StarDust – Quick User's guide v1.06

Company headquarters:
Address: Chemin des 2 Villers, 11 - 7812 Ath (V.N.D.) - Belgium
Phone (global): +32 479 99 27 68
Phone (Americas): + 57 300 675 13 69

Creation date: October 2008
Last Edited: July 2019
Last Edited by Frank Vanden Berghen

1. Introduction

Welcome to the Quick user's guide to ***StarDust*** !

This document will guide you through the process of creating and interpreting a segmentation model using StarDust. The concepts used to create a good segmentation analysis are quite abstract and a small knowledge of college-grade mathematics is required to read this guide. Don't worry! Stardust is fully automatized and really easy to use! This document is self-contained and no additional knowledge is required before reading it. Once you have grasp the small concepts explained in this guide, you will be able to create the most advanced segmentation models with StarDust! With StarDust you can easily explore terabytes of data and extract some useful knowledge. Discovering new insights about your customers should be fun and easy. …and now, it's the case with Stardust! It's a whole new (VR) world waiting to be discovered hidden inside your databases.

This document does not cover the creation of **predictive** models with TIMi.

During the course of this document we will analyse together a dataset named "census–income" (in statistics, the "data-tables" that are analysed are named "datasets"). This dataset contains data about the financial incomes of the inhabitants of the United State of America. Here is an extraction of this dataset:

key	**Is taxable income amount above $50K ?**	age	education	marital stat	race	sex	country of birth	weeks worked in year
1	0	73	High school graduate	Widowed	White	F	USA	0
2	0	58	Some college but no degree	Divorced	White	M	USA	52
3	0	18	10th grade	Never married	Asian	F	Vietnam	0
4	0	9	Children	Never married	White	F	USA	0
5	0	10	Children	Never married	White	F	USA	0
6	0	48	Some college but no degree	Married-civilian	Indian	F	USA	52
7	0	42	Bachelors degree(BA AB BS)	Married-civilian	White	M	USA	52
8	1	28	High school graduate	Never married	White	F	USA	30
9	0	47	Some college but no degree	Married-civilian	White	F	USA	52
10	0	34	Some college but no degree	Married-civilian	White	M	USA	52
11	0	8	Children	Never married	White	F	USA	0
13	0	51	Some college but no degree	Married-civilian	White	M	USA	52
14	1	46	High school graduate	Divorced	White	F	Columbia	52
15	0	26	Bachelors degree(BA AB BS)	Never married	White	F	USA	52
16	0	13	Children	Never married	Black	F	USA	0
17	0	47	Bachelors degree(BA AB BS)	Never married	White	F	USA	52
18	0	39	10th grade	Married-civilian	White	F	Mexico	0
19	0	16	10th grade	Never married	White	F	USA	0
20	0	35	High school graduate	Married-civilian	White	M	USA	49

In opposition to a predictive model, we don't have any "Target" column to explain during a segmentation analysis. What's interest us is the segment of customers inside your database.

The "census-income" dataset contains a special column: the "primary key column" or, in other words, the "primary key". The "primary key" contains a different value on each line of the dataset. Its utility is to be able to define in a unilateral way each line of our dataset. The concept of “primary key” is well known in the database world. If you want to know more about this subject, I suggest that you ask your database administrator. The “primary key column” in our dataset is named “key”.

StarDust is able to process datasets stored in many formats. Your datasets can be stored inside relational databases (like Microsoft SQL server, Oracle, Informatix, MySQL,...), SAS dataset files (.sas7bdat files) or simple “flat files”. The preferred storage format for TIMi is a CSV-flat-file compressed in RAR format.

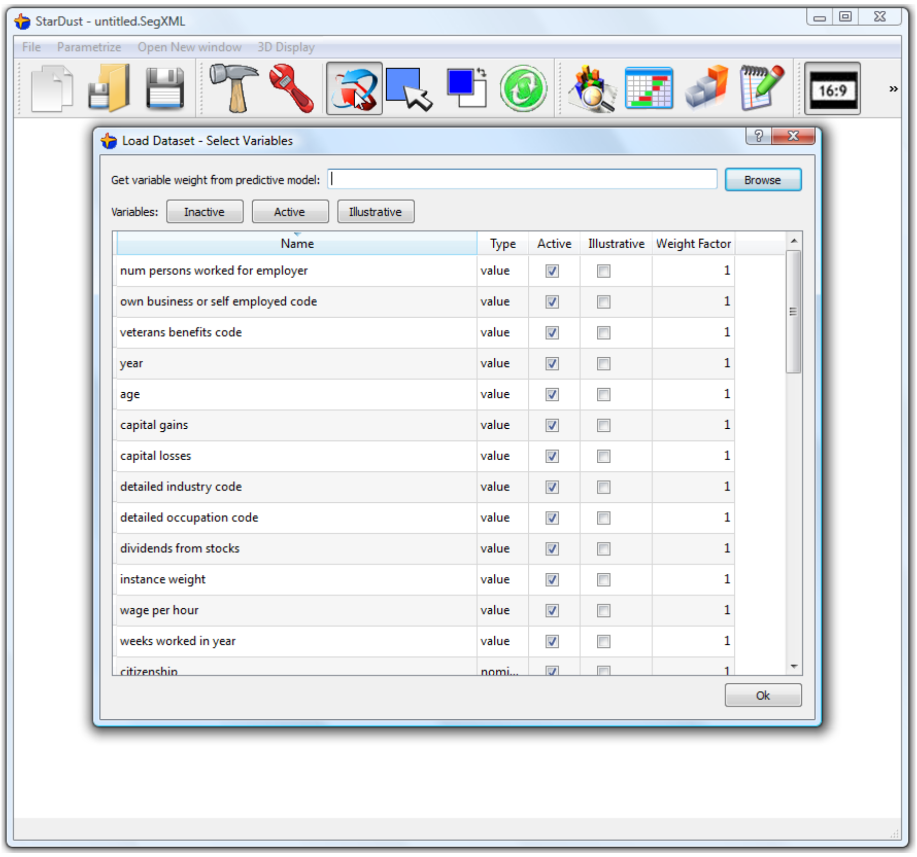
1.1. Quick start

For those of you that are in a hurry and already know everything there is to know about TIMi, PCA, K-means algorithm and Ward’s Algorithm, you can directly obtain a segmentation in less than 30 seconds and 5 mouse clicks in this way:

1. Run StarDust (click on the StarDust icon or on the Start Segmentation Analysis button inside the “.TypeXML file” editor). (This is mouse click “1”).

2. Open a dataset: To open a dataset a dataset in Stardust, you must actually open the associated “.DescXML file”. Click the button and browse for the “.DescXML file” associated with the dataset that you want to analyze. (This is mouse click “2”).

3. A new window appears:



Get variable weight from predictive model:

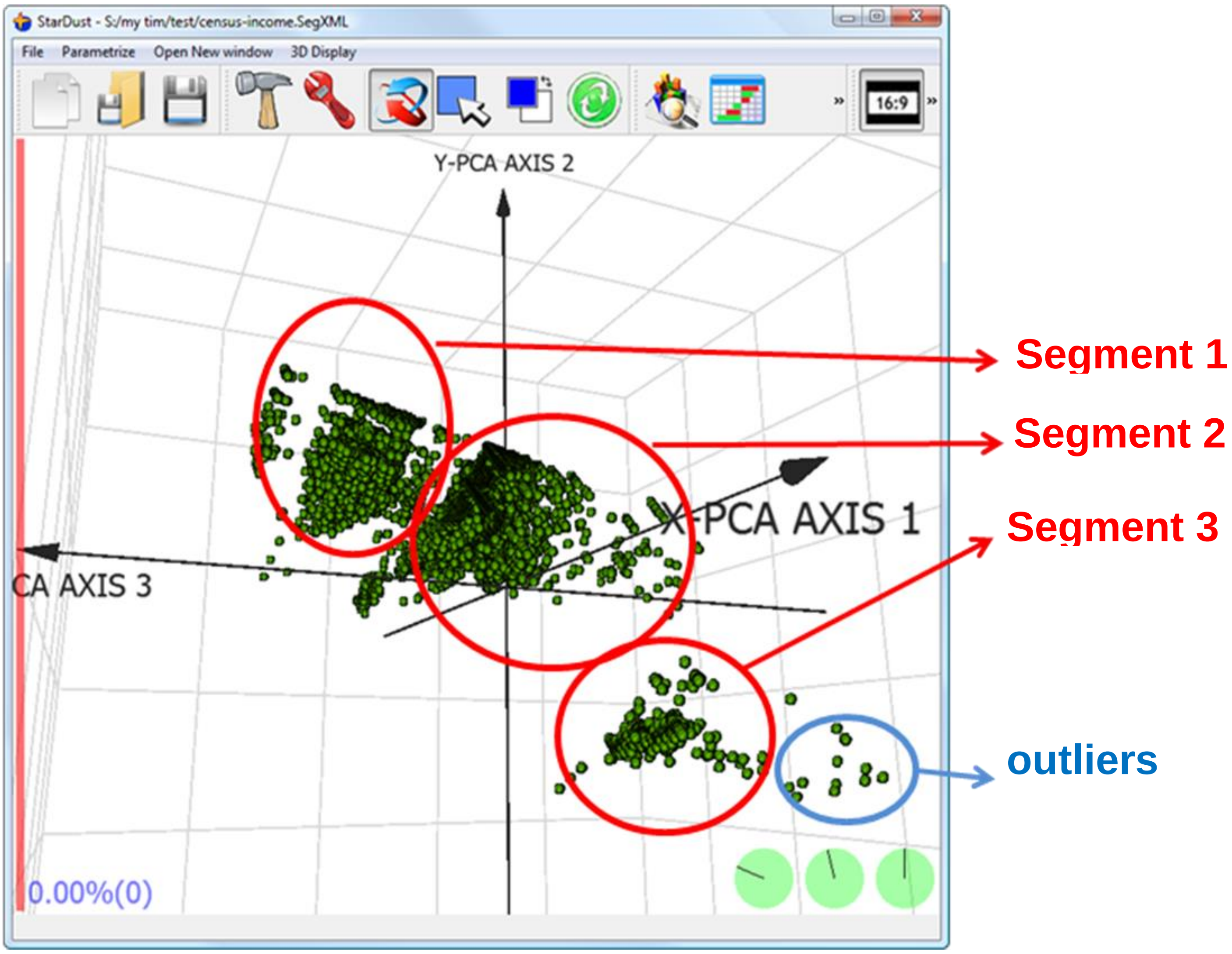
Variables: Inactive | Active | Illustrative

Name	Type	Active	Illustrative	Weight Factor
num persons worked for employer	value	✓	☐	1
own business or self employed code	value	✓	☐	1
veterans benefits code	value	✓	☐	1
year	value	✓	☐	1
age	value	✓	☐	1
capital gains	value	✓	☐	1
capital losses	value	✓	☐	1
detailed industry code	value	✓	☐	1
detailed occupation code	value	✓	☐	1
dividends from stocks	value	✓	☐	1
instance weight	value	✓	☐	1
wage per hour	value	✓	☐	1
weeks worked in year	value	✓	☐	1
citizenship	nomi...	✓	☐	1

This window allows you to, basically, select which column will be loaded into memory for analysis. For more information on this window, see the end of section 4.2.2. You can directly click the Ok button (This is mouse click "3").

4. The next "parameter window" allows you to configure the normalization of the variables, the PCA, the distance used inside the clustering algorithms (Euclidean, Cosine, Etc.), the K-means algorithm parameters in itself, the Ward's algorithm, the 3D display and so on... You can use the default values for all settings: click the Ok button (This is mouse click "4").

5. You get a first 3D view of your population. You can already directly "see for yourself" the segments (and some outliers). You obtain something like that:



If you cannot see any segments, it means that you dataset most certainly contains some outliers, see section 5.4.1 to know how to remove them.

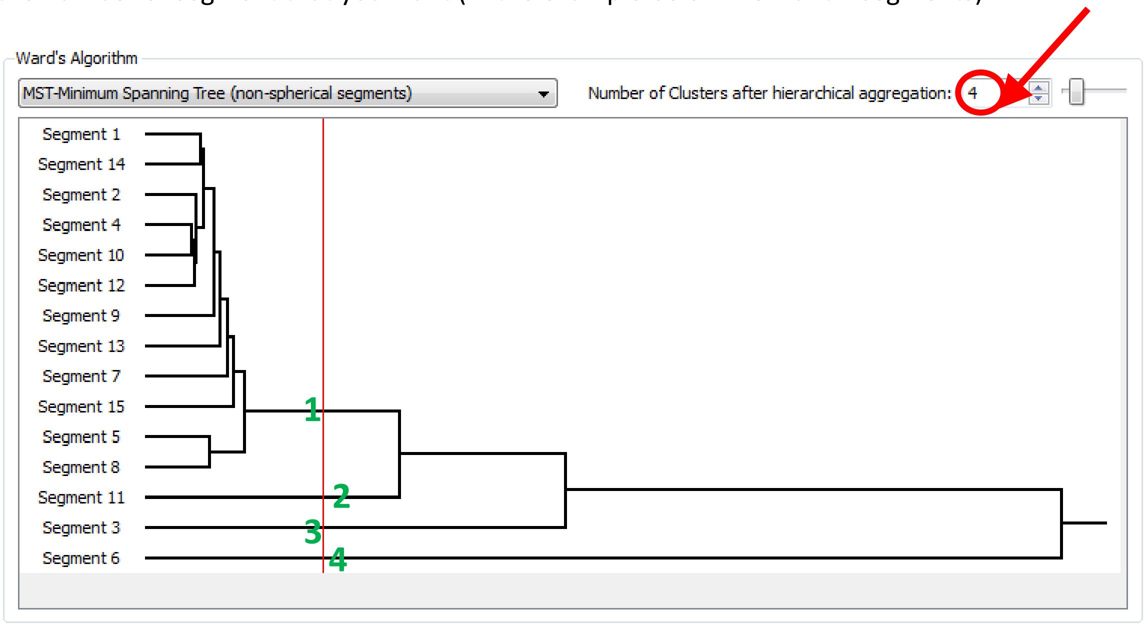
See the section 5.3 to know how to configure this 3D display to easily explore all your data. You can also use the very powerful "multivariate PCA zooming" techniques described in section 5.4.2 to visually explore more precisely a specific subset of your dataset.

6. Click repetitively on the button to obtain different segmentations (This is the final mouse click: mouse click number “5”). Each time you click on the button, a completely new segmentation is computed and shown.

   The (many) parameters used to create the segments are all accessible when you click the button or the button. The main difference between the button and the button is that the later is applying your modifications “in real-time”: i.e. You don’t have to click the Ok button to see the result of your modifications: you can directly see “**in real-time**” all the windows of StarDust updating.

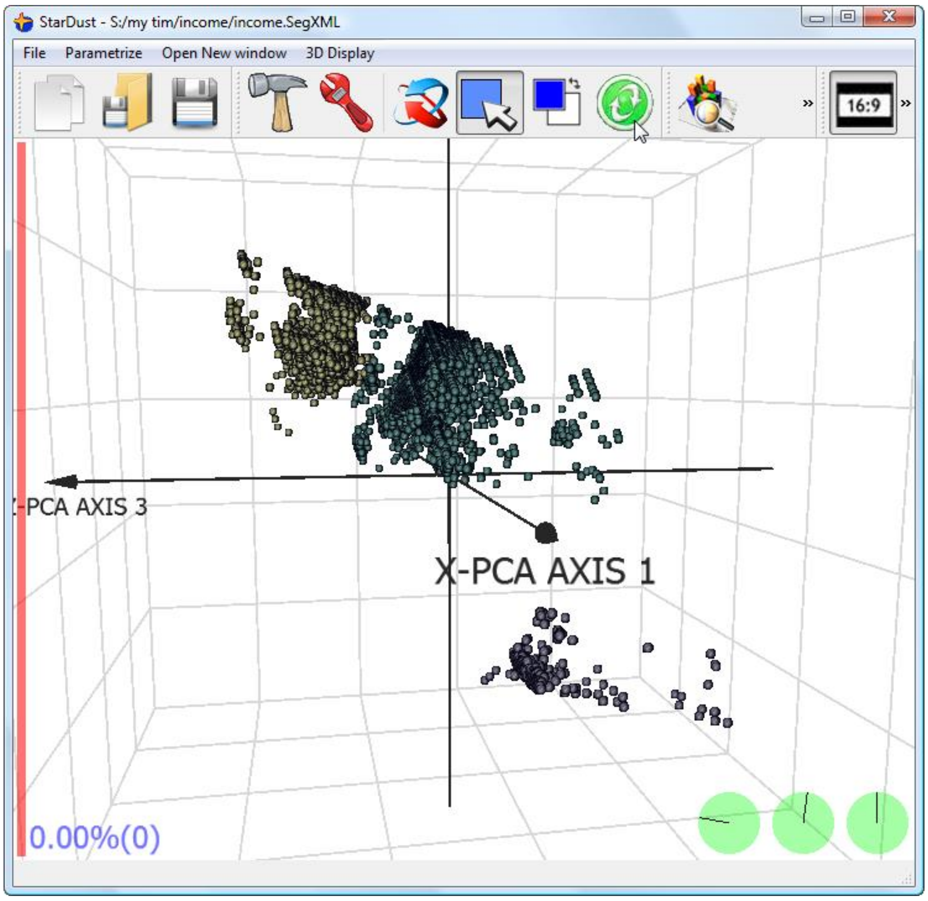
   See section 5.5 to know more about all the different parameters of the segmentation algorithm of StarDust.

   For example, the most important parameter is the number of segments that you want to obtain. To choose the right number of segments, click the button and select the “segmentation 2” tab: The position of the vertical red line inside the following window defines the number of segment that you want (in the example below we want 4 segments):

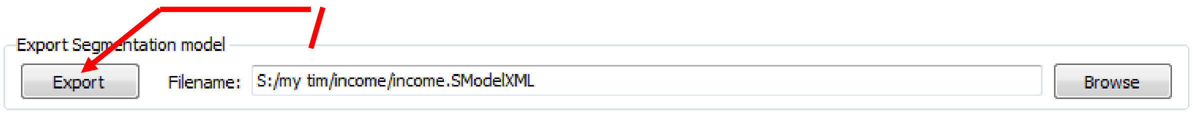


Number of Clusters after hierarchical aggregation: 4

   Click anywhere inside the “hierarchical-tree” window to move the red line: it changes the number of segments (See section 5.5.3 to have more information about this window and the Ward’s algorithm). You can instantaneously see the results of your decision inside the main 3D display window: When you move the red vertical line with your mouse, it changes the (number of) segments and the colours of the points inside the main 3D display are (instantaneously) changing. Inside the main 3D display, each colour represents a different segment: for example, in the example below there are 3 segments (because there are 3 different colours: yellow, purple, blue turquoise):



7. Once you have found a nice segmentation, you can save the segmentation model, to use it later in combination with TIMi: Select the “segmentation 3” tab and click the “export” button:



Filename: S:/my tim/income/income.SModelXML

(See section 5.5.6 for more information on segmentation models)

8. You can also “explain” the different segments, using different charts and summary tables. See sections 5.5.7 and 5.2.12 to know how to create nice and colourful reports that explains your segments to the business- users.

2. Introduction to the PCA techniques

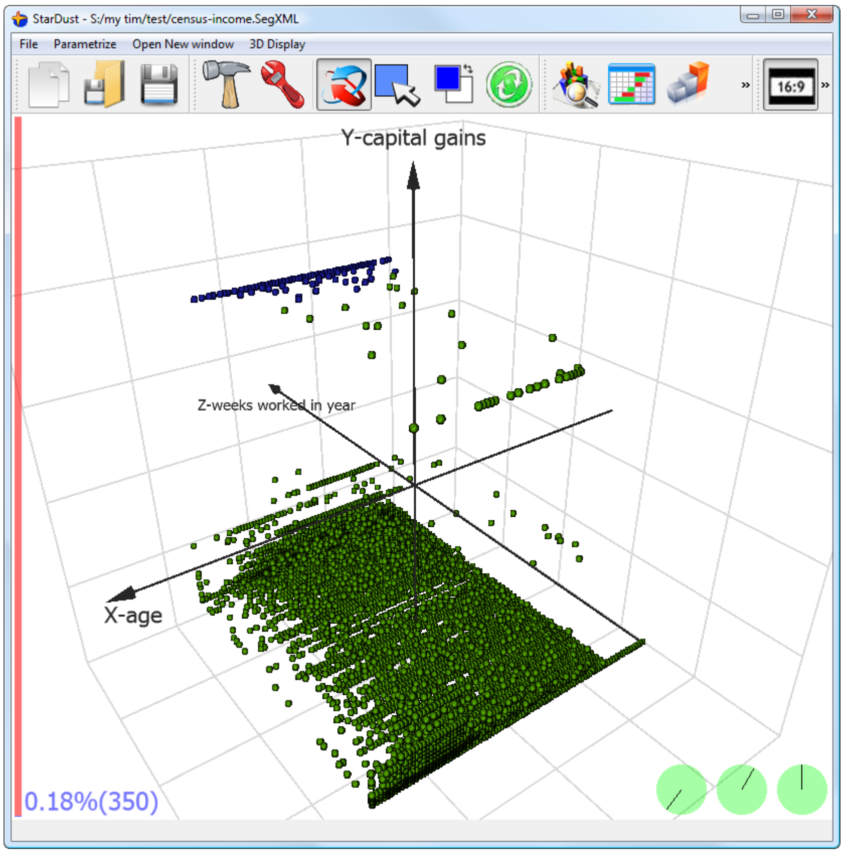
2.1. Dimensionality reduction

StarDust allows you to visually explore on your computer screen the entire studied population. This visual exploration allows you to find interesting figures about your data:

- Detect Outliers and non-valid data
- Find Segments of customers having similar behaviour
- …and Discover hidden patterns inside your data

Let's directly illustrate the study using the classical census-income database. Each row of the census-income database is an individual.

We can represent this database graphically in the following way:



On this graphic, each "screen-dot" is an individual. The coordinates of one dot (one individual) are:

- X is the "age"
- Y is the "capital gain"
- Z are "weeks worked in year"

We could have used other information about the individuals to obtain a 3D coordinate because a 3D coordinate is, in the end, only 3 numbers. … and these 3 numbers could come from other columns, also present inside the dataset: We could have used:

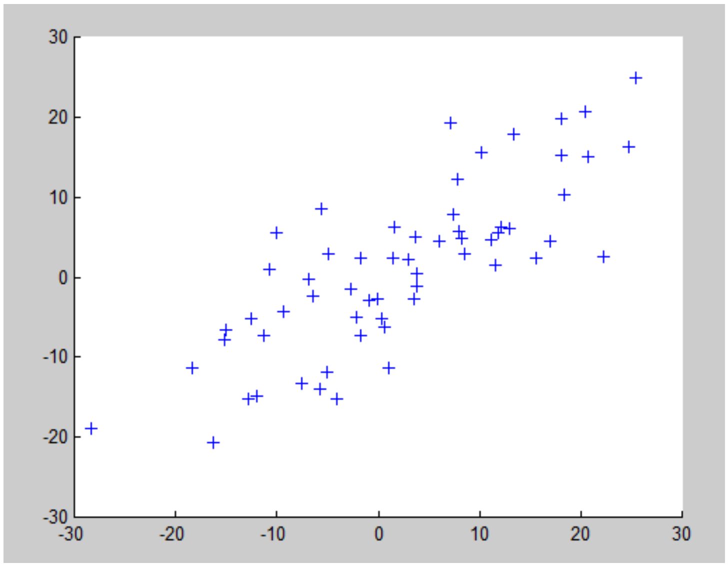
1. "number of people working for an employer"
2. "capital losses"
3. "dividends from stocks"
4. "wages per hour "
5. "age"

6. “capital gain”
7. “weeks worked in year”

There are 7 numerical variables for each individual inside our dataset. So, we could represent an individual by a point with some coordinates in 7D. Unfortunately the human eye can only see in 3 dimensions so we could only select 3 columns: the “age”, the “capital gain” and the “weeks worked in year”. If we want to represent an individual exactly, we need to be able to see in, not only 3D, but in 7D, so it’s not possible. BUT there is a “trick”. The common trick is using a PCA. PCA stands for “principal component analysis”. The PCA is a common « trick » to do « dimensionality reduction ».

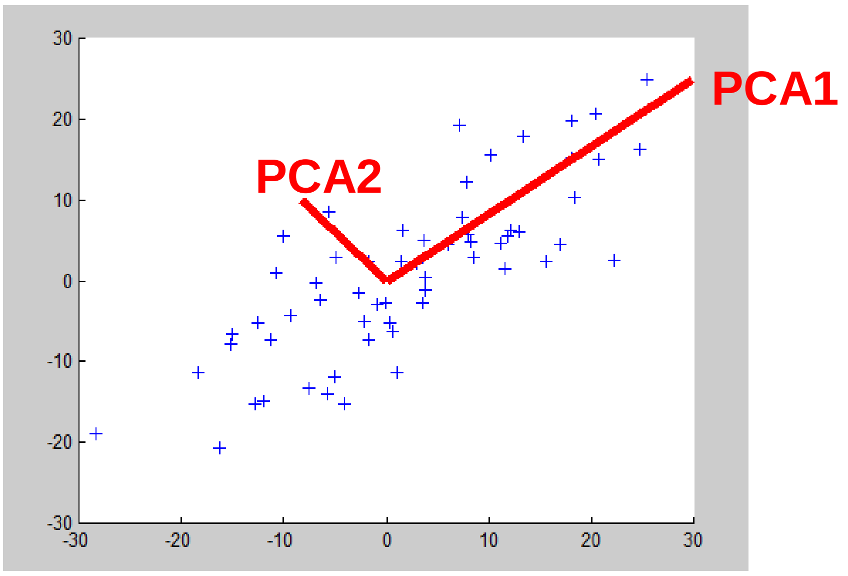
2.1.1. A first example: from 2D to 1D

To explain how to use the PCA technique to do « dimentionality reduction », we will start with a small example. We have here many points in 2D (2 dimensions) and we want to reduce the dimension to 1D:

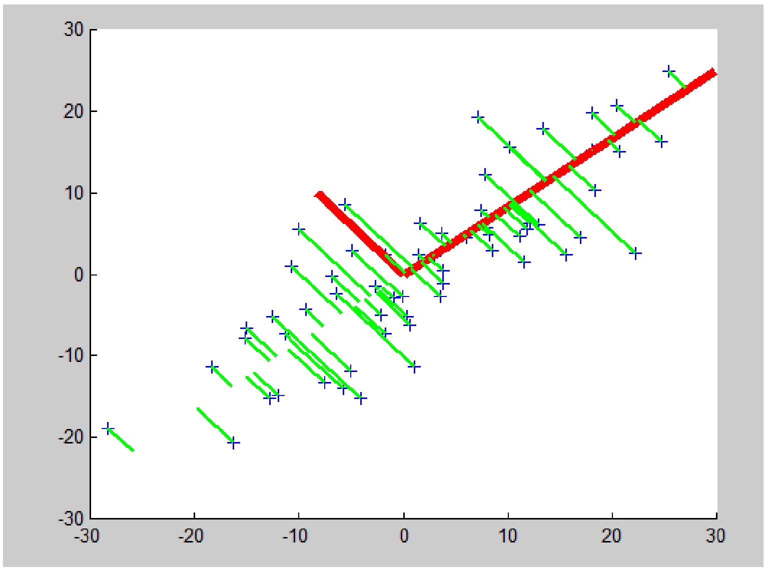


(For the census-income database, we have points in 7D and we want to reduce the coordinates in 3D).

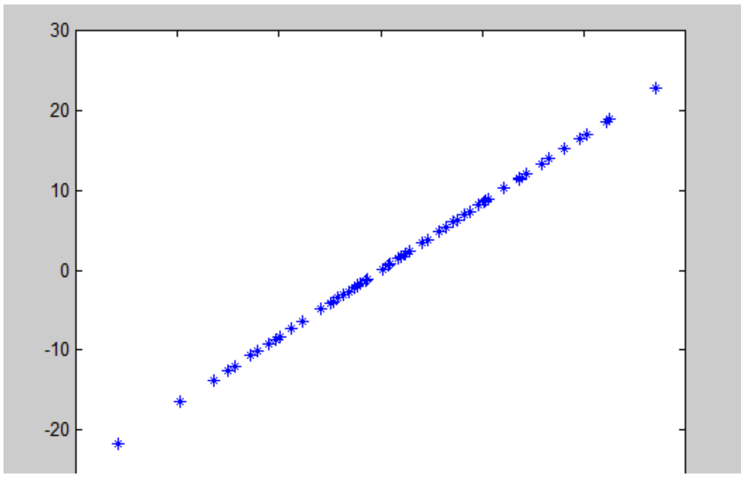
The output of the PCA analysis is a set of different perpendicular directions represented in red on the chart below:



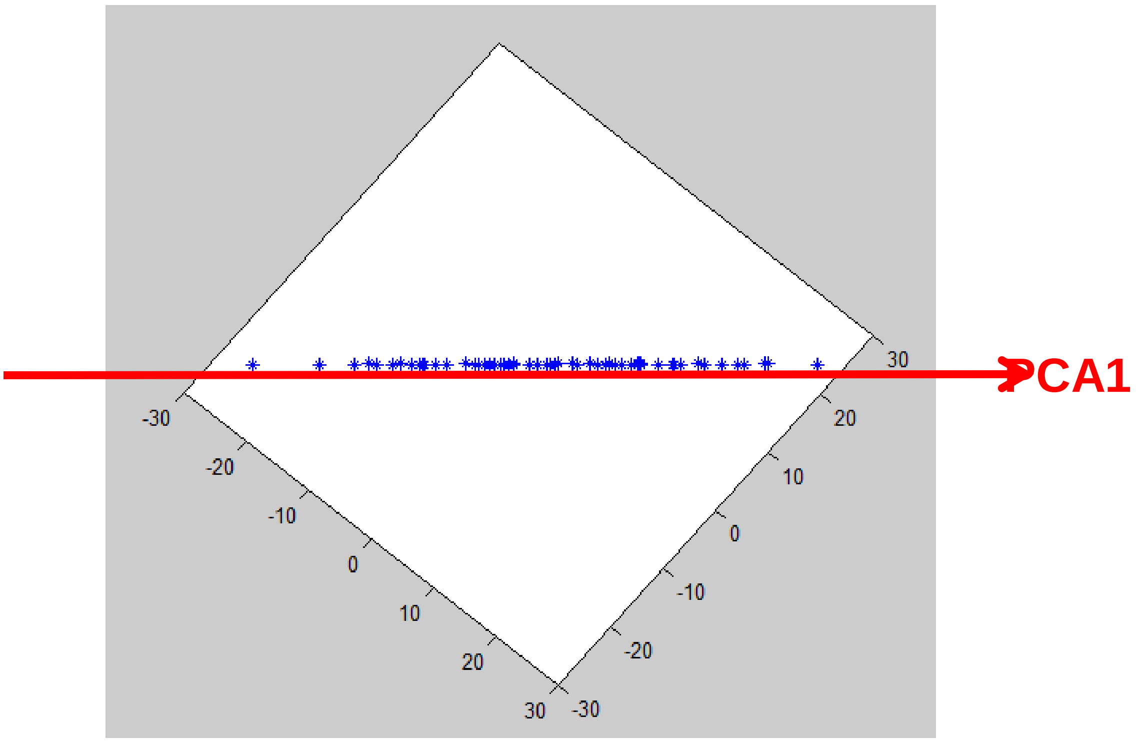
Let’s project all the blue points of the database on the first PCA axis (**PCA1**): the projection is illustrated in this chart in green:



We obtain a new dataset that is represented here in blue:



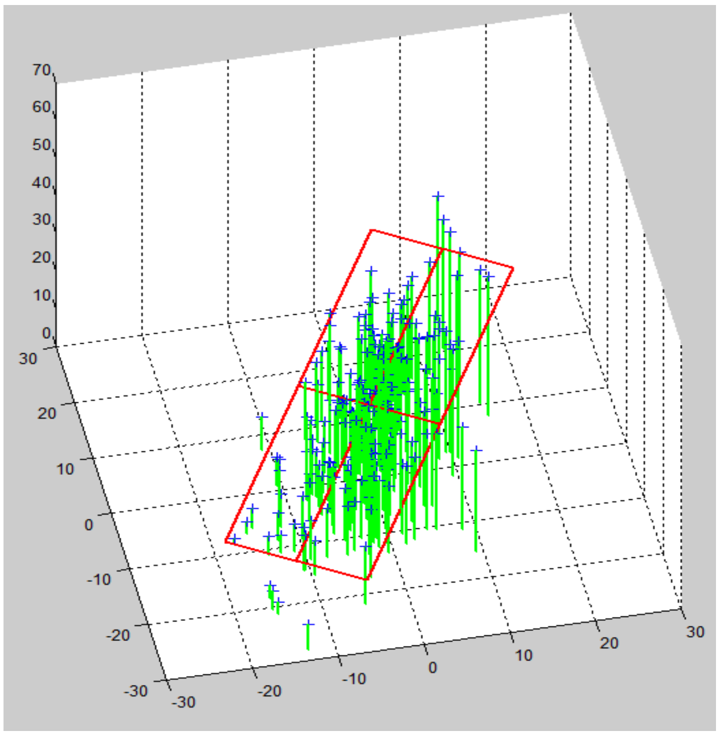
This new dataset can be “seen” in one dimension along the PCA1 direction:

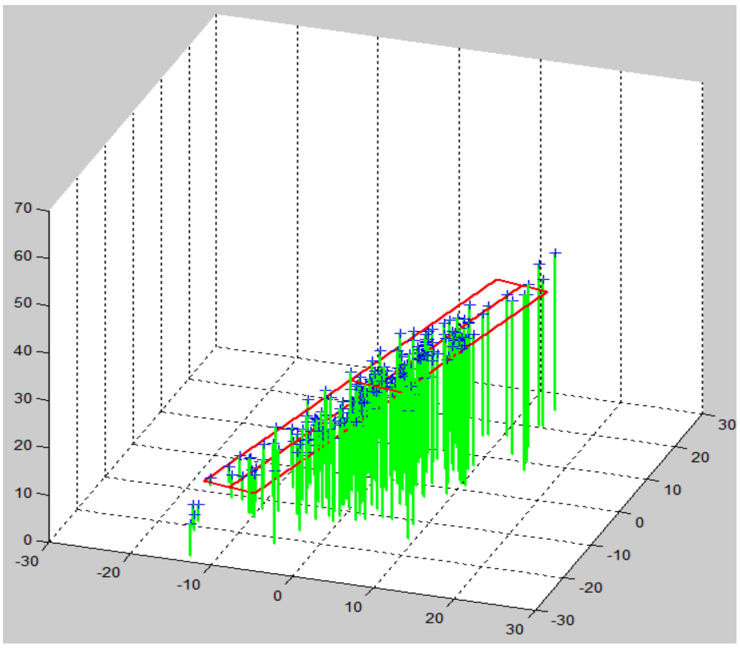


We just reduced the dimension: the 2D points are now in a one-dimensional-space. During the dimensionality reduction, we didn’t lose too much information because the distance between the (same) point “before” and “after” projection is small. This is thus a good dimensionality reduction.

2.2.2. A second example: from 3D to 2D

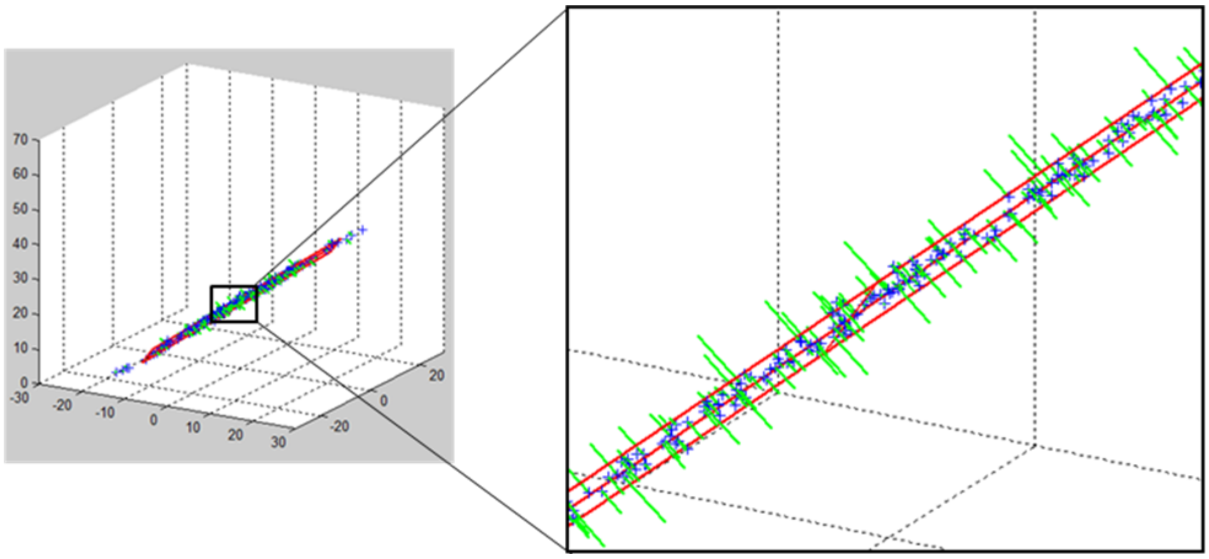
Here is another example:

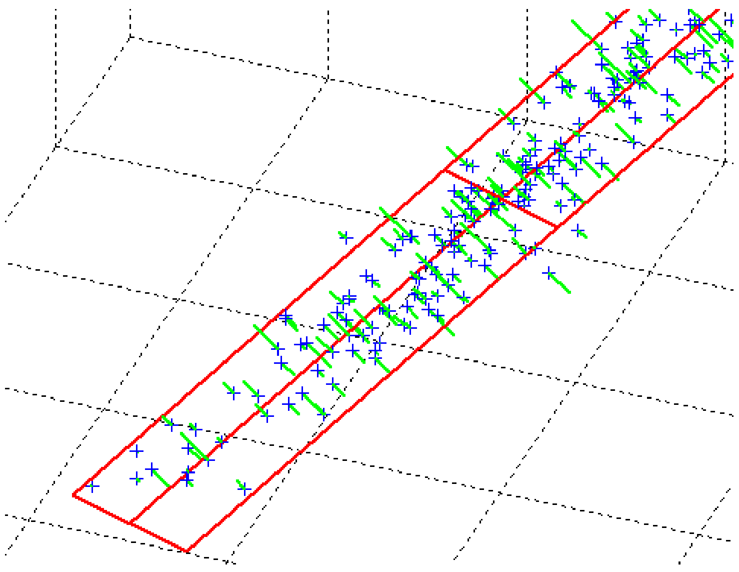


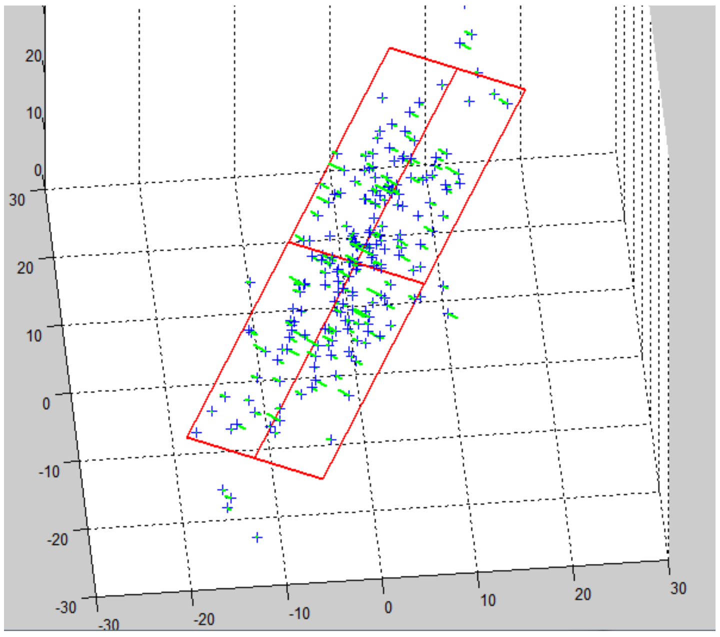


We start from 3D points and we want to reduce the dimension to 2D. The Points in 3D are in blue. The Projection of the points on the « floor » is represented with green lines. The PCA1 and PCA2 axises are in red.

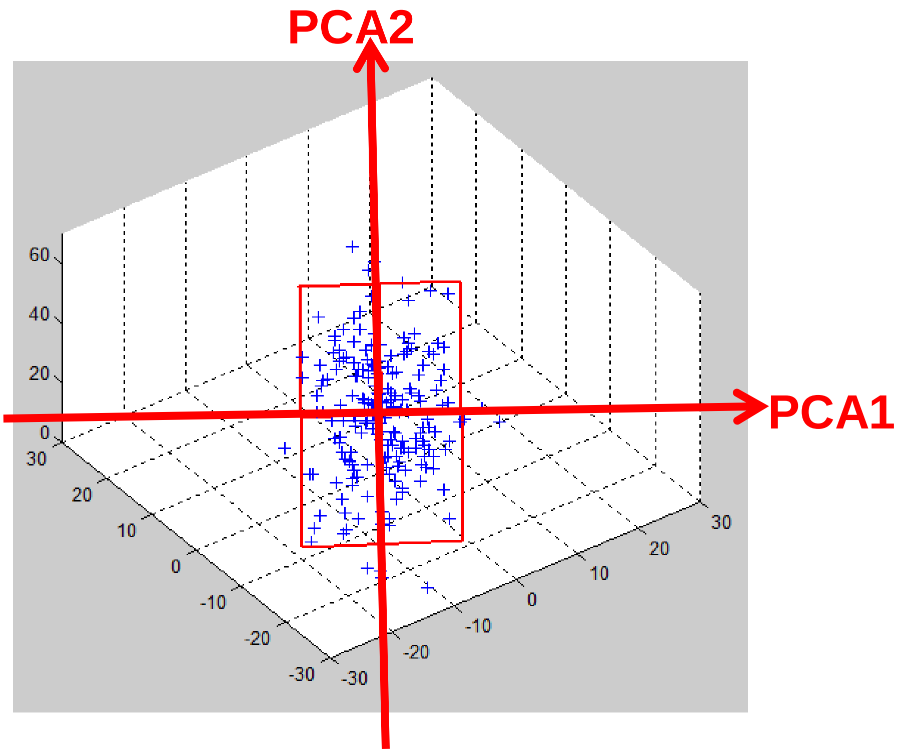
Let's project all the 3D points on the plane defined by the PCA1 and PCA2 axises. The points after projection are in Blue. The projection direction in itself is in green. Please, note that the distance between the points Before and After projection is very small, so we don't lose much information when we are projecting all the points on the PCA plane (we didn't disturb the "true original position" of the points too much when projecting):







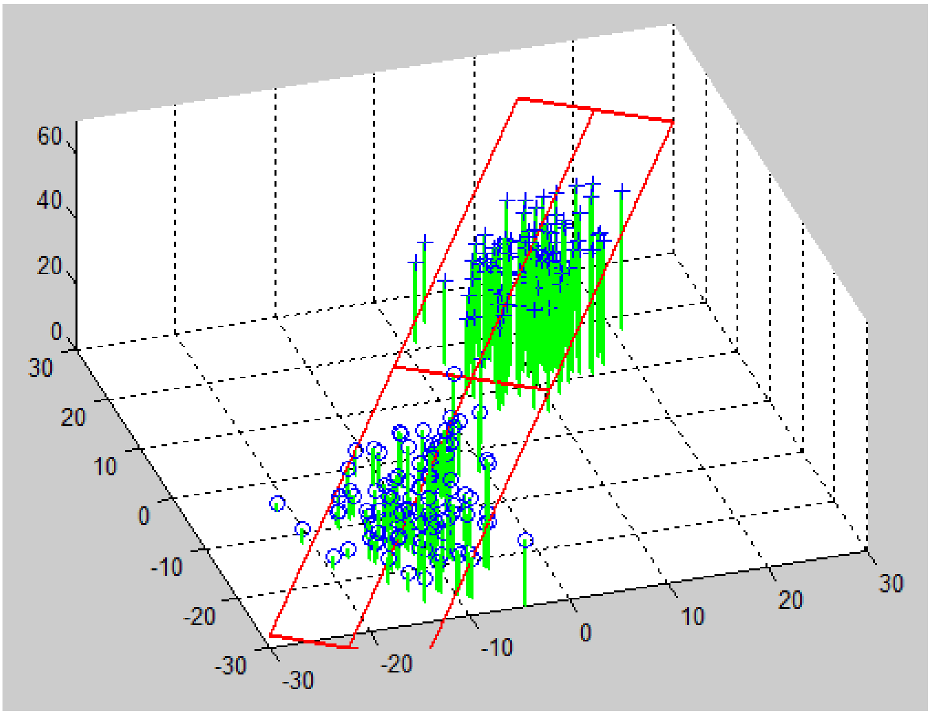
After projection, the 3D points are now in a 2D space formed by the PCA1 and PCA2 axises. We just reduced the dimension from 3D to 2D without losing too much information:



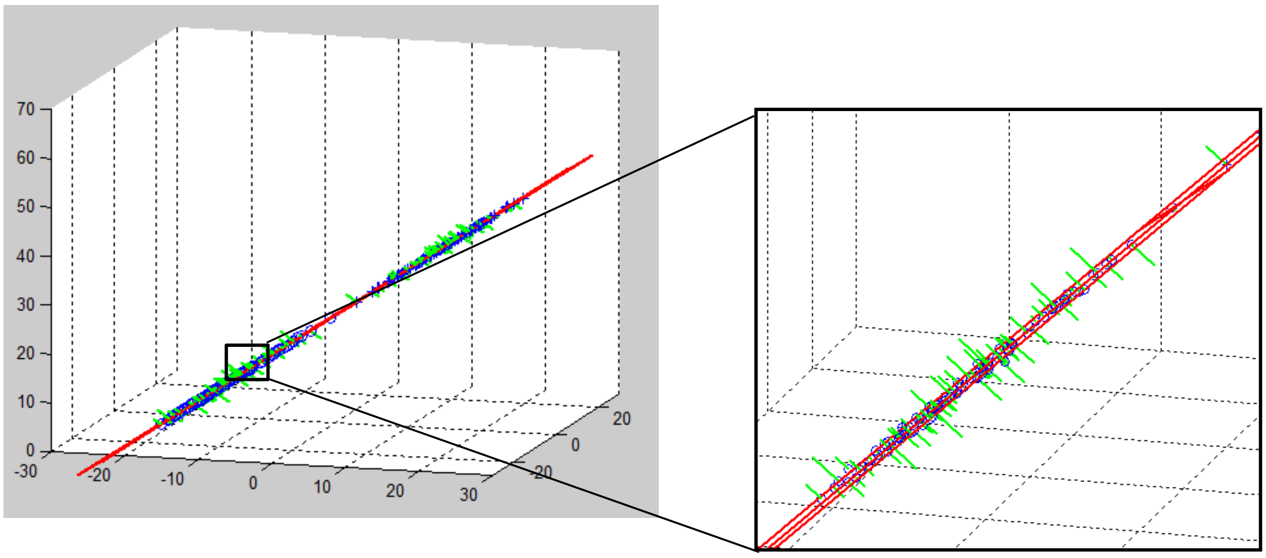
To summarize, the PCA is a technique that allows representation of high dimension « points » into a lower dimensional space. The first PCA axises are defining priviliged directions that are the best directions on which to « project » in order to « loose » as little information as possible.

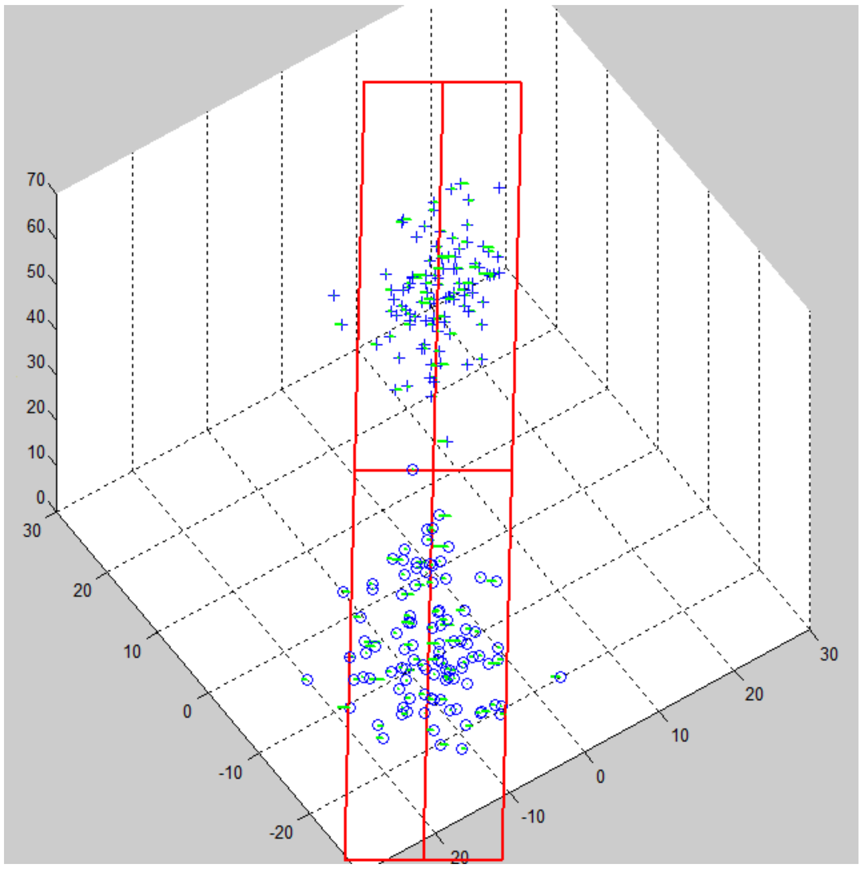
2.2.3. Viewing segments

Let's consider another example. We still want to reduce the dimension of the 3D points to only 2D. Please note that there are two different families of points: the crosses and the circles.



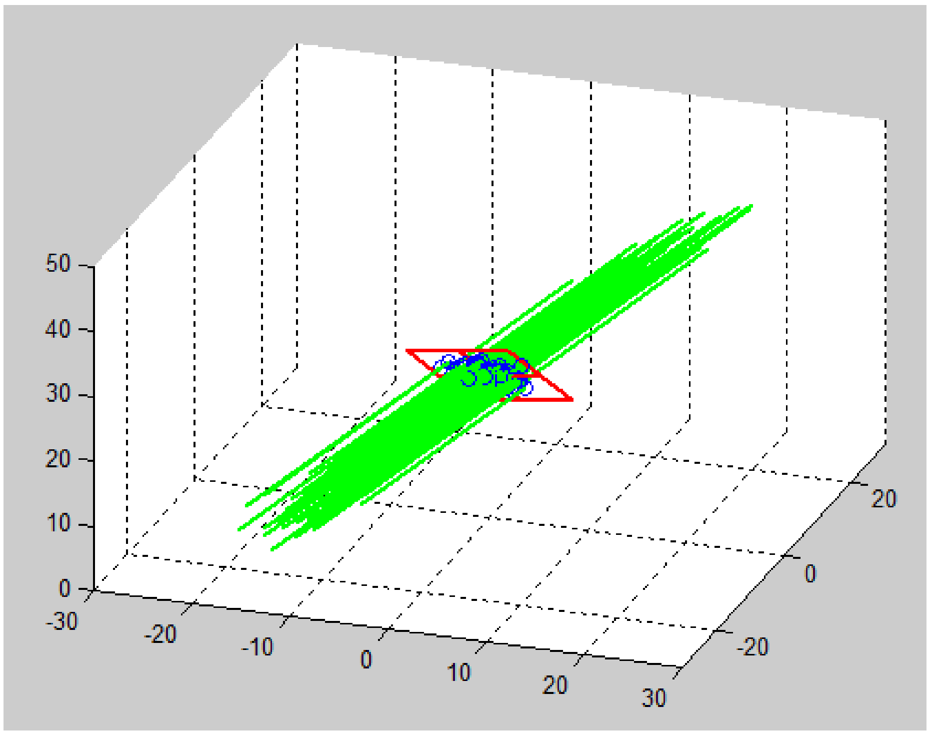
Let's project all the 3D points on the plane defined by the PCA1 and PCA2 axises. The points after projection are represented in Blue. The projection direction in itself is in green:



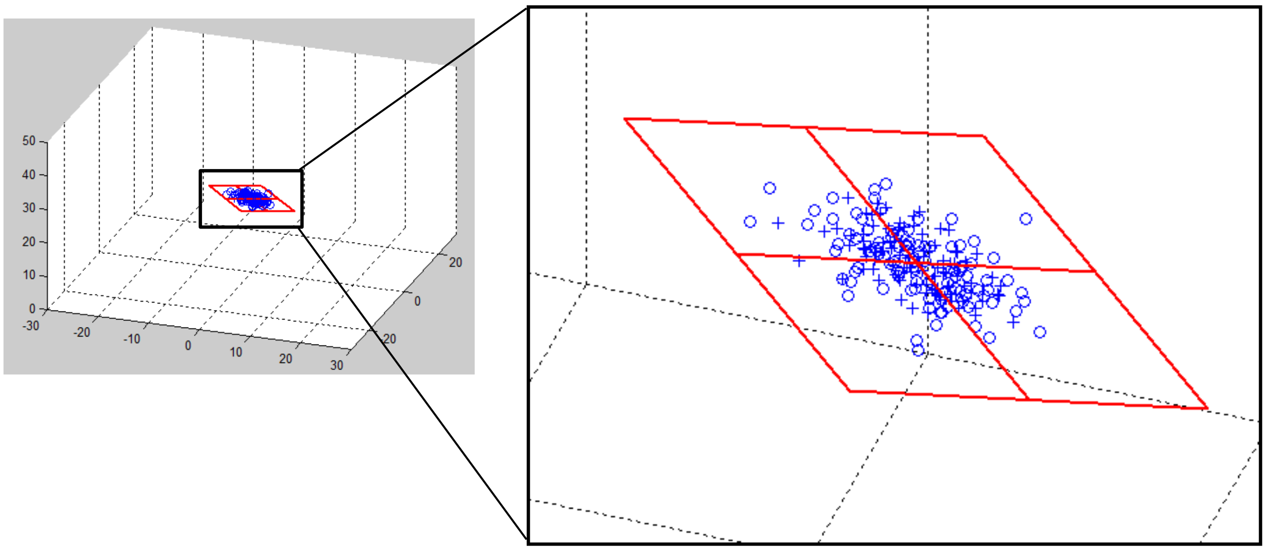


The PCA is working fine and, after projection, we can still differentiate the two families of points: the "crosses" are far away from the "circles". The PCA allows us to « visually » find segments inside the population: you can easily « see the segments » after a projection on the PCA axises.

What happens if we use some other directions than the PCA directions to perform dimensionality reduction? Let's project all the 3D points on a randomly chosen plane. The points after projection are in Blue. The projection direction in itself is in green. Note that the green lines are very « long »: it means that "this is a very bad projection" because the position of the points before and after projection is very different.



After projection, we cannot differentiate the two families of points anymore: the "crosses" are now mixed with the "circles":



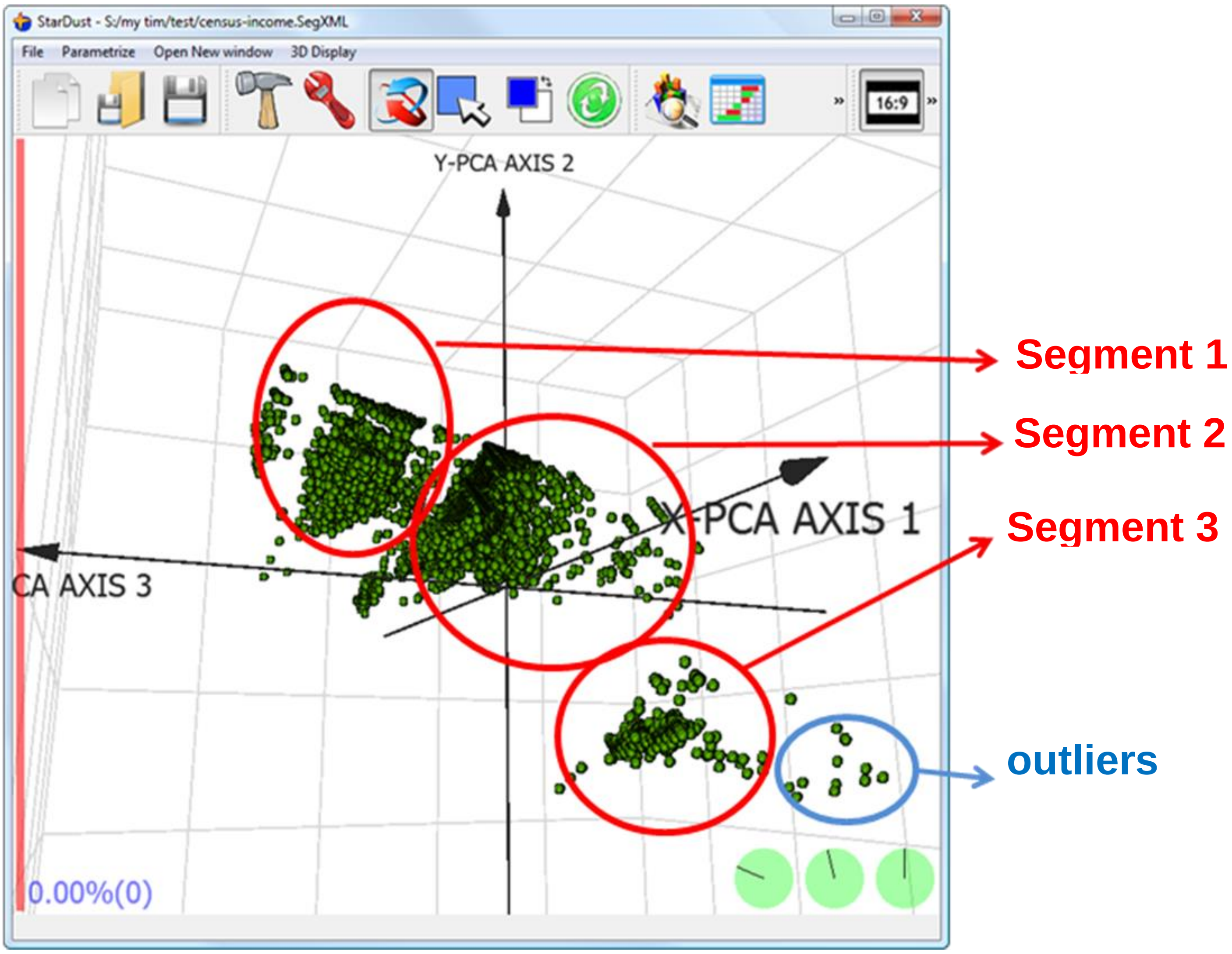
This is a really bad « dimensionality reduction » because we lost so much information that we are not able to see any segments anymore

2.2.4. Going back to the Census-Income dataset

Let's go back to the « Census-Income» database. For each individual, we have 7 variables (7 numbers):

1. "number of people working for an employer"
2. "capital losses"
3. "dividends from stocks"
4. "wages per hour "
5. "age"
6. "capital gain"
7. "weeks worked in year"

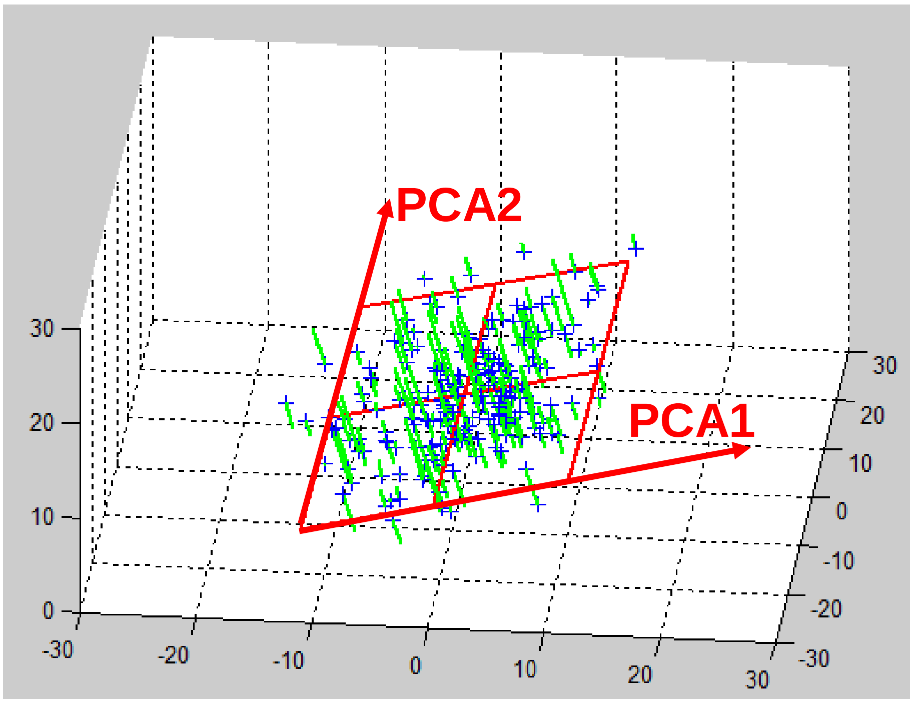
Thus, each individual is a 7D (seven dimensions) point. We will « project » these 7D points into 3 Dimension using the PCA technique. This is the result of the projection that is automatically delivered by Stardust:



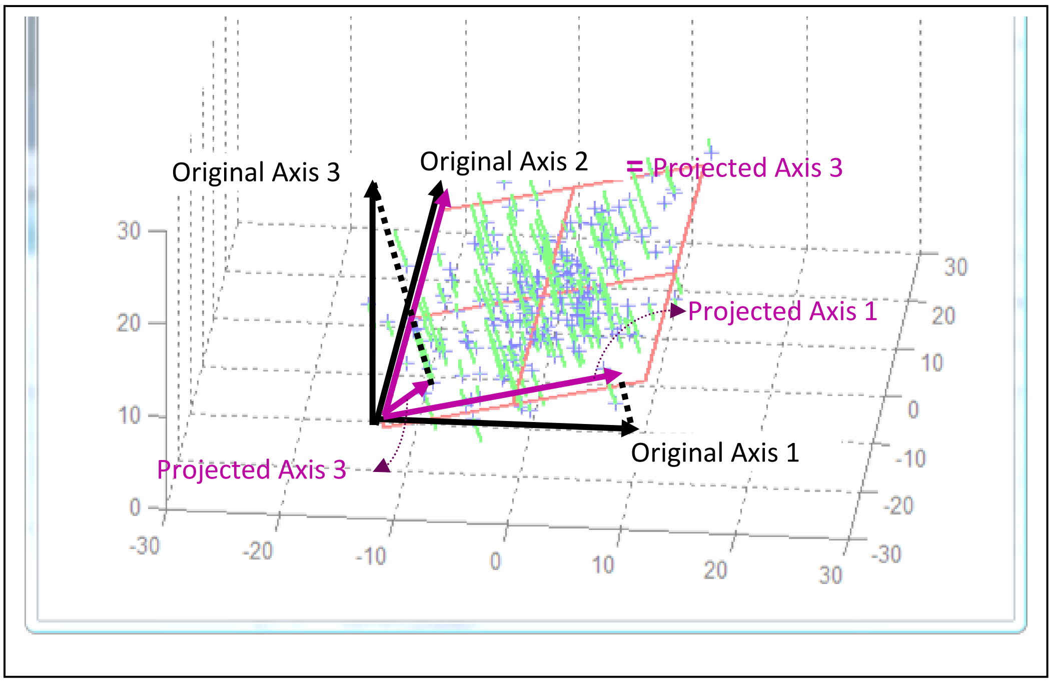
Using Stardust, you can directly « see » for yourself the 3 segments inside the dataset. By simple visual inspection, you discover existing, natural segments inside your population. You can even see some outliers.

2.2.5. Projecting the original axises inside the reduced space.

Let's take another small example:



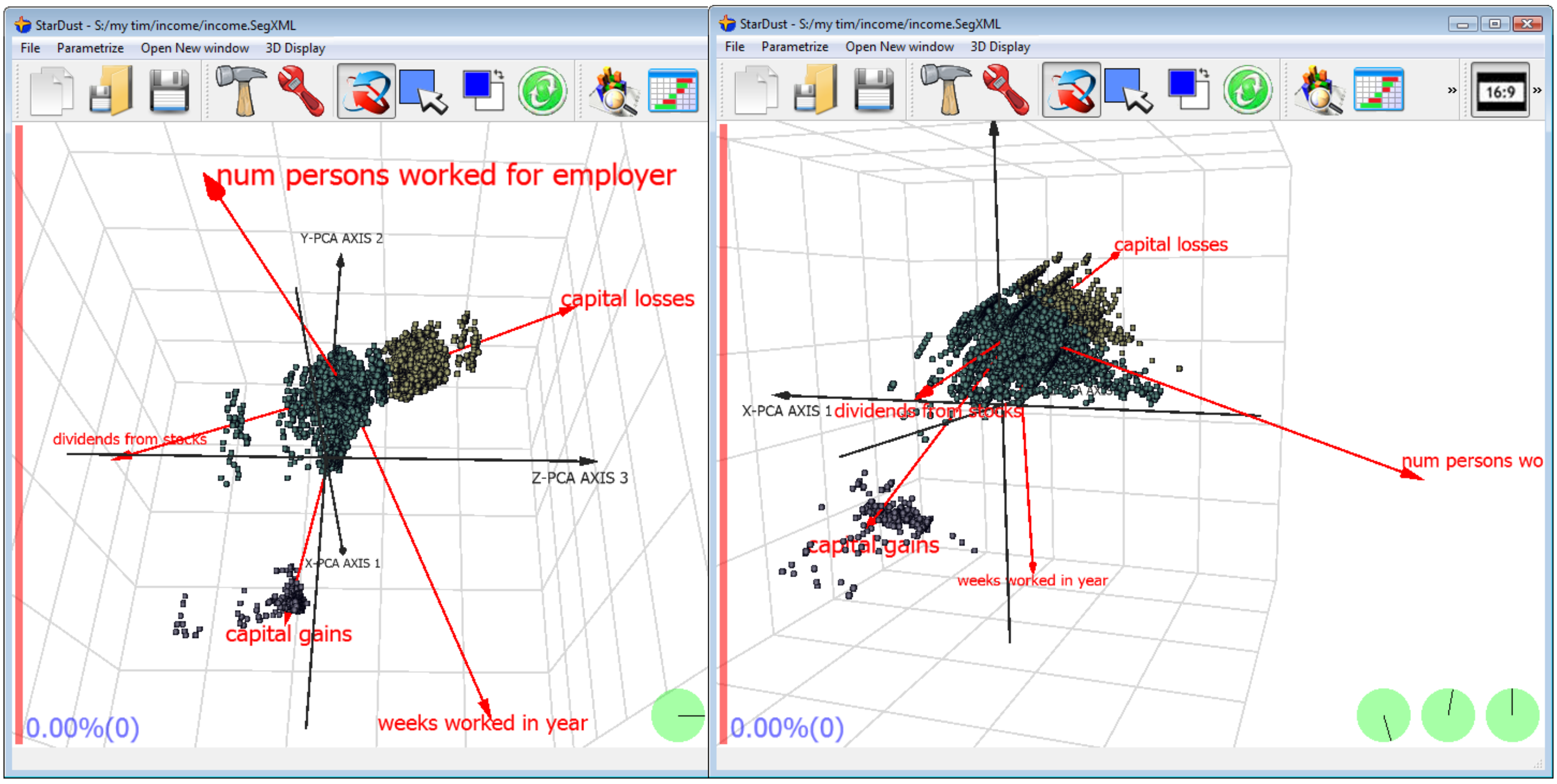
Here is an illustration on how the "original axises" of the original space BEFORE dimensionality reduction are "projected" inside the reduced space defined by the PCA1 and PCA2 axises:



The projection of one original axis inside the spaced defined by the first PCA axises is a "short" vector if the original direction is not preserved very-well after projection (see for example on the above figure, the "original axis 3" which is nearly perpendicular to the PCA plane). The quality of the representation of an original axis is thus proportional to the length of the corresponding vector in the reduced space.

Of course, only the axises that were included inside the "PCA-dimentionality-reduction-computations" can be projected inside the reduced space. The other axis can't be represented.

The projected axis that "points" in the same direction (inside the reduced space) are representing (positively or negatively) correlated concepts. For example, inside the census-income, we can observe the following:



The variables "*capital losses*", "*capital gains*" and "*dividends from stocks*" are more or less in the same alignment (the sign of the arrows are not the same but the direction is). This means that all these variables are representing more or less the same concept.

There are two more additional concepts that are "perpendicular": the "*num persons worked for empoyer*" and the "*weeks worked in year*". Thus, you can clearly see the 3 main concepts inside the database: "*capital losses*", "*num persons worked for empoyer*" and the "*weeks worked in year*".

You can also see (on the chart on the left) that the yellow segment is mainly composed of people that have high "*capital losses*".

You can also see (on the chart on the right) that the purple segment is mainly composed of people that have high "*capital gains*".

2.2. Distance definition

Now that we have seen how to visualize the segments, we will proceed to the next step that is "create a segmentation". Segmentation is the action of « grouping together », inside the same segment, customers that are « similar » or « close ».

With StarDust, you can:

1. Create « Segmentation Models » that automatically assign individuals to the right segment when receiving a new database.
2. Easily describe each segment from a business perspective thanks to a large set of intuitive charts that are generated automatically for you.

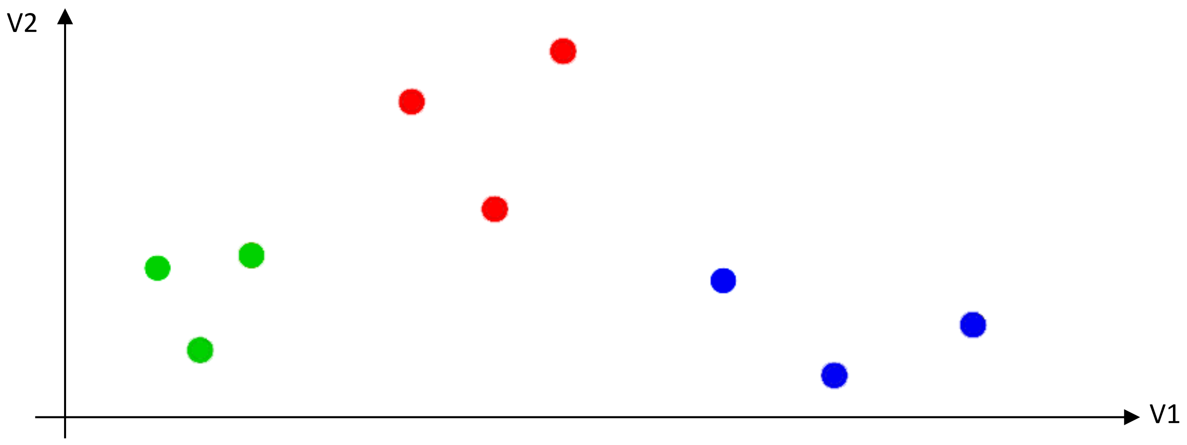
We will now see how to create a « Segmentation Model ».

2.2.1. The K-Means Algorithm

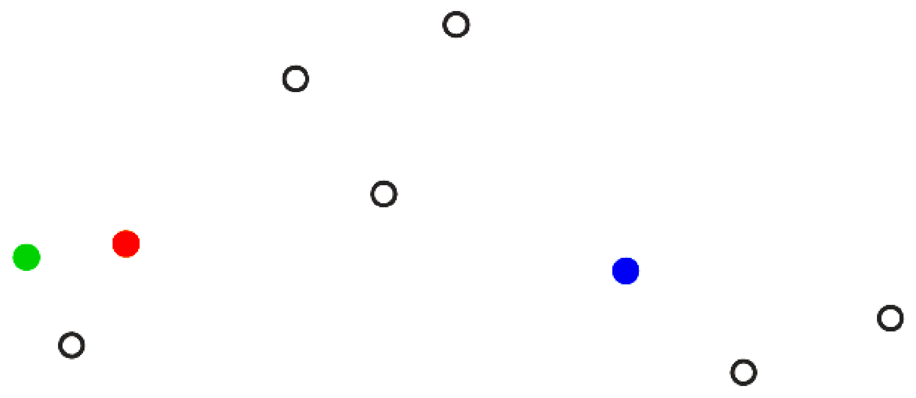
Let's start with a small example where each customer is represented by only 2 values (v1 and v2): each customer is a 2D point. A segmentation model is a model that assigns a color to each of these points. Each color represents a different segment. Here we have represented a segmentation model that finds 3 different segments in the dataset:

- Segment one is composed by the customers in green
- Segment two is composed by the customers in red
- Segment three is composed by the customers in blue

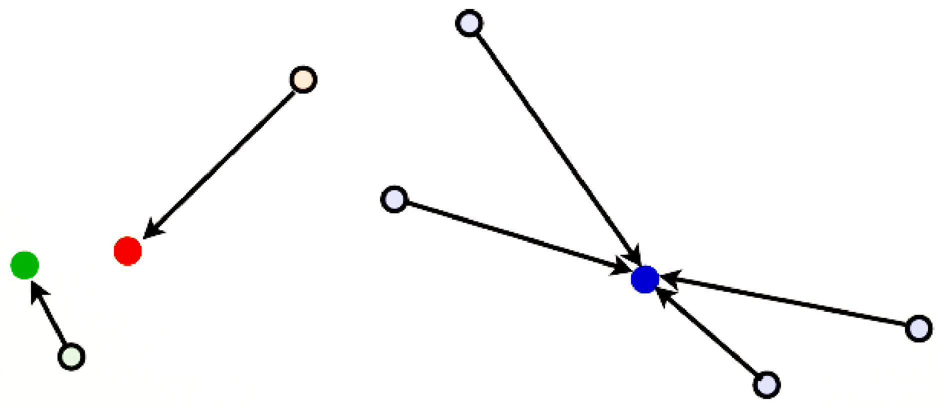
See illustration:



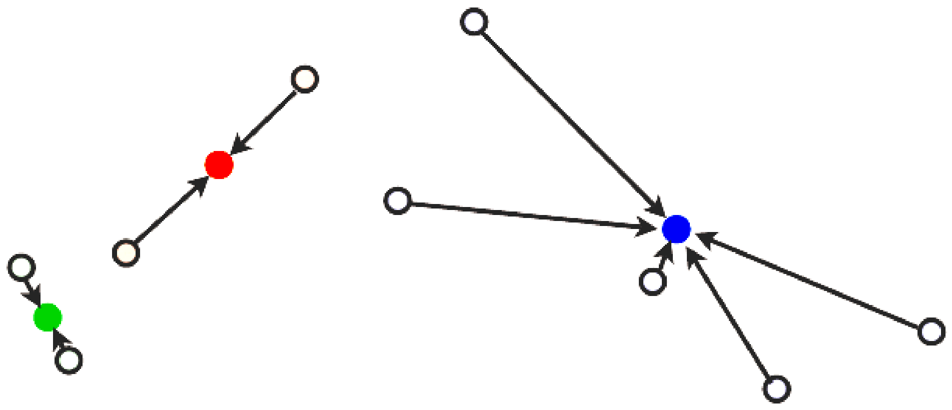
To construct a segmentation model, Stardust uses (amongst other algorithms) the K-means algorithm. The K-means algorithm works this way: first we assign randomly one customer to each segment:



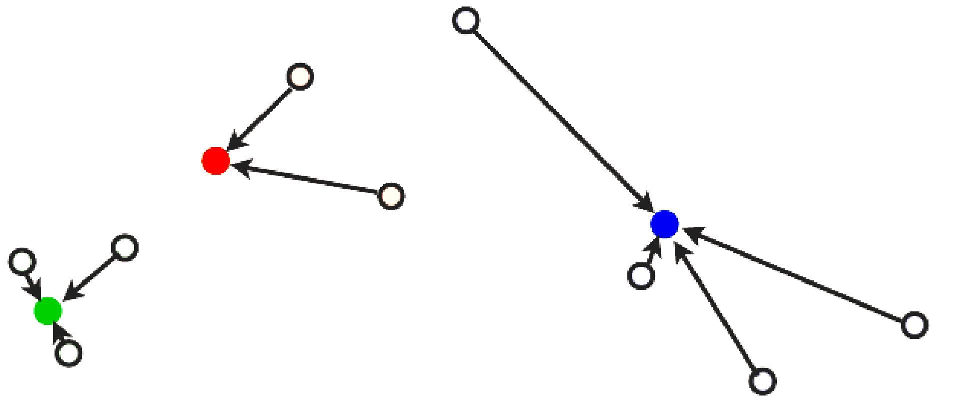
Thereafter, we assign all the customers to the nearest segment:



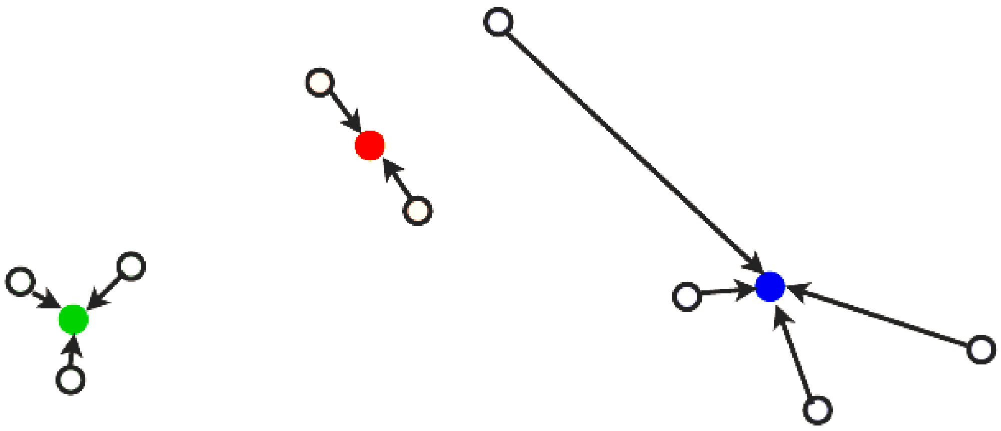
After, we re-compute the center of each segment.



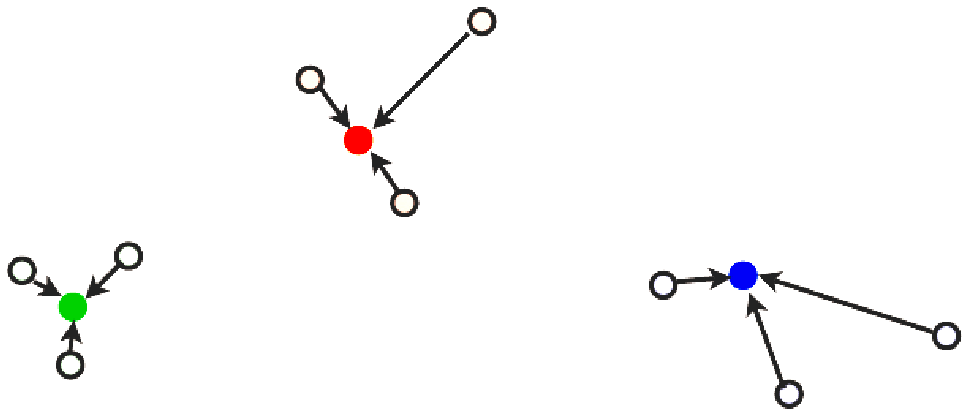
… and we repeat: We assign, once again, each point to the nearest “center”.



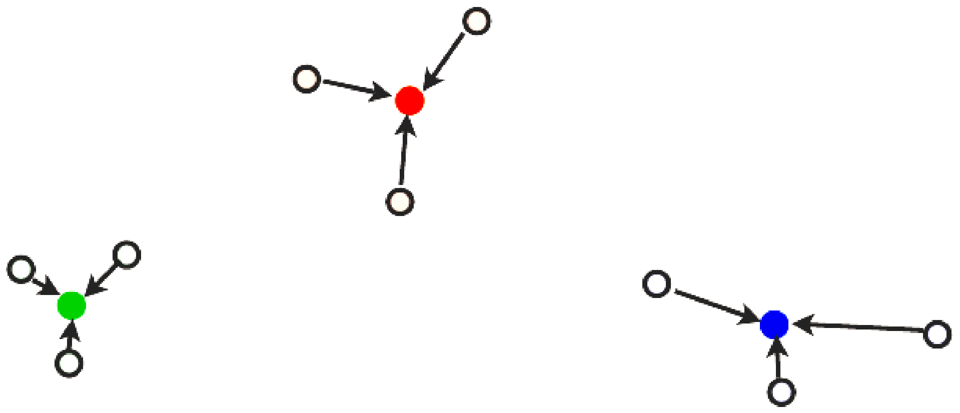
… and we re-compute the optimal centers.



… and we repeat:



… and we repeat until the segmentation no longer changes:



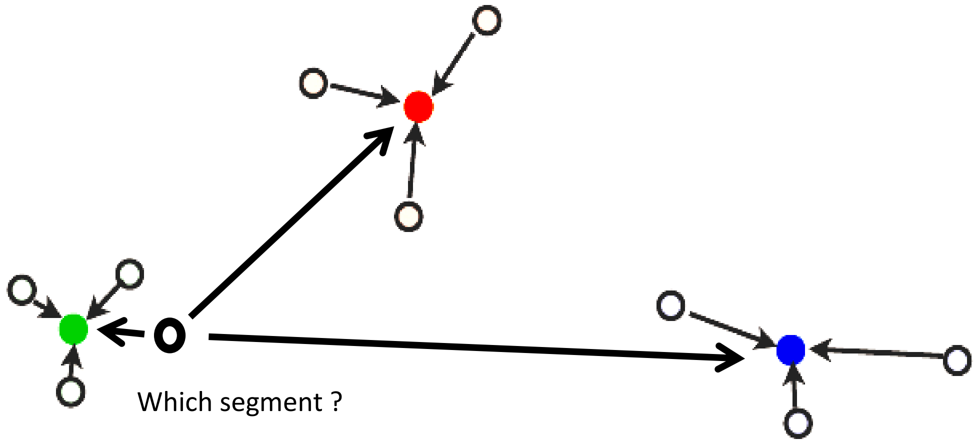
Please Note that the very first step of the K-means algorithm is to assign randomly one customer to each segment (and after iterate).

These "special" customers are named "seeds". Usually, different "seeds" will give different (but hopefully close) segmentations. Thus, there is no unique segmentation: Depending on the "seeds" you will usually find different segments.

Amongst all the different segmentations proposed by TIMi, you should choose the segmentation that has the best interpretation from a business-point-of-view. TIMi offers you many different intuitive charts that allow you, in a few mouse clicks, to interpret easily your segmentation from a business-perspective, and therefore, to easily select the segmentation that is the best for you (and always from a business-perspective!).

2.2.2. Automatic assignment of new customers to segments

Let's see what happens when a new customer arrives… to know in which segment this customer belongs, we have to compute the distance between this new customer and the center of each segment. In this example, the closest segment is the green one:



So, this new customer belongs to the green segment.

As you can see, the K-Means algorithm uses « distances » extensively. So we need to define good « distances ».

2.2.3. All the type of distances available inside StarDust

A complete « segmentation model » is composed of:

- A list of « centers » of each segment.
- A definition of the « distance » used to compute the distance to each center.

Let's have a closer look at the distances available inside StarDust.

With StarDust, you can mix several types of distances:

- The Standard Euclidean Distance
- The Pearson Distance (also named Cosine distance)

Each distance can be used inside different kinds of space:

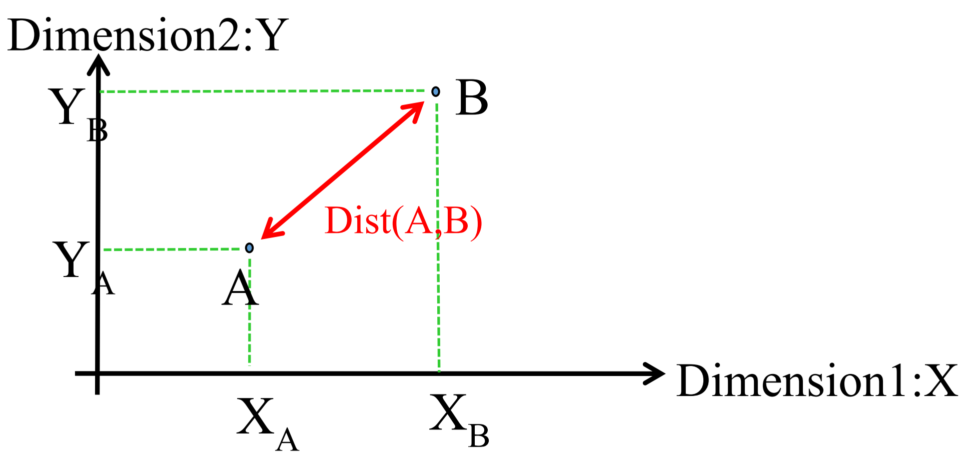
- The Original Space of the variable
- The Quantile Space
- The Space defined by the first axises of the PCA.
- …

The pearson (or Cosine) distance is very useful when using Stardust to do text mining. When doing “textmining”, each row of a dataset represents a text-document. The pearson (or Cosine) distance is commonly used to define distances between text-documents.

As a reminder, the Euclidean distance between the point A and B is expressed this way:

$$
\begin{aligned}
Dist(A,B) &= \sqrt{(X_A - X_B)^2 + (Y_A - Y_B)^2} \\
&= \sqrt{(X_A - X_B)^2 + (Y_A - Y_B)^2 + (Z_A - Z_B)^2 + \ldots} \\
&= \sqrt{(D1_A - D1_B)^2 + (D2_A - D2_B)^2 + (D3_A - D3_B)^2 + \ldots + (Dn_A - Dn_B)^2}
\end{aligned}
$$

It is simply the length of the red line here:



2.2.4. A first example about the importance of variable "normalization"

Let's now take a more real example. Let's assume that we have a dataset with 2 columns that are the 2 dimensions used inside the Euclidean Distance used to construct the segments. These 2 columns are:

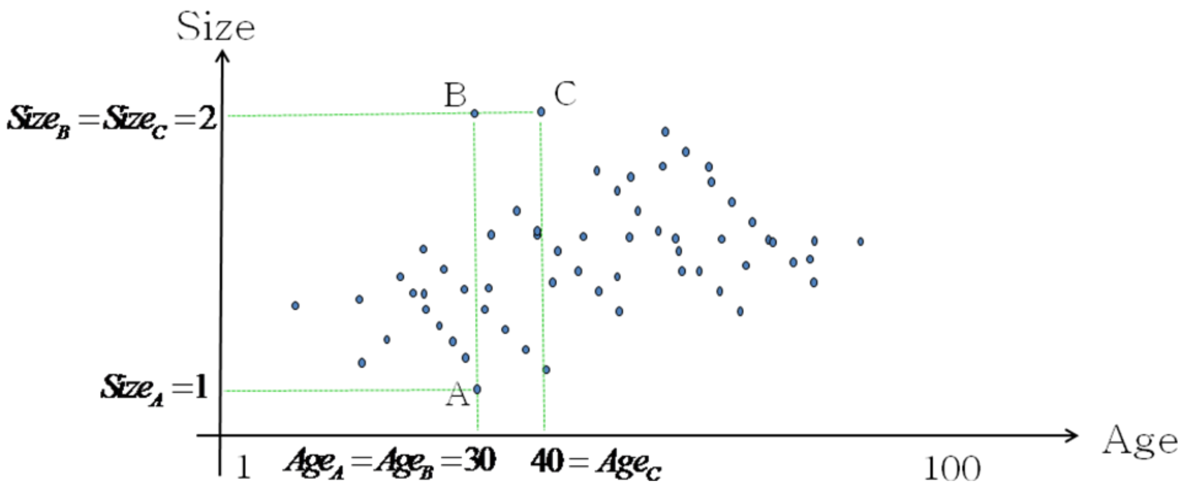
1. Age expressed in Year $0 < Age < 100$
2. Size expressed in Meter $0 < Size < 2$

The classical Euclidian Distance is: $Dist(A,B) = \sqrt{(Age_A - Age_B)^2 + (Size_A - Size_B)^2}$

This distance, however, is not very good.

Let's take an example. Let's compute the distance between 3 individuals: A, B and C.

A and B are separated by one meter of size. So the distance between A and B is one.
In opposition, B and C are separated by 10 years. So the distance between B and C is ten.

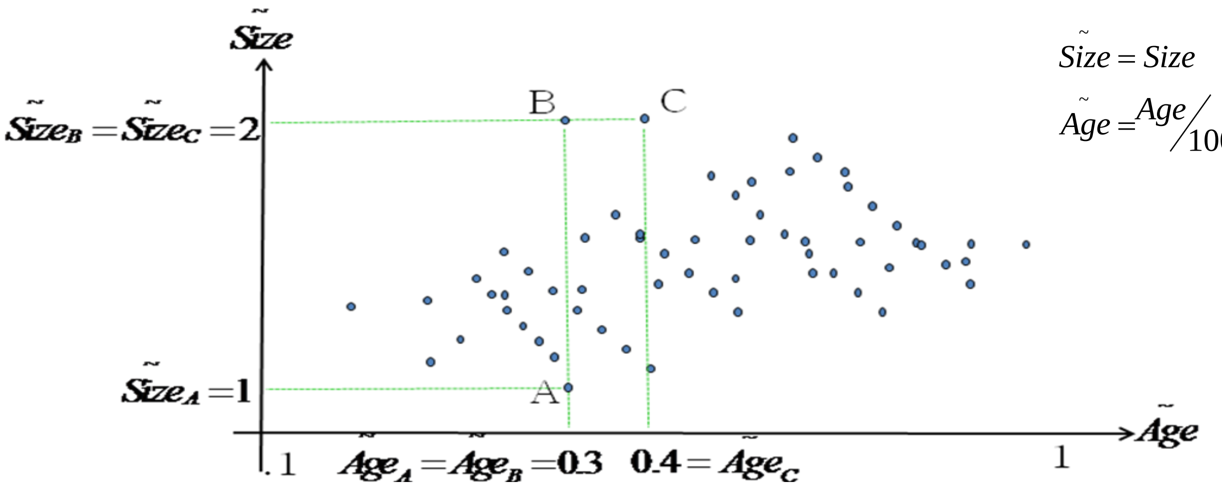


You can directly see that there is a problem here. Graphically, we can see that B is closer to C than A but the Euclidean Distance "says" the opposite:

$$Dist(A,B) = \sqrt{(Age_A - Age_B)^2 + (Size_A - Size_B)^2} = 1$$
$$Dist(B,C) = \sqrt{(Age_B - Age_C)^2 + (Size_B - Size_C)^2} = 10$$

So, there is a problem in the Euclidean Distance. The problem comes from the fact that the "age axis" goes from one to one hundred… and the "size axis" only goes from one to two: The range of the two axises must be similar in order to have a meaningful distance.

To correct this unfortunate situation, we will define a “normalized-distance” where each axis has been normalized to obtain more or less the same range. For the “normalized-distance”, the “age” has been divided by one hundred and the size stays the same:



$$\tilde{Size} = Size$$
$$\tilde{Age} = {Age}/{100}$$

Using this new “normalized-distance”, now, we have:

$$Normalized - Dist(A,B) = \sqrt{(\tilde{Age}_A - \tilde{Age}_B)^2 + (\tilde{Size}_A - \tilde{Size}_B)^2} = 1$$
$$Normalized - Dist(B,C) = \sqrt{(\tilde{Age}_B - \tilde{Age}_C)^2 + (\tilde{Size}_B - \tilde{Size}_C)^2} = 0.1$$

It’s now ok: B is closer to C than A.

As a conclusion: Normalization is Important! We must always use « normalized - distance» !

If we compare the “standard-distance” with the “normalized-distance”, we can see that:

$$Dist(A,B) = \sqrt{(Age_A - Age_B)^2 + (Size_A - Size_B)^2} = 1$$
$$Norm. - Dist(A,B) = \sqrt{(\tilde{Age}_A - \tilde{Age}_B)^2 + (\tilde{Size}_A - \tilde{Size}_B)^2}$$
$$= \sqrt{({Age_A}/{100} - {Age_B}/{100})^2 + (Size_A - Size_B)^2}$$
$$= \sqrt{\boxed{\frac{1}{10000}}(Age_A - Age_B)^2 + \boxed{1}(Size_A - Size_B)^2}$$

$$\tilde{Size} = Size$$
$$\tilde{Age} = {Age}/{100}$$

The equations defining these two distances are very close. There are nearly the same. The only difference is that, for “normalized-distance”, you have normalization factors ($1/10000$ and 1) in front of each term of the sum. Keep that in mind.

2.2.5. A second example about variables representing the same concept

Let's take another example: We have a dataset with 3 columns. These 3 columns are:

- The Age expressed in Days (*AgeD*)
- The Age expressed in Months (*AgeM*)
- The Size (*Size*)

Using these 3 axises, we can define this Euclidian Distance:

$$Dist(A,B)=\sqrt{(\widetilde{AgeD}_A-\widetilde{AgeD}_B)^2+(\widetilde{AgeM}_A-\widetilde{AgeM}_B)^2+(\widetilde{Size}_A-\widetilde{Size}_B)^2}$$

(Note that we are using « normalized » distance)

We can obtain the "*age expressed in days*"(*AgeD*) of an individual by multiplying by 31 the age of the same individual expressed in months. When we are using this relationship inside the equation of the Euclidean distance we arrive to this final equation.

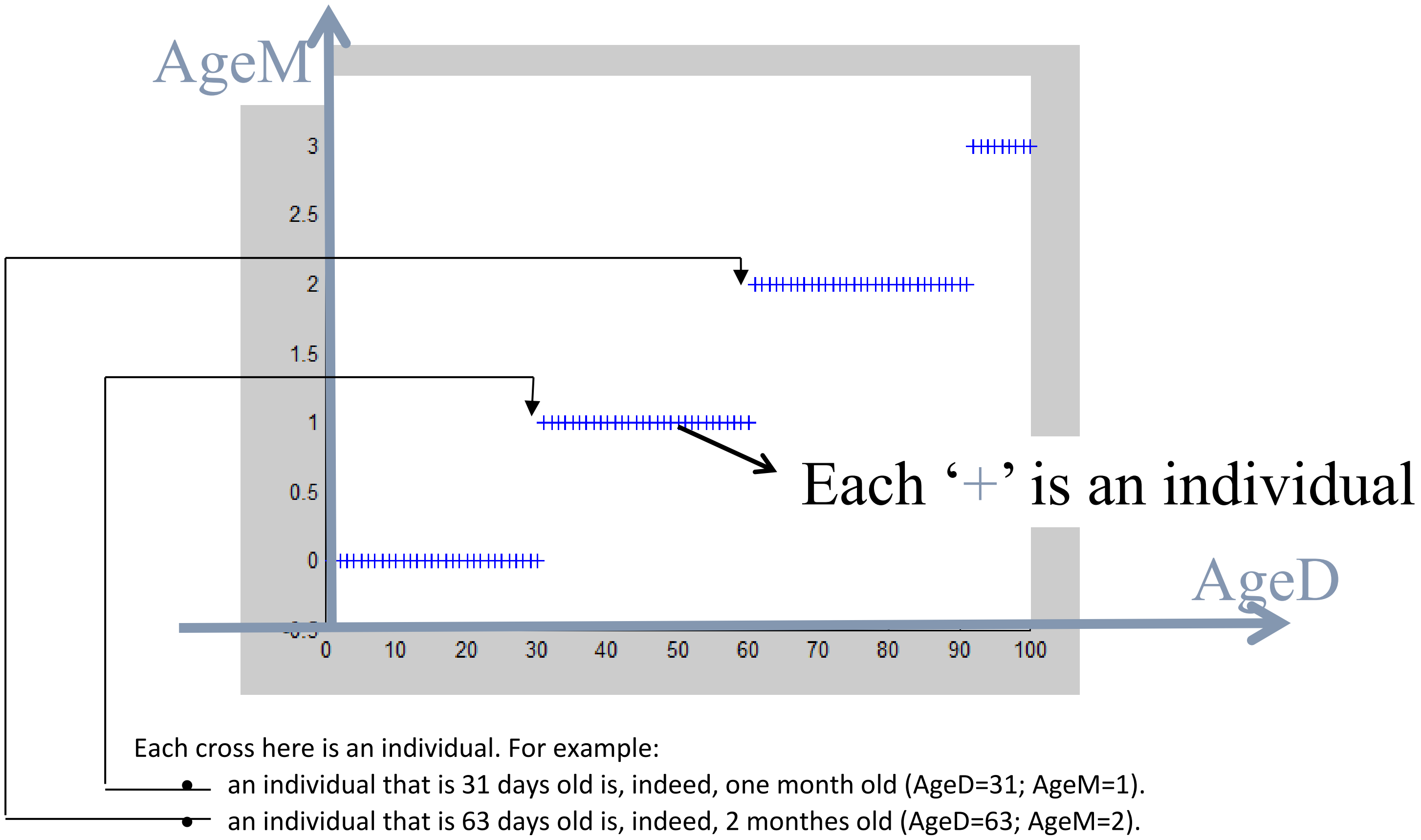
$$\widetilde{AgeD}\approx 31\,\widetilde{AgeM}$$

$$\begin{aligned}
Dist(X,Y)&=\sqrt{(\boxed{\widetilde{AgeD}}_A-\widetilde{AgeD}_B)^2+(\widetilde{AgeM}_A-\widetilde{AgeM}_B)^2+(\widetilde{Size}_A-\widetilde{Size}_B)^2}\\
&\approx\sqrt{(\boxed{31\widetilde{AgeM}}_A-31\widetilde{AgeM}_B)^2+(\widetilde{AgeM}_A-\widetilde{AgeM}_B)^2+(\widetilde{Size}_A-\widetilde{Size}_B)^2}\\
&=\sqrt{(31^2+1)(\widetilde{AgeM}_A-\widetilde{AgeM}_B)^2+(\widetilde{Size}_A-\widetilde{Size}_B)^2}\\
&=\sqrt{\boxed{962}\,(\widetilde{AgeM}_A-\widetilde{AgeM}_B)^2+(\widetilde{Size}_A-\widetilde{Size}_B)^2}
\end{aligned}$$

Normalization is lost !

There is something very wrong about this final equation: We are changing the normalization factor. The « weight » given to the « age concept » is wrongly multiplied by more than 900. The « normalization » of the variables is lost when two variables (here: the "age expressed in days" and the "age expressed in months") are representing the same concept.

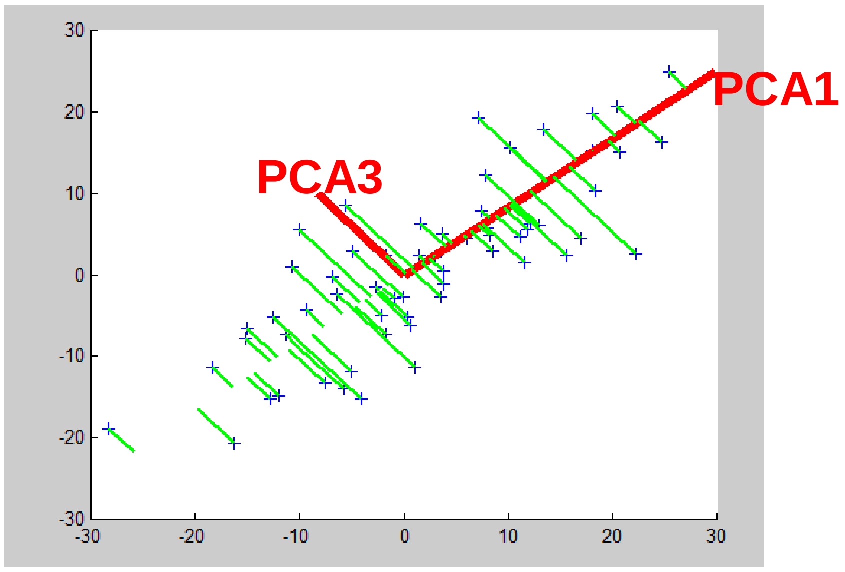
Let's represent graphically the "*age expressed in days*"(*AgeD*) and the "age *expressed in months*"(*AgeM*):



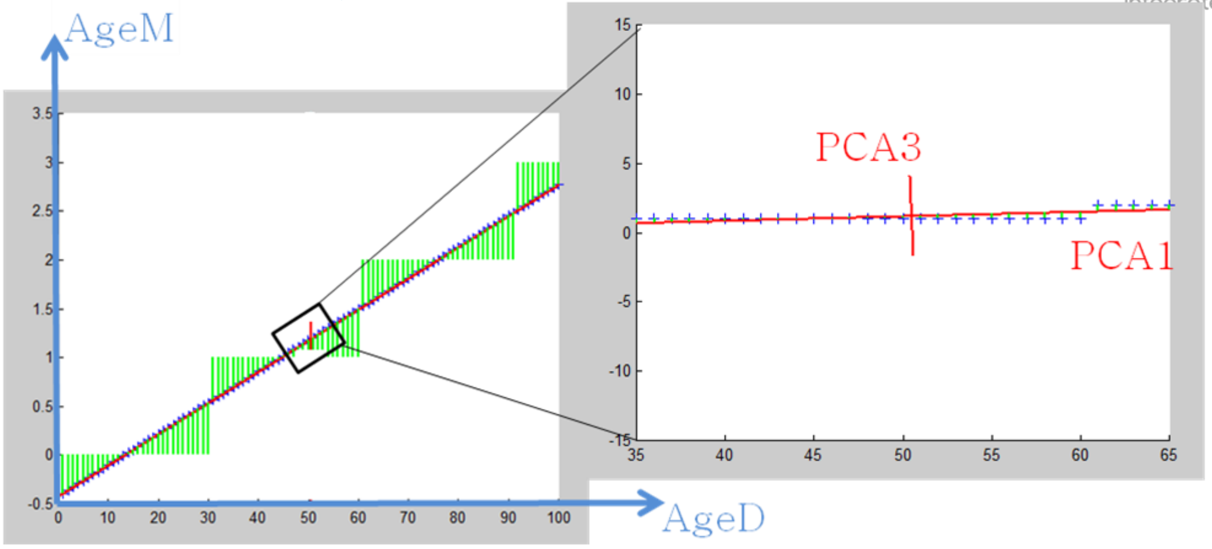
Each cross here is an individual. For example:

- an individual that is 31 days old is, indeed, one month old (AgeD=31; AgeM=1).
- an individual that is 63 days old is, indeed, 2 monthes old (AgeD=63; AgeM=2).

You may remember this chart from the previous section about "dimensionality reduction":



On this chart we computed the 2 PCA axises in order to reduce the dimension. We will do the same on this new population:



We obtain 2 PCA axises named "PCA1" and "PCA3". If we compute the coordinates of each individual inside the new coordinate system defined by the "PCA1" and "PCA3" axises. We will notice that:

- OLD individuals have a HIGH value along the "PCA1" axis
- YOUNG individuals have a LOW value along the "PCA1" axis.

So the PCA1 axis represents the "age" concept. The PCA3 direction represents the small error in converting the age from months to days. The values along the PCA3 are completely uninteresting so we will simply never use them.

So, instead of defining the distance on the original axises: "*age expressed in days*"(*AgeD*) , "*age expressed in months*"(*AgeM*) and "*size*"(*SIZE*):

$$Dist(A,B)=\sqrt{\underbrace{(\tilde{AgeD}_A-\tilde{AgeD}_B)^2+(\tilde{AgeM}_A-\tilde{AgeM}_B)^2}+\underbrace{(\tilde{Size}_A-\tilde{Size}_B)^2}}$$

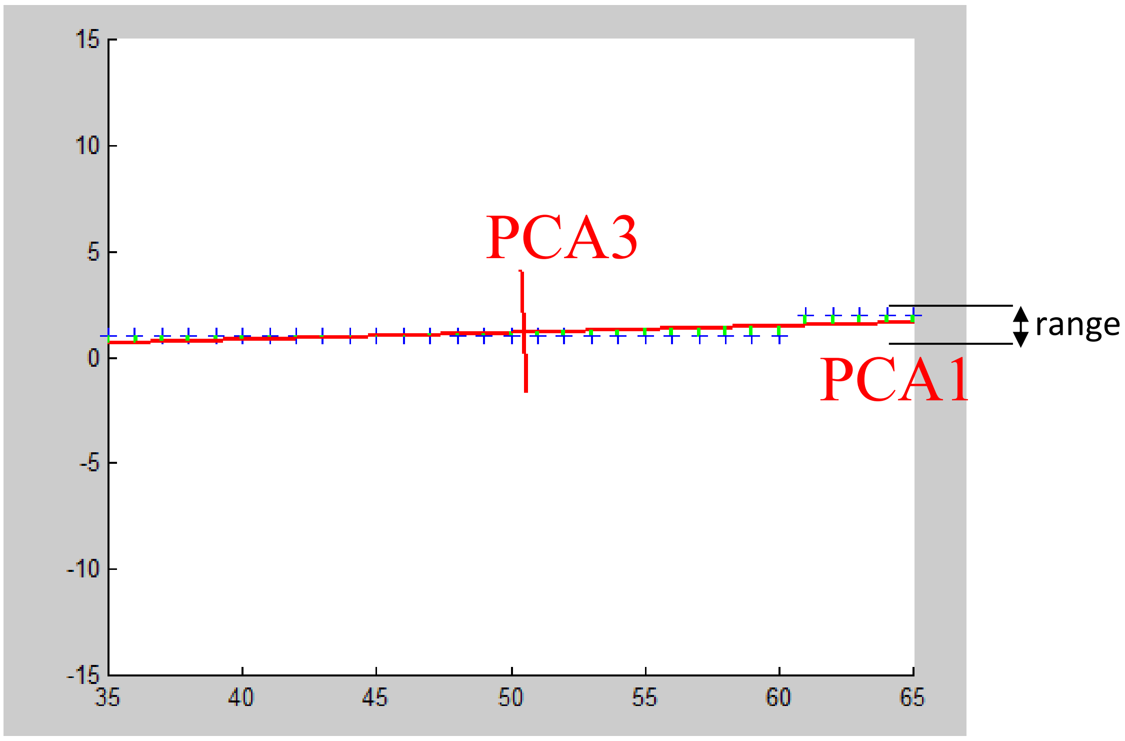
… we will now express the distance inside the PCA axises:

- PCA1 that represents the "age" concept
- PCA2 that represents the "size" concept
- PCA3 that represents the small error in converting the age from months to days.

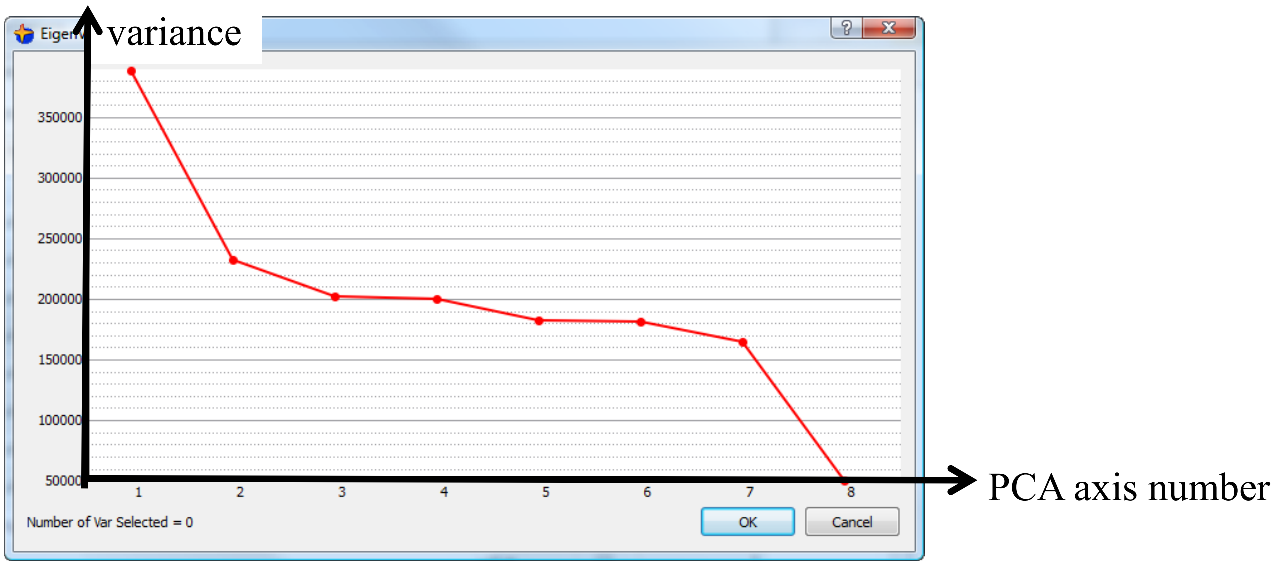
This gives us:

$$Dist(A,B)=\sqrt{\underbrace{(\tilde{PCA1}_A-\tilde{PCA1}_B)^2}_{\text{« Age » concept}}+\underbrace{(\tilde{PCA2}_A-\tilde{PCA2}_B)^2}_{\text{« Size » concept}}+\underbrace{\cancel{(\tilde{PCA3}_A-\tilde{PCA3}_B)^2}}_{\text{Conversion of Age from month's to days}}}$$

Note that we did not include inside this new definition of the Distance, the distance along the « PCA3 » direction because it's completely irrelevant. **You can see that it is irrelevant because the range of values in the PCA direction is very small**. This new definition of the distance is the best one possible. You can see the range along the « PCA3 » axis here:



This range is so small that it is totally useless and does not represent anything: it’s noise. So, it’s important to know the range of value along each PCA direction because you must drop from the distance-definition the PCA directions that have small variance. StarDust automatically displays the range of values along all PCA axis: In this graphic (from StarDust), we see that the range along the PCA8 axis is very small:



So, a good distance definition should only include terms from PCA1 to PCA7 and NOT PCA8: For example:

$$Dist(A,B) = \sqrt{\sum_{i=1}^{7} (\tilde{PCAi}_A - \tilde{PCAi}_B)^2}$$

2.3. Summary

To summarize, PCA is used to:

1. do dimensionality reduction.
2. create good « distances-definition ». We need a correct « distances-definition » so that the K-Mean algorithm works properly and delivers good segmentation models. Without a PCA, the classical « distances-definitions » base on Euclidean-distances are giving random weights to each concept, thereby producing a totally random and arbitrary segmentation.

As you can see, a correct methodology for a good segmentation analysis relies heavily on PCA analysis. Stardust is the only segmentation tool that is directly offering you, in a few mouse clicks, a complete « PCA analysis » in all the steps of the segmentation analysis.

We will now see how these two concepts are used inside StarDust to:

- Produce a segmentation model
- Describe each segments from a business perspective

3. TIMi Installation

To summarize, the installation procedure for the portable installation of TIMi is:

- Download the 3 files from:
  - For a 64-bit PC:
    - http://download.timi.eu/TIMi64_Full/unzip.exe
    - http://download.timi.eu/TIMi64_Full/unzip_TIMi64_Portable_in_C_SOFT.bat
    - http://download.timi.eu/TIMi64_Full/TIMiPortableFull_x64.zip
  - For a 32-bit PC:
    - http://download.timi.eu/TIMi32_Full/unzip.exe
    - http://download.timi.eu/TIMi32_Full/unzip_TIMi32_Portable_in_C_SOFT.bat
    - http://download.timi.eu/TIMi32_Full/TIMiPortableFull_win32.zip

- Place the 4 downloaded files inside the same directory and execute (double-click) the file "unzip_TIMi??_Portable_in_C_SOFT.bat". This will install TIMi inside the (default) directory:
  C:\soft\TIMiPortable
- **Optional (admin required):** Run the executable:
  "<root_of_install_dir>\Add_TIMi_ShortCut_On_**All**_Desktops.exe"
  This will add 2 shortcuts (one shortcut to run TIMi and one shortcuts to run Anatella) on **all** the desktops from all the users from the server.
- Follow the procedure described in the document TIMi_License_Guide_en.pdf to get the Serial Number for TIMi

You'll find mode details about the different options for the installation of TIMi inside this document:
http://download.timi.eu/docs/TIMi_Deployment.pdf

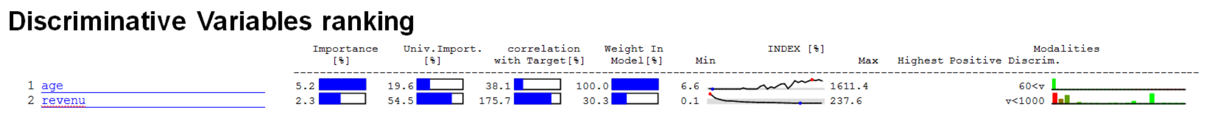
4. How to start StarDust ?

4.1. Introduction

StarDust can be used BEFORE and AFTER constructing a predictive model.

You can use StarDust BEFORE constructing a predictive model to find the different segments of customer inside your database. Thereafter, you can easily create one predictive model per segment. In some not so rare situations, creating several "small" predictive models for each of the segments (instead of one "large" predictive model on the whole database) can lead to a higher predictive accuracy.

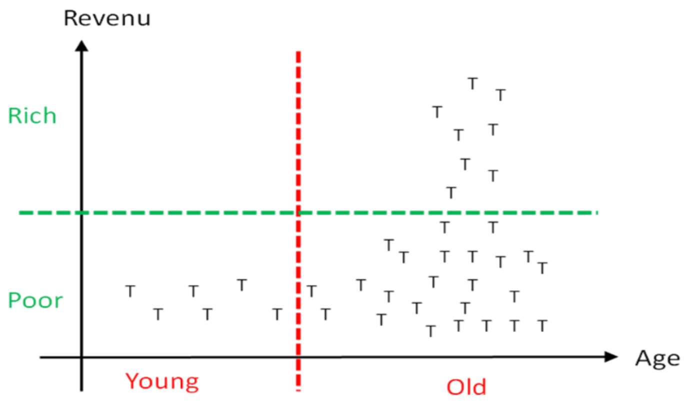
You can use StarDust AFTER constructing a predictive model to analyse in more detail your "target" group. Indeed TIMi already gives you a nice MSWord reports that already gives you detailed information about the profile of your target but, on some not so rare situation, you can still obtain a better understanding of the profile of your target using StarDust. Let's consider this example: the target is composed of 2 segments: Poor Young people and Rich Old Men: You will see inside the TIM analyst report something like that:



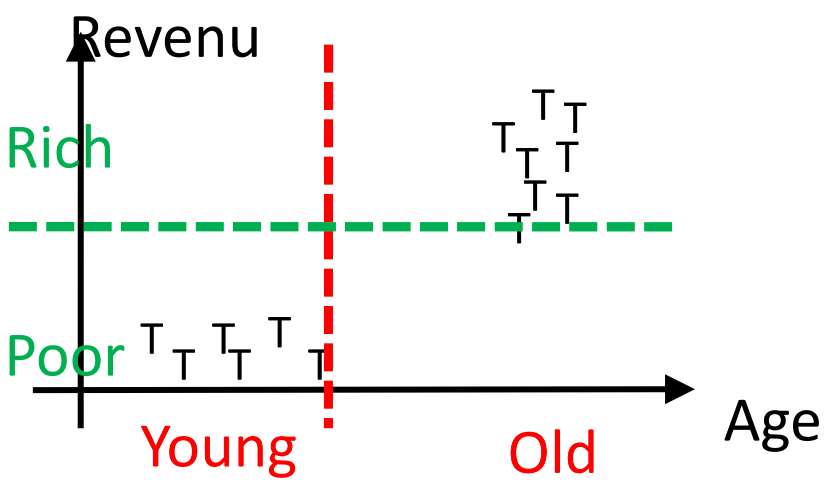
Discriminative Variables ranking

		Importance [%]	Univ.Import. [%]	correlation with Target[%]	Weight In Model[%]	INDEX [%] Min	Max	Modalities Highest Positive Discrim.
1	age	5.2	19.6	38.1	100.0	6.6	1611.4	60<v
2	revenu	2.3	54.5	175.7	30.3	0.1	237.6	v<1000

This extract from a TIMi Modeler analyst reports give you the (false) impression that your target is composed of old AND poor men. It really looks like your "target group" can be (falsely) represented this way:



... but in reality your "Target Group" can be correctly represented this way (note that there are two segments):



In real life, the situation depicted inside this example almost never exists. But, still, with StarDust, you can check in a few mouse clicks the real profile of your “Target Group”.

You can use StarDust to analyse your “Target Group” and find such, more precise, “insights” about your customers.

4.2. Using StarDust BEFORE a predictive analysis

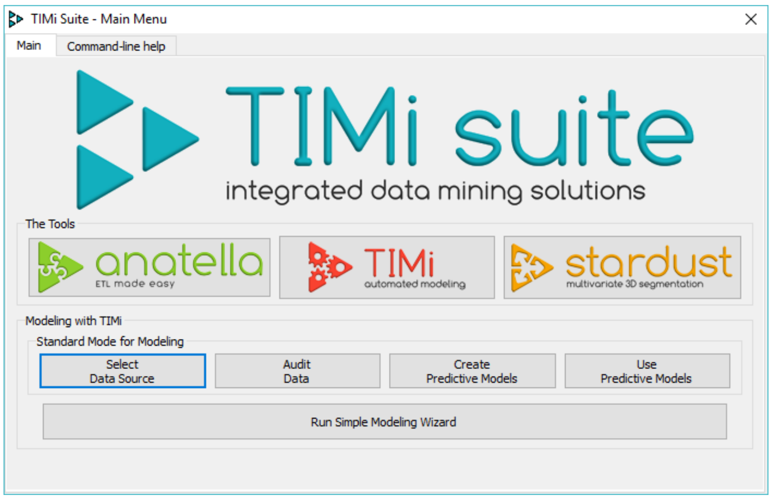
Let’s start to analyse our dataset “census-income” with StarDust!

To use StarDust, you must have a generated a “.DescXML” file about your dataset. This section demonstrates how to do that.

After the installation process of TIMi is completed, you should have on your desktop the following icon:

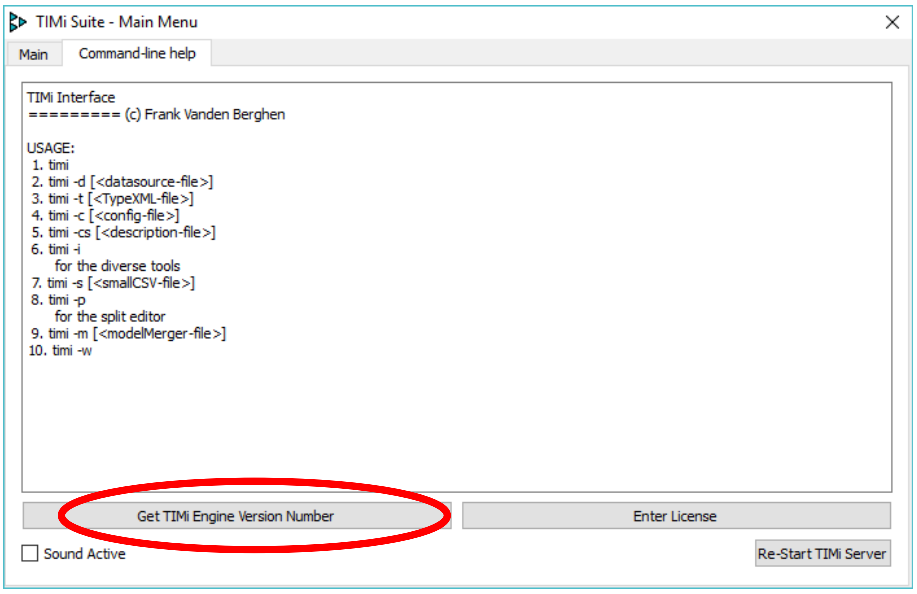
TIMi

Double-click the above TIMi icon and the “main window of TIMi” appears:



This user’s guide is for TIMi v10.22 or above. It’s strongly suggested that you update you copy of TIMi to the latest version before reading this document. All the graphics in this document are extracted from TIMi v10.22.

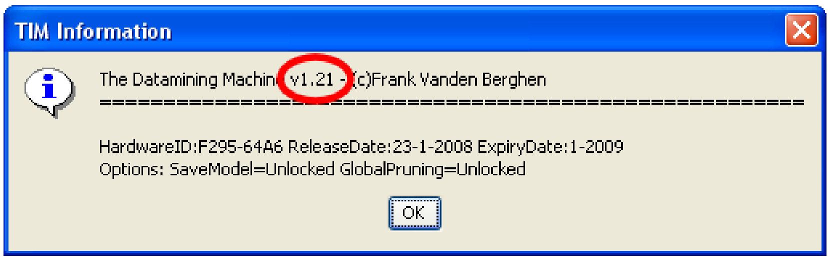
You can check your version of TIMi in the following way: inside the “main window of TIMi” click on the Command-Line help tab and on the "Get TIMi Engine Version Number" button as illustrated:



```
TIMi Interface
========= (c) Frank Vanden Berghen

USAGE:
 1. timi
 2. timi -d [<datasource-file>]
 3. timi -t [<TypeXML-file>]
 4. timi -c [<config-file>]
 5. timi -cs [<description-file>]
 6. timi -i
      for the diverse tools
 7. timi -s [<smallCSV-file>]
 8. timi -p
      for the split editor
 9. timi -m [<modelMerger-file>]
10. timi -w
```

The version number is inside the new window: see illustration:



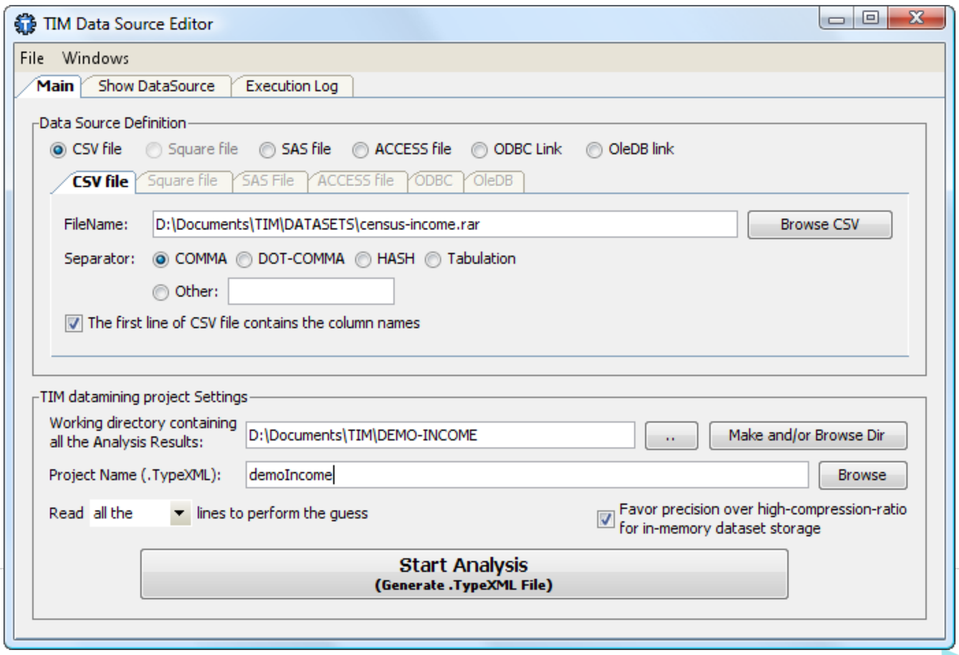
```
The Datamining Machine v1.21 - (c)Frank Vanden Berghen
===============================================================
HardwareID:F295-64A6 ReleaseDate:23-1-2008 ExpiryDate:1-2009
Options: SaveModel=Unlocked GlobalPruning=Unlocked
```

4.2.1. The DataSource Editor (Step 1)

Click on the Select Data Source button of the TIMi main window.

To have more detailed informations on how to operate the "DataSource Editor", please refer to the section 5 ("The Datasource Editor") of the "TIMi Modeler Quick User's Guide".

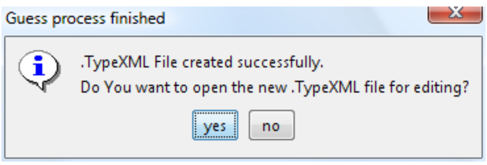
At the end, you should have something like this:



Data Source Definition
[✓ CSV file]
CSV file
FileName: D:\Documents\TIM\DATASETS\census-income.rar
Separator: [✓ COMMA] DOT-COMMA HASH Tabulation
Other:
[✓] The first line of CSV file contains the column names

TIM datamining project Settings
Working directory containing all the Analysis Results: D:\Documents\TIM\DEMO-INCOME
Project Name (.TypeXML): demoIncome
Read all the lines to perform the guess
[✓] Favor precision over high-compression-ratio for in-memory dataset storage

Click the **Start Analysis (Generate .TypeXML File)** button and wait a little. You should pretty soon see:


Guess process finished

.TypeXML File created successfully.
Do You want to open the new .TypeXML file for editing?

Click the **yes** button and the TIMi ".TypeXML file Editor" appears.

4.2.2. The Type Var file Editor (Step 2)

There are three ways to start the .TypeXML file Editor:

1. Create a new .TypeXML file using TIMi and open it directly after creation (this is normally what you just did)
2. Click the **Data Audit** button inside the main menu of TIMi.
3. Double-click on a "*.TypeXML" file inside a "file explorer" window.

Using the Type Var Editor you can specify the type of each column of the dataset. There are basically five types of columns:

a. **Value type.** Examples are: Age, Size, Cost, Price,…

   This type of column can only contain numbers and exhibits an ordering property. For example, a six-year old boy is younger than a twelve-year old boy and a twelve-year old boy is younger than an eighteen-year old boy. There is an order. So a column containing the age of a person should be of type "value". On the contrary, a column containing the zip code of the house of a person should be of type "nominal" because the zip code 1050 is NOT smaller or bigger than the zip code 1210. There is no order inside the zip code.

b. **Binary type.** Examples are: isMale, isForeigner,…

   This type of column can only contain a "true/false", "yes/no" semantic. These columns contain only two modalities (T/F) or three modalities (T/F/missing).

c. **Nominal type.** Examples are CarLabel, Region,…

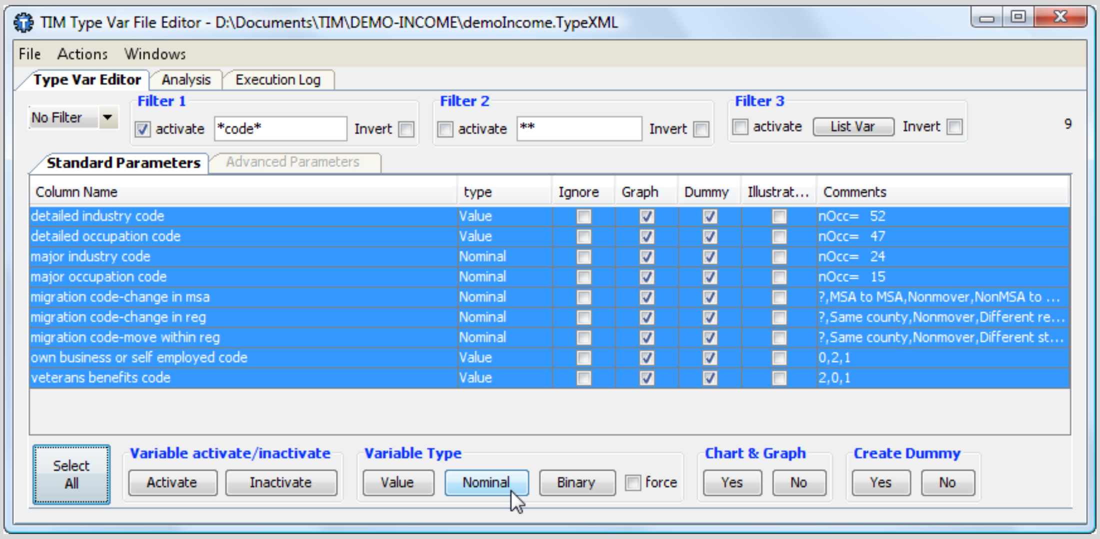
   This type of column contains anything that is not "value type" or "binary type".

d. **Target type.** What is the "target column"?

   This is type of column is only useful with doing "predictive" analysis. Since we are performing a "Segmentation" analysis, we won't have any column of this type.

e. **Key type.** What is the "primary key column"?

We will to set to the "nominal" type all the columns that contain "code":
1. Enter "*code*" inside the Filter 1.
2. Click the Select All button and then the To Nominal button.
See illustration below:



TIM Type Var File Editor - D:\Documents\TIM\DEMO-INCOME\demoIncome.TypeXML

File Actions Windows

Type Var Editor | Analysis | Execution Log

No Filter — Filter 1: ☑ activate *code* Invert ☐ — Filter 2: ☐ activate ** Invert ☐ — Filter 3: ☐ activate List Var Invert ☐ — 9

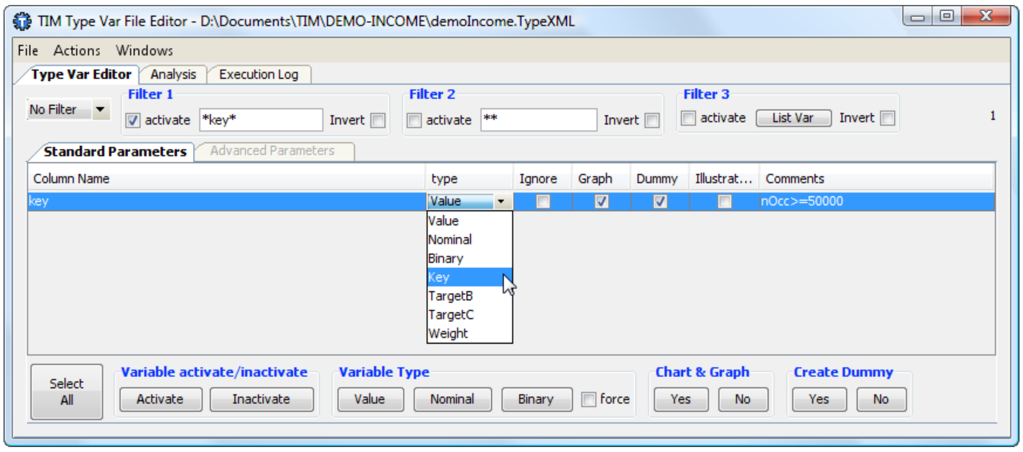
Standard Parameters | Advanced Parameters

Column Name	type	Ignore	Graph	Dummy	Illustrat...	Comments
detailed industry code	Value	☐	☑	☑	☐	nOcc= 52
detailed occupation code	Value	☐	☑	☑	☐	nOcc= 47
major industry code	Nominal	☐	☑	☑	☐	nOcc= 24
major occupation code	Nominal	☐	☑	☑	☐	nOcc= 15
migration code-change in msa	Nominal	☐	☑	☑	☐	?,MSA to MSA,Nonmover,NonMSA to ...
migration code-change in reg	Nominal	☐	☑	☑	☐	?,Same county,Nonmover,Different re...
migration code-move within reg	Nominal	☐	☑	☑	☐	?,Same county,Nonmover,Different st...
own business or self employed code	Value	☐	☑	☑	☐	0,2,1
veterans benefits code	Value	☐	☑	☑	☐	2,0,1

Select All — Variable activate/inactivate: Activate, Inactivate — Variable Type: Value, Nominal, Binary, ☐ force — Chart & Graph: Yes, No — Create Dummy: Yes, No

This manipulation HAS BEEN performed in the rest of this document.

You must define the column that contains the primary key. In our example the primary key is the column "Key". See illustration:



TIM Type Var File Editor - D:\Documents\TIM\DEMO-INCOME\demoIncome.TypeXML

File Actions Windows

Type Var Editor | Analysis | Execution Log

No Filter — Filter 1: ☑ activate *key* Invert ☐ — Filter 2: ☐ activate ** Invert ☐ — Filter 3: ☐ activate List Var Invert ☐ — 1

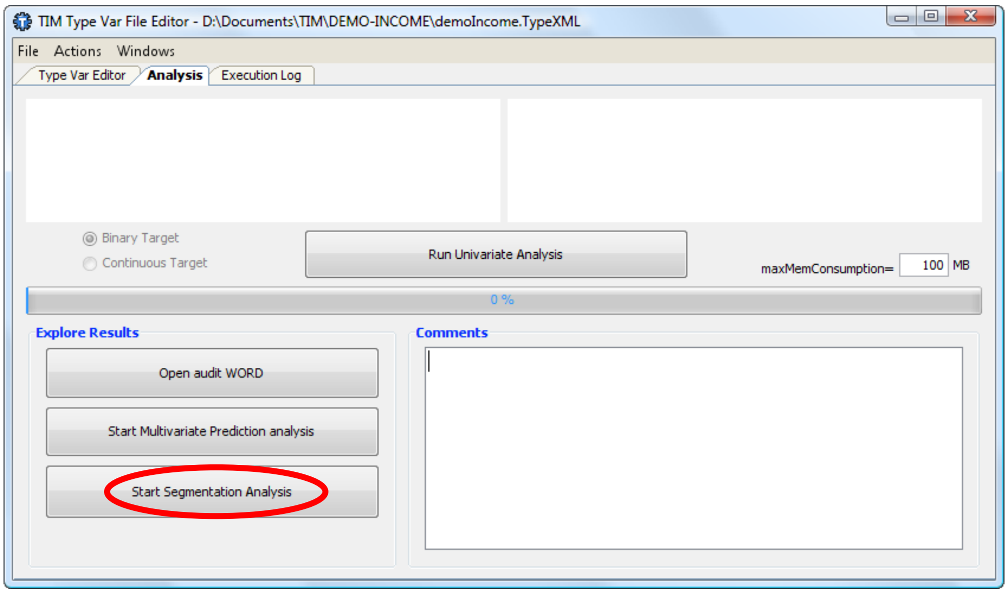
Standard Parameters | Advanced Parameters

Column Name	type	Ignore	Graph	Dummy	Illustrat...	Comments
key	Value	☐	☑	☑	☐	nOcc>=50000

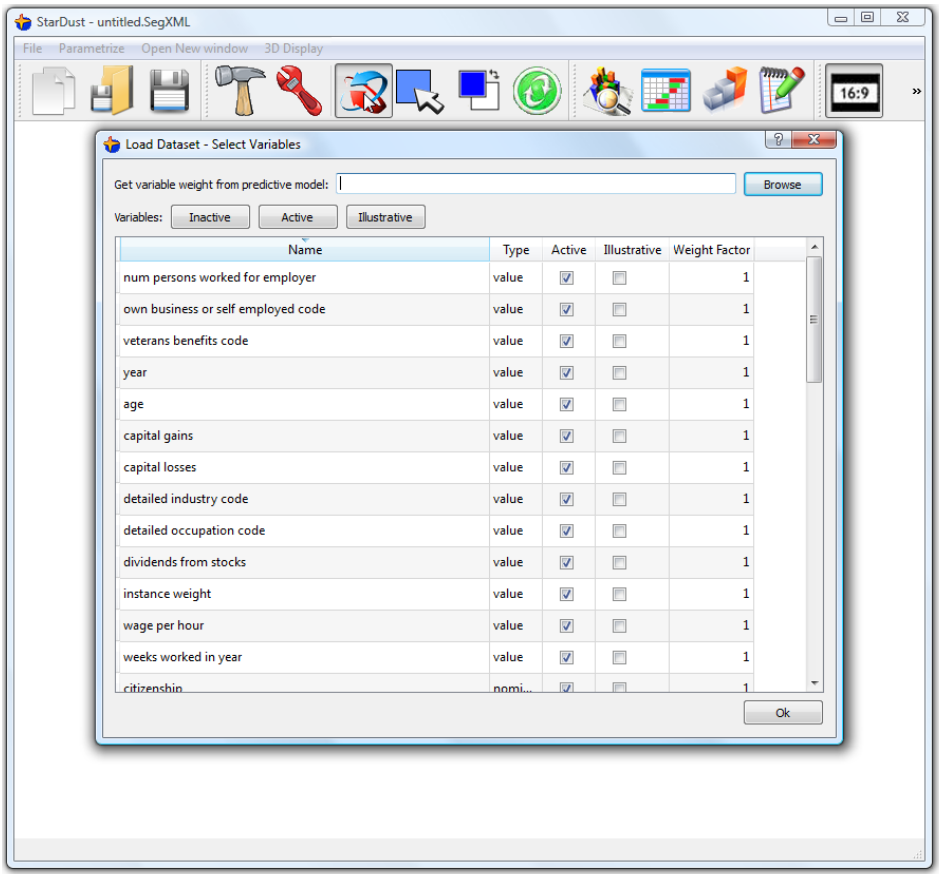
We do NOT have to define the column containing the target. For a segmentation analysis, there is no target.

The other parameters available inside the .TypeXML File editor are not important at this time. You can leave them at their default value.

Click on the "Analysis" tab and then on the Run Univariate Analysis button. When the univariate analysis is complete, you obtain a ".DescXML file". You can use this new ".DescXML file" to start a new segmentation analysis: Click the Start Segmentation Analysis button:


TIM Type Var File Editor - D:\Documents\TIM\DEMO-INCOME\demoIncome.TypeXML
File Actions Windows
Type Var Editor Analysis Execution Log
Binary Target
Continuous Target
Run Univariate Analysis
maxMemConsumption= 100 MB
0 %
Explore Results
Open audit WORD
Start Multivariate Prediction analysis
Start Segmentation Analysis
Comments

The main window of StarDust appears:


StarDust - untitled.SegXML
File Parametrize Open New window 3D Display
16:9
Load Dataset - Select Variables
Get variable weight from predictive model:
Browse
Variables: Inactive Active Illustrative
Ok

Name	Type	Active	Illustrative	Weight Factor
num persons worked for employer	value	✓		1
own business or self employed code	value	✓		1
veterans benefits code	value	✓		1
year	value	✓		1
age	value	✓		1
capital gains	value	✓		1
capital losses	value	✓		1
detailed industry code	value	✓		1
detailed occupation code	value	✓		1
dividends from stocks	value	✓		1
instance weight	value	✓		1
wage per hour	value	✓		1
weeks worked in year	value	✓		1
citizenship	nomi...	✓		1

This first screen allows you to:

1. Select the columns that will be loaded into memory for analysis. Usually, you load everything. For very large database of several tens of gigabytes, you could "skip" some columns to gain time and memory.
2. Select the weight given to each column. This weight can automatically be extracted from a predictive model. This feature is very important when performing a segmentation analysis AFTER a predictive analysis. See the next section (section 4.3) for more information at this subject.

All the ACTIVE and ILLUSTRATIVE variables will be loaded into memory. The active variables will be used to create the segmentation. In opposition, the ILLUSTRATIVE variables won't be use to create the segmentation but rather to explain it from a business-point of view.

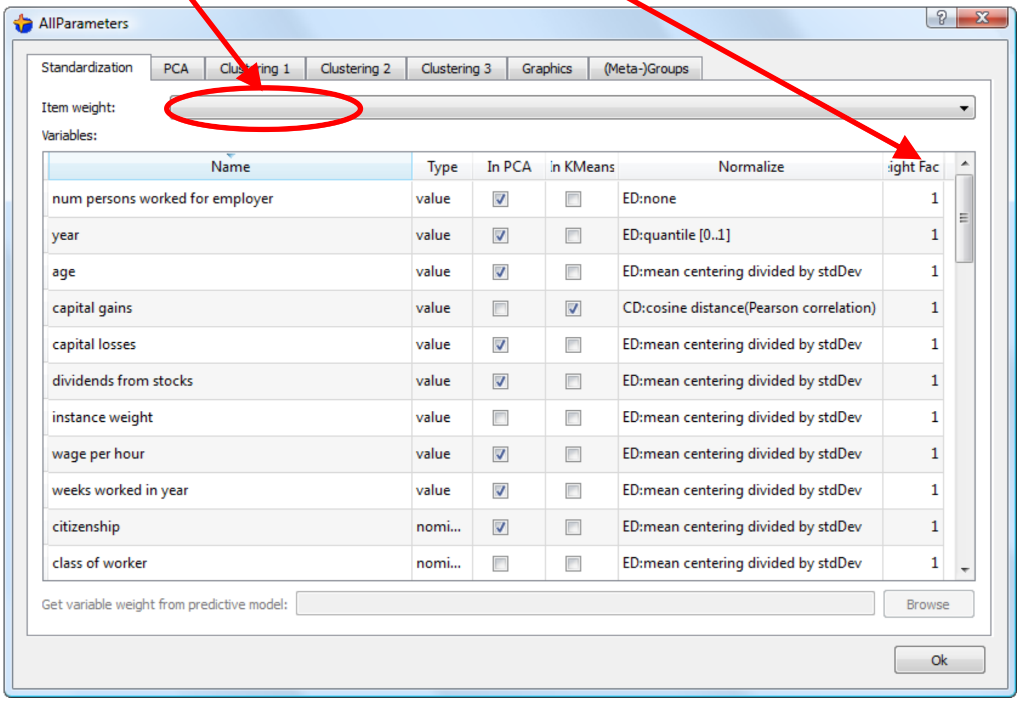
You can directly click the Ok button to start the segmentation analysis: the whole dataset is loaded into memory. Be patient: it can take some time, especially if the dataset is very large.

4.3. Using StarDust AFTER a predictive analysis

For your first lecture of this guide, I suggest you to skip this section a go directly to section 5. In this section, we will assume that you are familiar with the usage of StarDust.

Inside StarDust, all the algorithms (PCA, K-means & Ward) are able to take into account user-defined "weights" on the columns and on the rows.

A large weight on a row of your dataset means that this particular individual is very important. A large weight on a column of your dataset means that this column is very important. You can set the row-weights here: and the column-weights here:



Item weight:

Variables:

Name	Type	In PCA	In KMeans	Normalize	ight Fac
num persons worked for employer	value	✓		ED:none	1
year	value	✓		ED:quantile [0..1]	1
age	value	✓		ED:mean centering divided by stdDev	1
capital gains	value		✓	CD:cosine distance(Pearson correlation)	1
capital losses	value	✓		ED:mean centering divided by stdDev	1
dividends from stocks	value	✓		ED:mean centering divided by stdDev	1
instance weight	value			ED:mean centering divided by stdDev	1
wage per hour	value	✓		ED:mean centering divided by stdDev	1
weeks worked in year	value	✓		ED:mean centering divided by stdDev	1
citizenship	nomi...	✓		ED:mean centering divided by stdDev	1
class of worker	nomi...			ED:mean centering divided by stdDev	1

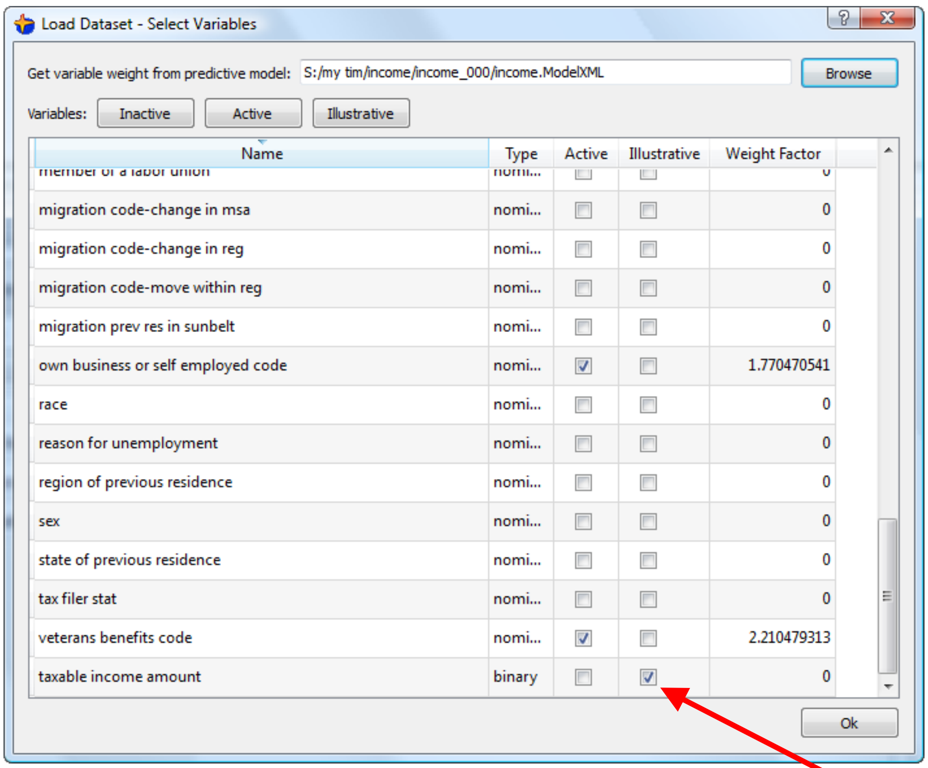
Get variable weight from predictive model:

The column-weights are designated as w_i in section 5.5.1.

The sum of the row-weights inside a segment is designated as w_{S_A} in section 5.5.3.2.

There exist some optimal column-weights that allow you to visualize in an optimal way the characteristics that separate your "Target Group" from the rest of the population. When you are using these optimal column-weights, you can easily "see" and "explore" your "Target Group" in relation to the other individuals.

For example, for the census-income database:



Load Dataset - Select Variables

Get variable weight from predictive model: S:/my tim/income/income_000/income.ModelXML [Browse]

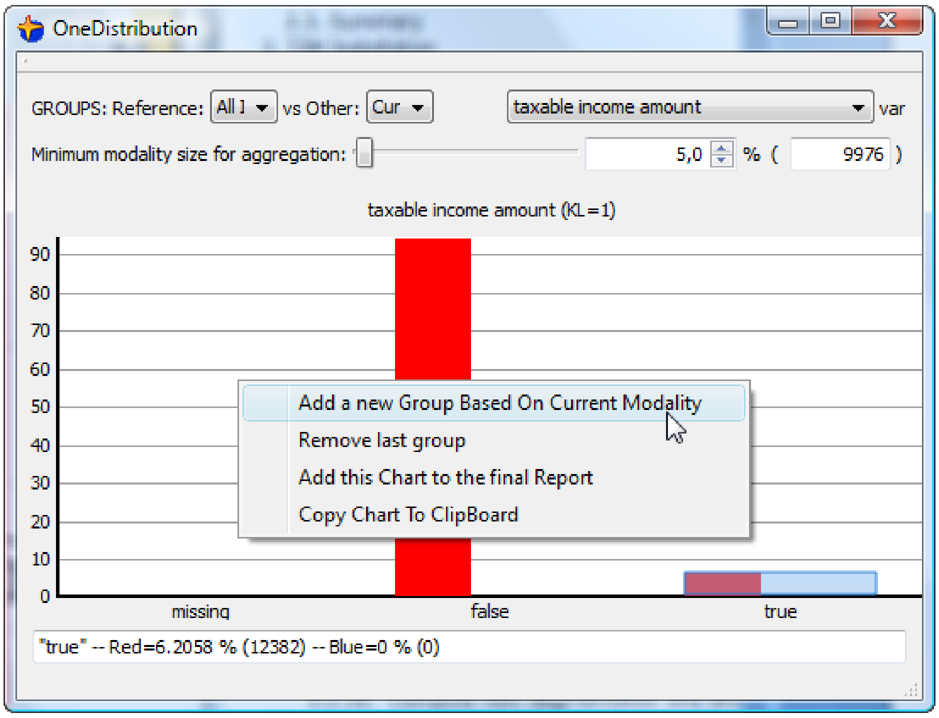
Variables: [Inactive] [Active] [Illustrative]

Name	Type	Active	Illustrative	Weight Factor
member of a labor union	nomi...	☐	☐	0
migration code-change in msa	nomi...	☐	☐	0
migration code-change in reg	nomi...	☐	☐	0
migration code-move within reg	nomi...	☐	☐	0
migration prev res in sunbelt	nomi...	☐	☐	0
own business or self employed code	nomi...	✓	☐	1.770470541
race	nomi...	☐	☐	0
reason for unemployment	nomi...	☐	☐	0
region of previous residence	nomi...	☐	☐	0
sex	nomi...	☐	☐	0
state of previous residence	nomi...	☐	☐	0
tax filer stat	nomi...	☐	☐	0
veterans benefits code	nomi...	✓	☐	2.210479313
taxable income amount	binary	☐	✓	0

[Ok]

Note that we added as illustrative variable the "taxable income amount"
A good idea is to also add as illustrative variable the "sex".

Thereafter, we load the dataset. We go directly to the chart representing the variable "Taxable income amount" (click The button) and we add the "True" modality as a new group.



OneDistribution

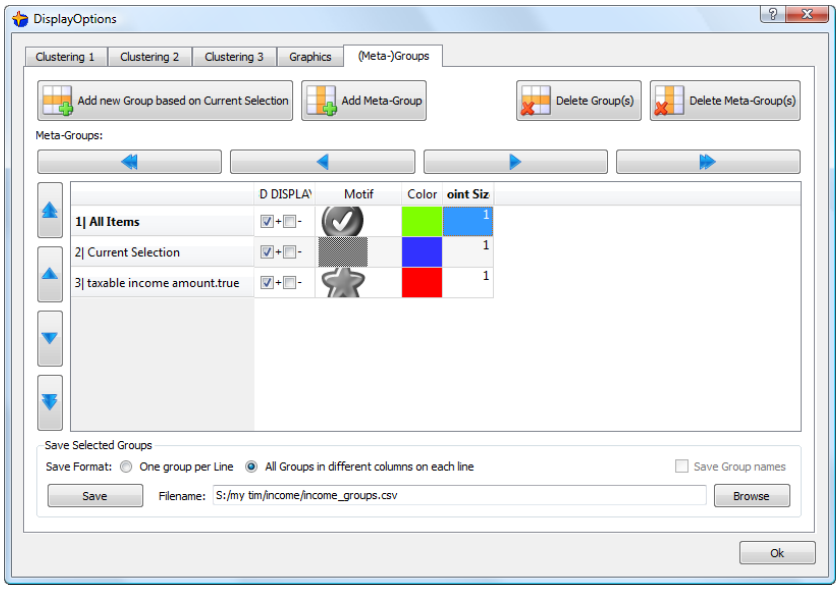
GROUPS: Reference: All I vs Other: Cur — taxable income amount — var

Minimum modality size for aggregation: 5,0 % (9976)

taxable income amount (KL=1)
90
80
70
60
50
40
30
20
10
0
missing
false
true
Add a new Group Based On Current Modality
Remove last group
Add this Chart to the final Report
Copy Chart To ClipBoard

"true" -- Red=6.2058 % (12382) -- Blue=0 % (0)

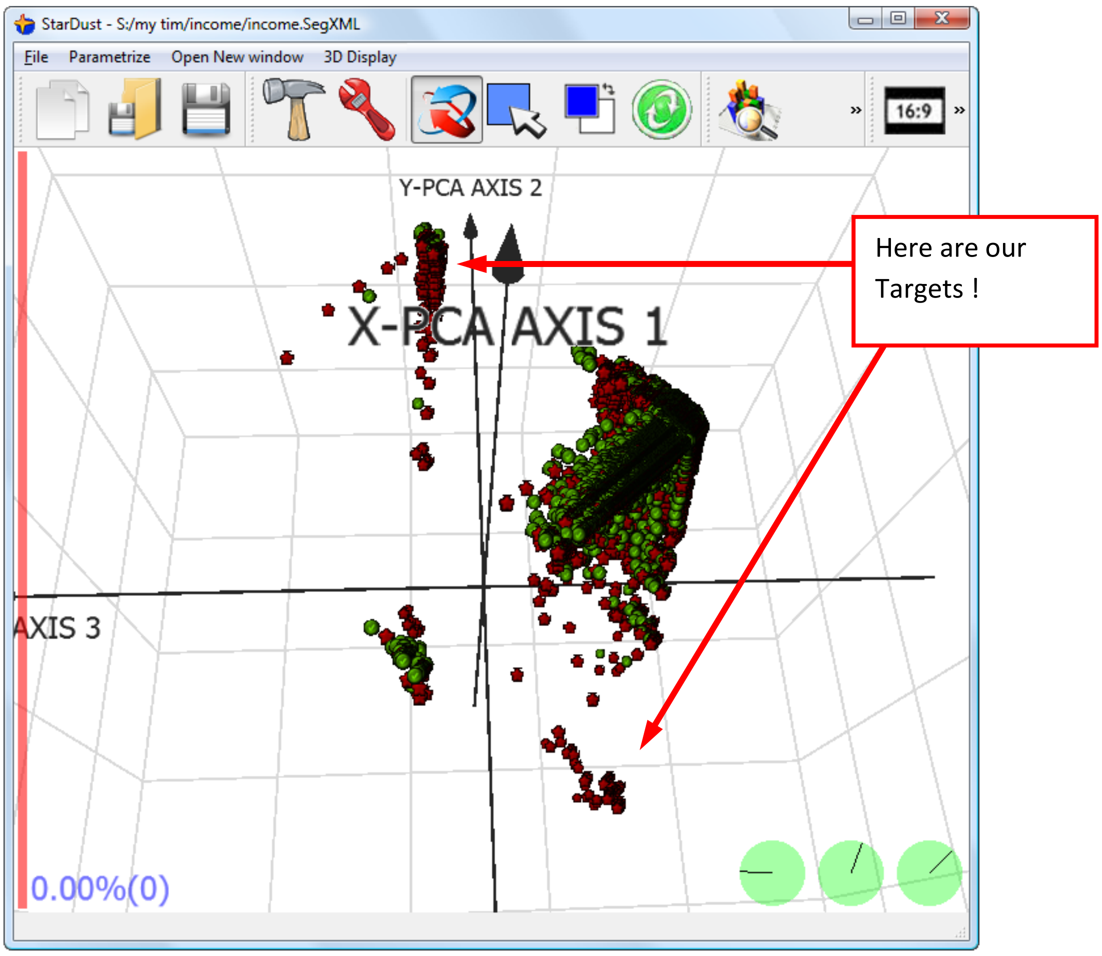
Let’s configure the display of our Groups: click the button and select the “(Meta-)Group” tab:



	D DISPLAY	Motif	Color	oint Siz
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| taxable income amount.true	☑+☐-	★	🟥	1

Save Selected Groups
Save Format: ○ One group per Line [● All Groups in different columns on each line] ☐ Save Group names
Save Filename: S:/my tim/income/income_groups.csv Browse
Ok

... and we finally obtain:



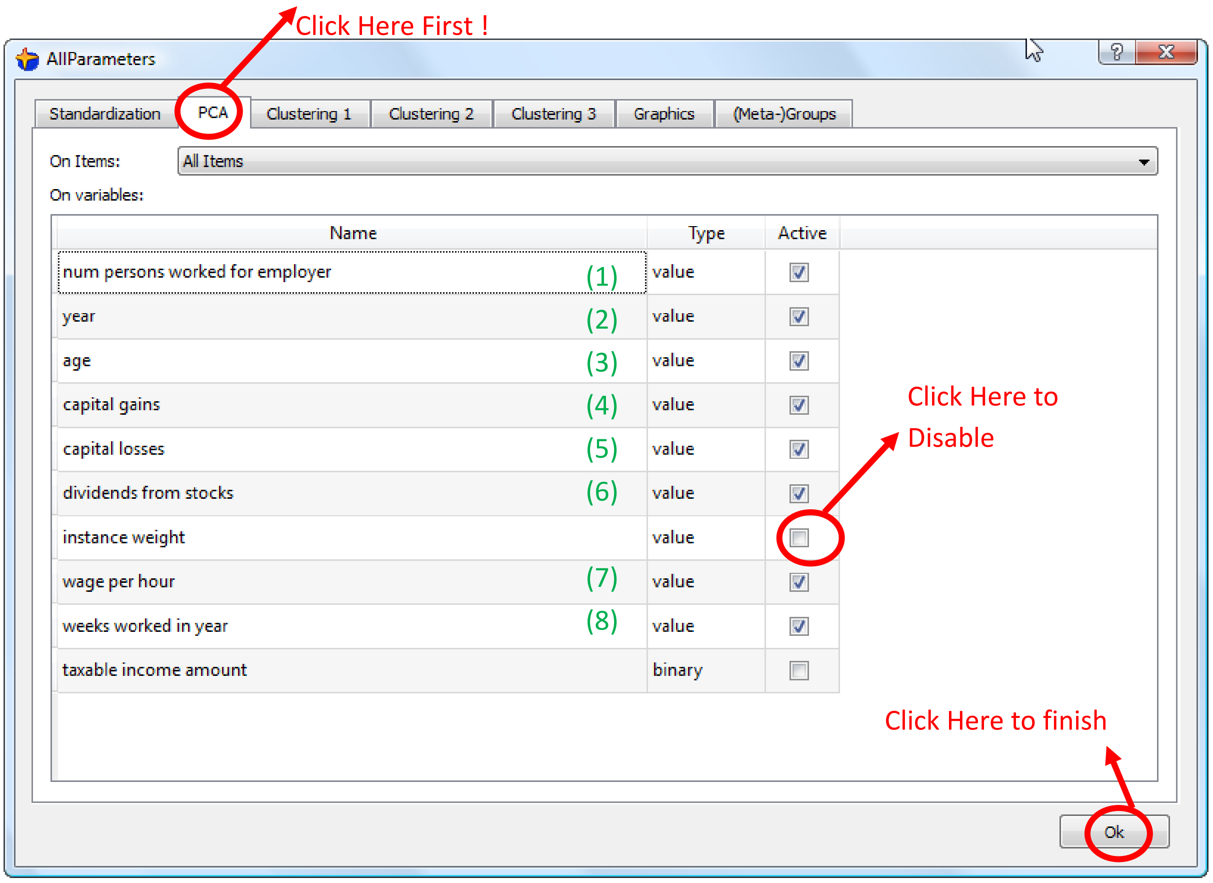
Apparently, in this case, the “Target Group” is roughly divided in two parts! This is interesting... We should investigate what’s the difference between these parts.

5. How To use « TIMi – StarDust module »?

When the dataset is fully loaded, the main parameter window of StarDust opens automatically.

5.1. Viewing the population

Each row of the dataset is an individual that will be represented by a small green 3D point. Stardust computes the 3D coordinates of each individual using the PCA technique (see section 2.1. about “dimensionality reduction”). You can configure which variable will be used inside the PCA on this screen:

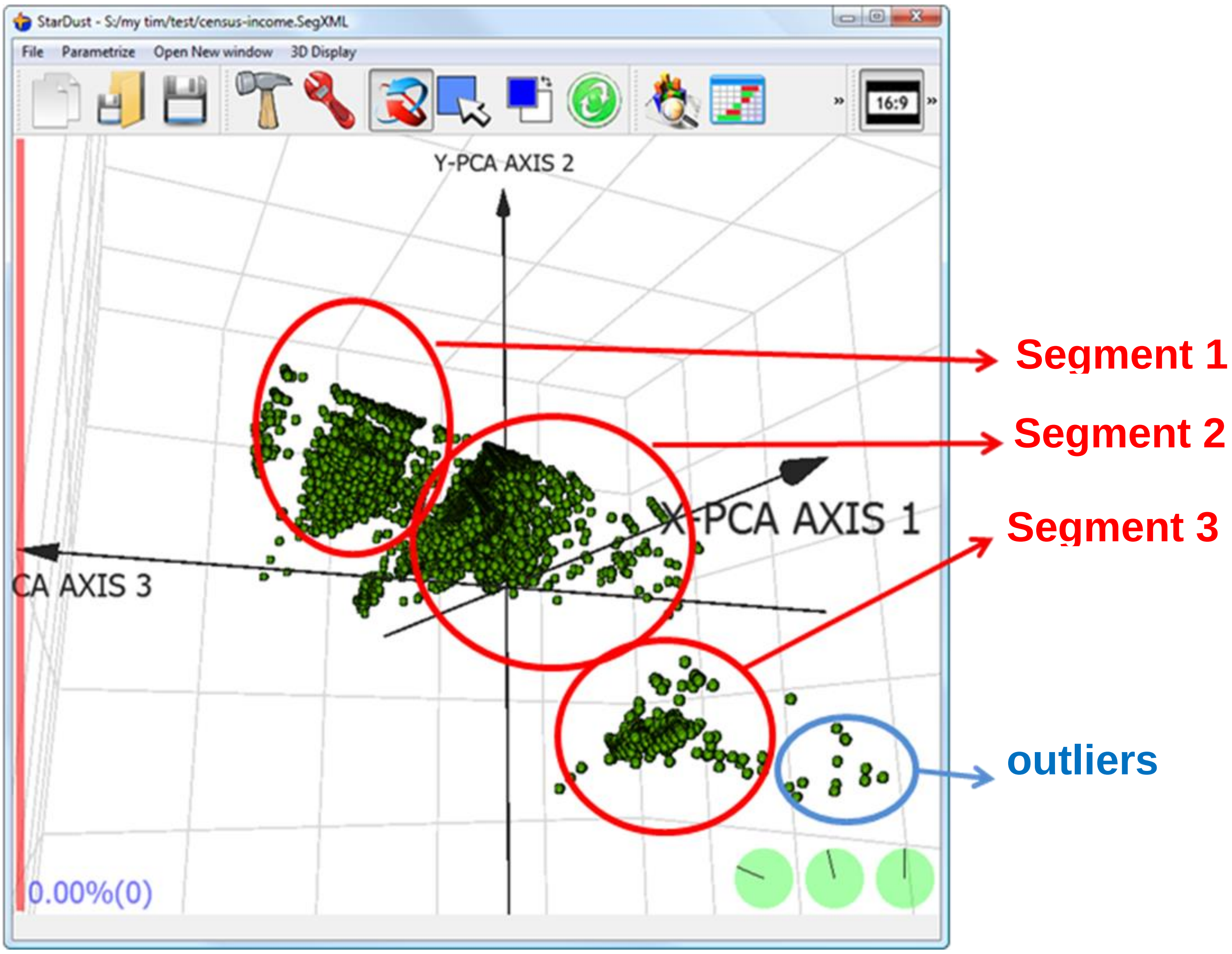


On Items: All Items

On variables:

Name		Type	Active
num persons worked for employer	(1)	value	✓
year	(2)	value	✓
age	(3)	value	✓
capital gains	(4)	value	✓
capital losses	(5)	value	✓
dividends from stocks	(6)	value	✓
instance weight		value	☐
wage per hour	(7)	value	✓
weeks worked in year	(8)	value	✓
taxable income amount		binary	☐

With the above settings, each individual is, at the very beginning, a point in a 8 dimensional space. Thanks to the PCA technique, we will obtain 3D coordinates for each individual as illustrated on the next StarDust screen:

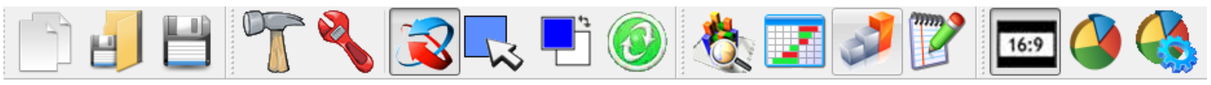


Inside the census-income database there are around 200 thousand individuals. This means that we are actually looking at a graph that contains 200 thousand points. We can zoom-in, rotate around the graph in 3 Dimension and in real-time.

Using Stardust, you can directly « see » for yourself the 3 segments inside the dataset. By simple visual inspection, you discover existing, natural segments inside your population. You can even see some outliers.

5.2. The central Toolbar.

The main tools available inside StarDust are grouped inside the central ToolBar:



A short explanation of each tool is given hereafter:

5.2.1. Start new segmentation Analysis [FILE]

Before analysing a new dataset, you need a ".DescXML file": See section 4.2.1 and 4.2.2 to know how to obtain a ".DescXML file". When you click the button, you will browse for the ".DescXML file" associated with the dataset that you want to analyze.

5.2.2. Open segmentation Analysis [FILE]

When you are interrupted in the middle of a segmentation Analysis, you can save the state of StarDust anytime by clicking : this will create a ".SegXML" file. When you want to return to the state previously saved, to resume your work, you click the button and browse for the corresponding ".SegXML" file.

5.2.3. Save segmentation Analysis [FILE]

Click the button to save the complete state of StarDust in order to be able to resume your work later on.

5.2.4. Parameter Windows

The button and the button are both opening a window that allows you to parameterize many different things in the display and the algorithms of StarDust. The set of parameters that are accessible through these two buttons is more or less equivalent. The main difference between the button and the button is that the later is applying your modifications "in real-time": i.e. You don't have to click the Ok button to see the result of your modifications: you can directly see "in real-time" all the windows of StarDust updating. Thus, the button uses more CPU resources. If you are manipulating extremely large amount of data, you should rather use the button, to minimise the stress on your CPU.

5.2.5. Camera move and rotation Mode [MOUSE MODE]

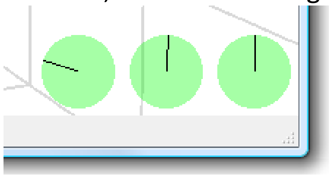
When you click on the button, you enter the "*Camera move and rotation Mode*". In this mode, the mouse operate in the following way:

- Right-Click hold and drag the mouse Up&Down : move the camera view Up&Down
- Right-Click hold and drag the mouse Left&Right : move the camera view Left&Right

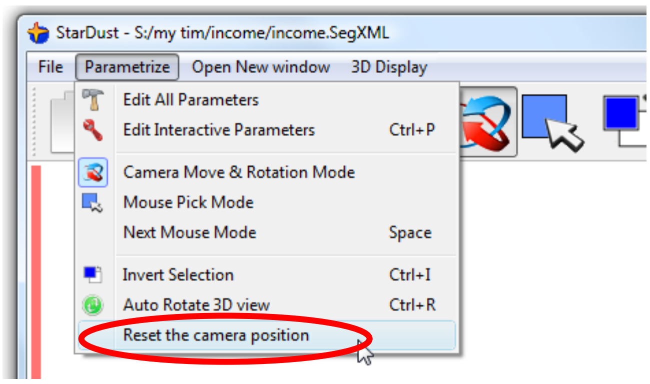
- Left-Click hold and drag the mouse Up&Down : rotate the camera view around the X Axis
- Left-Click hold and drag the mouse Left&Right : rotate the camera view around the Y Axis
- Left+Right-Click hold and drag the mouse Left&Right : rotate the camera view around the Z Axis

- Middle-Click: switch between the "*Camera move and rotation Mode*" and the "*Mouse Pick Mode*". You can also switch between modes with the "space" key.

On lower-right side of the main-window, there are three green translucent discs:



The dials inside these discs represent the current orientation of the camera. Sometime, after playing a little bit too long with the orientation of the camera, we can be a little bit "lost". Therefore, you can "reset" the orientation and the position of the camera by clicking here:



StarDust - S:/my tim/income/income.SegXML
File
Parametrize
Open New window
3D Display
Edit All Parameters
Edit Interactive Parameters Ctrl+P
Camera Move & Rotation Mode
Mouse Pick Mode
Next Mouse Mode Space
Invert Selection Ctrl+I
Auto Rotate 3D view Ctrl+R
Reset the camera position

5.2.6. Mouse Pick Mode [MOUSE MODE]

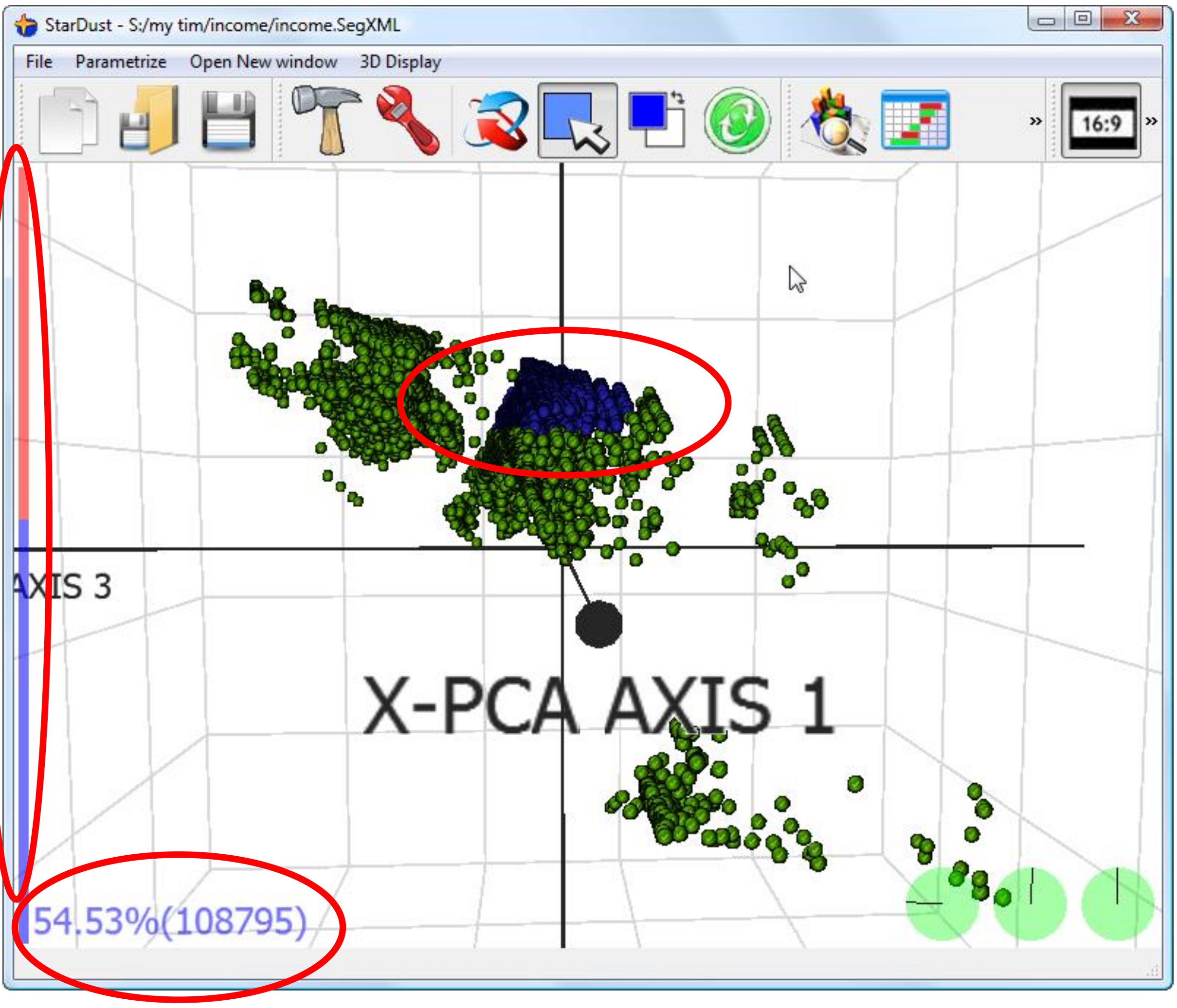
When you click on the button, you enter the "*Moue Pick Mode*". In this mode, the mouse operate in the following way:

- Left-Click hold and drag the mouse: the points in the blue rectangle are ADDED to the selection.
- Right-Click hold and drag the mouse: the points in the blue rectangle are SUBSTRACTED from the selection.

- Middle-Click: switch between the "*Camera move and rotation Mode*" and the "*Mouse Pick Mode*". You can also switch between modes with the "space" key.

All the individuals that are inside the selection are in dark-blue colour.

Inside the lower-left side of the window, you get some information about your selection: the amount of individual inside the selection in percentage and in absolute value. The vertical bar on the left-side of the window represents the same information:



5.2.7. Invert selection

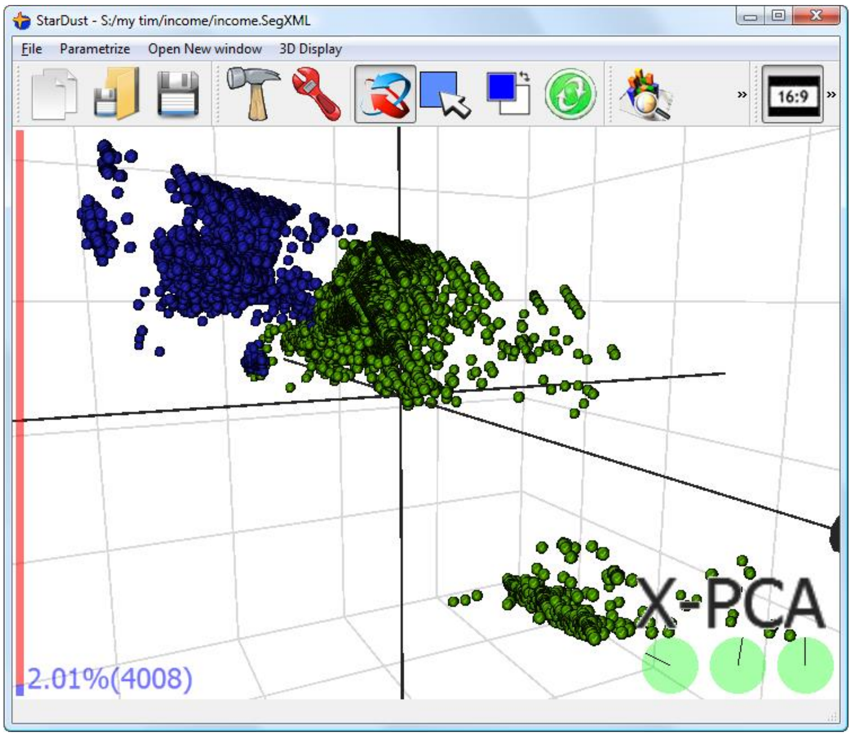
This button inverts the selection.

5.2.8. Auto-rotate 3D view

When you click the button, StarDust rotates the camera in every direction (Grand Tour). When you see a viewing angle that pleases you, click anywhere inside the window to stop the rotation. You can reset the orientation and the position of the camera by clicking “*Reset camera postion*” inside the “*Parametrize*” drop-down menu.

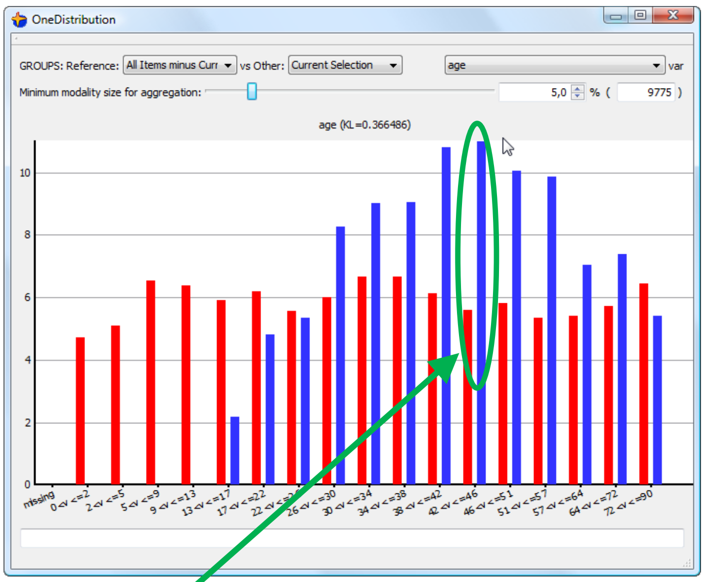
5.2.9. View one Var Distribution [CHART]

The button opens a window that allows you to compare the distribution of one variable between two different groups of individuals. For example, select with your mouse some individuals inside the main windows of TIMi:



... and then click on the button and select the “Age” Variable inside the combo-list on the upper-right side of the window.

You should now see this:


OneDistribution
GROUPS: Reference: All Items minus Curr vs Other: Current Selection
age
var
Minimum modality size for aggregation: 5,0 % (9775)
age (KL=0.366486)

The red histogram represents the distribution of the age amongst the whole population (minus the people inside the selection). We can see that, inside the whole population, the histogram is more or less "flat": all the ages are more or less represented equally.

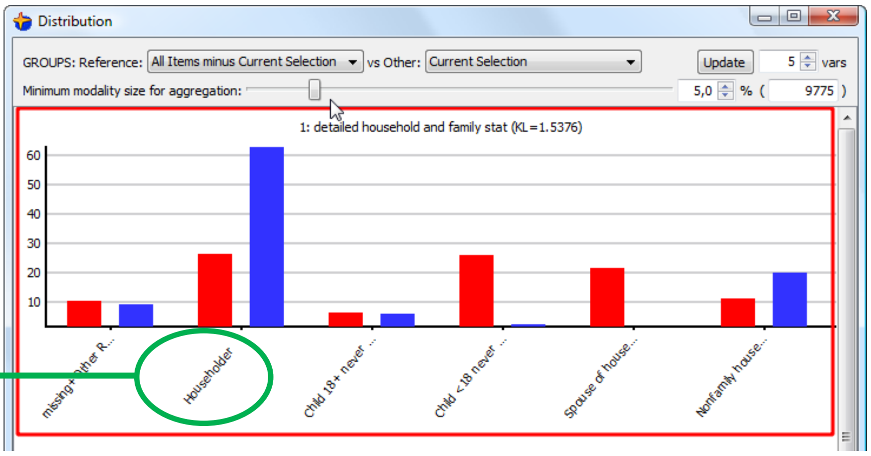
The blue histogram represents the distribution of the age amongst the selected individuals. We can see (look here:) that the people with age between 43 and 46 are more represented in the selection than inside the whole population.

A common measure of distance between 2 histograms (between 2 distributions), is the "**K**ullback-**L**iebler distance" or, in short, the "KL". In this example, the KL is 0.366486. If a variable possesses a high KL value, it means that the variable is strongly different between the 2 compared groups and, thus, strongly characterizes each group. We want to find good candidate-variable to characterizes a "*group of individual*" (a "*group of individual*" is usually a "*segment*"), we must look first at the variable with a high KL value. See the next section to do that.

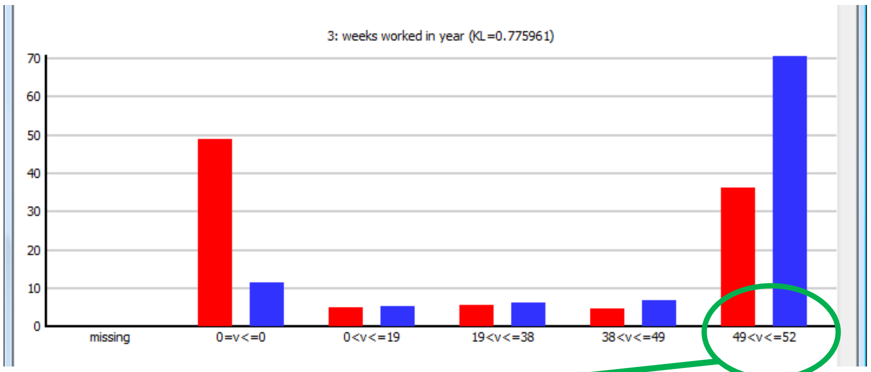
5.2.10. Compare variable distribution for many variables [CHART]

The button opens a window that allows you to compare the distribution of all the variables between two different groups of individuals. The variables are sorted from the highest KL to the lowest KL. Thus, the most interesting variables to characterize the current selection are the variables in top of the list.

For example, if we continue the example of the previous section, we will see the following window:


Distribution
GROUPS: Reference: All Items minus Current Selection
vs Other: Current Selection
Update
5 vars
Minimum modality size for aggregation:
5,0 % (9775)
1: detailed household and family stat (KL=1.5376)
missing+ other R...
Householder
Child 18+ never ...
Child <18 never ...
Spouse of house...
Nonfamily house...

... (the second variable in the list is not displayed because it's nearly the same as the first one)



... (other variables omitted)

This indicates us that our current selection is mainly characterized by people that are "householder". This is really the "main characteristic" because it's KL is 1.5376.

Another fact that characterized our selection is the fact that people, inside our selection, are always working (52 weeks per year), the full year (in opposition to the global population that includes many people that are not working at all!). This second characteristics is a lot less relevant than the first one because the KL is only 0.775961 for the variable "*weeks worked in year*" (in opposition to a KL of 1.5276 for the "householder" variable).

It could be interesting to have a "summary-table" of the KL of all the variables: see next section to do that.

5.2.11. View Summary Table [CHART]

The button opens a window that displays a summary table that contains the KL value of all the variables. You can sort the table in many different ways by clicking on the column headers. To remind you, the most interesting variables to display to characterize the current selection are the variables with the highest KL.

Instead of displaying the "**K**ullback-**L**iebler distance" (KL) for each of the variables, you can also use alternative distances that also characterize the distance between two distributions. The other alternative distances to the "KL" are the "Test-Value" distance and the "AUC" distance.

Note that, strictly speaking, we should not use the word "distance" but rather the word "dissimilarity" because a "distance" has several additional mathematical properties over a "dissimilarity". We will use indifferently both words.

A possible interpretation of the test-value is "*The test-value is the difference (positive or negative) between the mean of the 2 distributions evaluated in number of standard deviation of a normal distribution*".

The standard definition of the "Test-Value" of a variable ***X*** between 2 distributions (the first distribution is defined on a Reference group R and the second distribution is defined on a "Selection" group S) is computed this way:

* **For a continuous variable *X*:** Std Test-Value for the variable $X = \dfrac{\bar{X}_S - \bar{X}_R}{s(X_S)}$

$$\text{with} \quad s^2(X_S) = \frac{n_R - n_S}{n_R - 1} \frac{s^2(X_R)}{n_s}$$

Where we used the following notations:

R: is the "Reference" group: usually, the whole population.
S: is the "Selected" Group: usually, the mouse selection or the segment to analyze.
n_R : is the number of individuals inside the R group.
n_S : is the number of individuals inside the S group.
$\bar{X}_R$: is the mean inside the R group of the variable ***X*** to analyze.
$\bar{X}_S$: is the mean inside the S group of the variable ***X*** to analyze.
$s^2(X_R) = \dfrac{1}{n_r} \sum_{i \in R} (X_i - \bar{X}_R)^2$: is the variance inside the R group of the variable ***X*** to analyze.

* **For a nominal or binary variable *X*:** we create a Test-Value for each modality ***m*** of the variable ***X***:

$$\text{Std Test-Value for the modality m of the variable } X = \frac{m_S - {m_R n_S}/{n_R}}{s(X_S)}$$

$$\text{with} \quad s^2(X_S) = n_S \frac{n_R - n_S}{n_R - 1} s^2(X_R) \quad \text{and} \quad s^2(X_R) = \frac{m_R}{n_R}\left(1 - \frac{m_R}{n_R}\right)$$

Where we used the following notations:

R: is the "Reference" group: usually, the whole population.
S: is the "Selected" Group: usually, the mouse selection or the segment to anlyse.
n_R : is the number of individuals inside the R group.
n_S : is the number of individuals inside the S group.
m_R : is the number of occurrence of the modality ***m*** inside the R group.
m_S : is the number of occurrence of the modality ***m*** inside the S group.

The advantage of the test-value over the KL is that the test-value can be positive or negative, depending on the “sense” of the difference between the two distributions. In opposition, the KL value is always positive.

StarDust also proposes you to use a *simplified* definition of the “Test-Value” of a variable ***X*** between 2 distributions. It is computed this way:

* **<u>For a continuous variable *X*:</u>** Simplified Test-Value for the variable $X = \frac{\bar{X}_S - \bar{X}_R}{s(X_R)}$

…where we used the same notations as previously.

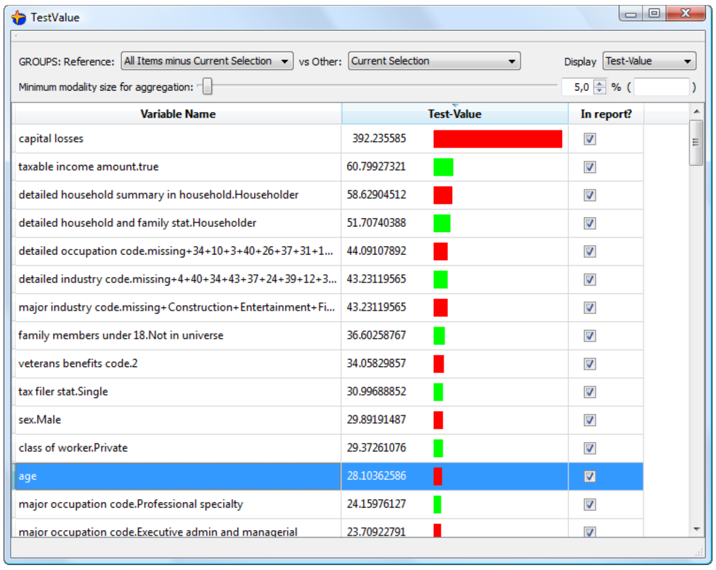
* **<u>For a nominal or binary variable *X*:</u>** we create a Test-Value for each modality ***m*** of the variable ***X***:

$$\text{Simplified Test-Value for the modality m of the variable } X = \frac{m_S - {}^{m_R n_S}/_{n_R}}{s(X_R)}$$

…where we used the same notations as previously.

The “test-value” is comparing the distribution of the same variable between two groups: the “Reference” group and the “Selection” group S. The equation of the “test-value” only makes sense if the “Selection” group S is completely contained inside the “Reference” group. This is why, in all the “test-value” computations, we are using a slightly modified definition of the “Reference” group R: our “Reference” group R is always the union between the “Reference” group given by the user inside the StarDust interface and the “Selection” group S.

For example, using the “test-value” we can see inside the table below that the mean of the age inside our selection is far **<u>greater</u>** than the mean of the age inside the whole global population:


TestValue

GROUPS: Reference: All Items minus Current Selection vs Other: Current Selection Display Test-Value

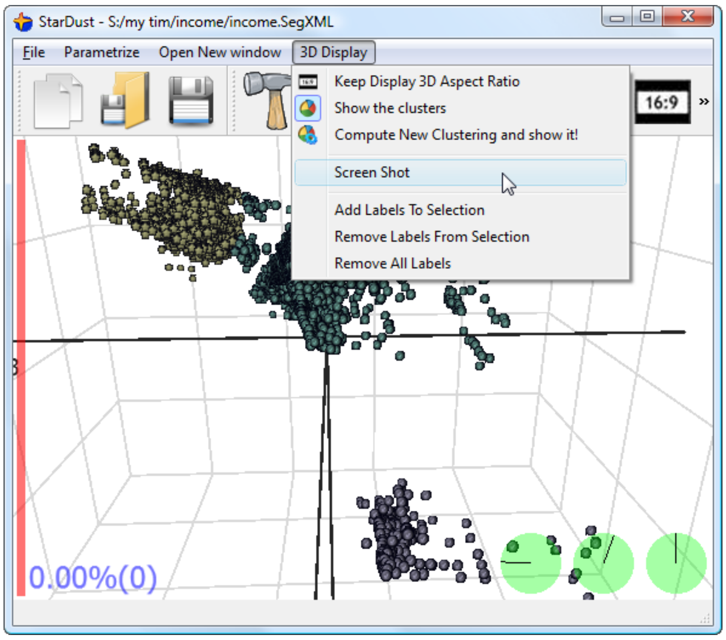
Minimum modality size for aggregation: 5,0 % ( )

Variable Name	Test-Value		In report?
capital losses	392.235585	🟥	✓
taxable income amount.true	60.79927321	🟩	✓
detailed household summary in household.Householder	58.62904512	🟥	✓
detailed household and family stat.Householder	51.70740388	🟩	✓
detailed occupation code.missing+34+10+3+40+26+37+31+1...	44.09107892	🟥	✓
detailed industry code.missing+4+40+34+43+37+24+39+12+3...	43.23119565	🟩	✓
major industry code.missing+Construction+Entertainment+Fi...	43.23119565	🟥	✓
family members under 18.Not in universe	36.60258767	🟩	✓
veterans benefits code.2	34.05829857	🟥	✓
tax filer stat.Single	30.99688852	🟩	✓
sex.Male	29.89191487	🟥	✓
class of worker.Private	29.37261076	🟩	✓
age	28.10362586	🟥	✓
major occupation code.Professional specialty	24.15976127	🟩	✓
major occupation code.Executive admin and managerial	23.70922791	🟥	✓

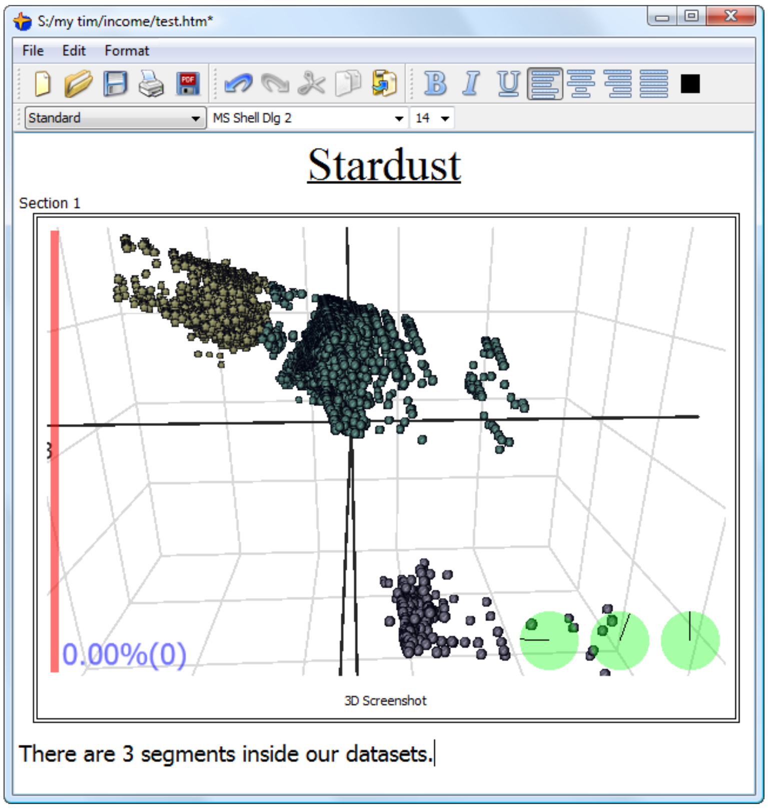
Based on the “positive Test-Value”, we can say that our selection contains “older” people. If we were only looking at the “KL” distance, the only thing that we can say is “the age is <u>different</u> inside the selection”. Thus, the “Test-Value” gives you a more precise description of your selection than the “KL” distance (because, with the Test-Value”, you have a “sign”: positive or negative: you can say “older” or “younger”, for example).

5.2.12. View final report

The button displays a small HTML notepad where you can write all the "insights" and interesting facts that you have found about your customers. All the charts, tables and 3D views of StarDust can easily and directly be "copy-pasted" inside the "Final Report Notepad". For example, when you click here:


StarDust - S:/my tim/income/income.SegXML
File
Parametrize
Open New window
3D Display
Keep Display 3D Aspect Ratio
Show the clusters
Compute New Clustering and show it!
Screen Shot
Add Labels To Selection
Remove Labels From Selection
Remove All Labels
16:9
0.00%(0)

... you are adding inside the "Final Report Notepad" the following:

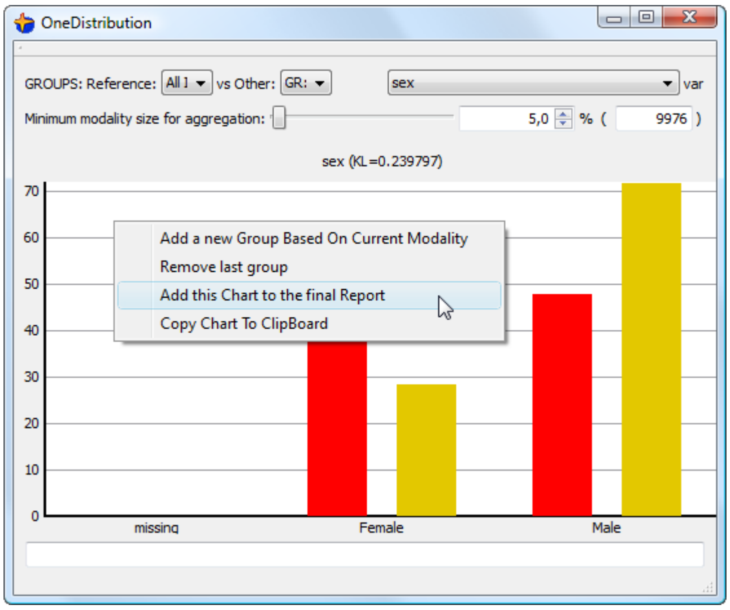


Stardust

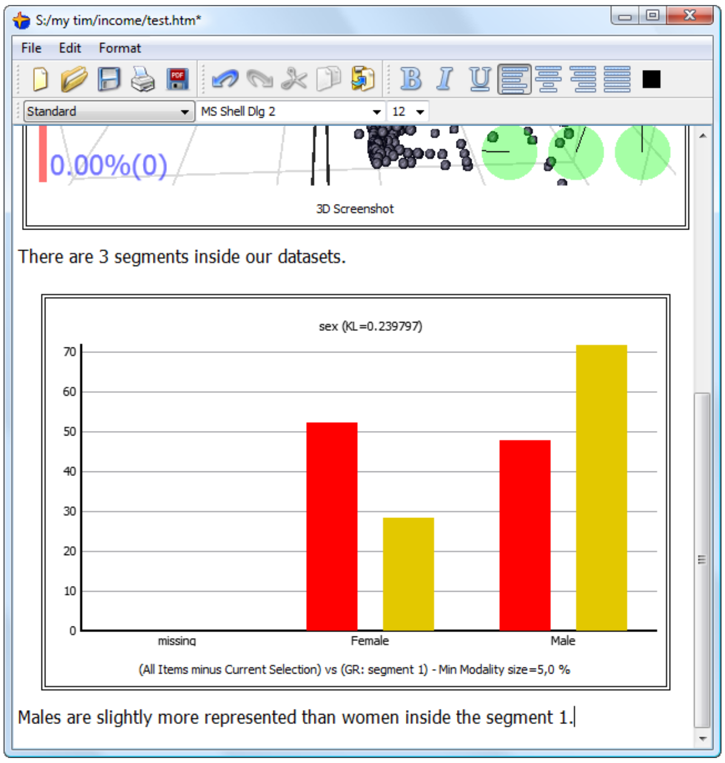
Section 1

There are 3 segments inside our datasets.

Another example: Right-click on any chart of StarDust and select “Add this chart to the final Report”:



... you obtain:


S:/my tim/income/test.htm*
File
Edit
Format
Standard
MS Shell Dlg 2
12
0.00%(0)
3D Screenshot
sex (KL=0.239797)
missing
Female
Male
(All Items minus Current Selection) vs (GR: segment 1) - Min Modality size=5,0 %

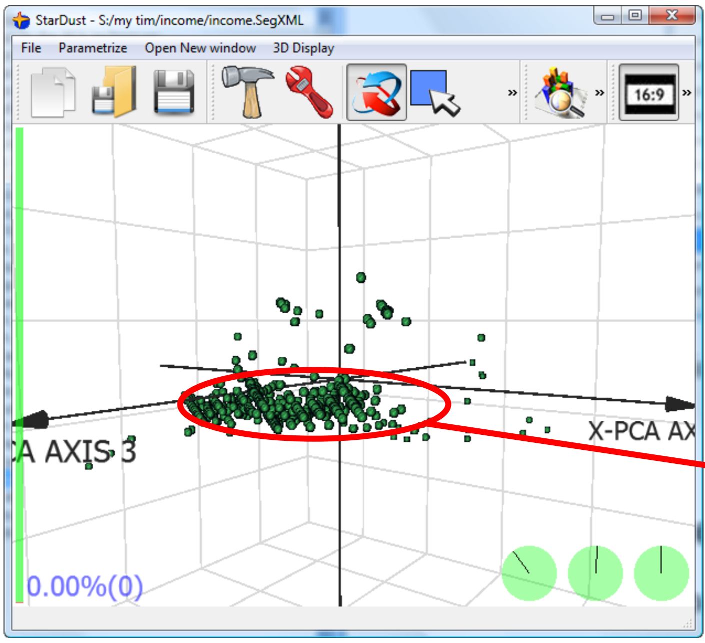
There are 3 segments inside our datasets.

Males are slightly more represented than women inside the segment 1.

5.2.13. 16:9 Keep Display Aspect Ratio?

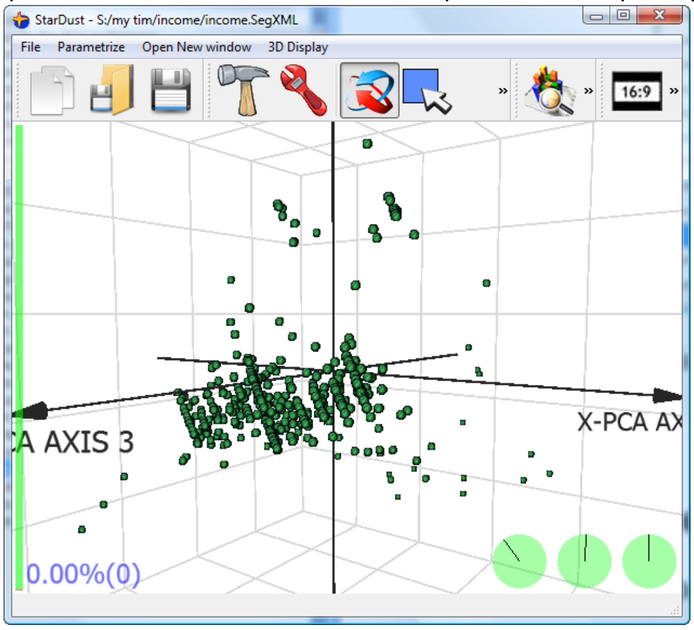
When you click the 16:9 button, you alternate between two representations of the same “cloud of points”:

1. The first representation is the most respectful of the real distances that exists between points (individuals) inside your data. This is the most accurate representation of your population. ...But sometime (really often in reality), you end-up watching a graphic where everything is so cluttered that you can’t see anything. For example:



When you obtain such display, one easy solution to still "be able to see something" is to click on the 16:9 button to "Strech" the "cloud of points" in order to completely fill-in the 3D cube.

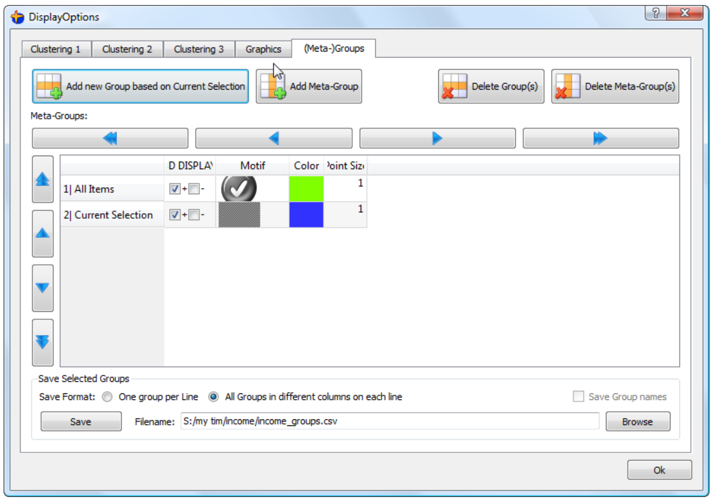
2. The second representation "stretch" the "cloud of points" to completely "fill-in" the 3D cube:



5.2.14. Show the Segments

When you click the button, you alternate between the two different ways of computing the colours of all the points inside the 3D display. These two different ways are:

1. The colours of the points are setup through the advanced 3D display "Group" manager. You can access this manager when clicking the button and after selecting the "(Meta-)Groups" tab:



DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

Add new Group based on Current Selection | Add Meta-Group | Delete Group(s) | Delete Meta-Group(s)

Meta-Groups:

	D DISPLAY	Motif	Color	Point Size
1| All Items	✓+ ☐-	✓	🟩	1
2| Current Selection	✓+ ☐-	▨	🟦	1

Save Selected Groups

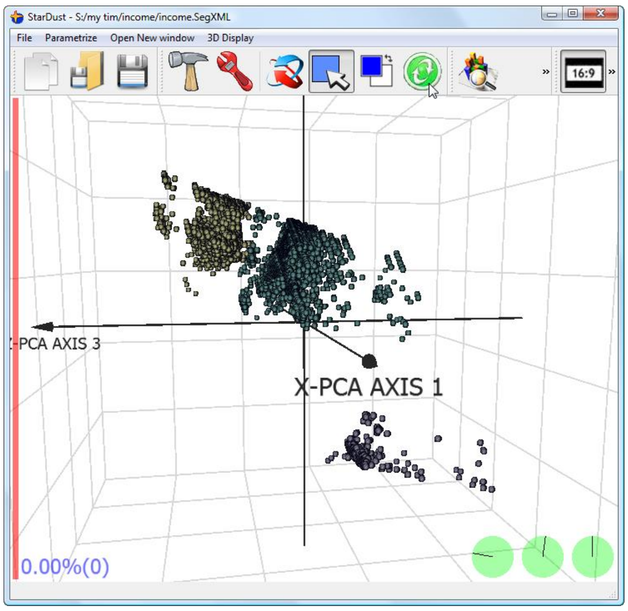
Save Format: One group per Line [✓ All Groups in different columns on each line] ☐ Save Group names

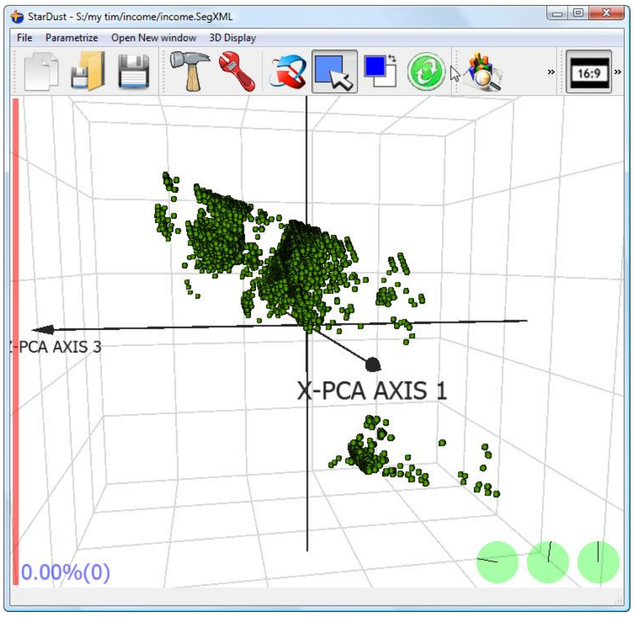
Save | Filename: S:/my tim/income/income_groups.csv | Browse

Ok

2. The colours of the points represent the different segments inside the population.

Typically, the button alternate between two “color-set”: for example:


StarDust - S:/my tim/income/income.SegXML
File Parametrize Open New window 3D Display
16:9
X-PCA AXIS 3
X-PCA AXIS 1
0.00%(0)


StarDust - S:/my tim/income/income.SegXML
File Parametrize Open New window 3D Display
16:9
X-PCA AXIS 3
X-PCA AXIS 1
0.00%(0)

Colours represent Segments

Colours are setup inside the « group » manager

5.2.15. Compute new segmentation and shows it!

As explained in the section 2.2.1., the K-Means algorithm is an algorithm with a strong random component: each time you run the K-Means algorithm, you find a slightly different segmentation. You should thus run the K-Means algorithm several times and "have a look" at several different segmentation. Thereafter, you can select the segmentation that has the best interpretation from a business-point-of-view. You can easily interpret you segments using the charts and tables accessible with the , and buttons.

An easy way to create many different segmentations is to click repetitively on the button. Each time you click on the button, a completely new segmentation is computed and shown.

5.3. Setup the 3D display

The "parameter window" contains 2 tabs that allow you to fully configure the 3D display of StarDust:

1. The "(Meta-)Groups" Tab
2. The "Graphic" Tab

You can access the "parameter window" by clicking on the button or the button.

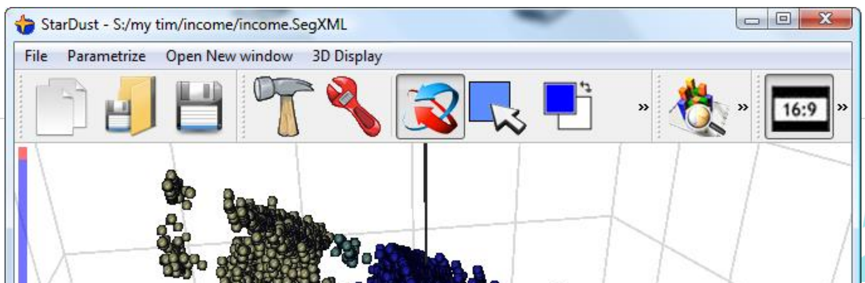
5.3.1. The (Meta-)Group tab

The "(Meta-)Groups" tab allows you to:

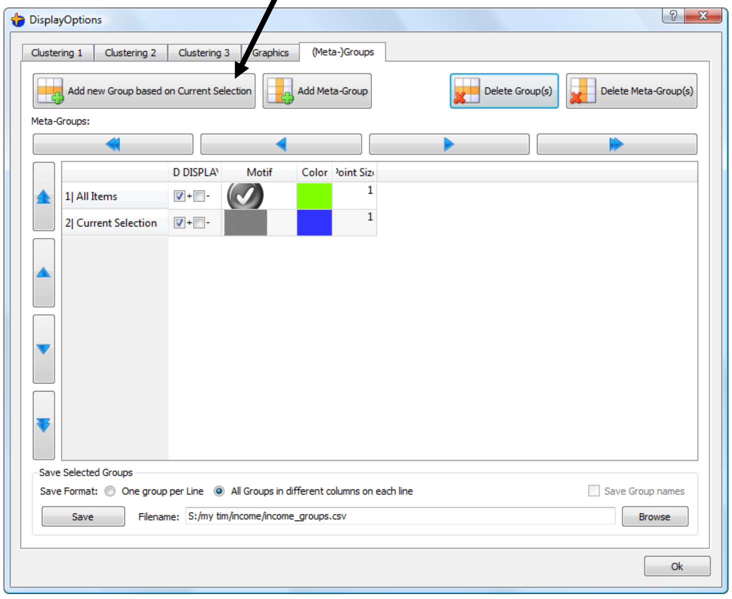
1. manage your groups (add/delete groups)
2. create derived "meta-groups". Meta-Groups are new groups of individual based on the union/intersection/subtraction of the basic groups.
3. configure the colors and motifs displayed inside the 3D view.

5.3.1.1. Creating a new group based on the current selection

The basic items that you manipulate on this tab are "Groups". Different "Groups" are represented by different rows inside the table. The easiest way to create a new group is to select with your mouse some individuals inside the main window of StarDust:



... and then you can mark these selected individual as belonging to the same group by clicking the Add new Group based on Current Selection button.


DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

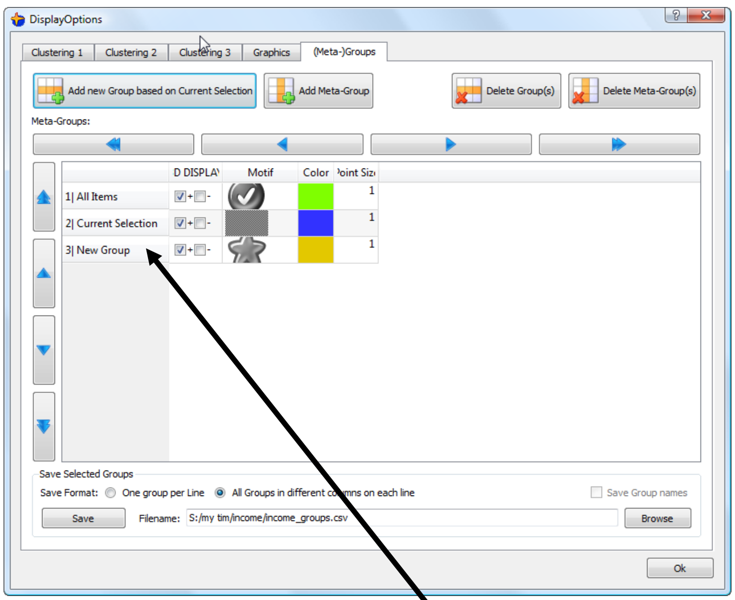
Add new Group based on Current Selection | Add Meta-Group | Delete Group(s) | Delete Meta-Group(s)

Meta-Groups:

	D DISPLAY	Motif	Color	Point Size
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1

Save Selected Groups
Save Format: One group per Line [✓ All Groups in different columns on each line] Save Group names
Save Filename: S:/my tim/income/income_groups.csv Browse

Ok


DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

Add new Group based on Current Selection | Add Meta-Group | Delete Group(s) | Delete Meta-Group(s)

Meta-Groups:

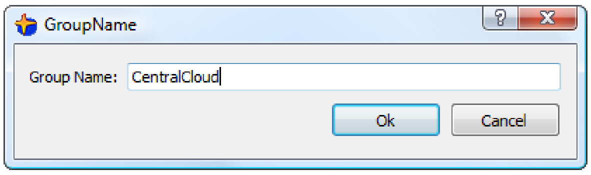
	D DISPLAY	Motif	Color	Point Size
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| New Group	☑+☐-	★	🟨	1

Save Selected Groups
Save Format: One group per Line [✓ All Groups in different columns on each line] Save Group names
Save Filename: S:/my tim/income/income_groups.csv Browse

Ok

You can "rename" the group that you just created by double-clicking on the "row header":

Let's give the name "CentralCloud" to our new group:


GroupName

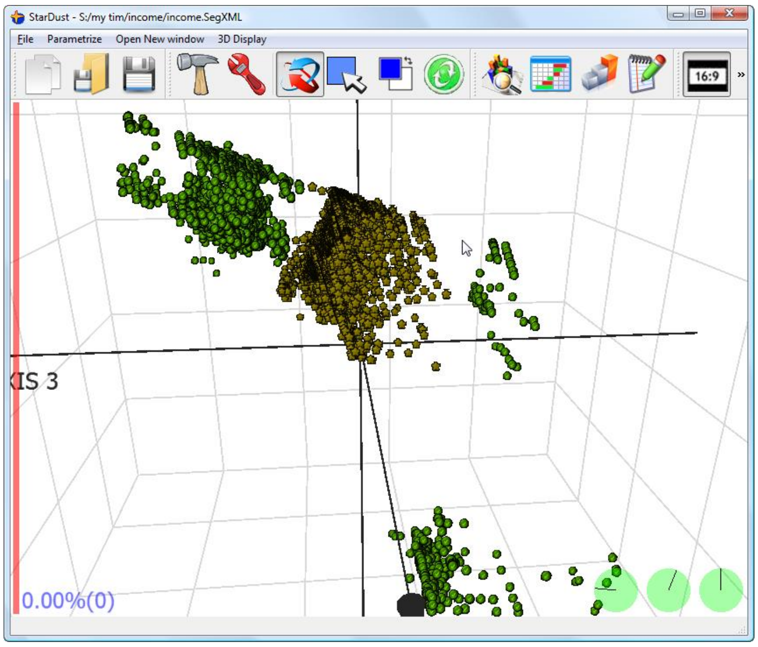
Group Name: CentralCloud

Ok | Cancel

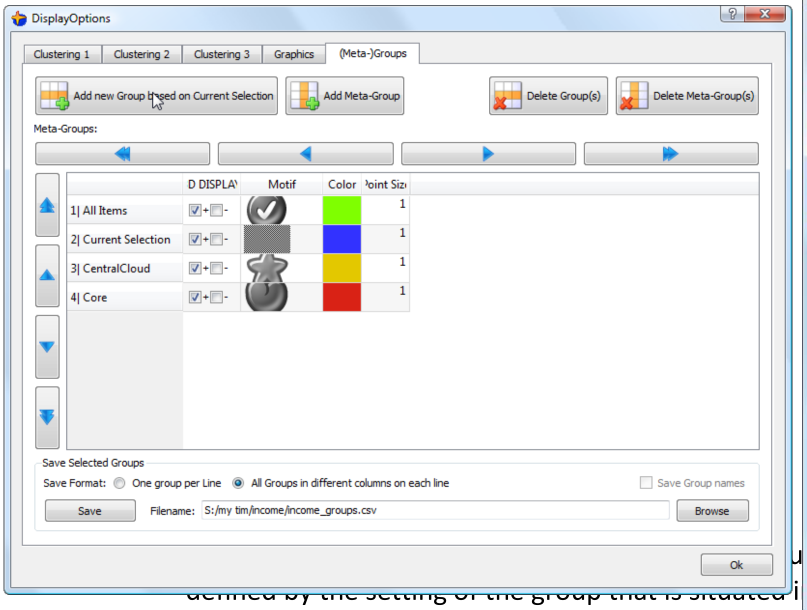
5.3.1.2. Configuring the colour and motif of the 3D points

To select which points are displayed, please refer to the section 5.3.2.1.

Let's now have a look at the main 3D view of StarDust. Deselect everything. The 3D display should now look like this (if it's not the case, please verify that the button on the Toolbar is OFF):



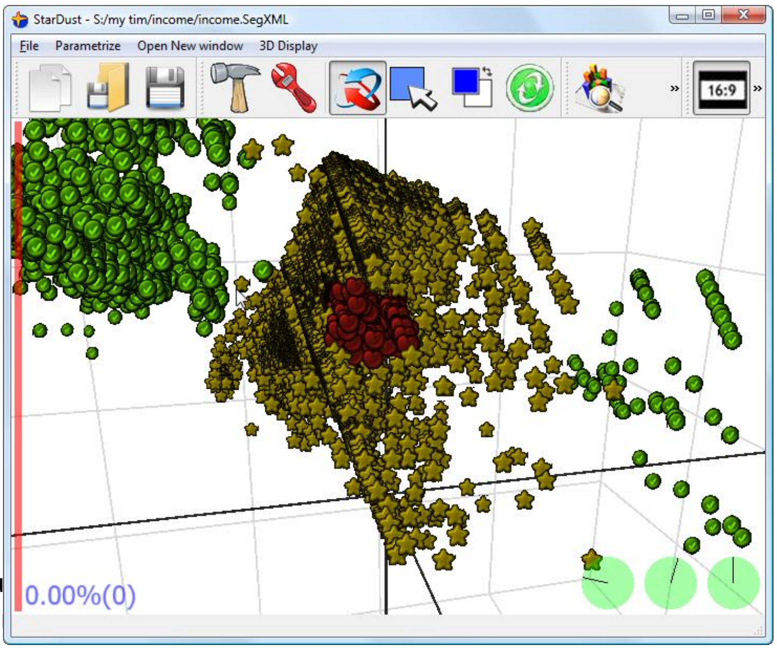
All the individuals inside the "CentralCloud" group are now represented with "yellow stars".
Let's create another group based on mouse selection. We will call this new group "Core". Here is an illustration:



Meta-Groups:

	D DISPLAY	Motif	Color	Point Size
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| CentralCloud	☑+☐-	★	🟨	1
4| Core	☑+☐-	●	🟥	1

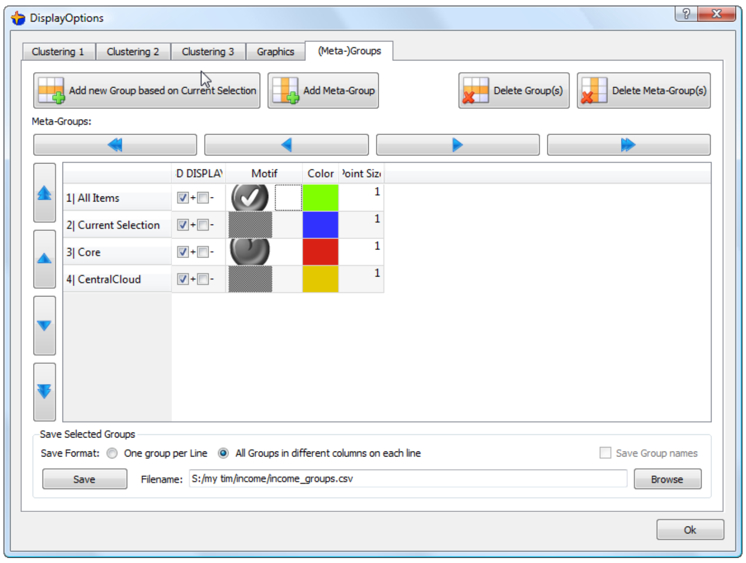
Save Selected Groups
Save Format: ( ) One group per Line (●) All Groups in different columns on each line ☐ Save Group names
Save Filename: S:/my tim/income/income_groups.csv Browse



u
defined by the setting of the group that is situated i

In the example above, the "Core" group is placed on a row that is below the "CentralCloud" row. The "Core" group has thus a higher priority than the "CentralCloud" group. The look of the individuals that belongs to both groups ("CentralCloud" and "Core") will only be defined by the settings of the "Core" group.

Let's exchange the rows "CentralCloud" and "Core"! Select the "core" group and click on the button. Replace the "star motif" with the "null motif" ▨. We obtain:



DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

Add new Group based on Current Selection | Add Meta-Group | Delete Group(s) | Delete Meta-Group(s)

Meta-Groups:

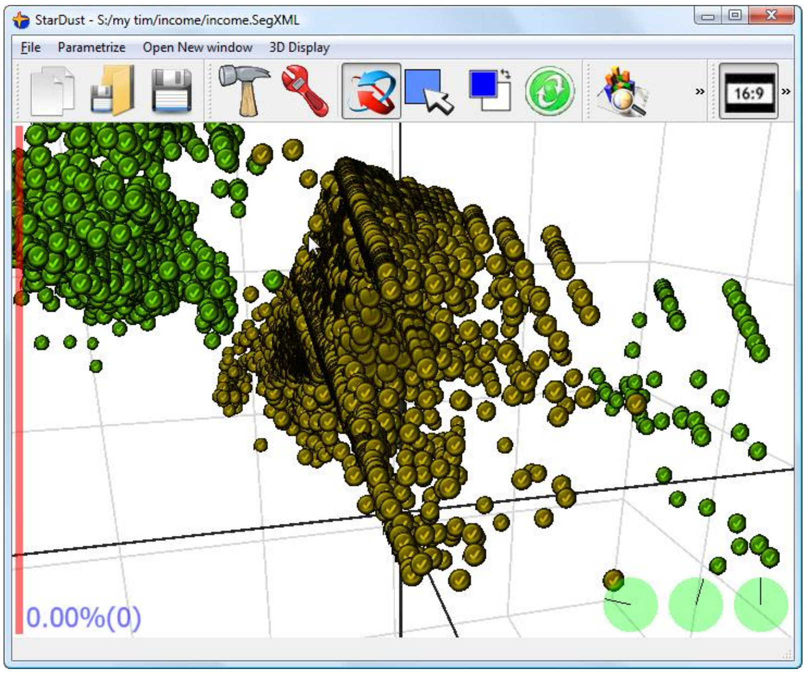
	D DISPLAY	Motif	Color	Point Siz
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| Core	☑+☐-	●	🟥	1
4| CentralCloud	☑+☐-	▨	🟨	1

Save Selected Groups

Save Format: ( ) One group per Line [✓ All Groups in different columns on each line] ☐ Save Group names

Save | Filename: S:/my tim/income/income_groups.csv | Browse

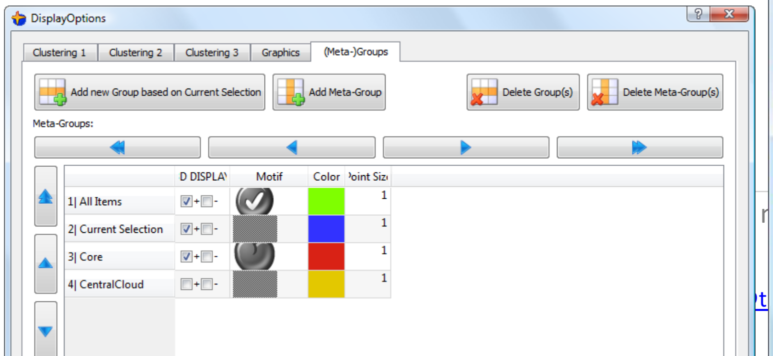
Ok



Since the "CentralCloud" Group is now lower than the "Core" group, it has the priority and thus all the points are now in yellow. However, we did select the the "null motif" ▨ for the "CentralCloud" Group. It means that the "CentralCloud" Group has NO influence on the motif of the points. The motif is thus decided based on the membership or not to the "Core" group. Indeed, you can see that some points have a "heart motif" ● .

You can also completely "de-activate" a group by disabling the "+" checkbox of the group inside the column "3D Display" (actually, 99% of the time, this is the "3D Display" column that you have to manipulate but on some rare occasions, you can use another column: see section 5.3.2.1. for more information).

The groups that have a disabled "+" checkbox have no influences at all on the display. For example, this is exactly "as if" the "CentralCloud" group does not exist:



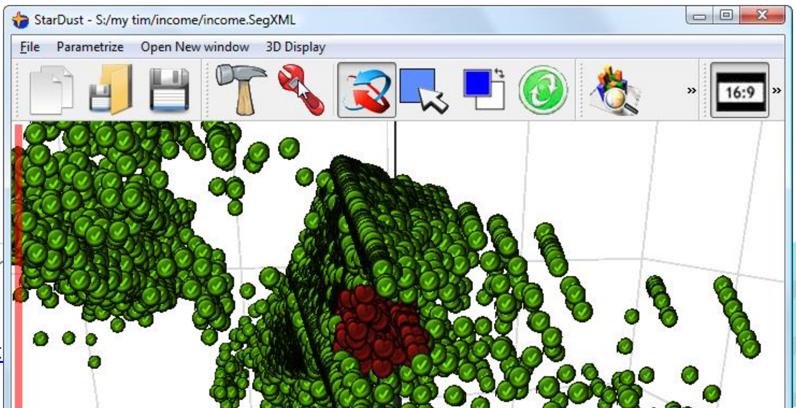
DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

Add new Group based on Current Selection | Add Meta-Group | Delete Group(s) | Delete Meta-Group(s)

Meta-Groups:

	D DISPLAY	Motif	Color	Point Siz
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| Core	☑+☐-	●	🟥	1
4| CentralCloud	☐+☐-	▨	🟨	1



5.3.1.3. Manipulating meta-Groups

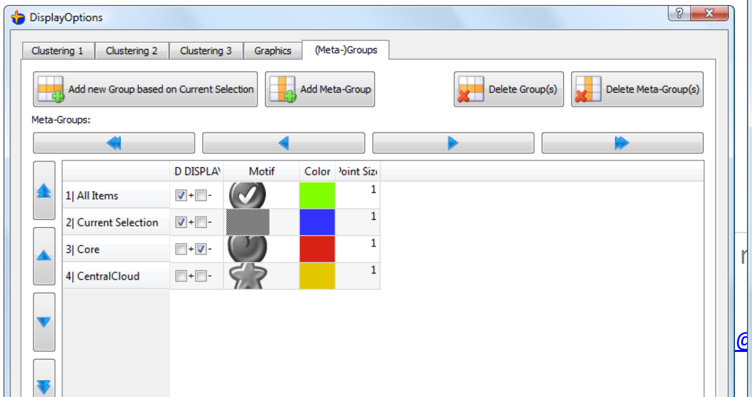
Meta-Goups are represented inside the table by columns.

You can add meta-group at anytime by clicking on the Add Meta-Group button: a new column will appear. You can change the name of the meta-group by double-clicking the column header.

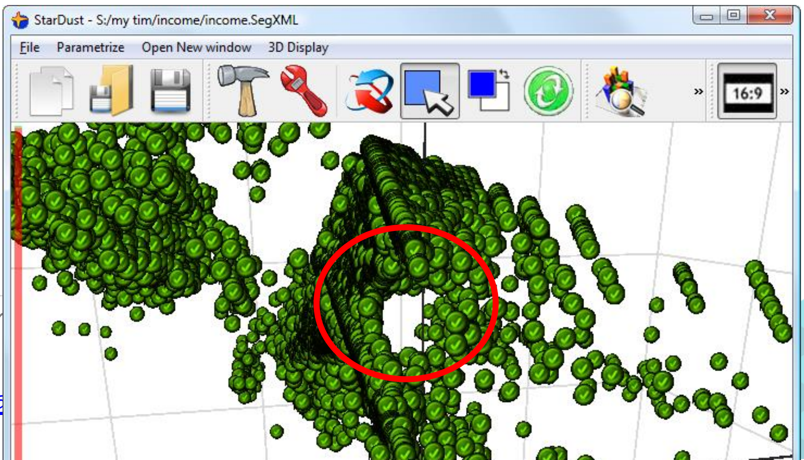
One meta-group that is more important than the other is the (default) meta-group "3D Display". By default, the individuals that belong to the "3D Display" meta-group are displayed on the main 3D view. The other individuals are not represented at all.

To know which individuals belong to the (default) meta-group "3D Display", you have to analyze the column with the label "3D display". All the individuals that belong to groups marked with the "+" checkbox enabled will be <u>added</u> to the meta-group. All the individuals that belong to groups marked with the "-" checkbox enabled will be <u>subtracted</u> to the meta-group. The order of the additions and the subtractions is important.

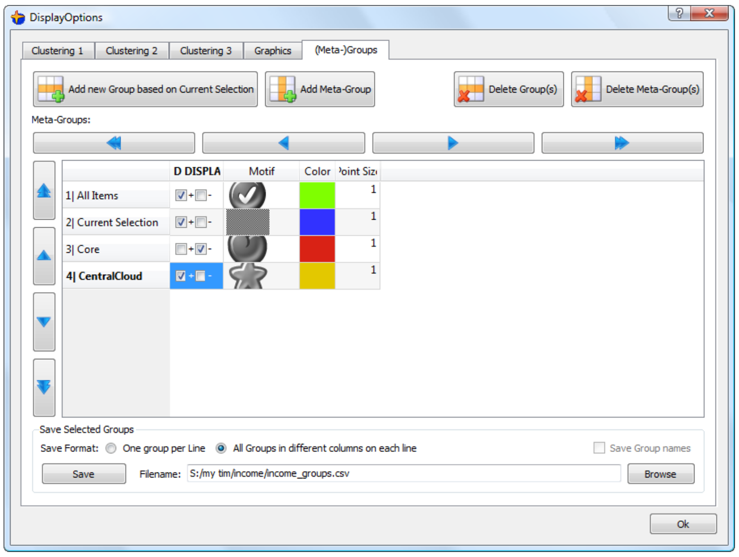
The same rule as in the previous section (5.3.1.2.) applies here: "*the lower in the table, the higher the priority*". For example:



	D DISPLAY	Motif	Color	Point Size
1| All Items	✓+ ☐-		🟩	1
2| Current Selection	✓+ ☐-	▨	🟦	1
3| Core	☐+ ✓-		🟥	1
4| CentralCloud	☐+ ☐-		🟨	1



In the example above, the “3D Display” meta-group is composed by “All Items” minus the “Core” group. This subtraction produces the “hole” that you can see above. Here is another example:


DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

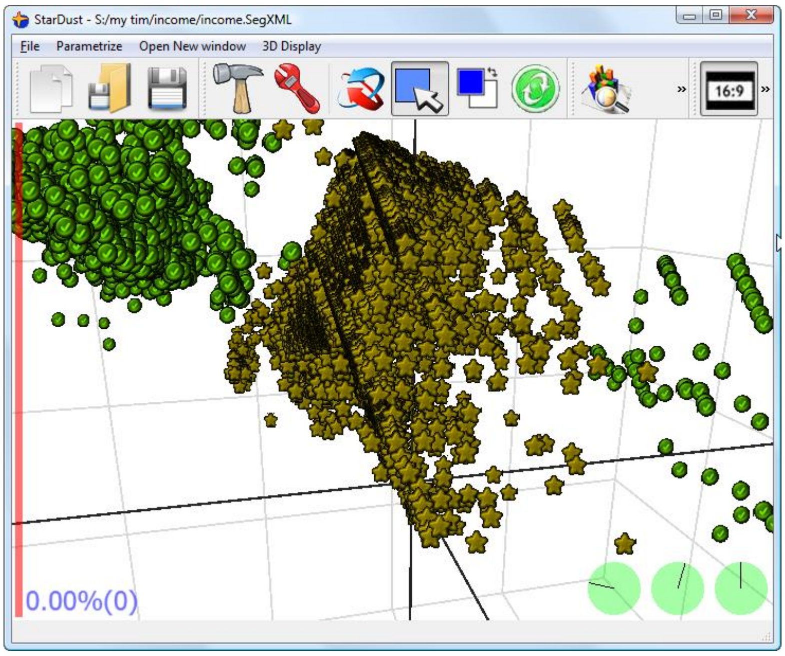
Meta-Groups:

	D DISPLA	Motif	Color	›oint Siz
1| All Items	☑+☐-	✓	🟩	1
2| Current Selection	☑+☐-	▨	🟦	1
3| Core	☐+☑-	●	🟥	1
4| CentralCloud	☑+☐-	★	🟨	1

Save Selected Groups

Save Format: One group per Line [✓ All Groups in different columns on each line] ☐ Save Group names

Filename: S:/my tim/income/income_groups.csv

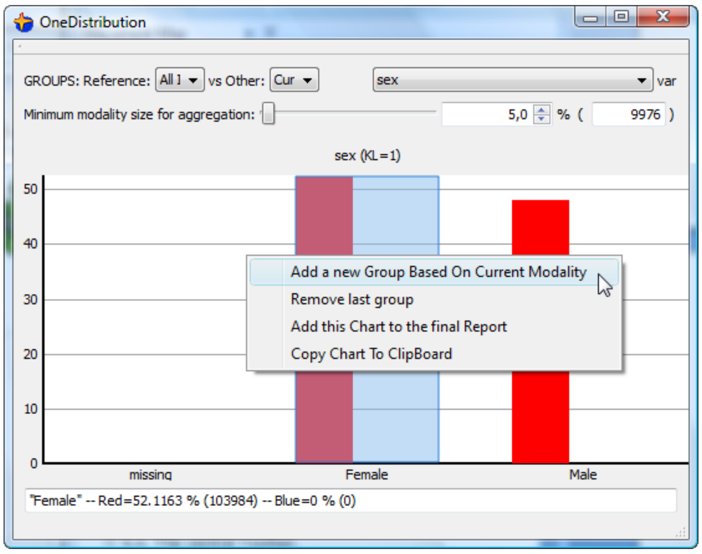


In the example above, the “3D Display” meta-group is composed by “All Items” minus the “Core” group, plus the “CentralCloud” group. The subtractions and the additions are performed in a “top-to-bottom” manner.

5.3.1.4. Creating Groups based on modalities of variables

You can create groups based on the mouse-selection as described in the previous section 5.3.1.1 but you can also create groups based on specific values of a variable. For example, to create a group that contains all the women:

1. click on the button.
2. select the “sex” variable.
3. select the “Female” modality.
4. Right-click the graphic and select “Add new group based on current modality”:


OneDistribution
GROUPS: Reference: All I vs Other: Cur
sex var
Minimum modality size for aggregation: 5,0 % (9976)
sex (KL=1)
Add a new Group Based On Current Modality
Remove last group
Add this Chart to the final Report
Copy Chart To ClipBoard
missing
Female
Male

"Female" -- Red=52.1163 % (103984) -- Blue=0 % (0)

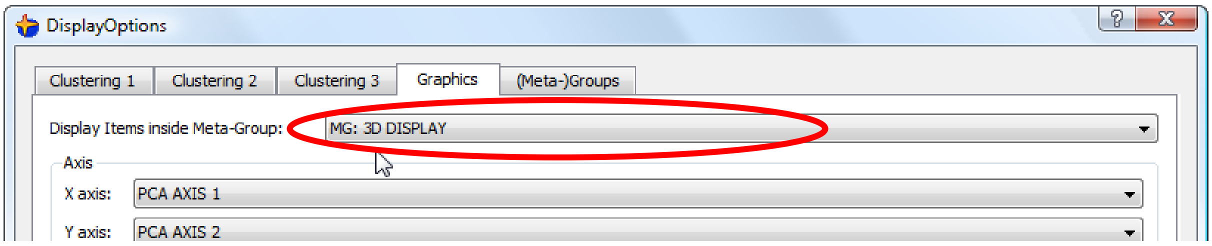
5.3.1.5. Creating Groups based on the segments

Click the `Create Groups based on Ward solution and Visible Items` button inside the "Segmentation 3" tab inside the Parameter Window (see section 5.2.5 on how to open the parameter window).

5.3.2. The graphic tab

5.3.2.1. (Meta-)Group selection for Display

By default, the individuals that belong to the "3D Display" meta-group are displayed on the main 3D view. The other individuals are not represented at all. You can change the default meta-group used to select the displayed individuals here:



DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

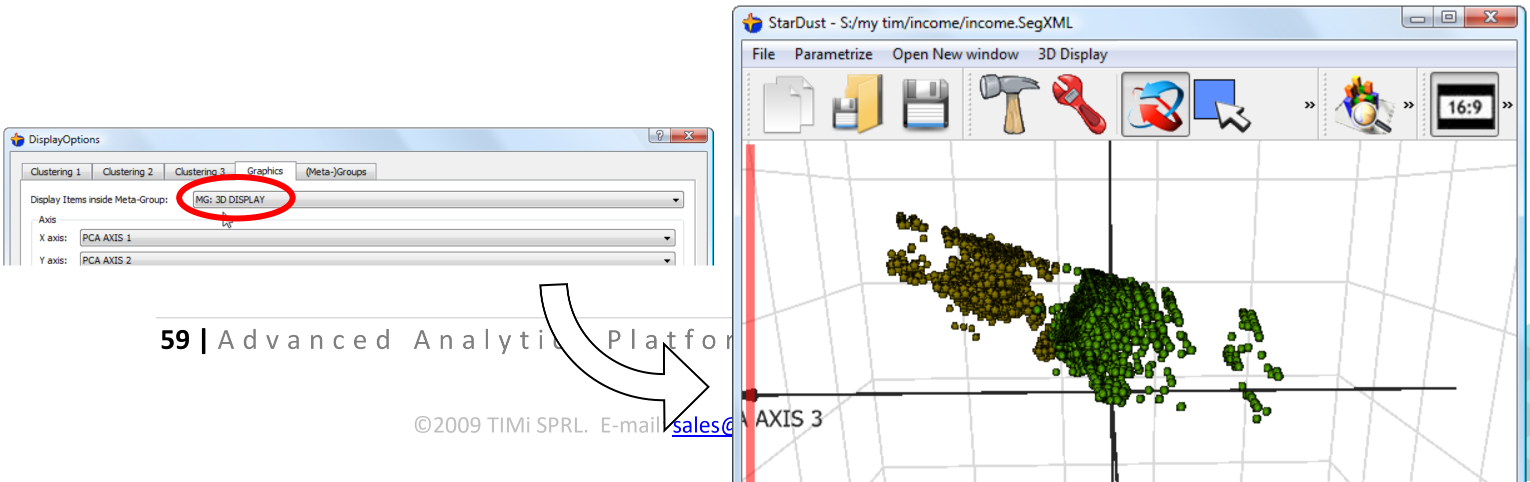
Display Items inside Meta-Group: MG: 3D DISPLAY

Axis

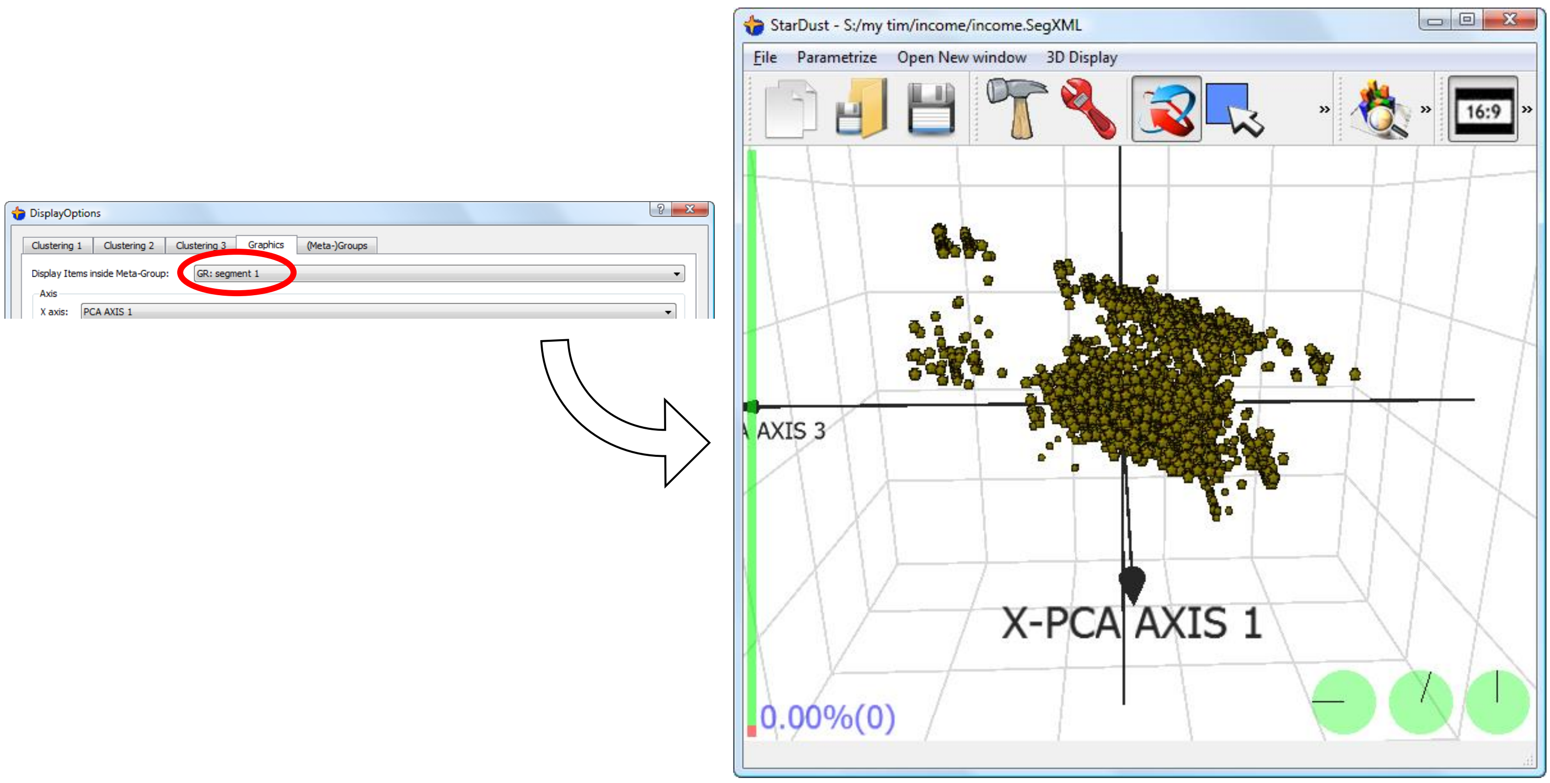
X axis: PCA AXIS 1

Y axis: PCA AXIS 2

This setting allows you to "zoom" on specific part of the space. For example, we can define a new group named "segment1" based on the mouse selection like illustrated here (the individuals inside the "segment1" group are "yellow stars"):



... and we can “zoom-in” inside the “segment1” group:


DisplayOptions

Display Items inside Meta-Group: GR: segment 1

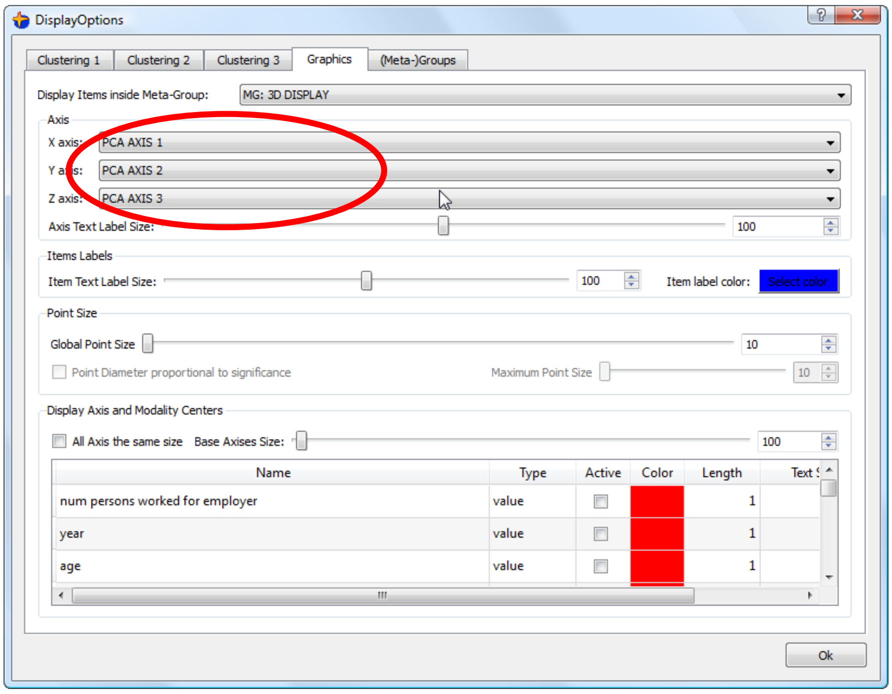
Axis

X axis: PCA AXIS 1

For even more powerful “zooming” options, see section 5.4.2.

5.3.2.2. Axises

You can change the 3 axises of the 3D display here:


DisplayOptions

Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

Display Items inside Meta-Group: MG: 3D DISPLAY

Axis
X axis: PCA AXIS 1
Y axis: PCA AXIS 2
Z axis: PCA AXIS 3
Axis Text Label Size: 100

Items Labels
Item Text Label Size: 100 Item label color: Select color

Point Size
Global Point Size 10
Point Diameter proportional to significance Maximum Point Size 10

Display Axis and Modality Centers
All Axis the same size Base Axises Size: 100

Name	Type	Active	Color	Length	Text S
num persons worked for employer	value	☐	🟥	1	
year	value	☐	🟥	1	
age	value	☐	🟥	1	

Ok

The PCA axises are the result of the PCA computation. You can setup the PCA computation.

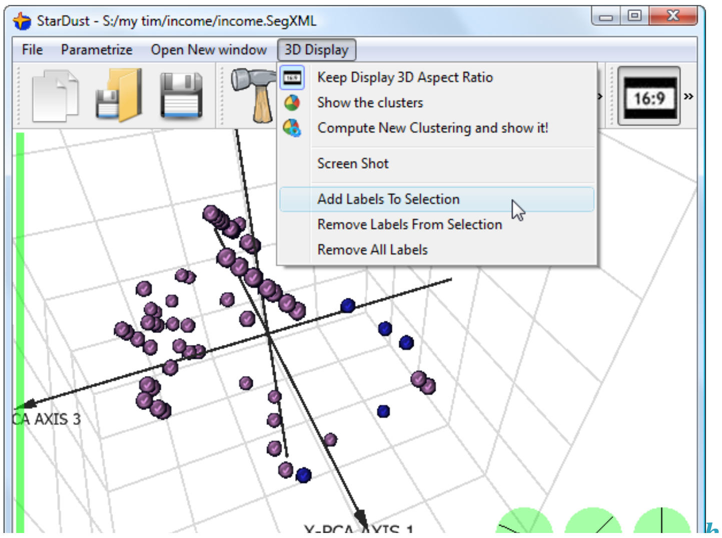
5.3.2.3. Item Labels

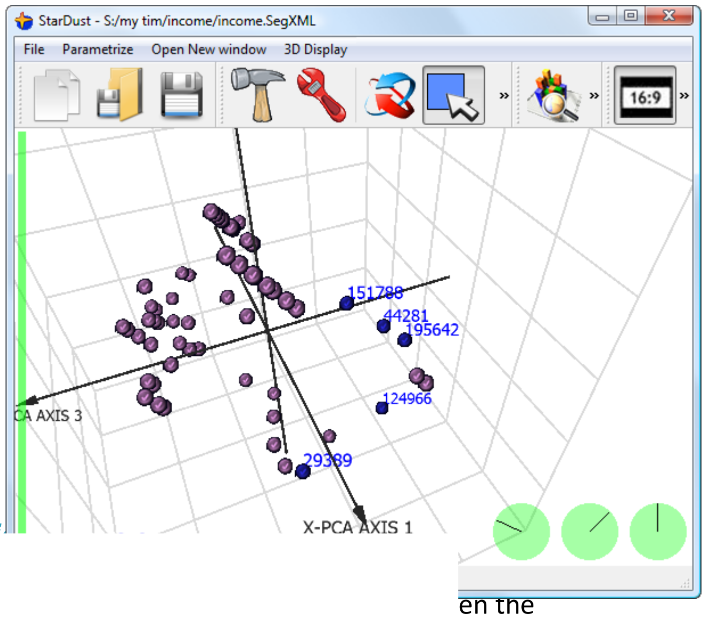
You can display inside the main 3D view the primary key associated with 3D each point. Inside the “graphic” tab of the “parameter window”, you can configure the size and the colour of the labels.

To add labels to your 3D display:

1. select with your mouse the points for which the primary keys must be displayed.
2. click “*Add Labels to Selection*” inside the drop-down-menu “*3D Display*”.

See illustration:





he 3D vi

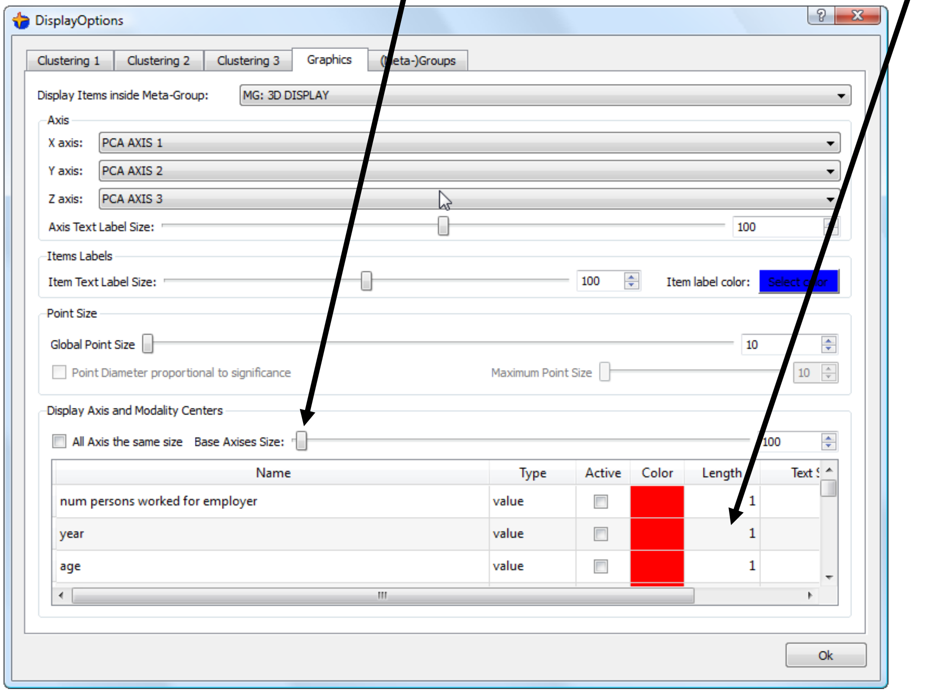
en the

3 axises used for display as defined in section 5.3.1.1. are “PCA” axises).

For more information on how the “original axis” of the dataset are projected inside the reduced spaced, please refer to section 2.2.5.

Using the “Axis” table, you can:

- show/hide an axis.
- change the size and position of the axis label.
- change the length of the vectors representing an original axis. To do that, you have a global “length-control” here: ... and a “per-axis” control here:


DisplayOptions

Display Items inside Meta-Group: MG: 3D DISPLAY

Axis
X axis: PCA AXIS 1
Y axis: PCA AXIS 2
Z axis: PCA AXIS 3
Axis Text Label Size: 100

Items Labels
Item Text Label Size: 100
Item label color: Select color

Point Size
Global Point Size 10
Point Diameter proportional to significance
Maximum Point Size 10

Display Axis and Modality Centers
All Axis the same size
Base Axises Size: 100

Name	Type	Active	Color	Length	Text S
num persons worked for employer	value	☐	🟥	1	
year	value	☐	🟥	1	
age	value	☐	🟥	1	

5.4. Configuration of the PCA analysis

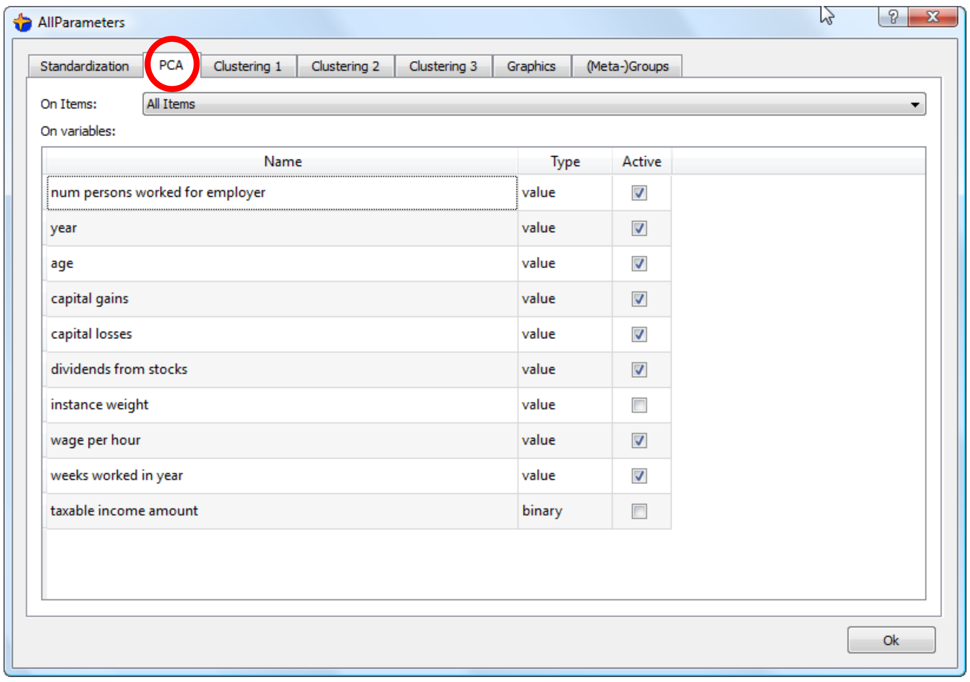
The PCA is used to:

1. <u>Do dimensionality reduction:</u> the main 3D view of Stardust is the result of a PCA analysis.
2. <u>Create good “distance-definition”:</u> the segmentation is based on the K-means algorithm that depends on the PCA analysis.

The **<u>same</u>** PCA is used for both usage: dimensionality-reduction and segmentation.

Since the **<u>same</u>** PCA is used for both usage, the segmentation algorithm should find the same segments that you can “guess” when looking at the 3D display.

The PCA is configured inside the “PCA” tab inside the parameter window (this tab is not accessible with the button, only with the button):


AllParameters

Standardization | [✓ PCA] | Clustering 1 | Clustering 2 | Clustering 3 | Graphics | (Meta-)Groups

On Items: All Items

On variables:

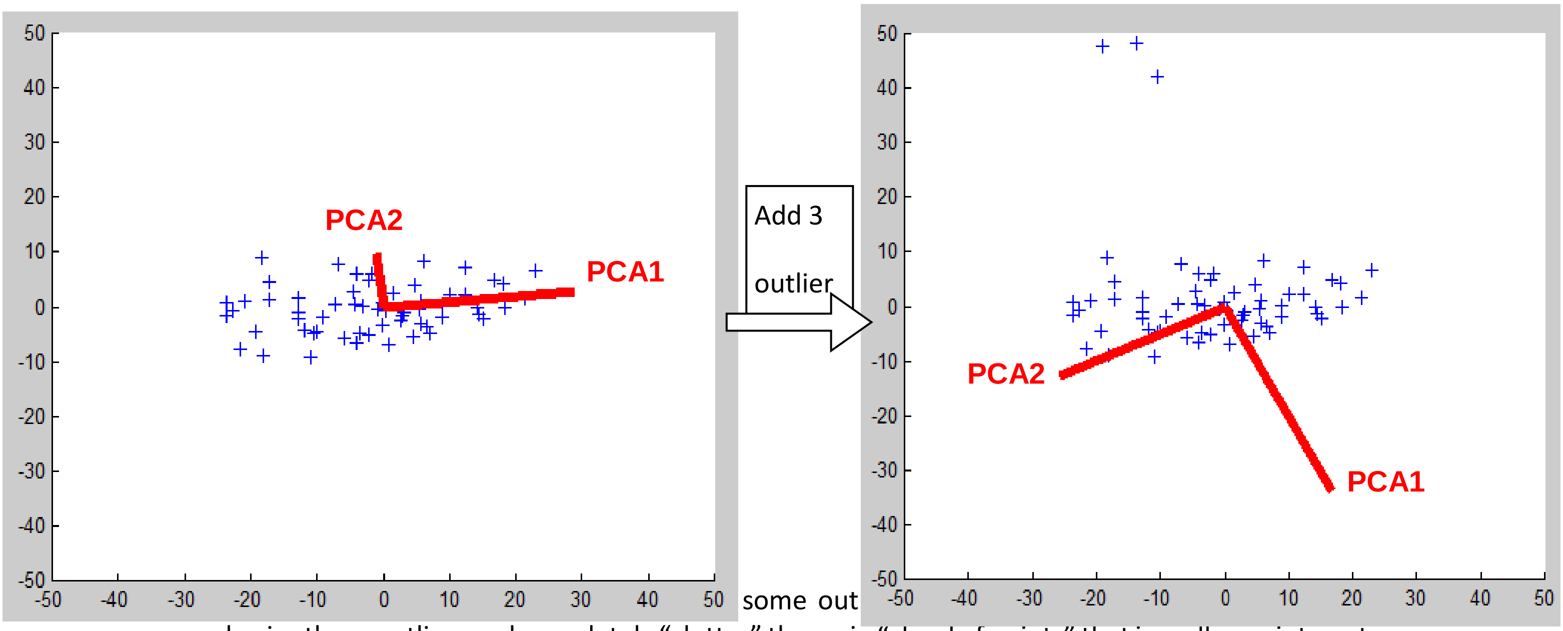
Name	Type	Active
num persons worked for employer	value	✓
year	value	✓
age	value	✓
capital gains	value	✓
capital losses	value	✓
dividends from stocks	value	✓
instance weight	value	☐
wage per hour	value	✓
weeks worked in year	value	✓
taxable income amount	binary	☐

Ok

5.4.1. Detecting and removing outliers using the PCA

To remind you: The first PCA axises are defining priviliged directions that are the best directions on which to « project » our “cloud of points” in order to « loose » as little information as possible.

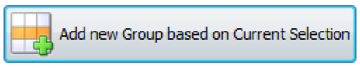
The PCA algorithm has one major drawback that we can use to our advantage: it’s “*sensitive to outliers*”. What does it means? It means that if the “cloud of points” on which the PCA is based contains some outliers, then the PCA algorithm fails and gives as output as set of direction that shows you the outliers. For example: Let’s go back to the example of section 2.1.1 and add 3 more “outlier points” inside the dataset used to construct the PCA:

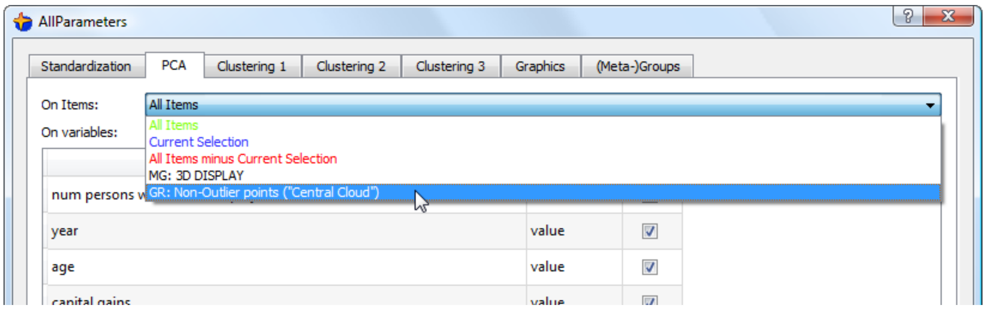


some out emphasize these outliers and completely “clutter” the main “cloud of points” that is really our interest. In such situation, we can:

1. Investigate the outliers:

a. what's the profile of the outliers? You can use the charts described in sections 5.2.10, 5.2.11 and 5.2.12.
b. What are the primary keys of these outliers? Please refer to section 5.3.2.3.

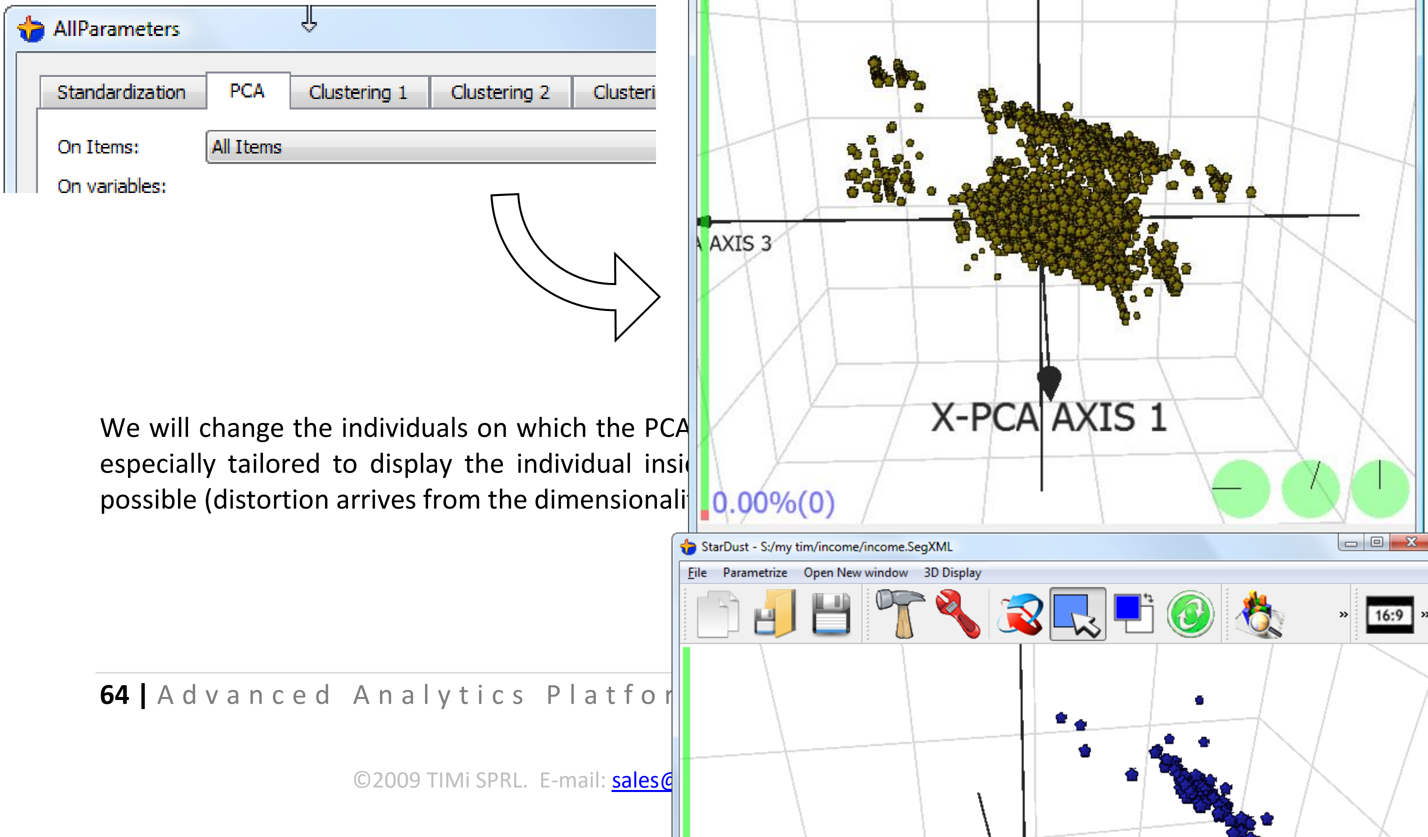
2. Create a new PCA on a "reduced" dataset that does not contain anymore the outliers: To do that:
   a. select with your mouse the "non-outlier" points (see section 5.2.7 to know how mouse selection works)
   b. create a new group based on your selection, using the  button (see section 5.3.1.1 to create a group).
   c. Select this new group as the dataset to analyse to create the PCA: click here:



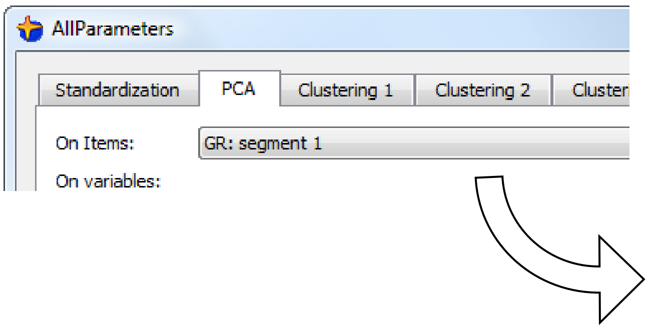
It's important to remove the outliers from the PCA computation before computing any segmentation, otherwise the segmentation will only give, as a result, some segments that emphasize the fact that an individual is an outlier or not.

5.4.2. "Zooming" on the data using the PCA

We will start from the basic "zooming" example given in section 5.3.2.1.



We will change the individuals on which the PCA
especially tailored to display the individual insi
possible (distortion arrives from the dimensionali



On Items: GR: segment 1

On variables:

We can see that the "segment1" group contains in reality 2 sub-segments. These sub-segments were not visible when the PCA was computed on the whole database.

Using the PCA in this way, we can really "zoom-in" in some specific part of the space and easily explore our database in a *real multivariate way* to discover new "insights" about our customers.

In opposition to simple OLAP softwares that are showing your database "*variables per variables*" (or "columns per columns") in a UNIvariate (or, some time, BIvariate) way, with StarDust, we can do "*MULTIvariate exploration*" because all the original axises of the database are represented at the same time on the 3D graph. You can even "see" where are "placed" the original axises using techniques described in section 3.2.5 and 5.3.2.4.

The real-time and powerful "*MULTIvariate exploration*" of your database with StarDust is a unique functionality amongst all currently available datamining tools.

Note that you must be careful when using the PCA to "zoom-in" inside your data because the same PCA is also used to create the segmentation.

This means that, if the PCA is currently used to "zoom-in" on the "Segment1" group (like here above), then all the newly created segmentation will emphasize the segments that are contained inside the "Segment1" group. This is not what you want.

However, it is very common (and strongly recommended) to change the default group (that is "All Items") on which the PCA is built to remove from the PCA analysis the "outliers": see previous section 5.4.1 at this subject.

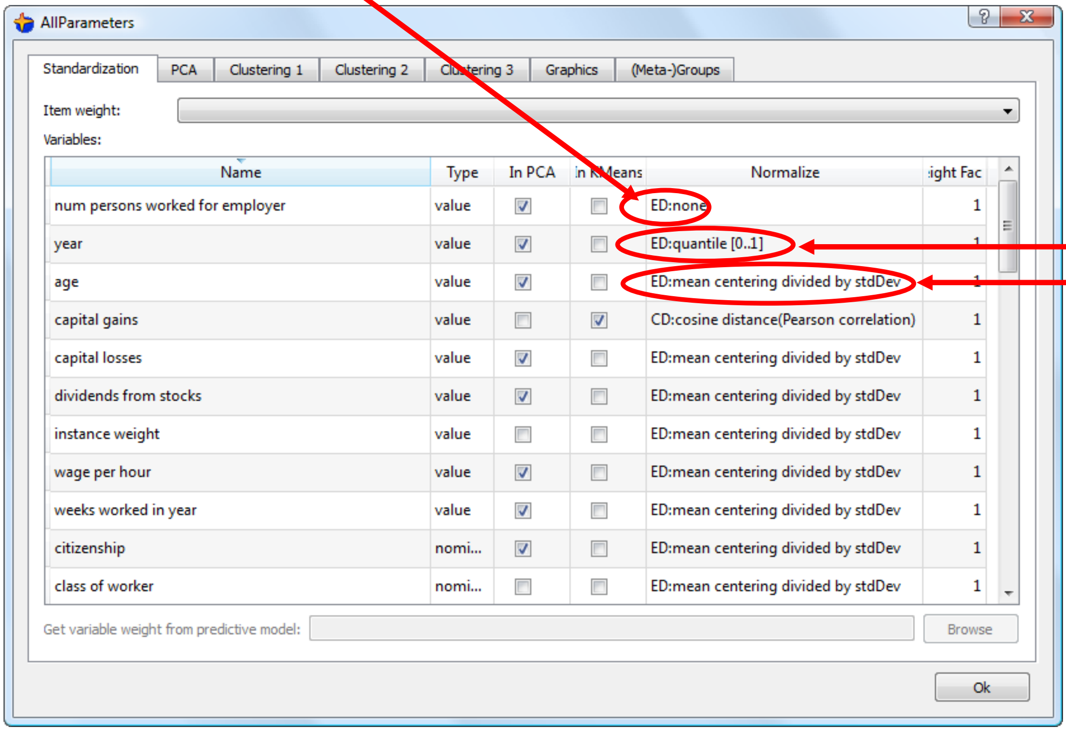
5.5. How to create a segmentation?

5.5.1. Definition of the Distance to use inside the segmentation

A segmentation is based on a "distance-definition".

Using StarDust, you can create very complex distance-definition. For example, the following axis can be included inside you distance-definition:

- On all continuous variables:
    - Non-normalized original axis X_i of the data: select "*ED:none*" inside the "*Normalize*" column inside the large table on the "*Standardization*" tab inside the parameter window (this tab is NOT accessible with the button, only with the button. See illustration:



Name	Type	In PCA	In KMeans	Normalize	ight Fac
num persons worked for employer	value	✓	☐	ED:none	1
year	value	✓	☐	ED:quantile [0..1]	1
age	value	✓	☐	ED:mean centering divided by stdDev	1
capital gains	value	☐	✓	CD:cosine distance(Pearson correlation)	1
capital losses	value	✓	☐	ED:mean centering divided by stdDev	1
dividends from stocks	value	✓	☐	ED:mean centering divided by stdDev	1
instance weight	value	☐	☐	ED:mean centering divided by stdDev	1
wage per hour	value	✓	☐	ED:mean centering divided by stdDev	1
weeks worked in year	value	✓	☐	ED:mean centering divided by stdDev	1
citizenship	nomi...	✓	☐	ED:mean centering divided by stdDev	1
class of worker	nomi...	☐	☐	ED:mean centering divided by stdDev	1

Get variable weight from predictive model:

-
    - Normalized-axis. There are two type of normalization:
        - Standard Normalization for inclusion inside an Euclidean-Distance:

$$X_{i,Std} = \frac{X_i - \bar{X}_i}{s(X_i)}$$

... where $X_{i,Std}$ is the ith column of the dataset that has been normalized

X_i is the ith column of the dataset

$\bar{X}_i$ is the mean of the ith column of the dataset

$s(X_i)$ is the standard devidation of the ith column of the dataset

Select "*ED: mean centering divided by StdDev*" in the same table:

- Quantile Normalisation: $X_{i,Quantile} = q(X_i)$

Select "*ED:quantile [0..1]*" in the same table:
q(x) is an operator that gives as output a number between 0 and 1.
The *q(x)* operator on a column *C* of the database is computed this way:

- Sort in increasing order all the numbers in *C* and remove all the duplicates. We obtain a new "sorted" column $\tilde{C}$.
- $q(x) = \frac{\text{position of } x \text{ inside } \tilde{C}}{\text{number of elements in } \tilde{C}}$

Thus, *q(x)* is zero when *x* is the smallest number inside the column *C* and *q(x)* is one when *x* is the largest number inside the column *C*.

The Euclidean-distance between two rows of the dataset (between the two points A and B) is $dist_E(A,B)$ and is defined this way:

$$dist_E(A,B) = dist_V(A,B) + dist_N(A,B)$$

$$\begin{aligned} dist_V(A,B) = & \sum_{i \in (V \cap E)} w_i (X_i(A) - X_i(B))^2 + \\ & \sum_{i \in (S \cap E)} w_i (X_{i,Std}(A) - X_{i,std}(B))^2 + \\ & \sum_{i \in (Q \cap E)} w_i (X_{i,Quantile}(A) - X_{i,Quantile}(B))^2 + \\ & \sum_{i \in P} w_{i,PCA} (PCA_{i,Std}(A) - PCA_{i,Std}(B))^2 \end{aligned}$$

$$dist_N(A,B) = \sum_{i \in (N \cap E)} w_i \; eq(X_i(A), X_i(B))^2$$

... where $dist_V(A,B)$ is a distance based on variables that only contains continuous numbers (values)
$dist_N(A,B)$ is a distance based on nominal variables only.
E is the set of "active" variables to include inside the Euclidean distance-definition.
V is the set of non-normalized continuous variables
S is the set of continuous variables that have been normalized using the "Standard" normalization
Q is the set of continuous variables that have been normalized using the "Quantile" normalization
N is the set of Nominal variables
P is the set of "active" PCA axises
eq(x,y) is an operator that returns one if the string 'x' equals 'y' and zero otherwise.
$X_i(A)$ is the content of the column 'i' and row 'A' of the dataset
$X_{i,Std}$ and $X_{i,Quantile}$ have been defined on the previous page.
$PCA_{i,Std} = \dfrac{PCA_i - \overline{PCA_i}}{s(PCA_i)}$ is the PCA_i axis after a "Standard" normalization
w_i and $w_{i,PCA}$ are user-specified (column-)weights.

The default values are: the sets *V,S,Q,E* are empty, the set *S* contains all the continuous variables, the set *P* contains the first ten PCA axises, the weights w_i and $w_{i,PCA}$ are all one.

If some variables are included inside a "cosine-distance" (the *C* set is non-empty), then the distance-definition between two points A and B is:

$$dist_{E+C}(A,B) = \sqrt{dist_V(A,B)} + dist_N(A,B) + dist_{\cos}(A,B)$$

$$dist_{\cos}(A,B) = \frac{w_{\cos}}{w_{euclidean}} \cos^{-1}\left(\frac{\sum_{i \in C} (X_{i,std}(A) - \tilde{A}_C)(X_{i,std}(B) - \tilde{B}_C)}{s(A_C)\, s(B_C)} \right)$$

$$w_{euclidean} = \frac{\tilde{w}_e |E|}{}$$

...where

... and where *C* is the set of variables that are, at the same time, included inside the "cosine-distance" (column "*normalize*") and that are "active" (column "*In K-Means*").

$|C|$ is the number of variables inside the set *C*

$w_{\cos}$ is a user-defined parameter that specified the relative weight of the cosine-distance compared to the Euclidean-distance.

$\tilde{w}_e$ is the average of the w_i and $w_{i,PCA}$ user-specified weights included inside the Euclidean-distance

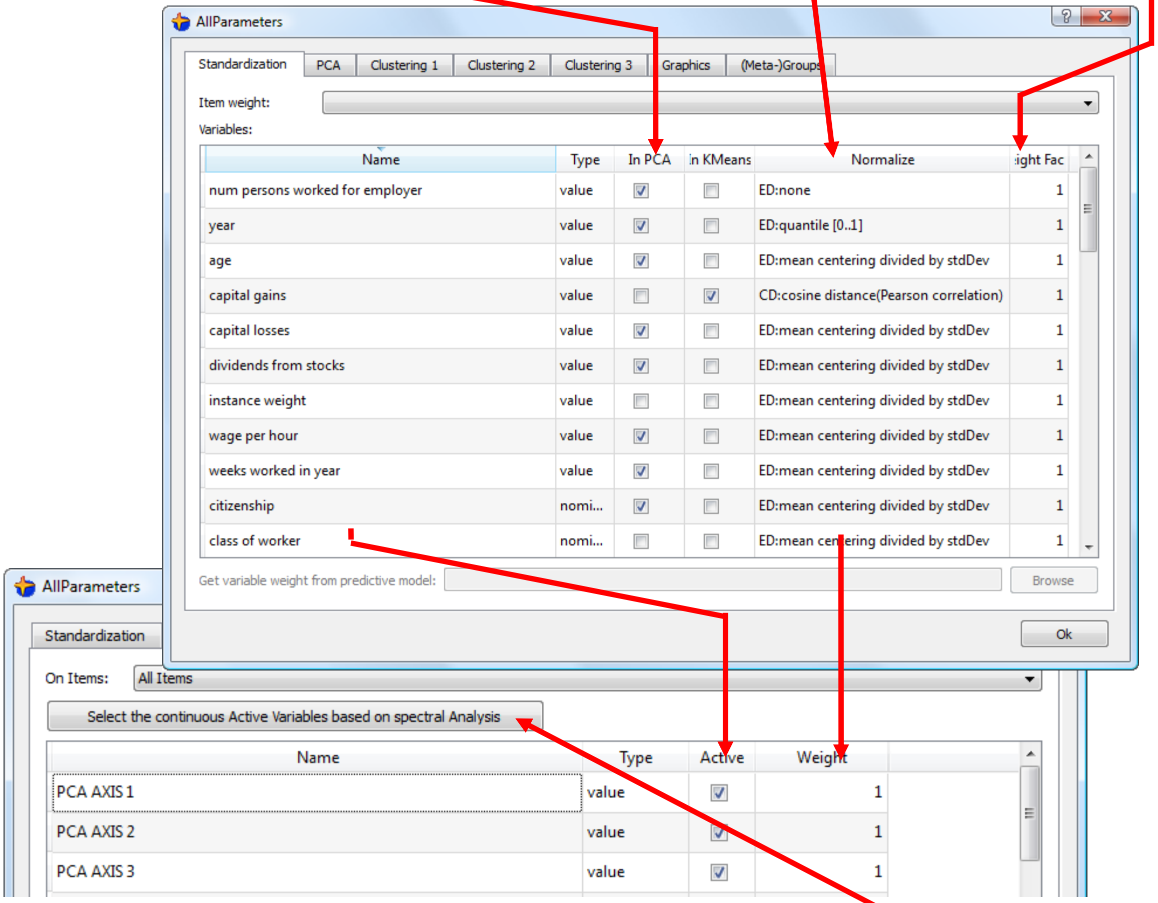
$\tilde{A}_C$ is the mean of a vector composed by the value of the columns $X_{i,std}(A)$ with $i \in C$ on the row A

$s(A_C)$ is the Standard deviation of a vector composed by the value of the columns $X_{i,std}(A)$ with $i \in C$ on the row A

The default values are: the set C is empty, the weight $w_{\cos} = 1$

You can define the sets *E,C* here: You can define the sets *V,S,Q,C* here: You can define w_i here:

You



Item weight:

Variables:

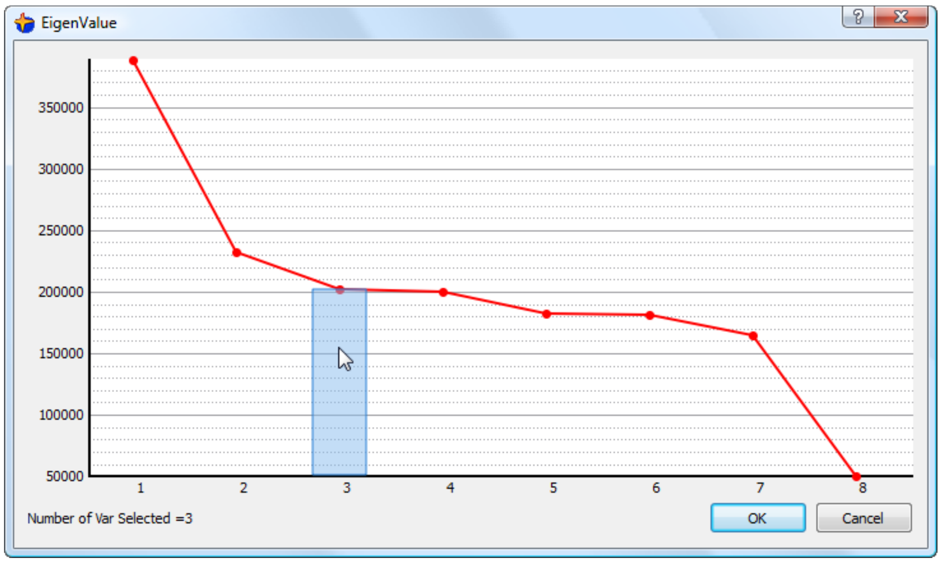
Name	Type	In PCA	In KMeans	Normalize	ight Fac
num persons worked for employer	value	✓		ED:none	1
year	value	✓		ED:quantile [0..1]	1
age	value	✓		ED:mean centering divided by stdDev	1
capital gains	value		✓	CD:cosine distance(Pearson correlation)	1
capital losses	value	✓		ED:mean centering divided by stdDev	1
dividends from stocks	value	✓		ED:mean centering divided by stdDev	1
instance weight	value			ED:mean centering divided by stdDev	1
wage per hour	value	✓		ED:mean centering divided by stdDev	1
weeks worked in year	value	✓		ED:mean centering divided by stdDev	1
citizenship	nomi...	✓		ED:mean centering divided by stdDev	1
class of worker	nomi...			ED:mean centering divided by stdDev	1

On Items: All Items

Name	Type	Active	Weight
PCA AXIS 1	value	✓	1
PCA AXIS 2	value	✓	1
PCA AXIS 3	value	✓	1

To help you to select the set *P* of the active PCA axises, you can click here:

When you click the Select the continuous Active Variables based on spectral Analysis button, the following window appears:



Number of Var Selected =3

Please refer to the section 2.2.5 to know how to select to right number of PCA axises to include inside your “distance-definition”.

The default, initial setting for the “distance-definition” is equivalent to the following “distance-definition”:

$$dist_{E+C}(A,B) = \sum_{i=1}^{10} (PCA_{i,Std}(A) - PCA_{i,Std}(B))^2$$

5.5.2. The parameters of the K-Means algorithm

The K-means algorithm is illustrated in detail in the section 2.2.1.

Here is a brief description of the standard K-means algorithm:

1. We assign randomly one individual to each segment. These “special” individual are named “seeds”. Usually, different “seeds” will give different (but hopefully close) segmentations. Thus, there is no unique segmentation: Depending on the “seeds” you will usually find different segments.
2. We assign all the individual to the nearest segment “center”. To compute which segment is the closest one, we use the “distance-definition” as defined in the previous section.
3. We re-compute the “center” of each segment: it’s the “center of mass” of all the points/ individual inside each each segment.
4. If we have reached the maximum number of iteration $n_{KMeanIterations}$ then stop.
   If some individuals have changed from segment, return to step 2, otherwise stop.

To summarize, the objective of the K-means algorithm is to find the center C_i of each segment. The "optimal, best centers" are the centers C_i that give the lowest "Energy". The "Energy" is defined this way:

$$Energy(C_1,\ldots,C_{n_s}) = \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_{E+C}(C_i, I_j)$$

... where n_s is the number of segments (supplied by the user).

S_i is the set of individuals belonging to the ith segment

C_i is the center of the ith segment

$dist_{E+C}(A,B)$ is the distance between the individuals A and B, as defined in the previous section

The K-means algorithm tries to find the "best centers" (i.e. the centers that give the lowest energy) but, depending on the "seeds" used, it will only find solutions (i.e. centers) that are more or less close to the "best centers". This is why, it's interesting to run the K-means algorithm several times (this number of run is a user-specified parameter named $n_{KMeanRuns}$) and to select, at the end, the "centers" (i.e. the segmentation) that gave the lowest "Energy".

Stardust uses a small variation on the standard K-means algorithm named "K-means++" that generates a far better segmentation than the "normal" K-Means. The "K-means++" is the same algorithm as the standard K-means expect that the "seeds" are computed in a slightly different way. The step 1 of the standard K-means algorithm is replaced by:

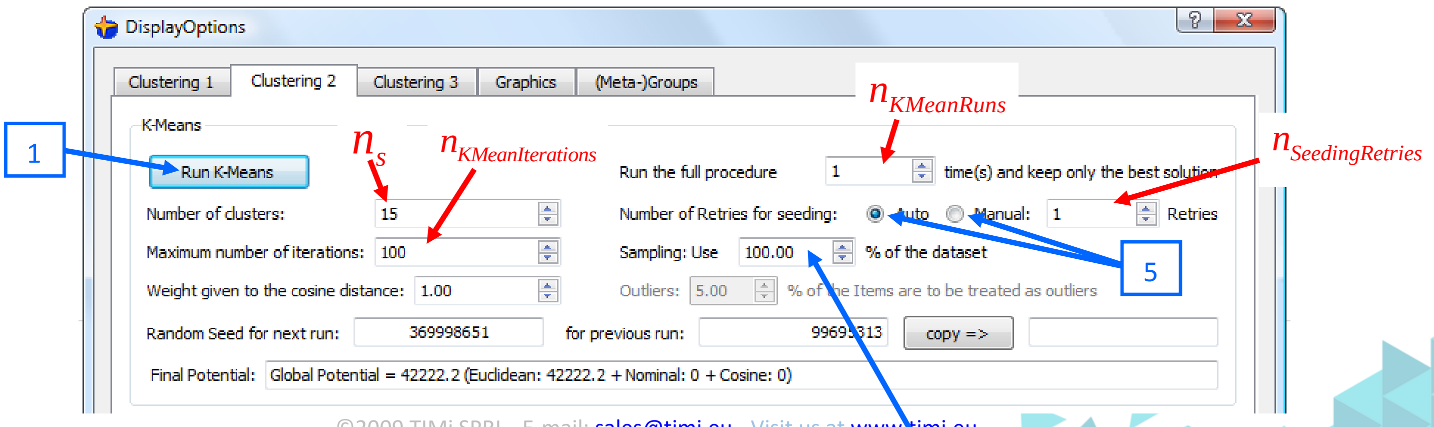
1. Choose an initial center C_1 uniformly at random amongst our dataset.
2. Choose the next center C_i, selecting $C_i = X$ with probability $\dfrac{D(X)^2}{\sum_{I \in dataset} D(I)^2}$

   ...where *I* is one Individual inside our dataset

   D(X) is the shortest distance from the point X to the closest center that we have already chosen
3. Repeat step 2 until we have choose a total of n_s centers.

The "K-means++" algorithm has also a strong random component, so it's good to run it several time (this number of run is a user-specified parameter named $n_{SeedingRetries}$) and to select, at the end, the "run" that gave the lowest $\sum_{I \in dataset} D(I)^2$ value.

The "K-Means" main parameters are defined here (the 5 controls **in blue** will be described below):



Number of clusters: 15

Maximum number of iterations: 100

Weight given to the cosine distance: 1.00

Random Seed for next run: 369998651

Run the full procedure 1 time(s) and keep only the best solution

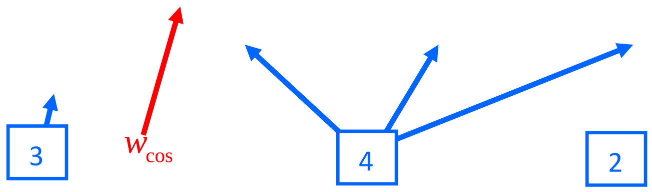
Number of Retries for seeding: [✓ Auto] Manual: 1 Retries

Sampling: Use 100.00 % of the dataset

Outliers: 5.00 % of the Items are to be treated as outliers

for previous run: 99695313

Final Potential: Global Potential = 42222.2 (Euclidean: 42222.2 + Nominal: 0 + Cosine: 0)



Here are some explanations relative to the **blue** controls in the screenshot here above:

5.5.2.1. *Control 1:* ***Run K-means*** Run K-Means

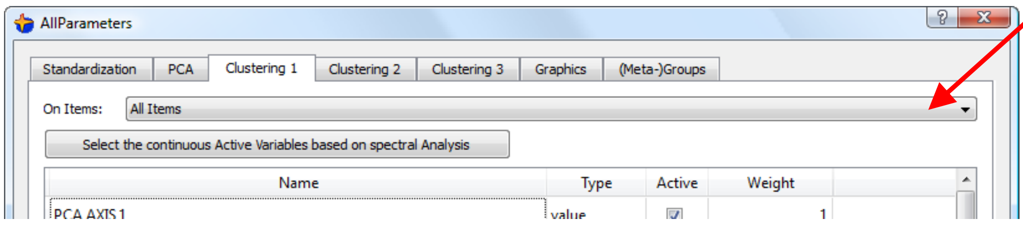
When you click the Run K-Means button, the K-means algorithm runs $n_{KMeanRuns}$ times and then "injects" the best solution found (with the lowest Energy) inside the Ward's Algorithm. See next section 5.5.3. about the Ward's Algorithm.

The Run K-Means button is completely equivalent to the button inside the toolbar in the main windows of StarDust.

5.5.2.2. *Control 2:* ***Sampling*** Sampling: Use 100.00 % of the dataset

The K-means algorithm can compute for a very long time when the dataset to analyze is very big. To reduce computation time, you can use the "sampling" parameter to reduce the size of the dataset "injected" into the K-means algorithm.

The dataset that will be sampled and "injected" into the K-means algorithm is defined here: In the ideal case, this dataset should not contain any outliers because the K-means algorithm is a little bit sensitive to outliers (this is due to the "squaring effect" of the Euclidean distance).



On Items: All Items

Name	Type	Active	Weight
PCA AXIS 1	value	✓	1

5.5.2.3. *Control 3:* ***Energy*** Final Potential: Global Potential = 42222.2 (Euclidean: 42222.2 + Nominal: 0 + Cosine: 0)

This control displays the "Energy" of the solution found by the K-means algorithm.
To remind you, the "Energy" is defined by:

$$Energy(C_1,\ldots,C_{n_s}) = \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_{E+C}(C_i, I_j) \qquad \text{(equation 1)}$$

This can be decomposed in two different ways:

1. __**When no variables are included inside a "cosine-distance"**__ (the *C* set is empty)
   We have the following "distance-definition":

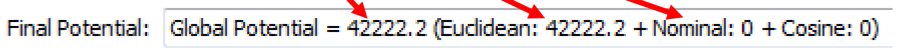
$$dist_E(A,B) = dist_V(A,B) + dist_N(A,B) \qquad \text{(equation 2)}$$

... combining equations 1 and 2, we find:

$$Energy = \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_V(C_i, I_j) + \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_N(C_i, I_j)$$

$$= Energy_{euclidean} + Energy_{nominal}$$

... which is displayed here:



Final Potential: Global Potential = 42222.2 (Euclidean: 42222.2 + Nominal: 0 + Cosine: 0)

2. **<u>When some variables are included inside a "cosine-distance"</u>** (the *C* set is non-empty)
   We have the following "distance-definition":

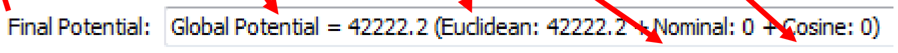
$$dist_{E+C}(A,B) = \sqrt{dist_V(A,B)} + dist_N(A,B) + dist_{\cos}(A,B) \qquad \text{(equation 3)}$$

... combining equations 1 and 3, we find:

$$Energy = \sum_{i=1}^{n_s} \sum_{j \in S_i} \sqrt{dist_E(C_i, I_j)} + \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_N(C_i, I_j) + \sum_{i=1}^{n_s} \sum_{j \in S_i} dist_{COS}(C_i, I_j)$$

$$= Energy_{euclidean} + Energy_{nominal} + Energy_{cosine}$$

... which is displayed here:



Final Potential: Global Potential = 42222.2 (Euclidean: 42222.2 + Nominal: 0 + Cosine: 0)

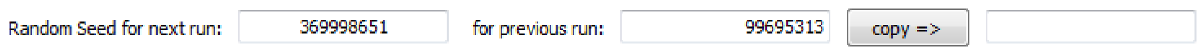
This display has several interest points:

1. It's interesting to look at the "*Global Energy*" because it's an estimation of the quality of your segmentation. The lower "*Global Energy*", the better the segmentation (at least from a pure mathematical point-of-view). This is a way to "compare" different segmentations and take the best one.

2. It's interesting to see how the "*Global Energy*" is distributed amongst the "*Euclidean Energy*", "*Nominal Energy*" and "*Cosine Energy*" parts, in order to better adjust the w_i, $w_{i,PCA}$ and $w_{\cos}$ parameters. For example, if you see that the largest portion of the "*Global Energy*" comes from the "*Cosine Energy*", you can decrease the $w_{\cos}$ parameter.
3. The "*Global Energy*" is proportional to the number of points inside your dataset. Furthermore, if there are some outliers inside your dataset, then, because of these outliers, the "*Global Energy*" will have a very high value. When you remove points from your dataset, the "*Global Energy*" will decrease. If you remove outlier points from your dataset, then the "*Global Energy*" decreases substantially. If you continue to remove (non-outlier) points from your dataset, then the "*Global Energy*" won't decrease very much. Thus, the "*Global Energy*" can be used to track if there still exists outliers inside your dataset. The presence of outliers inside your dataset deteriorates the quality of you segmentation and you should avoid them.
4. If you use a "*sampling parameter*"<100%, then, each time you run the K-means algorithm, you will use a slightly different dataset. If you are lucky enough (i.e. if you set the

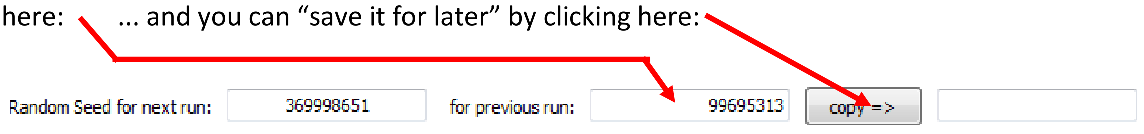
parameter $n_{KMeanRuns}$ high enough), one of these dataset will be composed only by "regular points" (i.e. It won't contain any outliers). The "*Global Energy*" of a dataset without any outliers is strongly smaller than the "*Global Energy*" of a dataset containing outliers. Thus, amongst all the segmentations results available (when $n_{KMeanRuns} >> 1$), the segmentation based on a dataset containing only "regular points" will always be selected as the final one. Thus, the sampling parameter can be used to "avoid" datasets containing outliers. A strong sampling (50%) will almost certainly remove all outliers but it will also give less information to the K-means algorithm, giving, at the end, a poor segmentation. Some trade-off is to be made. Observing the "*Global Energy*" allows you to set a good value for the "*sampling parameter*" and the $n_{KMeanRuns}$ parameter.

5.5.2.4. Control 4: Random seed

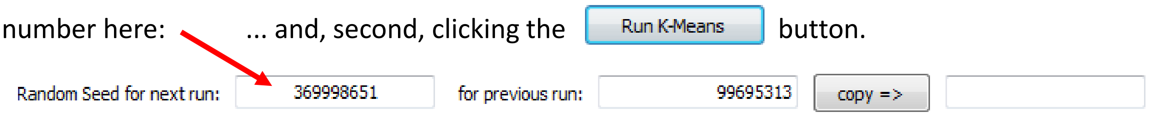

Random Seed for next run: 369998651 for previous run: 99695313 copy =>

The "K-Means++" algorithm is an algorithm with a strong "random" component. Inside Stardust, all the random elements of the "K-means++" are based on a "*pseudo-random number generator*". Such kind of generators generates sequences of random number. All the "random decisions" of the "K-means++" algorithm are taken based on the generated random sequence of number. The particularity of "*pseudo-random number generators*" is that you must initialize them using a first user-supplied number. This number is named "*random seed*". The same "*random seed*" will always produce exactly the same sequence of random number. It means that if you use the same "*random seed*" number for two different runs of the "K-Means++" algorithm, you will obtain exactly the same segmentation. In other words, the "*random seed*" number completely defines the segmentation delivered by the "K-Means++" algorithm.

You can "save" good segmentations by saving the corresponding "*random seed*" number. The "*random seed*" number that has just been used to produce the current segmentation is visible here: ... and you can "save it for later" by clicking here:

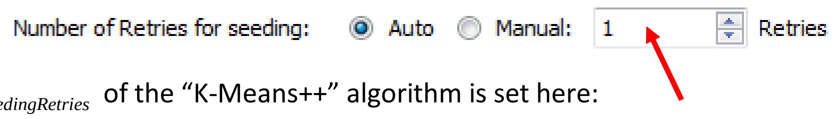

Random Seed for next run: 369998651 for previous run: 99695313 copy =>

You can "restore" good segmentations by, first, entering the corresponding "*random seed*" number here: ... and, second, clicking the Run K-Means button.


Random Seed for next run: 369998651 for previous run: 99695313 copy =>

Thus the "*random seed*" parameter allows you to quickly "switch" between different segmentations. This is a nice complement to the standard load&save functionalities of StarDust (see section 5.2.2 and 5.2.3 about the load&save segmentation analysis).

5.5.2.5. Control 5: retries for seeding


Number of Retries for seeding: [✓ Auto] Manual: 1 Retries

The parameter $n_{SeedingRetries}$ of the "K-Means++" algorithm is set here:

- In manual mode, the value of $n_{SeedingRetries}$ is exactly the one entered by the user inside StarDust.

- In "Auto" mode, the value of $n_{SeedingRetries}$ is $n_{SeedingRetries} = 2 + \ln(n_s)$ (n_s is the number of segments).

5.5.3. The Ward's algorithm

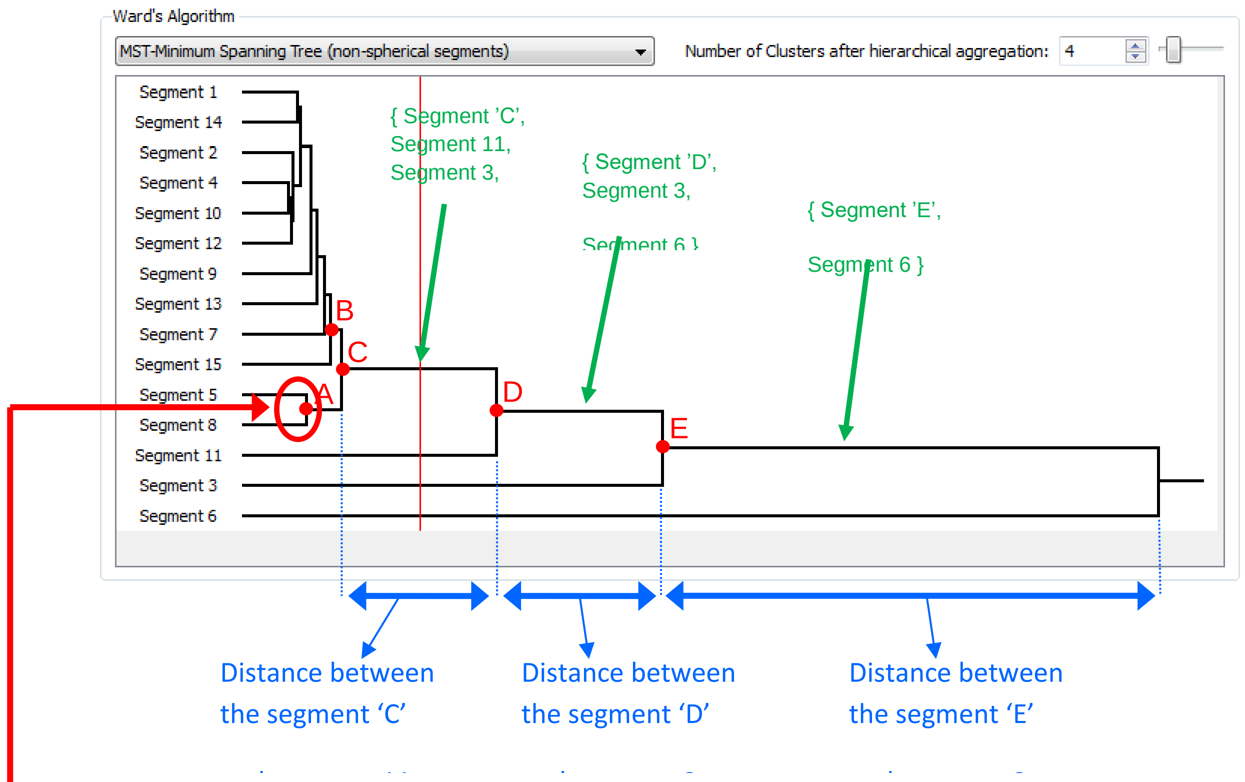
One major drawback of the KMeans algorithm is that you have to "guess yourself " how many segments your dataset contains. This is difficult. This is why we use the Ward's algorithm to find the exact number of segments. The Ward's algorithm is used after the K-means algorithm. The Ward's algorithm is an iterative algorithm. It works this way:

1. starts with the segments found by the K-Means algorithm.
2. find the two closest segments S_A and S_B amongst all the segments still available.
3. delete the segments S_A and S_B and replace them by a new segment S_{A+B} that is the sum of the segments S_A and S_B (this implies computing the position of the center of S_{A+B} segment).
4. if there are still some segments available go back to point 2.

Thus, at each iteration of the Ward's algorithm, the number of segment is decreased by one. We can now easily select how many segments we want by selecting how many iteration of the ward's algorithm we will do.

5.5.3.1. How to find the right number of segments?

To find the right number of segments inside your dataset, you have to look at the dendogram representation of the Ward's algorithm:


Ward's Algorithm

MST-Minimum Spanning Tree (non-spherical segments)

Number of Clusters after hierarchical aggregation: 4

Segment 1
Segment 14
Segment 2
Segment 4
Segment 10
Segment 12
Segment 9
Segment 13
Segment 7
Segment 15
Segment 5
Segment 8
Segment 11
Segment 3
Segment 6
A
B
C
D
E
{ Segment 'C', Segment 11, Segment 3,
{ Segment 'D', Segment 3, Segment 6 }
{ Segment 'E', Segment 6 }
Distance between the segment 'C'
Distance between the segment 'D'
Distance between the segment 'E'

This tree represents the hierarchical aggregation process. You read it from left to right.

For example, you can see here: that the segments 5 and 8 (that originates from the KMeans algorithm) are "grouped together" into a new segment named '**A**'. Each node of the tree represents an aggregation between two segments (and it represents also the resulting segment). Thus, each node represents one iteration of the Ward's algorithm.

The segments 'A' and the segments 'B' are grouped together to form the new segment 'C'.
The segments 'C' and the segments '11' are grouped together to form the new segment 'D'.
The segments 'D' and the segments '3' are grouped together to form the new segment 'E'.

The horizontal distance represents the distance represents the distance between the two segments that will be aggregated together: see the illustration in **blue**. You can see on the dendogram that the first "aggregations" are involving segments that are really close to each other. Only the last three "aggregations" are grouping together segments that are far away from each other. Thus, from the dendogram, we deduce that the optimal number of segment for this dataset is "4": see the illustration in **green**.

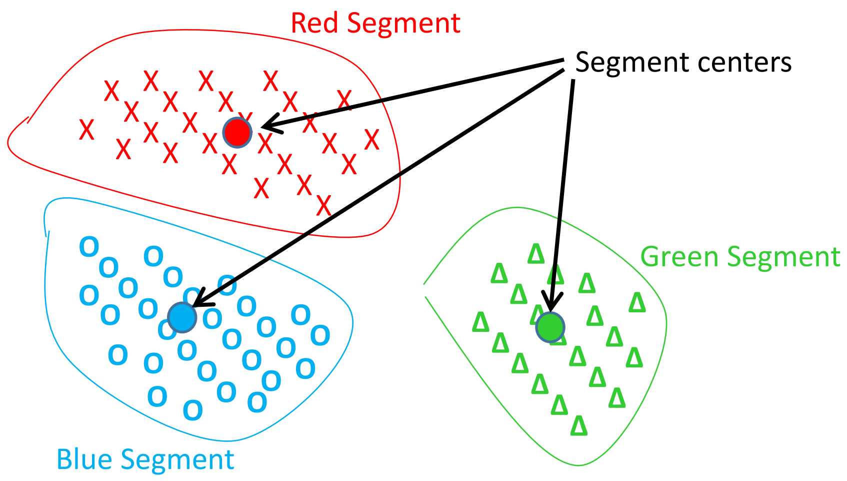
You can click anywhere on the dendogram to select the corresponding final number of segments. For example, in the illustration above, the vertical red line indicates that the user has decided to have "4" segments (because the vertical red line crosses 4 horizontal black lines).

When you change, inside the Ward's algorithm, the final number of segments, you can see, in real-time, the colour of the points changing inside the main 3D display of StarDust. Thus, you can instantaneously validate your decision. This is unique to StarDust. Other datamining softwares are forcing you to wait for possibly several minutes (sometime hours!) before being able to validate how many segments you want.

5.5.3.2 The different variations around the original Ward's algorithm

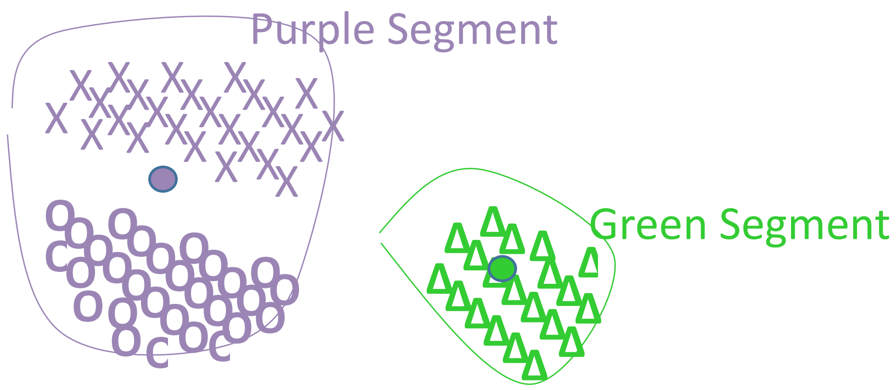
The Ward's algorithm has a strong limitation: it only works when the segments are spherical.

Let's take a first example: we start from the 3 segments (Red, Green Blue) delivered by the K-Means algorithm:



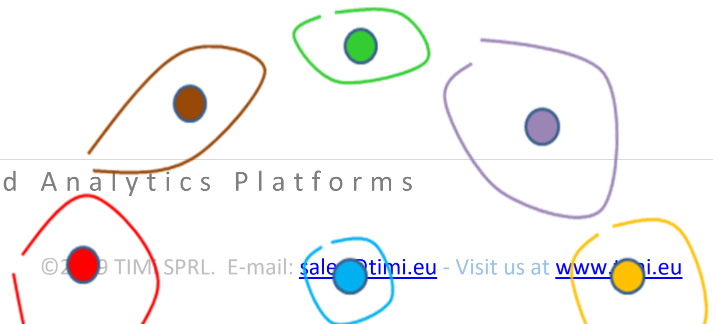
In the above graph, each cross, circle or triangle represents an individual.

Inside this first example, the centers of the red and blue segments are the closest centers, and thus one iteration of the standard ward's algorithm will "group" together the red and blue segments inside a new (purple) segment, as illustrated here:

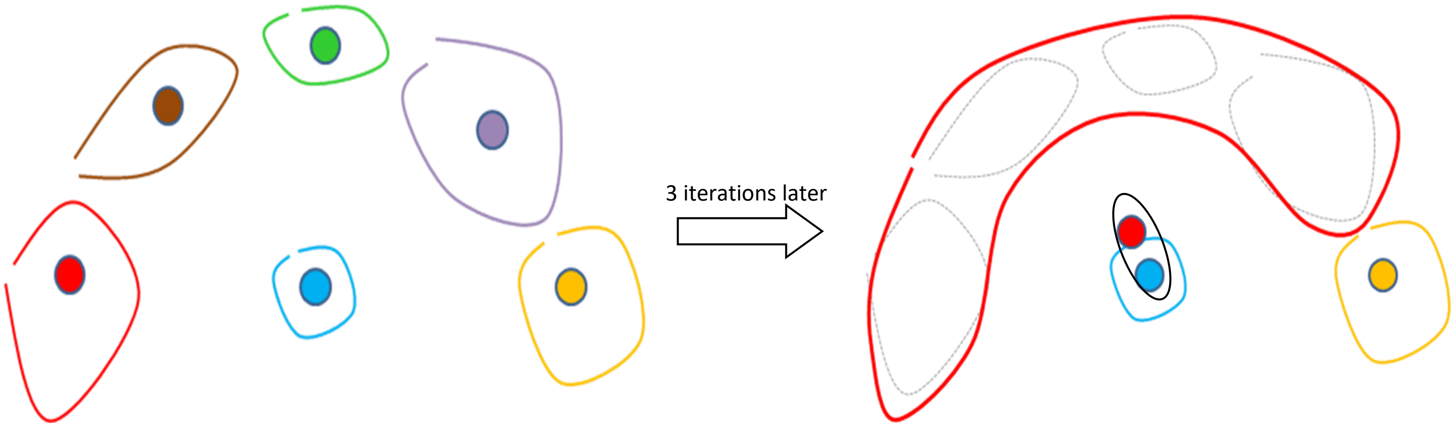


Please note that the center of the purple segment is at a completely new position that is the center of gravity of the all points inside the red and blue segments. On this first example, the Ward's algorithm is working as expected.

Let's consider a more difficult example (we did not represent the individual points anymore):



After 3 iterations of the original Ward's algorithm, we have:



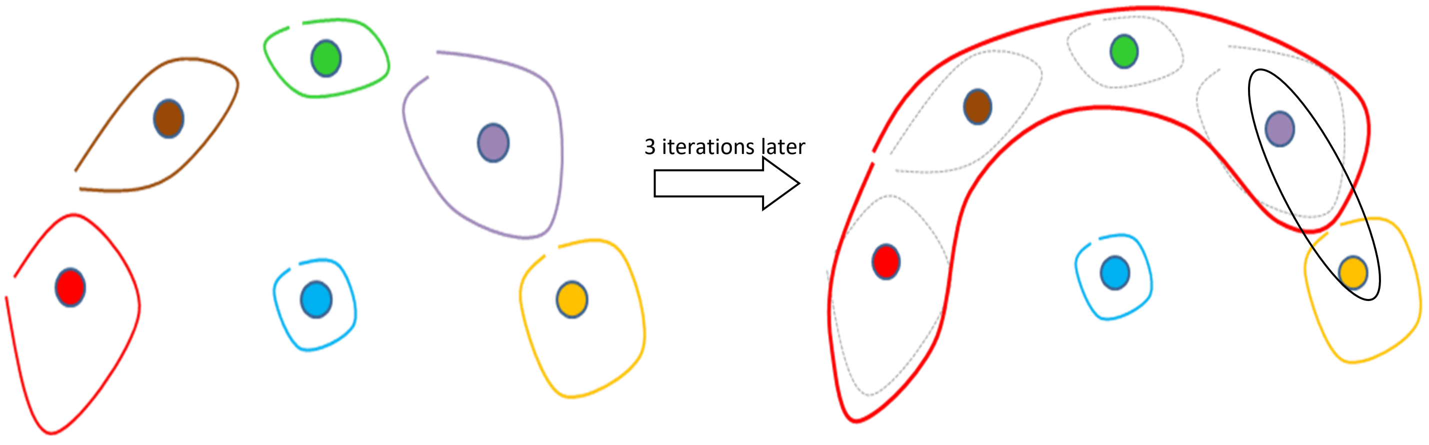
On the 4th iteration of the standard Ward's algorithm, the red segment will be erroneously aggregated with the blue segment (instead of the yellow segment) (because the center of the red segment is closer to the center of the blue segment than the center of the yellow segment). The problem comes from the fact the segments are not spherical and thus the center of the red segment falls completely "outside" the red segment.

To correct this situation, we will use another algorithm named the "**M**inimum **S**panning **T**ree" algorithm (or MST, in short). This new algorithm does not create "new centers" that could possibly be completely wrong (see the red center in the previous example): it only uses the "original" centers delivered by the K-means algorithm. It works this way:

1. Start with the segments found by the K-Means algorithm. Create one "Meta-segment" for each original segment.
2. Search for the two closest segments S_A and S_B amongst all the original segments delivered by the K-Means algorithm. Do NOT include inside the search all the pairs (S_A, S_B) where S_A and S_B belongs to the same "meta-segment".
3. Mark the segments S_A and S_B as belonging to the same "meta-segment" S_{A+B} that is the sum of the "meta-segment" including S_A and the "meta-segment" including S_B.
4. If there is only one "meta-segment" then stop, otherwise go back to point 2.

Thus, at each iteration of the MST algorithm, the number of meta-segment is decreased by one. We can now easily select how many meta-segments we want by selecting how many iteration of the MST algorithm we will do. The "meta-segments" of the MST algorithm are equivalent to the "segments" of the ward's algorithm.

The MST algorithm is working fine on the previous difficult example because at the 4th iteration of the MST algorithm the red segment will be correctly aggregated with the yellow segment.



TIMi is the only segmentation/datamining tool that offers you the MST algorithm that allows you to define "non-spherical" segments.

Both the standard Ward's algorithm and the MST algorithm are computing distances between segments. We have already talked a lot about distances between points/individuals of our dataset (see section 5.5.1) but not between segments. There are two different "*distances between segments*" available inside StarDust:

1. Generalized distance: $SegmentDist(S_A, S_B) = \frac{w_{S_A} \; w_{S_B}}{w_{S_A} + w_{S_B}} dist_{E+C}(C_A, C_B)$
2. Simple distance: $SegmentDist(S_A, S_B) = dist_{E+C}(C_A, C_B)$

... where S_A is the segment ***A***.

w_{S_A} is the sum of all the row-weight of the individuals belonging to segment ***A***.

C_A is the center of the segment ***A***.

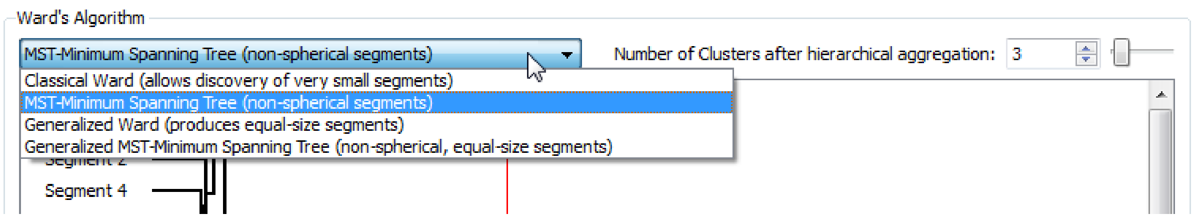
$dist_{E+C}(A,B)$ is the distance between the points ***A*** and ***B*** as defined in section 5.5.1

The Generalized distance has the tendancy to create equal-size segments.
The Simple distance creates segments of any size.

To summarize, we have:

	Simple distance (allow discovery of small segments)	Generalized Distance (produces equal-size segments)
standard Ward's algorithm (spherical segment)	Classical Ward (option 1)	Generalized Ward (option 3)
Minimum spanning tree (MST) algorithm (non-spherical segments)	MST- Minimum spanning tree (option 2)	Generalized MST- Minimum spanning tree (option 4)

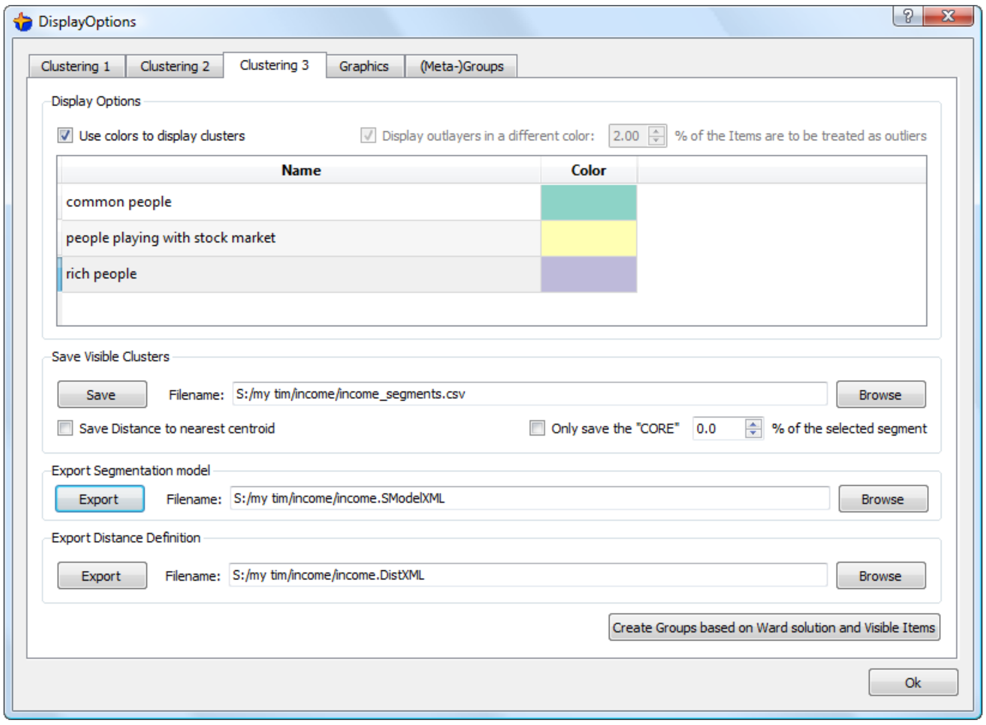
These 4 different options are available here:


Ward's Algorithm
MST-Minimum Spanning Tree (non-spherical segments)
Classical Ward (allows discovery of very small segments)
MST-Minimum Spanning Tree (non-spherical segments)
Generalized Ward (produces equal-size segments)
Generalized MST-Minimum Spanning Tree (non-spherical, equal-size segments)
Number of Clusters after hierarchical aggregation: 3
Segment 2
Segment 4

5.5.4. Setup the display of the segments

Inside Stardust, there are two different ways of computing the colours of all the points inside the 3D display: see section 5.2.14 about this subject. To summarize, these two different ways are:

1. The colours of the points are setup through the advanced 3D display "Group" manager. You can access this manager by selecting the "(Meta-)Groups" tab inside the "parameter window": see section 5.3.1.2.
2. The colour of the points represents the different segments inside the population. You can configure the colour of each segment by selecting the "Segmentation 3" tab inside the "parameter window":



[✓] Use colors to display clusters
[✓] Display outlayers in a different color: 2.00 % of the Items are to be treated as outliers

Name	Color
common people	🟩
people playing with stock market	🟨
rich people	🟪

Save Visible Clusters
Filename: S:/my tim/income/income_segments.csv
[ ] Save Distance to nearest centroid
[ ] Only save the "CORE" 0.0 % of the selected segment

Export Segmentation model
Filename: S:/my tim/income/income.SModelXML

Export Distance Definition
Filename: S:/my tim/income/income.DistXML

The ☐ Use colors to display clusters checkbox is completely equivalent to the button (see section 5.2.14 about the button).

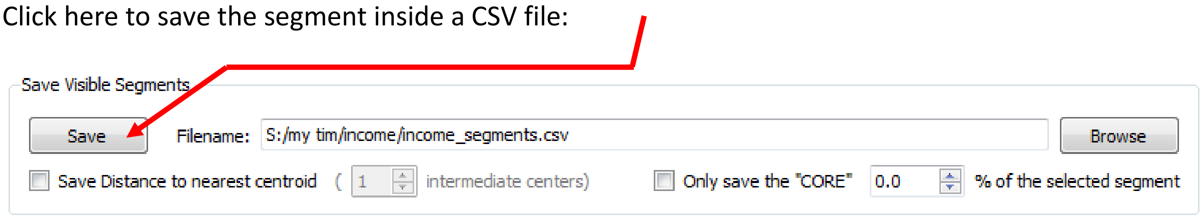
Inside the table on the "Segment 3" tab, you can:

- Click on the "Color" column to change the colour of each segment.

- Change the label of each segment.
  This label is used to describe the segments inside:
    - The "segmentation model" (see section 5.5.6.)
    - The CSV file described inside the next section (5.5.5)

5.5.5. Saving the segments in a CSV file.

Click here to save the segment inside a CSV file:



Save Visible Segments

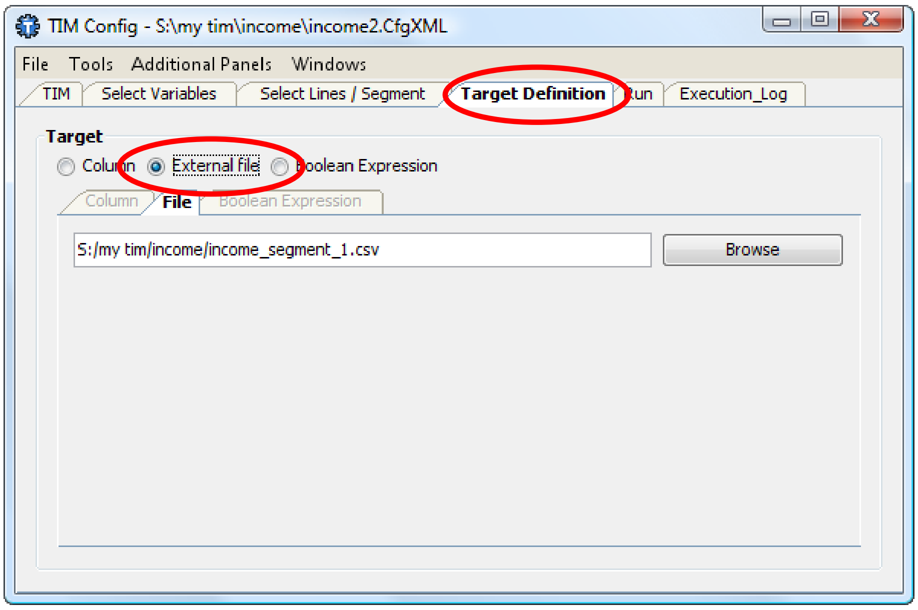
Filename: S:/my tim/income/income_segments.csv

☐ Save Distance to nearest centroid (1 intermediate centers) ☐ Only save the "CORE" 0.0 % of the selected segment

5.5.5.1. The "Save Core" option is enabled

When the "save core" option is enabled, StarDust will only save ONE segment into the CSV file. This segment is the line selected in the table (above) on the same "Segmentation 3" tab. No particular algorithm is used to compute the distance to the "center(s)" of the segment.

The generated CSV file can directly be used to define a "Target" for predictive analysis with TIMi. For example:



TIM Config - S:\my tim\income\income2.CfgXML

Target: Column [✓ External file] Boolean Expression

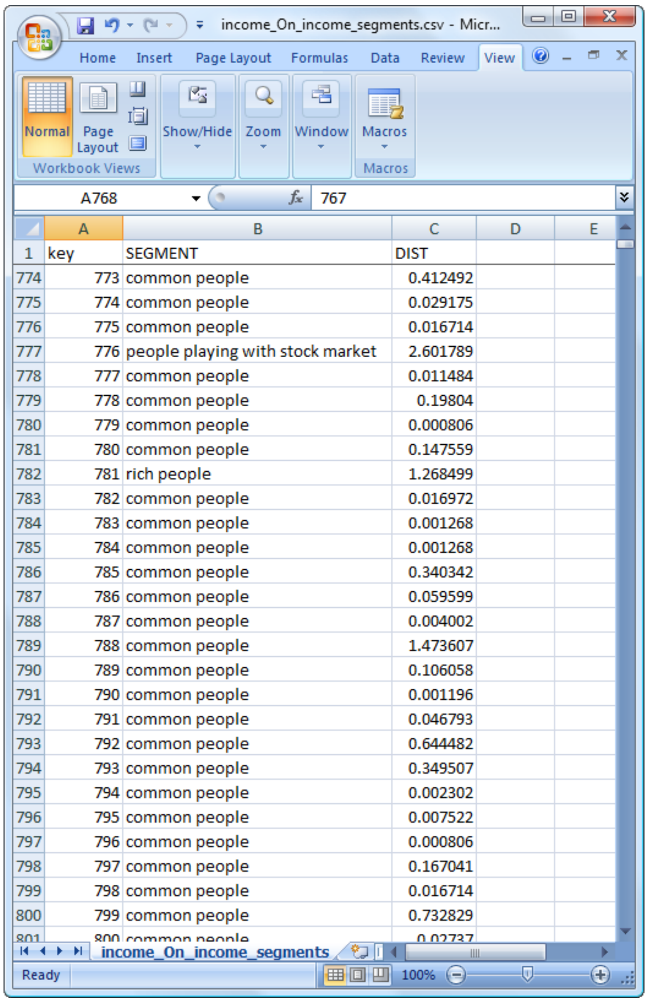
File: S:/my tim/income/income_segment_1.csv

Using the "save core" option of StarDust, you can use the full "descriptive power" of TIMi to describe in a perfect way each segments. To describe your segments from a business-point-of-view, the best solution (but also the most time-consuming) is to use the full <u>predictive</u> power of TIMi. Indeed, inside the final analyst report of TIMi, we will only see the exact small number of "de-correlated", "perpendicular" variables required to describe the current segment and no additional un-needed variable. In opposition, with StarDust, you don't really know if two variables are non-correlated and should thus both be included into the report describing a segment (you can have a vague idea of the correlation of two variables: see section 2.2.5 and 5.3.2.4 but it's not precise enough).

5.5.5.2. *"Save the Distance" to the nearest centroid.*

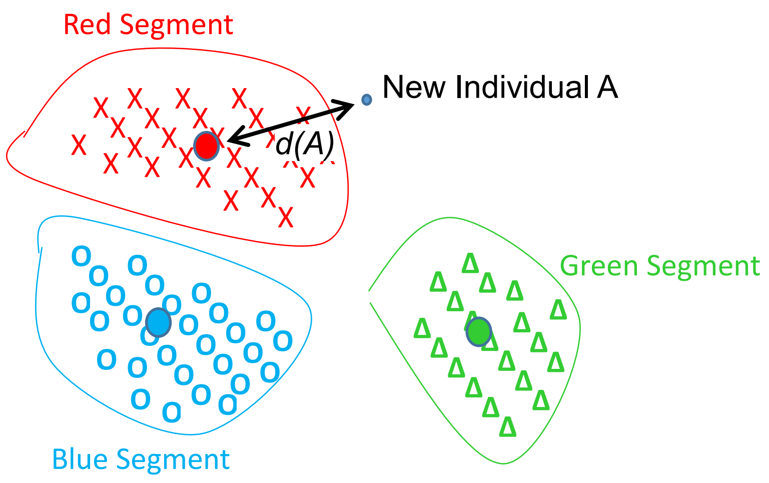
This option is only available when the "Save Core" option is disabled.

This will generate a prediction file that contains:



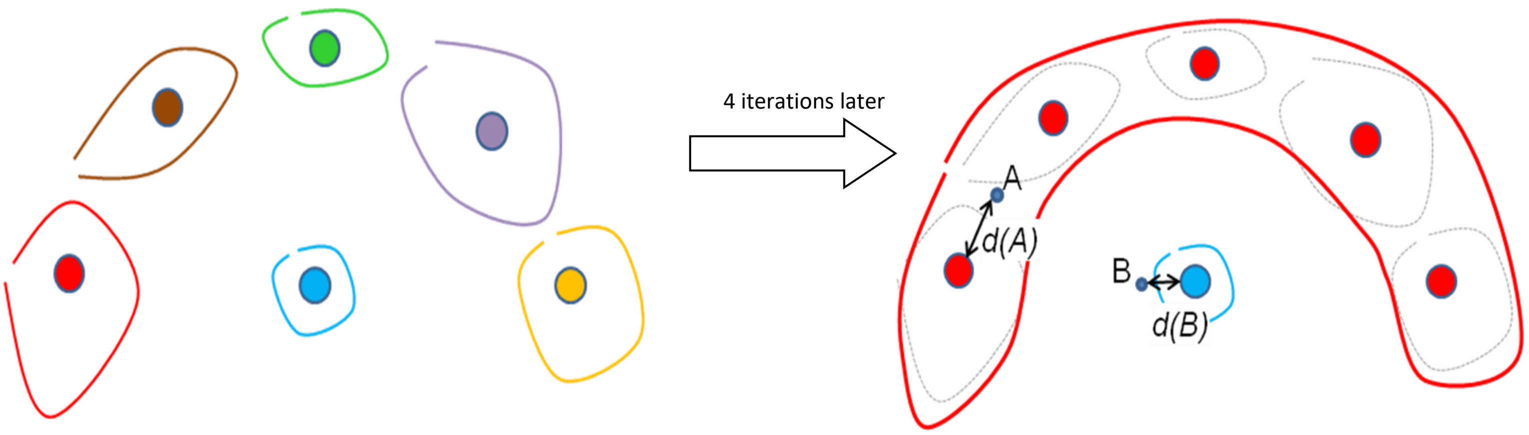
	A	B	C	D	E
1	key	SEGMENT	DIST		
774	773	common people	0.412492		
775	774	common people	0.029175		
776	775	common people	0.016714		
777	776	people playing with stock market	2.601789		
778	777	common people	0.011484		
779	778	common people	0.19804		
780	779	common people	0.000806		
781	780	common people	0.147559		
782	781	rich people	1.268499		
783	782	common people	0.016972		
784	783	common people	0.001268		
785	784	common people	0.001268		
786	785	common people	0.340342		
787	786	common people	0.059599		
788	787	common people	0.004002		
789	788	common people	1.473607		
790	789	common people	0.106058		
791	790	common people	0.001196		
792	791	common people	0.046793		
793	792	common people	0.644482		
794	793	common people	0.349507		
795	794	common people	0.002302		
796	795	common people	0.007522		
797	796	common people	0.000806		
798	797	common people	0.167041		
799	798	common people	0.016714		
800	799	common people	0.732829		
801	800	common people	0.02737		

Let's now illustrate how the distance to the nearest segment is computed. We will first give an easy example: Let's assume that we have three segments Red, Green and Blue. A new individual 'A' arrives:



Everything is fine: The individual 'A' belongs to the Red Segment because it's the closest one. The distance from the individual 'A' to the Red Segment is noted '*d(A)*'.

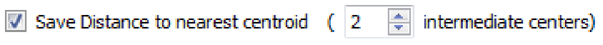
If the segments are non-spherical, we were forced to use the "**M**inimum **S**panning **T**ree" (MST) algorithm. When we are using the MST algorithm, the distance to the nearest segment is computed in a special way. Let's go back to the second example of section 5.5.3.2:

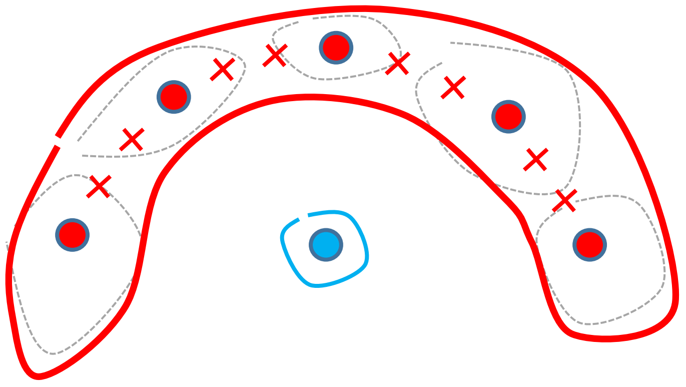


The main question is: "*Which point (A or B) has the smallest distance to its own segment?*" (The point 'A' belongs to the Red Segment and the point 'B' belongs to the Blue Segment). This question is interesting when you want to select the "real core" of each segment.

The distance from the individual 'A' to the Red Segment is noted '*d(A)*'.
The distance from the individual 'B' to the Blue Segment is noted '*d(B)*'.

Based on the distances '*d(A)*' and '*d(B)*', we can say that 'B' is the answer to the above question (because '*d(B)<d(A)*'). But graphically, we see the opposite! The point 'A' is the closest to its own (red) segment: it's in the middle of it! ... and B is really far apart from the blue segment! Obviously, there is something wrong here.

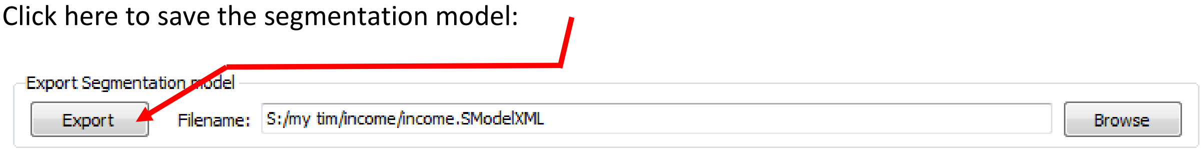
To correct this (bad) situation, you can ask StarDust to add some "intermediate centers". For example, when you enter:  ☑ Save Distance to nearest centroid (2 intermediate centers) ... you will have 2 additional "intermediate centers" (illustrated below with small crosses) that will be used to compute the distance to the nearest segment:



Using these “intermediate centers”, we now have ‘*d(A)<d(B)*’ which is correct: The issue is solved!

5.5.6. Exporting and using segmentation models.

Click here to save the segmentation model:



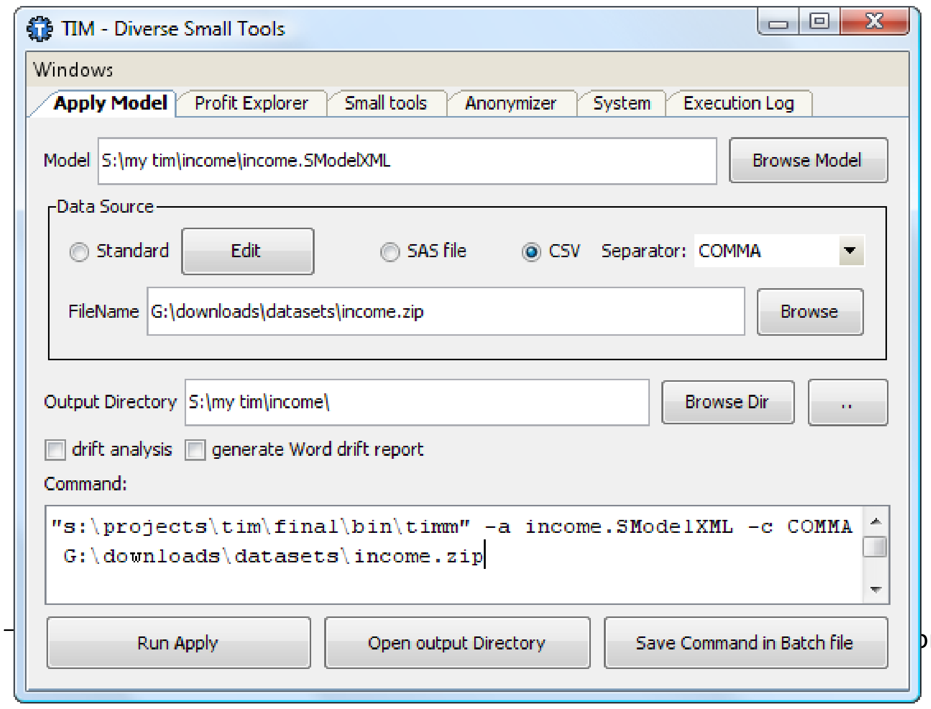
Export Segmentation model

Filename: S:/my tim/income/income.SModelXML

TIMi offers a tight integration between segmentation analysis and predictive analysis. The segmentation models created with StarDust can easily be used in conjunction with the predictive analytic engine of TIMi.

You can use segmentation models to:

1. Automatically predict at which segment a customers/points belongs to: Open the “diverse small tools”, select the “Apply model” tab: see illustration below:



TIM - Diverse Small Tools

Model: S:\my tim\income\income.SModelXML

Data Source: [✓ CSV] Separator: COMMA

FileName: G:\downloads\datasets\income.zip

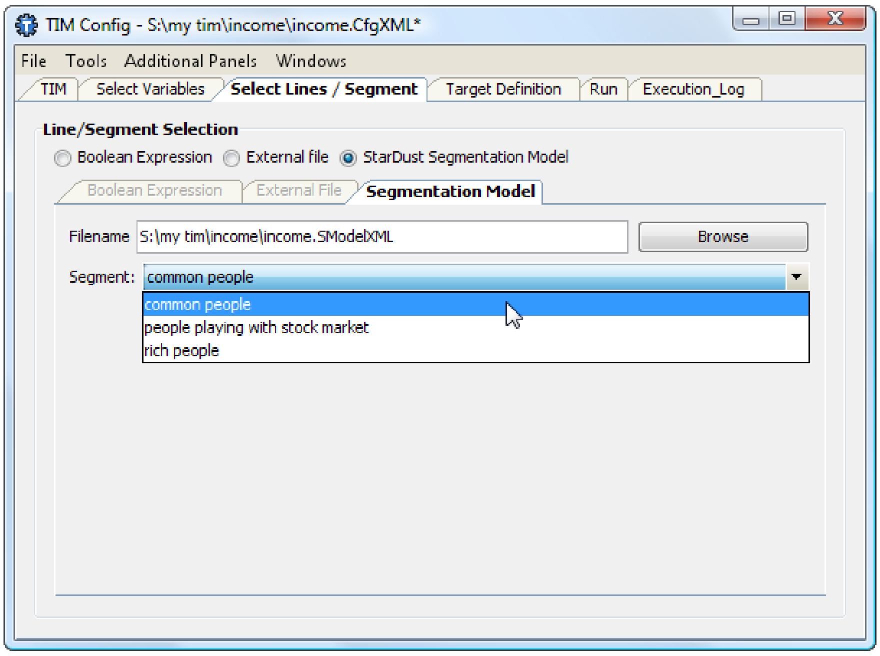
Output Directory: S:\my tim\income\

Command:

```
"s:\projects\tim\final\bin\timm" -a income.SModelXML -c COMMA
 G:\downloads\datasets\income.zip
```

on 5.5.5.2 (with

2. Create a predictive model on a subset of the rows of a dataset. This subset is defined by the selected segment:


TIM Config - S:\my tim\income\income.CfgXML*

File Tools Additional Panels Windows

TIM | Select Variables | **Select Lines / Segment** | Target Definition | Run | Execution_Log

Line/Segment Selection

Boolean Expression | External file | [✓ StarDust Segmentation Model]

Boolean Expression | External File | **Segmentation Model**

Filename: S:\my tim\income\income.SModelXML [Browse]

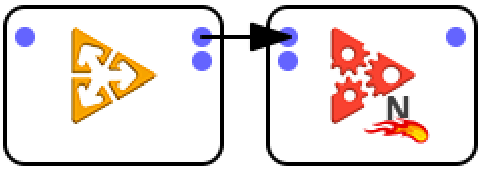
Segment: common people

- common people
- people playing with stock market
- rich people

In this way, you can easily create one predictive model per segment.

3. When you have several predictive models (one per segment), it can be difficult to compute the prediction for a specific individual because it is becoming a complex procedure: To do a prediction for one individual, you have to :
   a. Find the segment that the individual belongs to.
   b. Select the predictive model associated with the segment and apply it.

   The whole procedure can be completely automated using these two boxes inside Anatella:



5.5.7. Explaining the segments from a business-point-of-view

You can explain easily the segments using the intuitive charts explained in sections 5.2.10, 5.2.11 and 5.2.12. All these charts can directly be added to the final report in 1 mouse-click. Most of these charts can directly be exported to Excel in one mouse click.

A good "trick" to setup rapidly all the charts is to convert the segments into groups, as explained in section 5.3.1.5.

The charts of section 5.2.10, 5.2.11 and 5.2.12 allow you to see some of the best explanatory variables for each segment. You can use these charts to select the variables to "present" to the business-users to "explain" in the simplest way your segmentation. These charts are good for a preliminary analysis of your segments but they have some limitations:

1. You don't know how many variables to "present" to the business-users to "completely describe" a specific segment.
2. When you are "describing" a specific segment, you should avoid to present inside your report two correlated variables because they are describing the same ""concept" related to your

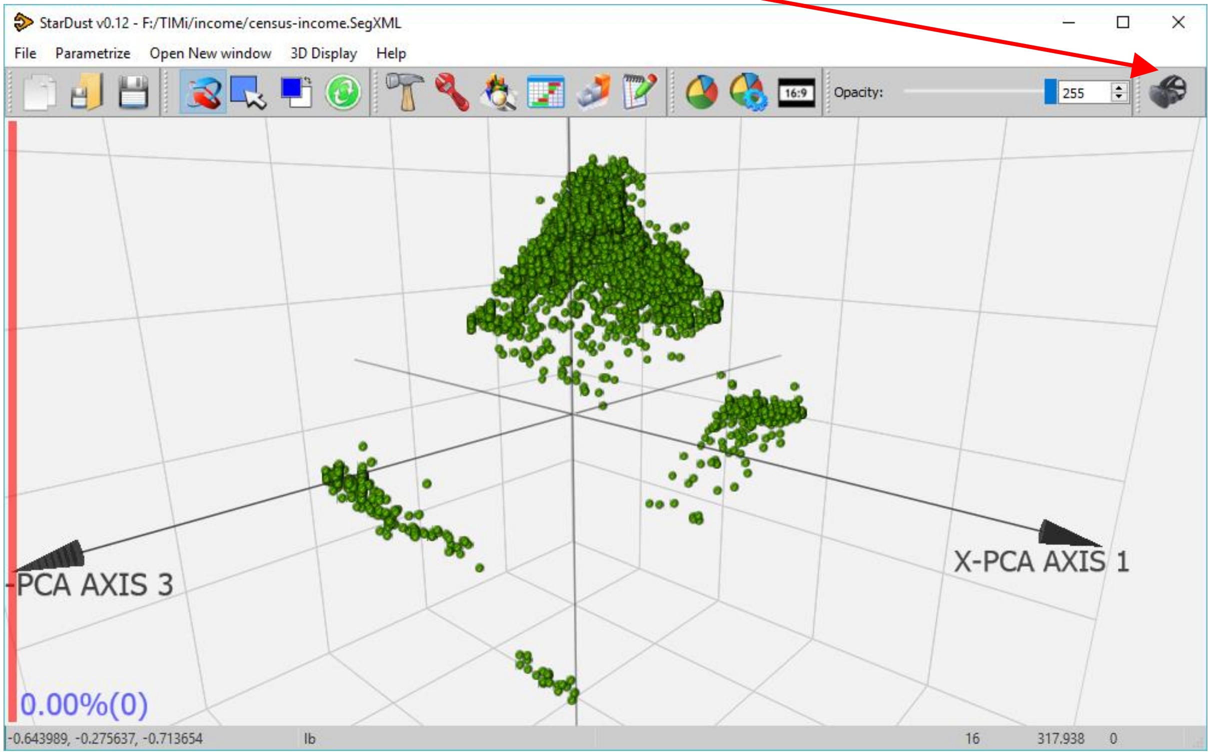
segment. Correlated variables will both have approximately the same (***high***) KL value and, thus, you will be "tempted" to include both variables inside you report. Thus, if you are using only the charts from sections 5.2.10, 5.2.11 and 5.2.12, you will have to do arbitrary and subjective choices about the variables to "present" to correctly "describe" a specific segment. There exists an objective and automated way of selecting the "right" set of variables to "describe" a specific segment. This "optimal" set of variables contains the smallest amount possible of different variables that characterizes a specific segment, the "best as possible": see section 5.5.5.1 to obtain this "optimal" subset of variables.

You can "note" all your findings and "insights about your customers" (to be sure to forget nothing) inside the small HTML notepad included with StarDust (see section 5.2.12).

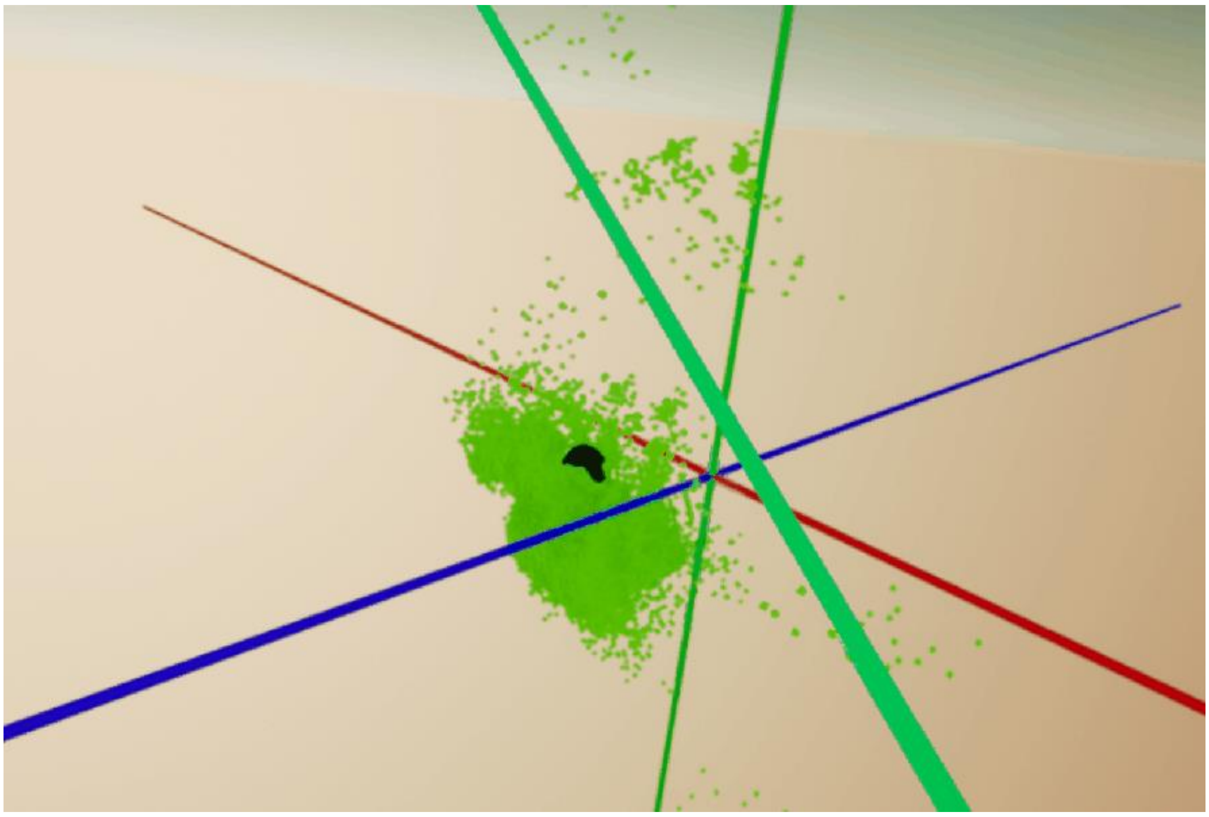
You can switch rapidly between different segmentations using the technique described in section 5.5.2.4 (Control 4: Random seed).

6. Virtual Reality (VR) display

Click here to start the Stardust VR module:



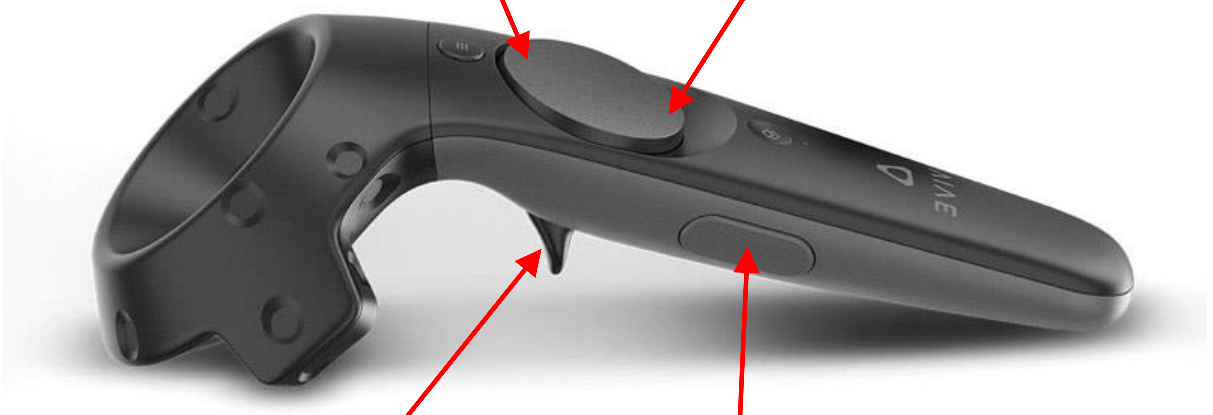
When you enter VR mode, you'll see inside your VR helmet the same 3D display as inside the Stardust window. This typically looks like:



The controls to move inside the VR space are:

Touch the top of the “Thumb” button to move forward in the viewing direction.

Touch the bottom of the “Thumb” button to move backward alongside the viewing direction.



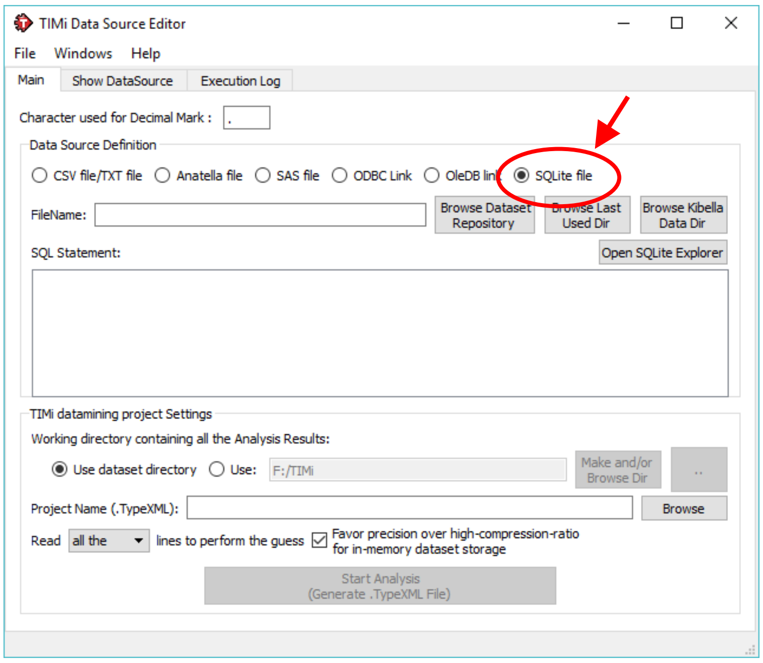
Shoot with the “Trigger” button on a dot in the 3D world to view all the features of this “Dot”. (additional details below)

Press the “Grip” buttons on one controller to rotate the world in the same way as rotating a bike handle bar.

Press the “Grip” buttons on two controllers simultaneously to rotate and re-scale (i.e. zoom in/out) the world.

To be able to “shoot” on a dot and view all the features of the “dot”, you must have the following:

The data source must be a SQLite file.



Character used for Decimal Mark : .

Data Source Definition

CSV file/TXT file · Anatella file · SAS file · ODBC Link · OleDB link · [✓ SQLite file]

FileName:

SQL Statement:

TIMi datamining project Settings

Working directory containing all the Analysis Results:

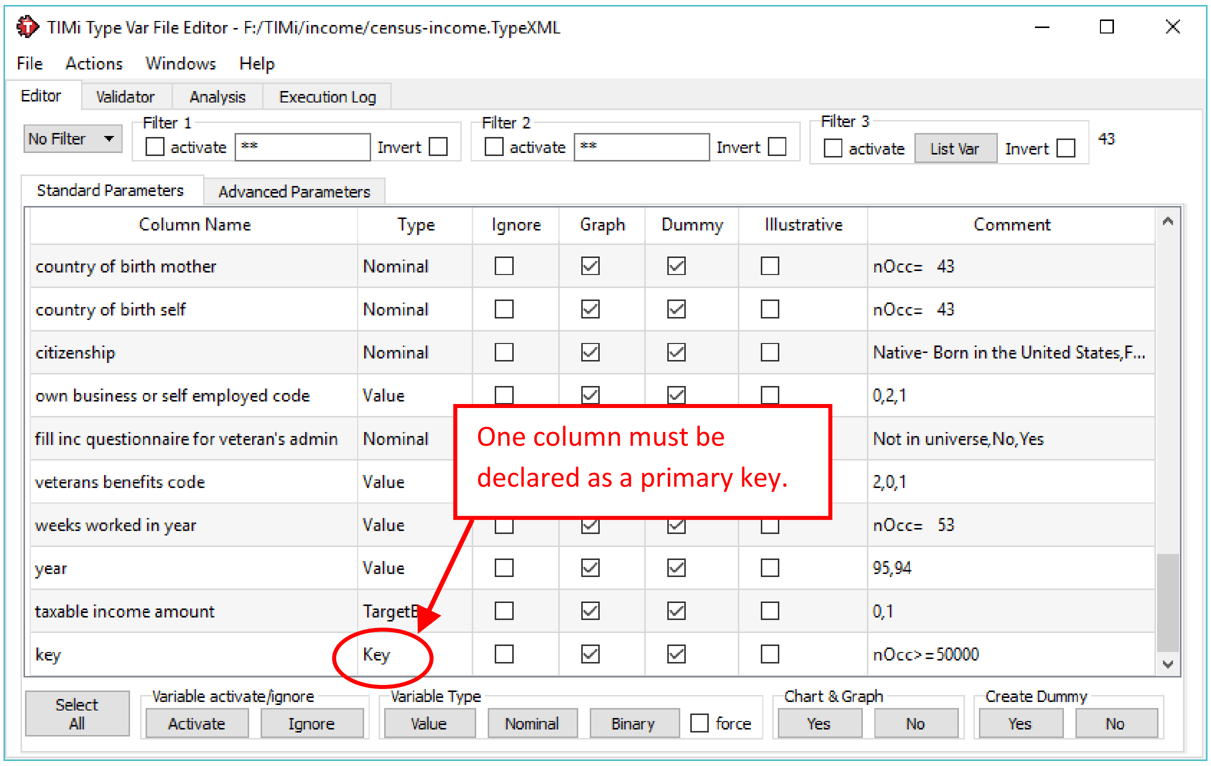
[✓ Use dataset directory] · Use: F:/TIMi

Project Name (.TypeXML):

Read all the lines to perform the guess

[✓] Favor precision over high-compression-ratio for in-memory dataset storage

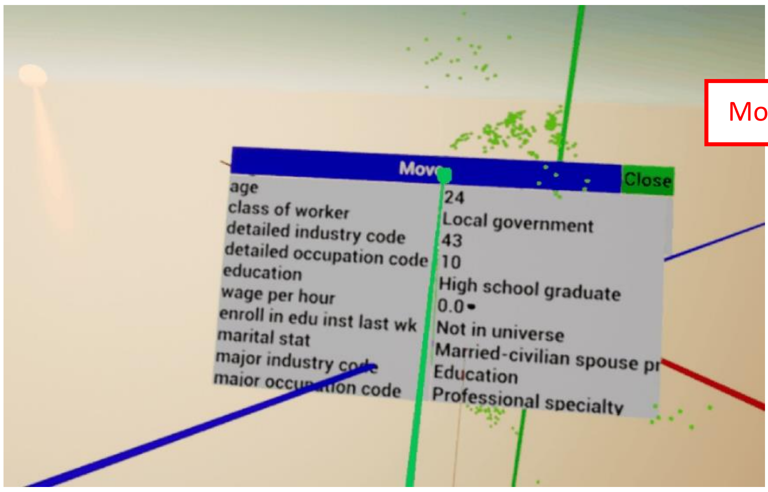
- You must have defined a Primary key when configuring the dataset:


TIMi Type Var File Editor - F:/TIMi/income/census-income.TypeXML
File Actions Windows Help
Editor Validator Analysis Execution Log
No Filter
Filter 1 activate ** Invert
Filter 2 activate ** Invert
Filter 3 activate List Var Invert
43
Standard Parameters Advanced Parameters
Select All
Variable activate/ignore Activate Ignore
Variable Type Value Nominal Binary force
Chart & Graph Yes No
Create Dummy Yes No

Column Name	Type	Ignore	Graph	Dummy	Illustrative	Comment
country of birth mother	Nominal	☐	✓	✓	☐	nOcc= 43
country of birth self	Nominal	☐	✓	✓	☐	nOcc= 43
citizenship	Nominal	☐	✓	✓	☐	Native- Born in the United States,F...
own business or self employed code	Value	☐	✓	✓	☐	0,2,1
fill inc questionnaire for veteran's admin	Nominal					Not in universe,No,Yes
veterans benefits code	Value					2,0,1
weeks worked in year	Value	☐	✓	✓	☐	nOcc= 53
year	Value	☐	✓	✓	☐	95,94
taxable income amount	TargetB	☐	✓	✓	☐	0,1
key	Key	☐	✓	✓	☐	nOcc>=50000

One column must be declared as a primary key.

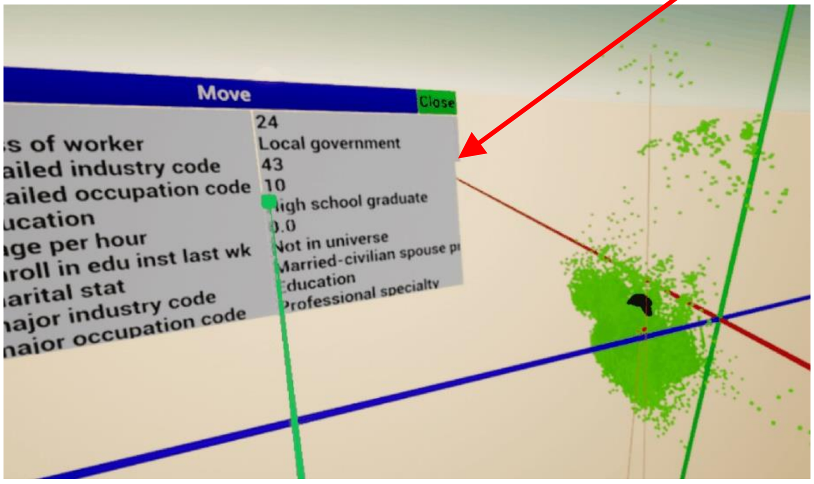
When you "shoot" on a "dot", you'll see a panel that displays all the features of the dot. Click the blue "Title Bar" to move the panel around:



age	24
class of worker	Local government
detailed industry code	43
detailed occupation code	10
education	High school graduate
wage per hour	0.0
enroll in edu inst last wk	Not in universe
marital stat	Married-civilian spouse pr
major industry code	Education
major occupation code	Professional specialty

Moving the panel around.

You can open several panels simultaneously and click on the scroll-bar here: to view all the features of the "dot":



The VR display is very handy to discover outliers and directly see the features and characteristics of all these outliers.

Press the [Escape] button on your keyboard to exit the VR display.

7. Conclusion

This concludes our (very) brief tour of the possibilities of StarDust for segmentation analysis.

I hope you enjoyed the ride! ;-)

See you!

Frank