Orange Labs

October, 2018

Khiops Enneade Guide 9.0.0.0 (mono table)

Abstract

Khiops Enneade is a tool which produces a clustering model using a modified version of the K-Means algorithm.

Khiops Enneade is available both in user interface mode and in batch mode, such that it can easily be embedded as a software component in a data mining deployment project.

Khiops Enneade belongs to the Khiops family.

This report describes the parameters of Khiops Enneade and all its functionalities, the management of the data dictionaries and the list of derivation rules that enables to construct new variables, and finally the analysis and evaluation reports. The user interface is presented as well, together with a quick start of the tool.

Summary

1. Presentation ... 4

2. Detailed functionalities ... 4

2.1. Dictionary file ... 5
2.1.1. Reload dictionary file ... 5
2.1.2. Build dictionary from data table ... 6
2.1.3. Dictionary file menu ... 7
2.1.3.1. Open 7
2.1.3.2. Close 7
2.1.3.3. Save 7
2.1.3.4. Save under 7
2.1.3.5. Dictionaries in file/Inspect current dictionary 7

2.2. Train database ... 8
2.2.1. Fill test database settings ... 8

2.3. Test database ... 9
2.3.1. Import train database settings ... 9

2.4. Analysis parameters ... 9
2.4.1. Predictors ... 9
2.4.1.1. K-Means advanced parameters 9
2.4.2. System parameters ... 12

2.5. Analysis results ... 13

2.6. Tools menu ... 13
2.6.1. Check database ... 14
2.6.2. Analyse database ... 14
2.6.3. Transfer database ... 15
2.6.4. Evaluate predictors ... 16

3. Inputs-outputs ... 17

3.1. Format of the database files ... 17

3.2. Format of the dictionary files ... 18
3.2.1. Dictionary file ... 18
3.2.2. Dictionary ... 18
3.2.3. Edition of dictionary files by means of Excel ... 19
3.2.4. Derivation rules ... 19

3.3. Reports ... 19
3.3.1. Preparation report ... 19
3.3.2. Modeling report ... 20
3.3.3. Evaluation report ... 21

4. Batch mode ... 23

4.1. Register and replay a batch scenario ... 23

4.2. Integration in a program to industrialize a data analysis process ... 24

4.3. List of command line options ... 24

4.4. Technical limitations ... 25
4.4.1 Preprocessing before the elaboration of the clustering models ... 25
4.4.2 Elaboration of the clustering models ... 25
4.4.3 Evaluation of the clustering models ... 26
4.4.4 Deployment of the clustering models ... 26
4.4.5 Memory overflow ... 26
4.4.5 Known bugs ... 26

1. Presentation

Khiops Enneade handles the data preparation phase and the modeling phase for a K-Means clustering. The clustering model produced could be used as scoring model for supervised learning tasks if a target variable has been specified at the beginning of the analysis.

The database must be formatted according to a text file format, with a line per record, one header line containing the variable names and a field separator (tabulation by default).

The first step is the specification of the data dictionary, that is the choice of the variable types (categorical or numerical) in the database to analyse. This dictionary is automatically built by Khiops Enneade owing to a parsing of the database file. The built dictionary is saved in a dictionary file, which basic syntax allows easy modifications. The Data Miner must then validate the variable types in the built dictionary, and eventually specify which variables to ignore in the analysis or construct new variables owing the derivation rule language.

The second step is to check the correctness of the database file. In this step, Khiops Enneade parses the database file and completely checks formatting or variable type errors.

In a third step Khiops Enneade elaborates a K-Means clustering using a pretreatment step. The pretreatment step is automatically adjusted but it could also be defined by the user (in user interface mode or in batch mode) for the numerical and the categorical variables.

The following reports are produced:

- A preparation report, for univariate analysis, is produced at the end of the data analysis, based on the train data set. It summarizes the information contained in each analysed variable.
- A modeling report summarizes the features of the built clusters.
- Two evaluation reports, based on the train and test data, evaluate the performance of the scoring model. If a target variable has been specified at the beginning of the analysis some classification performance indicators are computed as well and are added in the evaluation reports.

A new dictionary is produced allowing a deployment of the clustering model.

The fourth step is the deployment step. This is done by applying the clustering dictionary on new data, in order to compute recoding variables, cluster membership - and scores (if a target variable has been specified at the beginning of the analysis). This functionality can also be used to construct any new variable, described using the derivation rule language.

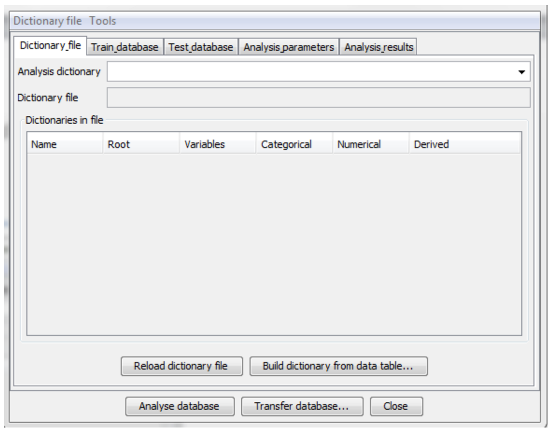
An Khiops Enneade session can be registered in a scenario file, which can be replayed by Khiops Enneade in batch mode. This allows to automatize data preparation, modeling and deployment in a Data Mining project and to easily integrate the process in any information system.

2. Detailed functionalities

This section is organized as the user interface of Khiops Enneade. The subtitles correspond to the subwindows or submenus of Khiops Enneade main window. The options described in each subsection correspond to the fields or menu actions available in the interface.

Note: The menu “Dictionary”, “Train Database” and “Test database” are the same as the Khiops menus. If the reader already known Khiops, he can focus on the “Analysis Parameters “ menu.

2.1. Dictionary file



A dictionary file is a text file with the extension .kdic. It contains the definition of one or several dictionaries, each one describing the set of variables to use in a data analysis. The dictionaries can be automatically built from the data table file owing to a file parsing, automatically enriched during data preparation or modeling, or manually modified by the Data Miner using a text editor (for example Notepad). The dictionary files built by Khiops Enneade use tabulations as field separators, which allows direct copy-paste interactions with Excel. This provides a way to quickly sort, select and modify large numbers of variable definitions.

Khiops Enneade allows to **Open** a **Dictionary file**. Opening a dictionary file amounts to loading its dictionaries into memory and making them available for data analysis. The **Save** and **Save under** actions write dictionaries to a dictionary file, whereas the **Close** action cleans the memory. The **Reload dictionary file** action reads again a dictionary file, which may have been modified using an external text editor. The list of available dictionaries can be browsed using Khiops Enneade.

Analysis dictionary: name of the dictionary related to the data to analyse. Mandatory field.

Dictionary file: name of the dictionary file related to the data to analyse. Read-only field that shows the name of the current dictionary file.

Dictionaries in file: list of available dictionaries, with statistics describing the used variables (Name, Variables, Categorical, Numerical, Derived).

Dictionary files are opened only during the execution of the actions. They are then loaded into memory and available everywhere inside Khiops Enneade.

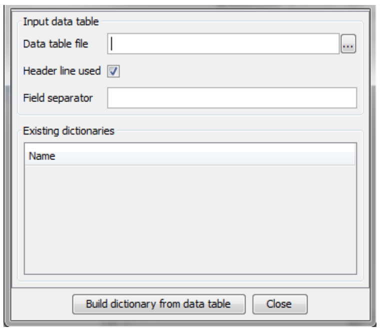
2.1.1. Reload dictionary file

Reload into memory the current dictionary file.

This action allows to modify the dictionary file using an external text editor (Notepad for example), to save the modifications, and to take them into account into Khiops Enneade by reloading the dictionary file.

In case of invalid dictionary file, the current dictionaries are kept in memory.

2.1.2. Build dictionary from data table



This action opens a window that allows to build dictionaries from data tables.

Data table file: name of the data table file to analyse. Mandatory field.

Header line used: (default: true). If the file have a header line, Khiops Enneade will use the header line fields as variables names; otherwise, the variables will be names Var1, Var2...

Field separator: by default, if nothing is specified, the tabulation is used as the field separator.

Build dictionary from data table: start the analysis of the data table file to build a dictionary. The first lines of the file are analysed in order to determine the type of the variables: Categorical, Numerical, Date, Time or Timestamp. After analysis, the user can choose the name of the dictionary.

Close: close the window. If dictionaries have been built, proposes to save them in a dictionary file

The values in the data table file are parsed in order to guess their type.

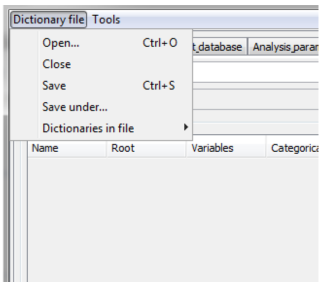
- values with format YYYY-MM-DD (e.g. 2014-01-15) are recognized as Date variables,
- values with format HH:MM:DD (e.g. 11:35:20) are recognized as Time variables,
- values with format YYYY-MM-DD HH:MM:DD are recognized as Timestamp variables,
- for other Date, Time or Timestamp formats (e.g. Date format DD/MM/YYYY), a specific meta-data value is used (see paragraph 3.2.2. Dictionary) to specify the format used for the variable,
- other values with numerical format are recognized as Numerical values,
- other values are recognized as Categorical values.

Dictionaries are built automatically for convenience, but they should be checked carefully by the data miner. For example, zip codes are made of digits and recognized as Numerical variables, whereas they are Categorical variables. Values such as 20101123 or 20030127 are recognized as Date with format YYYYMMDD, whereas they could be Numerical.

The Date, Time or Timestamp formats can be erroneous (example: 2010-10-10 is ambiguous w.r.t. the format: “YYYY-MM-DD” or “YYYY-DD-MM”). The meta-data must be corrected directly in the dictionary file is necessary. In some cases, a date, time or timestamp variable may have a format not recognized by Khiops Enneade. This the case for example for formats where the century is not specified (e.g: “YY-MM-DD”). In that case, the corresponding variable should be declared as Categorical (and not used), and a new variable can be built using derivation rules, as illustrated below.

```
Unused      Categorical  MyDate ;          // Date with unrecognized format "YY-MM-DD"
            Date         MyCorrectDate = AsDate(Concat("20", MyDate), "YYYY-MM-DD");     // Correction if all centuries are 20th
Unused      Numerical    MyCentury = If(LE(AsNumerical(Left(MyDate, 2)), 15), 20, 19);   // 20th for year below 15, 19th otherwise
            Date         MyCorrectDate2 = AsDate(
                                   Concat(AsCategorical(MyCentury), MyDate), "YYYY-MM-DD");  // Correction using MyCentury
```

2.1.3. Dictionary file menu



2.1.3.1. Open

An open dialog box asks the name of the dictionary file to open.

In case of invalid dictionary file, the current dictionaries are kept in memory.

2.1.3.2. Close

The dictionaries are removed (from memory only). The potential pending modifications are lost if they have not been saved.

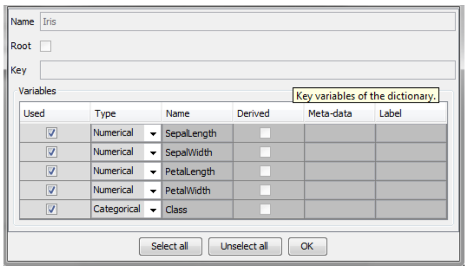
2.1.3.3. Save

The memory dictionaries are saved under the current dictionary file.

2.1.3.4. Save under

A save dialog box asks the name of the dictionary file to save.

2.1.3.5. Dictionaries in file/Inspect current dictionary



Name Iris

Root ☐

Key

Key variables of the dictionary.

Variables

Used	Type	Name	Derived	Meta-data	Label
✓	Numerical	SepalLength	☐		
✓	Numerical	SepalWidth	☐		
✓	Numerical	PetalLength	☐		
✓	Numerical	PetalWidth	☐		
✓	Categorical	Class	☐		

Allow to inspect and partly modify a dictionary chosen among the list of available dictionaries. The dictionary to inspect must be selected among the dictionaries in file.

The action is available both from the menu and using a right click button on the selected dictionary.

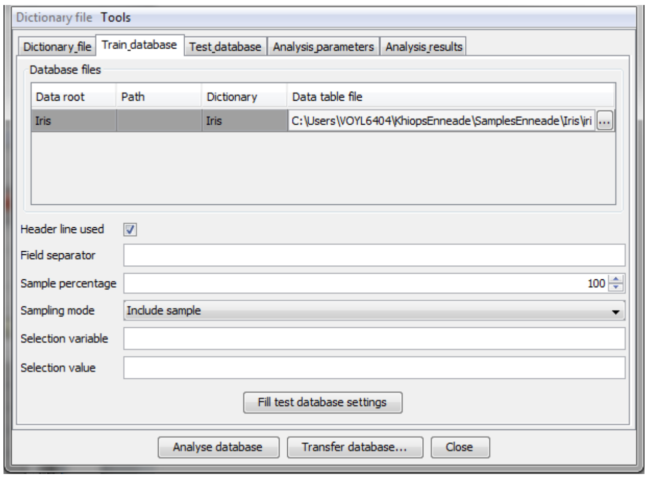
During the inspection of a dictionary, the list of its variables can be browsed into a sub-window. For each variable, the following properties are displayed: **Used**, **Type**, **Name**, **Derived**, **Meta-data** and **Label**.

The Data Miner can choose whether to keep or not the variable for data analysis, using the **Used** property. The **Select all** and **Unselect all** buttons allow to choose all or no variables.

It is also possible to change the **Type** of variables: Numerical, Categorical, Date, Time or Timestamp.

Remark: for large scale modifications in a dictionary, it is preferable to update the dictionary file using an external text editor (Notepad, WordPad…), to save the file with the external editor, and then to reload the dictionary.

2.2. Train database



Database files

Data root	Path	Dictionary	Data table file
Iris		Iris	C:\Users\VOYL6404\KhiopsEnneade\SamplesEnneade\Iris\iri

Header line used	✓
Field separator	
Sample percentage	100
Sampling mode	Include sample
Selection variable	
Selection value	

Database files: name of the database files to analyse.

Data table file: name of the data table file. Mandatory field.

Header line used: (default: true). If the file does not have a header line, Khiops Enneade considers the variables in the dictionary to analyse the fields in the file.

Field separator: by default, if nothing is specified, the tabulation is used as the field separator.

Khiops Enneade can be used to extract a subpart (or its exact complementary) of the records in a database file. This sampling is specified with a sample percentage of the records (to keep or to discard). The sampling is a random sampling, but is reproducible (the random seed is always the same).

Sample percentage: percentage of the samples (default: 100)

Sampling mode: to include or exclude the records of the sample (default: include sample). This allows to extract a train sample and its exact complementary as a test sample (if the same sample percentage is used both in train and test, in include sample mode in train and exclude sample mode in test).

Another way to build train or test samples is to use a selection variable and a selection value.

Selection variable: when nothing is specified, all the records are analysed. When a selection variable is specified, the records are selected when the value of their selection variable is equal to the selection value.

Selection value: used only when a selection variable is specified. In that case, the value must be a correct value (numerical value if the selection variable is a numerical variable).

2.2.1. Fill test database settings

This action allows to fill all the fields of the "Test data base" pane, by copying them from the "Train database" pane. The only change is the "Sampling mode" which value is inverted in the test pane, in order to get a test sample that is the exact complementary of the train sample.

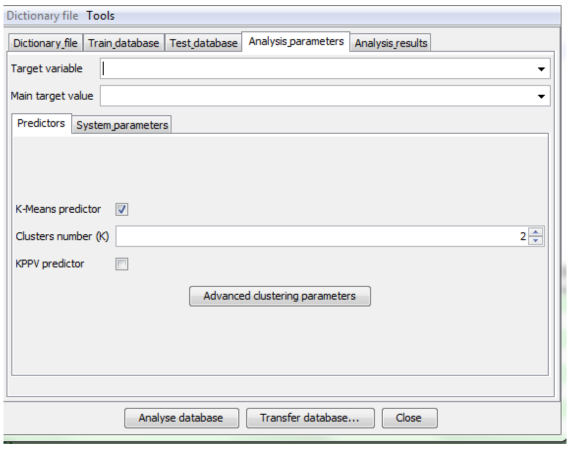
2.3. Test database

The test database is defined exactly in the same way as the train database.

2.3.1. Import train database settings

This action has the same effect as the action "Fill test database settings" of the train pane.

2.4. Analysis parameters



Target variable
Main target value
K-Means predictor ✓
Clusters number (K) 2
KPPV predictor

Target variable: name of the target variable. The learning task is always unsupervised learning (clustering) but a categorical target variable can be specified to obtain also a scoring model based on the cluster membership.

Main target value: value of the target variable in case of supervised learning, for the lift curves in the evaluation report.

2.4.1. Predictors

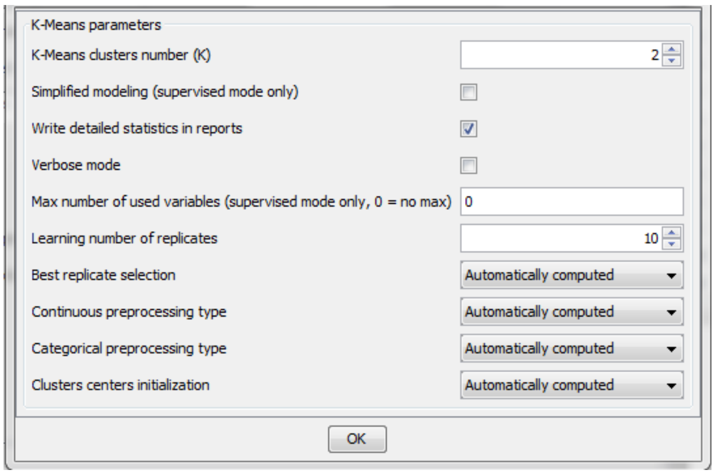
K-Means predictor: builds a K-Means model (default: true).

K-Means clusters number (K): Number of clusters. Note: the algorithm may find a lower number (default: 2).

KPPV predictor : builds a model, using the k-nearest neighbors algorithm.

2.4.1.1. K-Means advanced parameters

These parameters are user constraints that allow to control all the parameters of the K-Means process. Their use might decrease the performance, compared to the default mode (without user constraints).



K-Means parameters	
K-Means clusters number (K)	2
Simplified modeling (supervised mode only)	☐
Write detailed statistics in reports	☑
Verbose mode	☐
Max number of used variables (supervised mode only, 0 = no max)	0
Learning number of replicates	10
Best replicate selection	Automatically computed
Continuous preprocessing type	Automatically computed
Categorical preprocessing type	Automatically computed
Clusters centers initialization	Automatically computed

K-Means parameters

K-Means clusters number (K): Number of clusters. Note: the algorithm may return a lower number (default: 2).

Simplified modeling (supervised mode only):

Before clustering a MAP Naive Bayes predictor is computed without any clustering consideration; therefore considering only the "supervised classification task".

The MAP Naive Bayes predictor keeps the best subset of variables of the Selective Naive Bayes* predictor. It usually exploits few variables.

The clustering is then performed only with this subset. This allow to have a "simplified" model easier to interpret but potentially less accurate. The time to compute the clustering is therefore lower but with the price of the computation of the MAP Naive Bayes predictor.

To see the difference between the "simplified model" and the classic model the tool has to be used twice.

*The Selective Naive Bayes predictor performs a variable selection using a greedy heuristic. The selection consists in successive Forward and Backward passes, and is repeated several times. The Selective Naive Bayes is then averaged among all the evaluated models. The overall training time is O(*NK*log(*NK*)) where *N* is the number of instances and *K* the number of variables.

Write detailed statistics in reports : computes additional statistics. This option will require more memory, and may take a significant amount of computing time, depending on your dataset size. The additional statistics are listed below (and described in section 3.3):

- Continuous mean values
- Continuous median values
- Native attributes probabilities
- Percentage per lines - Native attributes probabilities

For more information on these statistics, see the "Evaluation report" section.

Verbose mode: (default: no)

Max number of evaluated variables : if a value is specified, variables will be sorted on their “level” (univariate predictive importance), and only the most predictive variables will be used for the clustering (supervised mode only). Default : 0 (all the variables will be used)

Note: If this option is used simultaneously with “simplified modeling” option (see below):

- and the specified value is lower than the number of variables of the MAP model : the model will incorporate the most predictive variables of the subset of the MAP model.
- and the specified value is higher than the number of variables of the MAP model : the specified value will be reduced to the number variables in the MAP model.

Learning replicate number: Number of times (default: 10) the clustering is repeated, each time with a new set of initial cluster centroid positions. Enneade returns the best solution, based on the “best replicate selection” option. You may increase this value to try to obtain a better solution.

Best replicate selection: Method used to determine which replicate is the best (default: Automatically computed, that is, Adjusted Rand Index in supervised mode, and Distance min in unsupervised mode)

- Distance min: the best replicate is the one which has the lowest distance.
- ARI (Adjusted Rand Index) max : the best replicate is the one which has the highest ARI value.
- Predictive Clustering min : A criterion for predictive clustering
- Davies Bouldin min : the best replicate is the one which has the lowest index.

Continuous preprocessing type: Method used for the pretreatment of the numerical attributes (default: Automatically computed):

- Automatically computed:
  - If a target variable is specified (*)(#): conditional info (CI) , i.e. negative log of the conditional probability of the source variable given the target variable (-log(p(X|Y))., after a supervised preprocessing
  - Otherwise: Rank Normalization
- No preprocessing: X is not preprocessed
- Unused variables: X is not used
- Rank normalisation: mean normalized rank (rank between 0 and 1) of the instances of the interval
- Center reduction: (X – Mean(X)) / StdDev(X)
- Binarization(*)(#) (after a supervised preprocessing)
- Hamming Conditional Info(*)(#) (a mix between log of the conditional probability (CI) and binarization)
- Conditional Info with priors : lop(P(X|C) * P(C)
- Entropy : P(X|C) * lop(P(X|C)
- Entropy with priors : P(X|C) * lop(P(X|C) * P(C)
- Normalization: (X – Min(X)) / (Max(X) – Min(X))

Note 1: The preprocessing with a (*) splits a single variable (a single column) into several columns.

Note 2: The preprocessing with a (#) may discard unuseful variables.

Categorical preprocessing type: Method used for the pretreatment of the categorical attributes (default: Automatically computed):

- Automatically computed:
    - If a target variable is specified (*)(#) : conditional info (CI), i.e. negative log of the conditional probability of the source variable given the target variable (-log(p(X|Y))., after a supervised grouping
    - Otherwise (*): Basic Grouping + binarisation
- Unused varaibles: X is not used
- Binarization(*)(#) (after a supervised grouping)
- Hamming Conditional Info(*)(#) (a mix between log of the conditional probability (CI) and binarization)
- Conditional Info with priors : lop(P(X|C) * P(C)
- Entropy : P(X|C) * lop(P(X|C)
- Entropy with priors : P(X|C) * lop(P(X|C) * P(C)
- Basic grouping + binarisation (*)(#): Basic value grouping method that builds one group for each frequent explanatory values. The infrequent values (below a frequency threshold) are unconditionally grouped in a "garbage" group. The remaining infrequent values are also grouped until the required number of groups is reached (the min frequency is set to 1 and the max number of groups is set to 10) + binarisation.

Note 1: The preprocessing with a (*) splits a single variable (a single column) into several columns.

Note 2: The preprocessing with a (#) may discard unuseful variables.

Clusters centers initialization : Method used for initialize the initial centers of the clusters (default: Automatically computed, that is : Kmean++R in supervised mode, and Kmean++ in unsupervised mode).

Available methods : Random; Sample; Kmean++; Kmean++R; Rochio, then split; Min-Max (random), Min-Max (deterministic); Variance partitioning; Class decomposition; Bisecting

2.4.2. System parameters

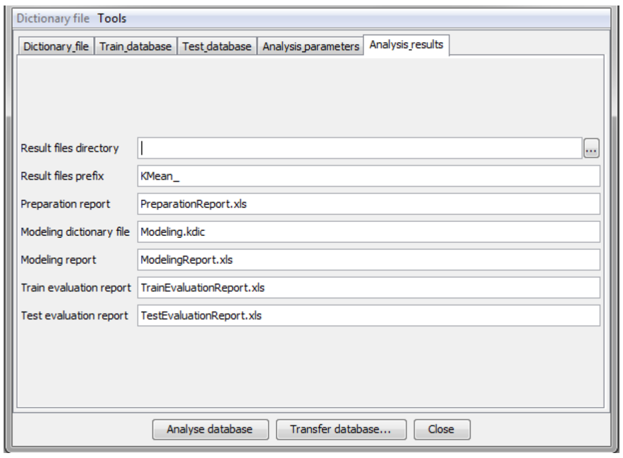
Max number of items in reports: allows to control the size of reports, by limiting the number of reported items, such as the number of lines or rows in contingency tables, the number of detailed groups of values or the number of values in each detailed group (default: 1 000 000).

Max number of error messages in log: allows to control the size of the log, by limiting the number of messages, warning or errors (default: 20).

Memory limit in Mo: allows to specify the max amount of memory available for the data analysis algorithms. By default, this parameter is set to the size of the RAM available for the Windows applications. This parameter can be decreased in order to keep memory for the other Windows application, or increased in the limit of the available RAM.

Temp file directory: name of the directory to use for temporary files (default: none, the system default temp file directory is then used).

2.5. Analysis results


Dictionary file Tools
Dictionary file | Train database | Test database | Analysis parameters | Analysis results
Analyse database
Transfer database...
Close

Result files directory	
Result files prefix	KMean_
Preparation report	PreparationReport.xls
Modeling dictionary file	Modeling.kdic
Modeling report	ModelingReport.xls
Train evaluation report	TrainEvaluationReport.xls
Test evaluation report	TestEvaluationReport.xls

Result files directory: name of the directory where the results files are stored (default: empty). By default, the results files are stored in the directory of the train database. If a result directory is specified, it can be:

- an absolute path (example "c:\project\scenario1"): the results files are stored in this directory
- a local path (example "scenario1"): the results files are stored in a sub-directory of the train database directory
- a relative path (example ".\scenario1"): the results files are stored in a sub-directory of current directory (Khiops Enneade executable start directory)

Result files prefix: (default: empty). This prefix is added before the name of each result file.

The following result file names allow to specify the name of each report or model resulting from an analysis. When a file name is missing, the corresponding report is not produced.

Preparation report: name of the data report file produced at the end of the univariate data analysis on the train database (default: PreparationReport.xls).

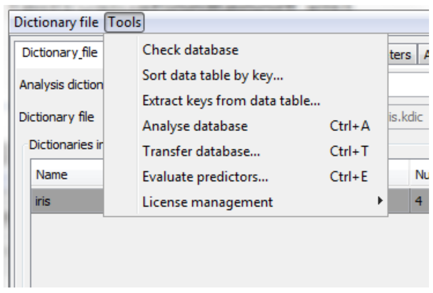
Modeling dictionary file: name of the report file produced once the predictors have been built (default: ModelingReport.xls).

Modeling report: name of the report file produced once the predictors have been built (default: Modeling.kdic).

Train evaluation report: name of the report file produced at the end of the evaluation of the predictors on the train database (default: TrainEvaluationReport.xls).

Test evaluation report: name of the report file produced at the end of the evaluation of the predictors on the test database (default: TestEvaluationReport.xls).

2.6. Tools menu



2.6.1. Check database

Prerequisite: the train database must be specified, and the dictionary related to the train database must be loaded in memory.

This action checks the integrity of the train database.

This action reads each line of the train database to perform the integrity checks. During formatting checks, the number of fields in the train database is compared to the number of variables in the dictionary. Data conversion checks are performed for the fields corresponding to numerical, date, time and timestamp variables. Error messages are displayed in the message log window.

Remark: errors during database checking are always displayed, but they are autocorrected (empty or erroneous numerical values are replaced by a system missing value, superfluous values are discarded). Therefore, a data analysis can always be performed, even though it might not be reliable in case of database errors.

2.6.2. Analyse database

Prerequisite: the name of the train dataset must be specified, and the dictionary related to the train database must be loaded in memory.

An analysis report describing the univariate statistics is then produced, as well as new dictionaries related to data preparation or to the built predictor. The built dictionary is :

- KM_<Dic>: dictionary containing the membership prediction, prediction score formulae, …, for a K-Means predictor, …

A modeling report summarizes the features of the built predictors.
An evaluation report is also produced on the train and test datasets.

<u>The predictor dictionaries</u>

At the end of the data analysis, Khiops Enneade builds a predictor and saves it by means of dictionaries including variables dedicated to prediction. The formulae used to compute prediction variables are stored in the dictionaries, enabling the deployment of prediction on unseen data. The data miner can select or unselect variables to deploy using the "Unused" keyword in the modelling dictionary. For example, to produce a score file, the data miner can select a key variable, in order to enable joins in databases, and the variable related to the probability of the class value of interest.

The main output variables in a dictionary are:

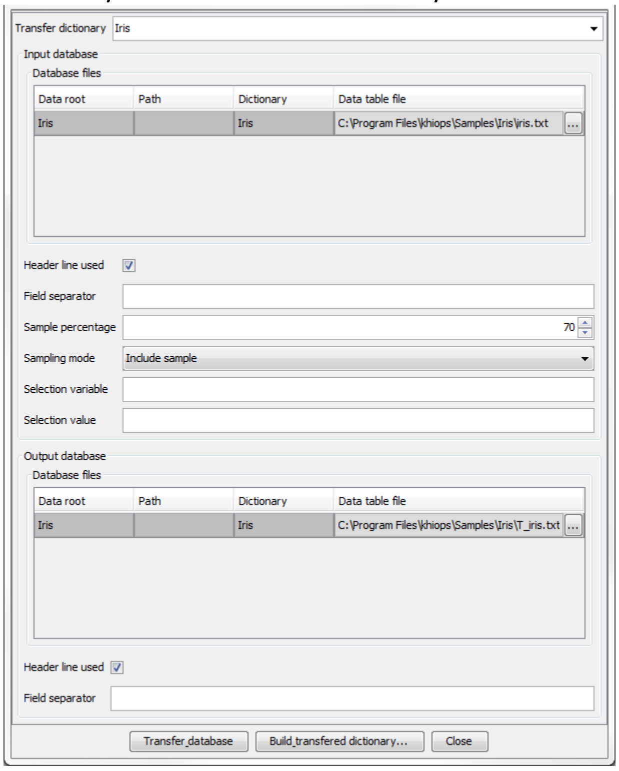
- DistanceCluster<idcluster>_L<norm>: Distance of the instances to each center <idcluster> using the L<norm> (<norm>=1, 2 or Cos). If L2 norm is used, then this distance is the squared distance.
- IdCluster: id of the cluster membership of the instance

If a target variable has been specified, with a target variable named <class>:

- Predicted<class>: predicted value
- Score<class>: score of the prediction
- Prob<class><value>: probability of each target value named <value>

2.6.3. Transfer database

Prerequisite: at least one dictionary must be loaded in memory.


Transfer dictionary: Iris

Input database

Database files

Data root	Path	Dictionary	Data table file
Iris		Iris	C:\Program Files\khiops\Samples\Iris\iris.txt

Header line used: ✓
Field separator:
Sample percentage: 70
Sampling mode: Include sample
Selection variable:
Selection value:

Output database

Database files

Data root	Path	Dictionary	Data table file
Iris		Iris	C:\Program Files\khiops\Samples\Iris\T_iris.txt

Header line used: ✓
Field separator:

This action opens a dialog box allowing to specify an input and output database, and a dictionary describing the variables to keep, discard or derive. The parameters of the dialog box are the following.

Transfer dictionary: dictionary used to select or derive new variables.

Input database:

- Data table file
- Header line used
- Field separator
- Sample percentage
- Discard mode
- Selection variable
- Selection value

Output database:

- Data table file
- Header line used
- Field separator

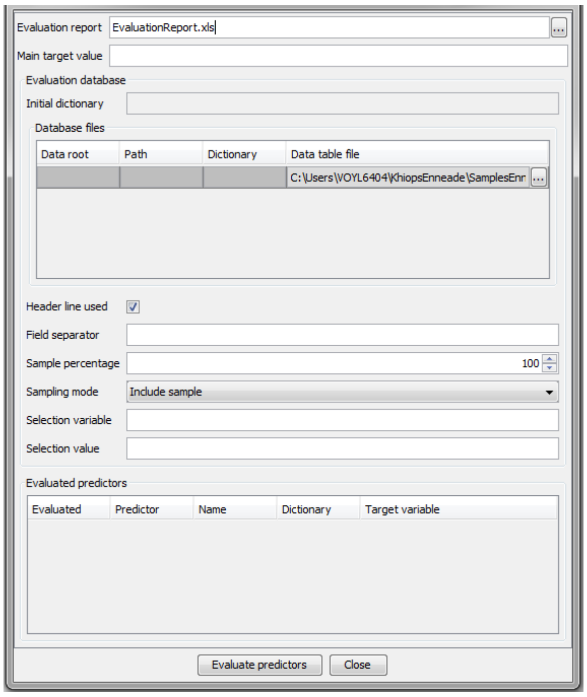
The "**Transfer database**" action reads the input data, applies the transfer dictionary to select all or part of the variables and add derived variables, and writes the output data.

This action can be used to generate a data preparation file, containing the recoded variables. It also can be used to deploy a scoring model, owing to the prediction variables contained in the predictor dictionaries.

The "**Build transferred dictionary**" action creates an output dictionary that enables to read and analyse the transferred file: it contains the transferred variables only, without any derivation rule in the dictionary.

2.6.4. Evaluate predictors

Prerequisite: at least one dictionary must be loaded in memory, and correspond to a predictor dictionary.



Evaluation report	EvaluationReport.xls
Main target value	

Evaluation database

Initial dictionary	

Database files

Data root	Path	Dictionary	Data table file
			C:\Users\VOYL6404\KhiopsEnneade\SamplesEnr

Header line used	✓
Field separator	
Sample percentage	100
Sampling mode	Include sample
Selection variable	
Selection value	

Evaluated predictors

Evaluated	Predictor	Name	Dictionary	Target variable

This action opens a dialog box allowing to an evaluation report, an evaluation database and to choose the predictor(s) to evaluate. The parameters of the dialog box are the following.

Evaluation report: name of the report file

Evaluation database:

- Data table file
- Header line used
- Field separator
- Sample percentage
- Discard mode
- Selection variable

- Selection value

Evaluated predictors: List of the predictor dictionaries, which are dictionaries among the loaded dictionaries that are recognized as predictor dictionaries. This array allows to choose (parameter "Evaluated" which predictor to evaluate).

- Evaluated: to choose whether to evaluate the predictor
- Predictor: Classifier
- Name: label of the predictor
- Dictionary: name of the predictor dictionary
- Target variable: name of the target variable

The "**Evaluate predictors**" applies the evaluated predictors on the evaluation database and writes an evaluation report. Whereas the "Analyse database" builds predictors and evaluates them immediately on the train and test databases, the "Evaluate predictor" action allows a differed evaluation of previously built predictors.

3. Inputs-outputs

3.1. Format of the database files

A database file is a text file, containing one line per record. By default, the first line contains the names of the variables. If no header line is used, the fields in the database file must appear in the same order as the variables in the related dictionary.

The values of the variables are separated by a field separator. The field separator is tabulation by default (empty), and can be space (S), semi-colon (;), comma (,) or any character.

Fields can contain separator chars provided that they are surrounded by double-quotes:

- any field can be surrounded by surrounded by double-quotes (e.g. *”city”* for *city*),
- for fields which content is surrounded by double-quotes:
  - the separator char can be used inside the field (e.g. *“NY, city” for NY, city*)
  - the double-quote can be used if it is paired (e.g. *“””NY””, city”* for *”NY”, city*),
  - the end of line character cannot be used inside a field (multiple-line fields are not allowed).

The numerical values may use the scientific notation (for example: 1.3E7). The decimal separator can be either the dot or the comma (the commas are recoded into dots). Missing or erroneous numerical values are replaced by a missing system value (–infinity, to avoid collision with any valid value).

Tabulations inside categorical values are replaced by blank characters, since they raise problem in visualisation tools that are base on text files with tabulation separated fields. The special char Ctrl-Z (ascii 26) is also replaced by a blank character. Space characters are discarded at the beginning and end of categorical values.

Date values are stored using the *YYYY-MM-DD* format, Time values using the *HH:MM:SS.* format and Timestamp values using the *YYYY-MM-DD HH:MM:SS.* format. Numerous other formats are available (see appendix). For these formats, the variable must be declared with a meta-data (with key *DateFormat*, *TimeFormat* or *TimestampFormat*) to specify the external format (see Paragraph 3.2.2. Dictionary).

3.2. Format of the dictionary files

3.2.1. Dictionary file

A dictionary file is a text file with extension .kdic, containing the definition of one or many dictionaries.

3.2.2. Dictionary

A dictionary allows to define the name and type of native variables in a data table file, as well as the constructed variables described by means of derivation rules.

```
Dictionary name [meta-data] [// label]
{
 {Variable definition}*
};
```

A dictionary is defined by its name and by the list of its variables, and optionally meta-data and a label.

Meta-data is a list of keys or key value pairs (*<key>* or *<key=value>* for numerical or categorical constant values). Meta-data is used internally by Khiops Enneade to store information related to dictionaries or variables (to annotate the results of analysis). It is also used to store the external format of Date, Time and Timestamp variables, in case where the default format is not used. In the following example, the three predefined meta-data keys *DateFormat*, *TimeFormat* and *TimestampFormat* are used to specify the input and output format of the related variables:

```
Date MyDate     ; <DateFormat="DD/MM/YYYY">
Time MyTime     ; <TimeFormat="HH.MM">
Timestamp       MyTimestamp        ; <TimestampFormat="YYYY-MM-DD_HH:MM:SS">
```

Each variable is defined by its type, its name and other optional information.

```
[Unused]   type    name   [= derivation rule];      [meta-data]     [// label]
```

A variable can be ignored in the data processing (memory loading, analysis, transfer) if the keyword *Unused* is specified before the variable definition. Even though, Khiops Enneade is still aware of the variable, which allows to construct new variables derived from the ignored variable.

The types are Categorical, Numerical, Date, Time or Timestamp for native variables.

The names are case sensitive and are limited to 128 characters. In the case where they use characters other than alphanumeric characters, they must be surrounded by back-quotes. Tabulations are not allowed inside variables names (replace by blank characters). Back-quotes inside variable names must be doubled.

Derivation rules are formulas that allow to compute the value of variable from other values coming from other variables, rules, or constants.

Each line in the definition of a dictionary can be commented, using "//" as a prefix.

Some technical types are used by Khiops Enneade to specify prepocessing or modeling methods: for example Structure(DataGrid), Structure(Classifier).

Example: dictionary file Iris.kdic with a constructed variable PetalArea

```
Dictionary Iris
{
  Unused    Numerical          SepalLength;
            Numerical          SepalWidth;
		Numerical	PetalLength;
		Numerical	PetalWidth;
		Numerical	PetalArea   = Product(PetalLength, PetalWidth);
		Categorical	Class;			// Class variable
};
```

3.2.3. Edition of dictionary files by means of Excel

The dictionary files, which are text files with tabulation separators, could be easily edited using Excel. Unfortunately, the use of derivation rules or categorical constants (surrounded by double quotes) is error prone in Excel (due to automatic data conversion in Excel). However, Excel can be used safely with the following process:

- Open the dictionary file using a basic text editor (for example: Notepad),
- Copy-paste all or part of the defined variables to Excel,
- Edit the variables in Excel (select, modify, sort…),
- Copy-paste of the edited variables back to the text editor.

Editing the variables using Excel allows to display the variables properties (Unused keyword, User type, Type, Name, Derivation rule, Comment, Level…) in Excel columns. This is then easy to perform sorts and modify the definition of variables.

3.2.4. Derivation rules

The derivation rules allow to construct new variables in a dictionary. The operands in a derivation rule can be a variable (specified by its name), a constant numerical or categorical value, or the result of another derivation rule. The derivation rules can be used recursively.

A constant categorical value must be surrounded by double quotes. A double quote character inside a categorical value must be doubled. When the length of a categorical value is too important, the value can be split into subvalues, concatenated using '+' characters.

A constant numerical value can be specified using scientific notation (for example: 1.3E7). The decimal separator is the dot. The missing value is represented as #Missing when used in a derivation rule.

There are no Date, Time Timestamp constants, but they can be produced using conversion rules (see appendix: e.g. *AsDate("2014-01-15", "YYYY-MM-DD")*);

The list of available derivation rules is given in appendix.

3.3. Reports

3.3.1. Preparation report

The preparation report is a tabulated text file with file extension .xls, directly editable using Excel, Word, Notepad...

It contains in a header a recall of the main parameters of the data analysis problem, general statistics related to the train database, and a synthetic array of univariate statistics for the categorical and numerical variables, sorted by decreasing predictive importance.

The remainder of the report consists of contingency tables for the discretized or grouped variables, also sorted by decreasing predictive performance. These detailed statistics are reported for informative variables only.

The fields in the synthetic array of the univariate statistics are defined below.

General evaluation of the variable	
Rank	Rank of the variable, sorted by decreasing importance. This rank is also a convenient identifier, which eases search operations in report files.
Name	Name of the variable
Level	Evaluation of the predictive importance of the variable. The Level is a value between 0 (variable without predictive interest) and 1 (variable with optimal predictive importance).
Groups/Intervals	Number of groups/intervals resulting from the discretization/grouping preprocessing of the variable.
Values	Number of initial values of the variable.
Fields specific to categorical variables	
Mode	Most frequent initial value.
Mode coverage	Coverage of the mode.
Fields specific to numerical variables	
Min	Min value of the variable.
Max	Max value of the variable.
Mean	Mean value.
Std dev	Standard deviation.
Missing number	Number of missing values
Analysis results on variables (uncommented: internal use)	
Constr.cost	
Prep. cost	
Data cost	
User information on variables (optional fields)	
Derivation rule	Derivation rule used to compute the variable. User defined, or automatically constructed by the variable construction functionality of Khiops Enneade.

3.3.2. Modeling report

This report is available if predictors have been built.

It summarizes the predictor, with the name and number of used variables.

Predictor details are reported as well.

For the KMeans preditors:

- EVA obtained by the best replicate (EVA = criterion described in equation 14 of “Bayesian instance selection for the nearest neighbor rule”. S. Ferrandiz, M. Boullé. Machine Learning, 81(3):229-256, 2010.)
- Mean Squared distance
- Davies-Bouldin indexes, by attributes
- Centroids (virtual centers): the centroids found by the K-Means algorithm using the variables after the pretreatment step.
- Initial centroids (virtual centers) found before convergence, just after the initialization step.
- Real instances (nearest to centroids) : The instances nearest to centroids using the variables after the pretreatment step.
- Real native instances (nearest to centroids) : The instances nearest to centroids using the variables before the pretreatment step.

- Preprocessing and clustering levels (uncommented: internal use)

3.3.3. Evaluation report

This report is available if a train or test database has been specified and if predictors have been built.

Note: The fields with a (*) are filled only if a Target Variable has been specified.

Note: The fields with a (**) are filled only if a Target Variable has been specified AND all the preprocessing have been selected as "Automatically computed".

Note: The fields with a (☐) are filled only if all the preprocessing have been selected as "Automatically computed".

<u>Predictors performance:</u>

- (*) Accuracy: evaluates the proportion of correct prediction
- (*) Compression: evaluates the predicted target probabilities using a negative log likelihood approach and is normalized (between 0 and 1) using the baseline predictor
- (*) AUC: area under the ROC curve (AUC); evaluates the ordering of the predicted scores per target value.

<u>Confusion matrix:</u>

- (*) A confusion matrix is reported for each classifier, to compare the predicted values (prefixed by $) and the actual values.

<u>K-Means statistics:</u>

These are the metrics corresponding to the best replicate. They will always appear in the evaluation report :

- Mean squared distance
- Davies-Bouldin index
- Ratio : inter clusters inerty / total inerty
- Adjusted Rand Index by clusters
- Predictive Clustering Index (not described here)

<u>Table "Gravity centers ", for every cluster, from the left column to the right:</u>

- Cluster: id of the cluster
- Frequency: Number of instances belonging to the cluster
- Coverage: Percentage of instances belonging to the cluster
- (*) C columns (one column per Class if a Target Variable has been specified):
    - Percentage of the instances belonging to the cluster 'id' and of the Class '1'

  - Percentage of the instances belonging to the cluster 'id' and of the Class '2'
  - …
  - Percentage of the instances belonging to the cluster 'id' and of the Class 'C'

(**) Table "Mean values for Numerical attributes":
for every numerical explanatory variable used after the preprocessing step

- Column 1 - Var name: Name of the variable
- Column 1+1 to 1+k (k is the number of clusters):
  - Mean values of the variable of the Column 1 ("Var name") for the instances (which have not a missing value for THIS variable) belonging to "Cluster k"
- Column "global": Mean values of the variable of the Column 1 ("Var name") for all the instances (which have not a missing value for THIS variable)
- Column "missing" indicates the percentage of missing values for the variable of the Column 1 ("Var name")

(**) Table "Median values for Numerical attributes":
for every numerical explanatory variable used after the preprocessing step
Note : when computing this table if the available memory is not sufficient the median values are approximated on a percentage of the dataset. In this case the title of the table gives the percentage used.

- Column 1 - Var name: Name of the variable
- Column 1+1 to 1+k (k is the number of clusters):
  - Median values of the variable of the Column 1 ("Var name") for the instances (which have not a missing value for THIS variable) belonging to "Cluster k"
- Column "global": Median values of the variable of the Column 1 ("Var name") for all the instances (which have not a missing value for THIS variable)
- Column "missing" indicates the percentage of missing values for the variable of the Column 1 ("Var name")

(**)

(☐) Table "Native attributes proba":
for every explanatory variable used after the preprocessing step, from the left column to the right

- Column 1 - Var name: Name of the variable
- Column 2 - Modality/Interval: Intervals or Group name
- Column 2+1 to 2+k (k is the number of clusters):
  - Percentage of instances belonging to "Cluster k" for which the variable of the Column 1 ("Var name") takes its values in the Intervals or Group name of the column 2 ("Modality/Interval") – the percentage is computed over all Intervals or Group name of the variable of the Column 1 ("Var name").
- Column "global"
  - Percentage of instances (whatever the cluster) for which the variable of the Column 1 ("Var name") takes its value in the Interval or Group name of the column 2 ("Modality/Interval")

(☐) Table "Percentage per line - Native attributes proba":
for every explanatory variable used after the preprocessing step, from the left column to the right

- Column 1 - Var name: Name of the variable
- Column 2 - Modality/Interval: Intervals or Group name
- Column 2+1 to 2+k (k is the number of clusters):
  - Percentage of instances belonging to "Cluster k" for which the variable of the Column 1 ("Var name") takes its values in the Intervals or Group name of the column 2 ("Modality/Interval") - the percentage is computed over all the clusters.
- Column "global"
  - Always equal to 1 per definition

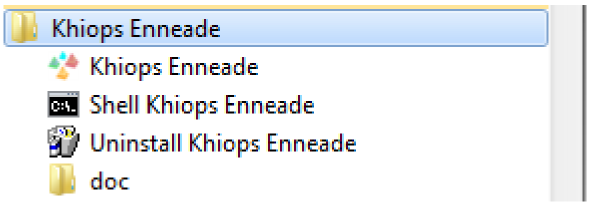
(*) Lift-curves:

- If a main class value is specified, lift curves are built. They plot the fraction of actual correct prediction on the y-axis for each fraction of the dataset on the-x-axis, when the dataset is sorted by decreasing predicted probability of the main target value.

4. Batch mode

4.1. Register and replay a batch scenario

For a recurrent use of Khiops Enneade, it is interesting to register a scenario in a command file. Such a command file can easily be modified using a text editor, and then replayed by Khiops Enneade in batch mode.



To run Khiops Enneade, open a Shell khiops Enneade, then type KhiopsEnneade (-h for batch mode options with ability to record or replay a scenario).

To register a scenario, use the –o option on the command line.

KhiopsEnneade –o scenario._kh

All the actions or modifications of field values in the graphical user interface are registered in the command file scenario._kh.

It is a good practise to create a working directory for each Data Mining project. This directory contains all the files for databases, dictionaries, scenarios and reports.

Remark: by default, Khiops Enneade keeps a command file named scenario._kh for the last use of the tool, in the "lastrun" subdirectory of the installation directory.

The following scenario corresponds to a use of Khiops Enneade with the Iris sample. The scenario opens the Iris dictionary and performs a classification analysis of the database. To replay this scenario:

- install the scenario file (named iris._kh) in the Iris directory
- Open a Shell Khiops Enneade and go to Iris directory
- Start KhiopsEnneade using –i on the command line

KhiopsEnneade –i iris._kh

Scenario file iris._kh

```
// -> Khiops Enneade
ClassManagement.OpenFile      // Open...

// -> Open
ClassFileName C:\Program Files\khiops\Samples\Iris\Iris.kdic  // Dictionary file
OK                   // Open
// <- Open

TrainDatabase.DatabaseFiles.List.Key Iris        // List item selection
TrainDatabase.DatabaseFiles.DataTableName C:\Program Files\khiops\Samples\Iris\iris.txt    // Data table file
TrainDatabase.SampleNumberPercentage 70  // Sample percentage
TestDatabase.ImportTrainDatabaseSettings   // Import train database settings
AnalysisSpec.TargetAttributeName Class   // Target variable
AnalysisResults.ResultFilesDirectory ClassificationAnalysis  // Result files directory
ComputeStats               // Analyse database
Exit                 // Close
// <- Khiops Enneade

// -> Khiops Enneade
OK                   // Close
// <- Khiops Enneade
```

Warning: the two last "Close" actions must be commented to replay the scenario and remain in the graphical user interface of Khiops (otherwise, all the Khiops Enneade session is replayed in batch mode, even the close action that exits from Khiops).

4.2. Integration in a program to industrialize a data analysis process

Khiops Enneade can easily be called from a program written in any computer language, such as C, C++, Java, or Python, in order to build an industrial data analysis process.

You first have to prepare the input data, dictionary file and datasets, write a scenario file in which you set up all needed parameters, mainly the location of the input and output data files and the data analysis operations to be performed, and finally call khiops (in bin directory) by program using the command line options.

4.3. List of command line options

```
KhiopsEnneade [-i <input command file name>] [-o <output command file name>]
  [-e <error log file name>] [-t <task progression log file name>]
  [-r <searched string>:<replace string>]...
  [-b] [-v] [-h]
    -i: commands to replay
    -o: to record commands
    -e: to store error messages
    -t: to store last task progression messages
    -r: zero to many search-replace applied to input command file
    -b: batch mode, with no log window
```

-v: verbose mode during command replay
-h: help

4.4. Technical limitations

Numerical precision:

- in memory, the numerical values are stored on 8 bytes, with an exponent between 10^{-100} and 10^{100}, and a mantissa of 10 significant digits,
- risk of loss of numerical precision when using some derivation rules (for example, adding a very small value and a very large value),
- risk of loss of numerical precision during a database transfer if input values use a mantissa of more than 10 digits.

Management of categorical values:

- in memory, the categorical values are stored using zero-terminated strings, which is compatible with ANSI files, UTF-8 files (at least in batch mode; the Java interface may not work), but not with double-bytes character sets such as Unicode,
- categorical values are kept at most once in memory,
- for example, a variable "address" of average length 100 characters and containing 10000 different values will requires about 1 Mo RAM,
- risk of lacking from RAM in case of categorical variables containing large numbers of different values of large size (for example: managing 100000 different values of average length 10000 characters requires about 1 Go RAM).

4.4.1 Preprocessing before the elaboration of the clustering models

No known limitation whatever the “preprocessing” used (see the list of avalaible preprocessings Section 2.4.1.1) EXCEPT if no column of the training dataset fits into memory (in this case a message will appear in the log window).

Note : That means that the “statitics” used to preprocess the dataset before the elaboration of the clustering models are computed using all the training examples (even if, after, the clustering models is elaborated using a percentage of the dataset see Section 4.4.2).

4.4.2 Elaboration of the clustering models

In this actual version the dataset (after preprocessing) has to fit in memory to be able to train the clustering model.

If the database do not fit in memory the log window indicates the percentage of the database used to train the model (see Section 4.4.5)

Computation time for training the K-Means predictor: each replicate take less than a second, on the “Adult” dataset (PC 64 bits / 12 Go RAM)

4.4.3 Evaluation of the clustering models

No known limitation all indicators and the tables included in an Evaluation report are computed “incrementatly” (the databases are processed one record at a time.) **EXCEPT**:

- The table “Median values for Numerical attributes” when computing this table if the available memory is not sufficient the median values are approximated on a percentage of the dataset. In this case the title of the table gives the percentage used (see Section 4.4.5).
    - This table is computed only if check box “write detailed statitics” is activated (by default activated, see Section 2.4.1.1)

4.4.4 Deployment of the clustering models

Limitation of database transfer:

no known limitation: the databases are processed one record at a time.

4.4.5 Memory overflow

In spite of conservative evaluation of required memory, Khiops Enneade may crash down with memory overflow, during the data preparation phase. In this case, a “memory overflow” message is present in the tool log file. Asking Khiops Enneade to use less memory (see “System parameters”) is likely to solve this problem. No crash is expected during the training and evaluation phase : again, a conservative memory evaluation is done, depending on the various K-Means parameters (number of loaded attributes, supervised mode or not, preprocessing parameters…). If this memory evaluation exeeds the available memory, then Khiops Enneade will be computed only on a percentage of the database (i.e., not all database lines will be used, but all attributes will be kept)

4.4.5 Known bugs

When using the “simplified modelling” option, Khiops Enneade may crash down if required memory exceeds available memory. This is a known bug which will be corrected in future versions. This bug could appear only for very large dataset (very large number of instances AND very large number of explanatory variables) for which the tool is not really dedicated (since it is close to explanatory analysis).