Orange

February 2024

Tutorial

KHIOPS 10.2

KHIOPS & KHIOPS VISUALIZATION

KHIOPS COCLUSTERING & KHIOPS COVISUALIZATION

MULTI-TABLE FUNCTIONALITIES

Khiops

3 

- **Khiops**
  - Optimal data preparation based on discretization and value grouping
  - Scoring models for classification and regression
  - Correlation analysis between pairs of variables

- **Khiops Visualization**
  - Analysis of Khiops results using an interactive visualization tool

36 

- **Khiops Coclustering**
  - Correlation analysis of two or more variables using a hierarchical coclustering model

- **Khiops Covisualization**
  - Exploratory analysis of Khiops Coclustering results using an interactive visualization tool

- **Multi-table functionalities**
  - Multi-table database
  - Automatic feature construction
  - Multi-table functionalities in Khiops and Khiops Coclustering

60 

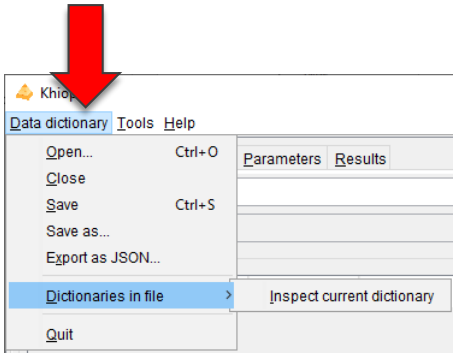
Khiops & Khiops Visualization

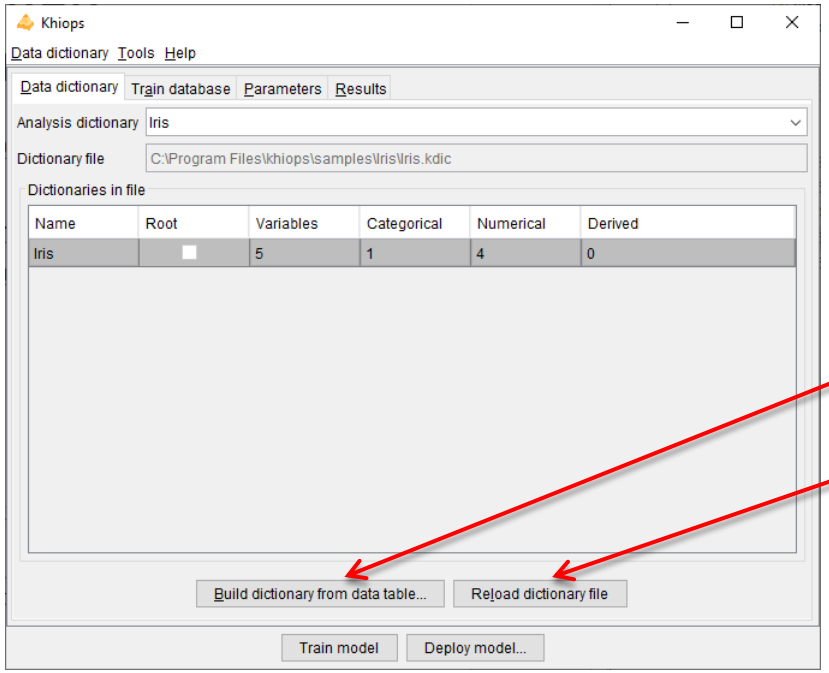
- Khiops
  - Optimal data preparation based on discretization and value grouping
  - Scoring models for classification and regression
  - Correlation analysis between pairs of variables

- Khiops Visualization
  - Analysis of Khiops results using an interactive visualization tool

Supervised classification

- **Step 1 :** Open an existing dictionary file
  (ex: sample Iris.kdic)
  - Dictionary file: contains one or more dictionaries
  - Dictionary: description of variables of a database to use during analysis





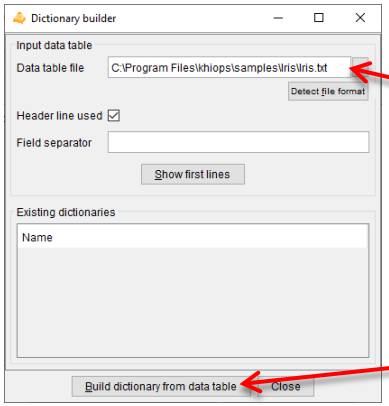
Available actions :

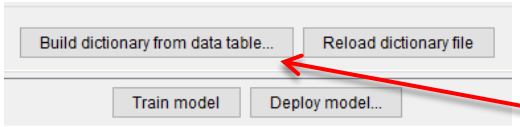
- Open, Save, Save as, Close
- Edition (menu « *Dictionary file/Inspect current dictionary* », or NotePad)
- Build dictionary from data table
- Reload dictionary file
  - useful if it has been modified from an external editor

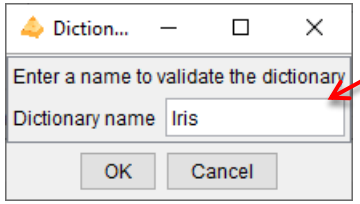
Supervised classification

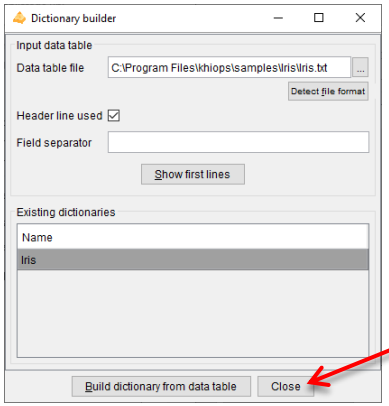
- **Step 1, bis :** Build a new dictionary from a data table

(If no available dictionary)





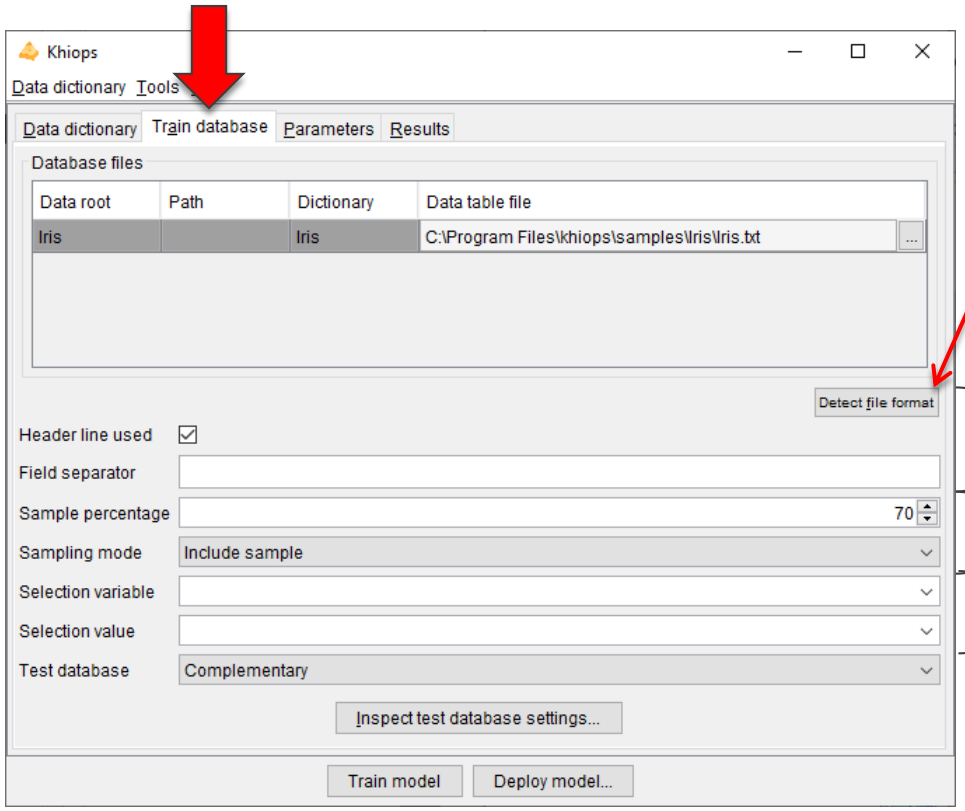




- 1. In pane Data dictionary
  - Click on button Build dictionary from data table...
- 2. In the dialog box
  - Specify the data table file
  - Build the dictionary
- 3. Specify the dictionary name
  - Repeat step 2 to build other dictionaries
- 4. Close to save dictionary file
- 5. In pane Data dictionary
  - Inspect the built dictionary
    - Check variables types
    - Select used variables

Supervised classification

- **Step 2 :** Specify train database



Detect file format : heuristic help that scans the first few lines to guess the file format. The header line and field separator are updated on success, with a warning or an error in the log window only if necessary.

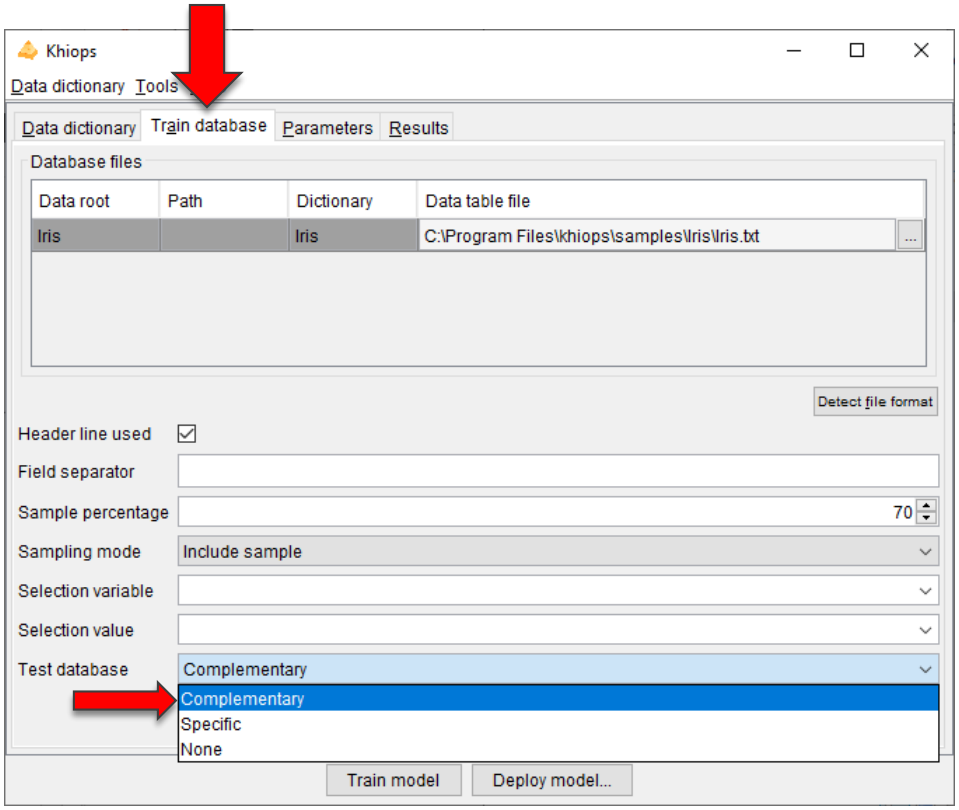
File Format

Sample percentage : default 70%

Controlled way of selecting the instances by the means of a selection variable and selection value

Supervised classification

- **Step 2, bis :** Specify test database



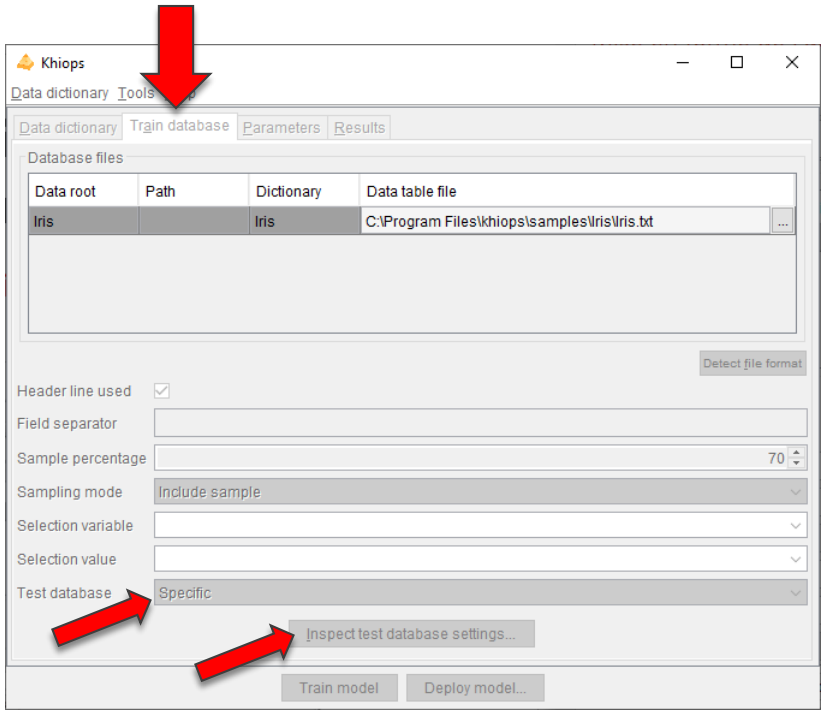
Three possibilities :

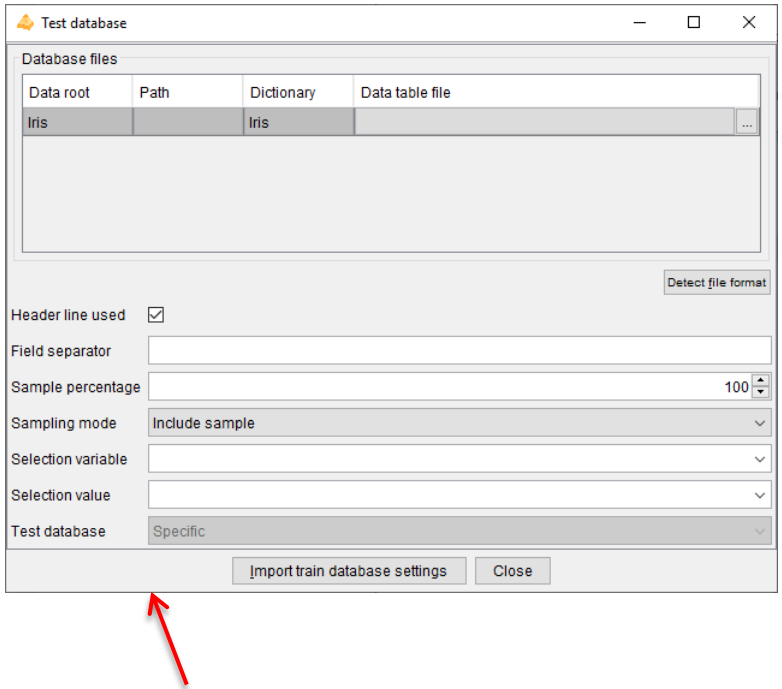
Complementary
Specific
None

Complementary (default)
The test database is the complementary of the train database according to the chosen sample percentage

Supervised classification

- **Step 2, ter :** Specify test database



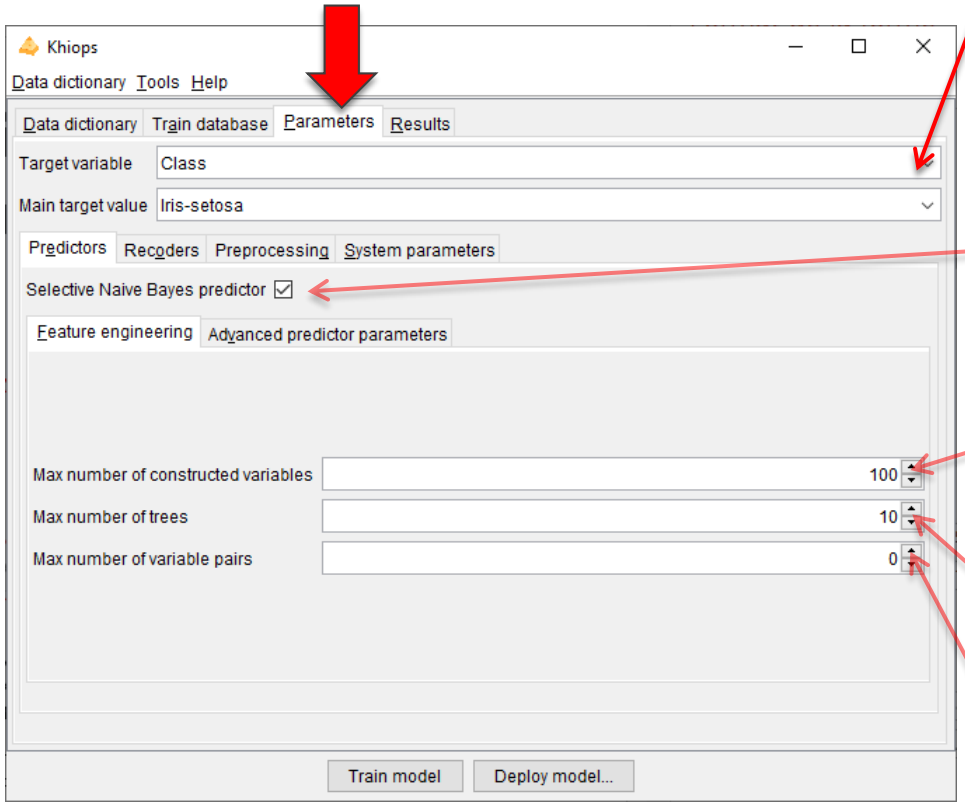


Specific

The test database has it own independent specification : specific file, sampling, selection

Supervised classification

- **Step 3 : Parameters**



Type of selected target variable implies type of analysis

Categorical -> supervised classification
Numerical -> regression
Empty -> unsupervised analysis

Selective Naïve Bayes predictor
default true, to be set to false if only data preparation is wanted (without modeling)

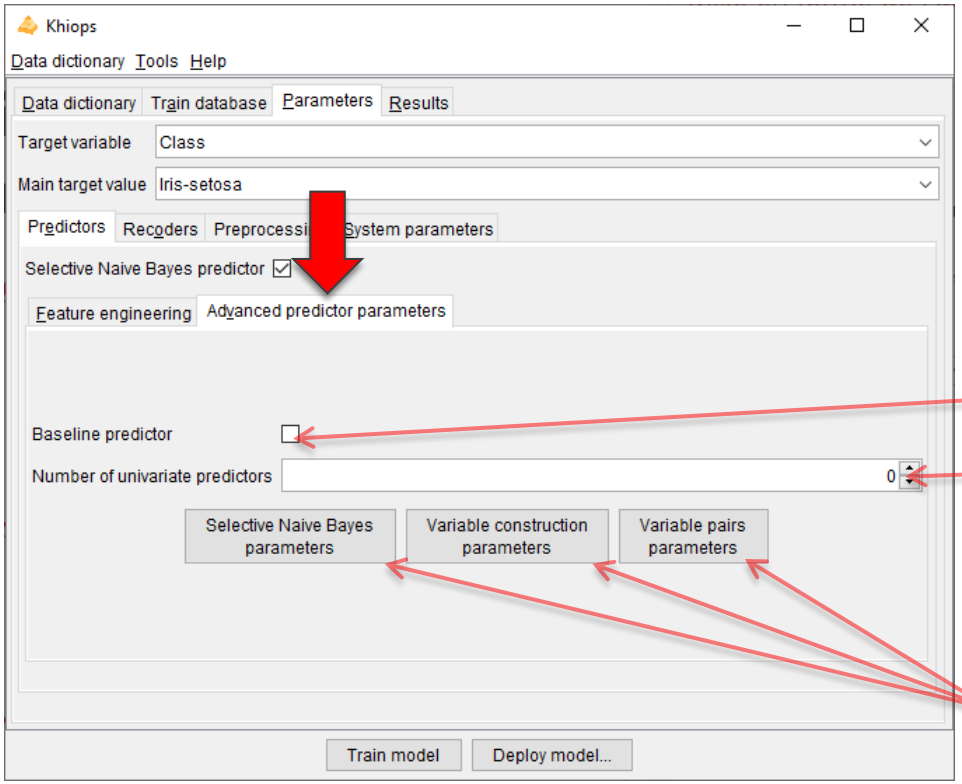
Constructed variables are computed in multi-table schema and allow to extract numerical or categorical values resulting from computing formula applied to existing variable (default 100)

The constructed trees allow to combine variables, either native or constructed (default 10)

The pairs of variables are analyzed during data preparation using a bivariate discretization method (default 0)

Supervised classification

- **Step 3 bis :** Advanced predictor parameters (optional)



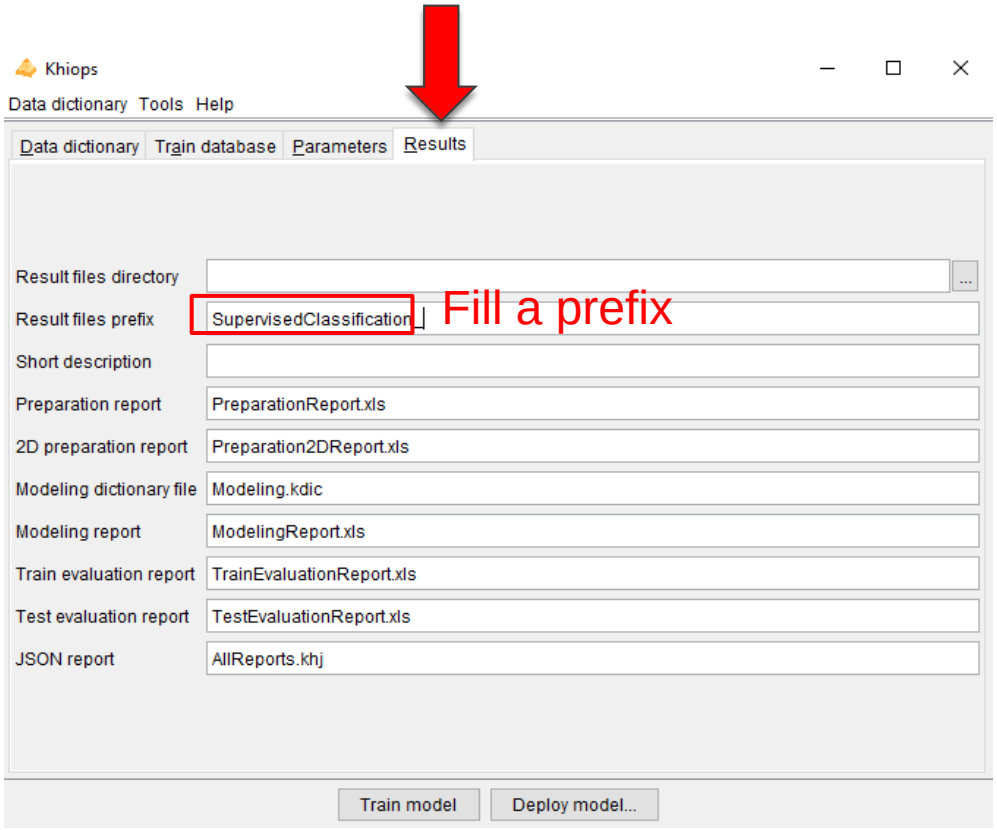
Two other optional predictors
(only in the supervised case)

- **Baseline :** prediction of the majority class (default false)
- **Univariate:** predictors exploiting one single variable (default none)

Advanced parameters to inspect

Supervised classification

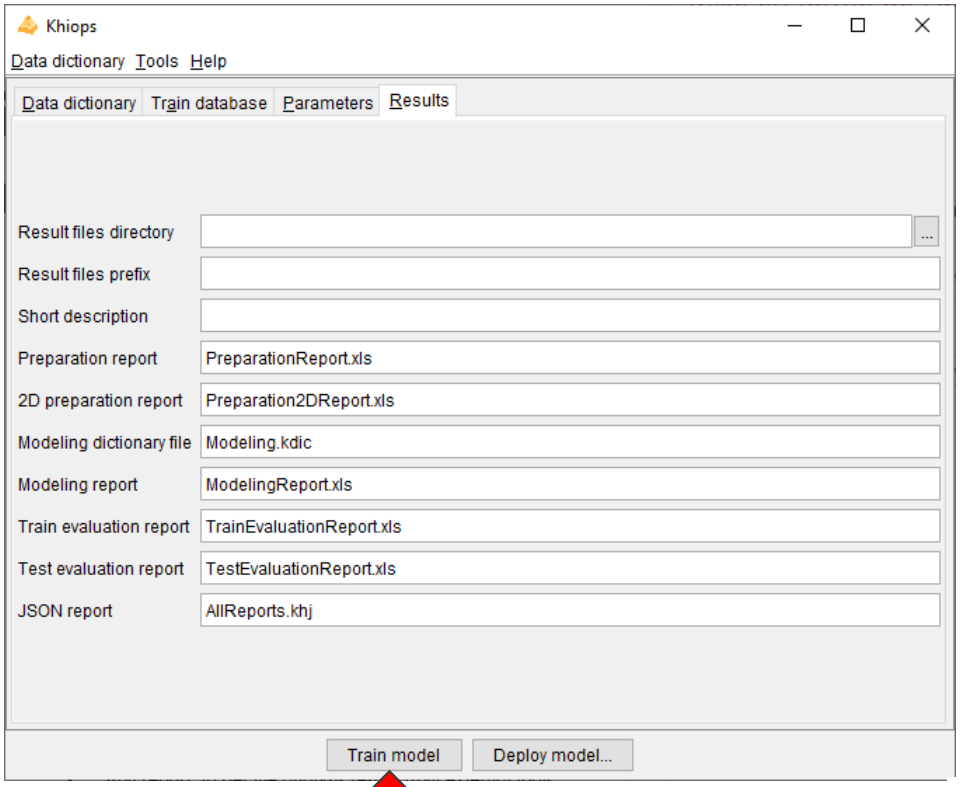
- **Step 4 : Results**



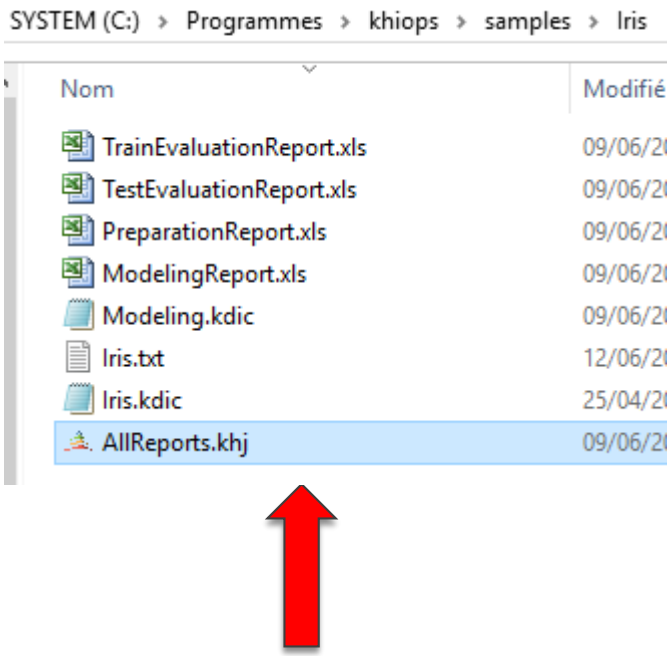
- Directory where all results files are written
- Prefix (ex: in case of several experiments)
- Brief description to summarize the current analysis
- Description of trained univariate preparation models
- Description of trained bivariate preparation models
- Technical description for deployment purposes
- Description of trained models with selected variables
- Evaluation on train database
- Evaluation on test database
- Json report, to get the analysis results from external tools

Supervised classification

- **Step 5 :** Start the analysis



1 – Train model



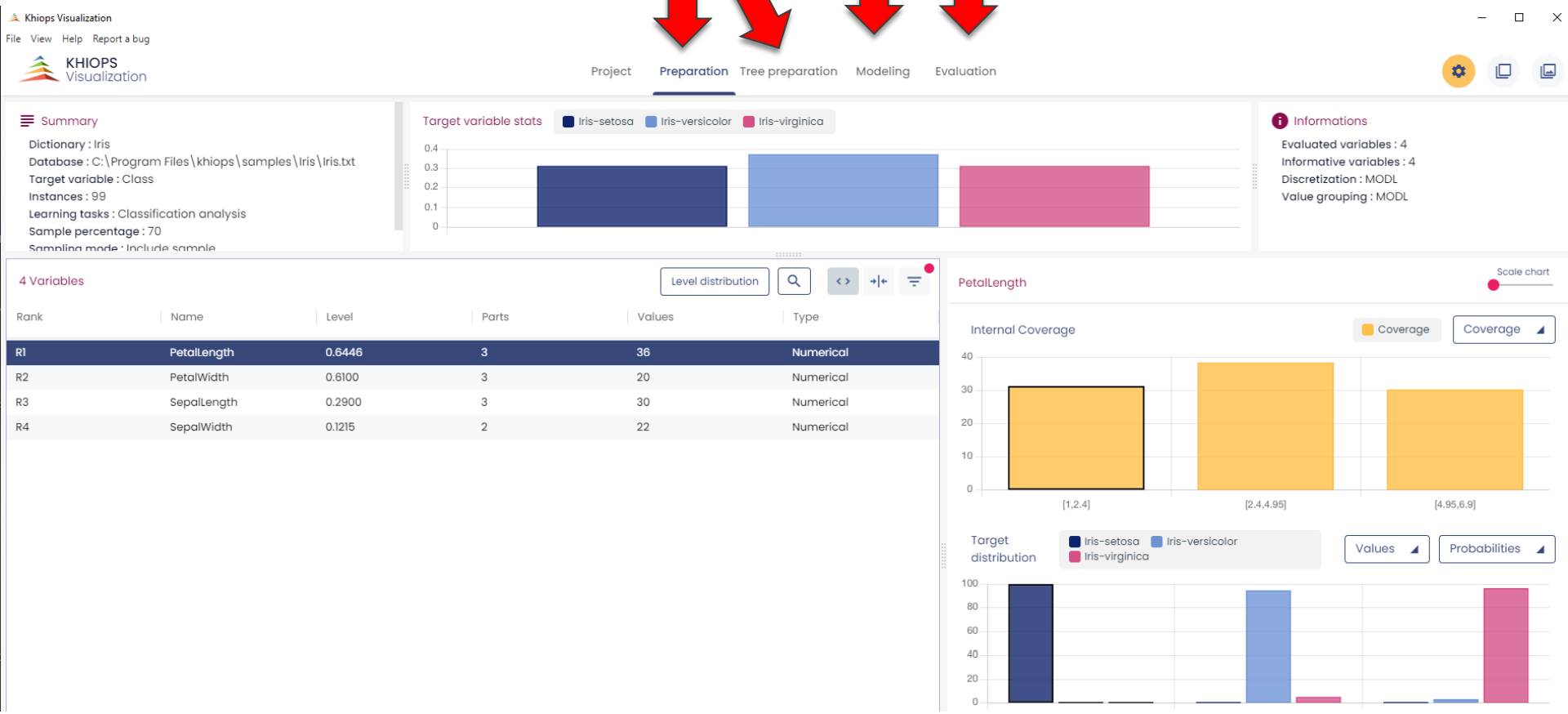
2 - Inspect the results using Khiops Visualization
(double-click on *.khj* file)

Exploratory of classification results using Khiops Visualization

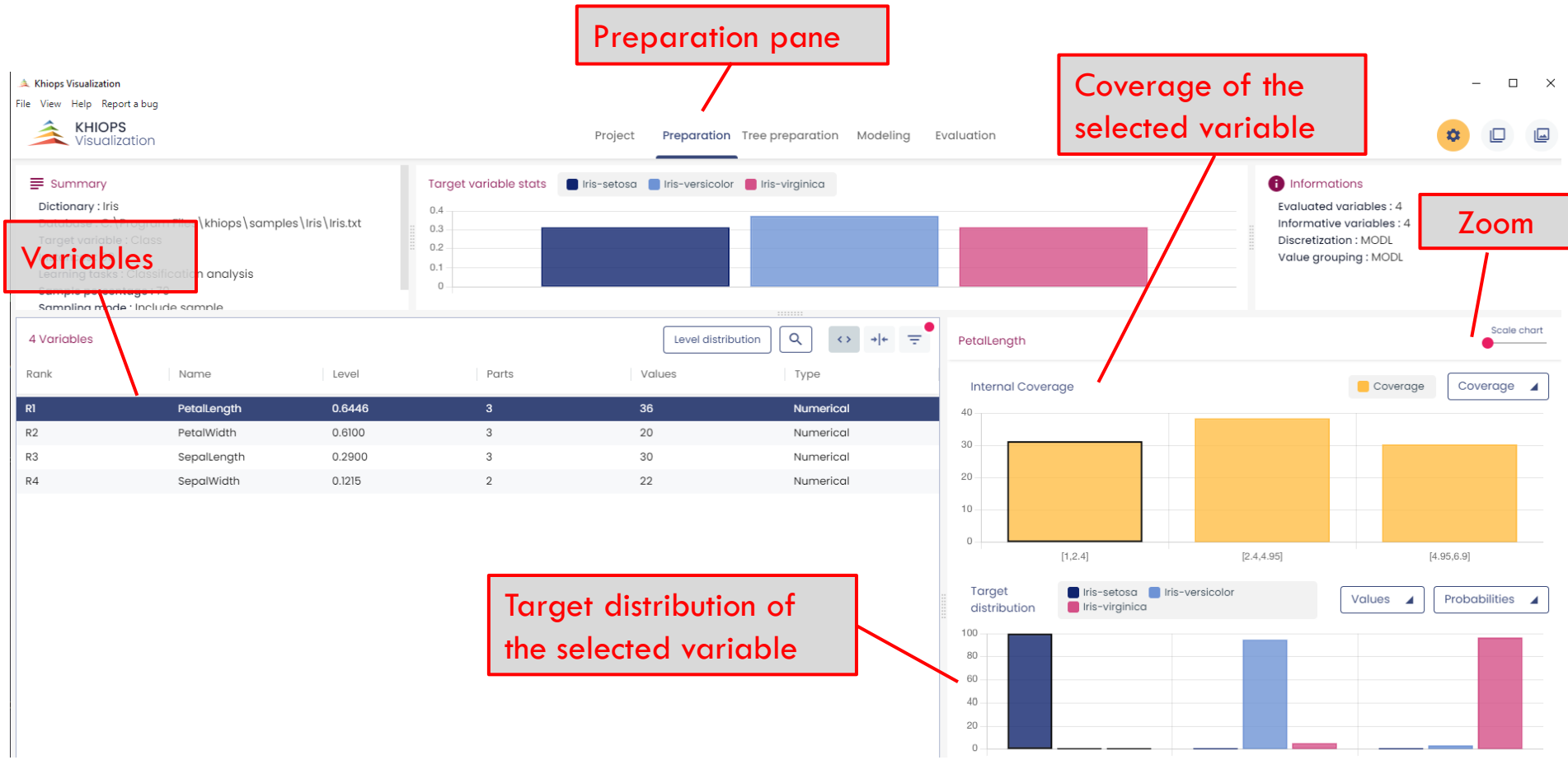
1 - Preparation

2 - Modeling

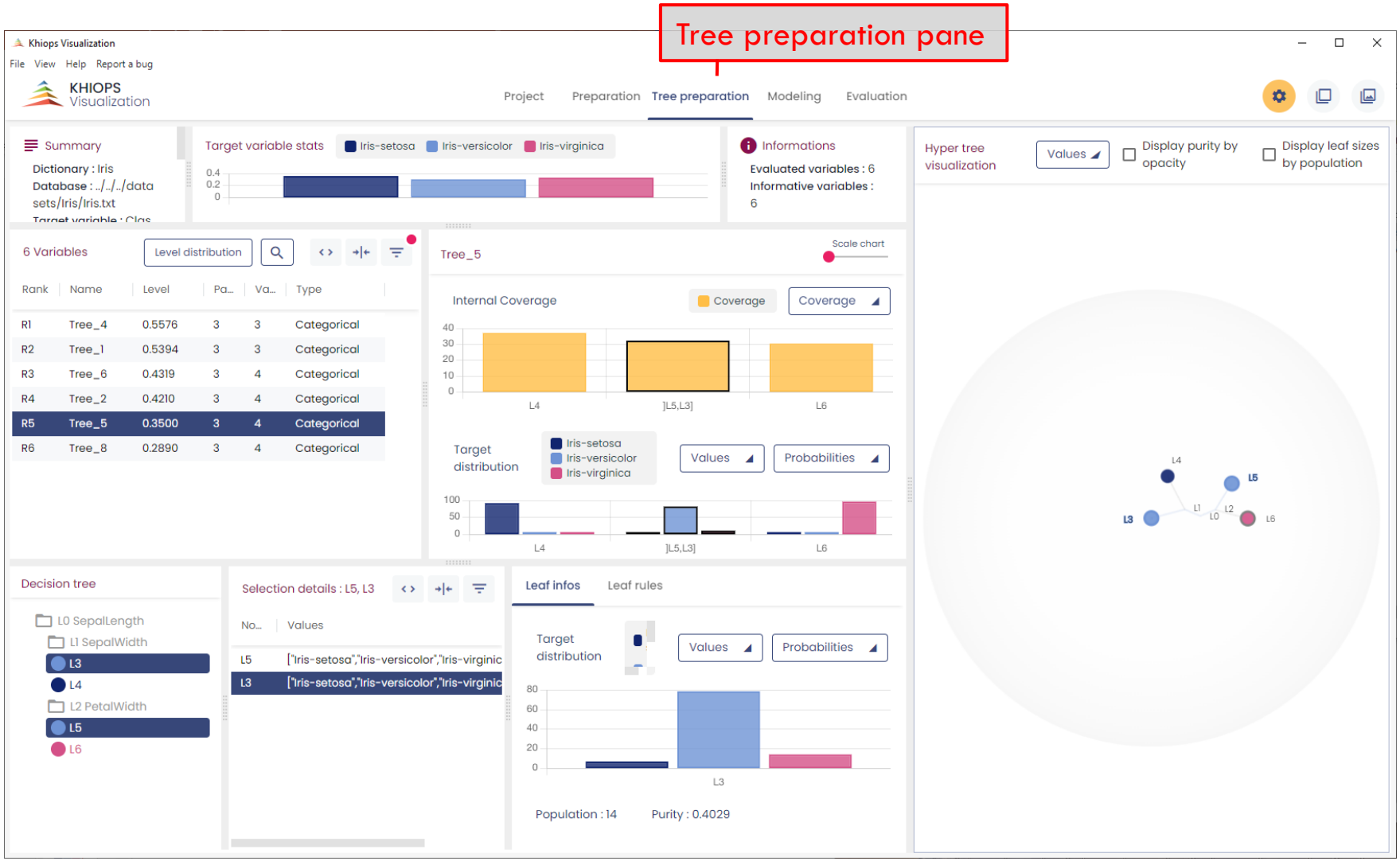
3 - Evaluation



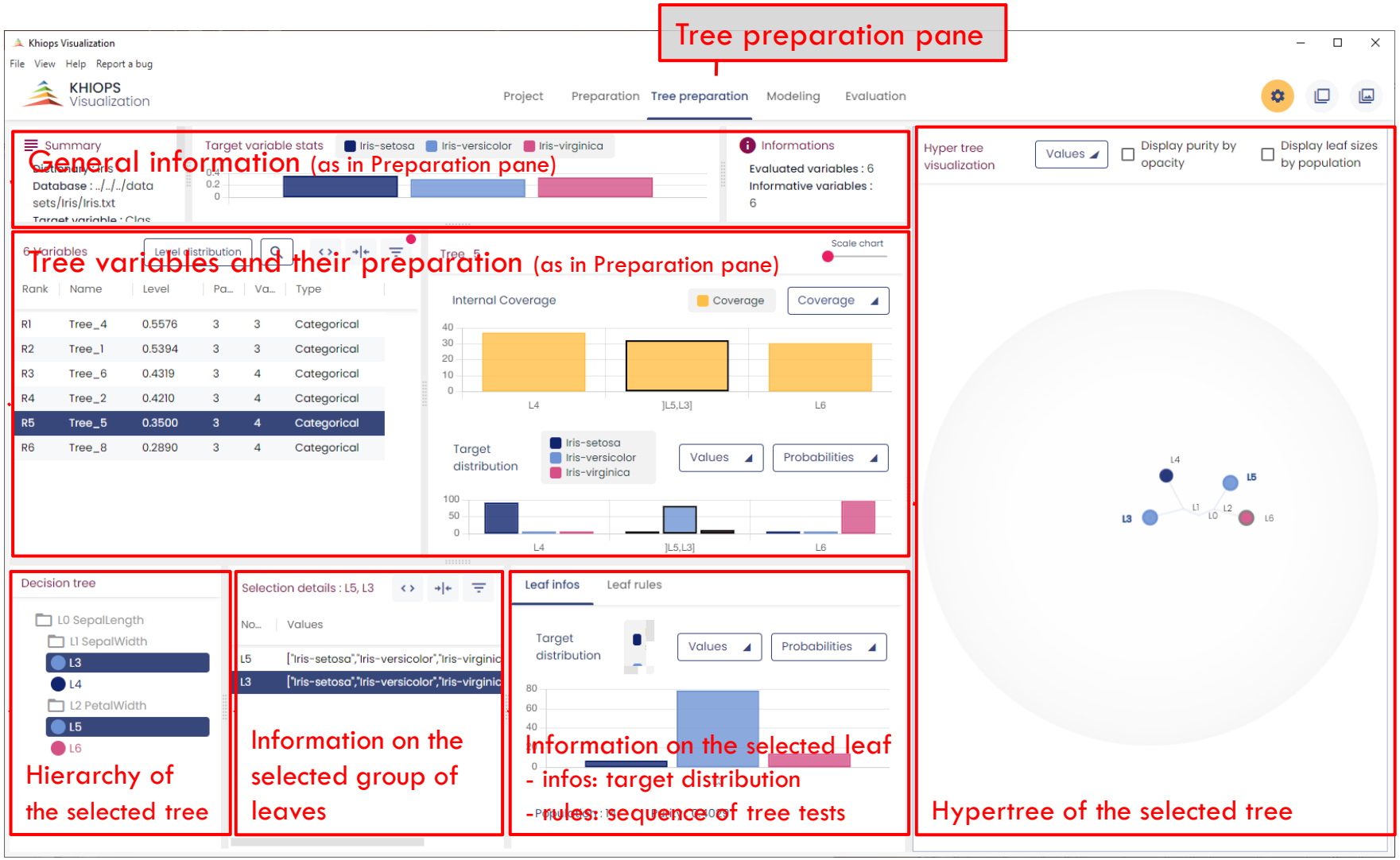
Exploratory of classification results using Khiops Visualization



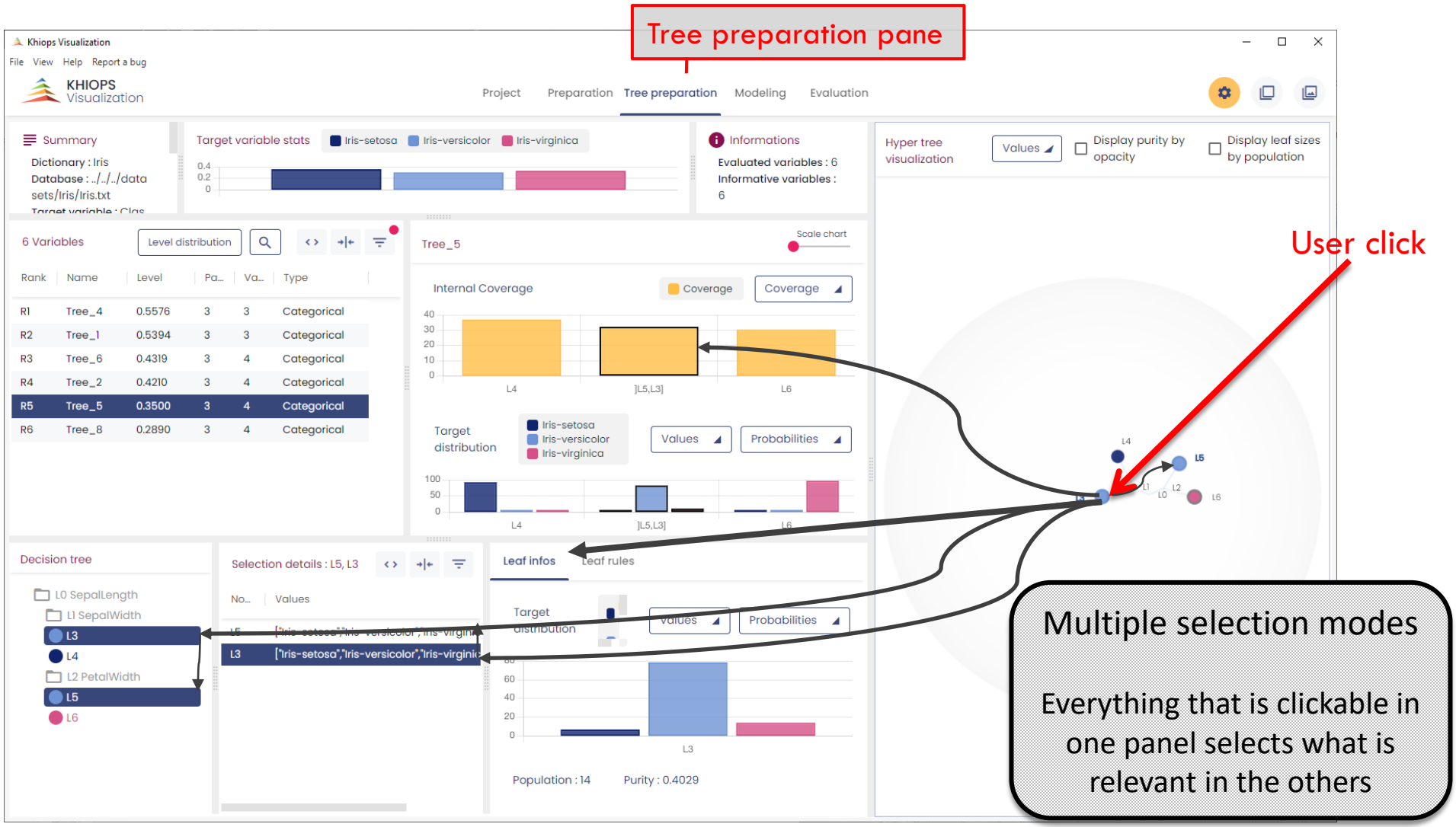
Exploratory of classification results using Khiops Visualization



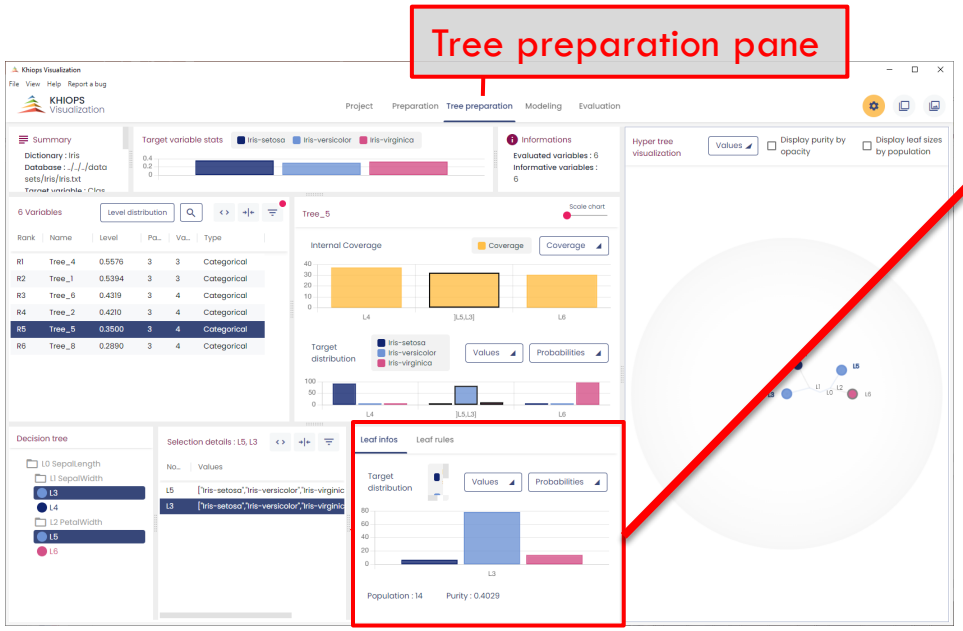
Exploratory of classification results using Khiops Visualization



Exploratory of classification results using Khiops Visualization



Exploratory of classification results using Khiops Visualization



Information on selected leaf

Leaf infos

Target distribution in leaf

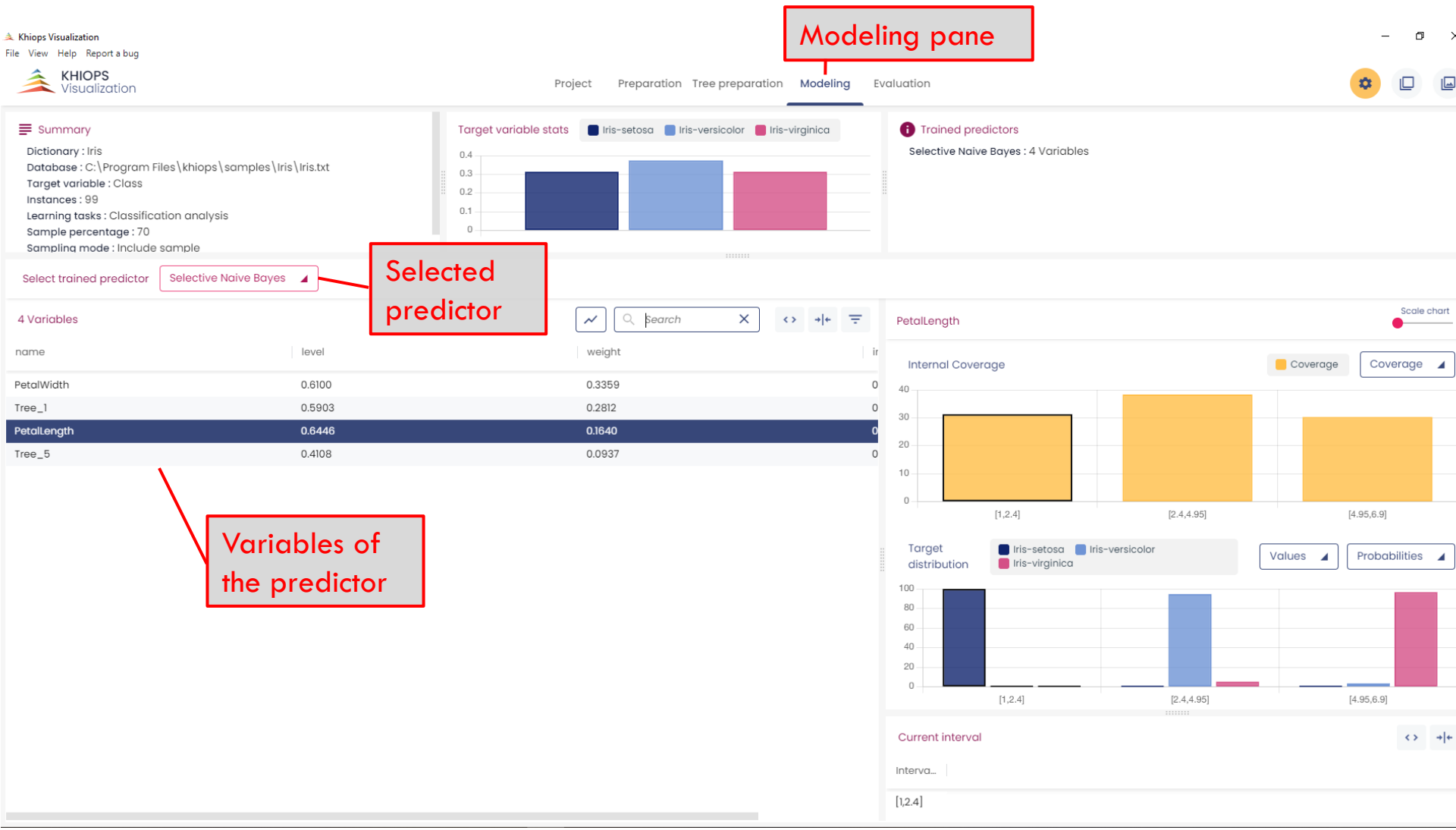
Leaf rules

Sequence of trre rules leading to the leaf

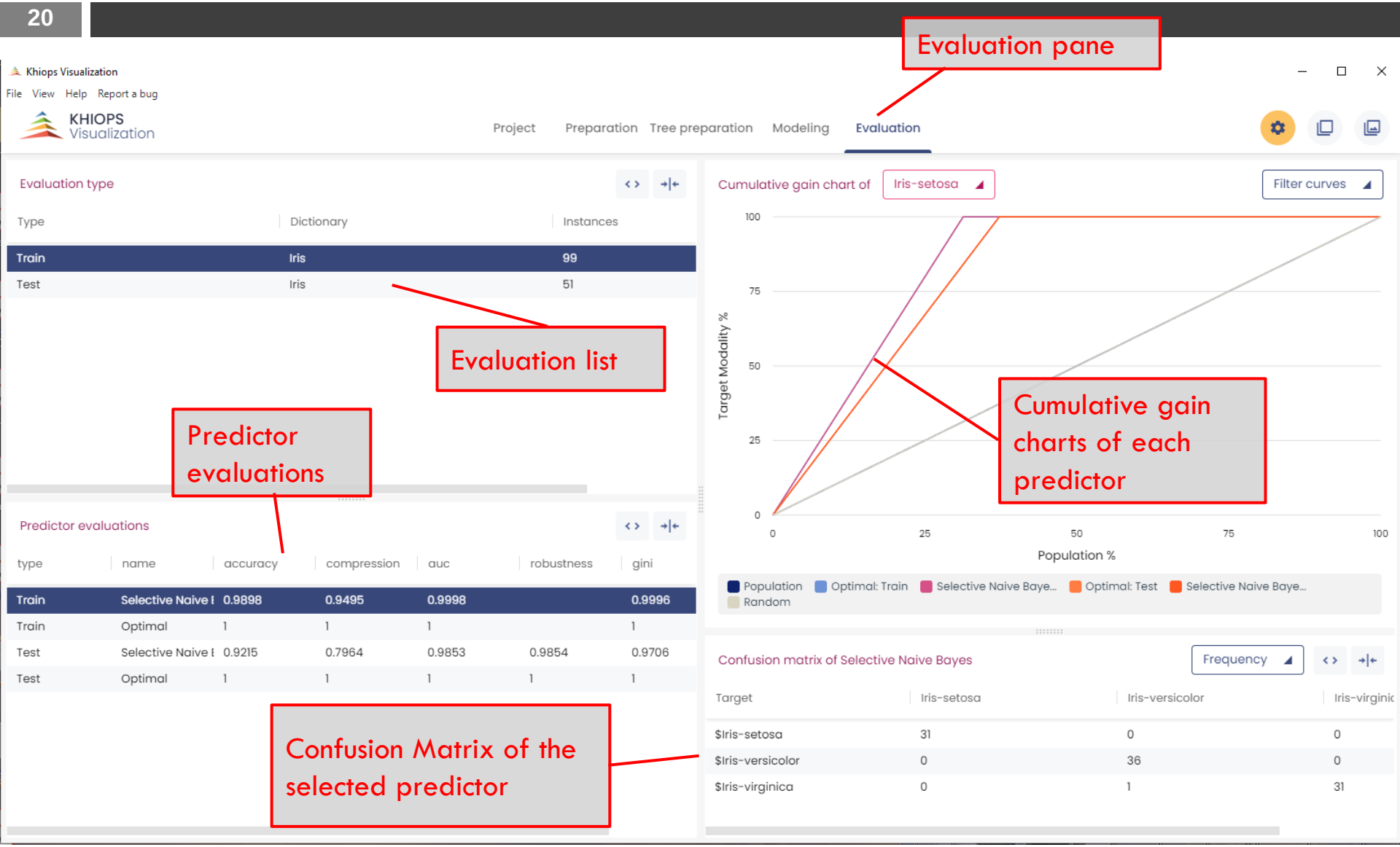
Leaf rules : L4

Variable ↑	Type	Partition
SepalLength	Numerical	[4.3, 5.75]
SepalWidth	Numerical	[2.95, 4.4]

Exploratory of classification results using Khiops Visualization



Exploratory of classification results using Khiops Visualization



Exercises A and B …

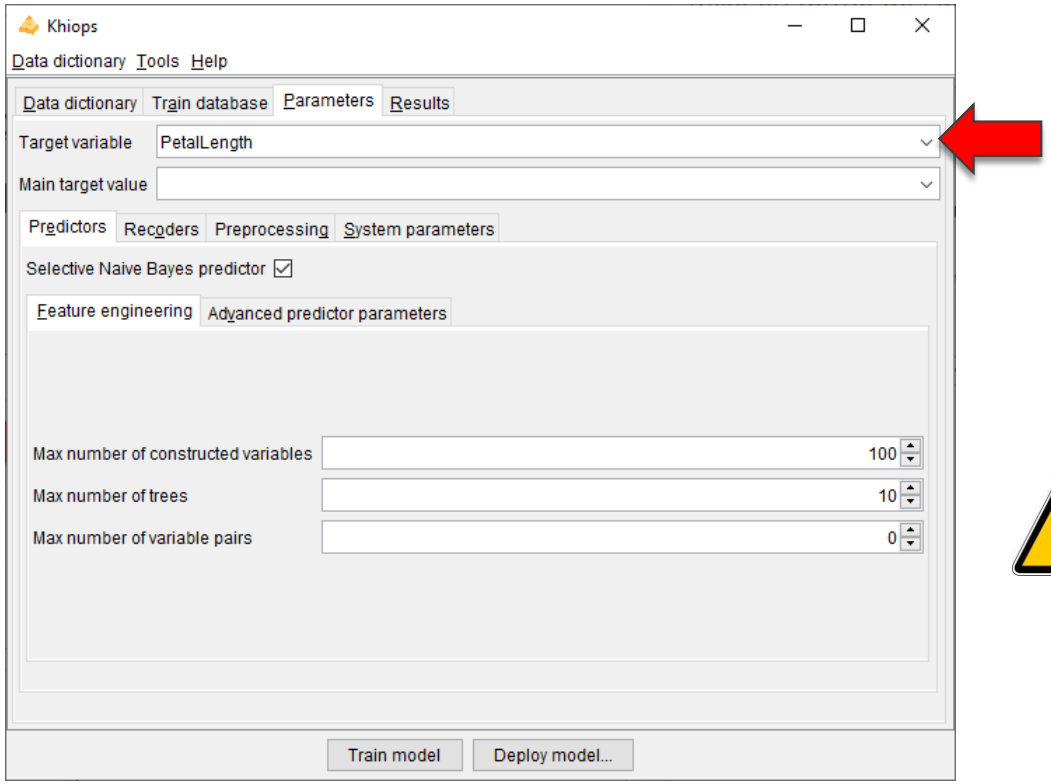
A : Perform a supervised classification on sample database Iris

B : Perform a Supervised classification on sample database Adult

➡ Interpret the analysis results

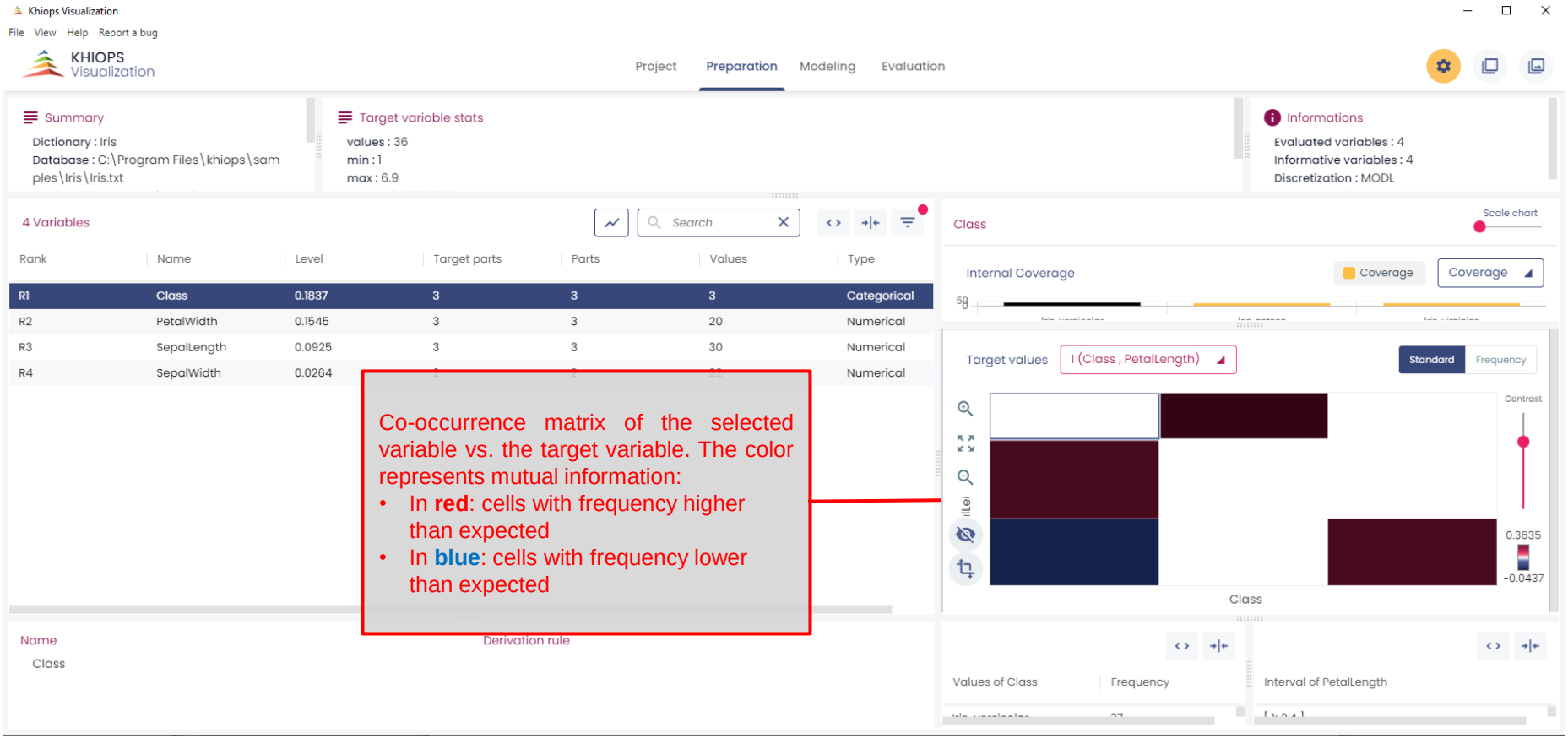
Regression (supervised)

- Same as classification
  with a numerical target variable



In this case, bivariate analysis and tree construction are not available!

Exploratory of regression results using Khiops Visualization



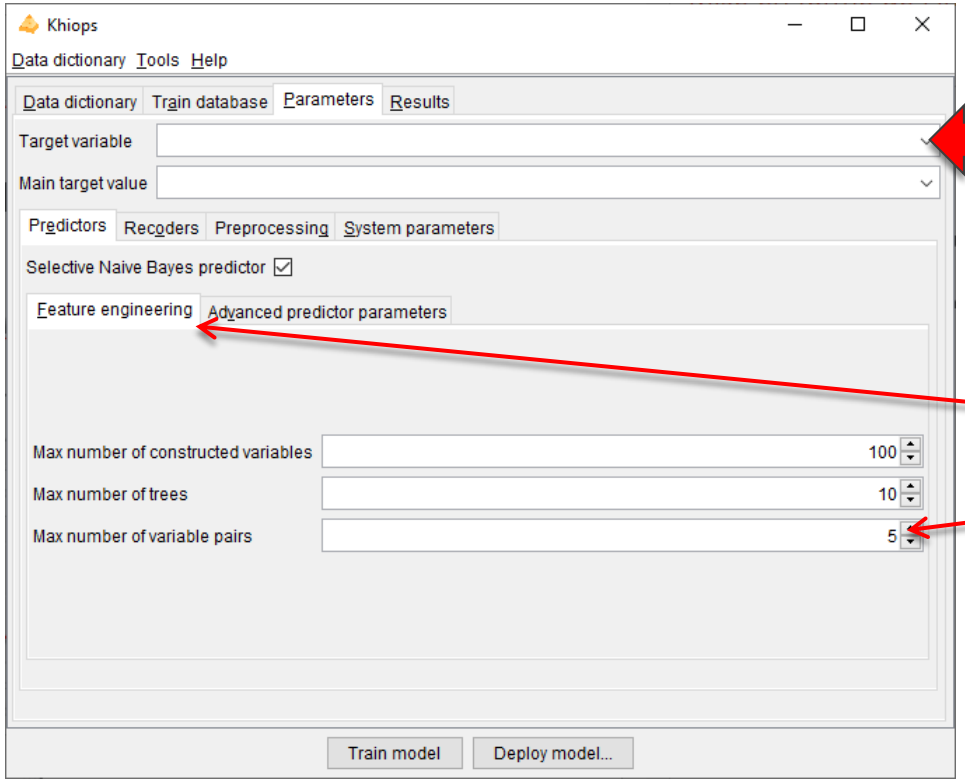
Exercise C …

C : Perform a regression of variable PetalLength of Iris

➡ Interpret the analysis results

Correlation analysis
(unsupervised, bivariate)

- Train a correlation model between two variables
  (categorical, numerical, both)



1 – Target variable must be empty

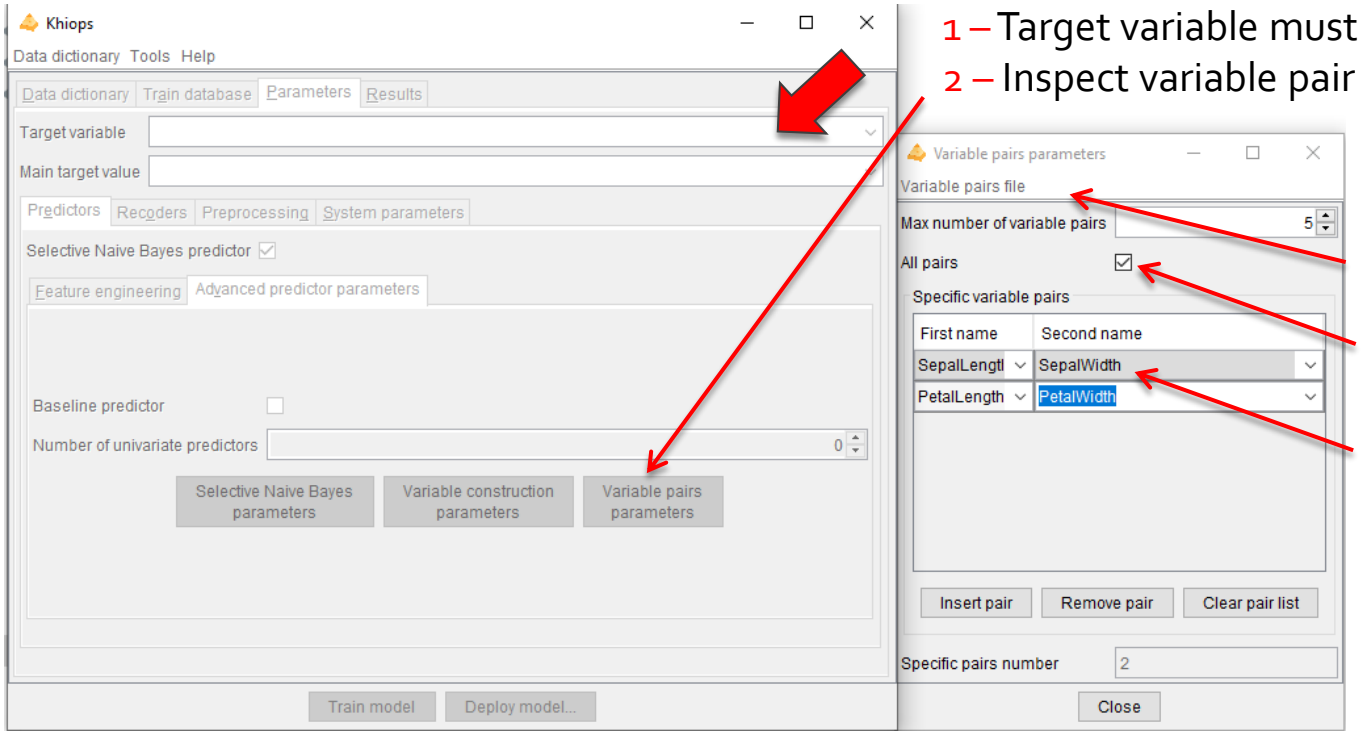
2 – Activate bivariate analysis

a – Feature engineering pane

b – Choice of a max number of pairs to analyze

Correlation analysis (unsupervised, bivariate)

- Train a correlation model : advanced parameters



1 – Target variable must be empty

2 – Inspect variable pair parameters

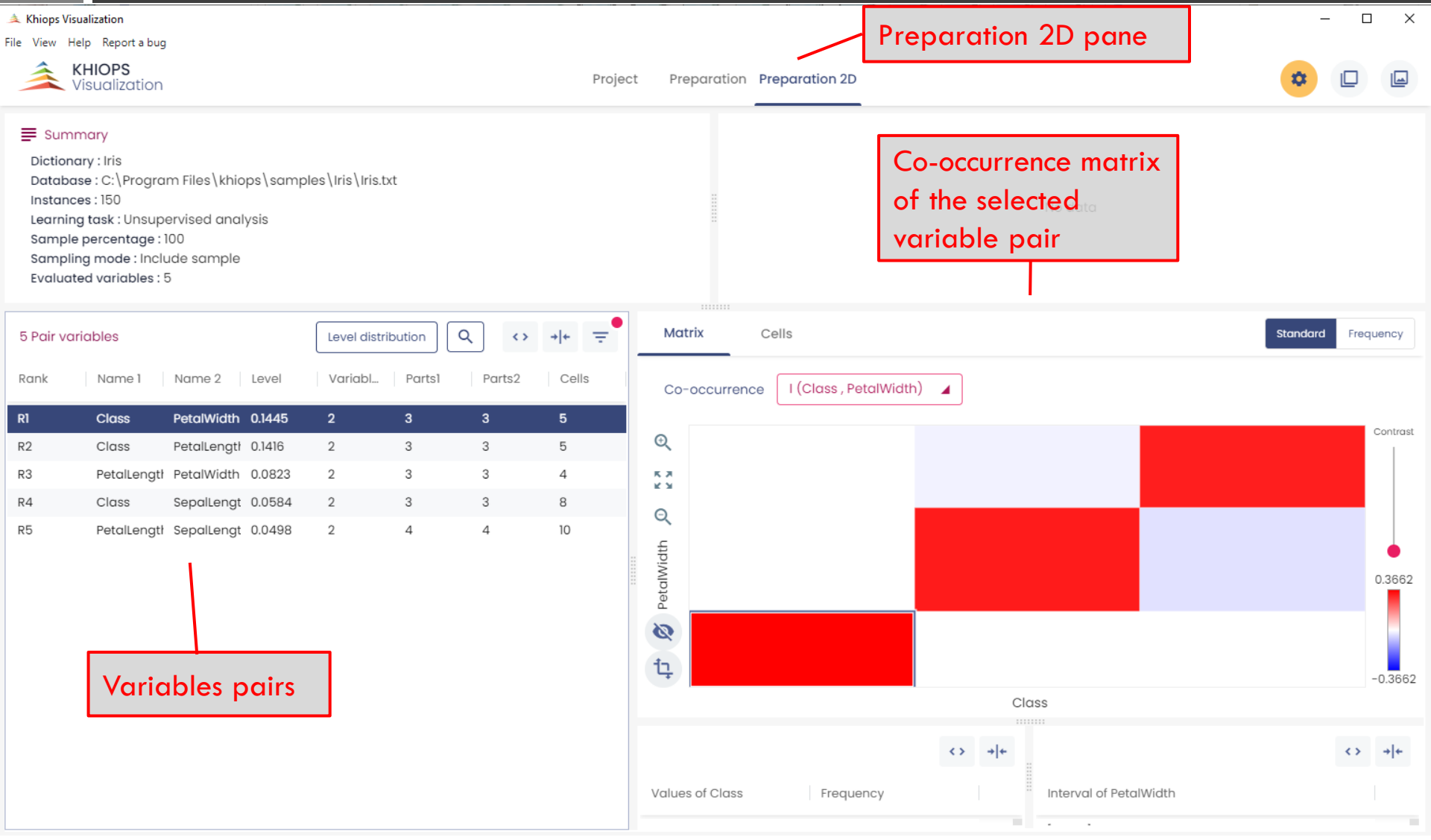
3 – Specify the pairs

a – import/export variable pairs file

b - all potential pair

c - individual pairs or families of variable pairs involving certain variables to analyze

Exploratory of correlation results using Khiops Visualization



Exercises D, E, F and G…

D : Perform the correlation analysis of the two most correlated variables of Iris
(*tip: analyze all pairs to identify the most informative*)

E : Idem with variables *PetalLength* and *PetalWidth*
(*tip: inspect the Variable pairs parameters)*

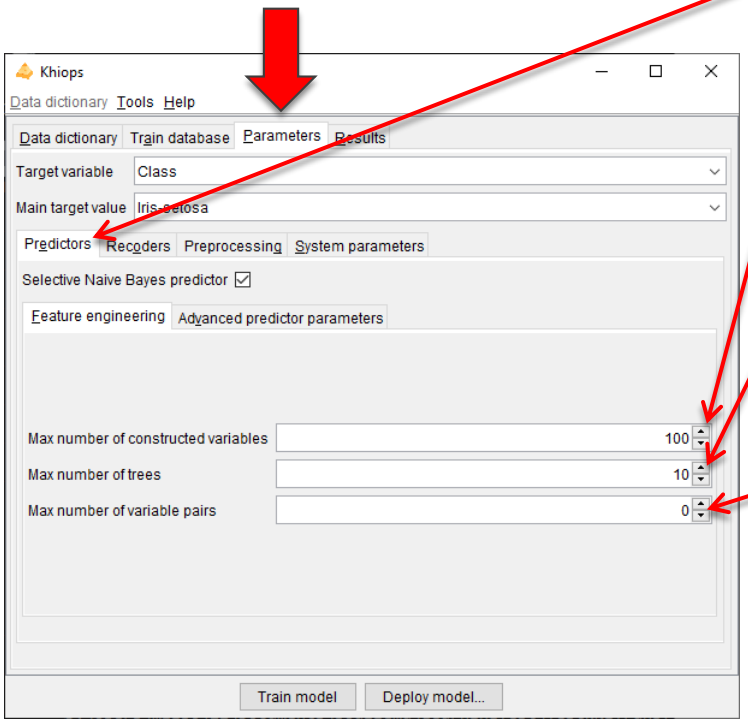
F : Idem with new constructed variables *PetalArea* and *SepalArea*
(*tip: use the derivation rule Product in dictionary,*
see KhiopsGuide: sections « Derivation rules » and « Appendix »)

G : Perform the correlation analysis of all pairs of Adult involving variable *native_country*

➡ Interpret the analysis results

Variable construction

- Parameters



Predictors

Feature engineering

- Max number of constructed variables
  - to build an analyze table from a multi-table schema (see later)
  - automatic extraction of complex information to obtain accurate classifiers
- Max number of trees
  - combines natives or constructed variables to extract complex information
  - better accuracy, at the expense of interpretability
- Max number of pairs of variable
  - to understand correlation between variables
  - use rather for exploratory analysis rather than for better accuracy

Recommendation

- start with few constructed variables, and increase incrementally
- idem for trees
  - no tree for simpler, faster and more interpretable predictors
  - more and more trees for more accurate predictors

Exercise H …

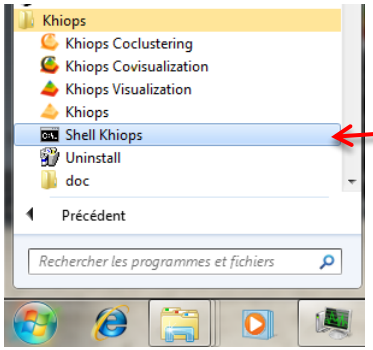
A : Perform a supervised classification on sample database Letter
Build 0, 10, 50 trees

➡ Interpret the analysis results, and the trade-off between number of trees, training time and test accuracy

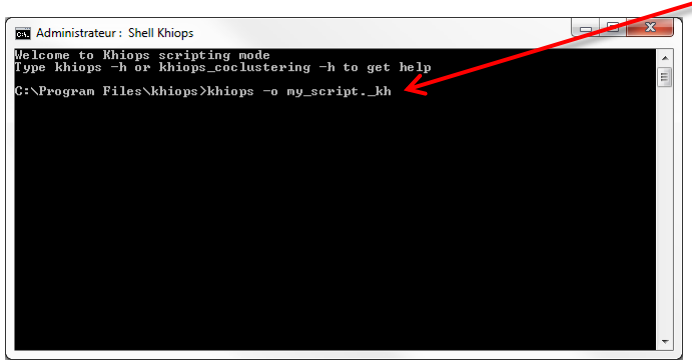
Integration in information systems

- Batch mode
  - to record and replay Khiops scripts
  - to perform any Khiops task from any programming language
  - see next slide
- Khiops Native Interface (KNI)
  - dynamic link library (DLL) for online deployment of Khiops models
  - package to download from https://khiops.org
- Python Khiops Library (pykhiops)
  - to perform any Khiops task from python
  - to inspect any Khiops analysis results from python
  - python package available from https://khiops.org
- JSON file exports
  - Khiops dictionaries and analysis results can be exported from the Khiops tool to exploit Khiops results from any programming language

Batch mode



Start a Shell Khiops



Record a script « automatically» using Khiops user interface

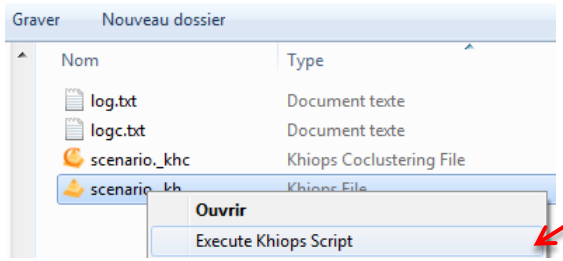
khiops –o my_script._kh

o = output

- Replay a script from the shell

khiops –i my_script._kh

i = input



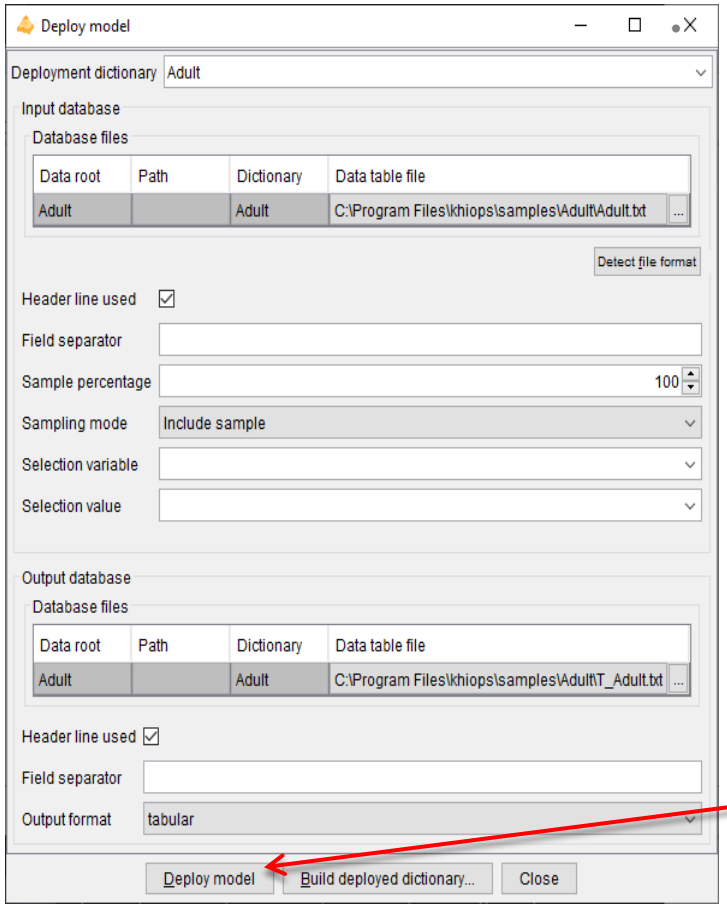
Replay a script from Windows Explorer

right click on script file

Exercise I …

I : Record a script file, then replay it …

Deploy a model



Steps for model deployment

- 1- Start from a modeling dictionary « *Modeling.kdic* »
  - In « Data dictionary » pane
- 2- Choose the variables to deploy
  - Inspect the modeling dictionary In « Data dictionary » pane by right-click in the "Dictionaries in file" list
  - Suppress the « *Unused* » tag from identifier variables
  - Select the prediction variables to deploy
- 3- Menu : « *Tools -> Deploy model* »
- 4- Deploy model dialog box
  - Select deployment dictionary
  - Select input database
  - Select output database
  - Click on « Deploy model » button

Exercise J …

J: Deploy a classifier on database Iris

Khiops Coclustering & Khiops Covisualization

- **Khiops Coclustering**
  - Correlation analysis of two or more variables using a hierarchical coclustering model

- **Khiops Covisualization**
  - Exploratory analysis of Khiops Coclustering results using an interactive visualization tool

Khiops Coclustering &
Khiops Covisualization

- Train a coclustering model
  - Use of Khiops Coclustering back-end tool
  - Co-partition of two or more categorical or numerical variables
  - At each level of the hierarchy, the merge of clusters with the minimum information loss is performed
  - Write results in a coclustering report file « *.khcj* »

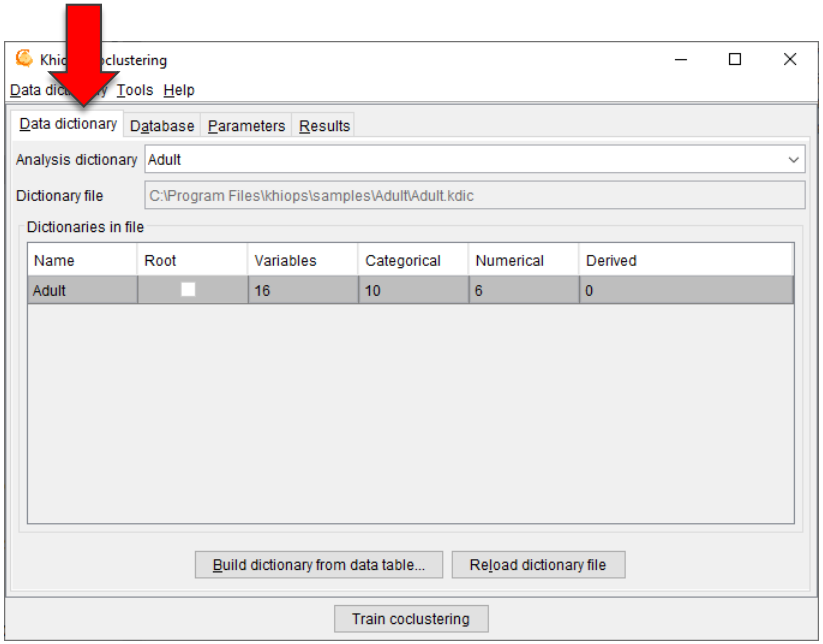
- Exploratory analysis of the results
  - Use of Khiops Covisualization tool
  - Navigation in the hierarchy of models

Train a coclustering model

- **Step 1 :** Open an existing dictionary

  (ex: sample Adult.kdic)

  - Description of variables to use during analysis



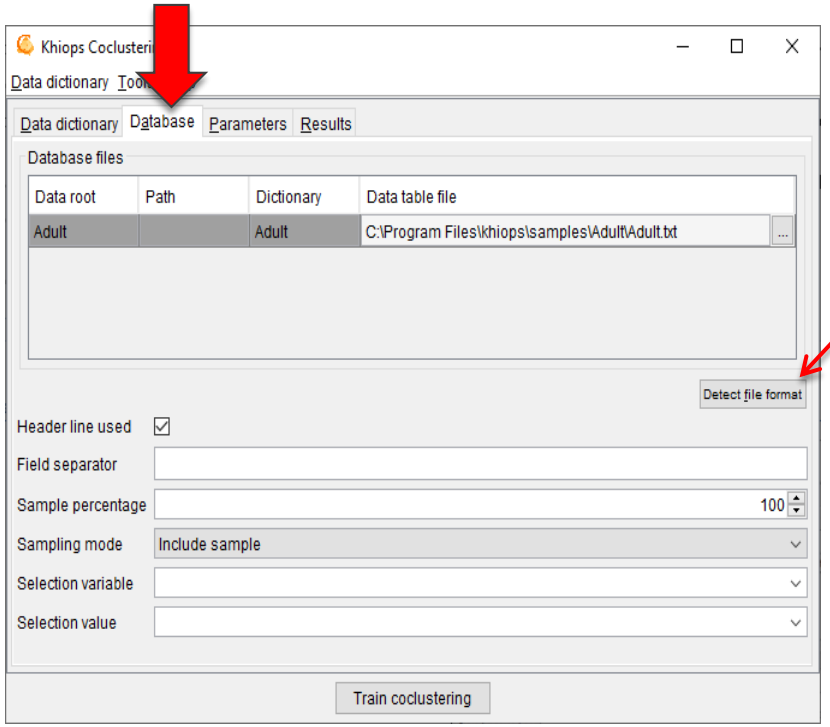
Available actions :

- Open, Save, Save as, Close
- Edition (menu « Dictionary file/Inspect current dictionary », or NotePad)
- Reload dictionary file
- Build dictionary from data table

```
Dictionary    Adult
{
              Numerical      Label;
              Numerical      age;
              Categorical    workclass;
              Numerical      fnlwgt;
              Categorical    education;
              Numerical      education_num;
              Categorical    marital_status;
              Categorical    occupation;
              Categorical    relationship;
              Categorical    race;
              Categorical    sex;
              Numerical      capital_gain;
              Numerical      capital_loss;
              Numerical      hours_per_week;
              Categorical    native_country;
              Categorical    class;
};
```

Build a coclustering model

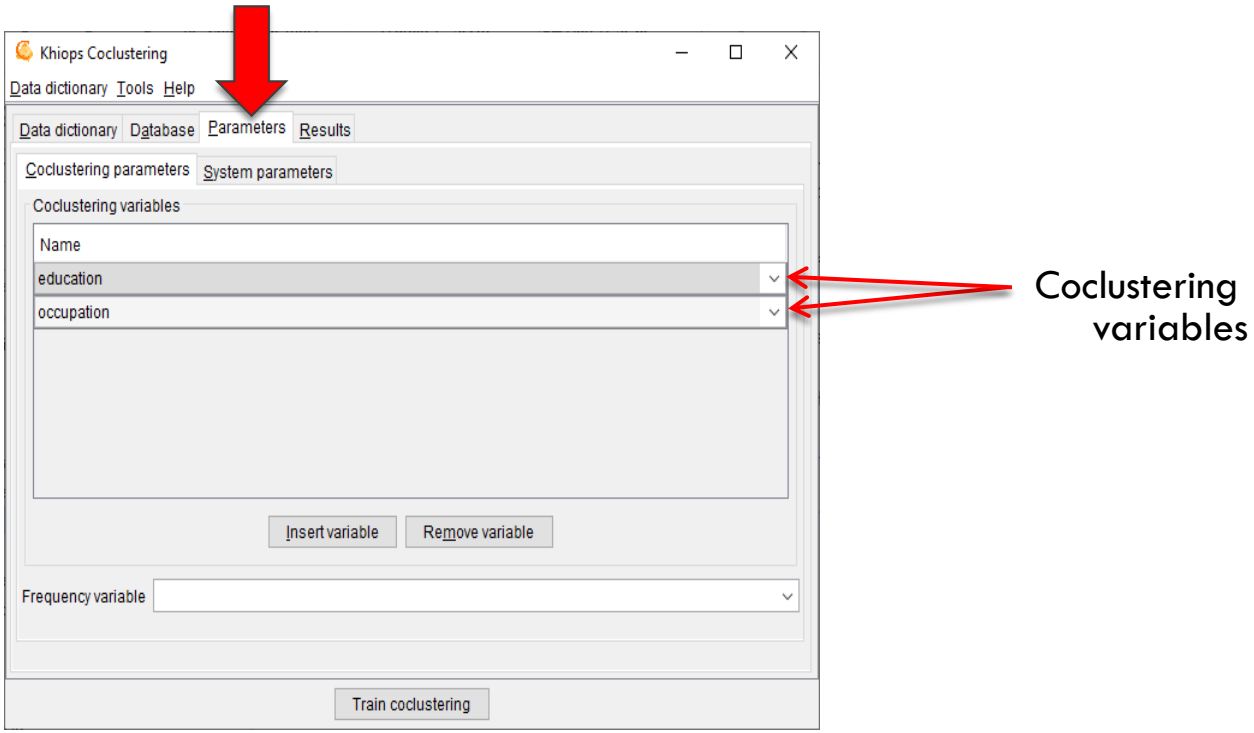
- **Step 2 :** Specification of used database



Detect file format : heuristic help that scans the first few lines to guess the file format. The header line and field separator are updated on success, with a warning or an error in the log window only if necessary.

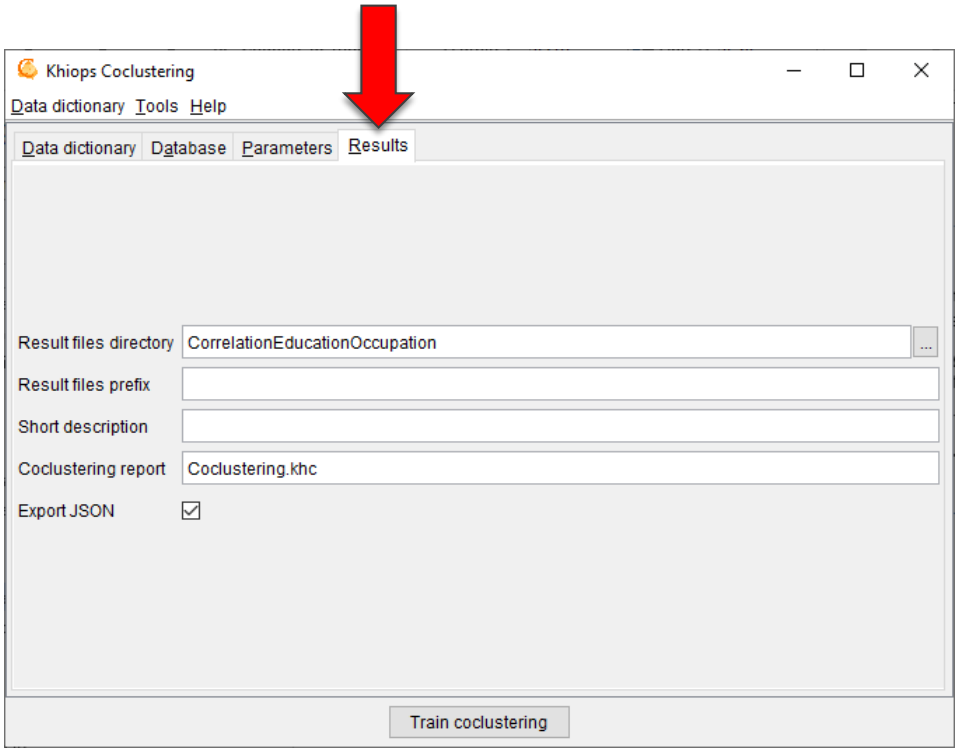
Build a coclustering model

- **Step 3 :** Specification of coclustering variables



Build a coclustering model

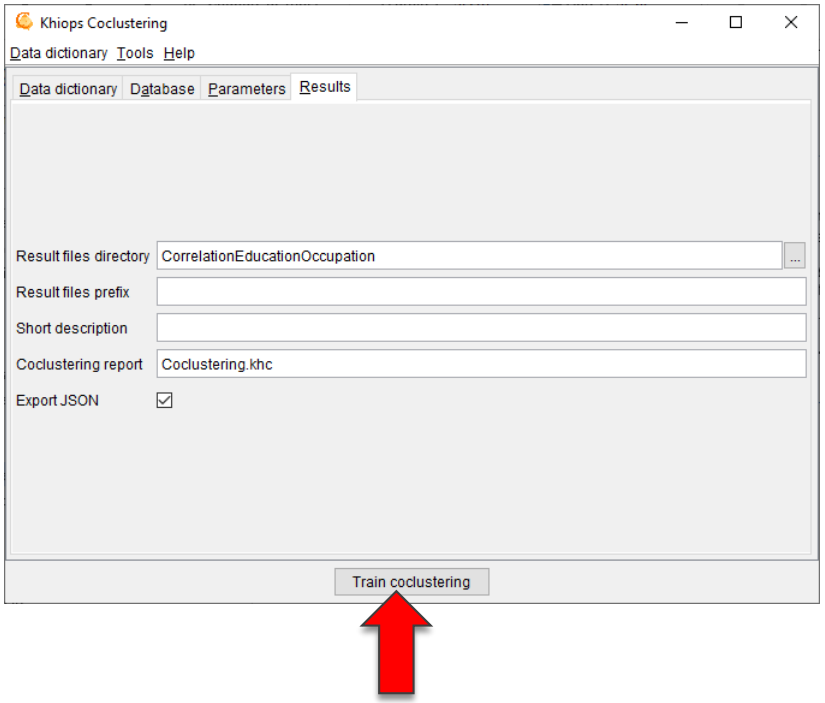
- **Step 4 : Results**



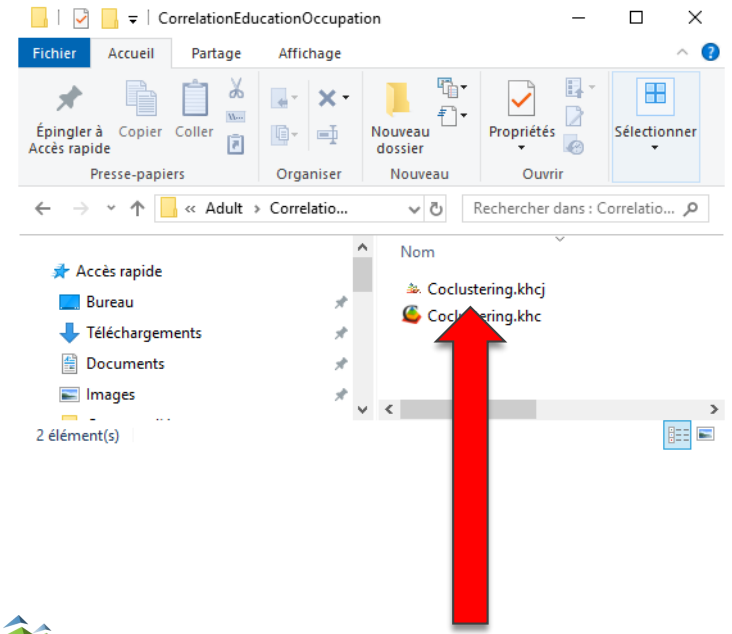
- Directory where result file is written
- Prefix (*ex: in case of several experiments*)
- Synthetic coclustering report (cf. Khiops Covisualization)
- Json report, to get the analysis results from external tools

Build a coclustering model

- **Step 5 :** Start the analysis



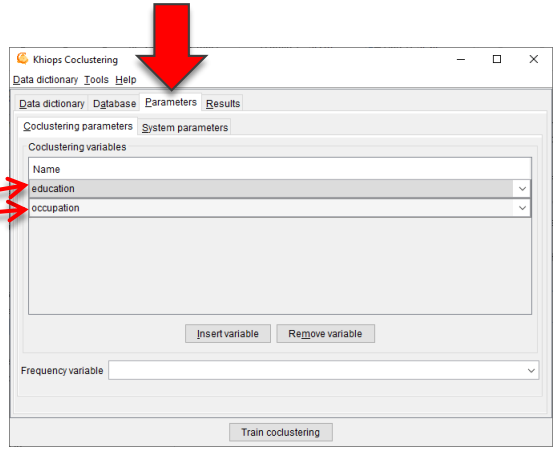
1 – Train the coclustering

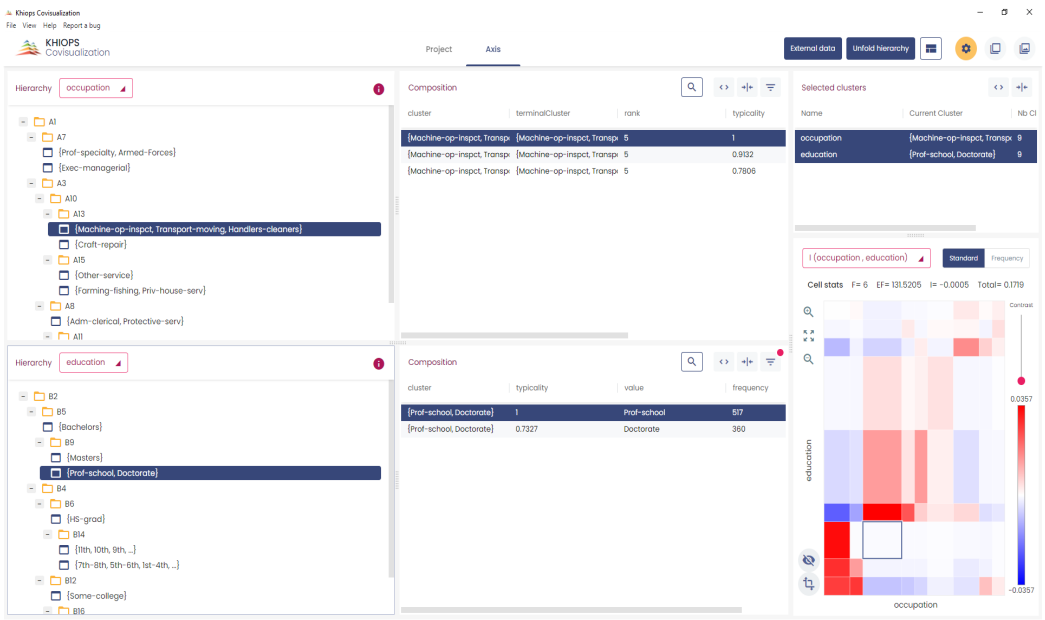


2 - Inspect the results using Khiops Covisualization
(double-click on *.khcj* file)

Example: base Adult education*occupation

- With Khiops Coclustering
  - Analysis of pair of variables education*occupation





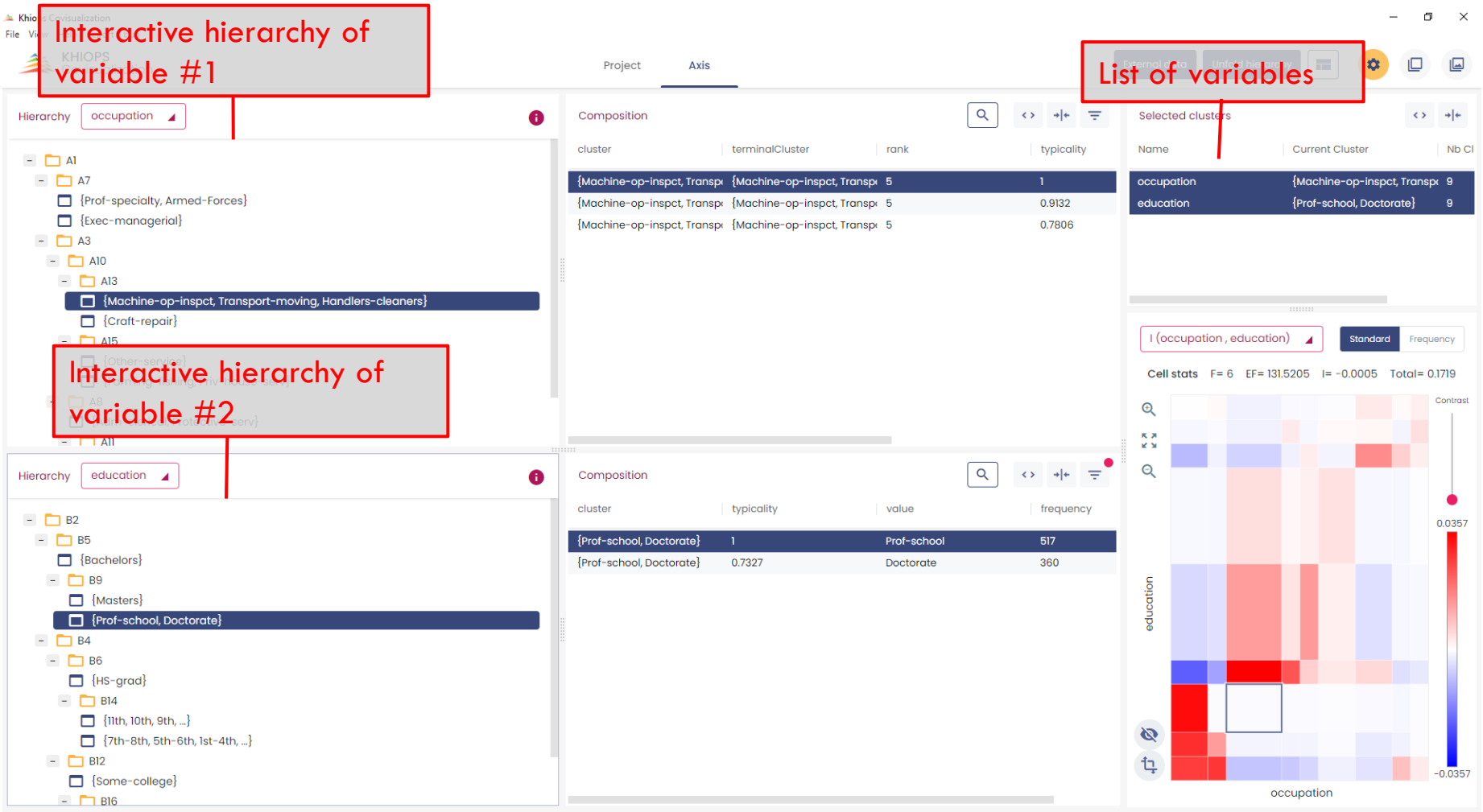
- With Khiops Covisualization
  - Exploratory analysis of the results

Exercise J …

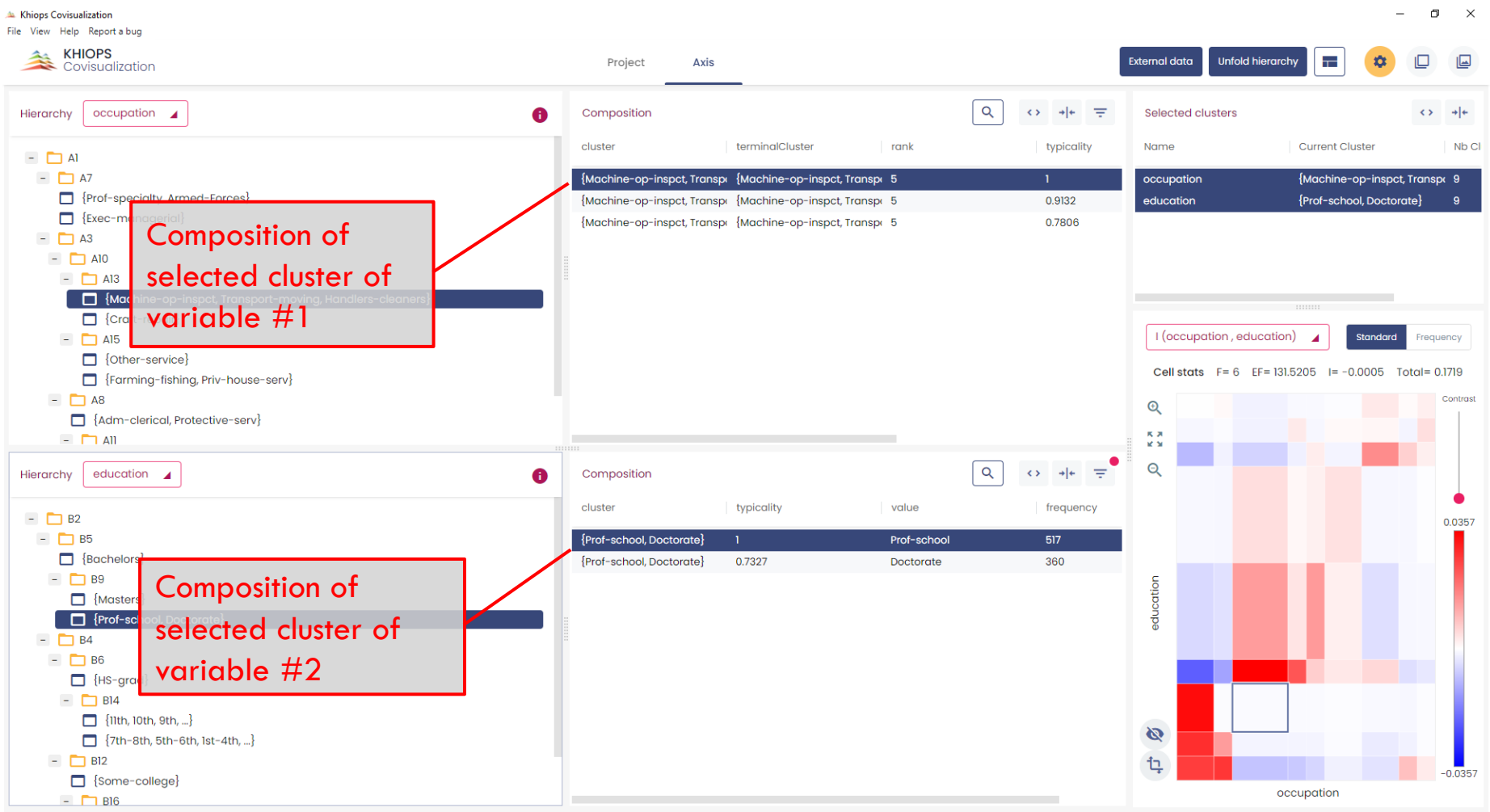
J : Train a coclustering model
on two categorical variables of sample database Adult

➡ Explore the analysis results

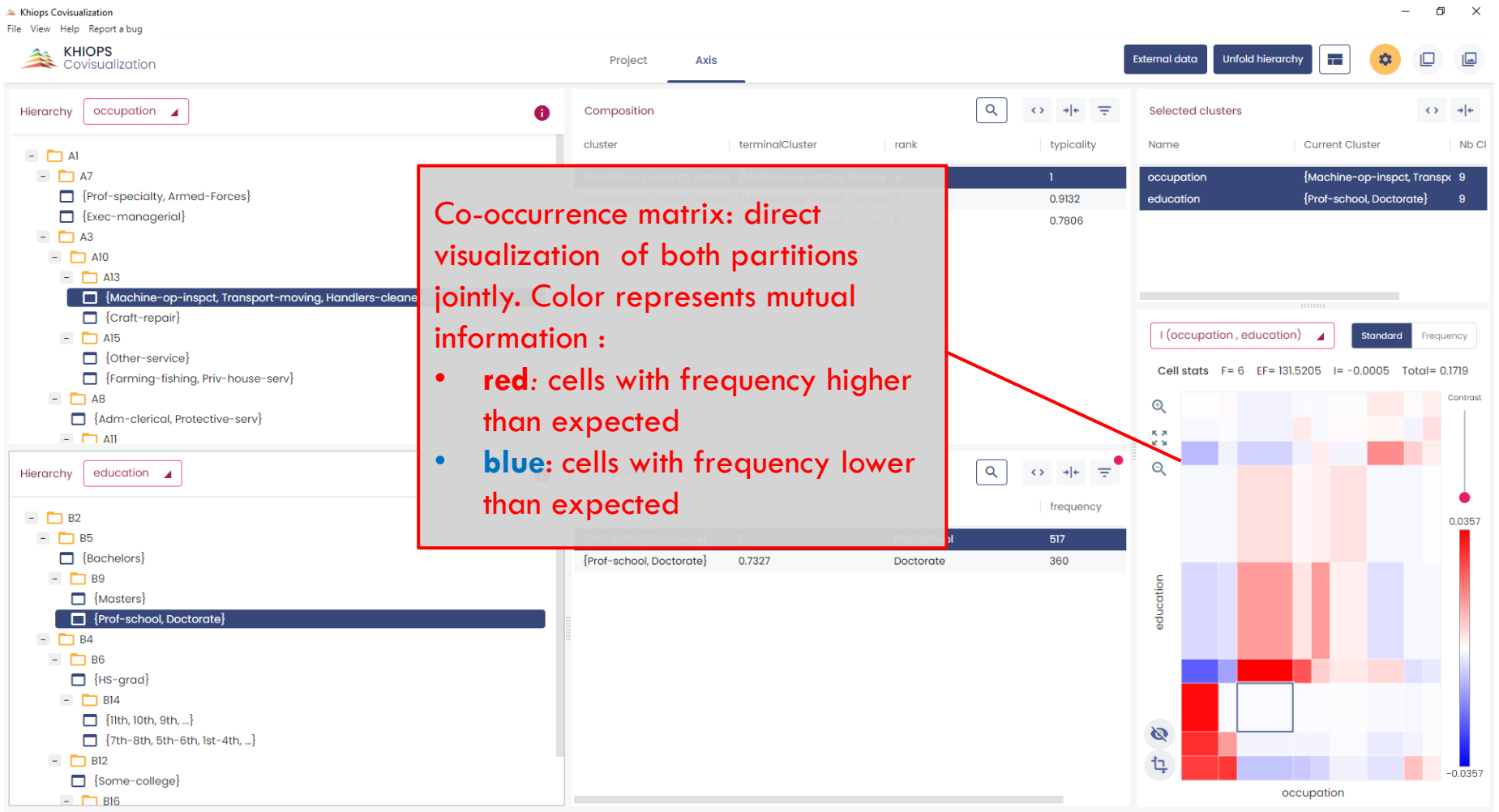
Khiops Covisualization



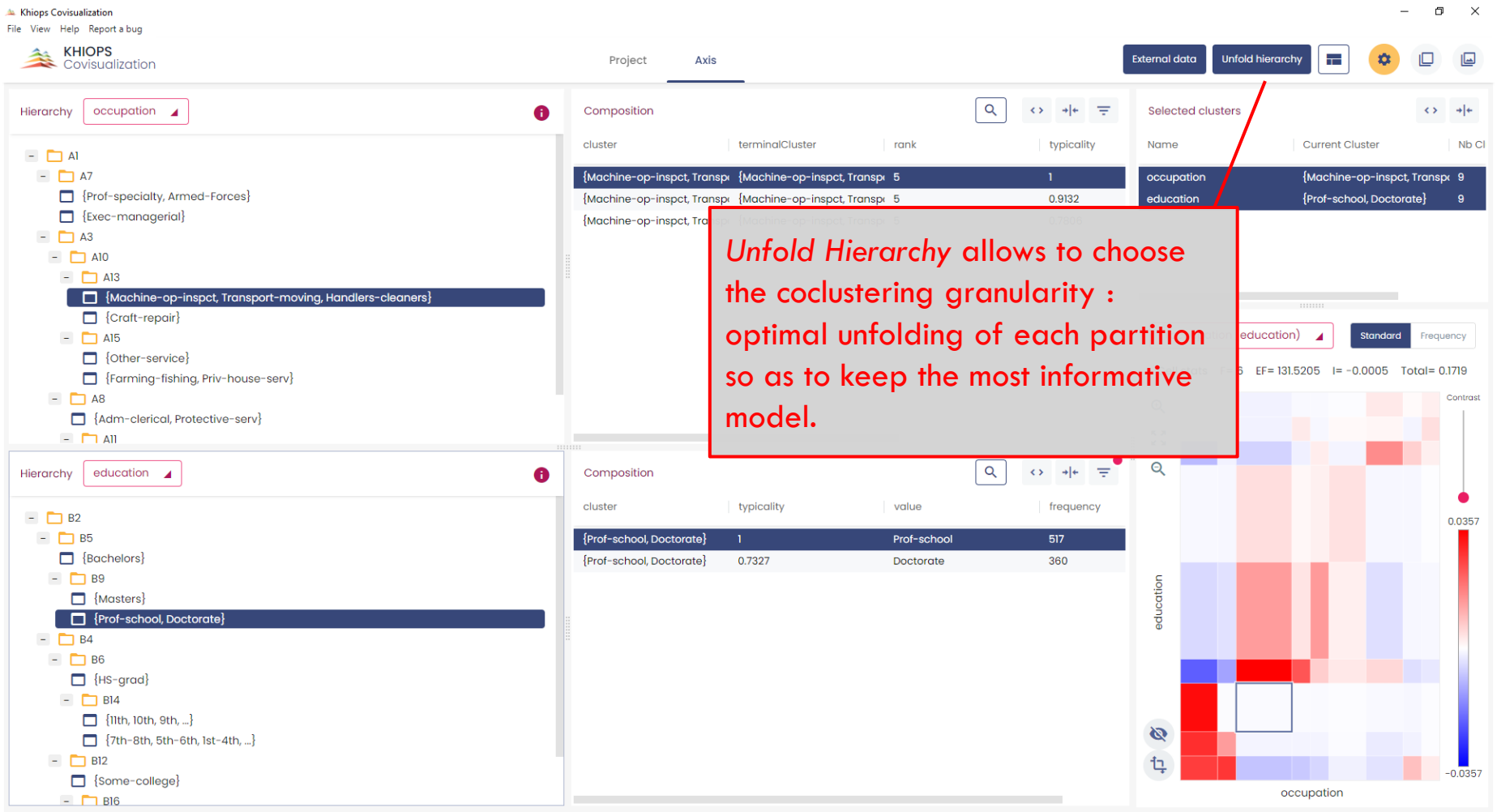
Khiops Covisualization



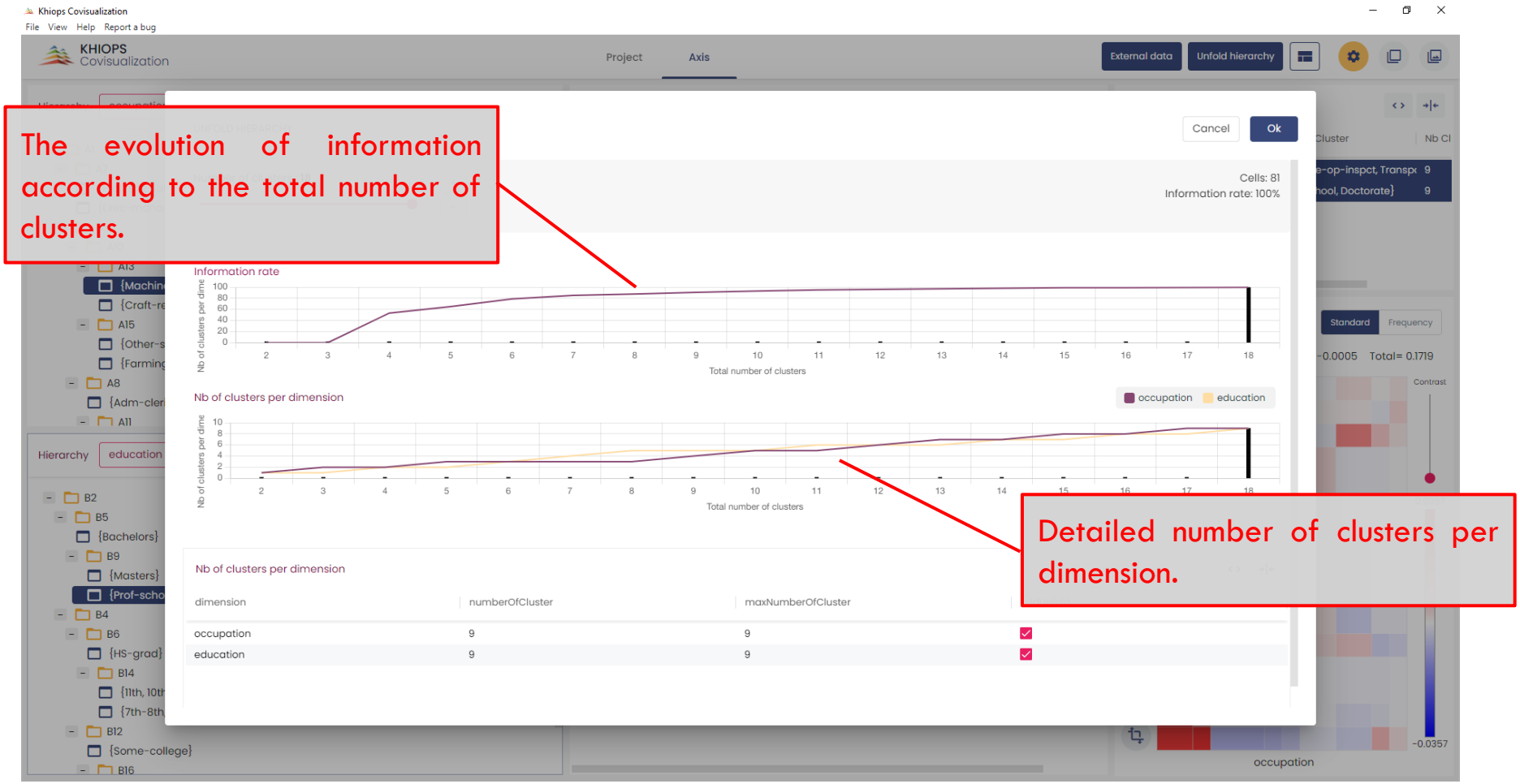
Khiops Covisualization



Khiops Covisualization

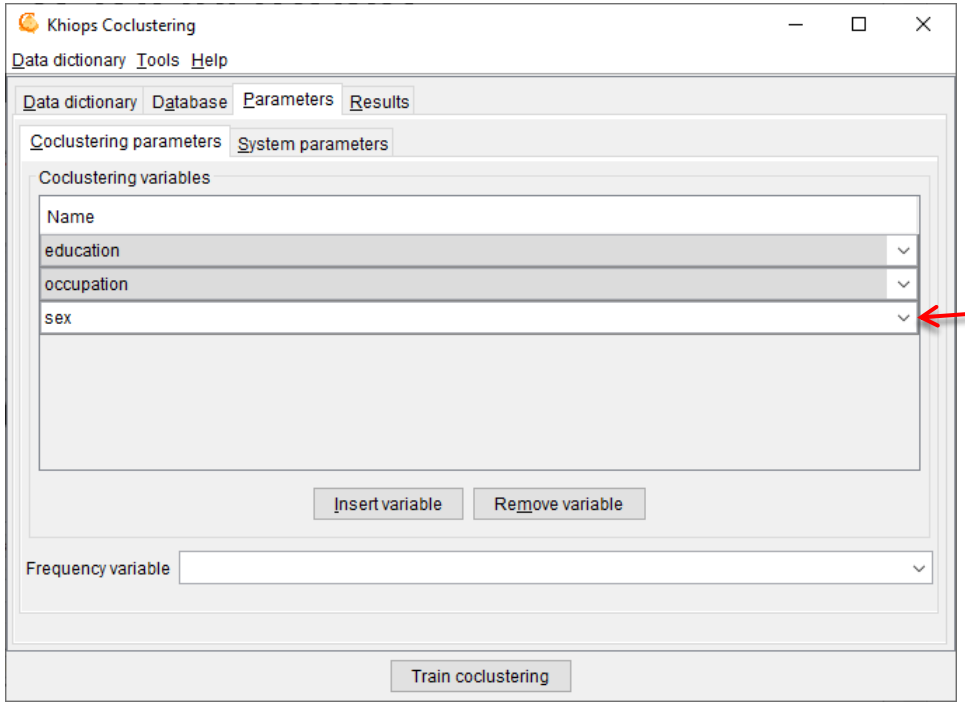


Khiops Covisualization



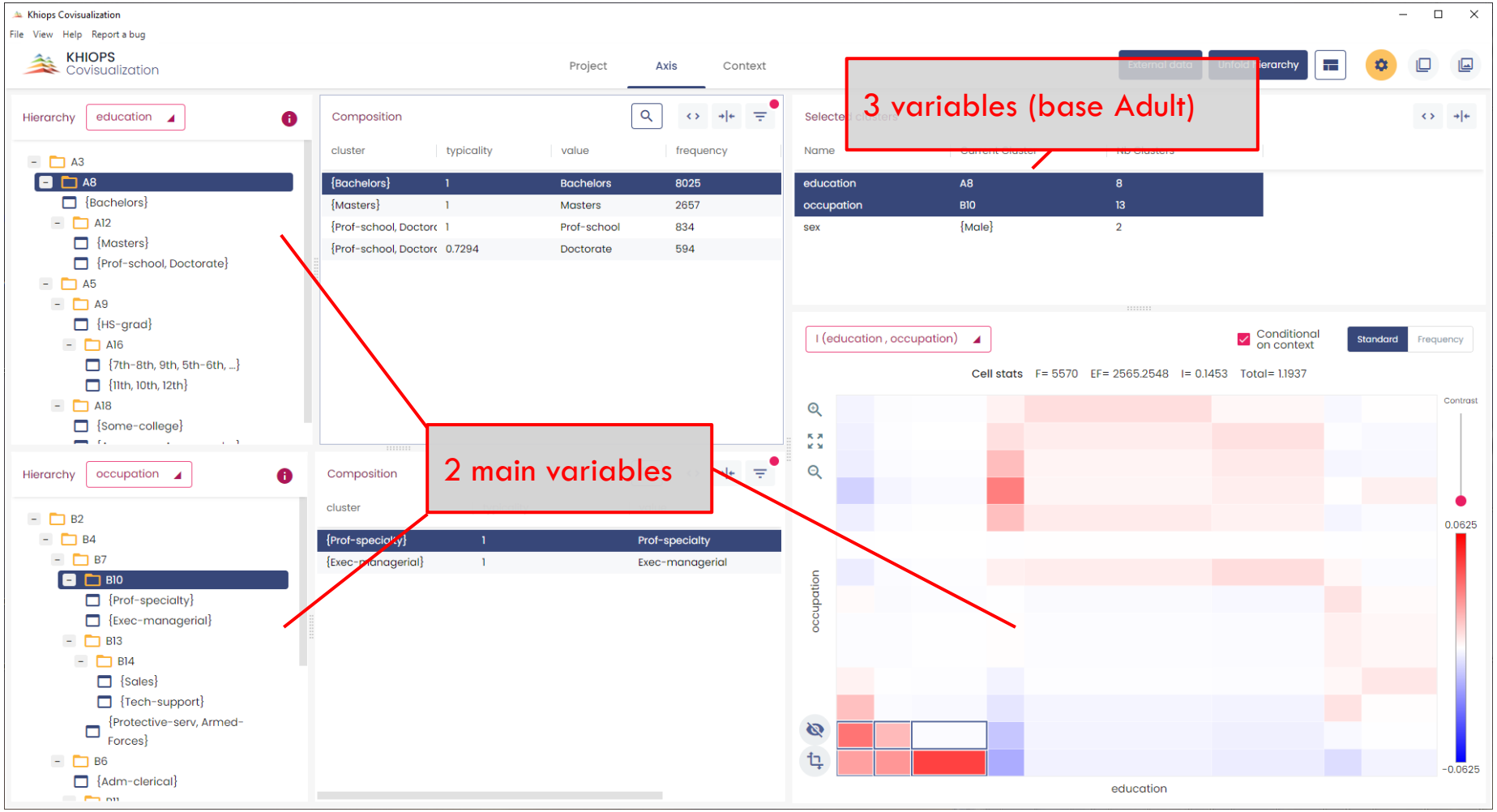
Training a triclustering

- Same as coclustering (Step 3) by inserting a third variable

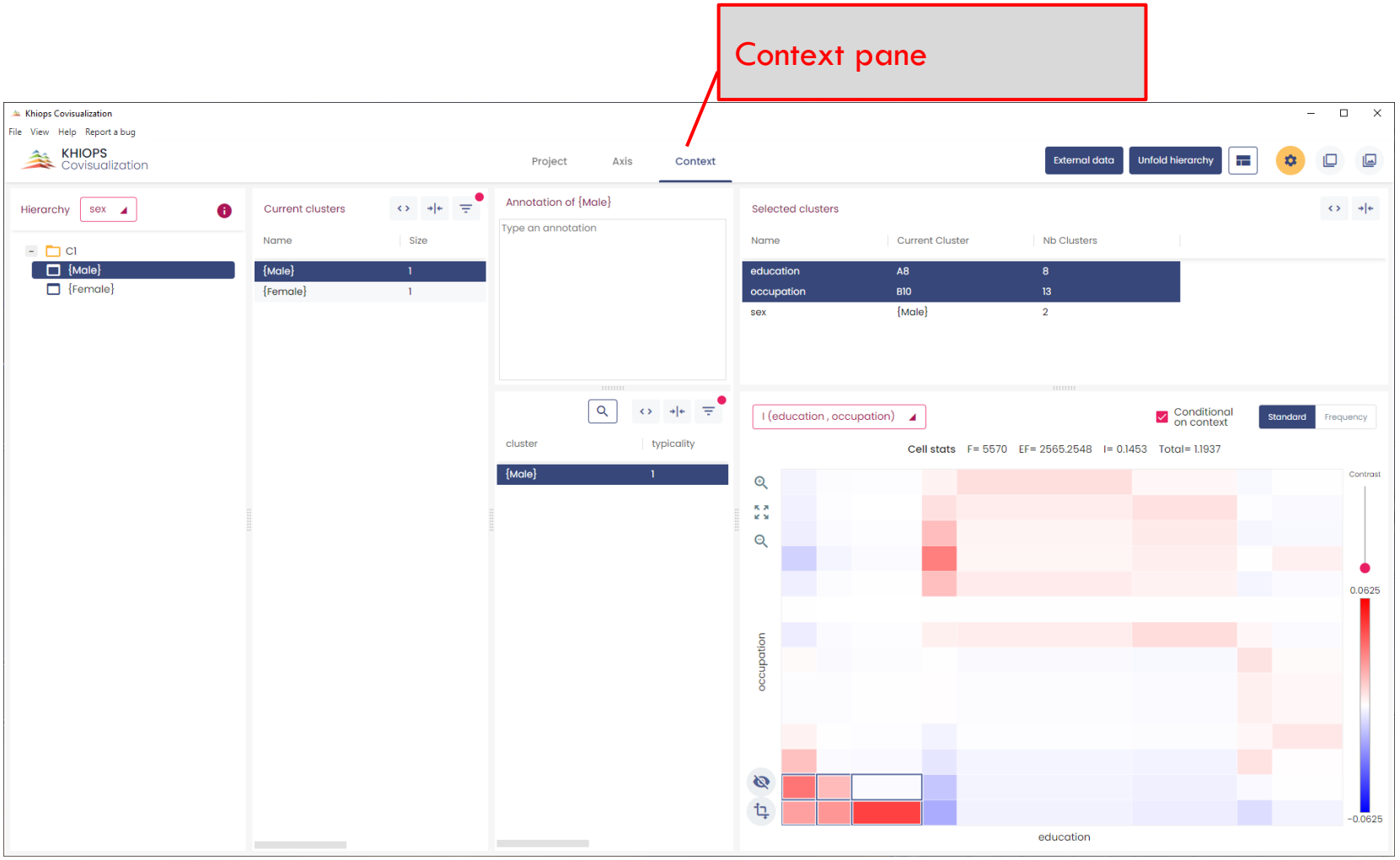


The third variable

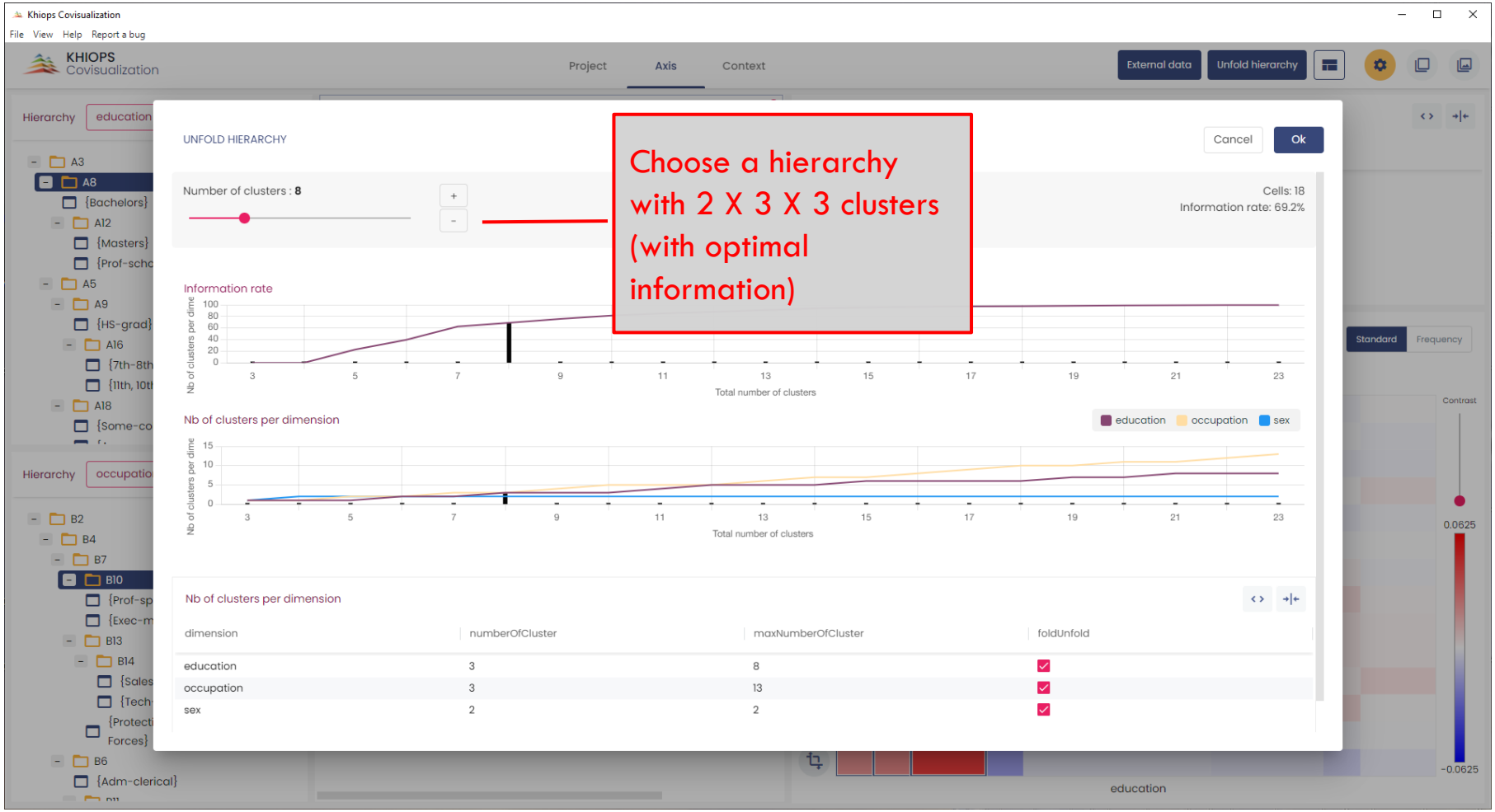
Exploring a triclustering



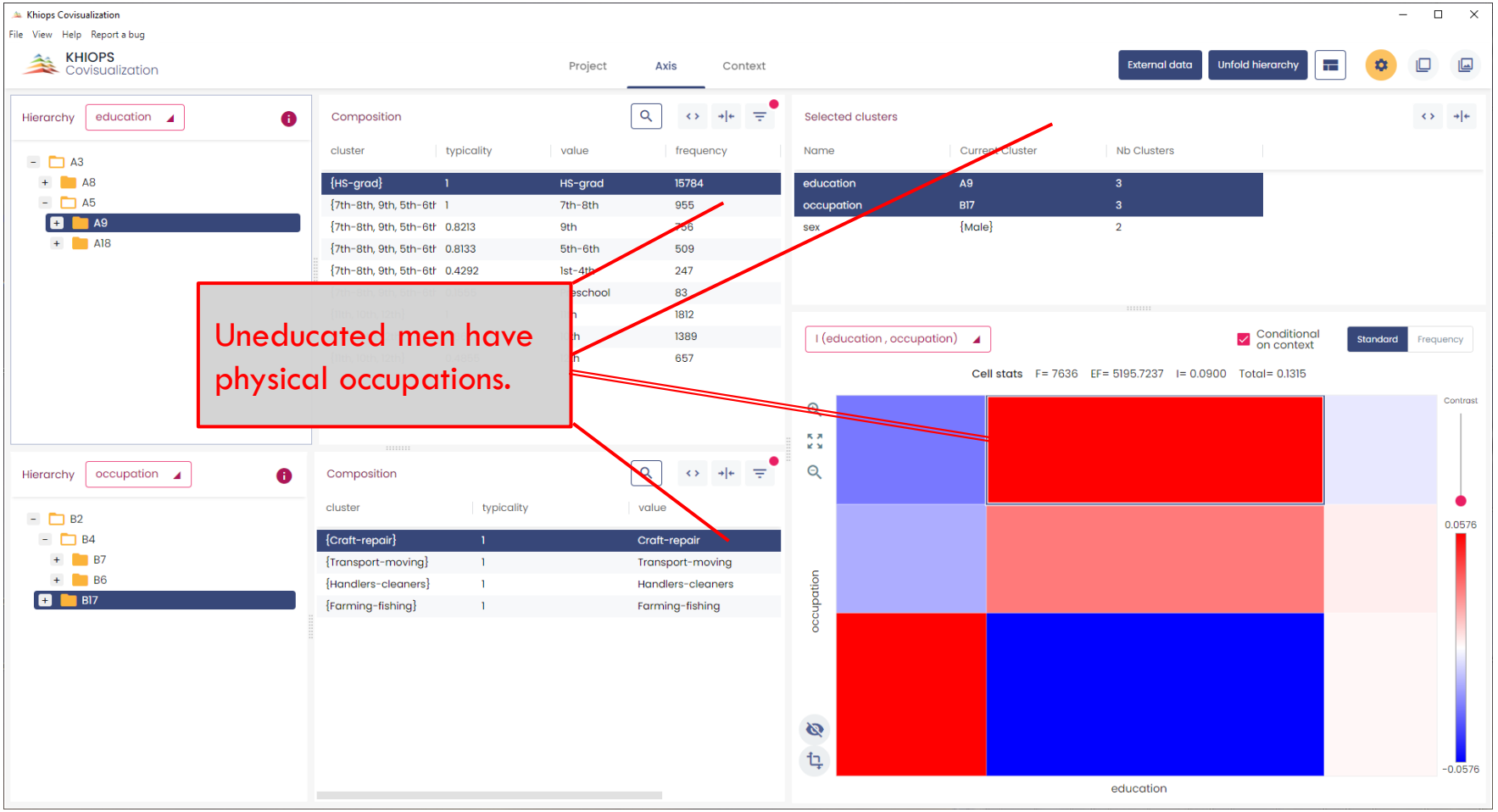
Exploring a triclustering



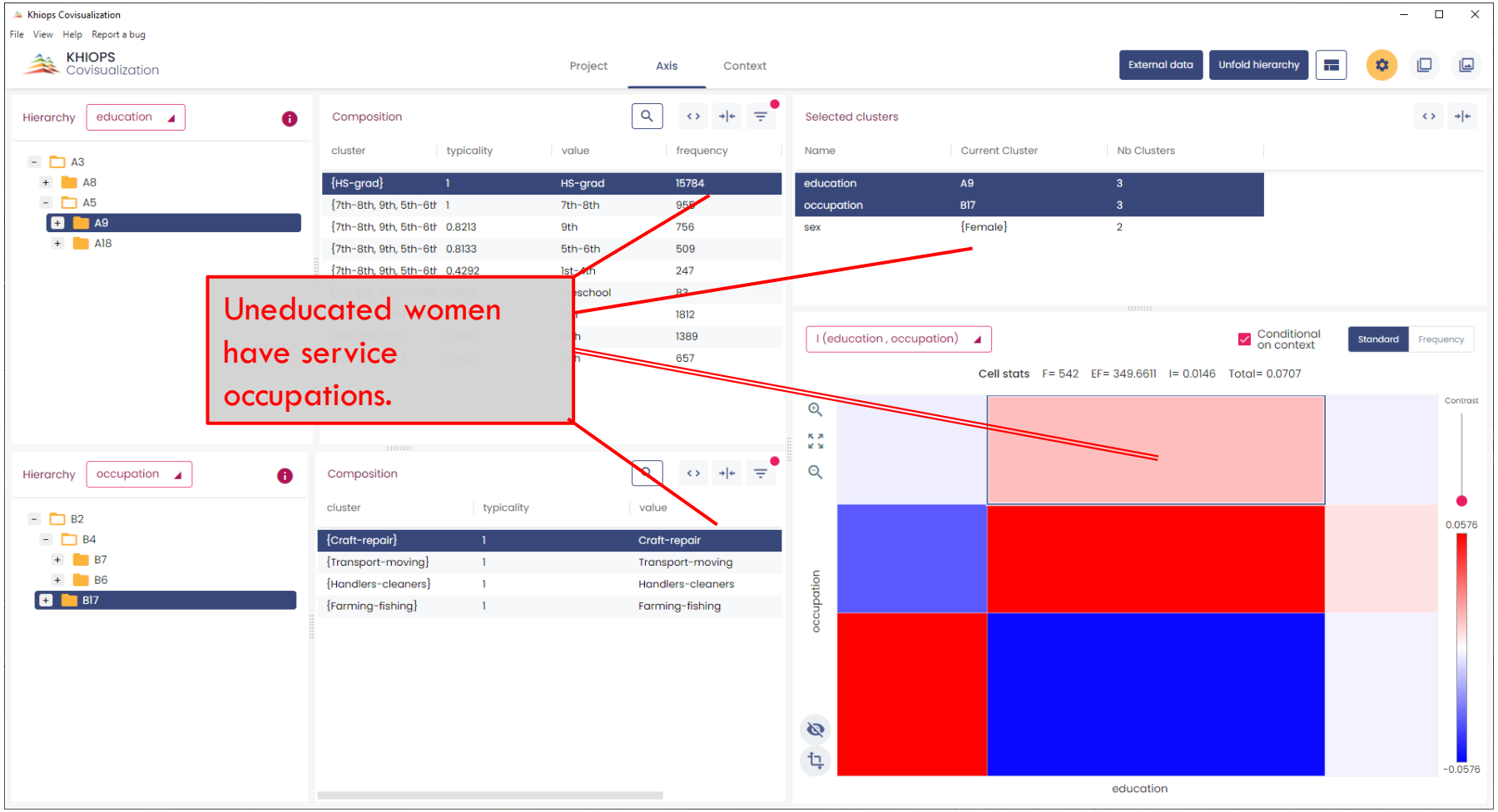
Exploring a triclustering



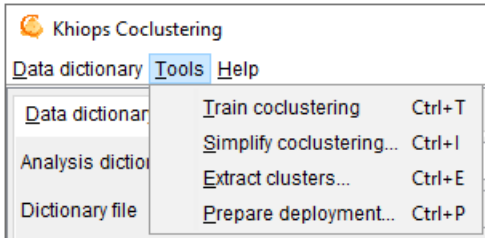
Exploring triclustering



Exploring a triclustering

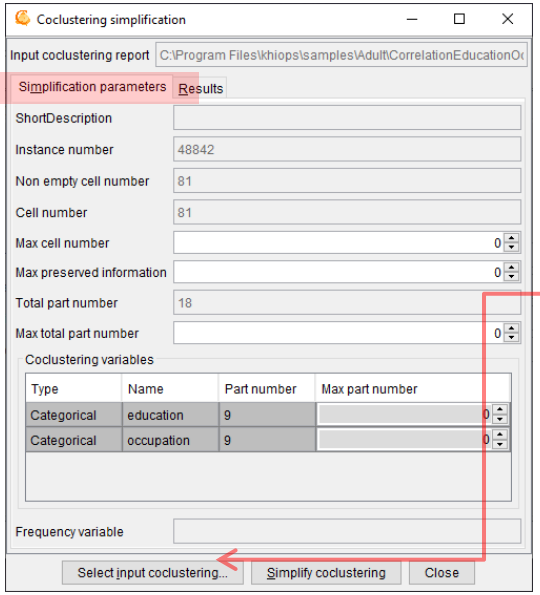


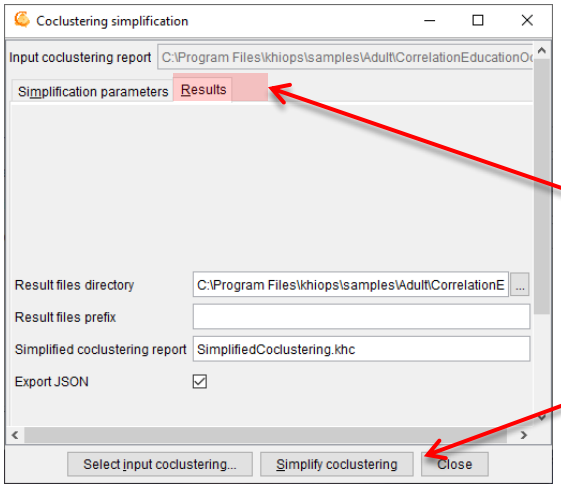
Exploiting a coclustering model



- Tools menu
  - Train coclustering
    - Input: dictionary and database file
    - Train a coclustering model
  - Simplify coclustering
    - Input: coclustering model
    - Build a simplify coclustering model given user constraints
  - Extract clusters
    - Input: coclustering model
    - Extract clusters in a text file for a given coclustering variable
  - Prepare deployment
    - Input: dictionary and coclustering model
    - Enables the deployment of a coclustering model on new data by the means of a Khiops deployment dictionary
    - See multi-table section of the tutorial

Simplifying a coclustering model

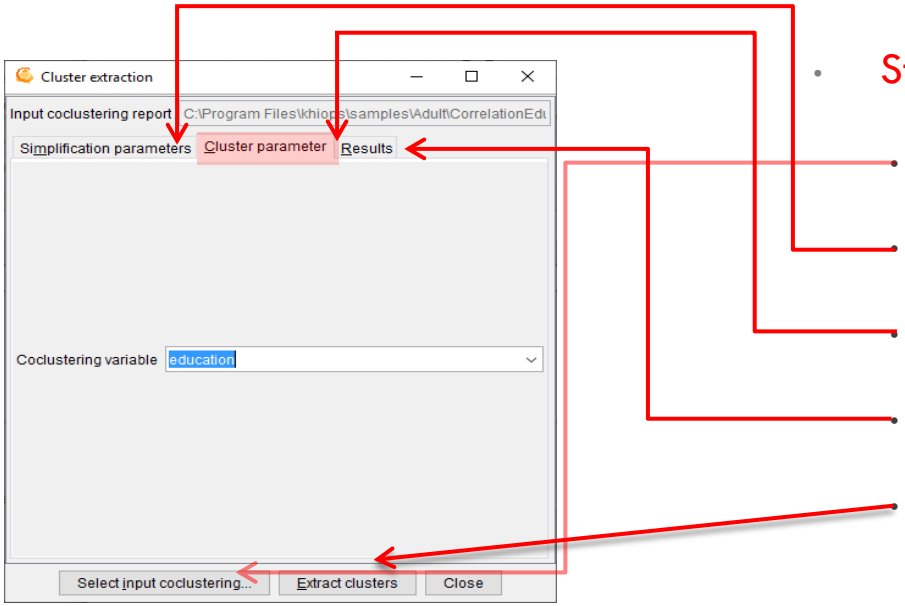




- Steps for coclustering model simplification
  - 1- Select input coclustering (.khc)
  - 2- Specify user simplification constraints
    - Max cell number :
      - max number of cells to keep in the simplified coclustering
    - Max preserved information
      - max percentage of information to keep in the simplified coclustering
    - Max total part number
      - max for the sum of the part number per coclustering variable
    - Per coclustering variables (in the array)
      Max part number
      - max number of part to keep for this variable in the simplified coclustering

      (0 : no constraint)
  - 3- Select result files directory
  - 4- Click on « *Simplify coclustering* »

Extracting clusters in a text file



- Steps for cluster extraction
  - 1- Select input coclustering (.khc)
  - 2- Specify user simplification constraints
  - 3- Select coclustering variable containing the clusters
  - 4- Select result files directory
  - 5- Click on « *Extract clusters* »

Output cluster file

- Text file with header line and separator tabulation
- Columns:
  - **Cluster:** name of the cluster (group of values)
  - **Value:** name of the value contained in the cluster
  - **Frequency:** frequency of the value
  - **Typicality:** interest measure of the value within its cluster

Exercise K, L …

K : Simplify previously built adult coclustering model

Keep 50% of the information in the model

➡ Explore the simplified analysis results with Khiops covisualization

L : Extract clusters from variable education of adult coclustering model

➡ Inspect the cluster file with a text editor

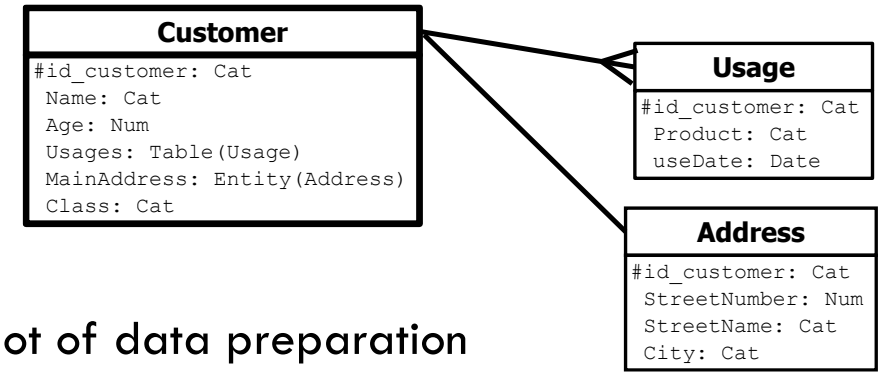
Multi-table functionalities

- Multi-table functionalities
  - Multi-table database
  - Automatic feature construction
  - Multi-table functionalities in Khiops and Khiops Coclustering

Why extending to multi-table?

- Why extending to multi-table?
  - Most data mining tools work on instances*variables flat tables
  - Real data often have a structure coming from databases
  - The input representation is richer using multi-table specification
  - Data mining methods may benefit from explicit richer domain description

- Real data is usually structured
  - Example
    - Marketing: Customer with shopping list
    - Web analytics: cookie with web log
    - Telecommunications: Customer with call detail records
    - Bioinformatics: DNA segments with ordered list of nucleotides
    - …



- Data mining with structured data requires a lot of data preparation
  - Constructing a representation in a flat table
    - Expert knowledge necessary to constructed new variables
    - Time expensive process to get a flat table usable for data analysis
  - This process is unreliable
    - Risk of missing informative variables
    - Risk of constructing and selecting irrelevant variables

Khiops multi-table

- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops
  - Multi-table dictionary
    - to describe star-schema input representation
  - Multi-table database
    - to store input data on multiple files
  - Feature construction language
    - to drive automatic feature construction
  - Sort functionality on large files
  - Powerful analytic functionalities
    - Automatic feature construction
    - Recoding of multi-table databases to get a flattened representation
    - Modeling and deployment at the multi-table level

- Impact on Khiops Coclustering
  - Deployment of coclustering models
    - For example, given a text*word coclustering model, assign new texts to their closest cluster

Khiops multi-table

- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops
  - Multi-table dictionary
    - to describe star-schema input representation
  - Multi-table database
    - to store input data on multiple files
  - Feature construction language
    - to drive automatic feature construction
  - Sort functionality on large files
  - Powerful analytic functionalities
    - Automatic feature construction
    - Recoding of multi-table databases to get a flattened representation
    - Modeling and deployment at the multi-table level

- All other Khiops functionalities are available similarly
  - Classification, regression, correlation analysis
  - Deployment, recoding, evaluation
  - …

Khiops multi-table

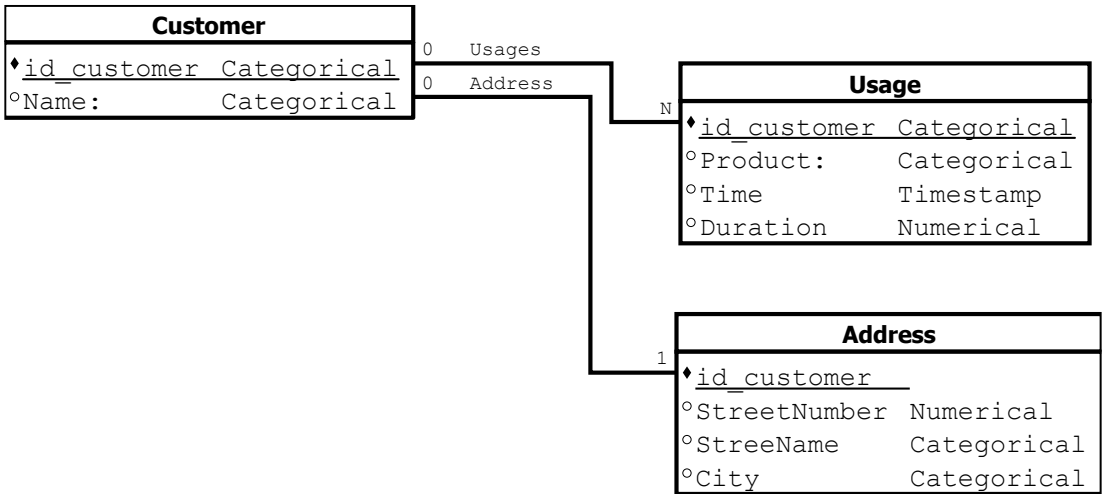
- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops
  - Multi-table dictionary
    - to describe star-schema input representation
  - Multi-table database
    - to store input data on multiple files
  - Feature construction language
    - to drive automatic feature construction
  - Sort functionality on large files
  - Powerful analytic functionalities
    - Automatic feature construction
    - Recoding of multi-table databases to get a flattened representation
    - Modeling and deployment at the multi-table level

- All other Khiops functionalities are available similarly
  - Classification, regression, correlation analysis
  - Deployment, recoding, evaluation
  - …

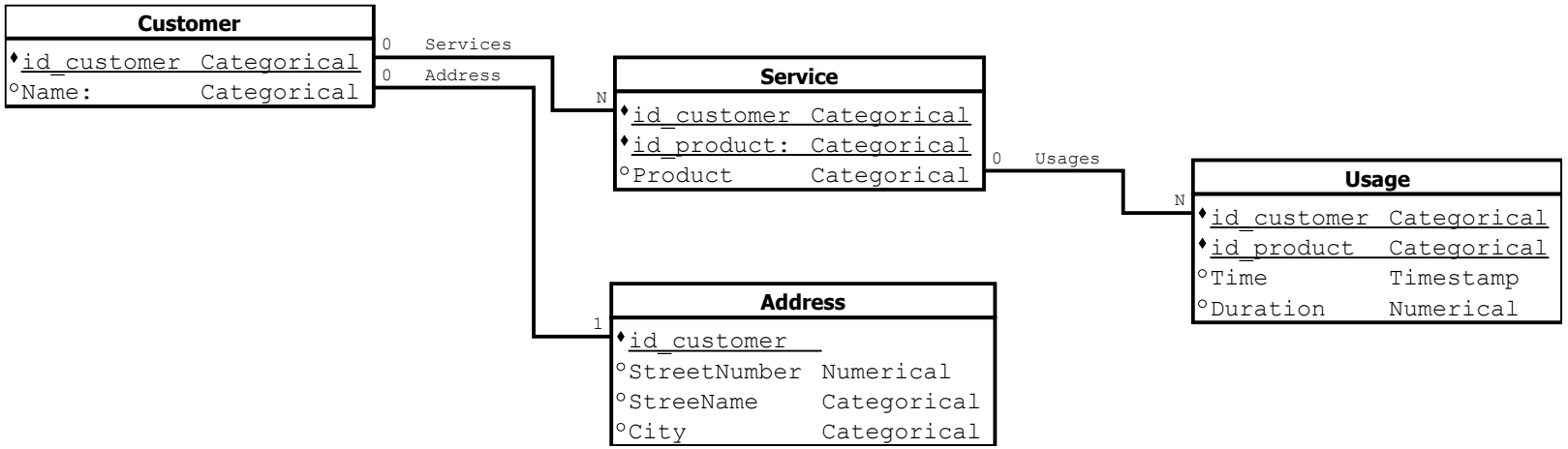
Star schema

- One root entity
  - secondary tables in 0-1 relationship: Entity
  - secondary tables in 0-n relationship: Table



Snowflake schema

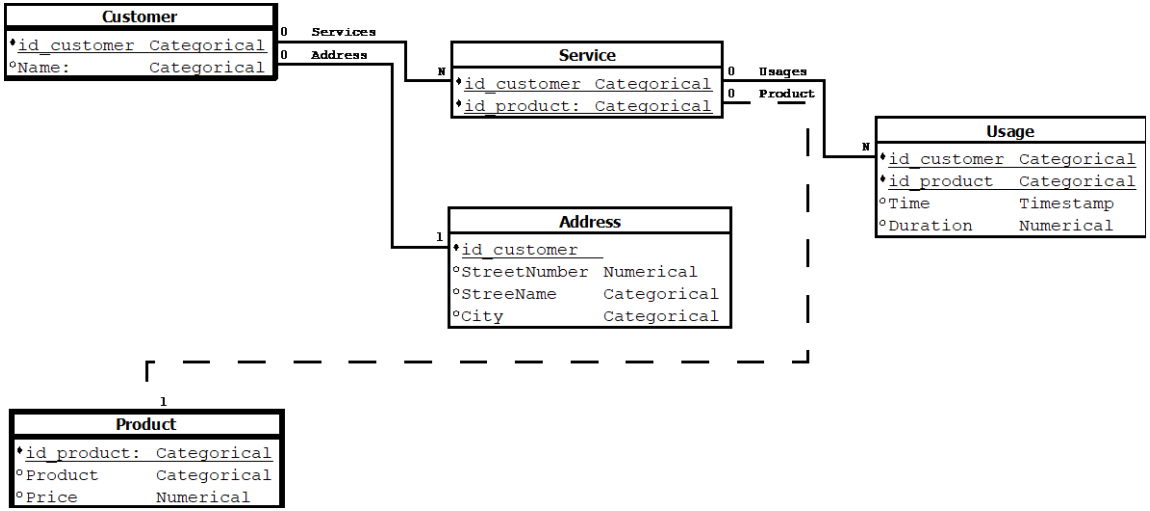
- One root entity
  - secondary tables in 0-1 relationship: Entity
  - secondary tables in 0-n relationship: Table
- Each table may have secondary tables



- Example in samples/Customer
  - detailed explanations in sample

External tables

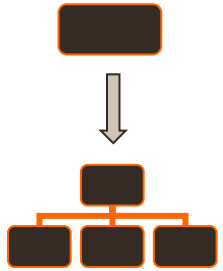
- One root entity
  - secondary tables in 0-1 relationship: Entity
  - secondary tables in 0-n relationship: Table
- Each table can have secondary tables
- External tables
  - to reuse common table shared by all analysis entities
  - can be referenced from any table, with specific keys



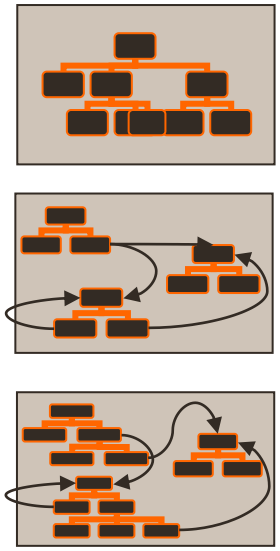
- Example in samples/CustomerExtended
  - detailed explanations in sample

Multi-table schemas: synthesis

- Khiops 8.0:
  - from mono-table to star schema
    - Automatic variable construction
    - a technological disruption



- Khiops 9.0:
  - extended data schema
    - Snowflake schema
    - External data
    - Multiple snowflake schema



Example of a multi-table database

French road accidents database

The **AccidentsSummary** is described using the following star schema:

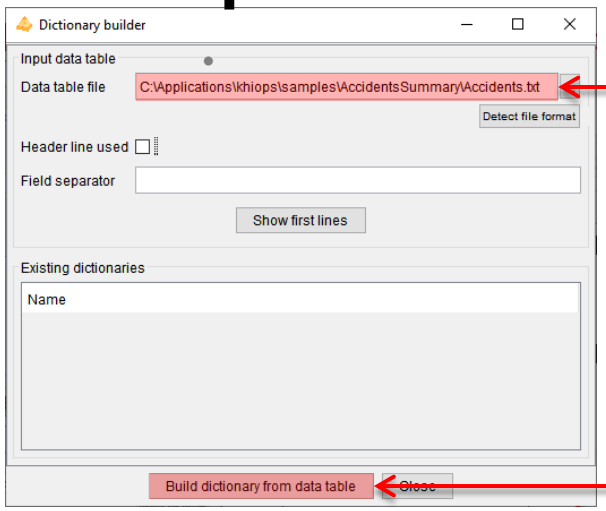
```
Accident
|
| -- 1:n -- Vehicle
```

Each accident has associated one or more vehicles. In the Khiops dictionary Accident-Vehicle 1:n relationship is described with the **Table** keyword. The key linking both tables is **AccidentId**.

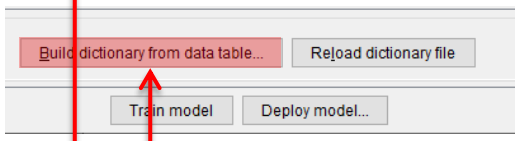
Objective: predict fatal traffic accidents (target variable: Gravity field of Accident table)

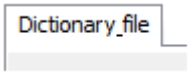
Build a multi-table dictionary

- **Step 1:** Build one dictionary per data table



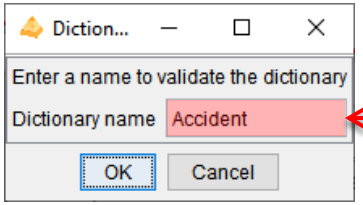
a : Build the first dictionary for the `Accidents.txt` table

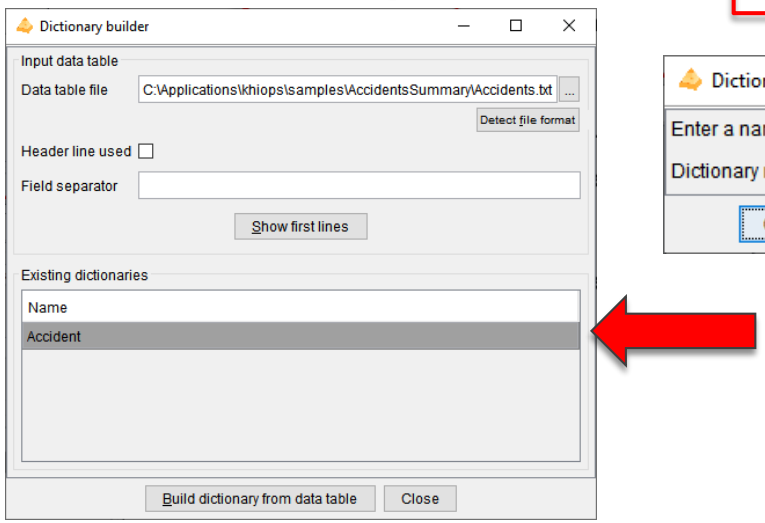




1. In pane
   - Click on button *Build dictionary…*

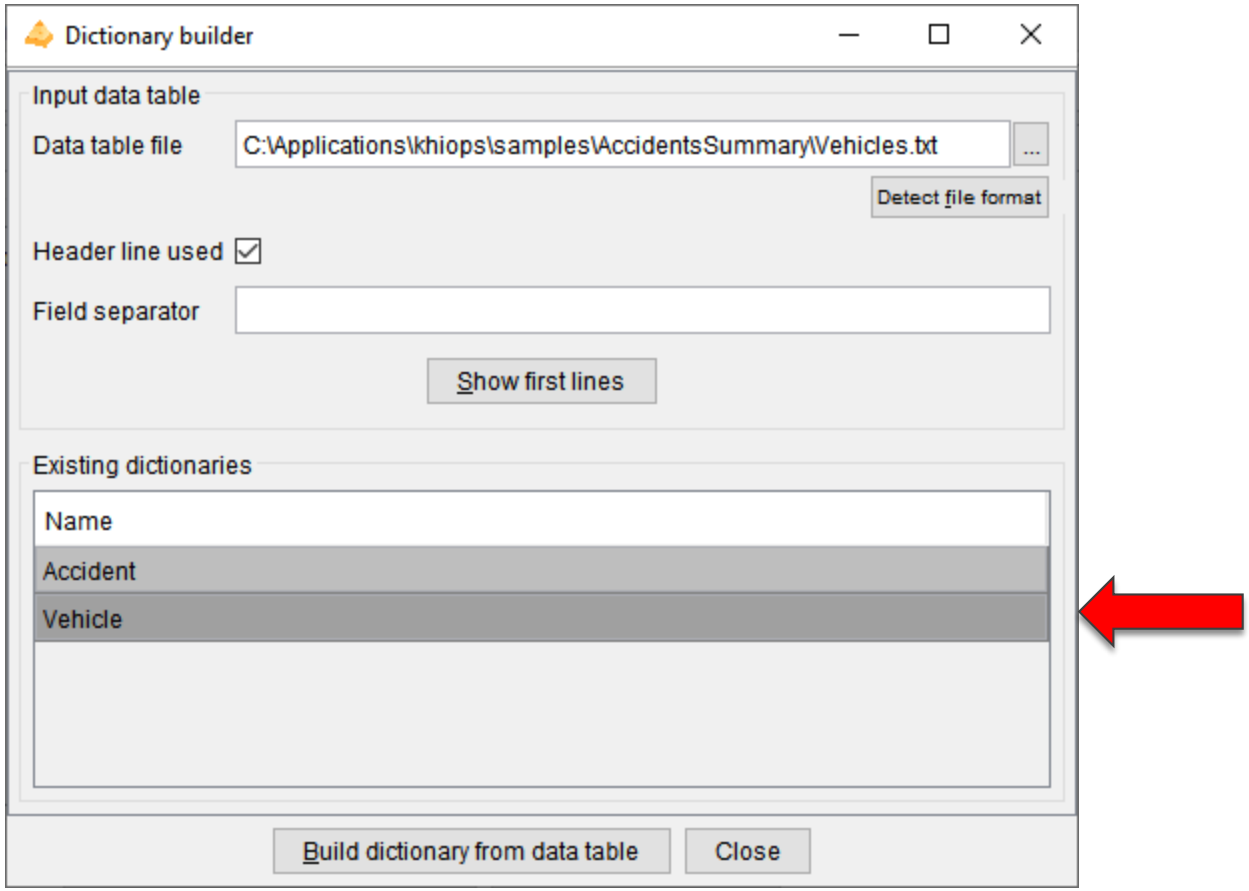
2. Build the first dictionary
   - Specify the data table file: `Accidents.txt`
   - Build the dictionary
   - Specify the dictionary name : `Accident`





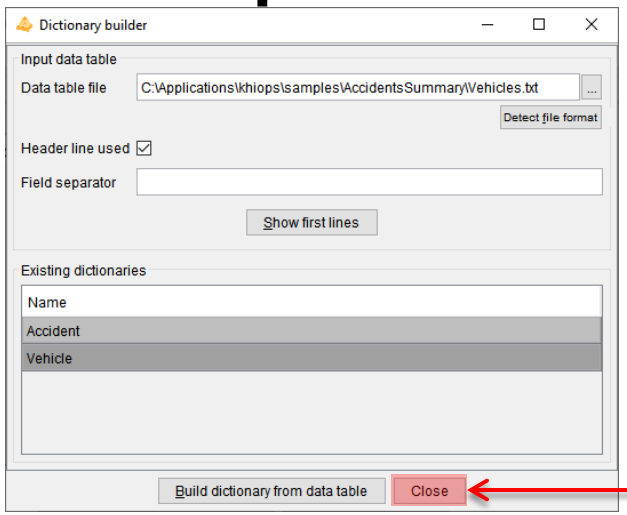
Build a multi-table dictionary

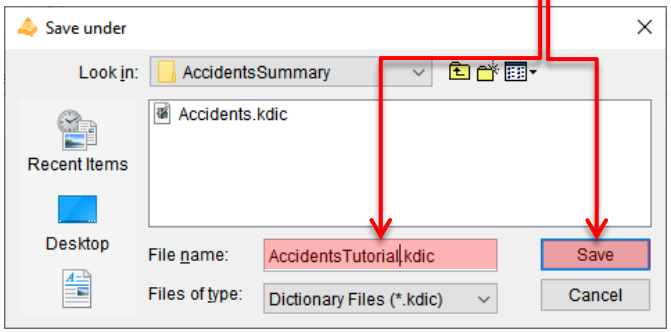
- **Step 1:** Build one dictionary per data table
  - b : Repeat for the `Vehicles.txt` table



Build a multi-table dictionary

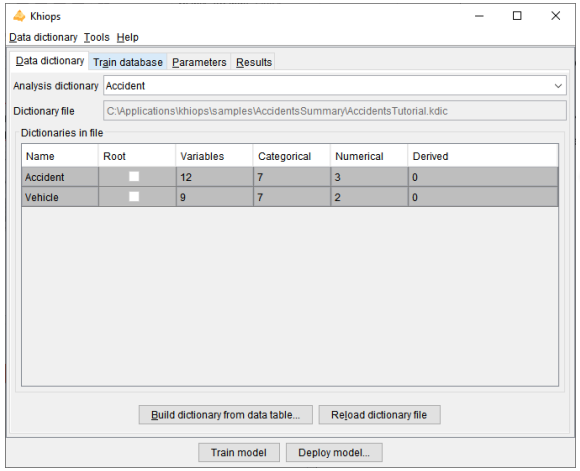
- **Step 1:** save the constructed dictionary into a `.kdic` file





4. Save in dictionary file
   - Close
   - Specify the dictionary file name:

     `AccidentsTutorial.kdic`
   - Save



Build a multi-table dictionary

- **Step 2:** Describe the table relationships in the `.kdic` file

5. Open the dictionary file with a text editor

5.1 Specify the root entity

5.2 Fix the types of the fields in **green**

5.3 Specify the key fields for each entity

Key fields must be Categorical and not derived

5.4 Specify the relation between the root entity and the secondary entity

Add a variable per relation to root dictionary

- *Table* for 0-n relationship
- *Entity* for 0-1 relationship

6. Save the dictionary file

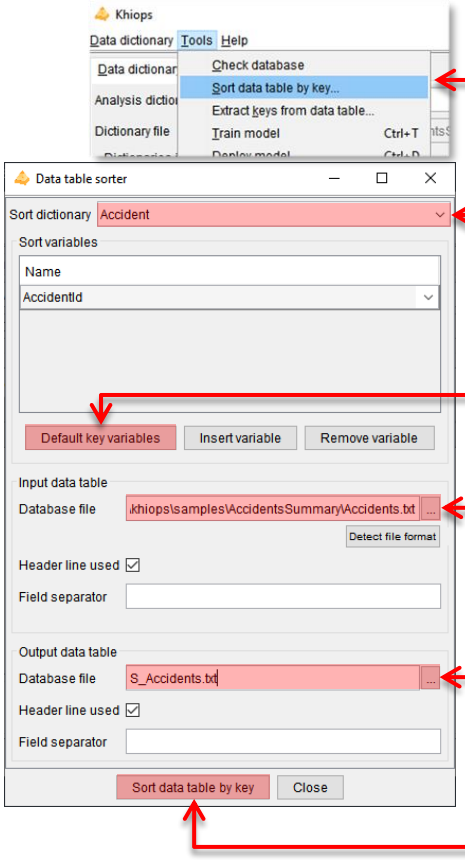
```
Root Dictionary Accident(AccidentId)
{
        Categorical AccidentId;
        Categorical Gravity;
        Date  Date;
        Time  Hour;
        Categorical Light;
        Categorical Department;
        Categorical Commune;
        Categorical InAgglomeration;
        Categorical IntersectionType;
        Categorical Weather;
        Categorical CollisionType;
        Categorical PostalAddress;
        Table(Vehicle) Vehicles;
};

Dictionary  Vehicle(AccidentId, VehicleId)
{
        Categorical AccidentId;
        Categorical VehicleId;
        Categorical Direction;
        Categorical Category;
        Numerical   PassengerNumber;
        Categorical FixedObstacle;
        Categorical MobileObstacle;
        Categorical ImpactPoint;
        Categorical Maneuver;
};
```

Sort data table files (if necessary)

For multi-table analyses data table files must be sorted by their keys

- Sorting is done only once before any Khiops analysis
  - Note: Records of the root table **must** be unique by key
- It is necessary for efficiency, specially when treating large databases
  - Records of the root and secondary tables are read synchronously from their data table files



1. Open the multi-table dictionary file
   This allows to obtain the definition of the tables to sort

2. Menu: *Tools* → *Sort data table by key*
   In the *Data table sorter* window, for each data table to sort

2.1 Specify the sort dictionary

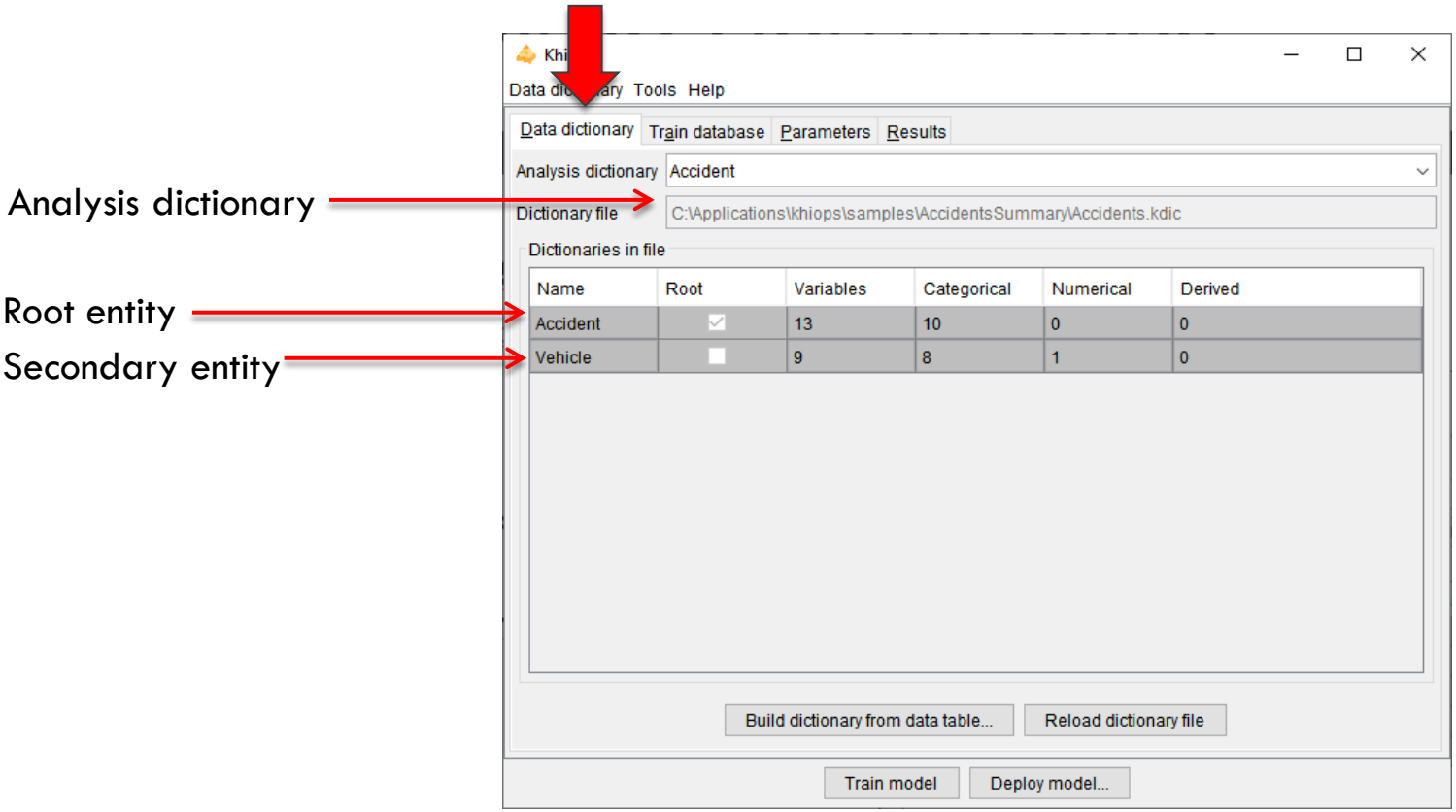
2.2 Specify the sort variables
   Default key variables to use the keys defined in dictionary

2.3 Specify the input and output data table files

2.4 Sort
- The output file is sorted by key
- All native variables are kept (used or not in the dictionary)
- Derived variables are ignored

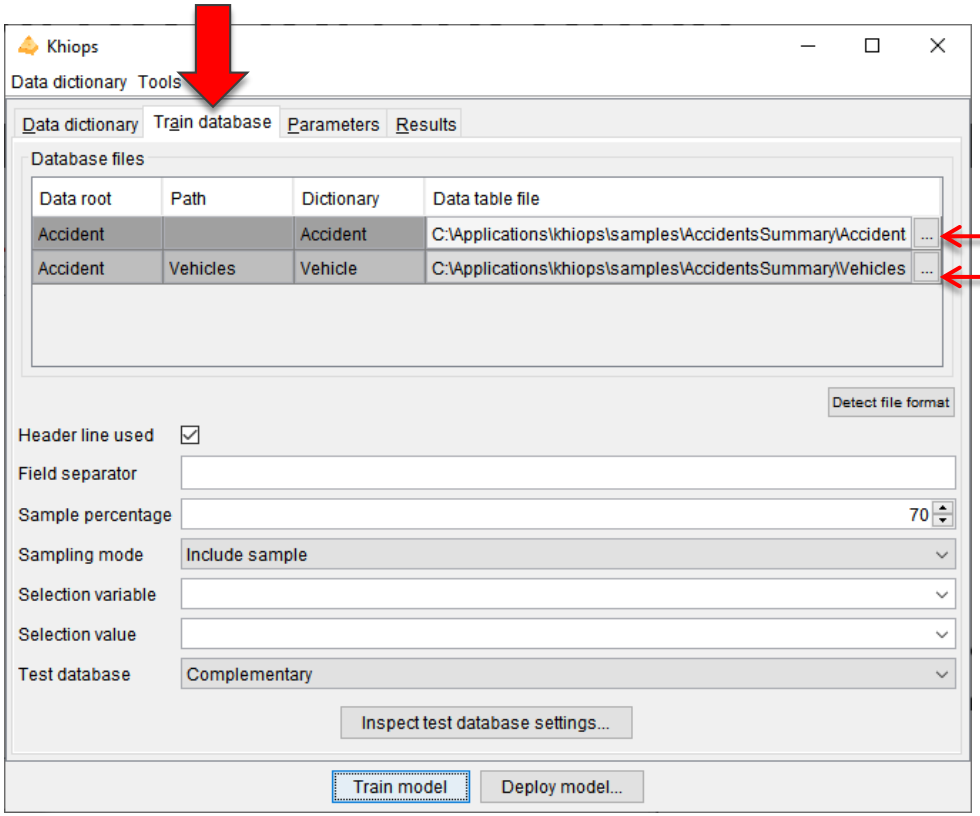
Supervised classification

- **Step 1, bis :** Open the *Accidents.kdic* dictionary file



Supervised classification

- **Step 2 :** Specify train and test databases
  - Specify the root and secondary data table files

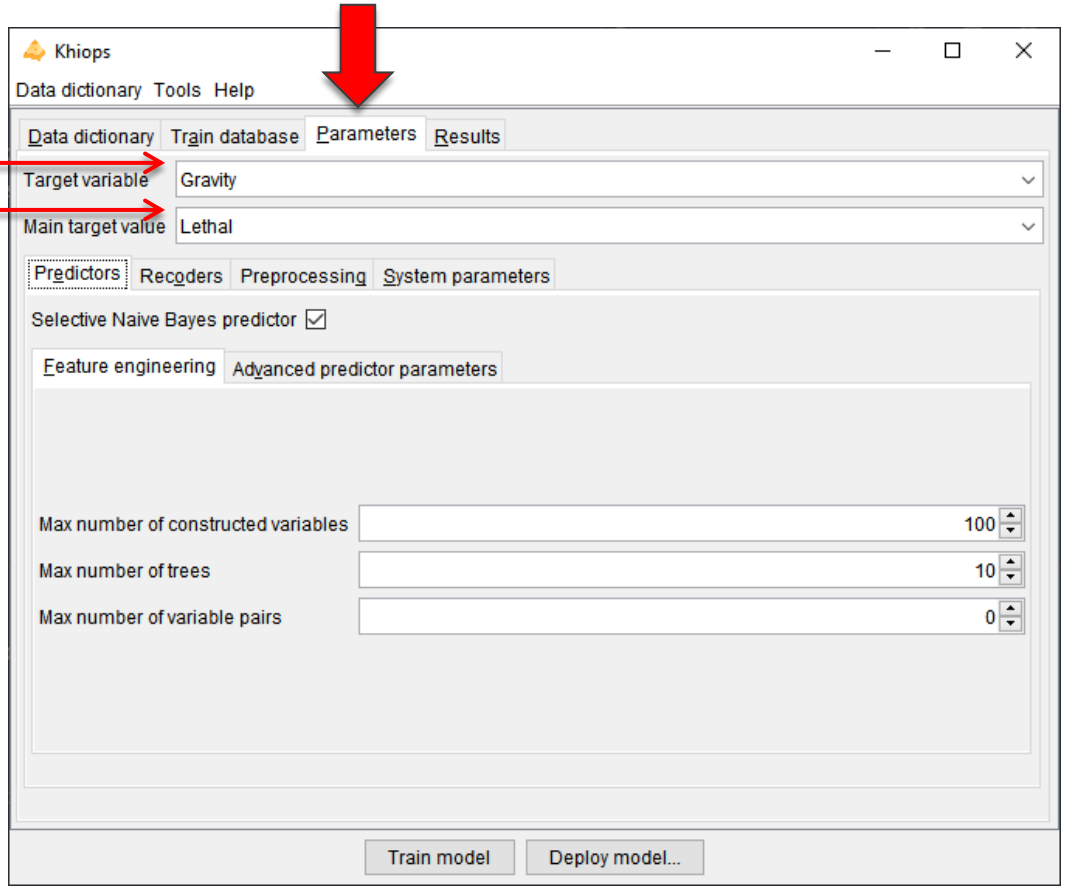


Supervised classification

- **Step 3 :** Parameters

Target variable

Main target value

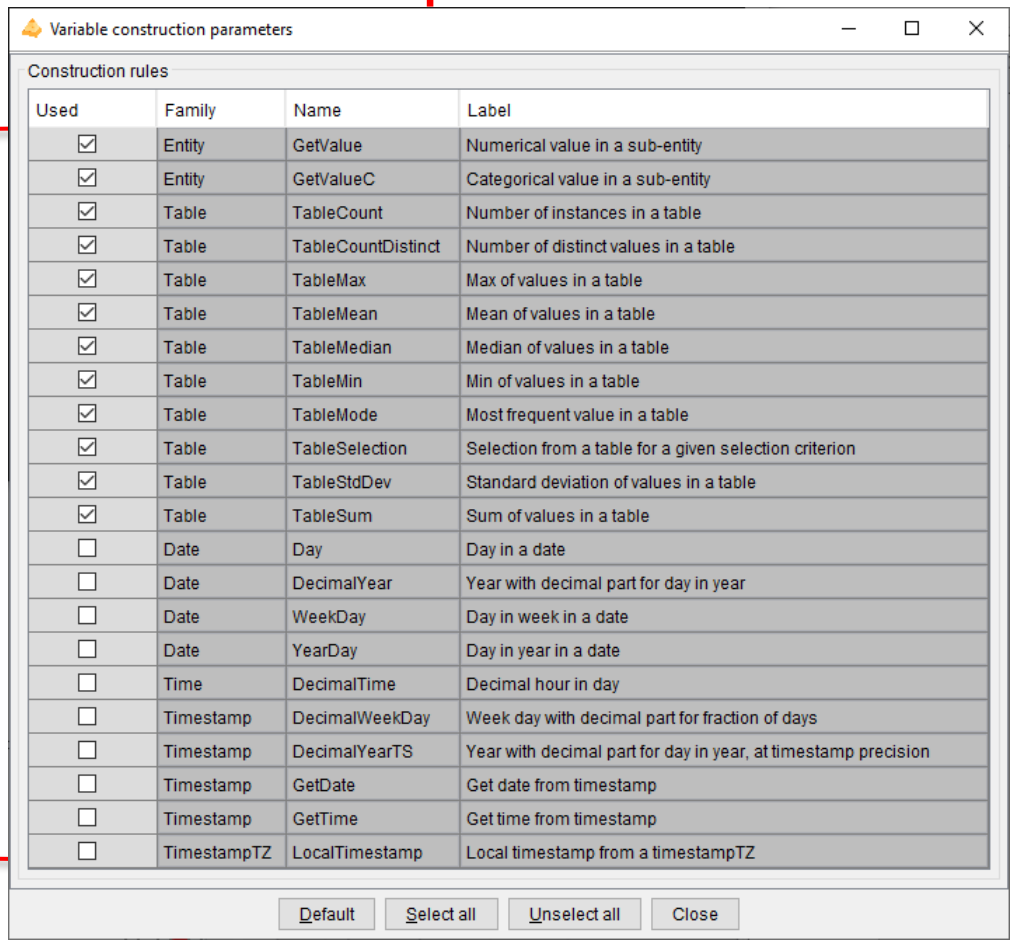


Supervised classification

- **Step 4 :** Variable construction parameters

Optional

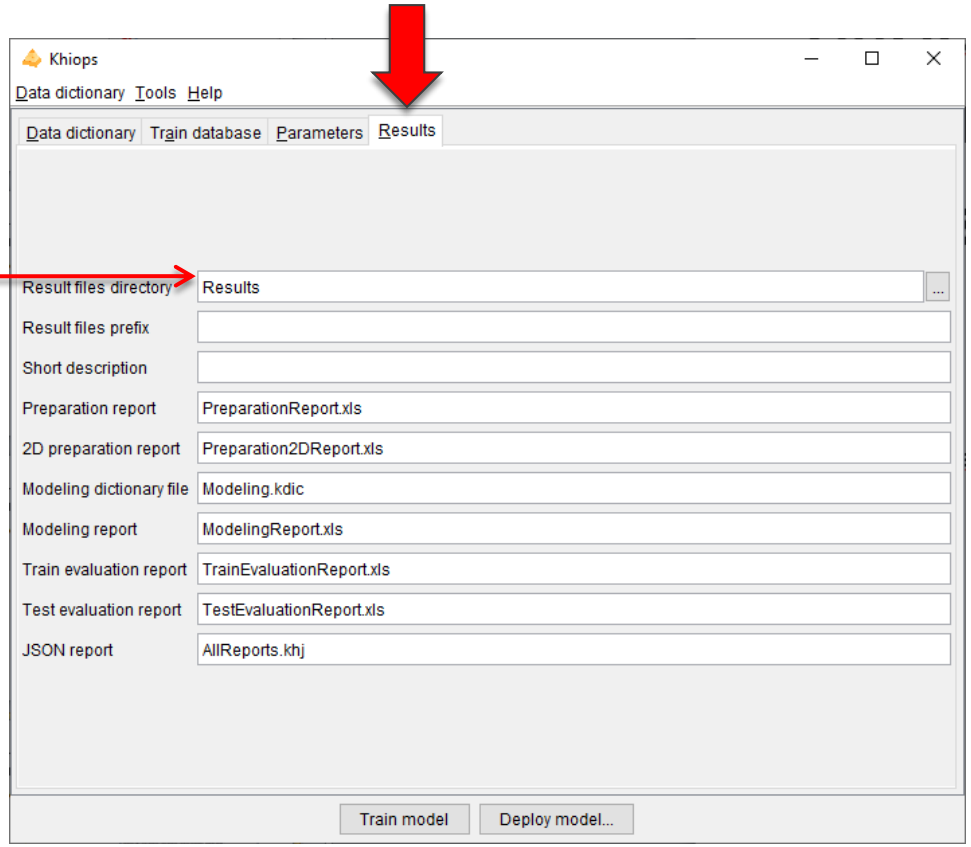
Choice of construction rules



Supervised classification

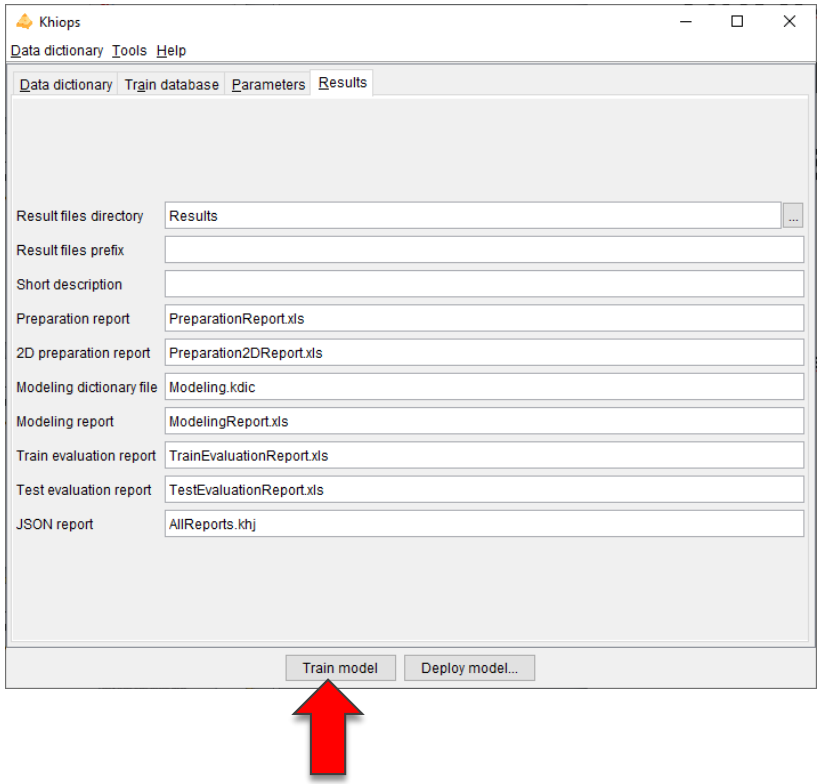
- **Step 5 :** Analysis results

Results files directory

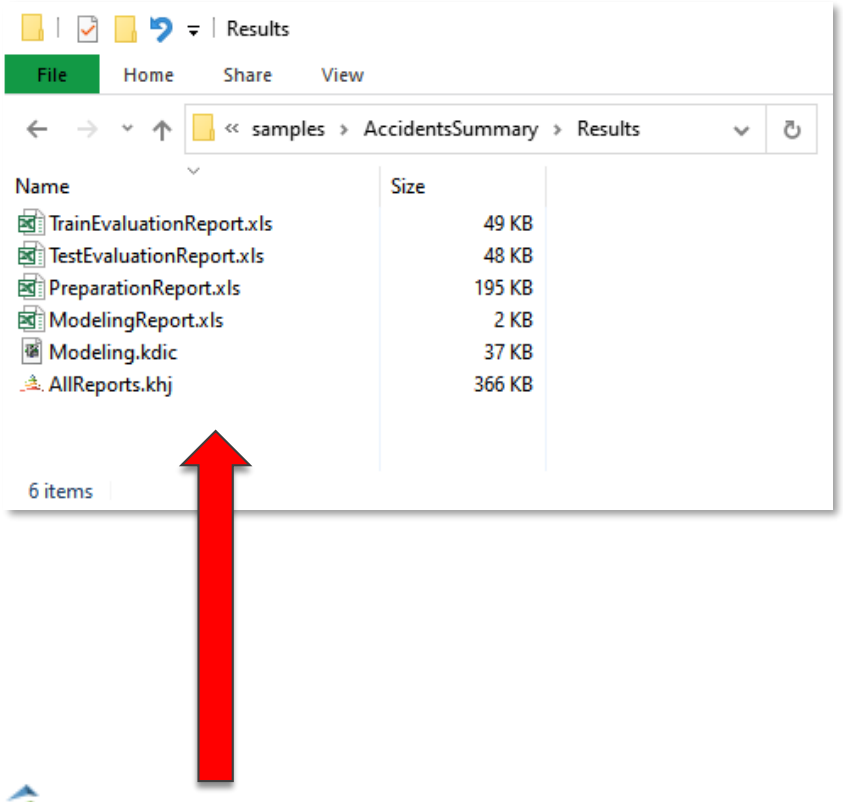


Supervised classification

- **Step 6 :** Start the analysis

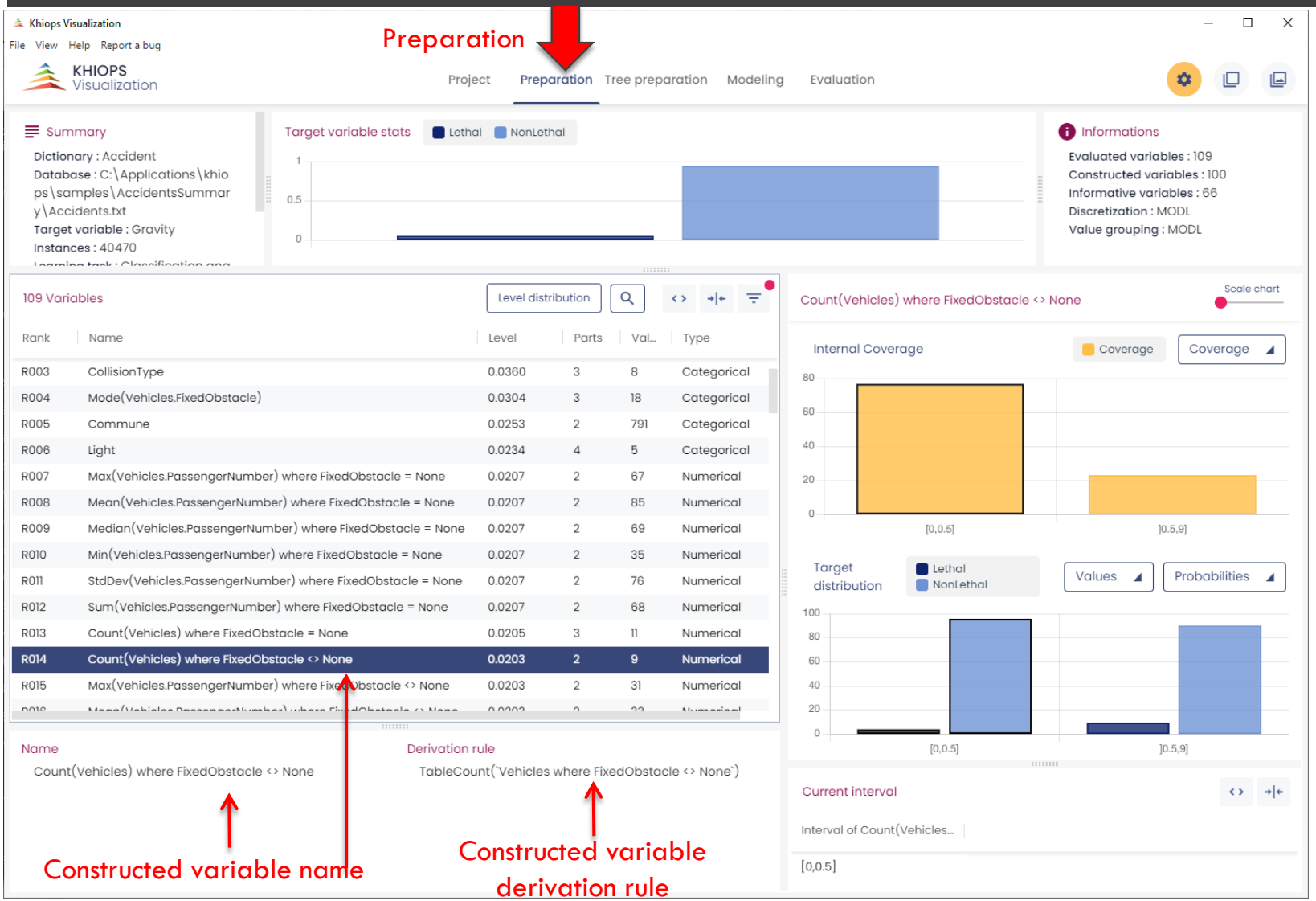


1 - Train model



2 - Inspect the results using Khiops Visualization
(double-click on *.khj* file)

Exploratory of classification results using Khiops Visualization

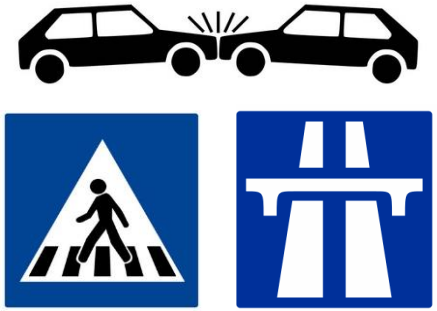


Example of a complex multi-table database

French road accidents database (full version)

This the full version of the **`AccidentsSummary`** dataset.
It is described using the following snowflake schema:

```
Accident
|
| -- 1:n -- Vehicle
|              |
|              |-- 1:n -- User
|
| -- 1:1 -- Place
```



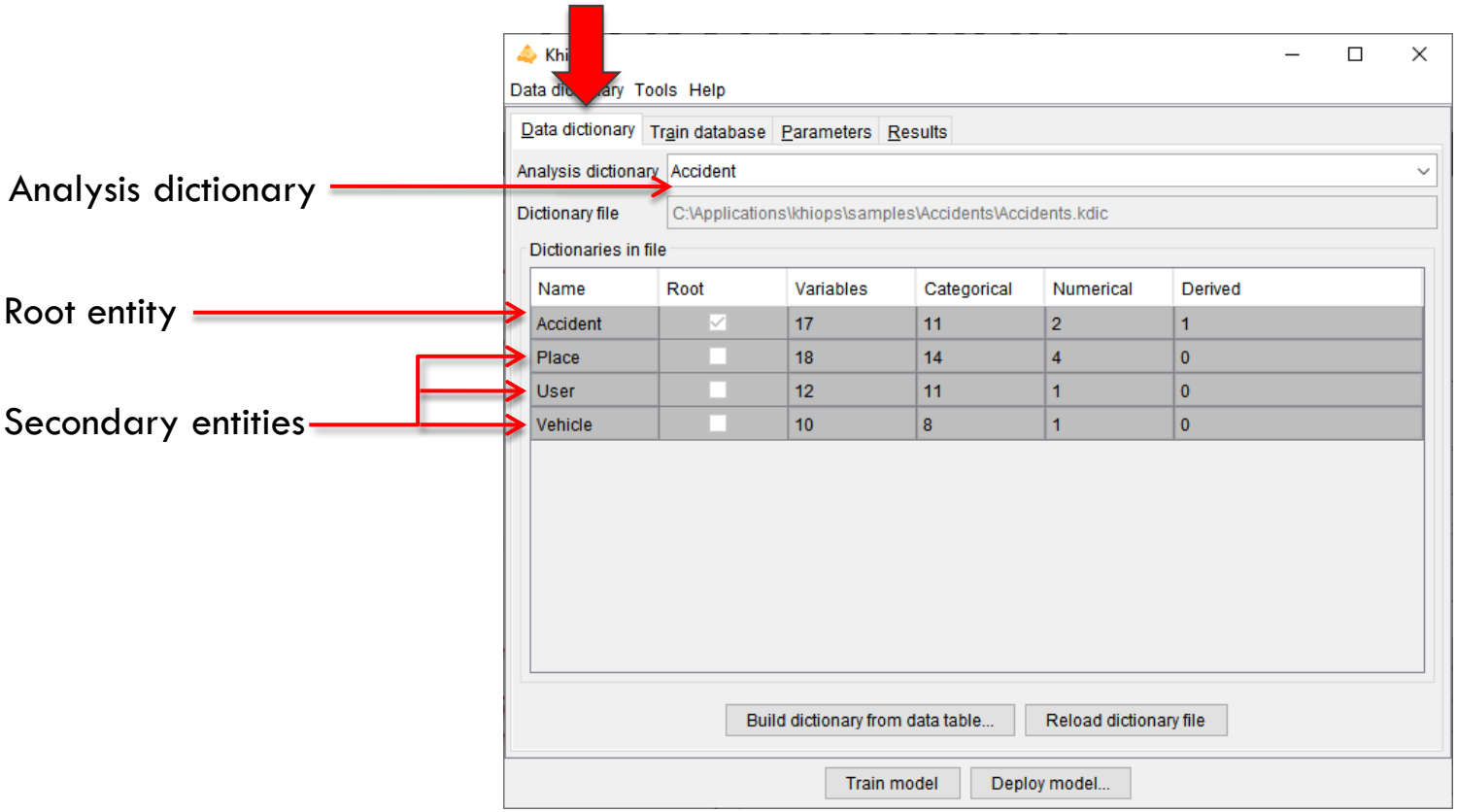
Each accident has associated one or more vehicles and one unique place. The vehicles involved in an accident have in turn associated one or more road users (passengers and pedestrians).

In the Khiops dictionary the Accident-Place relationship (1:1) is described with the `Entity` keyword, whereas the Accident-Vehicle and Vehicle-User relationships (1:n) with the `Table` keyword.

Objective: predict fatal traffic accidents (target variable: `Gravity` field of `Accident` table)

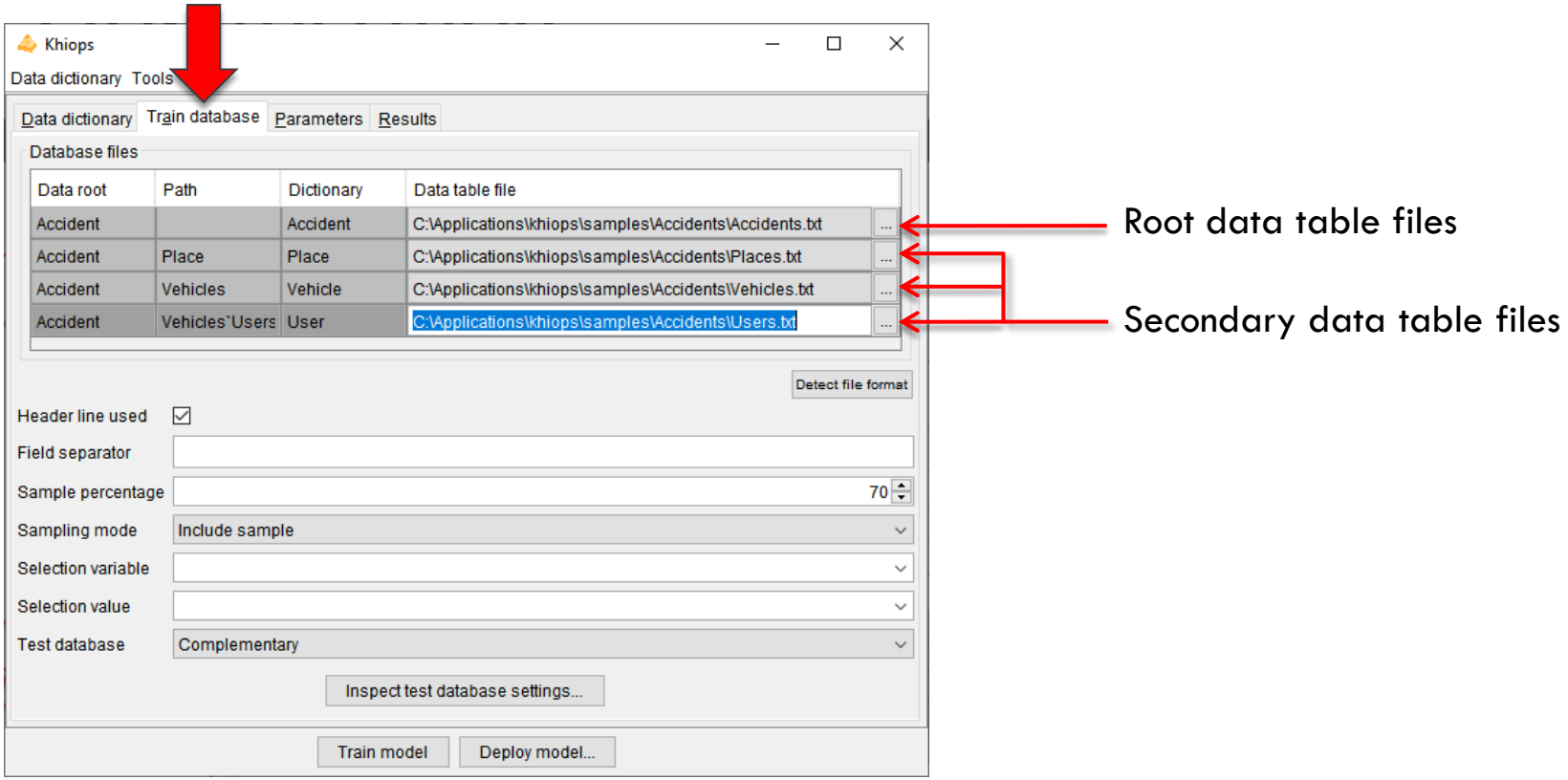
Supervised classification

- **Step 1 :** Open the *Accidents.kdic* dictionary file



Supervised classification

- **Step 2 :** Specify train and test databases
  - Root and other data table files have to be specified

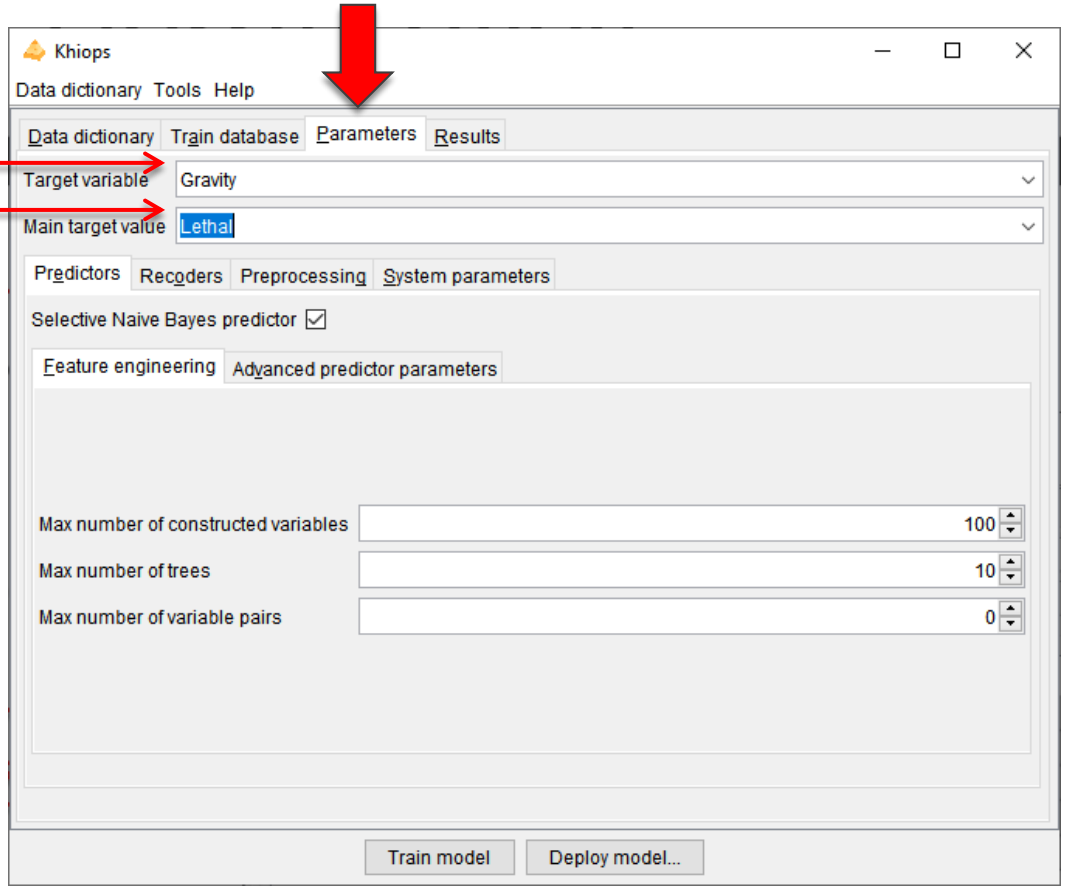


Supervised classification

- **Step 3 :** Parameters

Target variable

Main target value

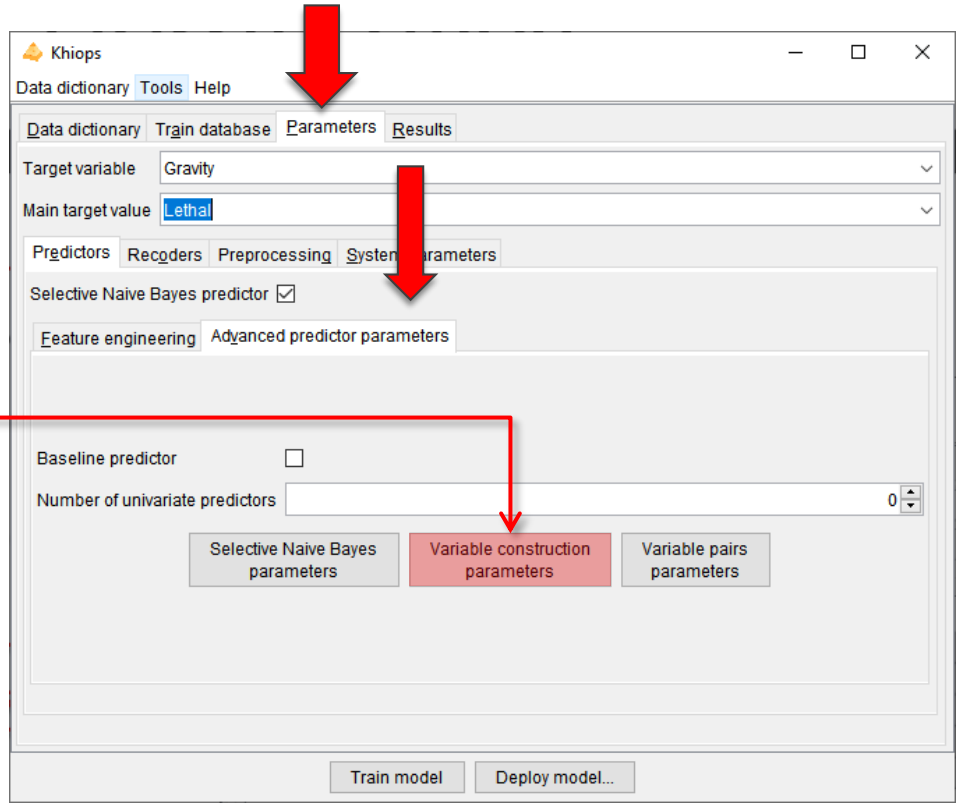


Supervised classification

- **Step 4 :** Variable construction parameters

Optional

Choice of construction rules

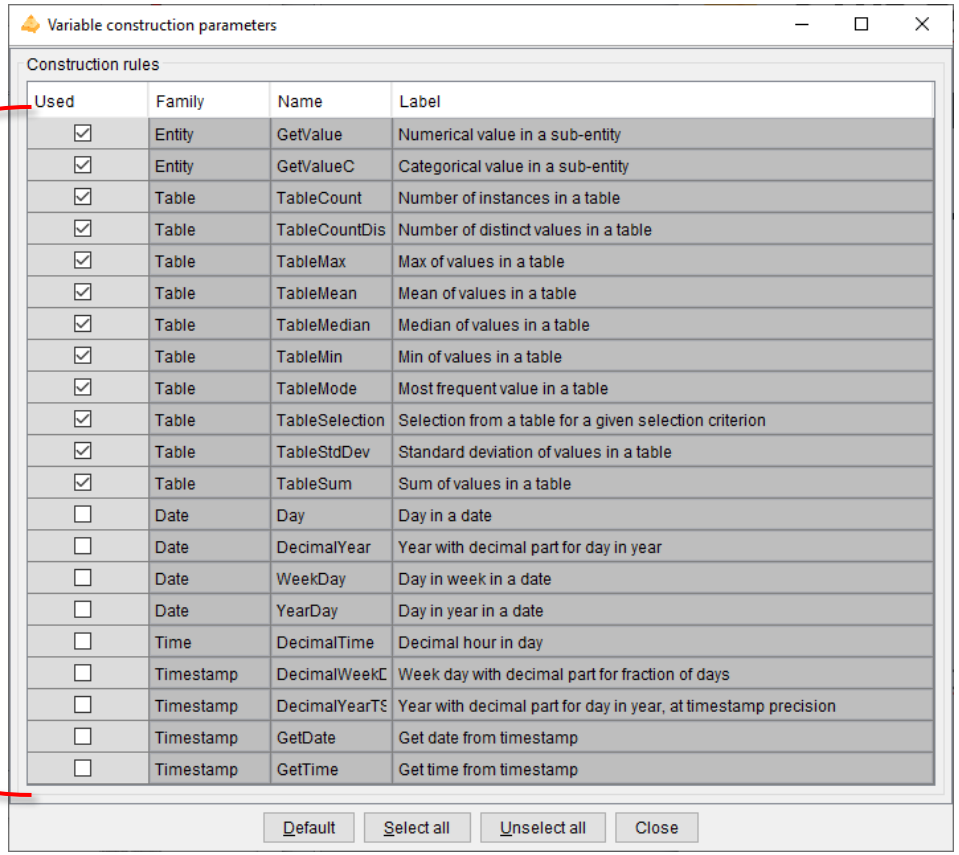


Supervised classification

- **Step 4 :** Variable construction parameters

Optional

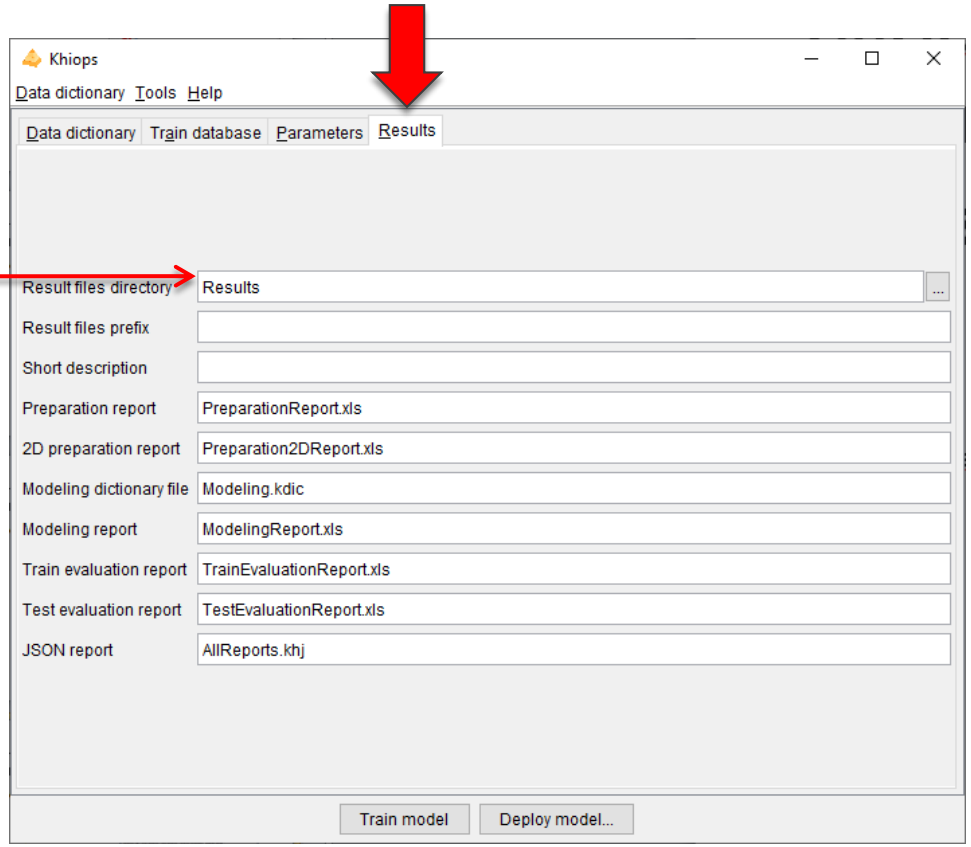
Choice of construction rules



Supervised classification

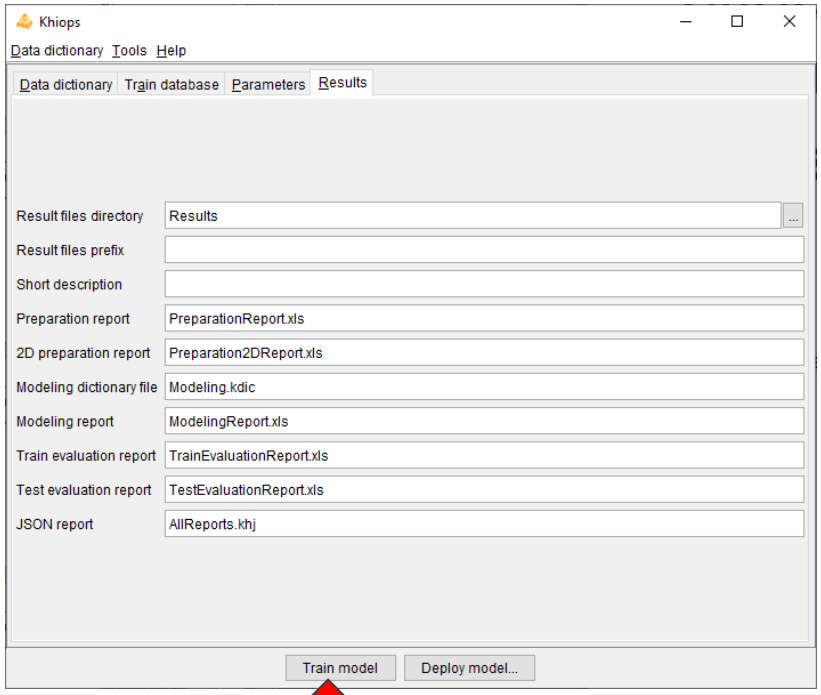
- **Step 5 :** Analysis results

Results files directory

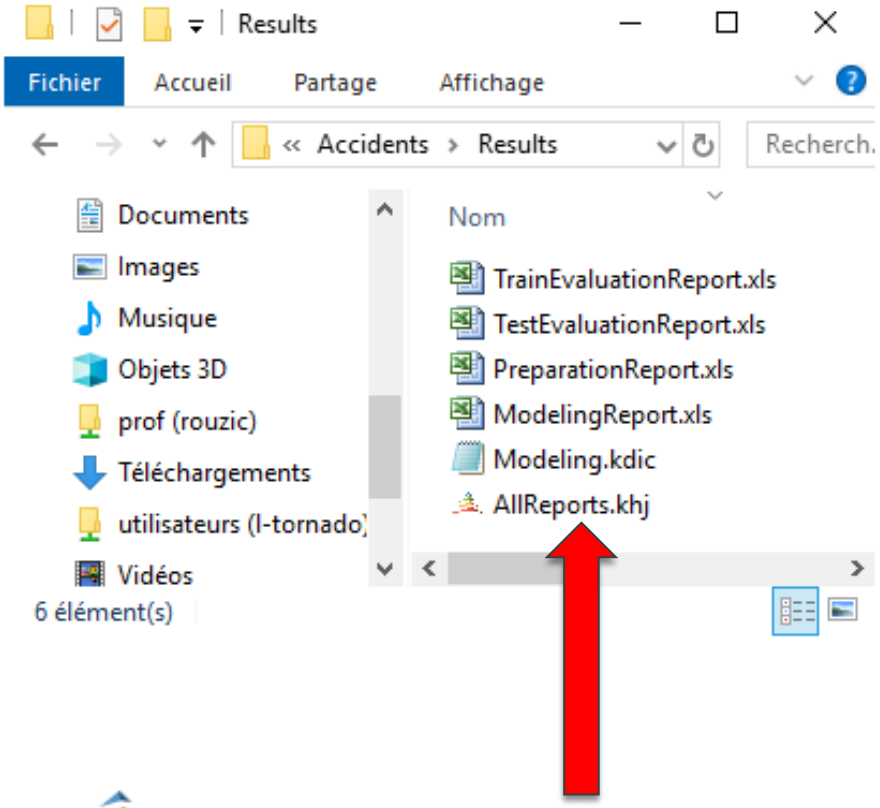


Supervised classification

- **Step 6 :** Start the analysis

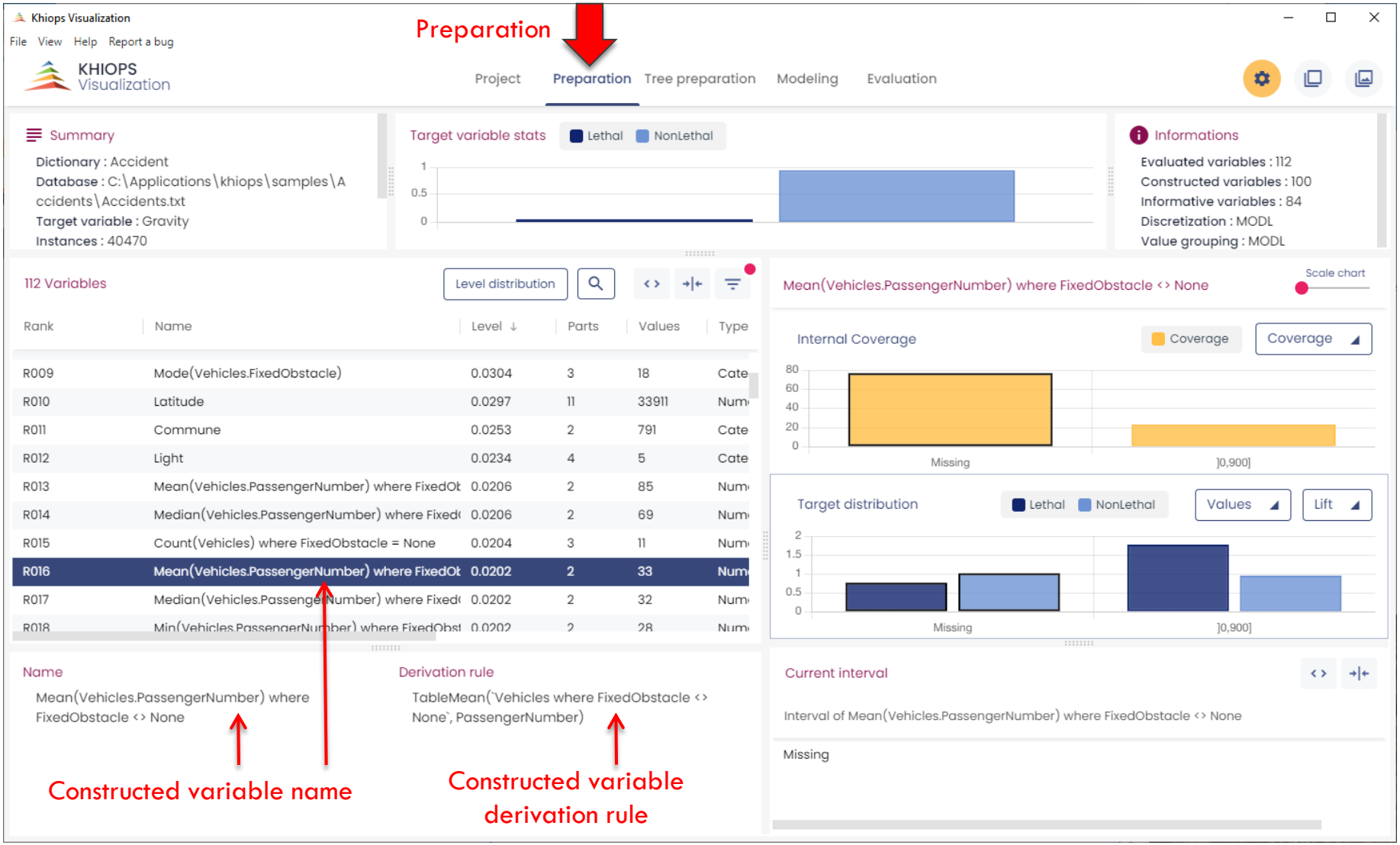


1 - Train model



2 - Inspect the results using Khiops Visualization
(double-click on *.khj* file)

Exploratory of classification results using Khiops Visualization



Khiops multi-table

- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops Coclustering
  - Deployment of coclustering models
    - Given a text*word coclustering model, assign new texts (with their words) to their closest cluster
    - Given a cookie*page coclustering model, assign new cookies (with their pages) to their closest cluster
    - Given a curve*X*Y triclustering model, assign new curves (with their X*Y points) to their closest cluster

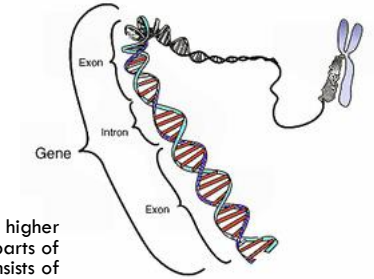
- In this tutorial

  - Build a triclustering model on the SpliceJunctionDNA data table
    - Clusters of sequence samples
    - Intervals of positions in the sequences
    - Clusters of DNA chars

  - Prepare a deployment model
    - Build a deployment dictionary

  - Deploy the model on the multi-table SpliceJunction database
    - Assign new DNA sequences to trained clusters of sequences

Splice junction multi-table database



- Molecular Biology (Splice-junction Gene Sequences)
  - Objective:
    - Recognition of boundaries between exons and introns in DNA sequences
    - Splice junctions are points on a DNA sequence at which `superfluous' DNA is removed during the process of protein creation in higher organisms. The problem posed in this dataset is to recognize, given a sequence of DNA, the boundaries between exons (the parts of the DNA sequence retained after splicing) and introns (the parts of the DNA sequence that are spliced out). This problem consists of two subtasks: recognizing exon/intron boundaries (referred to as EI sites), and recognizing intron/exon boundaries (IE sites). (In the biological community, IE borders are referred to as ``acceptors'' while EI borders are referred to as ``donors''.)

- Database dictionary
  - Root entity: splice junction
    - SampleId
    - Class (EI, IE, NEG)
    - Sequence of DNA
  - Secondary entity: DNA
    - SampleId:
    - Pos: position in the sequence
    - Char (A, C, G, T)

- Database files
  - SpliceJunction.txt
  - SpliceJunctionDNA.txt

SpliceJunction.txt

SampleId	Class
AGMKPNRSB-NEG-1	N
AGMORS12A-NEG-181	N
AGMORS9A-NEG-481	N
AGMRSKPNI-NEG-1141	N
ATRINS-ACCEPTOR-1678	IE
ATRINS-ACCEPTOR-701	IE
ATRINS-DONOR-521	EI
ATRINS-DONOR-905	EI
...	

SpliceJunctionDNA.txt

SampleId	Pos	Char
AGMKPNRSB-NEG-1	1	C
AGMKPNRSB-NEG-1	2	A
...		
AGMKPNRSB-NEG-1	58	A
AGMKPNRSB-NEG-1	59	C
AGMKPNRSB-NEG-1	60	A
AGMORS12A-NEG-181	1	A
AGMORS12A-NEG-181	2	G
...		
AGMORS12A-NEG-181	59	G
AGMORS12A-NEG-181	60	G
AGMORS9A-NEG-481	1	T
AGMORS9A-NEG-481	2	G
AGMORS9A-NEG-481	3	G
...		

Exploratory analysis of DNA sequences:

. find clusters of similar DNA sequences

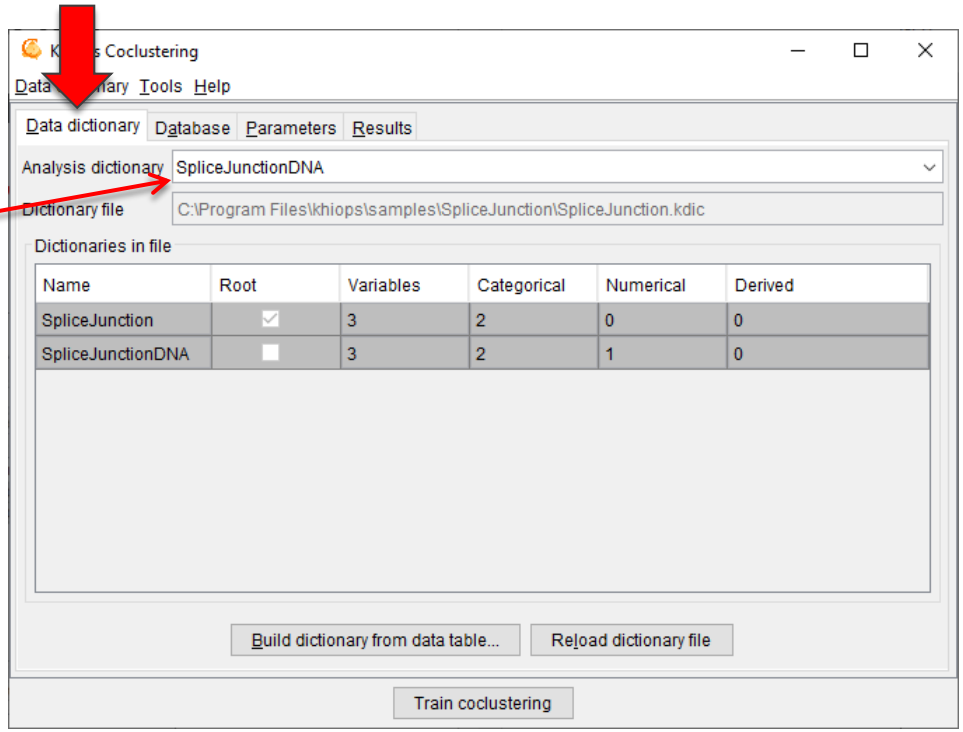
. using a triclustering SampleId x Pos x Char

Train a triclustering model

- **Step 1 :** Open an existing dictionary

  (ex: sample SpliceJunction.kdic)

Analysis dictionary
(secondary entity)

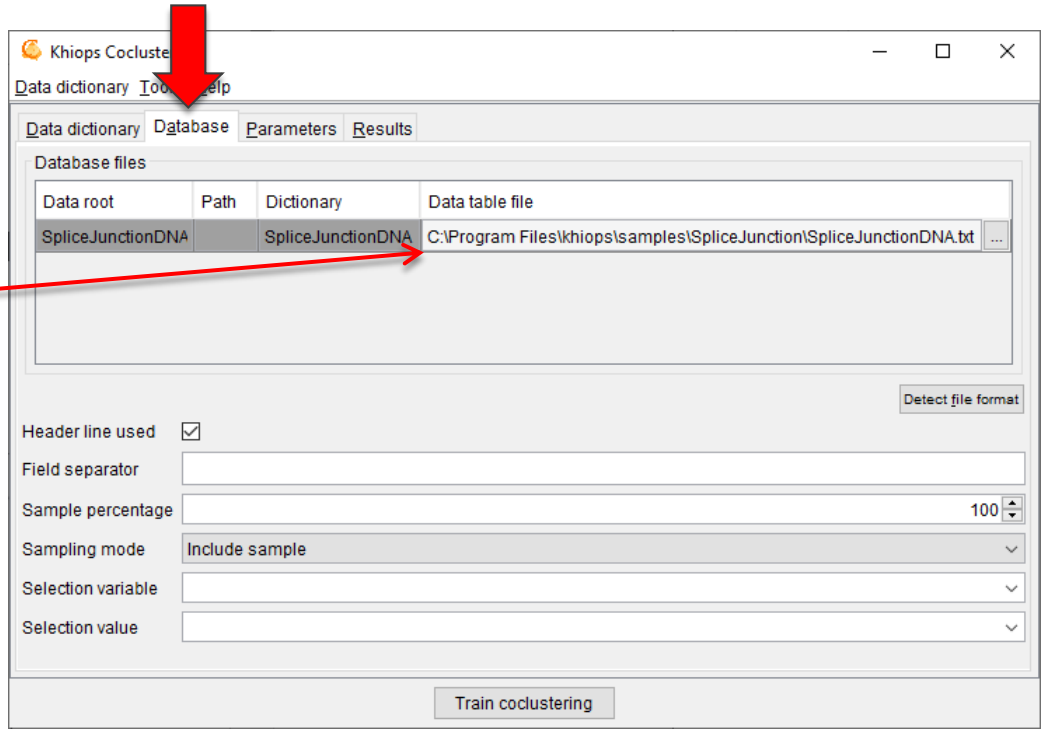


Train a triclustering model

- **Step 2 :** Specification of used database

Data table file

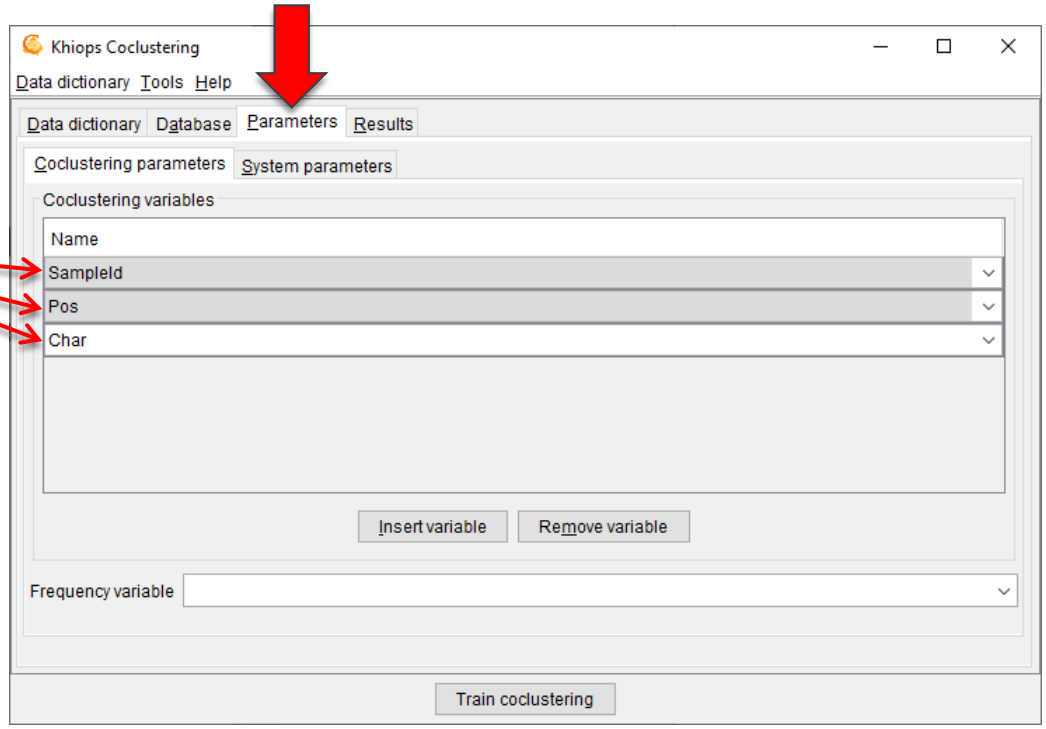
(one single file for analysis of the secondary entity)



Train a triclustering model

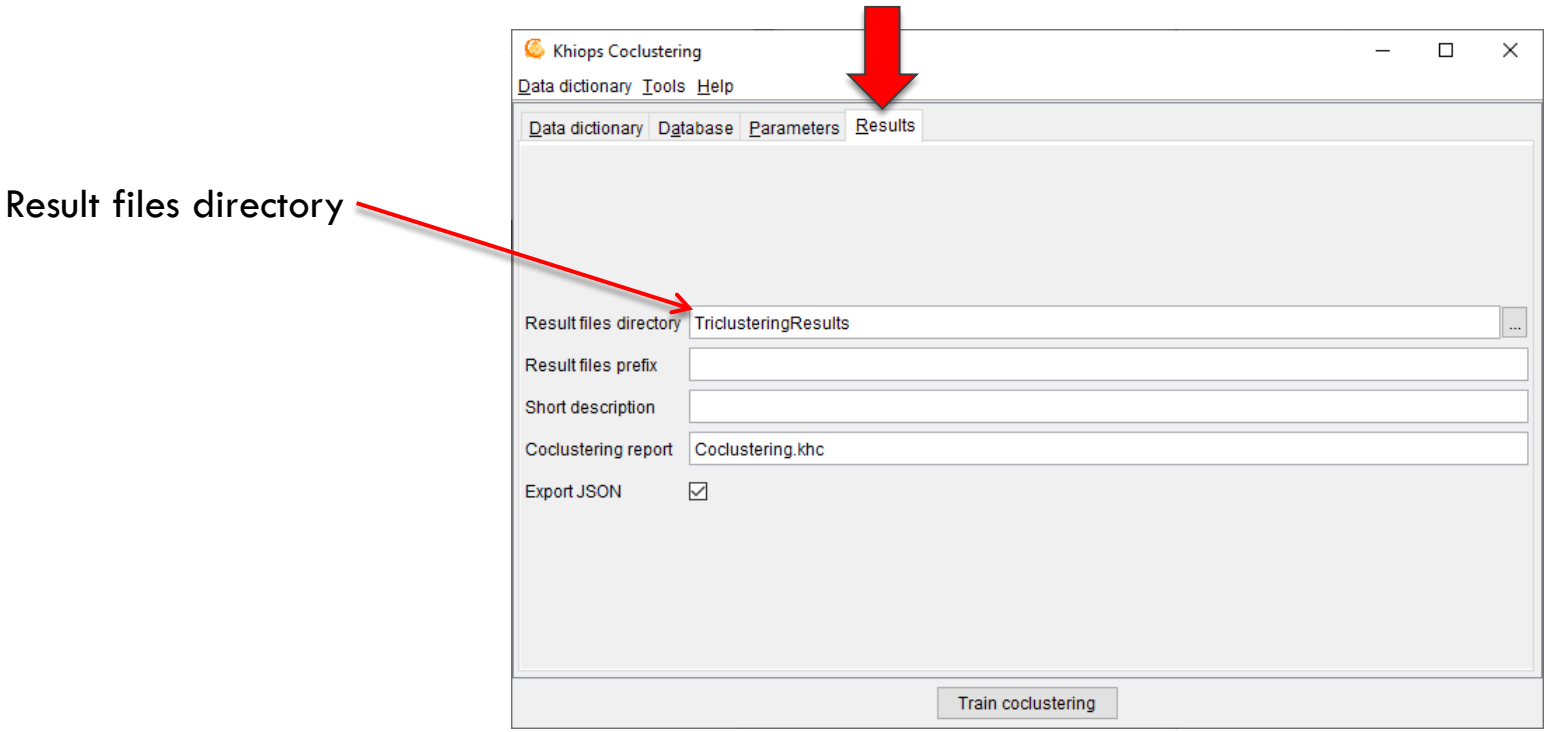
- **Step 3 :** Specification of triclustering variables

Triclustering variables



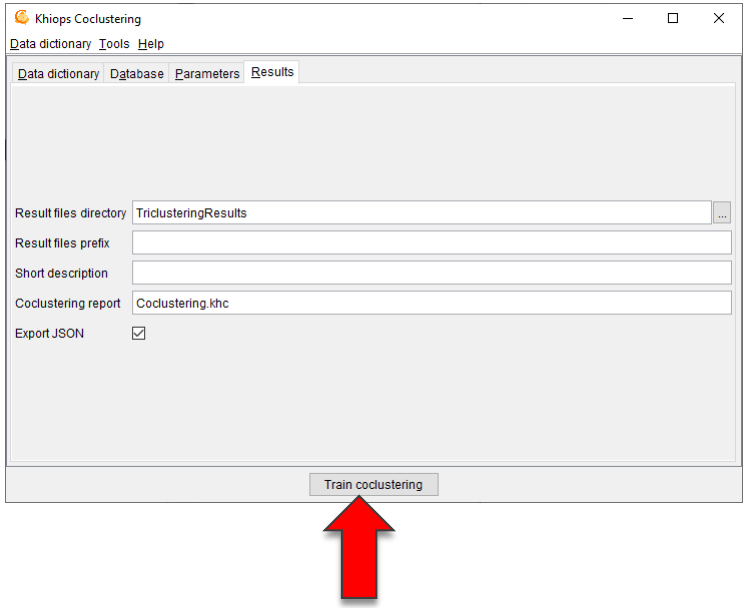
Train a triclustering model

- **Step 4 :** Analysis results

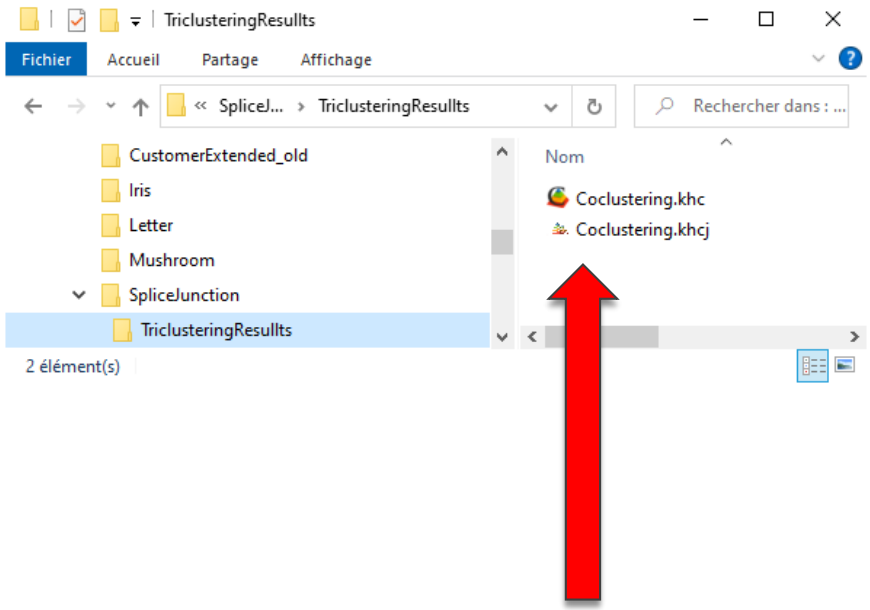


Train a triclustering model

- **Step 5 :** Start the analysis



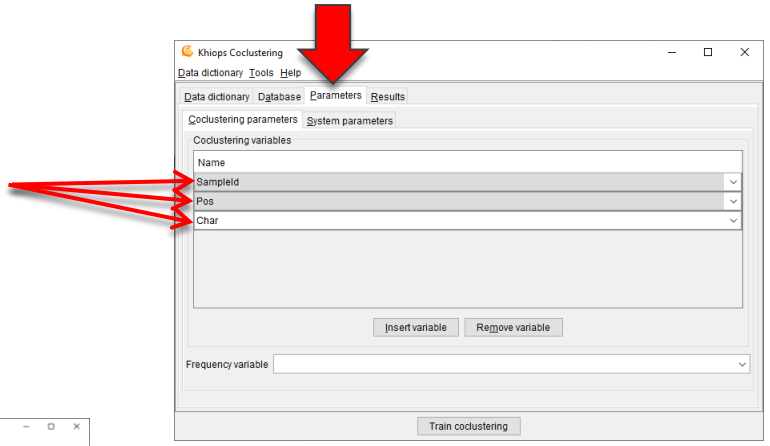
1 – Start the analysis

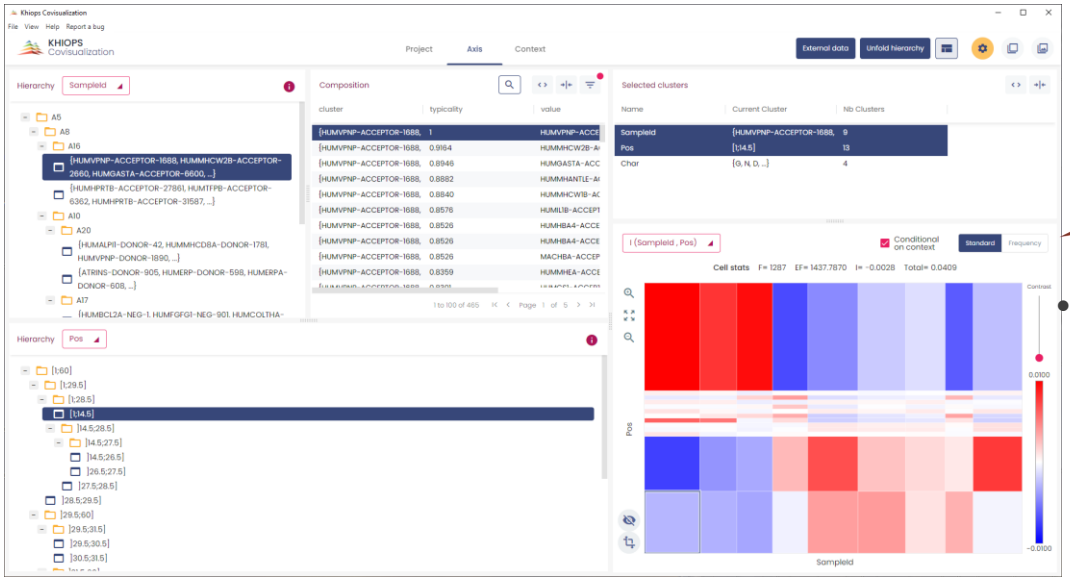


2 - Inspect the results using Khiops Covisualization
(double-click on *.khcj* file)

Khiops covisualisation: base SpliceJunctionDNA

- With Khiops Coclustering
  - Analysis of correlation between variables SampleId*Pos*Char





- With Khiops Covisualization
  - Exploratory analysis of the results

Khiops multi-table

- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops Coclustering
  - Deployment of coclustering models
    - Given a text*word coclustering model, assign new texts (with their words) to their closest cluster
    - Given a cookie*page coclustering model, assign new cookies (with their pages) to their closest cluster
    - Given a curve*X*Y triclustering model, assign new curves (with their X*Y points) to their closest cluster

- In this tutorial
  - Train a triclustering model on the SpliceJunctionDNA data table
    - Clusters of sequence samples
    - Intervals of positions in the sequences
    - Clusters of DNA chars

  - Prepare a deployment model
    - Build a deployment dictionary

  - Deploy the model on the multi-table SpliceJunction database
    - Assign new DNA sequences to trained clusters of sequences

Prepare a deployment model

- **Prerequisite :** a multi-table database
  - dictionary file
  - data files

    (ex: sample SpliceJunction)

Root entity

for deployment on new instances

Secondary entity

previously analyzed using triclustering

SpliceJunction.kdic

```
Root  Dictionary  SpliceJunction(SampleId)
{
   Categorical        SampleId ;
   Categorical        Class;
   Table(SpliceJunctionDNA)  DNA;
};

Dictionary  SpliceJunctionDNA(SampleId)
{
   Categorical        SampleId ;
   Numerical          Pos;
   Categorical        Char;
};
```

SpliceJunction.txt

SampleId	Class
AGMKPNRSB-NEG-1	N
AGMORS12A-NEG-181	N
AGMORS9A-NEG-481	N
AGMRSKPNI-NEG-1141	N
ATRINS-ACCEPTOR-1678	IE
ATRINS-ACCEPTOR-701	IE
ATRINS-DONOR-521	EI
ATRINS-DONOR-905	EI
…	

SpliceJunctionDNA.txt

SampleId	Pos	Char
AGMKPNRSB-NEG-1	1	C
AGMKPNRSB-NEG-1	2	A
…		
AGMKPNRSB-NEG-1	58	A
AGMKPNRSB-NEG-1	59	C
AGMKPNRSB-NEG-1	60	A
AGMORS12A-NEG-181	1	A
AGMORS12A-NEG-181	2	G
…		
AGMORS12A-NEG-181	59	G
AGMORS12A-NEG-181	60	G
AGMORS9A-NEG-481	1	T
AGMORS9A-NEG-481	2	G
AGMORS9A-NEG-481	3	G
…		

Prepare a deployment model

- **Step 1 :** Open an existing dictionary

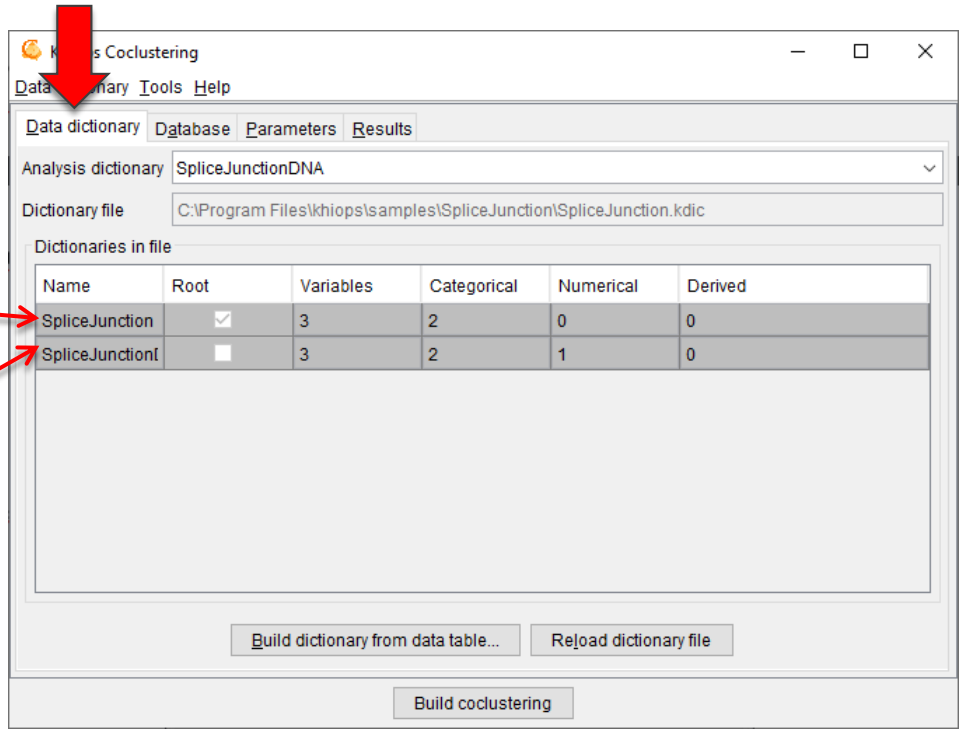
  (ex: sample SpliceJunction.kdic)

Root entity

for deployment on new instances

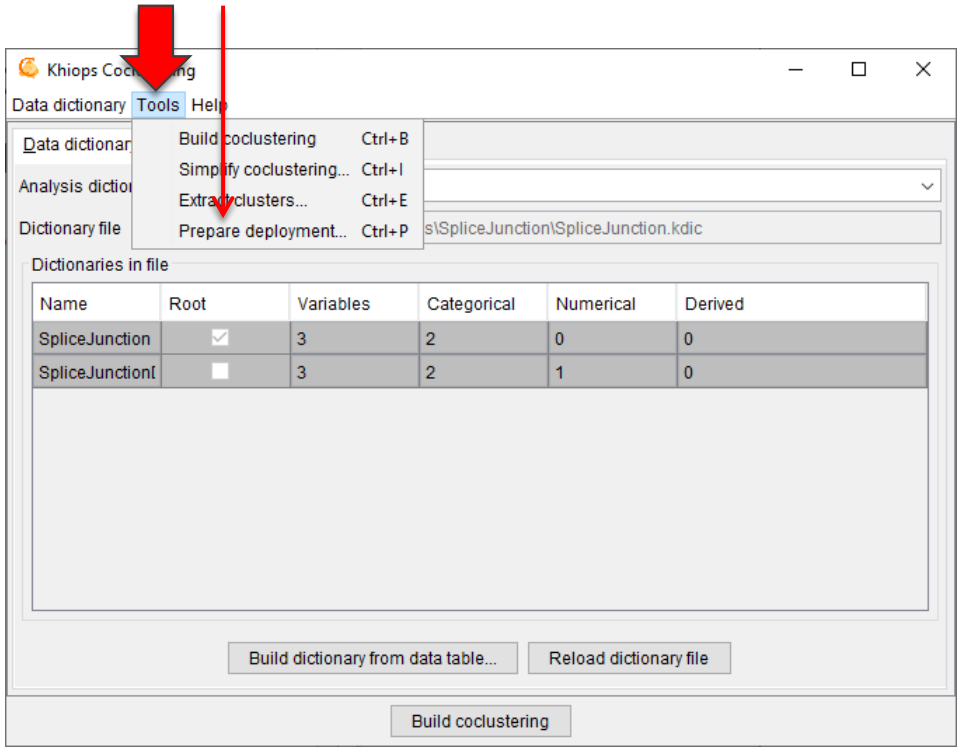
Secondary entity

previously analyzed using triclustering



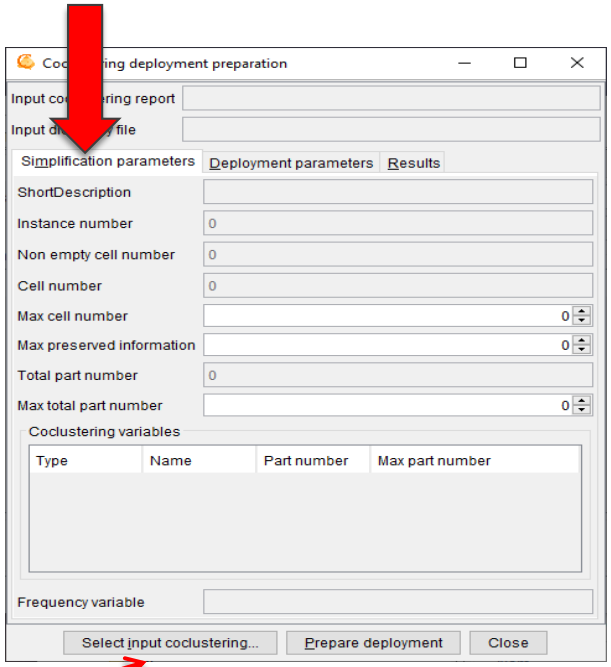
Prepare a deployment model

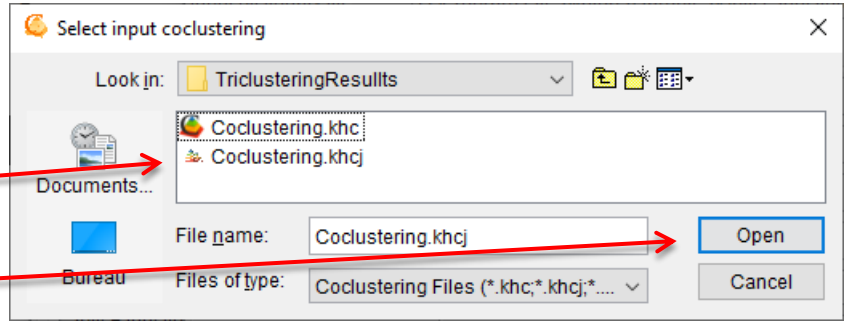
- **Step 2 :** Start « *Tools – Prepare deployment* »



Prepare a deployment model

- **Step 3 :** Select input coclustering file
  - (ex: previously trained triclustering model)



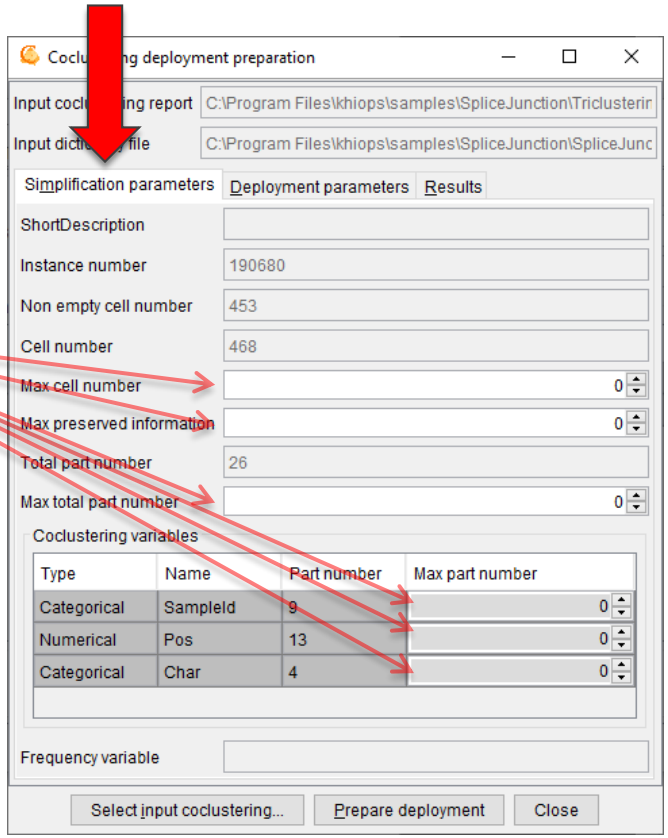


1. click on button
2. select a triclustering model file
3. open

Prepare a deployment model

- **Step 4 :** The triclustering model is summarized in the first pane
  - if necessary, specify simplification parameters

Simplification parameters



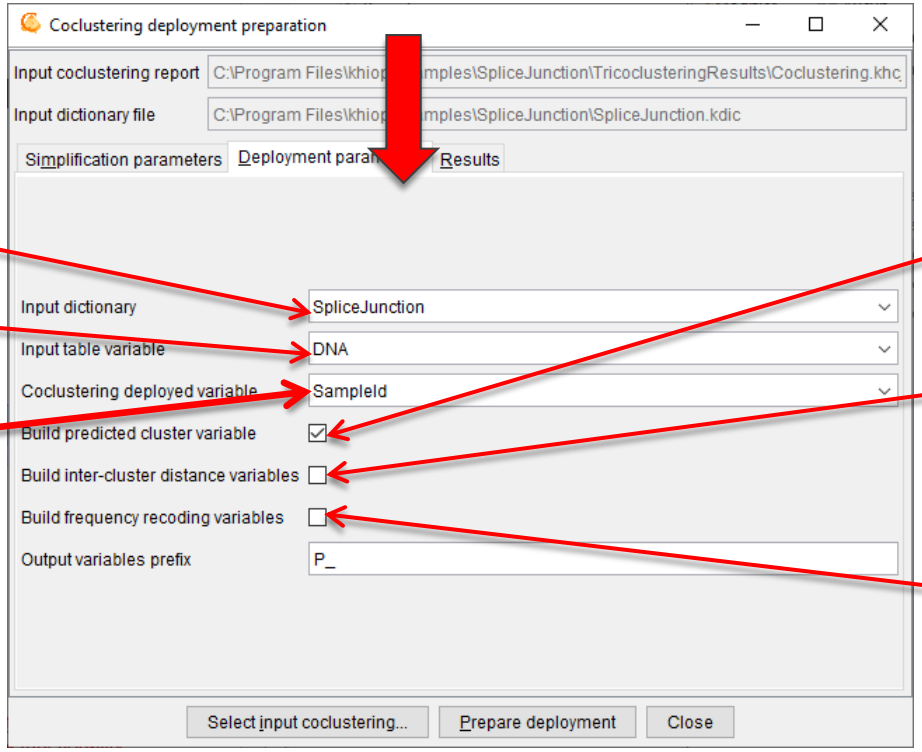
Prepare a deployment model

- **Step 5 :** Specify deployment parameters

Specification of input dictionary to enrich

```
Root Dictionary SpliceJunction(SampleId)
{
   Categorical      SampleId ;
   Categorical      Class;
   Table(SpliceJunctionDNA)  DNA;
};

Dictionary SpliceJunctionDNA(SampleId)
{
   Categorical      SampleId ;
   Numerical        Pos;
   Categorical      Char;
};
```



Specification of deployment variables to build

One variable to assign closest cluster

. closest cluster of *SampleId*

. Several variables for distance to each cluster

. distance to each cluster of *SampleId*

. Several variables for secondary record number per interval/group of the other dimensions of the triclustering

. frequency per interval of *Pos*

. frequency per group of *Char*

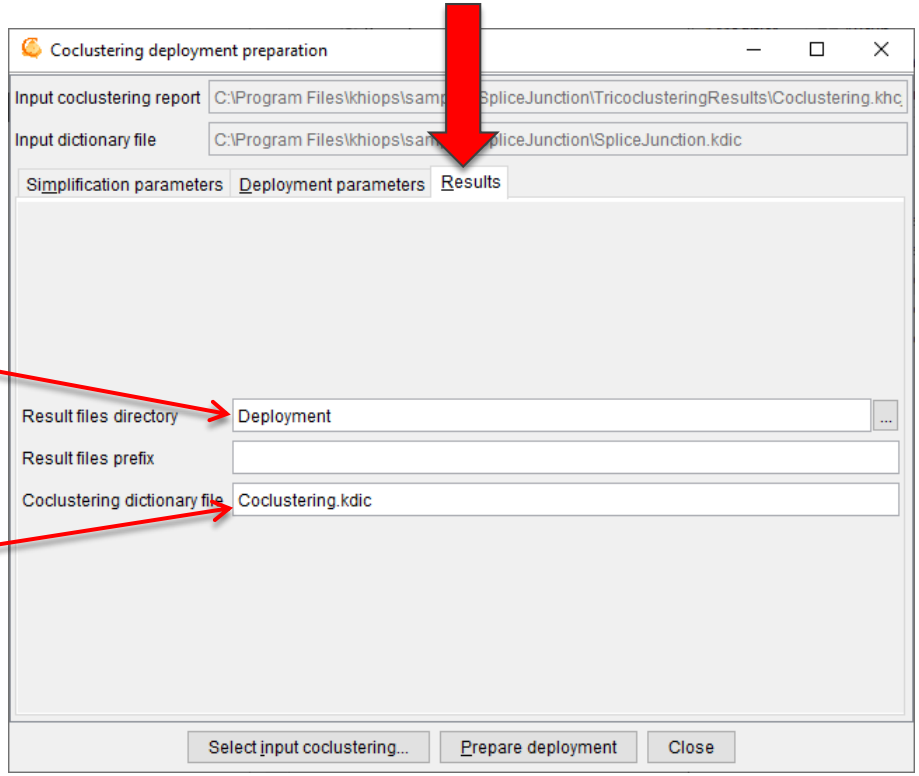
Prepare a deployment model

- **Step 6 :** Specify result parameters

Result files directory

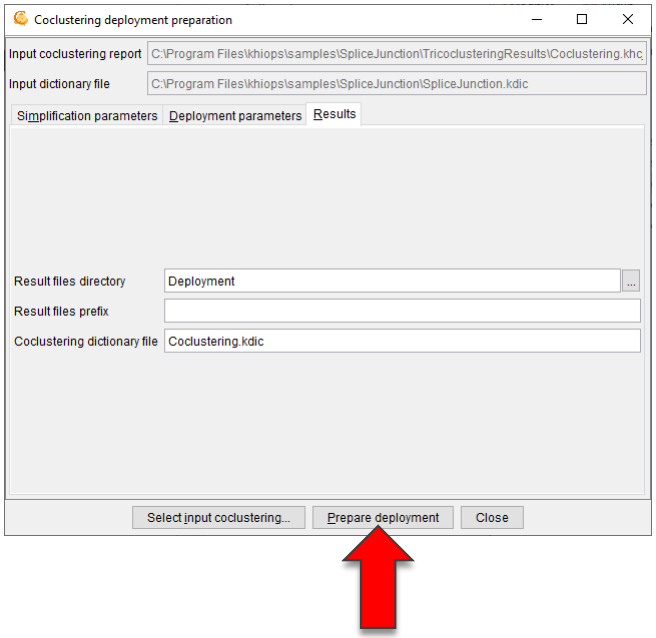
Deployment dictionary file

(to deploy cluster information on new data)

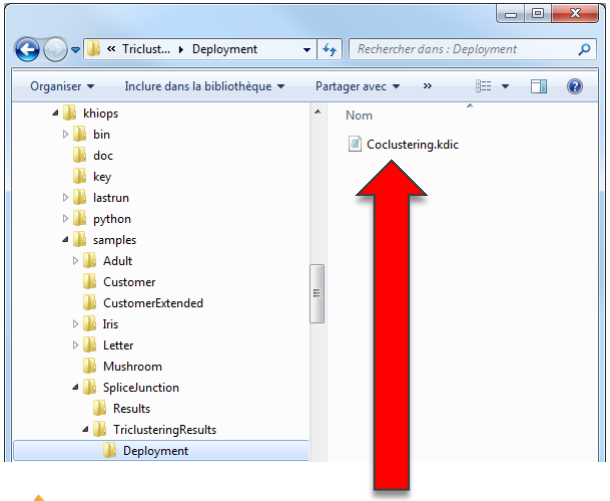


Prepare a deployment model

- **Step 7 :** Build the deployment dictionary



1 – Build the deployment dictionary



2 – The deployment dictionary is ready for use with Khiops « *Transfer database* » functionality

Khiops multi-table

- Khiops can deal with multi-table databases
  - star schema: one root entity and several 0-1 or 0-n secondary entities
  - snowflake schemas and beyond

- Impact on Khiops Coclustering
  - Deployment of coclustering models
    - Given a text*word coclustering model, assign new texts (with their words) to their closest cluster
    - Given a cookie*page coclustering model, assign new cookies (with their pages) to their closest cluster
    - Given a curve*X*Y triclustering model, assign new curves (with their X*Y points) to their closest cluster

- In this tutorial
  - Train a triclustering model on the SpliceJunctionDNA data table
    - Clusters of sequence samples
    - Intervals of positions in the sequences
    - Clusters of DNA chars

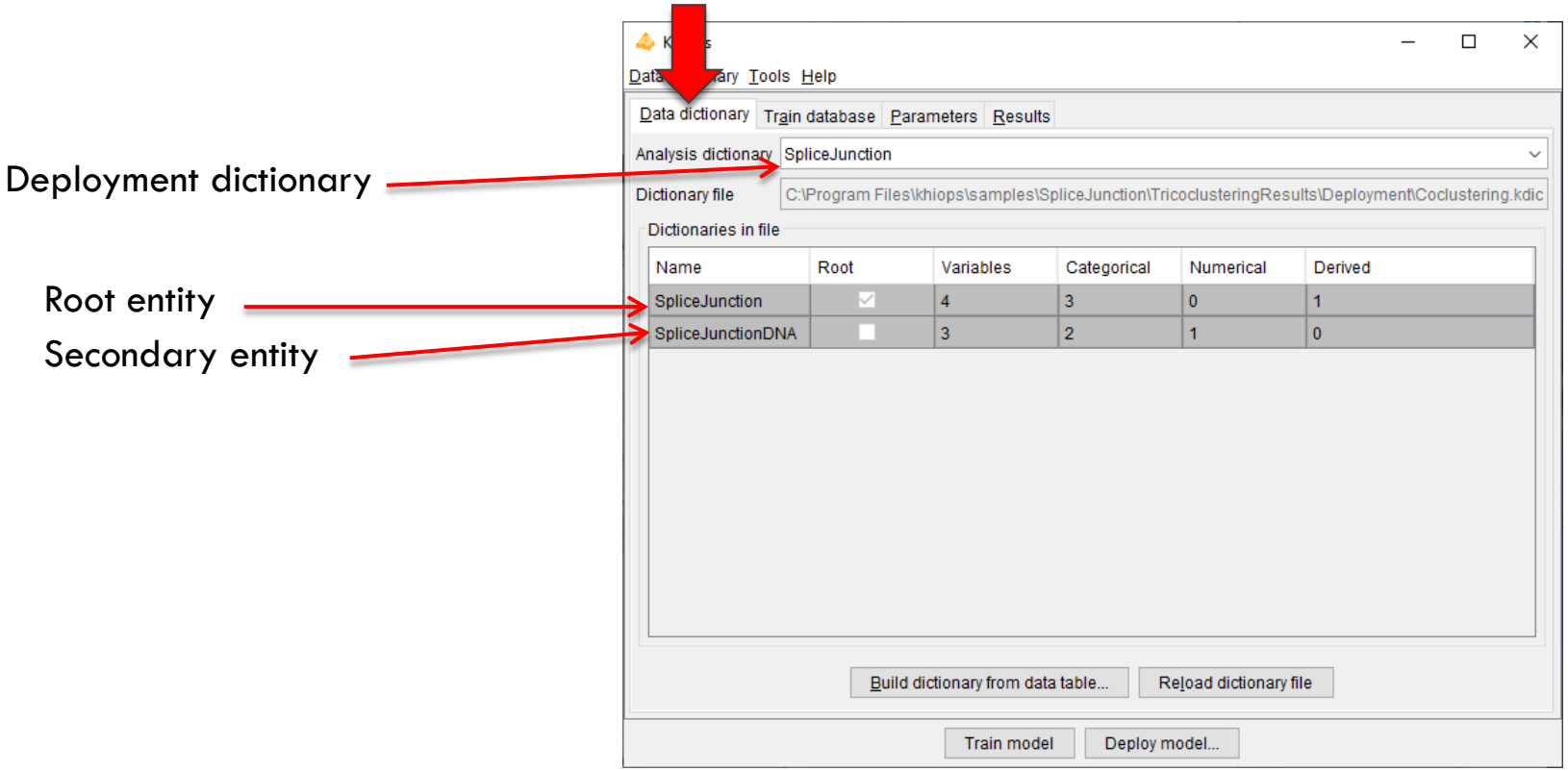
  - Prepare a deployment model
    - Build a deployment dictionary

  - Deploy the model on the multi-table SpliceJunction database
    - Assign new DNA sequences to trained clusters of sequences

Deploy the model

- **Step 1 :** Open the deployment dictionary file with Khiops

  (ex: Samples\SpliceJunction\TriclusteringResults\Deployment\Coclustering.kdic)



Deploy the model

- **Step 2 :** If necessary, select deployment variables

  (use « *Inspect current dictionary* » by right-click on dictionary SpliceJunction)

Initial variables

Used by default

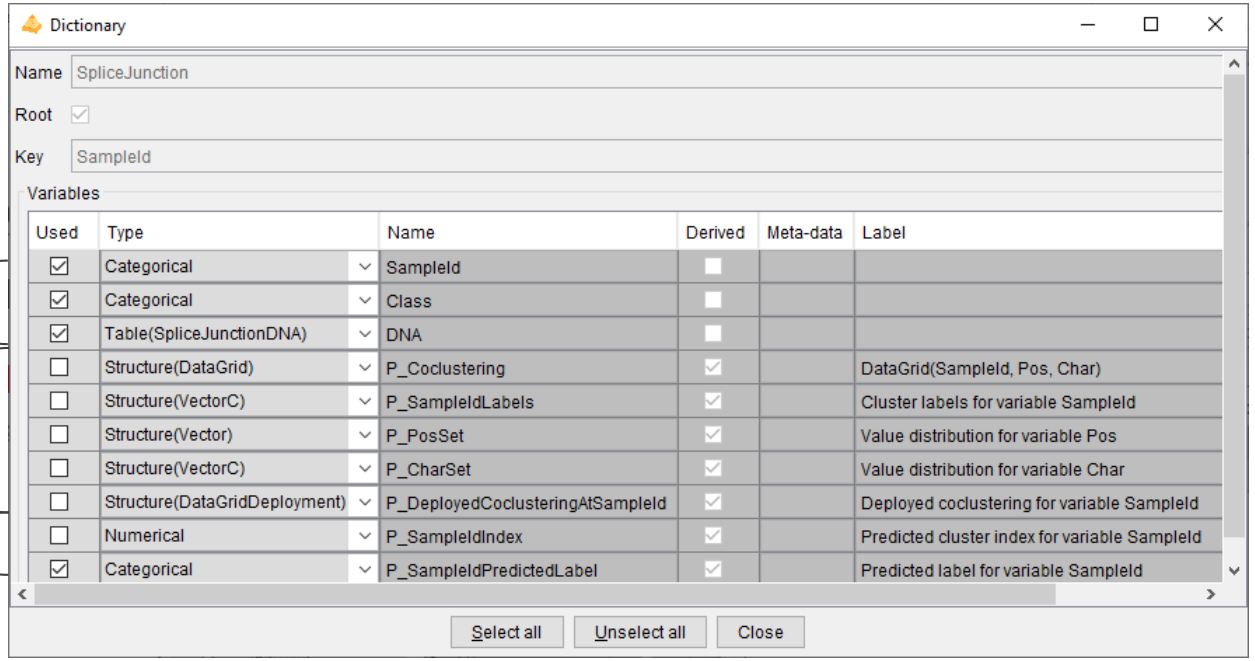
Model variables

(technical variables)

Deployment variables

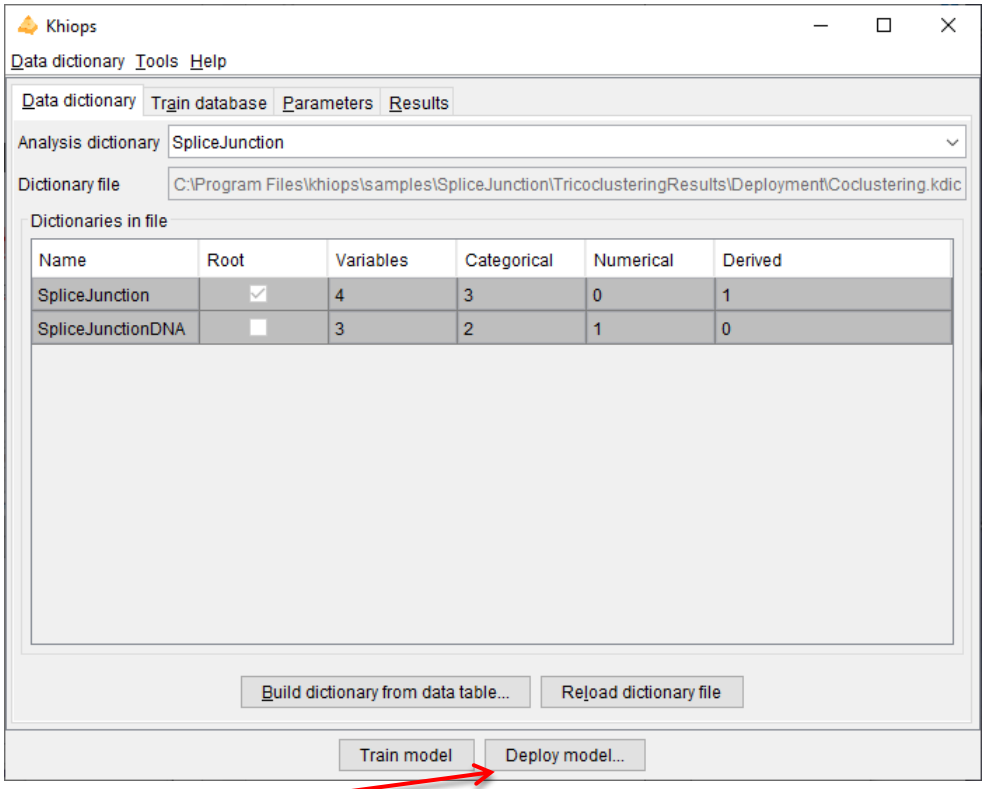
Cluster index (unused by default)

Cluster label (used by default)



Deploy the model

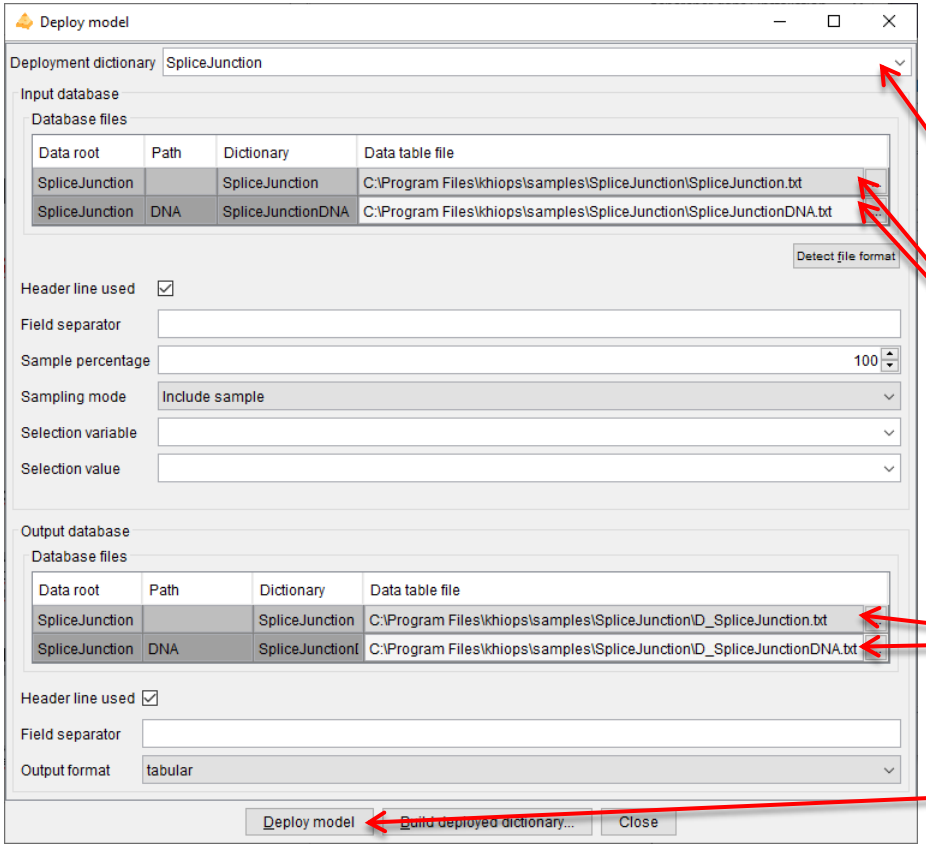
- **Step 3 :** Open the « *Deploy model* » dialog box



click on button

Deploy the model

- **Step 4 :** Specify the file transfer parameters



1 Specify the deployment dictionary

2 Specify the input data table files
- splice junction samples with their DNA sequence
- all files are mandatory

3 Specify the output data table files
- secondary files are optional

4 Deploy
- The output files are enriched with new fields derived from the triclustering analysis

End of tutorial: summary

- Khiops
  - Optimal data preparation based on discretization and value grouping
  - Scoring models for classification and regression
  - Correlation analysis between pairs of variables

- Khiops Visualization
  - Analysis of Khiops results using an interactive visualization tool

- Khiops Coclustering
  - Correlation analysis of two or more variables using a hierarchical coclustering model

- Khiops Covisualization
  - Exploratory analysis of Khiops Coclustering results using an interactive visualization tool

- Multi-table functionalities
  - Multi-table database
  - Automatic feature construction
  - Multi-table functionalities in Khiops and Khiops Coclustering