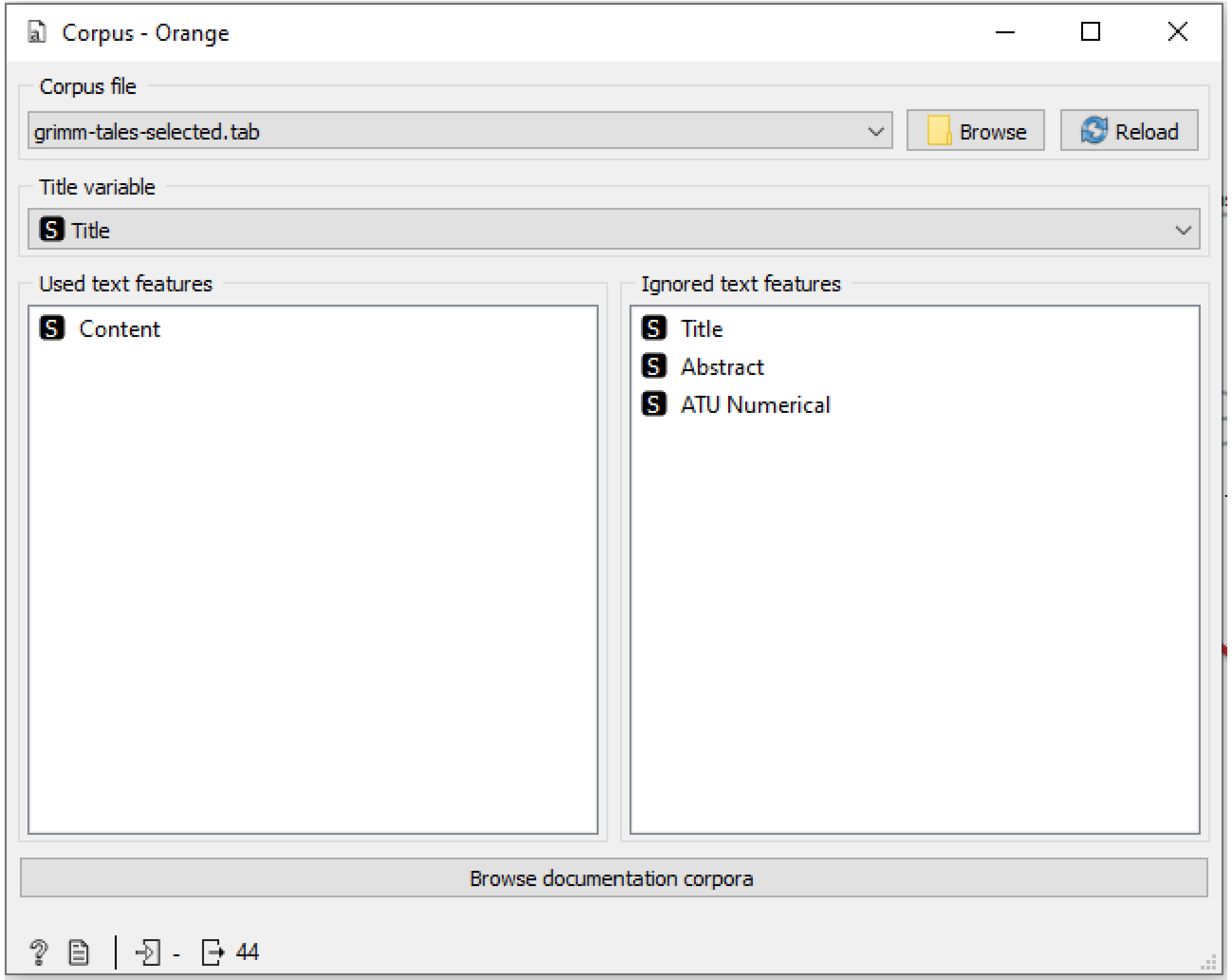


Here we have used HOUSING dataset to perform SVM.

Text Classification

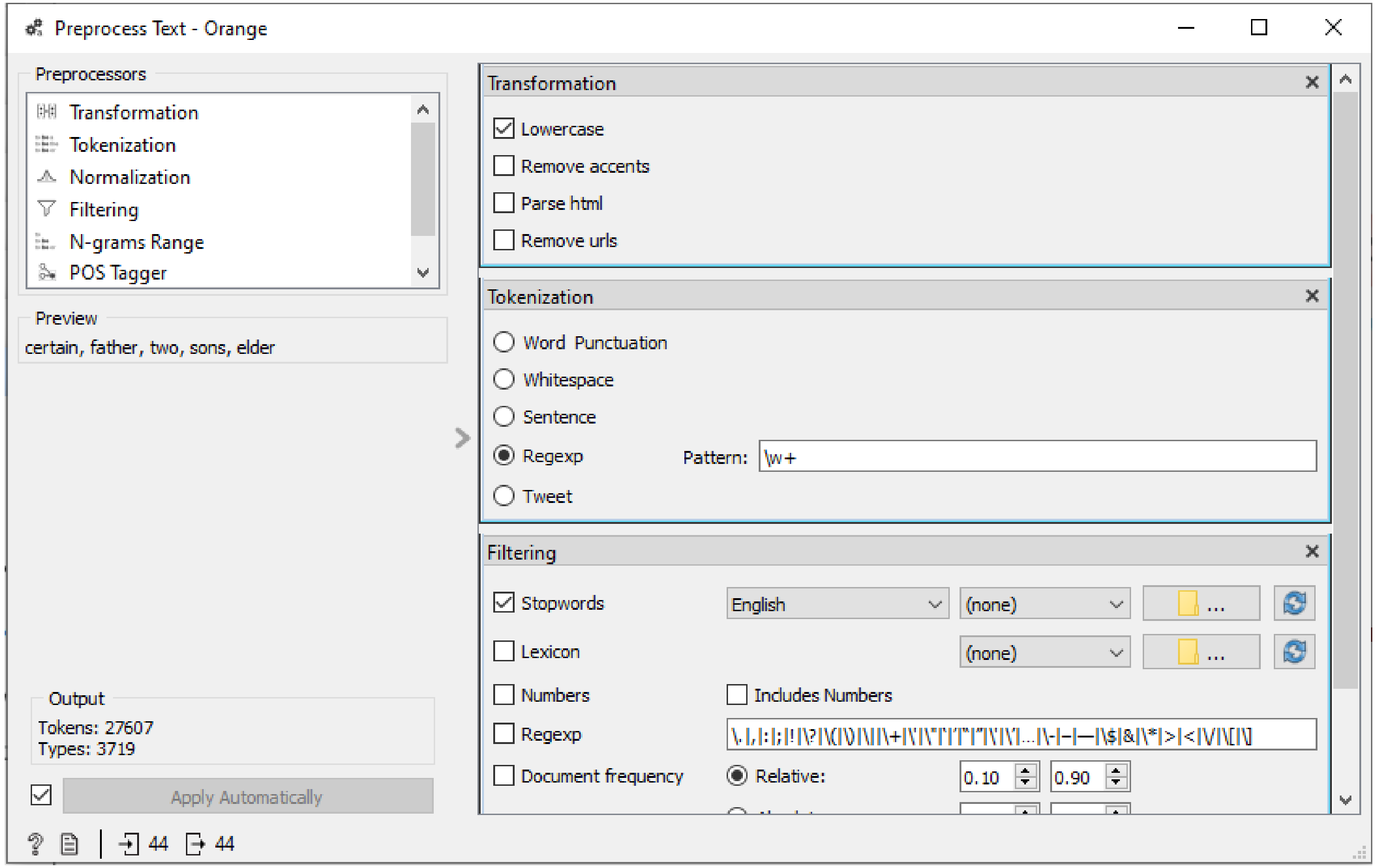
Corpus

- Load a corpus of text documents, tagged with categories, or change the data input signal to the corpus.
- It is a collection of documents
- It can work in two modes:
  - When no data on input, it reads text from files (using excel, csv, tab files)
  - When user provide input data.



Preprocess widget

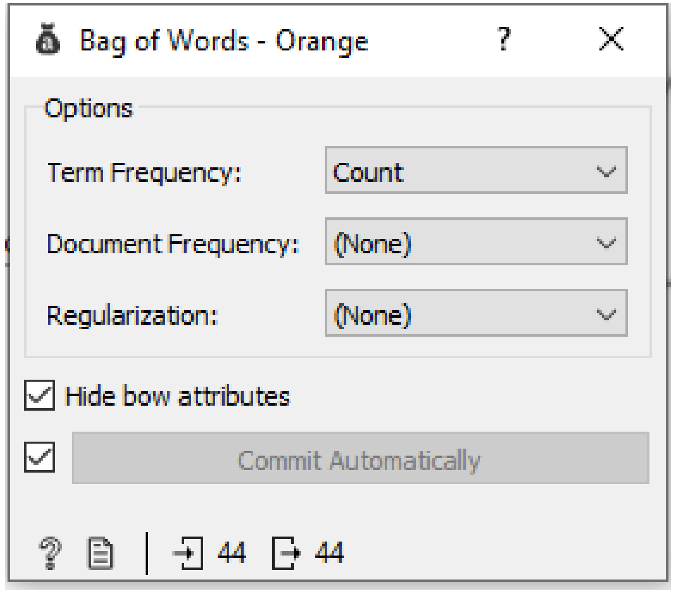
- Preprocesses data with selected methods.
- Preprocessing is crucial for achieving better-quality analysis results. The Preprocess widget offers several preprocessing methods that can be combined in a single preprocessing pipeline. Some methods are available as separate widgets, which offer advanced techniques and greater parameter tuning.



Bag of Words Widget

- Generates a bag of words from the input corpus.
- **INPUT:** Corpus - A collection of documents.
- **OUTPUT:** Corpus - Corpus with bag of words features appended.

- Bag of Words model creates a corpus with word counts for each data instance (document). The count can be either absolute, binary (contains or does not contain) or sublinear (logarithm of the term frequency). Bag of words model is required in combination with Word Enrichment and could be used for predictive modelling.



Naïve Bayes (Bayes’ Theorem) Widget

- A fast and simple probabilistic classifier based on Bayes' theorem with the assumption of feature independence.

  - **Inputs**
    - Data: input dataset
    - Preprocessor: preprocessing method(s)
  - **Outputs**
    - Learner: naive bayes learning algorithm
    - Model: trained model

- Naive Bayes learns a Naive Bayesian model from the data. It only works for classification tasks.

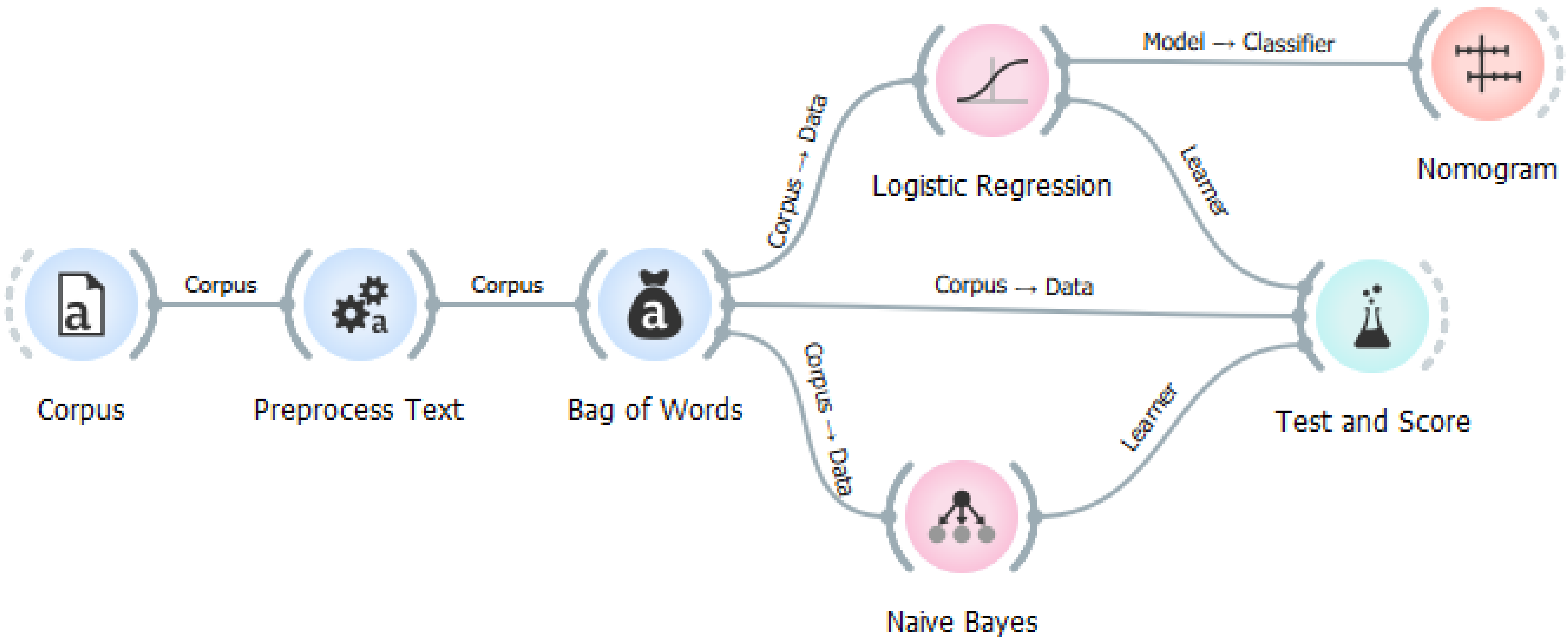
Nomogram Widget

- Nomograms for visualization of Naive Bayes and Logistic Regression classifiers.

- The Nomogram enables some classifier's (more precisely Naive Bayes classifier and Logistic Regression classifier) visual representation. It offers an insight into the structure of the training data and effects of the attributes on the class probabilities. Besides visualization of the classifier, the widget offers interactive support for prediction of class probabilities.

- The probability for the chosen target class is computed by '1-vs-all' principle, which should be taken in consideration when dealing with multiclass data

Text Classification



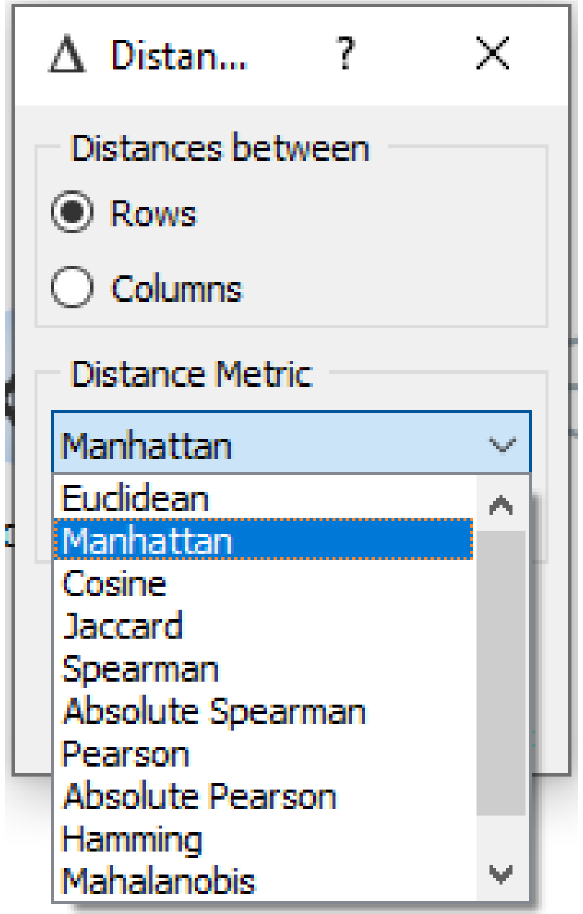
Clustering

Distances Widget

- Computes distances between rows/columns in a dataset.

- By default, the data will be normalized to ensure equal treatment of individual features. Normalization is always done column-wise.

- The resulting distance matrix can be fed further to Hierarchical Clustering for uncovering groups in the data



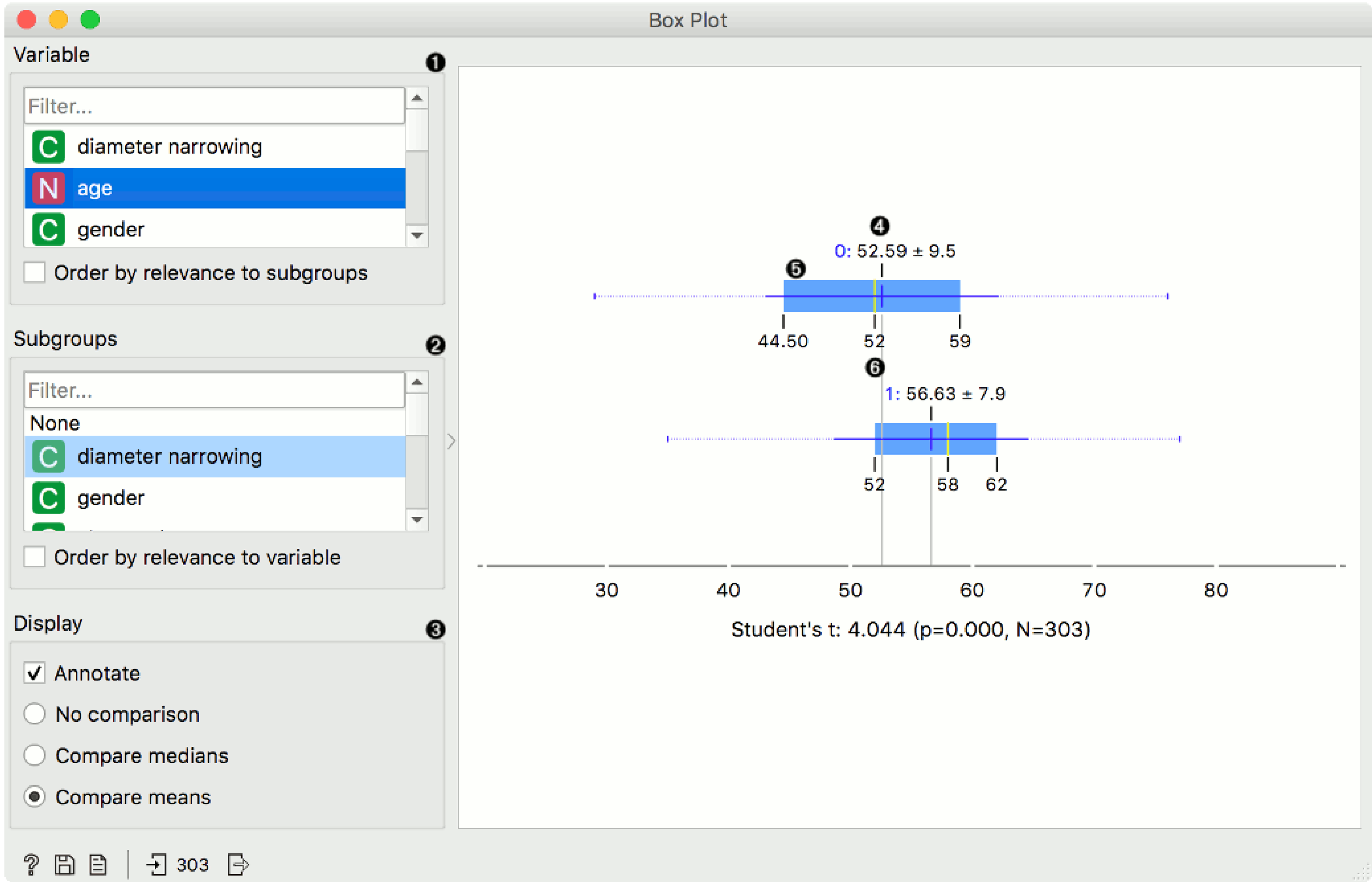
Hierarchical Clustering

- Groups items using a hierarchical clustering algorithm.

- The widget computes hierarchical clustering of arbitrary types of objects from a matrix of distances and shows a corresponding dendrogram. Distances can be computed with the Distances widget.
- The widget supports the following ways of measuring distances between clusters:
  - **Single linkage** computes the distance between the closest elements of the two clusters
  - **Average linkage** computes the average distance between elements of the two clusters
  - **Weighted linkage** uses the Weighted Pair Group Method with Arithmetic Mean is a simple agglomerative (bottom-up) hierarchical clustering method.
  - **Complete linkage** computes the distance between the clusters' most distant elements
  - **Ward linkage** computes the increase of the error sum of squares

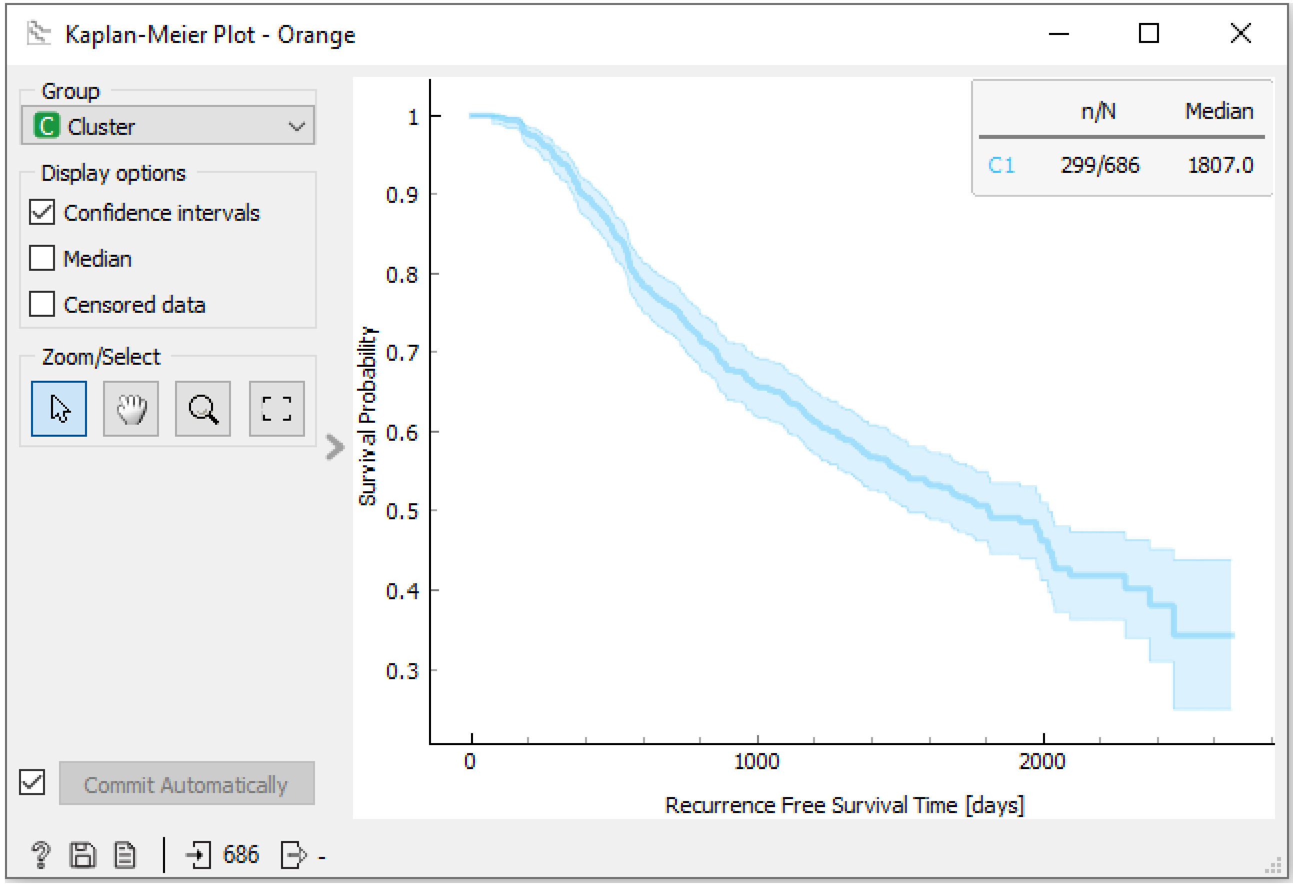
Box Plot Widget

- Shows distribution of attribute values.
- It is a good practice to check any new data with this widget to quickly discover any anomalies, such as duplicated values (e.g., gray and grey), outliers, and alike.

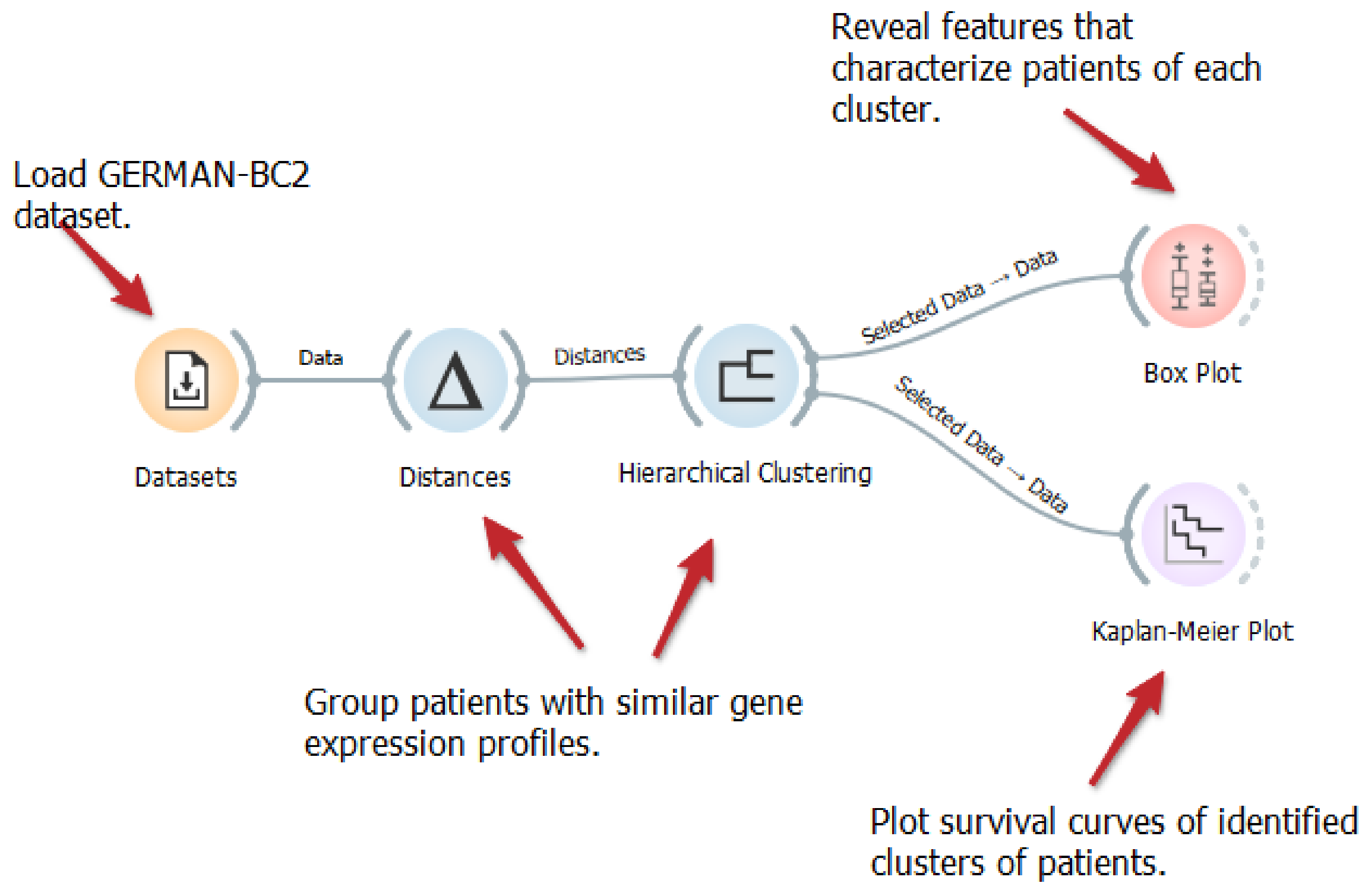


Kaplan-Meier Widget

- Visualisation of Kaplan-Meier estimator.
- Kaplan-Meier Plot is a visual representation of the estimated survival function that shows the probability of an event at a respective time interval.
- It works on survival datasets, which by definition include time and event observations (Example: Cancer cell).
- The plot allows visualization of the survival functions of different groups based on feature values.



Hierarchical Clustering



Tweets Prediction using Cross Validation