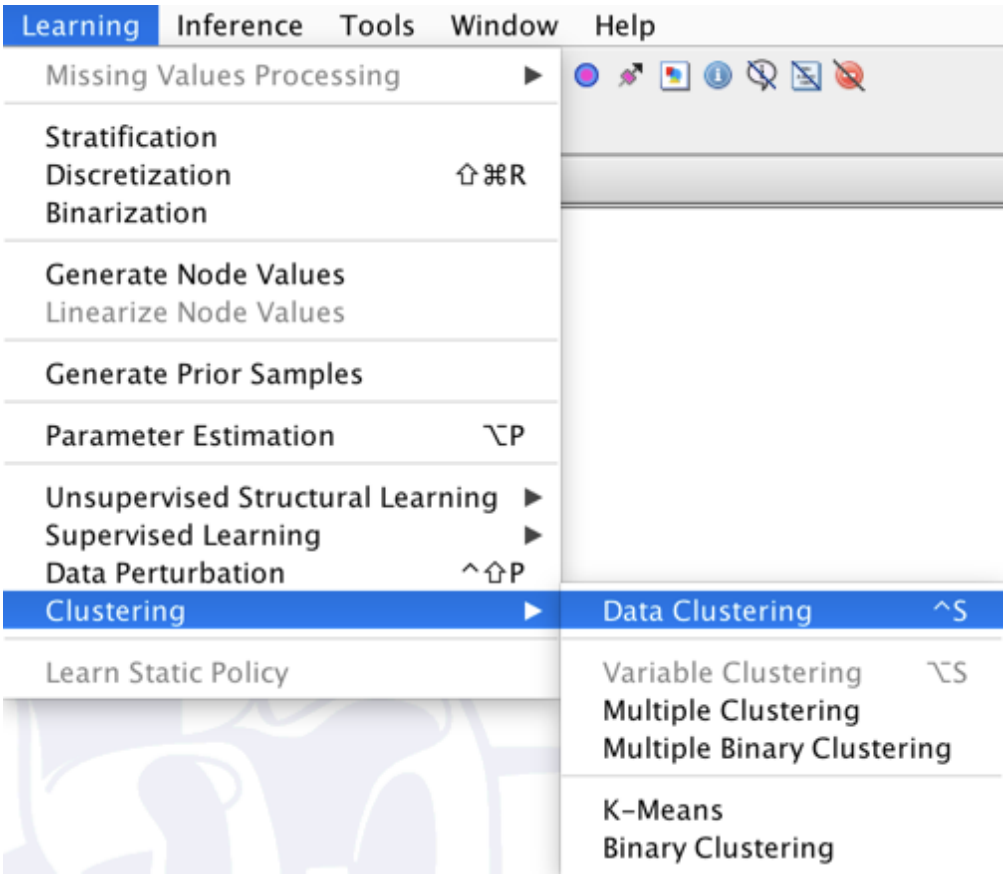
Data Clustering (7.0)

Context

Learning | Clustering | Data Clustering



Data Clustering is an unsupervised learning algorithm that creates a new variable, ***[Factor_i]***. The states of this variable represent the induced clusters.

Data Clustering can be used for different purposes:

1. For finding observations that look the same;
2. For finding observations that behave the same;
3. For representing an unobserved dimension;
4. For summarizing a set of variables;
5. For compactly representing the joint probability distribution.

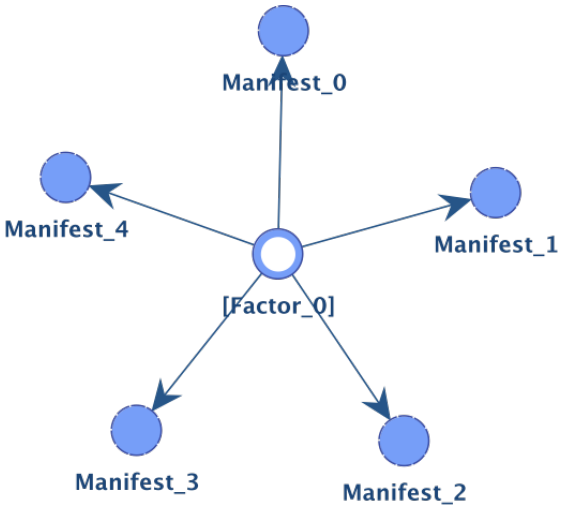
From a technical perspective, the segments should be:

1. Homogeneous/pure;
2. Have clear differences with the other segments;
3. Stable.

From a functional perspective, the segments should be:

1. Easy to understand;
2. Operational;
3. A fair representation of the data.

Data Clustering with Bayesian networks is typically based on a **Naive** structure in which the newly created variable ***[Factor_i]*** is the parent of the variables that are used for clustering (usually called the **Manifest** variables).



This variable being hidden, i.e. with 100% of missing values, the marginal probability distribution of ***[Factor_i]*** and the conditional probability distributions of the **Manifest** variables are initialized with random distributions. Thus, an **Expectation-Maximization (EM)** algorithm is used to fit these distributions with the data:

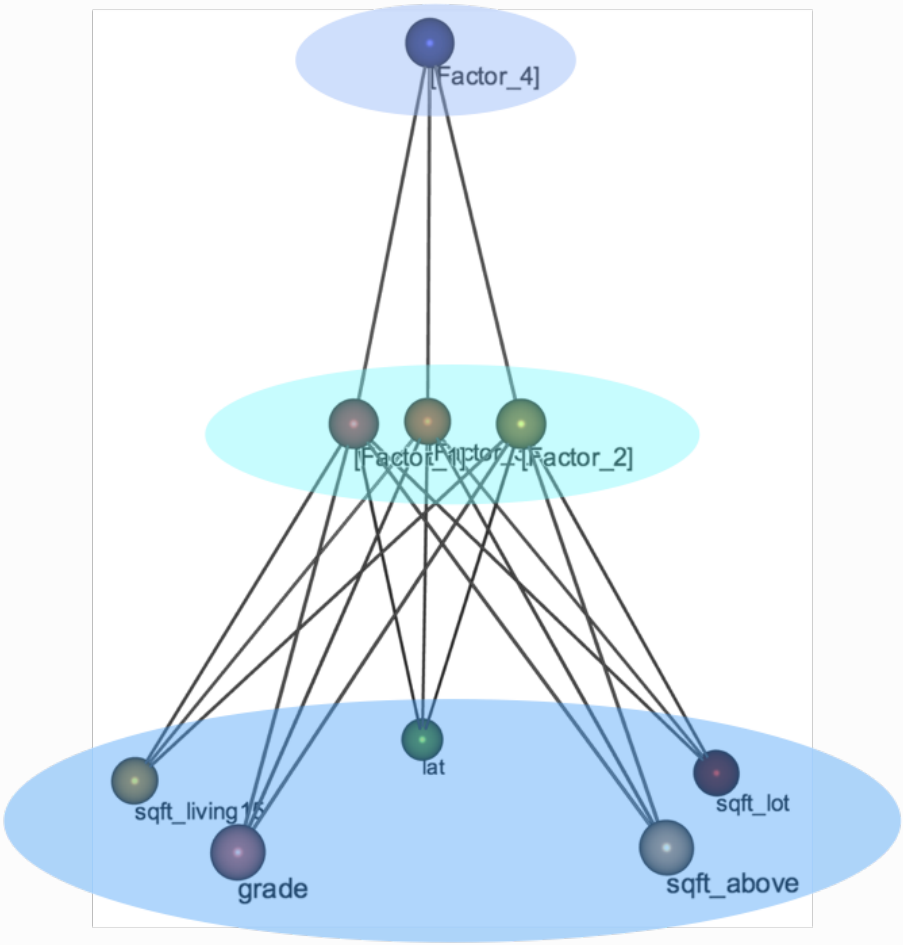
1. **Expectation**: the network is used with its current distributions for computing the posterior probabilities of ***[Factor_i]***, for the entire set of observations described in the data set; These probabilities are used for soft imputing ***[Factor_i];***
2. **Maximization**: based on these imputations, the distributions of the network are updated via **Maximum-Likelihood.** The algorithm goes back to **Ex pectation,** until no significant changes occur to the distributions.

History

Data Clustering has been updated in versions 5.1 and 5.2.

New Feature: Meta-Clustering

This new feature has been added for improving the stability of the induced solution (3rd technical quality). It consists of creating a dataset made of a subset of the **Factors** that have been created while searching for the best segmentation, and using **Data Clustering** on these new variables. The final solution is thus a summary of the best solutions that have been found (4th purpose).



The five **Manifest** variables (bottom of the graph) are used in the dataset for describing the observations.

The **Factor variables** ***[Factor_1], [Factor_2],*** and ***[Factor_3]*** have been induced with **Data Clustering.** They are then imputed to create a new dataset.

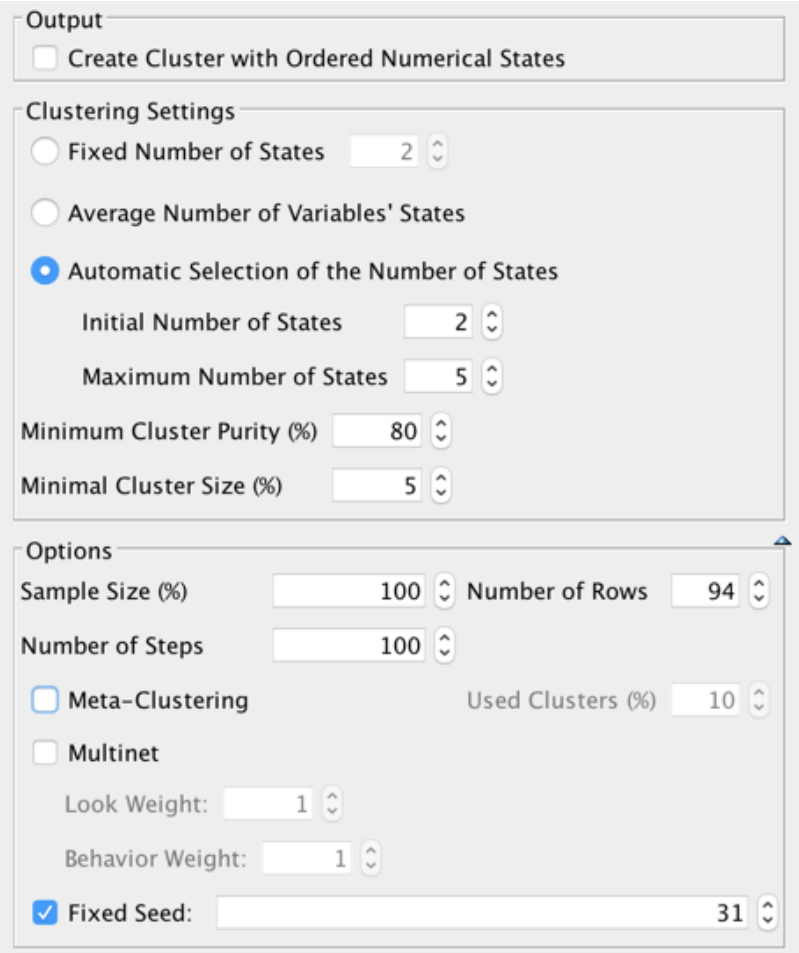
In this example, three **Factor variables** are used for creating the final solution ***[Factor_4].***

Example

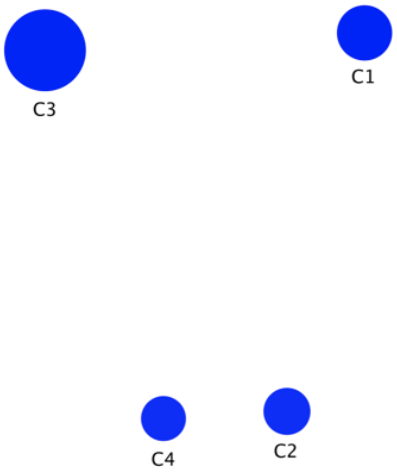
Let's use a dataset that contains house sale prices for King County, which includes the city of Seattle, Washington. It describes homes sold between May 2014 and May 2015. More specifically, we have extracted 94 houses that are more than 100 years old, that have been renovated, and come with a basement. For simplicity, we are just describing the houses with the 5 **Manifest variables** below, discretized into 2 bins.

- *grade*: Overall grade given to the housing unit
- *sqrt_above*: Square footage of house, apart from basement
- *sqft_living15*: Living room area in 2015
- *sqft_lot*: Square footage of the lot
- *lat*: Latitude coordinate

The wizard below shows the settings used for segmenting these houses:



After 100 steps, segmenting the houses into 4 groups is the best solution. Below, the mapping (**Analysis | Report | Target | Relationship with Target Node | Mapping**) shows the created states/segments:

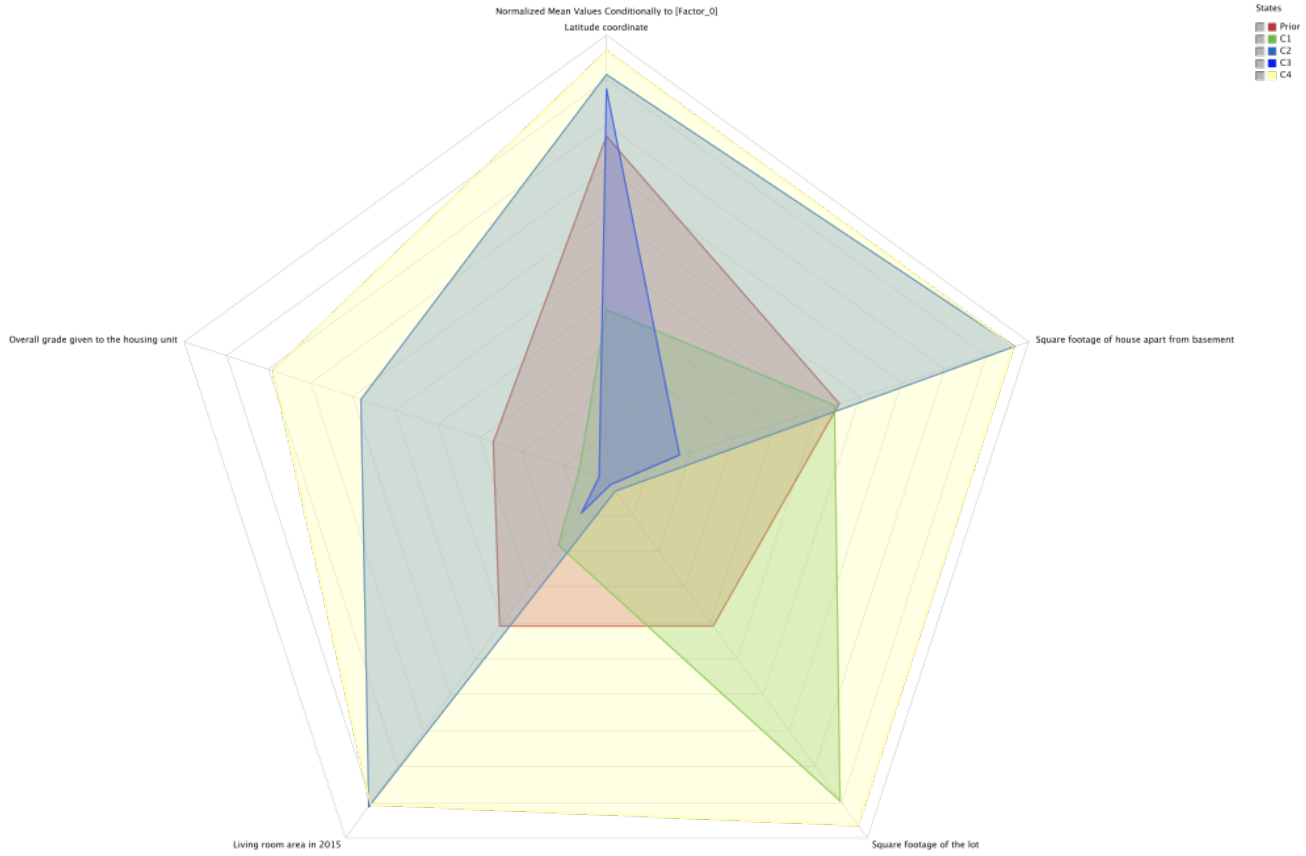


- the size of each segment is proportional to its marginal probability (i.e. how many houses belong to each segment),
- the intensity of the blue is proportional to the purity of the associated cluster (1st technical quality), and
- the layout reflects the neighborhood.

Marginal Probabilities	
C3	38.636%
C1	25.505%
C2	18.434%
C4	17.424%

Target Mean Purity: 92.216%		
State	Purity	Neighborhood
C1	93.817%	C4 3.261% C3 2.656%
C2	91.937%	C3 3.888% C4 3.319% C1 0.856%
C4	91.698%	C1 5.216% C2 3.037%
C3	91.648%	C1 6.03% C2 2.279%

This radar chart (**Analysis | Report | Target | Posterior Mean Analysis | Radar Chart**) allows interpreting the generated segments. As we can see, they are easily distinguishable (2nd technical quality).

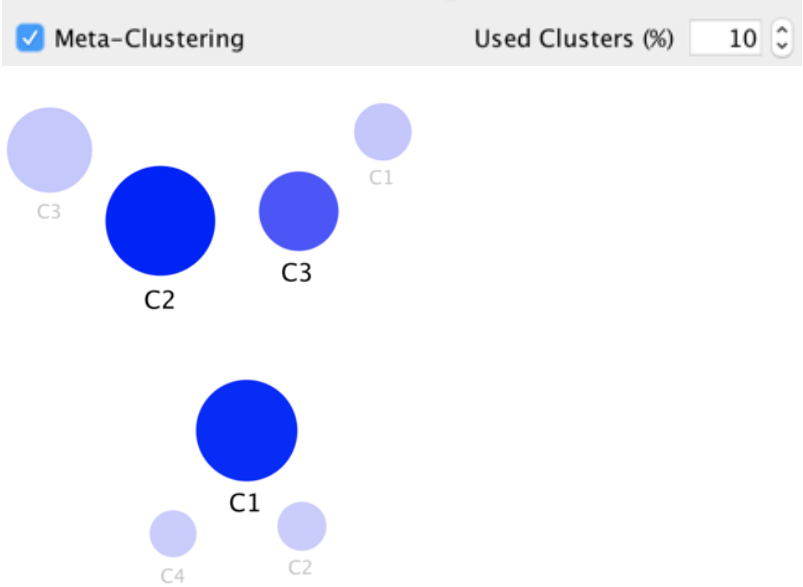


Thus, the solution with 4 segments satisfies the first two technical qualities listed above. However, what about the 3rd one, the stability? Below are the scores of the 10 best solutions that have been generated while learning:

Summary of the obtained results	
4 clusters	3.7519065884907032
3 clusters	3.7644372541928637
3 clusters	3.764451147237728
3 clusters	3.7645967754808156
3 clusters	3.7646493166454835
3 clusters	3.765328405128466
3 clusters	3.7653995397762134
3 clusters	3.7655747122206145
3 clusters	3.7656383949395953
3 clusters	3.7656948109530917

Even though the best solution is made of 4 segments, this is the only solution with 4 clusters, all the other ones have nearly the same score, but with 3 clusters. Thus, we can assume that a solution with 3 clusters would be more stable.

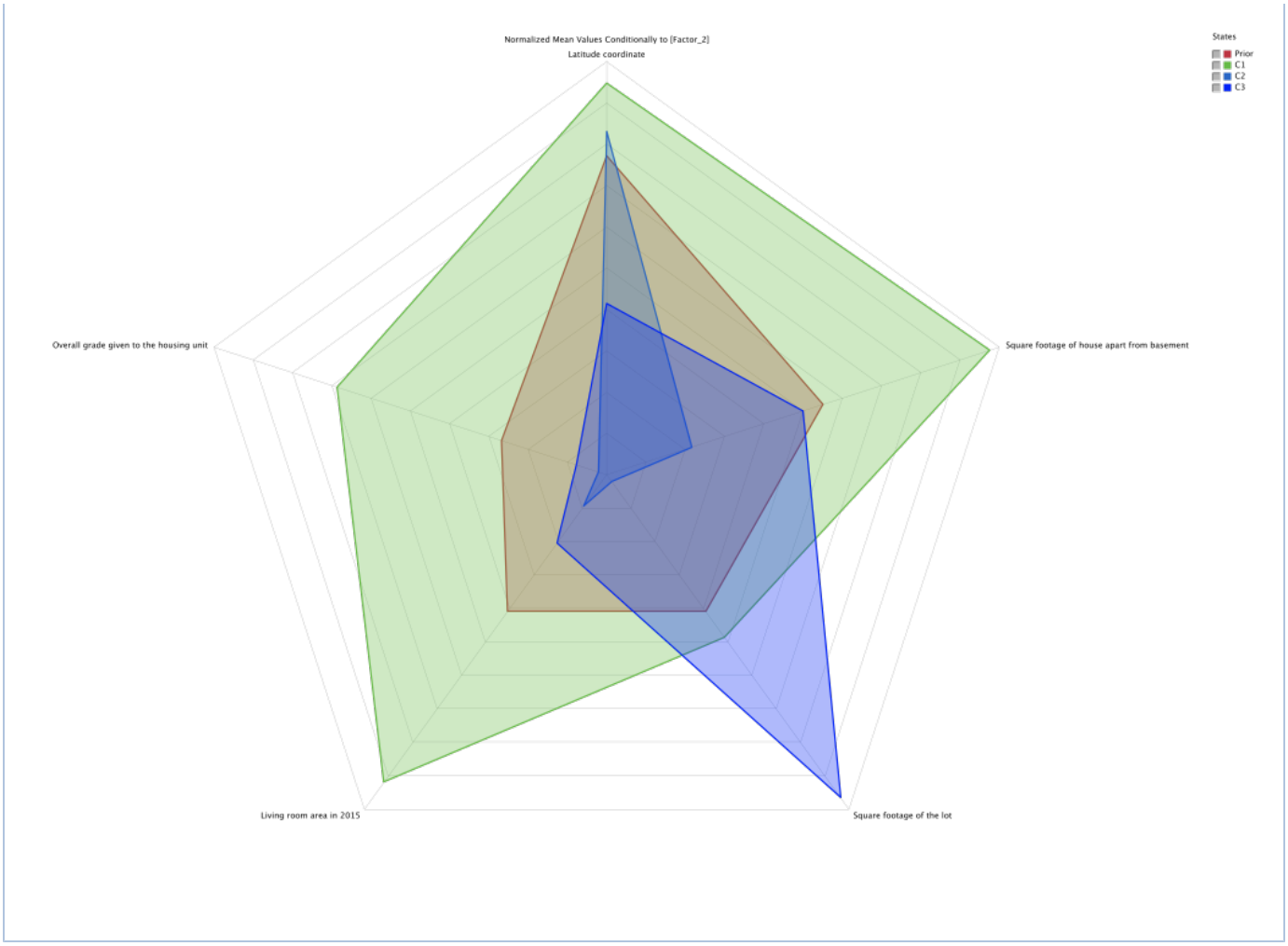
Using **Meta-Clustering** on the 10 best solutions (10%) indeed generates a final solution made of 3 clusters.



This mapping juxtaposes the mapping of the initial solution with 4 segments (lower opacity) and of the one corresponding to the meta-clustering solution.

The relationships between the final and initial segments are as follows:

- C1 groups C4 and C2 (the main difference between C4 and C2 was *Square footage of the lot*),
- C2 corresponds to C3
- C3 corresponds to C1



New Feature: Multinet

As stated in the **Context**, **Data Clustering** with Bayesian networks is typically done with **Expectation-Maximization (EM)** on a **Naive** structure. Thus, it is based on the hypothesis that the **Manifest** variables are conditionally independent of each other given ***[Factor_i]***. Therefore, the **Naive** structure is well suited for finding observations that look the same (1st purpose), but not so good for finding observations that behave similarly (2nd purpose). The behavior should be represented by direct relationships between the **Manifests**.

Our new **Multinet** clustering is an **EM**^{2} algorithm based both on a **Naive** structure (*Look*) and on a set of **Maximum Weight Spanning Trees (MWST)** (*Beh avior*). Once the distributions of the **Naive** are randomly set, the algorithm works as follows:

1. **Expectation_Naive**: the **Naive** network is used with its current distributions for computing the posterior probabilities of ***[Factor_i]***, for the entire set of observations described in the data set; These probabilities are used for hard-imputing ***[Factor_i]***, i.e. choosing the state with the highest posterior probability;
2. **Maximization_MWST**: ***[Factor_i]*** is used as a breakout variable. An **MWST** is learned on each subset of data.
3. **Expectation_MWST**: the joint probabilities of the observations are computed with each **MWST** and used for updating the imputation of ***[Factor_i].***
4. **Maximization_Naive:** based on this updated imputation, the distributions of the **Naive** network are updated via **Maximum-Likelihood.** Then, the algorithm goes back to **Expectation_Naive,** until no significant changes occur to the distributions.

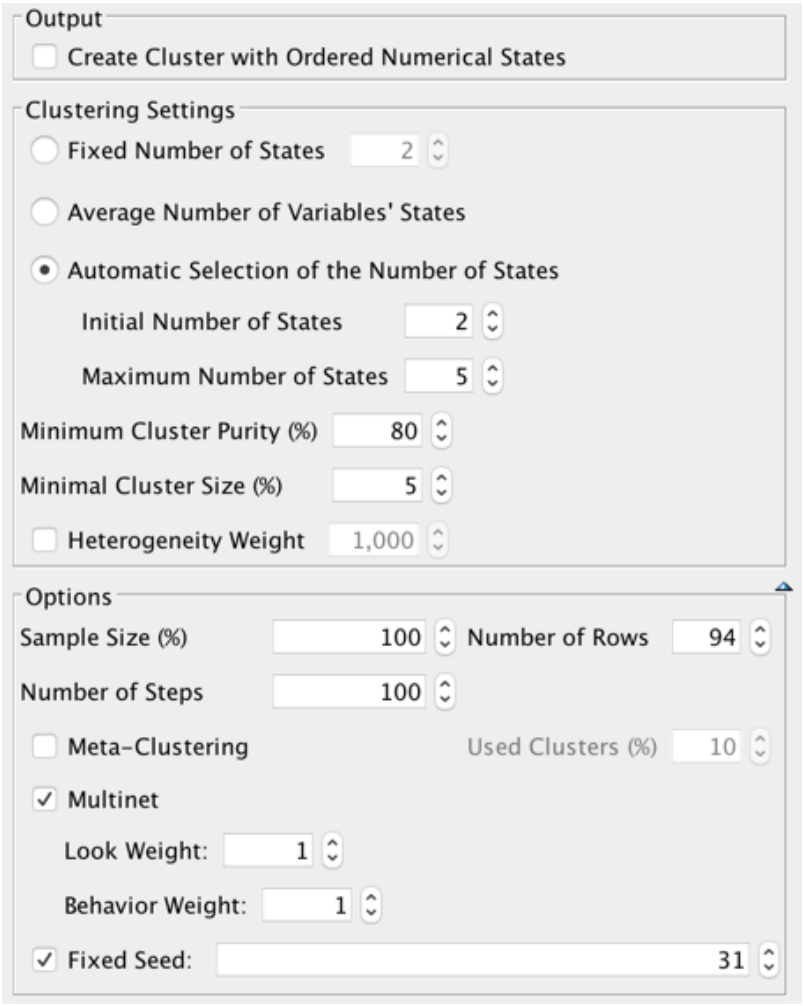
Two parameters allow changing the *Look/Behavior* equilibrium. They can be considered as probabilities to run the **Naive** and **MWST** steps at each iteration.

Setting a weight of 0 for *Behavior* defines a **Data Clustering** quite similar to the usual one, but based on hard imputation instead of soft imputation.

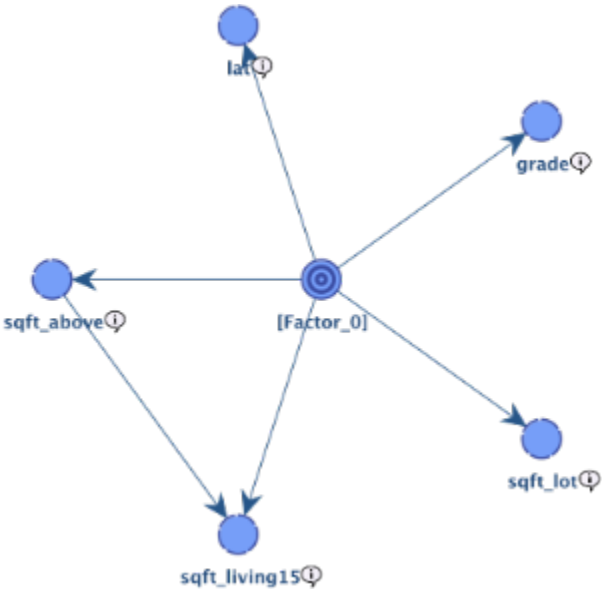
Example

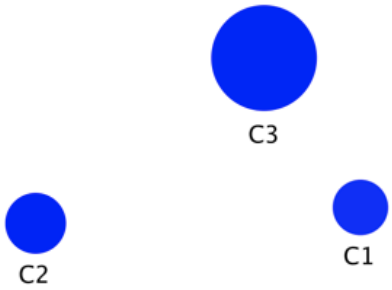
Let's use the same data set that describes houses in Seattle.

The wizard below show the settings we used for segmenting the houses:

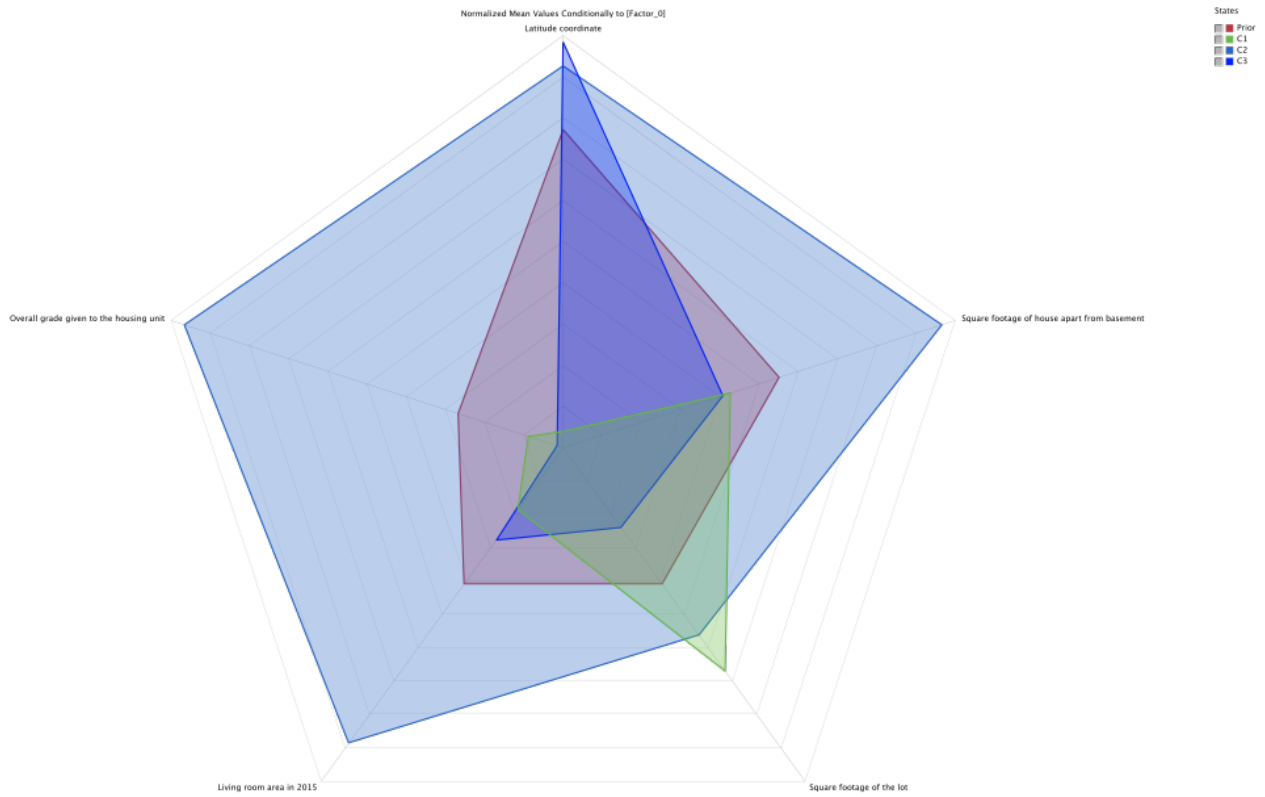


After 100 steps, segmenting the houses into three groups is the best solution. The final network is a **Naive Augmented Network**, with a direct link between two **Manifest** variables, which are, therefore, not independent given the segmentation, i.e. the *Behavior* part. Note that this dependency is valid for *C3* only, which can be seen after performing inference with the network.





The radar chart allows analyzing the *Look* of the segments.



New Feature: Heterogeneity Weight

The assumption that the data is homogeneous, given all the **Manifest Variables**, can sometimes be unrealistic. There may be significant heterogeneity in the data across unobserved groups, and it can bias the machine-learned Bayesian networks. This phenomenon is known as **Unobserved Heterogeneity,** i .e. an unobserved variable in the dataset.

Data Clustering represents a solution for searching for such hidden unobserved groups (3rd purpose). However, whereas the default scoring function in **Da ta Clustering** is based on the entropy of the data, finding heterogenous groups requires modifying the scoring function.

We thus defined a **Heterogeneity Index** HI:

$$HI = \frac{\sum_{f \in F} p(f) I_{\mathcal{D}_f}}{I_{\mathcal{D}}} - 1$$

,

with

$$I_{\mathcal{D}} = \sum_{X \in \mathcal{M}} I_{\mathcal{D}}(X, T)$$

where:

- F is the induced **Factor,**
- f are the states of the **Factor**, i.e. the segments used to split the data,
- $p(f)$ is the marginal probability of state f, i.e. the relative size of the segment,
- $\mathcal{M}$ is the set of **Manifest** variables,
- $\mathcal{D}$ is the entire data set,
- $\mathcal{D}_f$ is the subset of the data that corresponds to the segment f
- $I_{\mathcal{D}}(X, T)$ is the **Mutual Information** between the **Manifest** variable X and the **Target** node T computed on the data set $\mathcal{D}$.

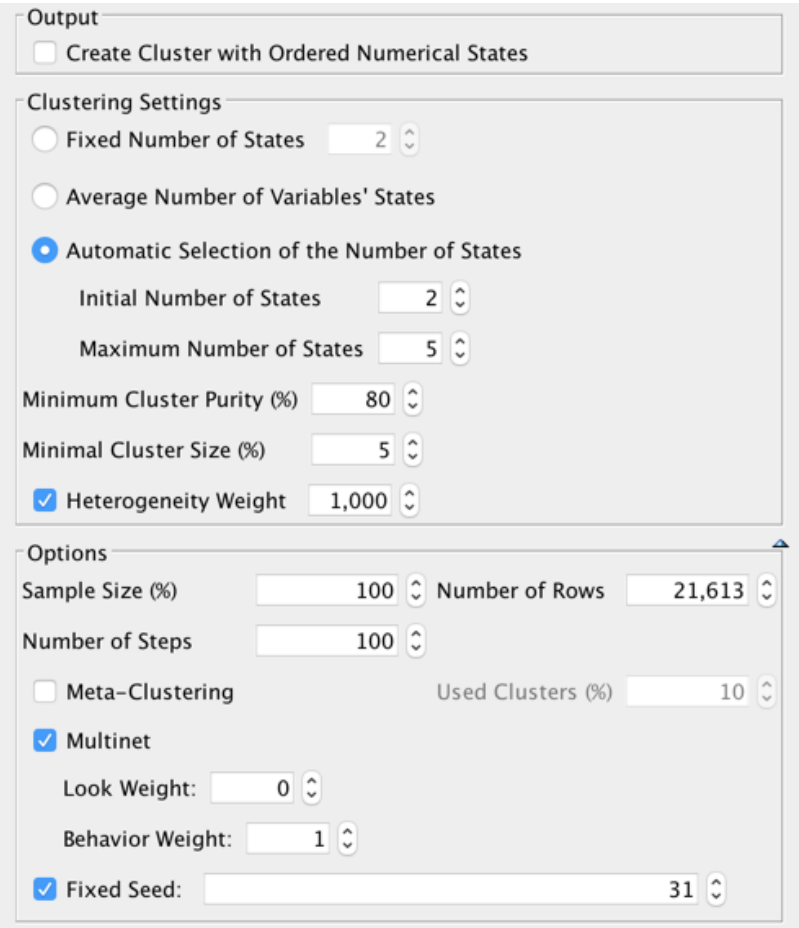
The **Heterogeneity Weight** allows setting a weight of the **Heterogeneity Index** in the score, which will, therefore, bias the selection of the solutions toward segmentations that maximize the **Mutual Information** of the **Manifest** variables with the **Target Node**.

Example

Let's use the entire data set that describes houses in Seattle, with this subset of **Manifest** variables:

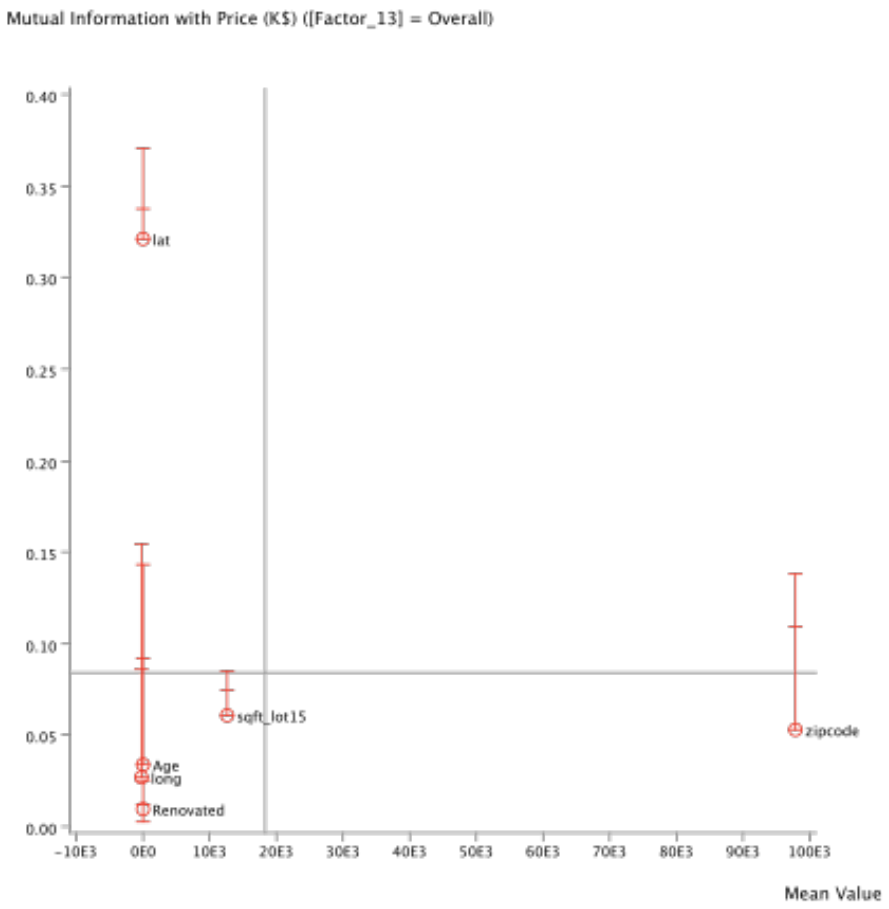
- *Renovated*: indicates if the house has been renovated
- *Age*: Age of the house
- *sqft_living15*: Living room area in 2015
- *long*: Longitude coordinate
- *lat*: Latitude coordinate
- *Price (K$)*: Price of the house**.**

After setting *Price (K$)* as a Target Node and selecting all the other variables, we use the following settings for **Data Clustering**:



This returns a solution with 2 segments, generating a **Heterogeneity Index** of 60%. This indicates, therefore, that using ***[Factor_i]*** as a breakout variable would increase the sum of the **Mutual Informations** of the **Manifest** variables with the **Target Node** by 60 %.

The **Multi-Quadrant** below highlights the improvement of the **Mutual Information**. The points correspond to the **Mutual Informations** on the entire data set, and the vertical scales show the variations of the **Mutual Informations** by splitting the data based on the values of ***[Factor_i].***



The **Heterogeneity Index** is computed on the **Manifest** variables that are used during the segmentation only. In order to take into account other variables in the computation of the index, these variables have to be included in the segmentation, with a weight of 0 for preventing them to influence the segmentation.

New Feature: Random Weights

By default, the weight associated with a variable is set to 1. Whereas a weight of 0 renders the variable purely illustrative, a weight of 2 is equivalent to duplicating the variable in the dataset. The option **Mutual Information Weight**, introduced in version 5.1, allows weighting the variable by taking into account its **Mutual Information** with the **Target** node.

As of version 7.0, a new option, **Random Weights**, allows to modify the weight values randomly while trying the find the best segmentation. The amplitude of the randomness is inversely proportional to the current number of trials, therefore starting with the maximum level of randomness and ending with almost no weight modification. This option can be useful for escaping from local minima.


Node Weights
Mutual Information Weight
0.1
Random Weights
Edit Node Weights