EDM User Manual

Educational Data Mining Workbench Manual V4.0

Content

Revision History

Name	Date	Reason for Change	Version
John Paul Contillo	20111121	First draft	V1.00
Alipio Gabriel	20111122	Edit the context of the draft	V1.00
Alipio Gabriel	20111123	Add and edit the content	V1.00
J.Contillo	20120221	User manual for version 2	V2.00
Gamaliel dela Cruz	20120526	Edit content	V3.00
Francis Bautista	20120607	Formatting and editing	V3.00
John Paul Contillo	20111121	Content Addition	V3.10
Francis Bautista	20120728	Formatting and editing	V3.20
Nadia Leetian	20120814	Edit content	V3.50
Dominique Isidro	20120821	Edit content	V3.51
Francis Bautista	20121013	Addition of content	V3.52
Francis Bautista	20121103	Addition of content	V3.53
Francis Bautista	20130214	Addition of content	V4.00

Introduction

In recent years, educational data mining methods have afforded the development of detectors of a range of constructs of educational importance, from gaming the system [3] to off-task behaviour [2] to motivation [5] to collaboration and argumentation moves [6]. The development of these detectors has been supported by the availability of machine learning packages such as RapidMiner [7], WEKA [9], and KEEL [1]. These packages provide large numbers of algorithms of general use, reducing the need for implementing algorithms locally, however they do not provide algorithms specialized for educational data mining, such as the widely used Bayesian Knowledge-Tracing [4]. Furthermore, effective use of these packages by the educational research and practice communities presumes that key steps in the educational data mining process have already been completed. For example, many of these detectors have been developed using supervised learning methods, which require that labelled instances, indicative of the categories of interest, be provided. Typically, many labelled instances – on the order of hundreds, if not thousands – are required to create a reliable behaviour detector. Labelling data is a time consuming and laborious task, made even more difficult by the lack of tools available to support it.

A second challenge is the engineering and distillation of relevant and appropriate data features for use in detector development [9]. The data that is directly available from log files typically lacks key information needed for optimal machine-learned models. For instance, the gaming detectors of both [3] and [8] rely upon assessments of how much faster or slower a specific action is than the average across all students on a problem step, as well as assessments of the probability that the student knew the cognitive skills used in the current problem step. This information can be distilled and/or calculated by processing data across an entire log file corpus, but there are currently no standard tools to accomplish this. Feature distillation is time-consuming, and many times a research group re-uses the same feature set and feature distillation software across several projects (the second author, for instance, has been using variants of the same feature set within Cognitive Tutors for nine years). Developing appropriate features can be a major challenge to new entrants in this research area. To address this "data labeling bottleneck" and the difficulty in distilling relevant features for machine learning, we are developing an *Educational Data Mining (EDM) Workbench*. A beta version of this Workbench, now available online at http://penoy.admu.edu.ph/~alls/downloads, is described in this user manual. The Workbench currently allows learning scientists to:

1) Label previously collected educational log data with behaviour categories of interest (e.g. gaming the system, help avoidance), considerably faster than is possible through previous live observation or existing data labelling methods.

2) Collaborate with others in labelling data.
3) Automatically distil additional information from log files for use in machine learning, such as estimates of student knowledge and context about student response time (i.e. how much faster or slower was the student's action than the average for that problem step).

Through the use of this tool, we hope that the process of developing a detector of relevant metacognitive, motivational, engagement, or collaborative behaviours can eventually be sped up. Just the use of "text replays", on previously collected log data has been shown to speed a key phase of detector development by about 40 times, with no reduction in detector goodness [3].

This user manual is intended as a guide to the functions and features of the EDM Workbench. Please send comments and suggestions to mrodrigo@ateneo.edu.

Definition of Terms

Batch
A group of log files. The criteria for grouping are determined by the user.
Examples of the criteria for grouping include source and timing

Clip
A subset of logs from a given batch

Column
A single attribute within the dataset

Dataset
The data from the imported files

DataGrid
The central area where all the datasets are displayed.

EDM
Educational Data Mining

Log
A record of a single action

Log File

A file that contains a collection of logs

Model

A detector of meta-cognitive and motivational behaviour

Row

A set of attributes in the dataset that usually refers to 1 log

Interface

Refers to the system graphical user interface

Overall Description

The EDM Workbench is a tool that helps researchers with processing data from various sources for developing meta-cognitive and behavioural models. The concept diagram in figure 1 illustrates the system functionalities and entities interacting with it.

The EDM Workbench's functions allow users to:

- Define and modify behaviour categories of interest
- Label previously collected educational log data with the categories of interest considerably faster than current methods
- Collaborate with others in Labelling data by providing ways to communicate and document Labelling guidelines and standards
- Validate inter-rater reliability between multiple labellers of the same educational log data corpus
- Automatically distil additional information from log files for use in machine learning
- Export student behaviour data to tools which enable sophisticated secondary analysis

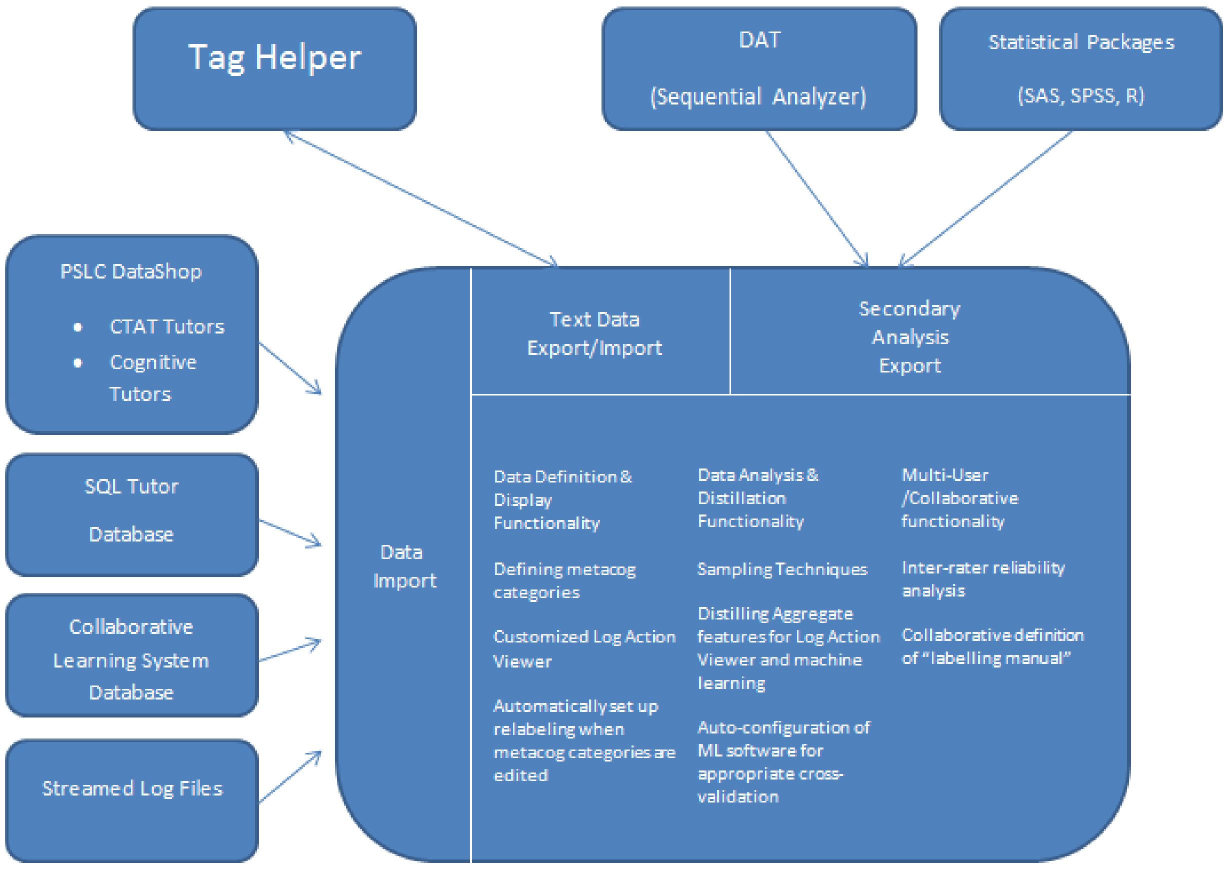

Tag Helper
DAT
(Sequential Analyzer)
Statistical Packages
(SAS, SPSS, R)
PSLC DataShop
CTAT Tutors
Cognitive Tutors
SQL Tutor Database
Collaborative Learning System Database
Streamed Log Files
Data Import
Text Data Export/Import
Secondary Analysis Export
Data Definition & Display Functionality
Defining metacog categories
Customized Log Action Viewer
Automatically set up relabeling when metacog categories are edited
Data Analysis & Distillation Functionality
Sampling Techniques
Distilling Aggregate features for Log Action Viewer and machine learning
Auto-configuration of ML software for appropriate cross-validation
Multi-User /Collaborative functionality
Inter-rater reliability analysis
Collaborative definition of "labelling manual"

Figure 1: EDM Workbench Entity Diagram

Overall Use Cases

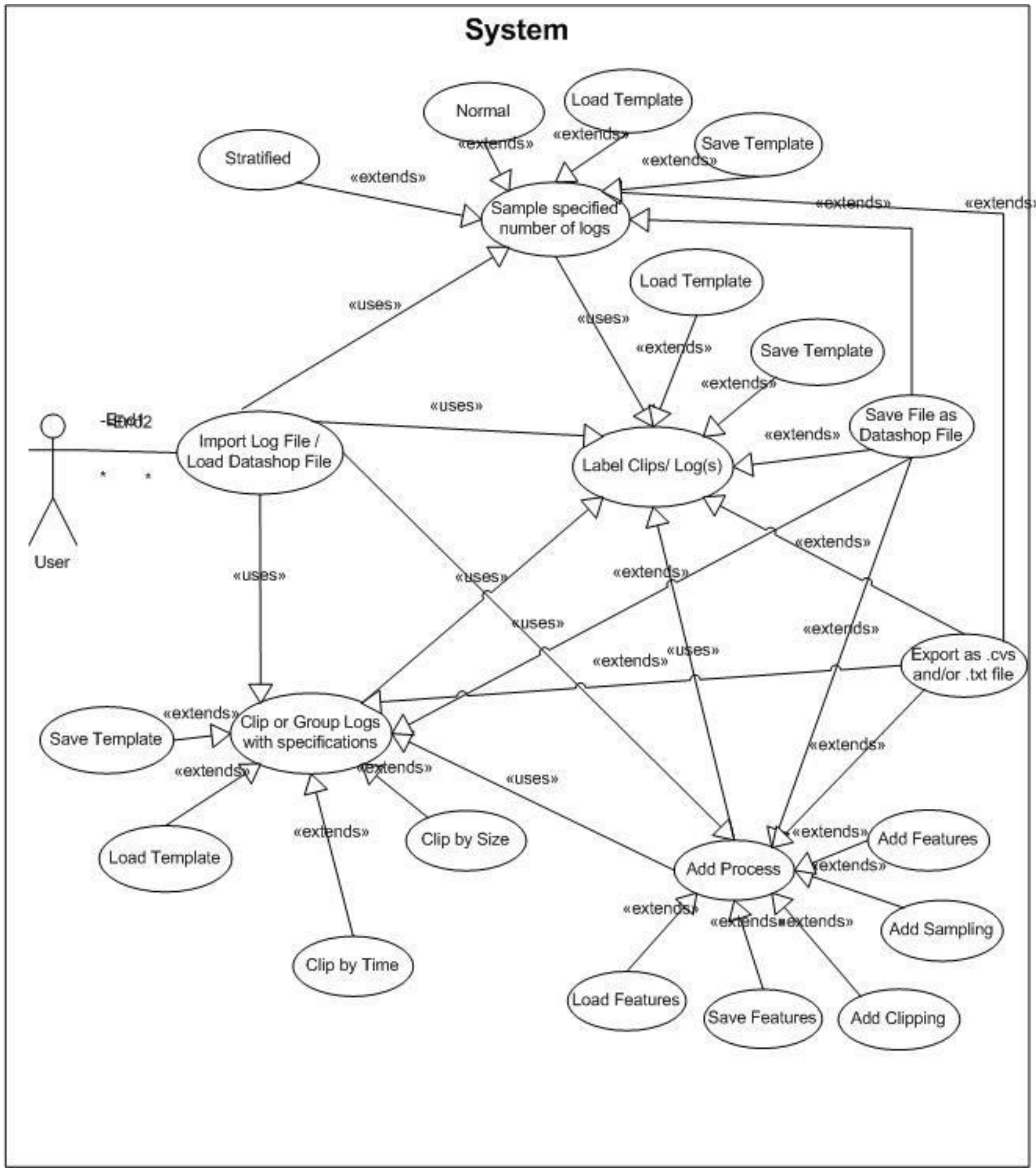


Figure 2: EDM System Process Map

Chapter 1. System Overview

This section, discusses the interface of the system (from Top to Bottom) including its features, buttons, and functions.

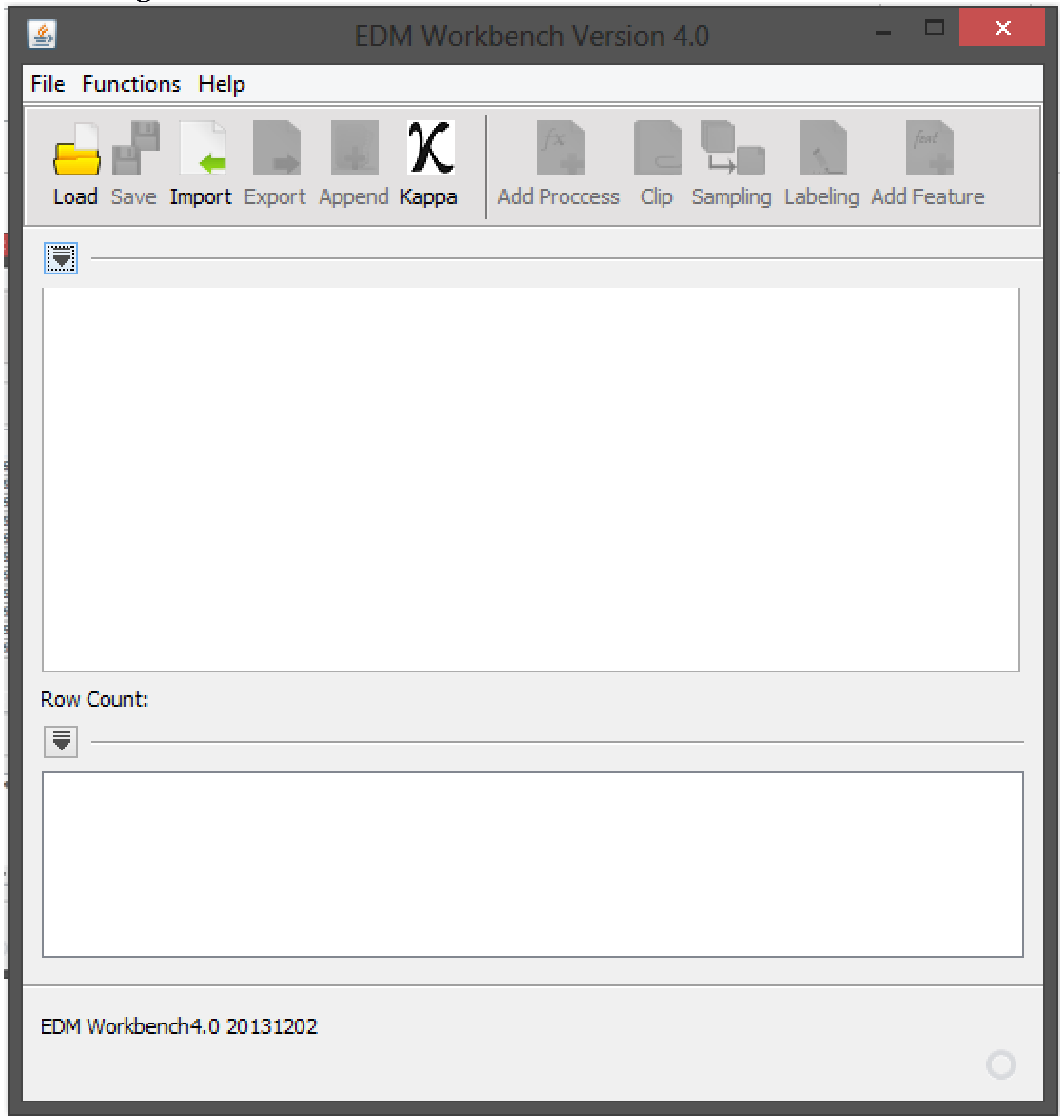


Figure 3: EDM workbench upon system launch

Title Bar

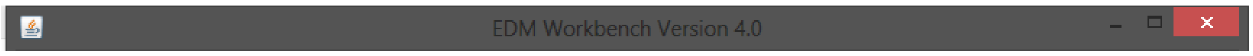
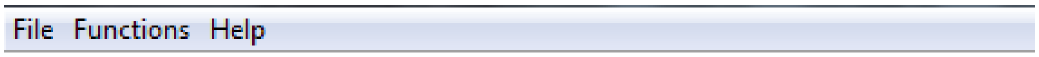


Figure 4: System Title Bar

The name of the system (may change in later versions e.g. EDM Workbench version (4.0) is displayed here.

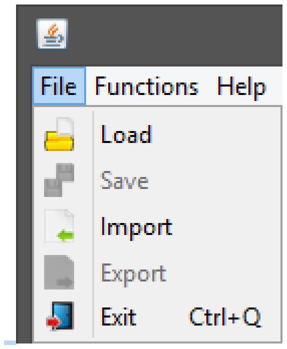
▪ Menu Bar



▪ Figure 5: EDM Menu Bar

Composed of 3 Menu options (File, Functions, and Help) consisting of actions buttons.

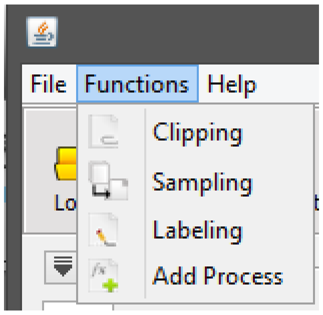
○ File Menu



The **File Menu** is composed of 5 actions (Load, Save, Import, Export and Exit) that handle the files and logs to be displayed and/ or saved in the DataGrid.

Figure 6: File Menu Dropdown

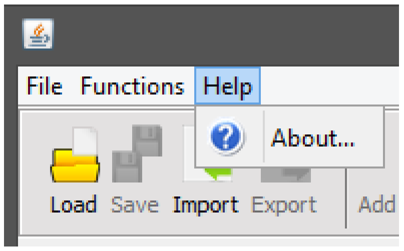
○ Function Menu



The **Function Menu** consists of 4 log processing actions that will either be enabled or disabled depending on the state of the system.

Figure 7: EDM Function menu Dropdown

○ Help Menu



The **Help Menu** contains the "About" action that displays the system description and the current product version (e.g. 20120227).

Figure 8: EDM Help Menu showing the About button

▪ Tool Bar

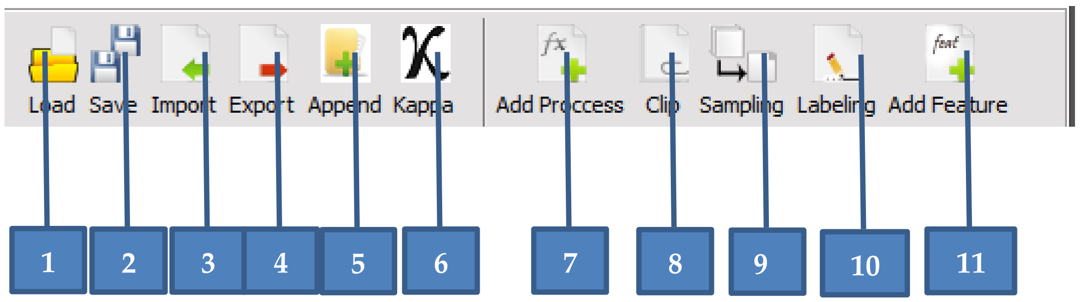


Figure 9: EDM Toolbar with activated buttons

The **Tool bar** is composed of action buttons that are also found in the menu bar for ease of use.

1. Load Button

Loads log files which were previously saved using the EDM Workbench and stored in an EDM Workbench-specific.zip file. The file contains logs that may have been previously processed, clipped, sampled, or labelled by the user together with some Workbench-specific information. Note that, because of the additional information, the zip file may not be opened using archiving software such as WinZip or WinRar. Once loaded, the user may make further changes to the file.

2. Save Button

Saves the logs from the active tab in the DataGrid and all its properties such as clipped formats and labels into EDM format.

3. Import Button

Allows the user to import logs or batches of logs such as Datashop or comma-separated value(.csv files) to be processed, clipped, sample or labelled by the user.

4. Export Button

Exports the final output from the active tab in the DataGrid as a CSV file or in other specified file formats.

5. Append Button

Appends a dataset (csv/txt) to the current dataset as displayed in the DataGrid. The data sets must have the same column names for this function to work.

6. Kappa Button

Compares the level of agreement between two separate data sets of the same file type. Operation returns the integer 1 if the data sets agree with each other perfectly, and 0 if they do not match at all. A decimal returned shows incomplete agreement between the data sets; however a value closer to one is "more true" than a value closer to zero.

7. Add Process Button

Allows the user to add and possibly save an action to a sequence of actions.

8. Clip Button

Groups logs from a given batch based on user-specified parameters.

9. Sampling Button

Selects rows from the dataset based on user parameters.

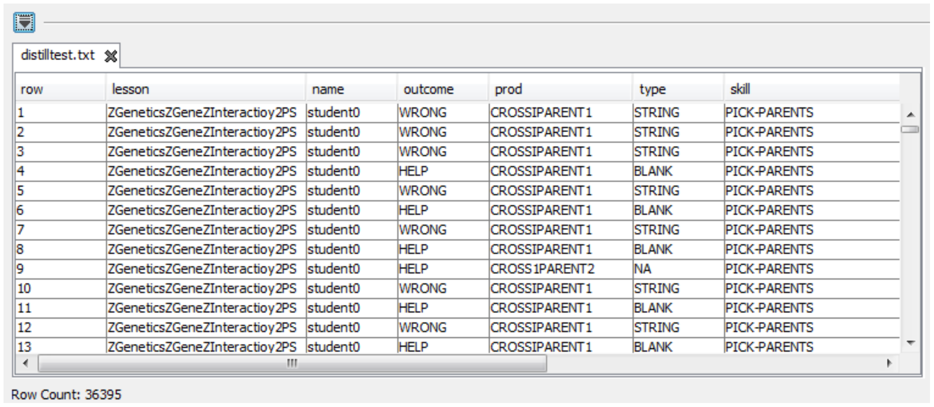
10. Labelling Button

Allows the user to supply "ground truth" labels for clip

11. Add Feature

Allows the user to tailor functions to their specification.

▪ DataGrid



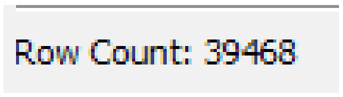
distilltest.txt

row	lesson	name	outcome	prod	type	skill
1	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
2	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
3	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
4	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSSIPARENT1	BLANK	PICK-PARENTS
5	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
6	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSSIPARENT1	BLANK	PICK-PARENTS
7	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
8	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSSIPARENT1	BLANK	PICK-PARENTS
9	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSS1PARENT2	NA	PICK-PARENTS
10	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
11	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSSIPARENT1	BLANK	PICK-PARENTS
12	ZGeneticsZGeneZInteractioy2PS	student0	WRONG	CROSSIPARENT1	STRING	PICK-PARENTS
13	ZGeneticsZGeneZInteractioy2PS	student0	HELP	CROSSIPARENT1	BLANK	PICK-PARENTS

Row Count: 36395

Figure 10: EDM DataGrid

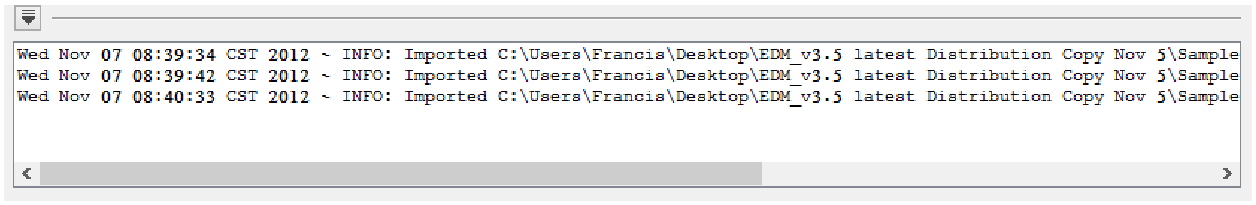
The **DataGrid** displays the logs that are active and are to be processed. The down arrow button hides the data grid.



Row Count: 39468

Row Count controls the amount of rows shown in the active tab

Status Box



```
Wed Nov 07 08:39:34 CST 2012 ~ INFO: Imported C:\Users\Francis\Desktop\EDM_v3.5 latest Distribution Copy Nov 5\Sample
Wed Nov 07 08:39:42 CST 2012 ~ INFO: Imported C:\Users\Francis\Desktop\EDM_v3.5 latest Distribution Copy Nov 5\Sample
Wed Nov 07 08:40:33 CST 2012 ~ INFO: Imported C:\Users\Francis\Desktop\EDM_v3.5 latest Distribution Copy Nov 5\Sample
```

Figure 11: System Status Box

The **Status Bar** displays feedback information such as status, error messages, time elapsed and others.

Loading Animation

Loading animation has been added to export, import, load, and save functions to easily identify if the program has either hanged or is still functioning.

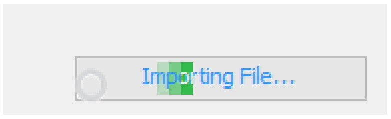


Figure 12: Loading Animation

Chapter 2. System Manual

Import

The EDM Workbench allows users to import logs in DataShop text format and CSV. The data is assumed to be stored in a flat file, organized in rows and columns. The first row of the import file is assumed to contain each column's name. Each succeeding row represents one logged transaction, usually between the student and tutor but possibly between two or more students as in the case of collaborative learning scenarios. The successfully-imported logs may be saved in the Workbench's format for work files—a compressed file containing the data in CSV format plus metadata specific to the EDM Workbench.

Import log file by clicking Import Button Import located either in File menu (Figure 6) or Toolbar (Figure 9). The system will then pop-up a dialog box asking what type of logs you want to import (CSV or Datashop Text file Figure 13). Click the Select Button after selecting the type of Log.

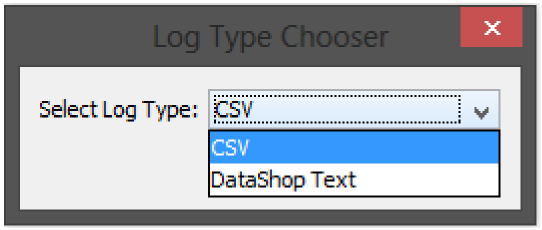
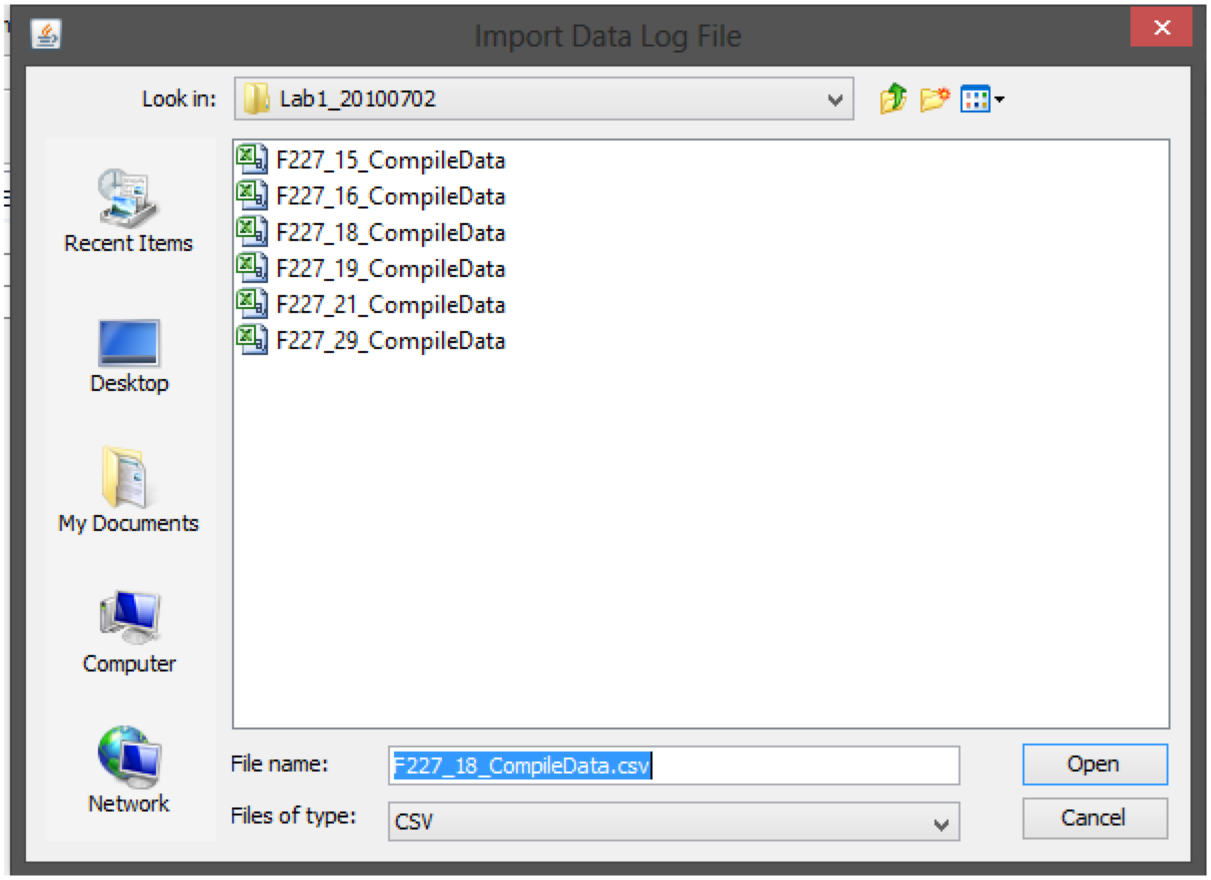


Figure 13: Log Selection

Another dialog box will ask for the location of the log file.



F227_15_CompileData
F227_16_CompileData
F227_18_CompileData
F227_19_CompileData
F227_21_CompileData
F227_29_CompileData

File name: F227_18_CompileData.csv
Files of type: CSV

Figure 14: Selection of Data File to be imported

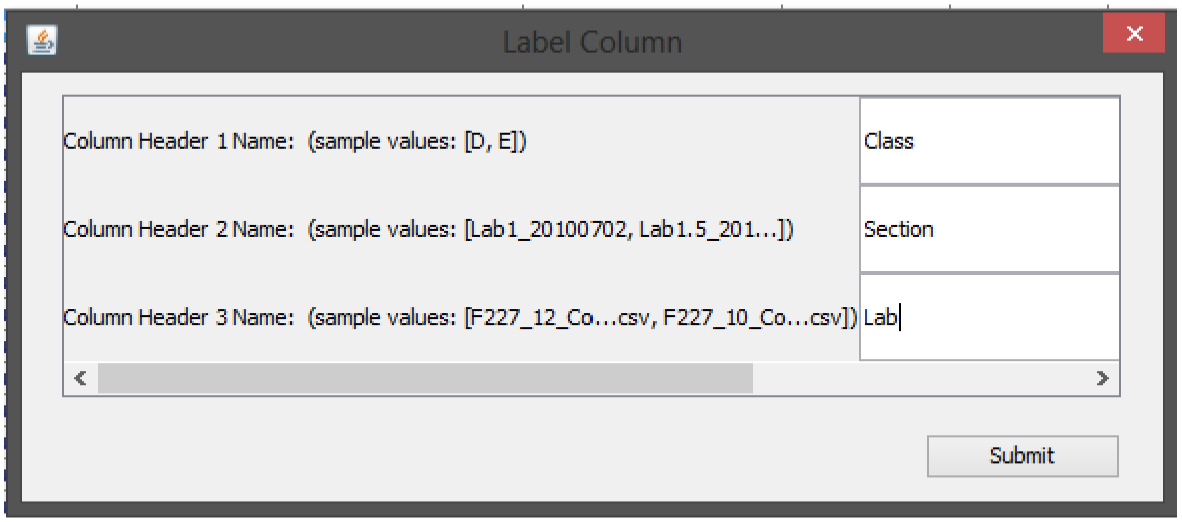
Case 1: Importing a single log file

If a user imports a single log file after locating and choosing the log file, the Workbench displays the file in the DataGrid (Figure 10).

Case 2: Importing batches of log files

The Workbench can also import nested folders of data, where each folder level represents a meaningful subset of the data. For example, if data from a section of students is collected several times over a school year, the researcher may have one folder for the school year, one subfolder for each section within the school year, one subfolder for a session within each section, and finally one file or folder for each student within a session. The Workbench allows users to label each level of subfolder, creating new columns for these labels, appending them to the data tables during importation process.

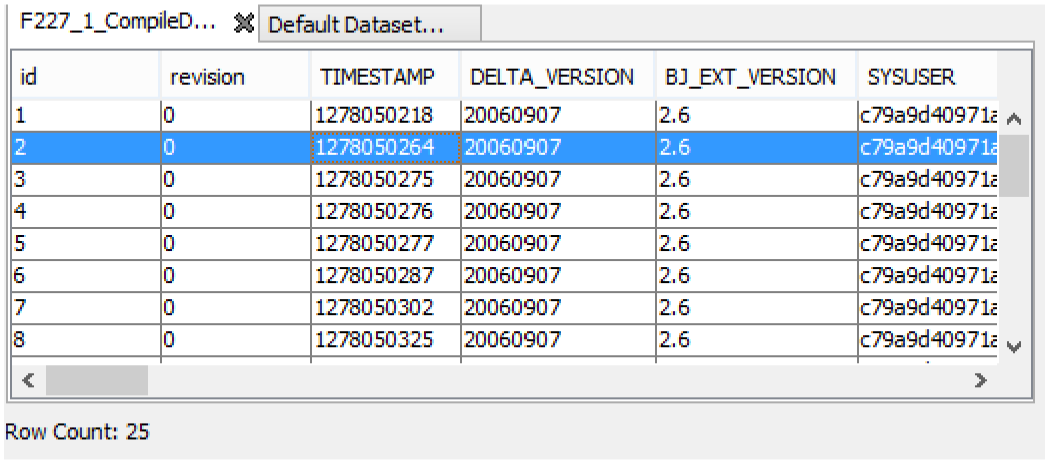
After locating and choosing the batch of log files another dialog box will appear asking for a label describing the log files imported (e.g Class) (Figure 14). Clicking Submit aggregates all the logs and displays them in the DataGrid.



Column Header 1 Name: (sample values: [D, E])	Class
Column Header 2 Name: (sample values: [Lab1_20100702, Lab1.5_201...])	Section
Column Header 3 Name: (sample values: [F227_12_Co...csv, F227_10_Co...csv])	Lab

Figure 15: Label Column with sample parameters

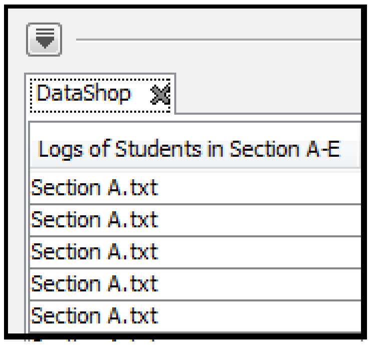
Once the logs are loaded, the DataGrid should be populated (Figure 16). All actions buttons, save for the Labelling button, should be enabled at this point.



id	revision	TIMESTAMP	DELTA_VERSION	BJ_EXT_VERSION	SYSUSER
1	0	1278050218	20060907	2.6	c79a9d40971a
2	0	1278050264	20060907	2.6	c79a9d40971a
3	0	1278050275	20060907	2.6	c79a9d40971a
4	0	1278050276	20060907	2.6	c79a9d40971a
5	0	1278050277	20060907	2.6	c79a9d40971a
6	0	1278050287	20060907	2.6	c79a9d40971a
7	0	1278050302	20060907	2.6	c79a9d40971a
8	0	1278050325	20060907	2.6	c79a9d40971a

Row Count: 25

Figure 16: EDM sample Data Set

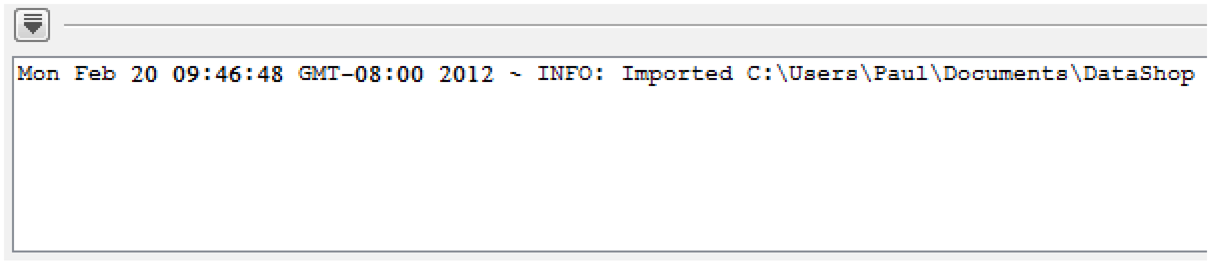


DataShop

Logs of Students in Section A-E

Section A.txt
Section A.txt
Section A.txt
Section A.txt
Section A.txt

Figure 17: EDM Workbench Data Shop Tab



```
Mon Feb 20 09:46:48 GMT-08:00 2012 ~ INFO: Imported C:\Users\Paul\Documents\DataShop
```

Figure 18: Status bar with timestamp and file directory

The **Status bar** displayed the information of the file imported together with the location **C:\User\Paul\Documents\Datashop** and the current time **Monday February 20 9:46 AM and 48 seconds**.

- ## Clipping

The EDM Workbench allows the user to define the set of features by which the data should be grouped, so that clips do not contain rows from different groups. For example, if the data should be grouped by student, a single clip will contain data from only one student and not multiple students. The Workbench also specifies the clip size, either by time or by number of transactions. Delineation of clips by beginning and ending events is not yet possible, but is a feature planned for future implementation. The Workbench then generates the clips for analysis, according to a sampling scheme discussed in the next section

To clip the dataset, click Clip Button located either in the Function menu (Figure 7) or Toolbar (Figure 9). The system will then display a form with the column names (the basis for grouping e.g. group data with the same Logs of Student in Section A-E with the same Anon Student Id and with the same Time and so on). Clips can be divided by **Size**, **Time** or **Per Value Changed.**

○ Size as Clip

Type

By choosing **Size** as the **Clip Type**, the user will need to specify the desired number of transactions in a clip.

"*Complete Clips Only*" when checked, the system will only select clips where the number of logs is equal to the inputted clip size.

"*Allow Overlap*" when checked, the system will produce clips with overlapping logs. Given logs {1,2,3,4,5} and a clip size of 3, three clips will be produced: {1,2,3}, {2,3,4}, and {3,4,5}.

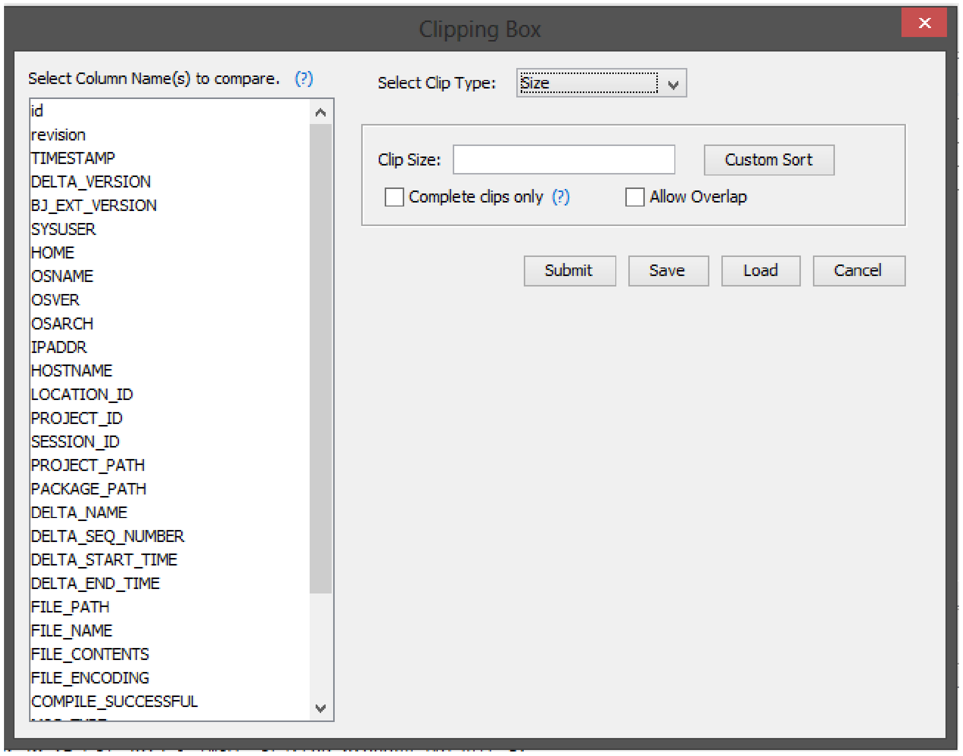
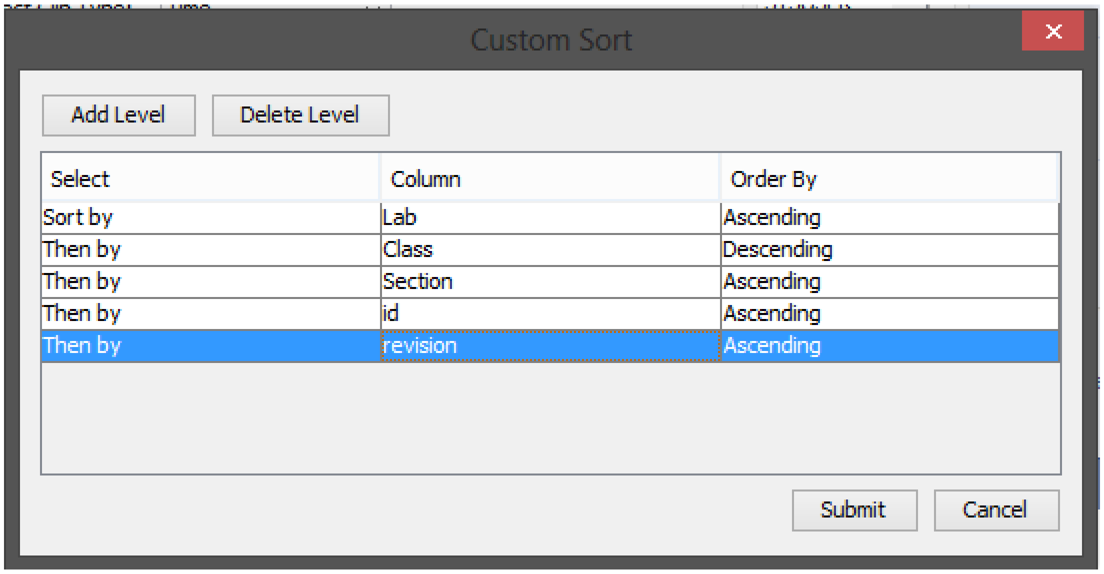


Figure 19: EDM Clipping Window

▪ Custom Sort Button

This allows the user to set how the transactions within a clip are ordered by sorting them according to criteria. **Add Level** Button adds another sorting criterion while **Delete Level** deletes the selected Row. Clicking the **Submit** button will implement the selected formatting properties.



Select	Column	Order By
Sort by	Lab	Ascending
Then by	Class	Descending
Then by	Section	Ascending
Then by	id	Ascending
Then by	revision	Ascending

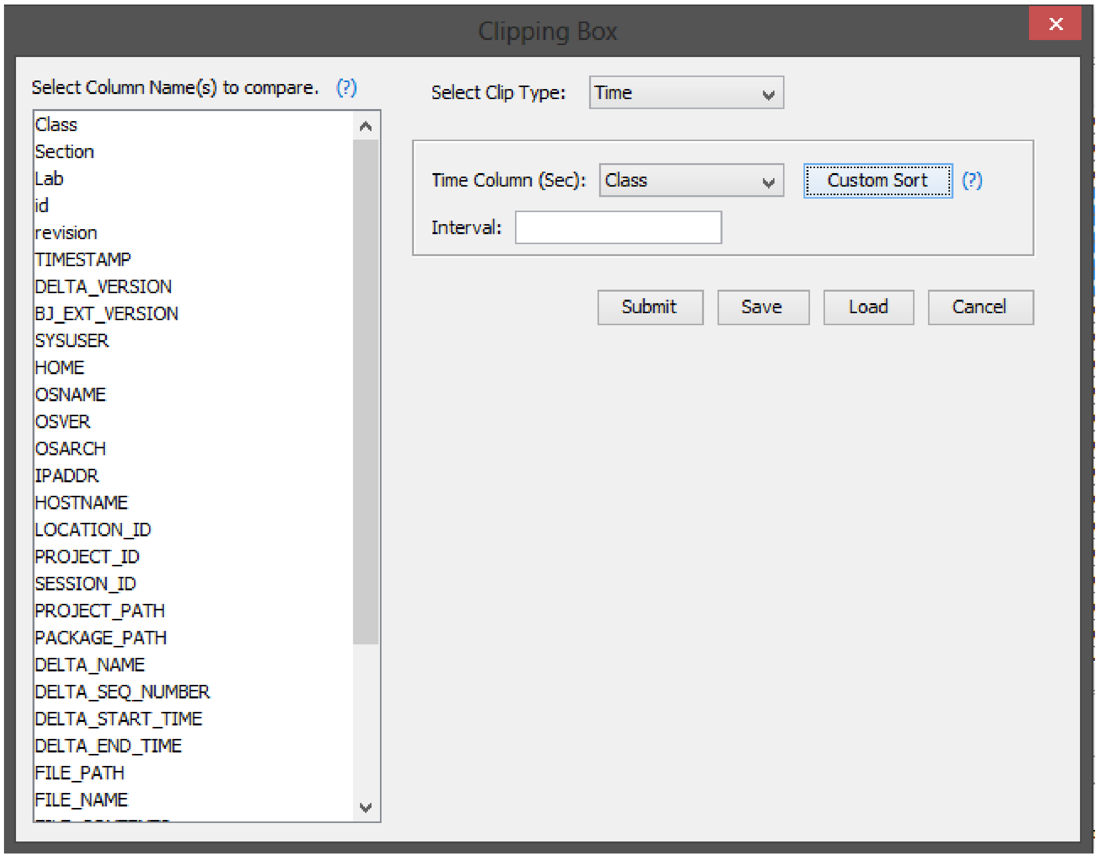
Figure 20: EDM Custom Sort

- ### Time as Clip Type

  By choosing **Time** as the **Clip Type**, the user will specify a time period per clip (e.g. 1 clip = 5 minutes interval). The column name with a time element (measured in seconds) must be specified. When done, click the submit button and double click the clips to view the inclusive logs.

- ### Per Value Change as Clip Type

  Per Value Change creates a new clip every time the value within the specified column changes.


Clipping Box

Select Column Name(s) to compare. (?)

Class
Section
Lab
id
revision
TIMESTAMP
DELTA_VERSION
BJ_EXT_VERSION
SYSUSER
HOME
OSNAME
OSVER
OSARCH
IPADDR
HOSTNAME
LOCATION_ID
PROJECT_ID
SESSION_ID
PROJECT_PATH
PACKAGE_PATH
DELTA_NAME
DELTA_SEQ_NUMBER
DELTA_START_TIME
DELTA_END_TIME
FILE_PATH
FILE_NAME

Select Clip Type: Time

Time Column (Sec): Class

Interval:

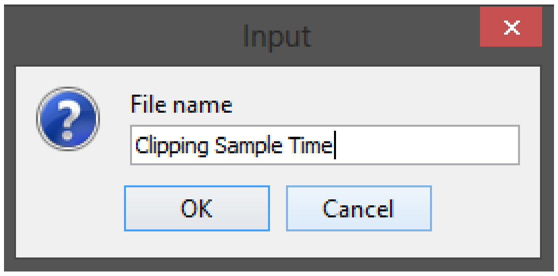
Figure 21: Window showing the Time as Clip Type

- ### Cancel Button

  This cancels clipping.

- ### Save Button

  The **Save** button saves the set properties applied in the Clipping Form. The user supplies a file name and clicks OK.


Input

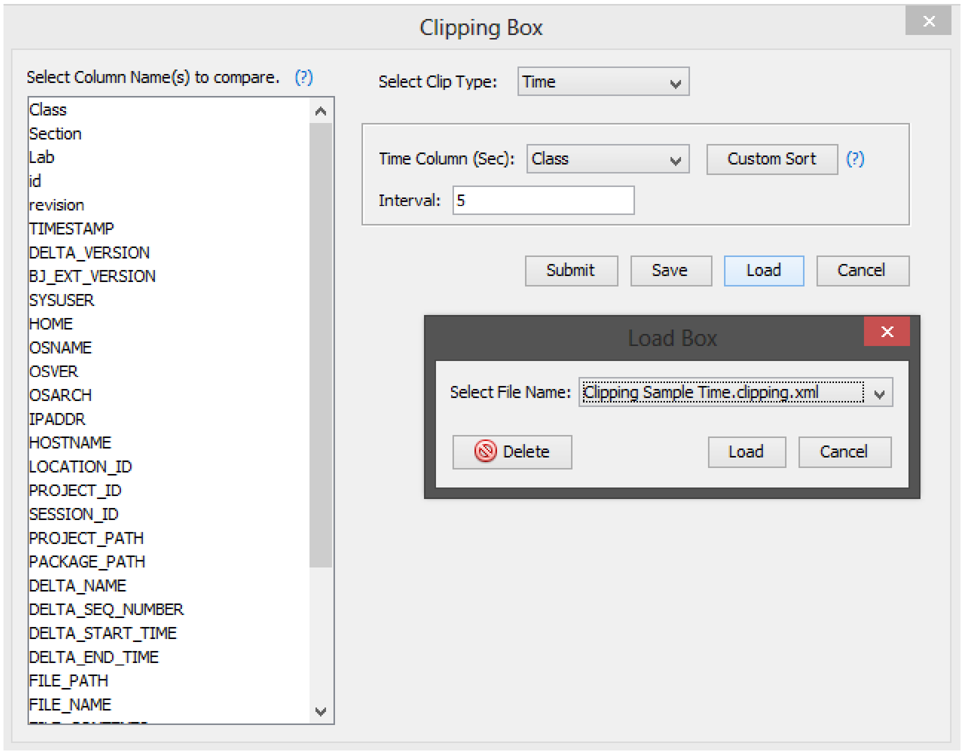
File name

Clipping Sample Time

Figure 22: Save Dialogue

- ## Load Button

  Allows the user to select and load a previously-saved file from a drop-down list. (see Figure 23).



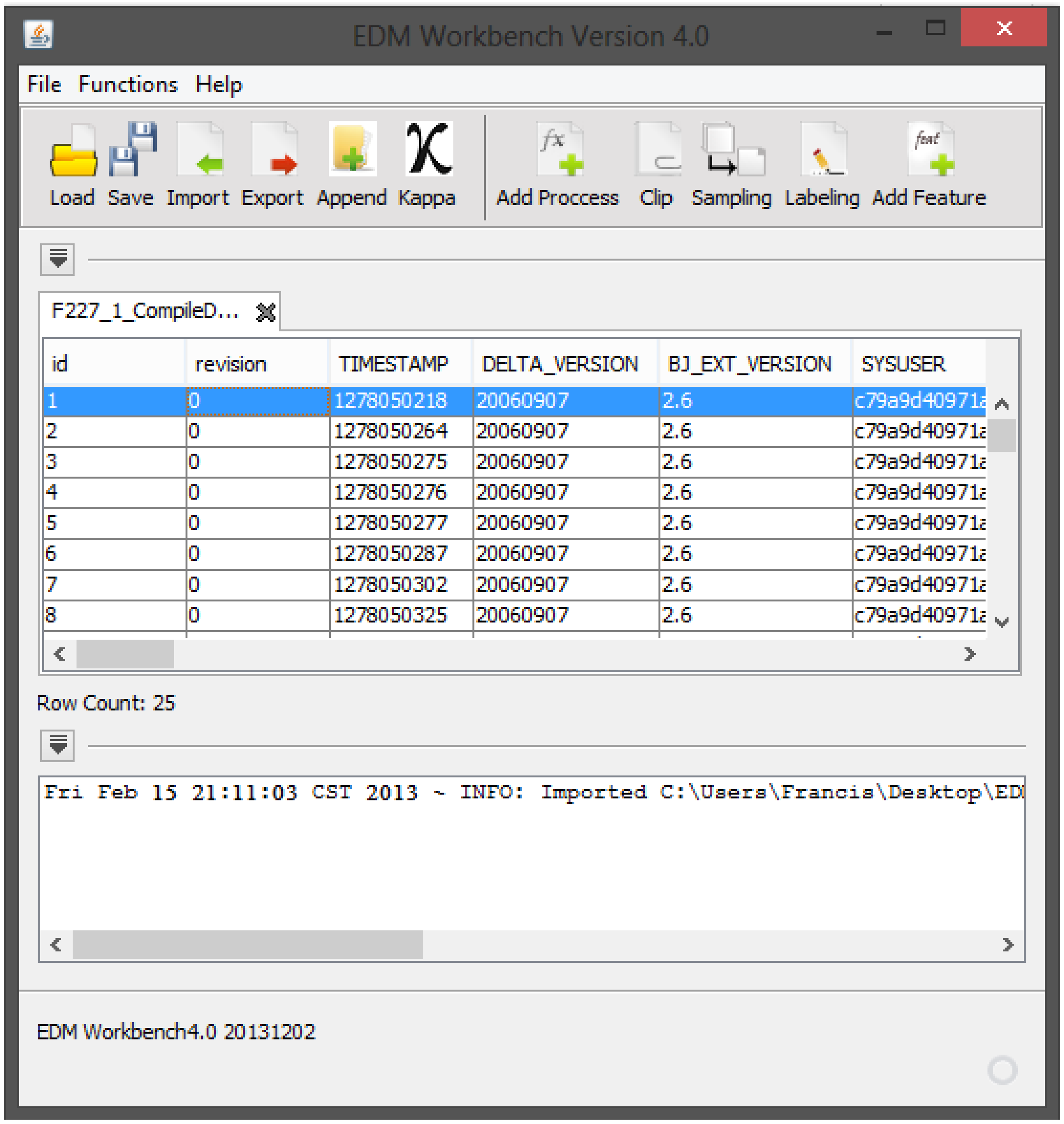
Select Clip Type: Time
Time Column (Sec): Class
Custom Sort (?)
Interval: 5
Select File Name: Clipping Sample Time.clipping.xml

Figure 23: Load Window

Note: From the list of clipping.xml files, the selected template is Clipping Sample Time.clipping.xml

- ## Submit Button

  This closes the Clipping Form, clips the dataset from the current tab, and displays it with its properties set in a new tab. Double click a row to view the logs within it.


EDM Workbench Version 4.0
File Functions Help
Load Save Import Export Append Kappa Add Proccess Clip Sampling Labeling Add Feature
F227_1_CompileD...

id	revision	TIMESTAMP	DELTA_VERSION	BJ_EXT_VERSION	SYSUSER
1	0	1278050218	20060907	2.6	c79a9d40971a
2	0	1278050264	20060907	2.6	c79a9d40971a
3	0	1278050275	20060907	2.6	c79a9d40971a
4	0	1278050276	20060907	2.6	c79a9d40971a
5	0	1278050277	20060907	2.6	c79a9d40971a
6	0	1278050287	20060907	2.6	c79a9d40971a
7	0	1278050302	20060907	2.6	c79a9d40971a
8	0	1278050325	20060907	2.6	c79a9d40971a

Row Count: 25

```
Fri Feb 15 21:11:03 CST 2013 ~ INFO: Imported C:\Users\Francis\Desktop\ED
```

EDM Workbench4.0 20131202

Figure 24: Clip submission

Sampling

The data sampling feature of the Workbench allows the user to specify how clips are sampled from the data set. (It can also be used to sample at the action/transaction level). The user can specify the sample size, and whether the Workbench will randomly take the sample across the entire population or whether the workbench will stratify the sampling based on one or more variables.

Note that the Workbench allows the user to sample the data at any point of the process — after importing, after clipping, or after labelling – depending on the user's analytical goals.

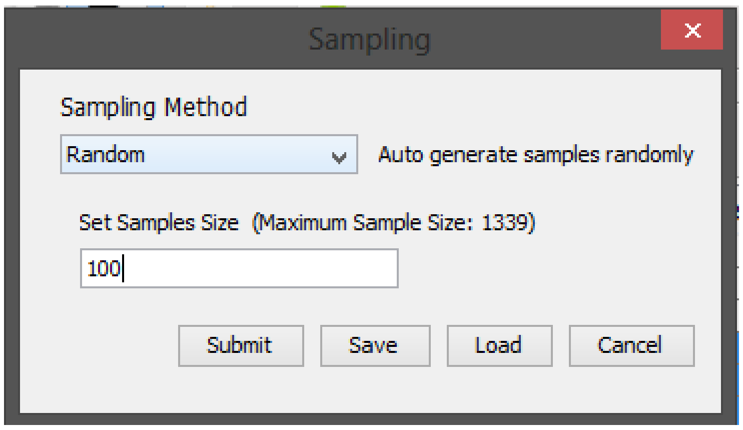
To start sampling the dataset, click Sampling Button located either in the **Function** menu (Figure 7) or **Toolbar** (Figure 9). Sampling functionalities involve creating subsets from the dataset using automatic select and grouping options. A user may take samples or a subset from the loaded dataset and save as a new dataset. Sampling can be stratified or random.

○ Random Sampling

To randomly select samples from a selected dataset:

Select Sampling Method > **Random**

Indicate the number of samples in the Sample Size textbox.



Sampling Method: Random — Auto generate samples randomly

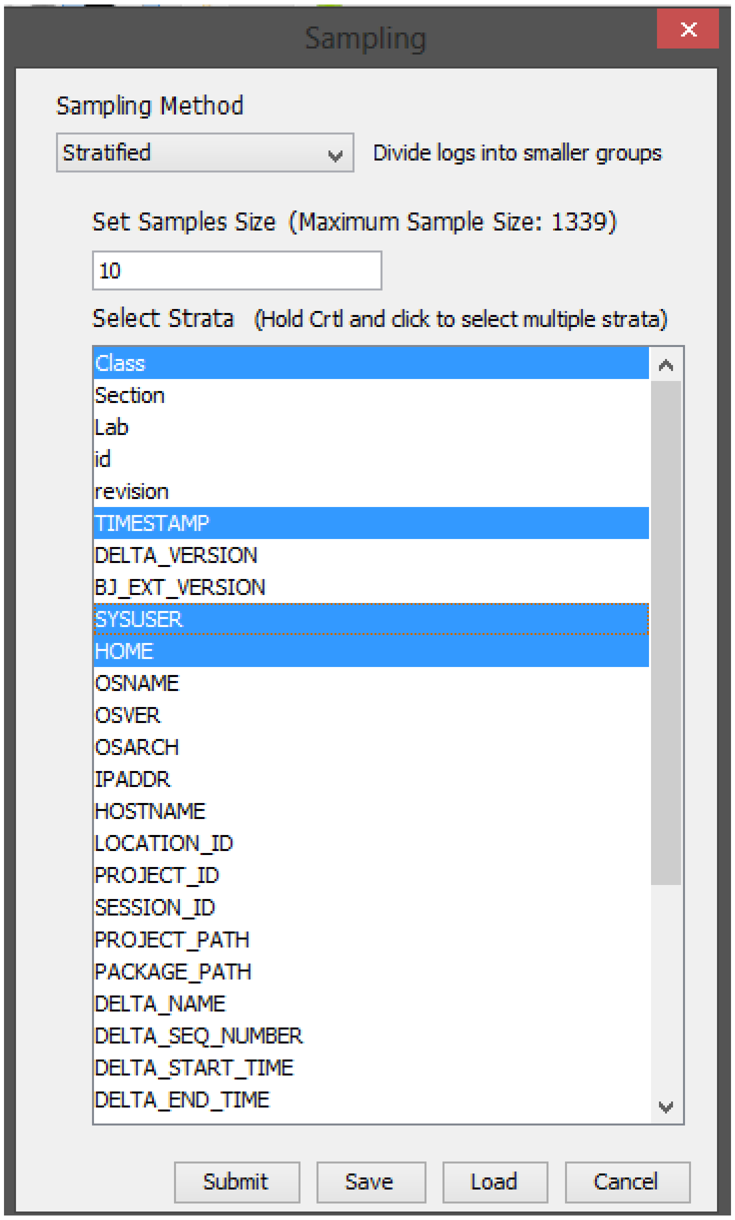
Set Samples Size (Maximum Sample Size: 1339): 100

Figure 25: Sampling method selection

Note: The size inputted in the textbox should not exceed the indicated maximum sample size. If the user specifies a number greater than the maximum, the operation returns all the rows in the dataset.

- ## Stratified Sampling

Stratified sampling randomly selects data from within specified subgroups to produce a stratified sample.
Select “Sampling Method” > **Stratified**
Set the number of samples in the Sample Size textbox
In the **Strata** list, click the column names that define the groupings.
(Figure 25).



Sampling Method
Stratified — Divide logs into smaller groups
Set Samples Size (Maximum Sample Size: 1339)
10
Select Strata (Hold Crtl and click to select multiple strata)

Class
Section
Lab
id
revision
TIMESTAMP
DELTA_VERSION
BJ_EXT_VERSION
SYSUSER
HOME
OSNAME
OSVER
OSARCH
IPADDR
HOSTNAME
LOCATION_ID
PROJECT_ID
SESSION_ID
PROJECT_PATH
PACKAGE_PATH
DELTA_NAME
DELTA_SEQ_NUMBER
DELTA_START_TIME
DELTA_END_TIME

Figure 26: Strata selection

- ### Save Button

  Save Button saves the properties as a template.

- ### Load Button

  The Load button, allows the user to choose a previously-saved sampling template from a list and apply it to the current dataset.

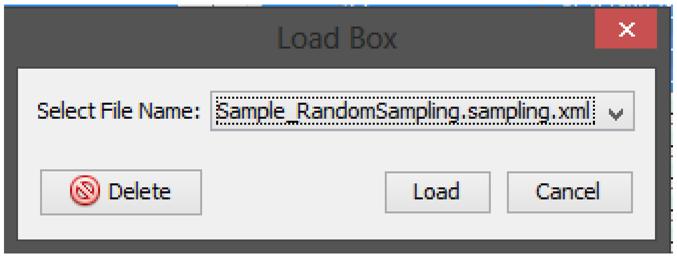


Figure 27: Load Prompt

- ### Submit Button

  The submit button closes the Sampling Form, implements the sampling process and then displays the result in a new tab.

▪ Add Process

This allows the user to create a script composed of multiple processes and run them in a single thread.

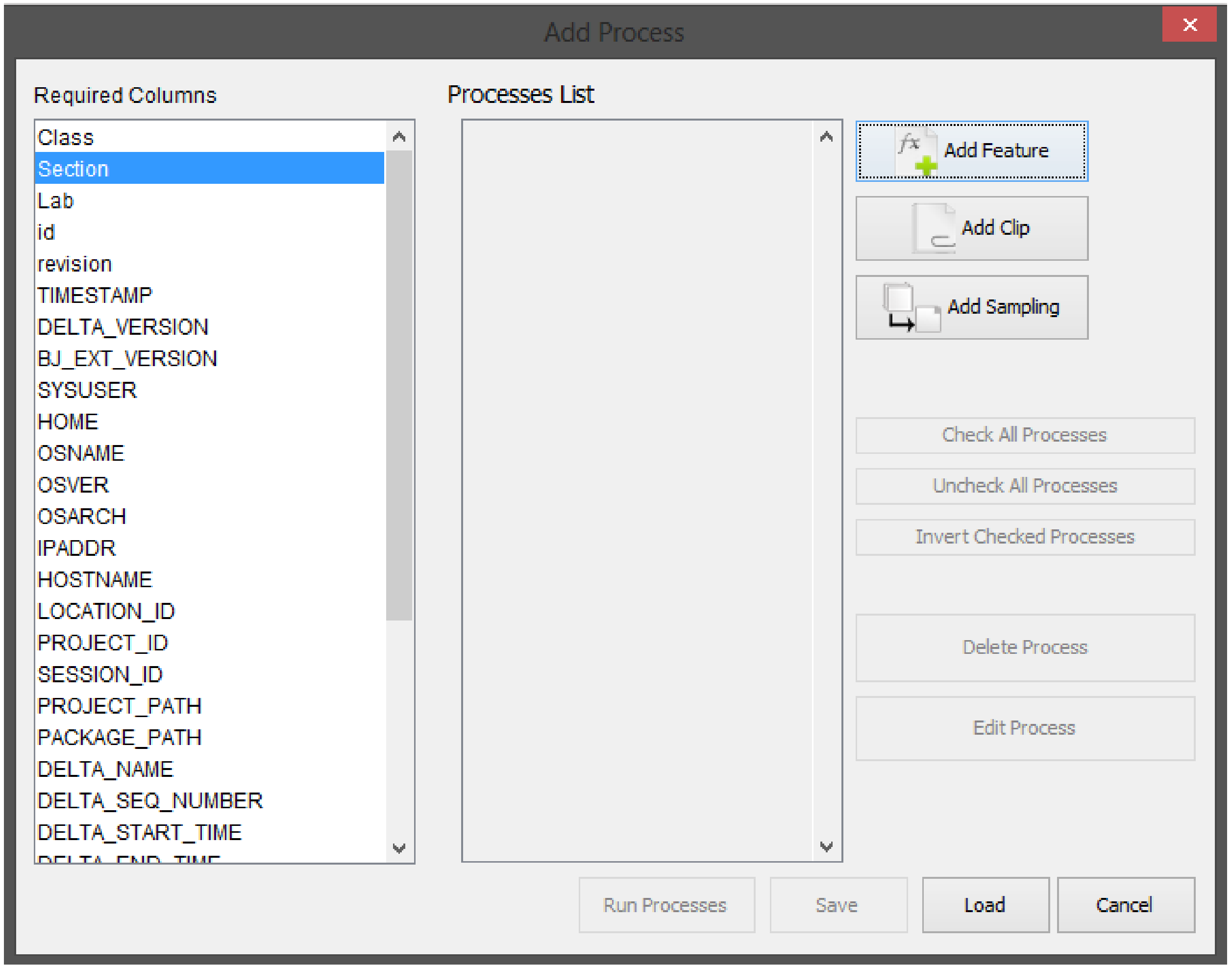
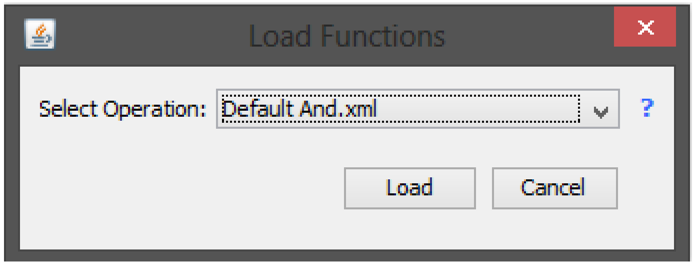


Figure 28: Feature selection window

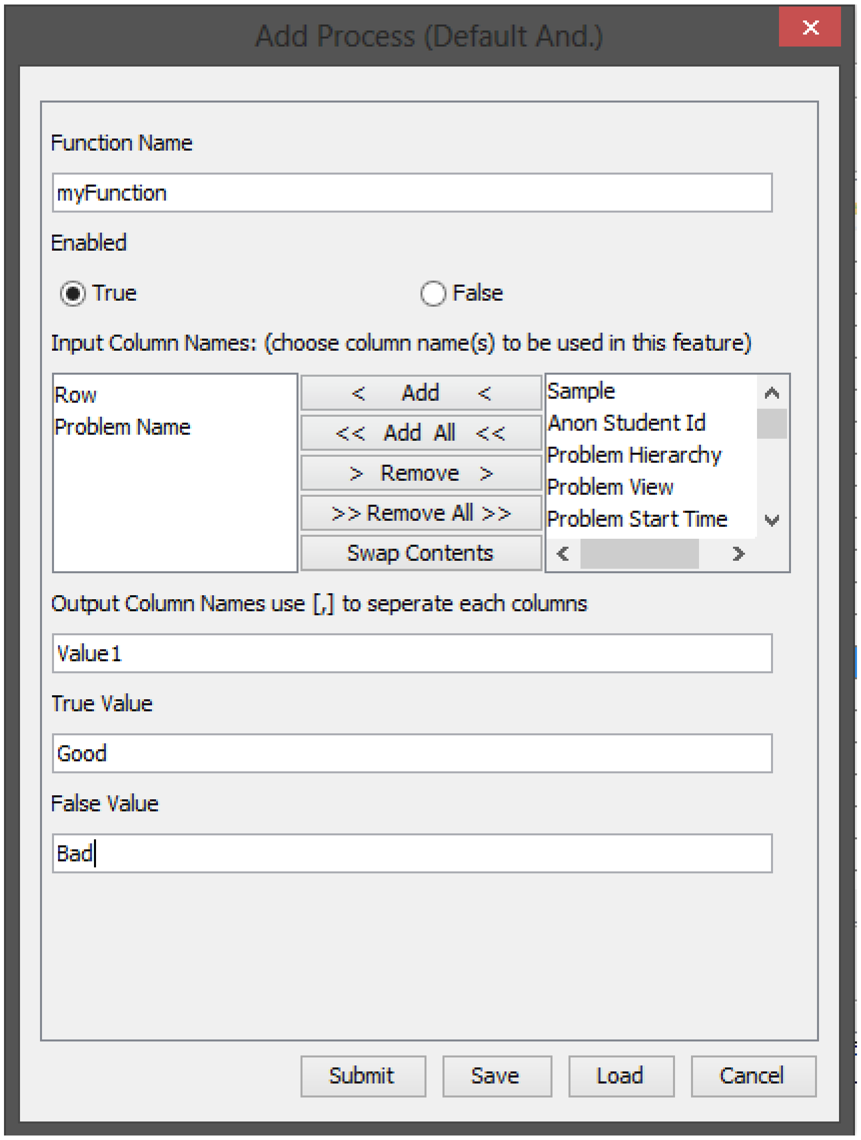
○ Add Feature

This function allows users to add features to the dataset through the application of predefined operations.



Select Operation: Default And.xml

Figure 29: Load Function Dialogue



Function Name: myFunction

Enabled: [✓ True] False

Input Column Names: (choose column name(s) to be used in this feature)

Row	Sample
Problem Name	Anon Student Id
	Problem Hierarchy
	Problem View
	Problem Start Time

Output Column Names use [,] to seperate each columns: Value1

True Value: Good

False Value: Bad

Figure 30: Modified function window with the feature And selected

- ## Add Feature Operations
  - ### Default And

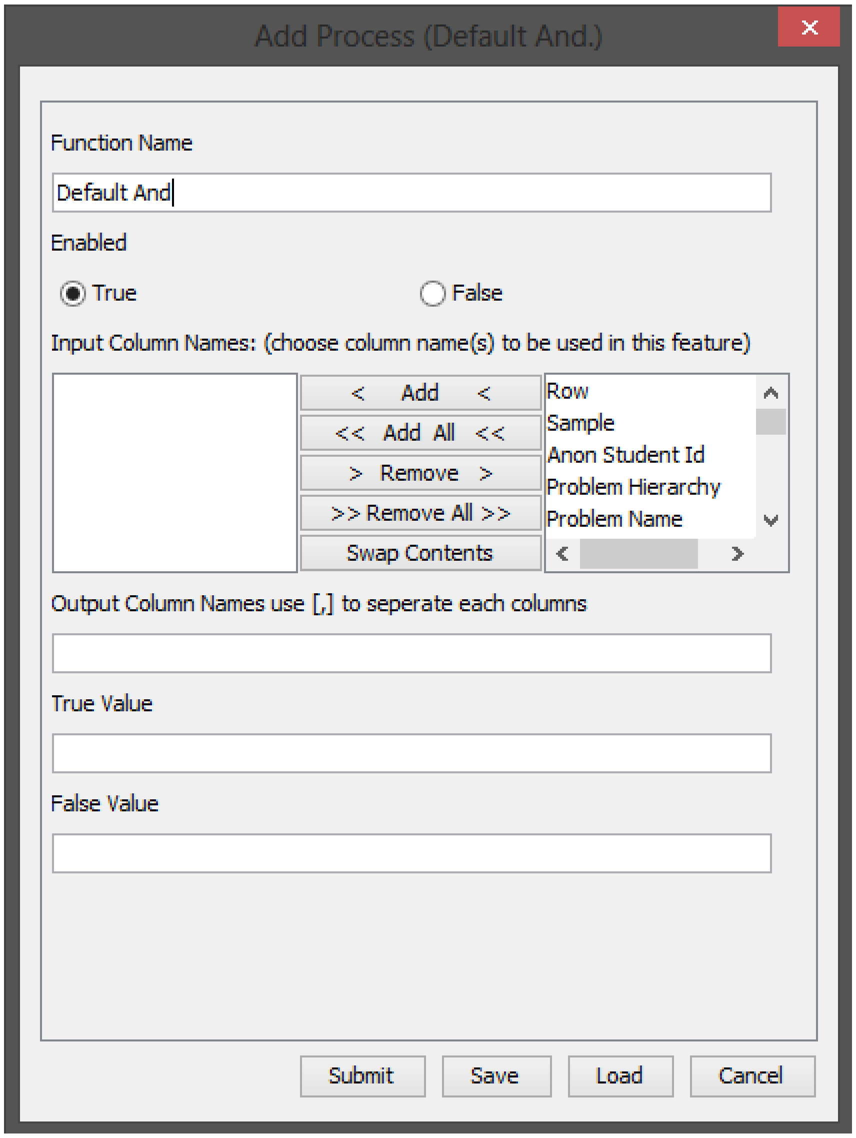
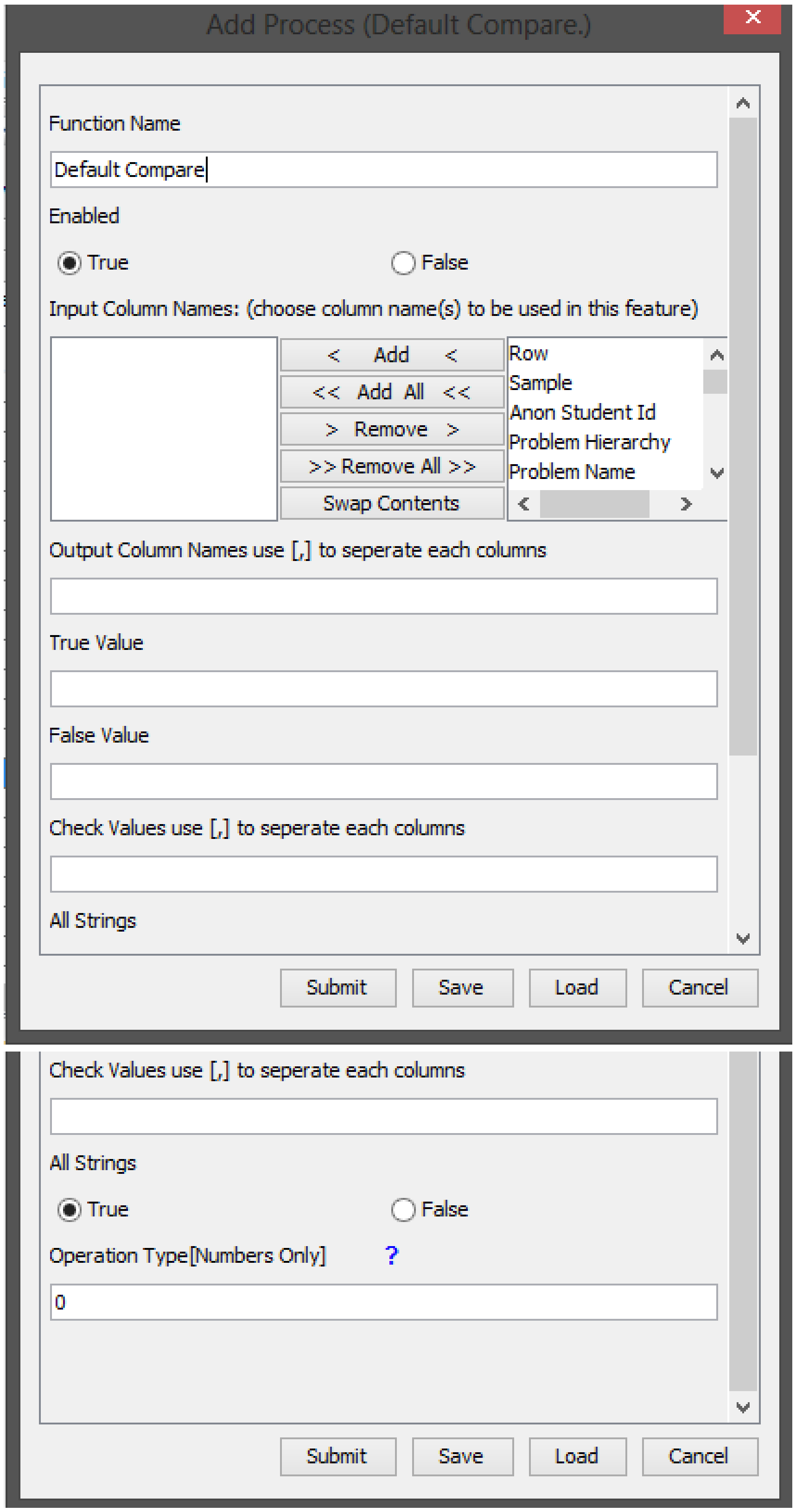


Figure 31: Default And function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not. *True Value* assigned to the result in the **Output Column Name** if operation returns a true. *False Value* assigned to the result in the **Output Column Name** if operation returns a false.

- ## Default Compare


Add Process (Default Compare.)

Function Name

Default Compare

Enabled

[✓ True] False

Input Column Names: (choose column name(s) to be used in this feature)

Row
Sample
Anon Student Id
Problem Hierarchy
Problem Name

Output Column Names use [,] to seperate each columns

True Value

False Value

Check Values use [,] to seperate each columns

All Strings

[✓ True] False

Operation Type[Numbers Only] ?

0

Figure 32: Default Compare window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

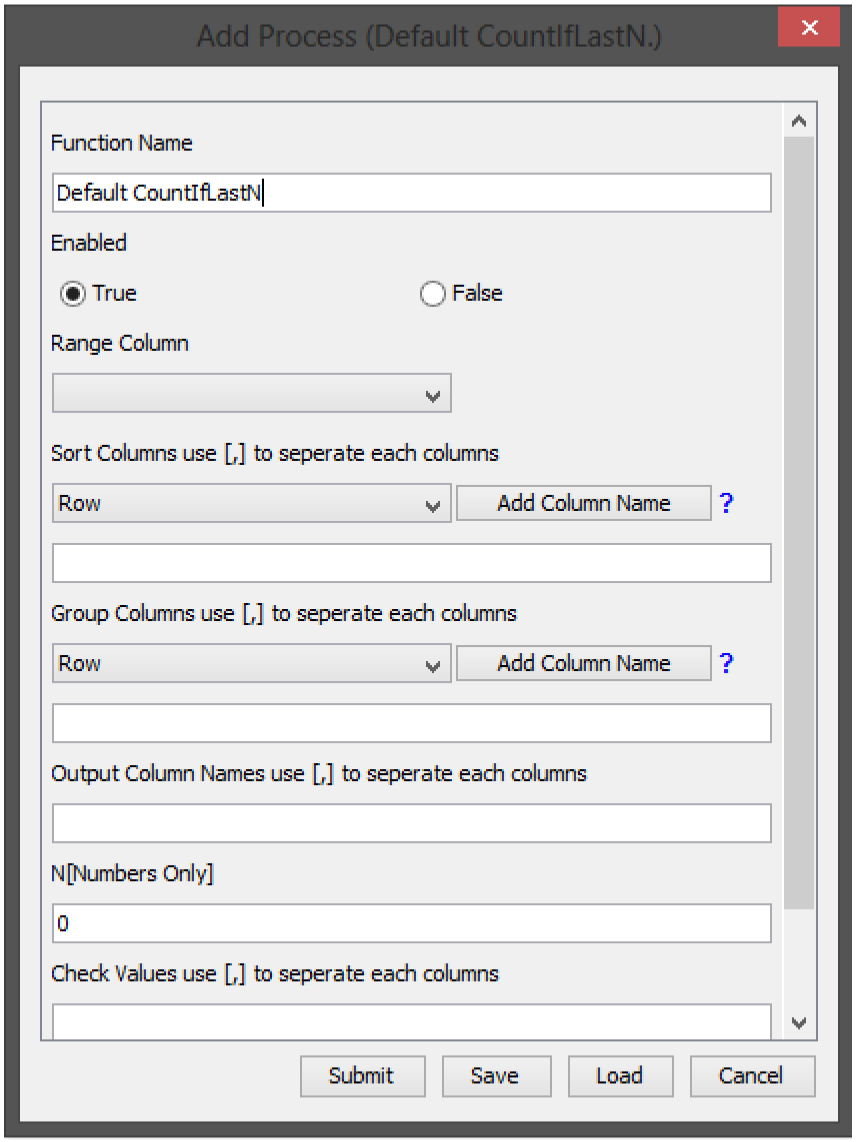
False Value assigned to the result in the **Output Column Name** if operation returns a false.

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

All String checks if all the column values are strings, not numbers or any other type.

Operation Type contains values from 1-6 that correspond to different operations. Strings or integers can be compared in this feature.

- ## Default CountIfLastN



Function Name
Default CountIfLastN

Enabled
(● True) ○ False

Range Column

Sort Columns use [,] to seperate each columns
Row

Group Columns use [,] to seperate each columns
Row

Output Column Names use [,] to seperate each columns

N[Numbers Only]
0

Check Values use [,] to seperate each columns

Figure 33: Default CountIfLastN function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

False Value assigned to the result in the **Output Column Name** if operation returns a false.

***Range Column* -** Range of values used for computation.

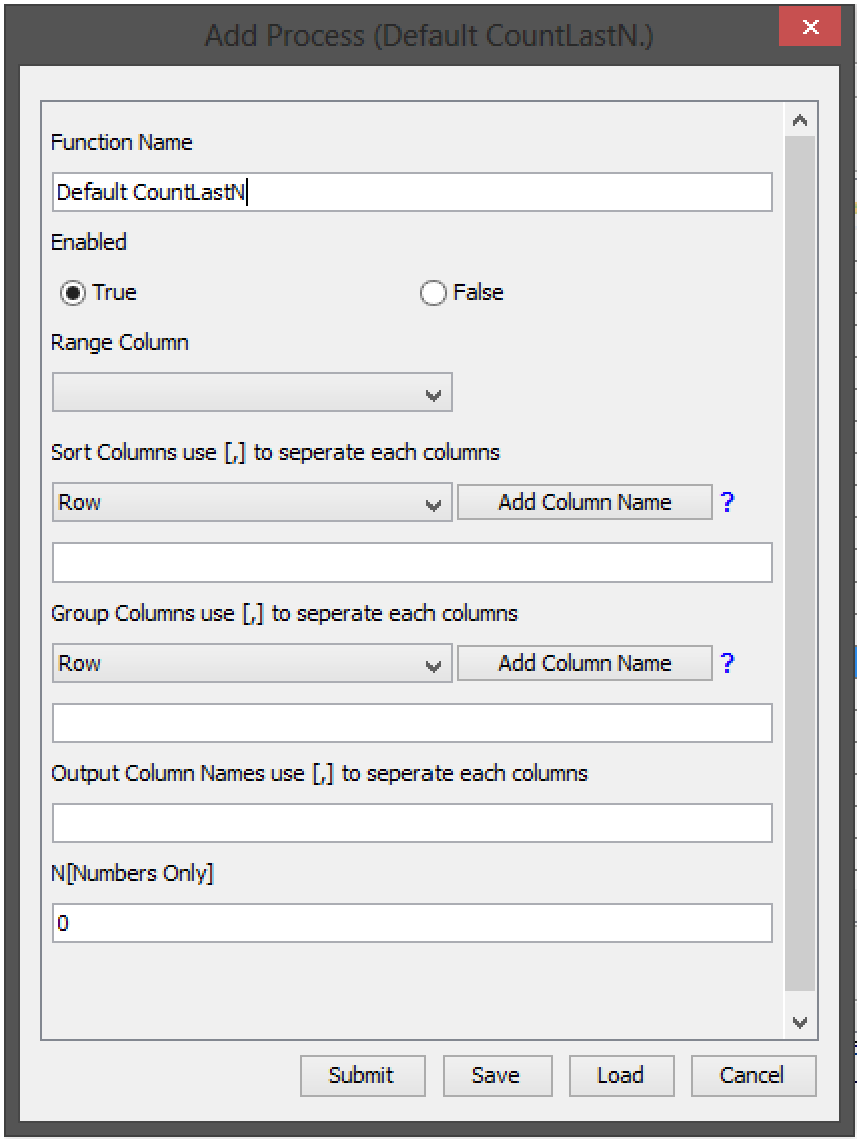
***Sort Column* -** used for sorting the rows within the
same group.

***Group Column* -** Used for grouping rows with the same values for selected columns.

N[Numbers Only] if more elements in a group are found, only the last N items are kept for processing/start count every N rows.

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

- ## Default CountLastN


Add Process (Default CountLastN.)

Function Name: Default CountLastN

Enabled: (● True) ○ False

Range Column:

Sort Columns use [,] to seperate each columns: Row

Group Columns use [,] to seperate each columns: Row

Output Column Names use [,] to seperate each columns:

N[Numbers Only]: 0

Figure 34: Default CountLastN function Window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

False Value assigned to the result in the **Output Column Name** if operation returns a false.

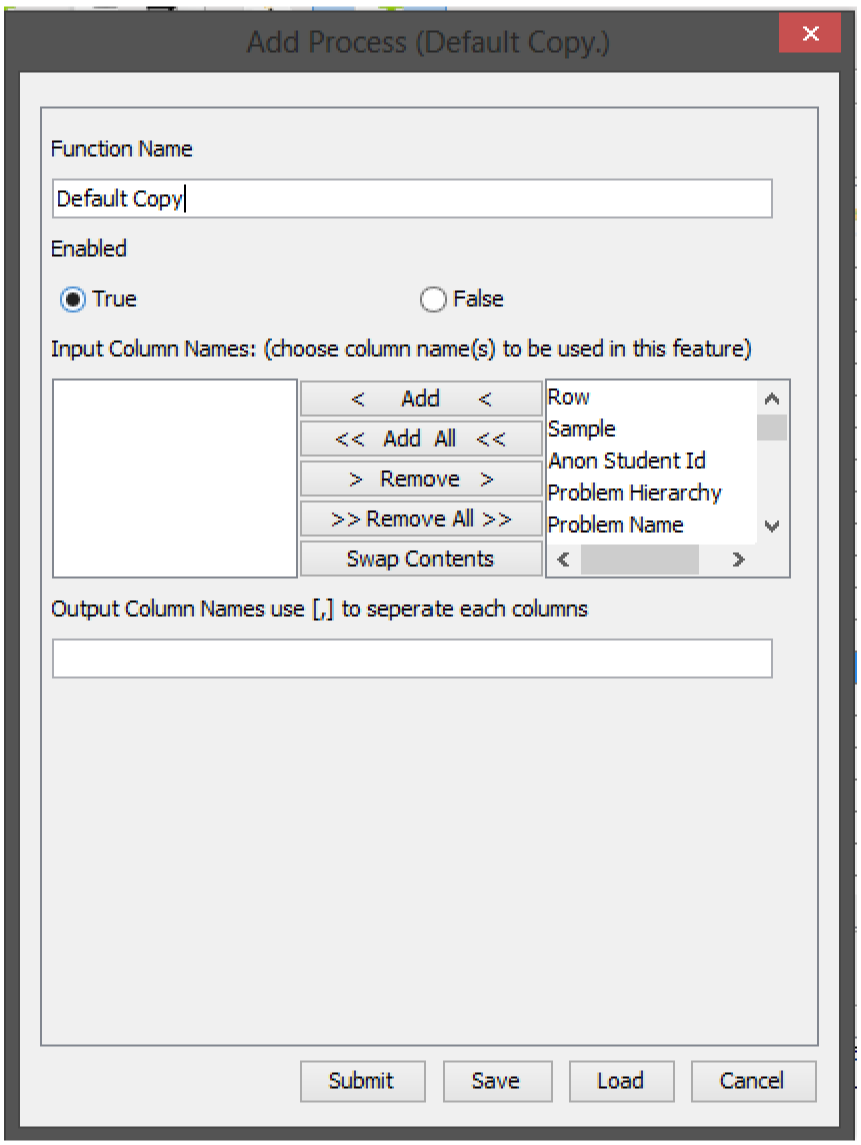
Range Column - Range of values used for computation.

Sort Column - used for sorting the rows within the
same group.

Group Column - Used for grouping rows with the same values for selected columns.

N[Numbers Only] if more elements in a group are found, only the last N items are kept for processing/start count every N rows.

• Default Copy


Add Process (Default Copy.)

Function Name

Default Copy

Enabled

(● True) ○ False

Input Column Names: (choose column name(s) to be used in this feature)

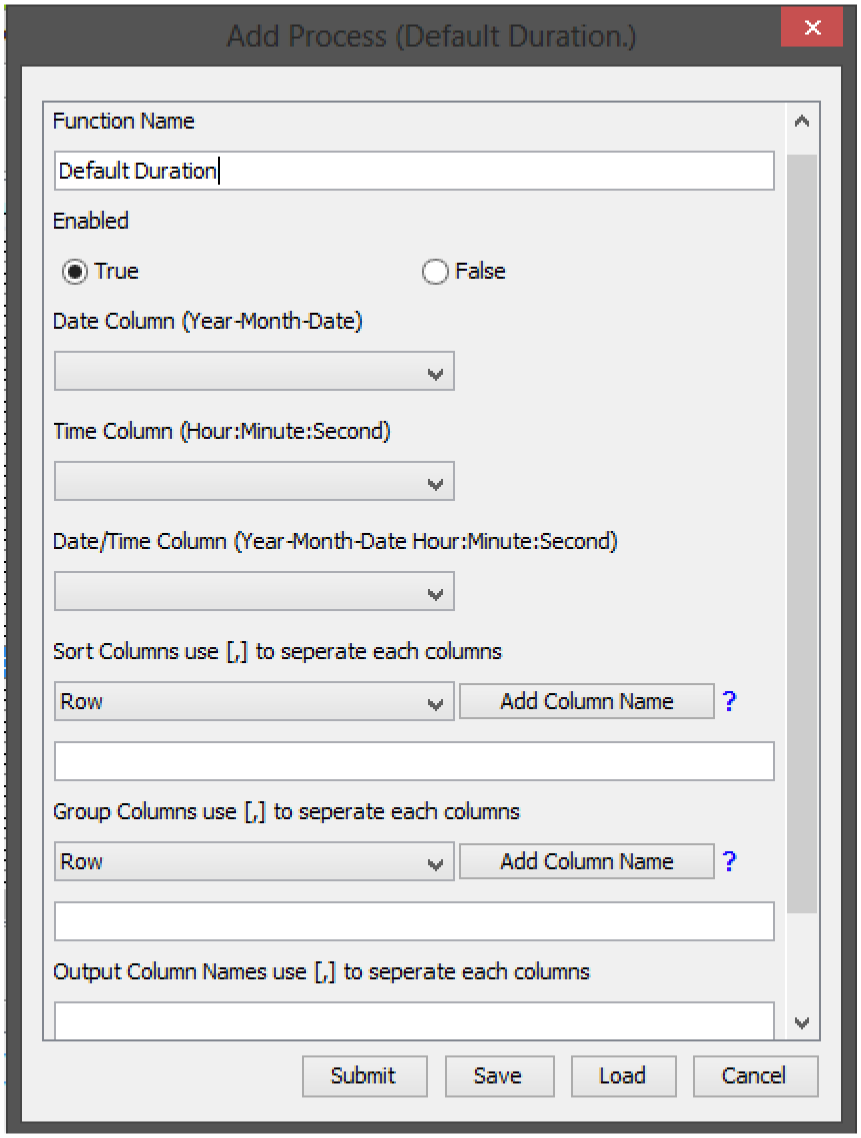
Output Column Names use [,] to seperate each columns

Figure 35: Default Copy function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

- ## Default Duration


Add Process (Default Duration.)

Function Name: Default Duration

Enabled: [✓ True] False

Date Column (Year-Month-Date)

Time Column (Hour:Minute:Second)

Date/Time Column (Year-Month-Date Hour:Minute:Second)

Sort Columns use [,] to seperate each columns: Row

Group Columns use [,] to seperate each columns: Row

Output Column Names use [,] to seperate each columns

Figure 36: Default Duration function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

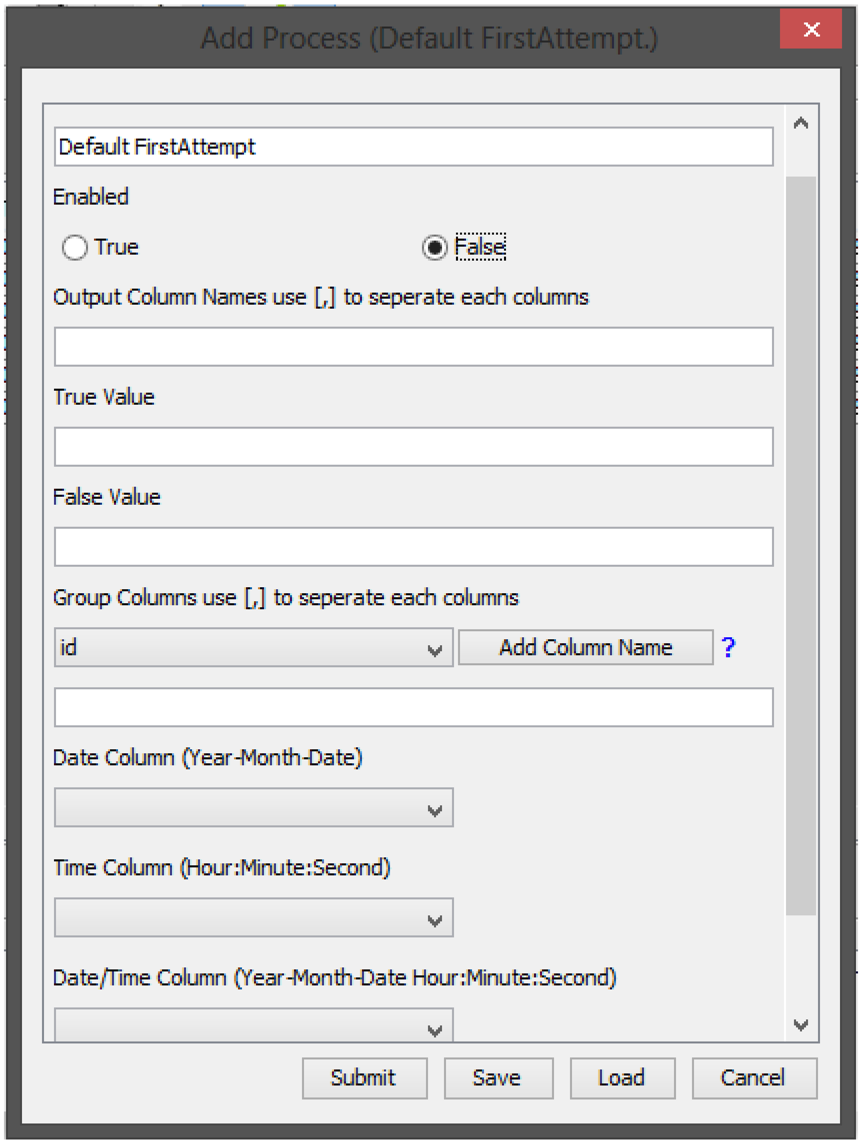
Date Column's value should be in the Date (Year-Month-Date)format.

Time Column's value should be in the Time (Hour:Minute:Second) format.

Sort Column - used for sorting the rows within
the same group.

Group Column - Used for grouping rows with the same values for selected columns.

- ## Default FirstAttempt


Add Process (Default FirstAttempt.)

Default FirstAttempt

Enabled

True [✓ False]

Output Column Names use [,] to seperate each columns

True Value

False Value

Group Columns use [,] to seperate each columns

id

Date Column (Year-Month-Date)

Time Column (Hour:Minute:Second)

Date/Time Column (Year-Month-Date Hour:Minute:Second)

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

***Sort Column* -** used for sorting the rows within
the same group.

***Group Column* -** Used for grouping rows with the same values for selected columns.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

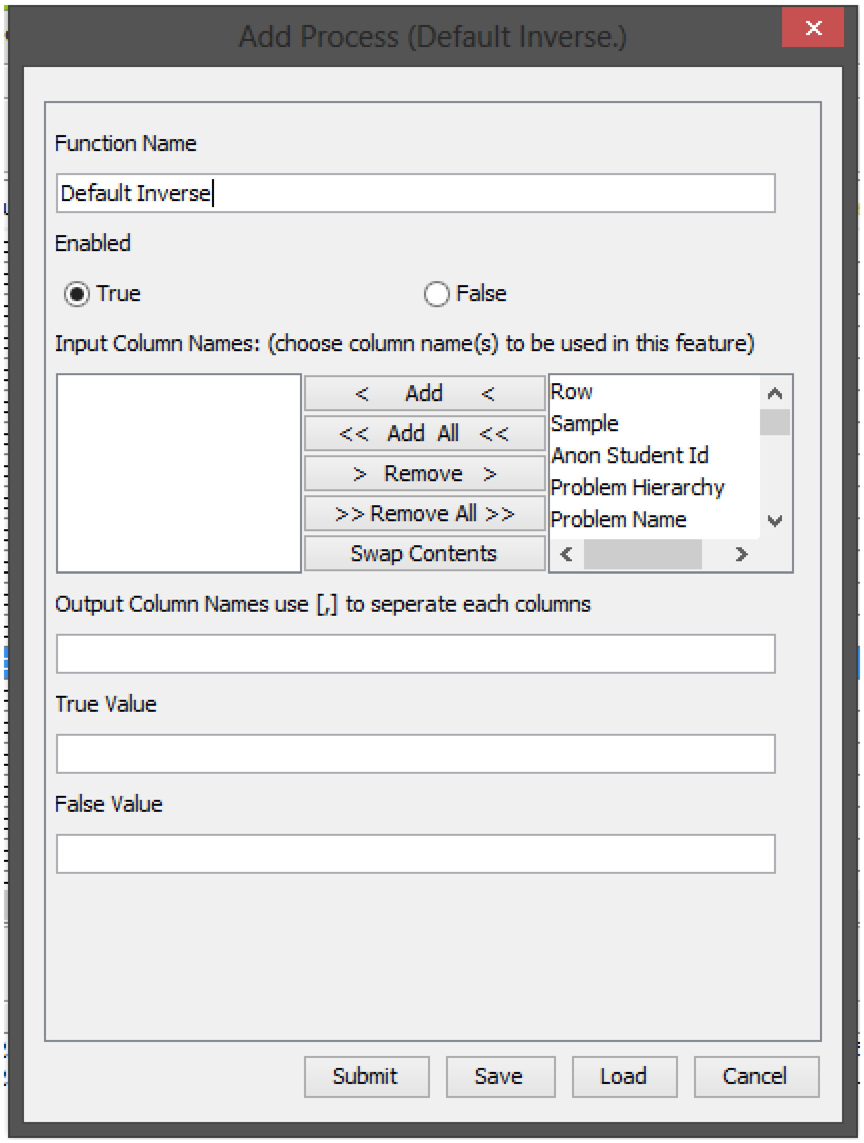
False Value assigned to the result in the **Output Column Name** if operation returns a false.

Date Column's value should be in the Date (Year-Month-Date)format.

Time Column's value should be in the Time (Hour:Minute:Second.) format.

Date/Time Column's value should be in the Date and Time (Year-Month-Date Hour:Minute:Second) format.

- ## Default Inverse


Add Process (Default Inverse.)

Function Name

Default Inverse

Enabled

(● True) ○ False

Input Column Names: (choose column name(s) to be used in this feature)

Row
Sample
Anon Student Id
Problem Hierarchy
Problem Name

Output Column Names use [,] to seperate each columns

True Value

False Value

Figure 37: Default Inverse function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

False Value assigned to the result in the **Output Column Name** if operation returns a false.

- ## Default ListUniques

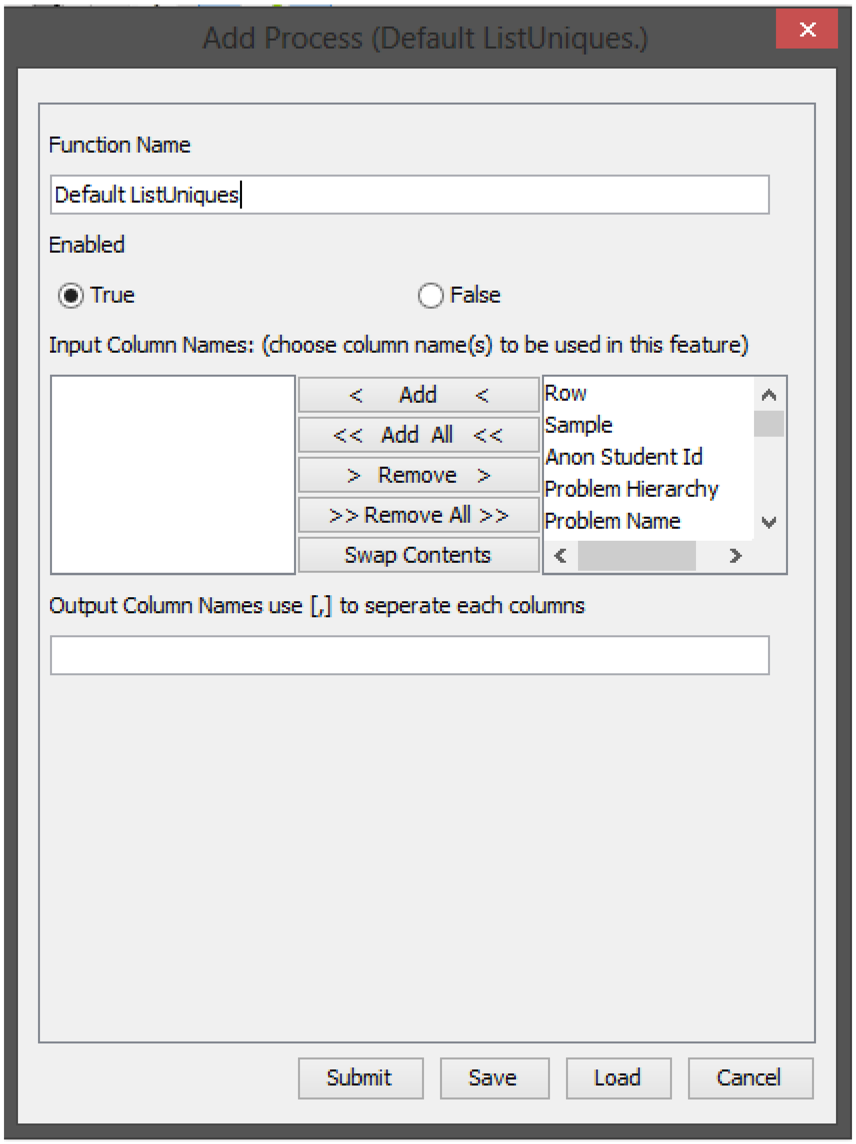
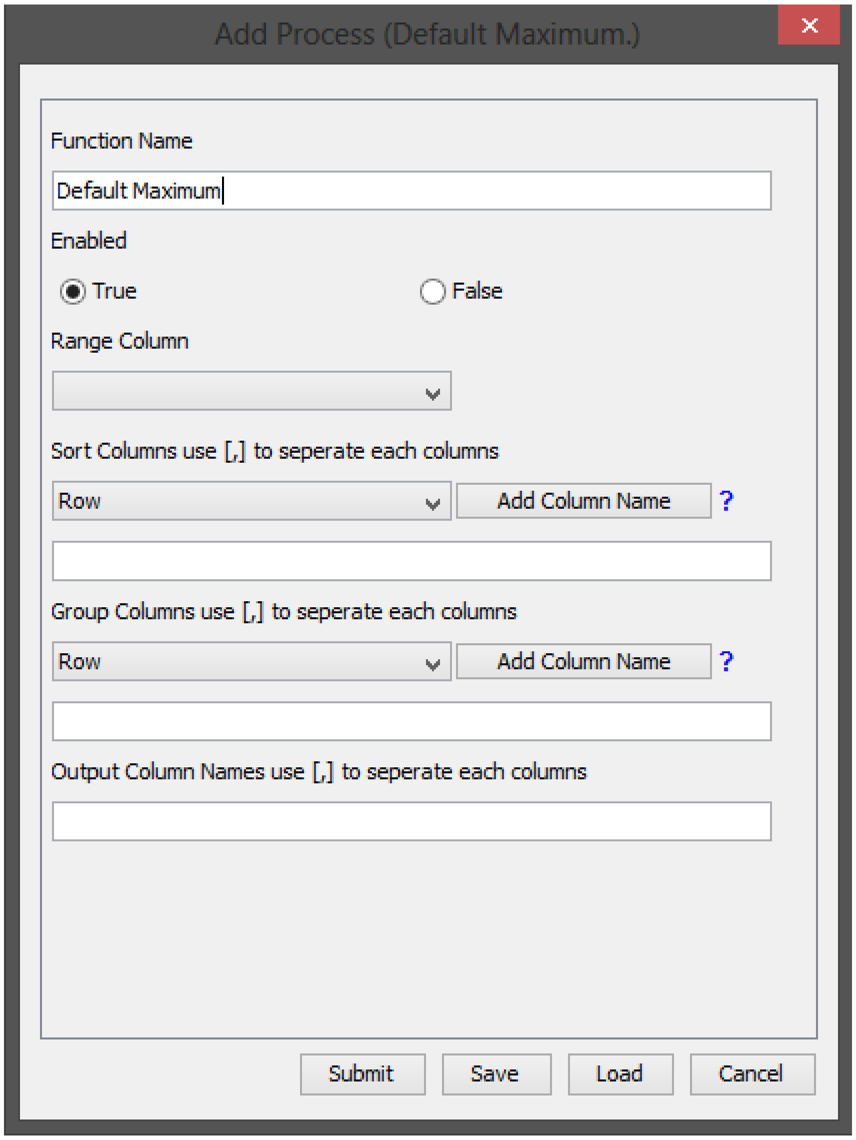


Figure 38: Default ListUniques function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

• Default Maximum


Add Process (Default Maximum.)

Function Name

Default Maximum

Enabled

[✓ True] False

Range Column

Sort Columns use [,] to seperate each columns

Row

Group Columns use [,] to seperate each columns

Row

Output Column Names use [,] to seperate each columns

Figure 39: Default Maximum function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column - used for sorting the rows within
the same group.

Group Column - Used for grouping rows with the same values for selected columns.

- **Default Mean**

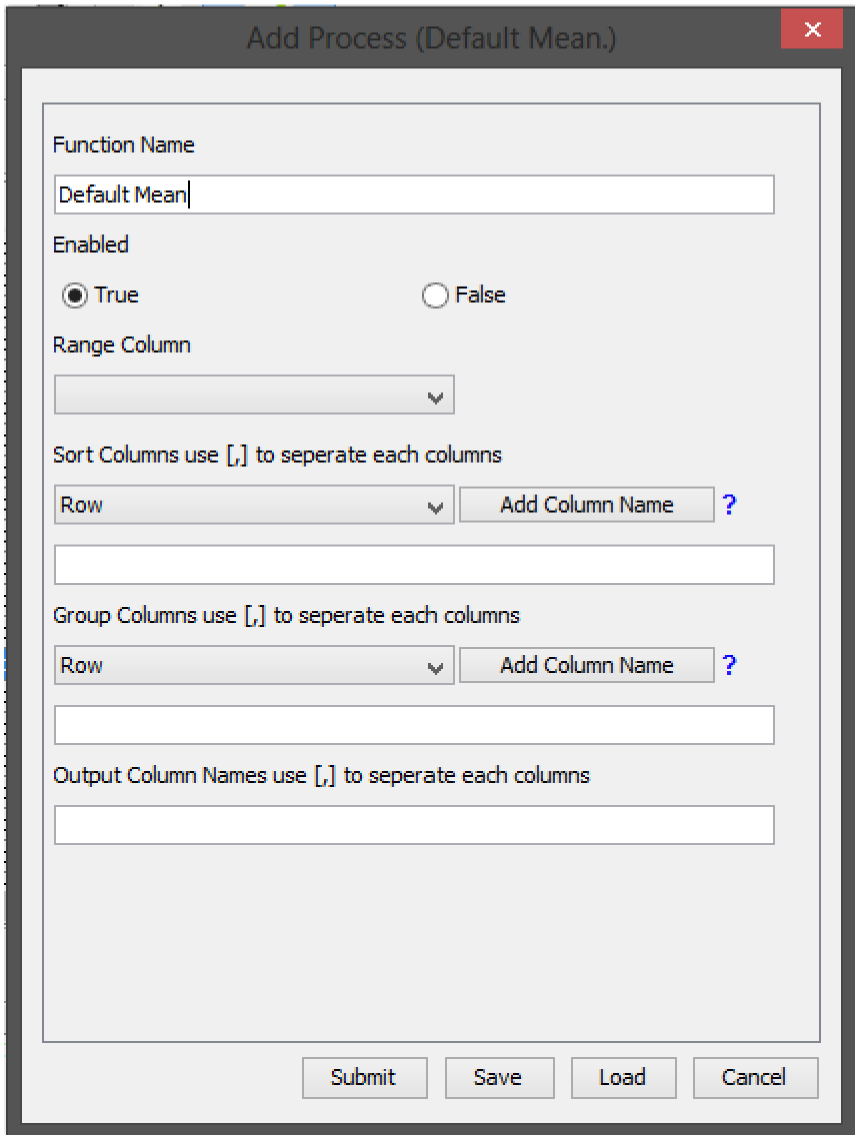
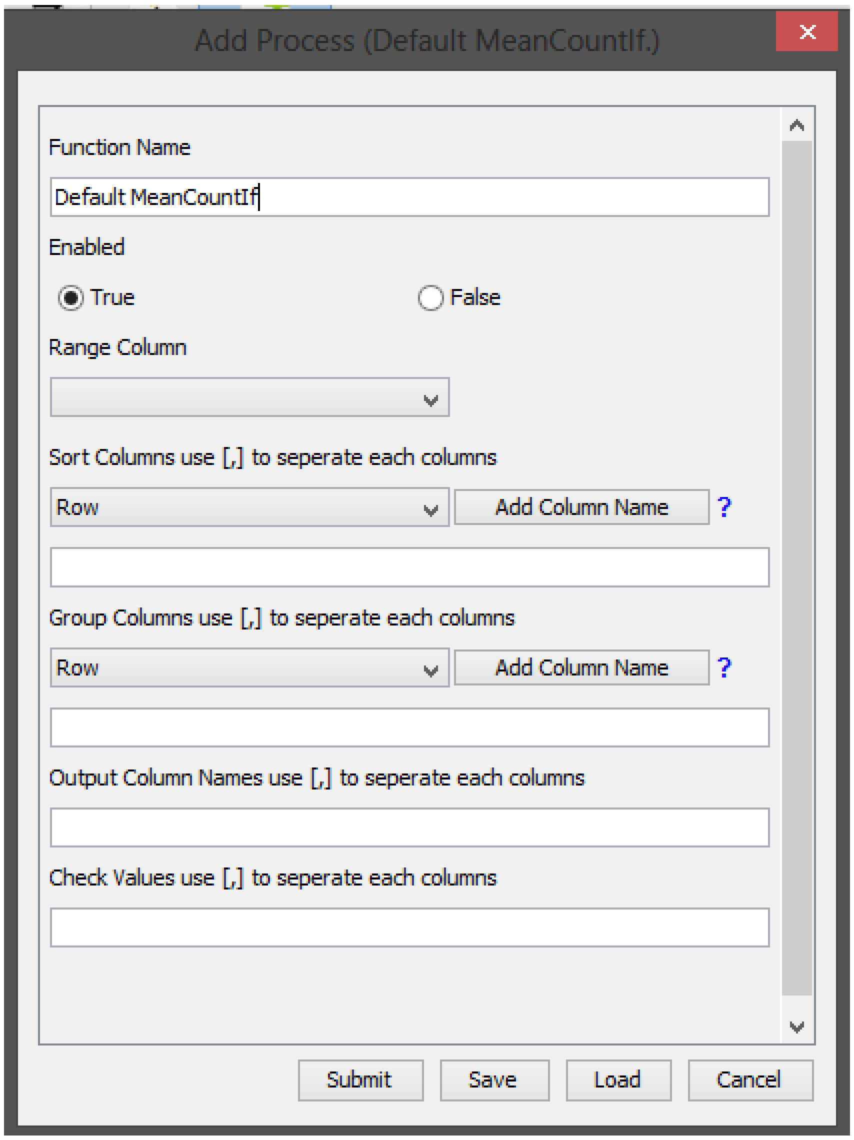

Add Process (Default Mean.)
Function Name
Default Mean
Enabled
True
False
Range Column
Sort Columns use [,] to seperate each columns
Row
Add Column Name
?
Group Columns use [,] to seperate each columns
Row
Add Column Name
?
Output Column Names use [,] to seperate each columns
Submit
Save
Load
Cancel

Figure 40: Default Mean function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.
***Sort Column* -** used for sorting the rows within
the same group.
***Group Column* -** Used for grouping rows with the same values for selected columns.

- **Default MeanCountIf**


Add Process (Default MeanCountIf.)

Function Name

Default MeanCountIf

Enabled

(● True) ○ False

Range Column

Sort Columns use [,] to seperate each columns

Row

Group Columns use [,] to seperate each columns

Row

Output Column Names use [,] to seperate each columns

Check Values use [,] to seperate each columns

Figure 41: Default MeanCountIf function window

Parameters Needed:

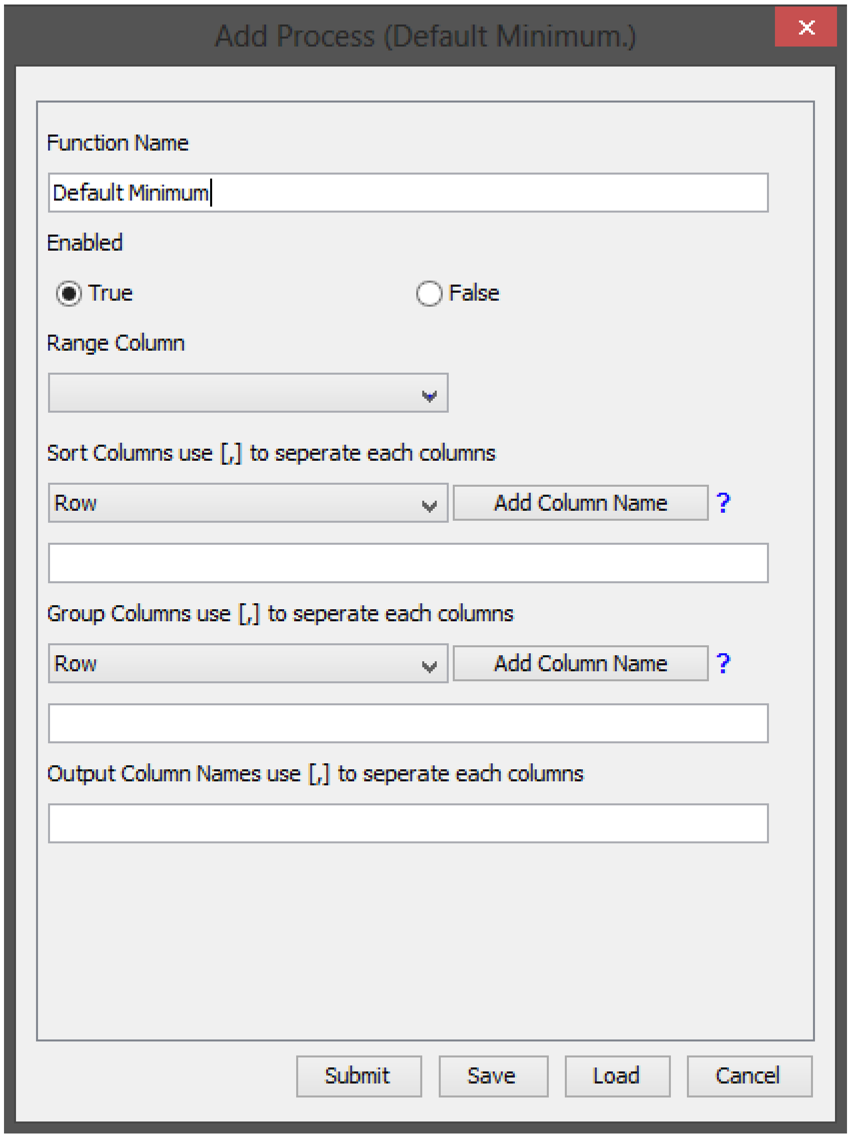
Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column **-** used for sorting the rows within
the same group.

***Group Column* -** Used for grouping rows with the same values for selected columns.

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

- ## Default Minimum



Function Name: Default Minimum

Enabled: [✓ True] False

Range Column:

Sort Columns use [,] to seperate each columns: Row

Group Columns use [,] to seperate each columns: Row

Output Column Names use [,] to seperate each columns:

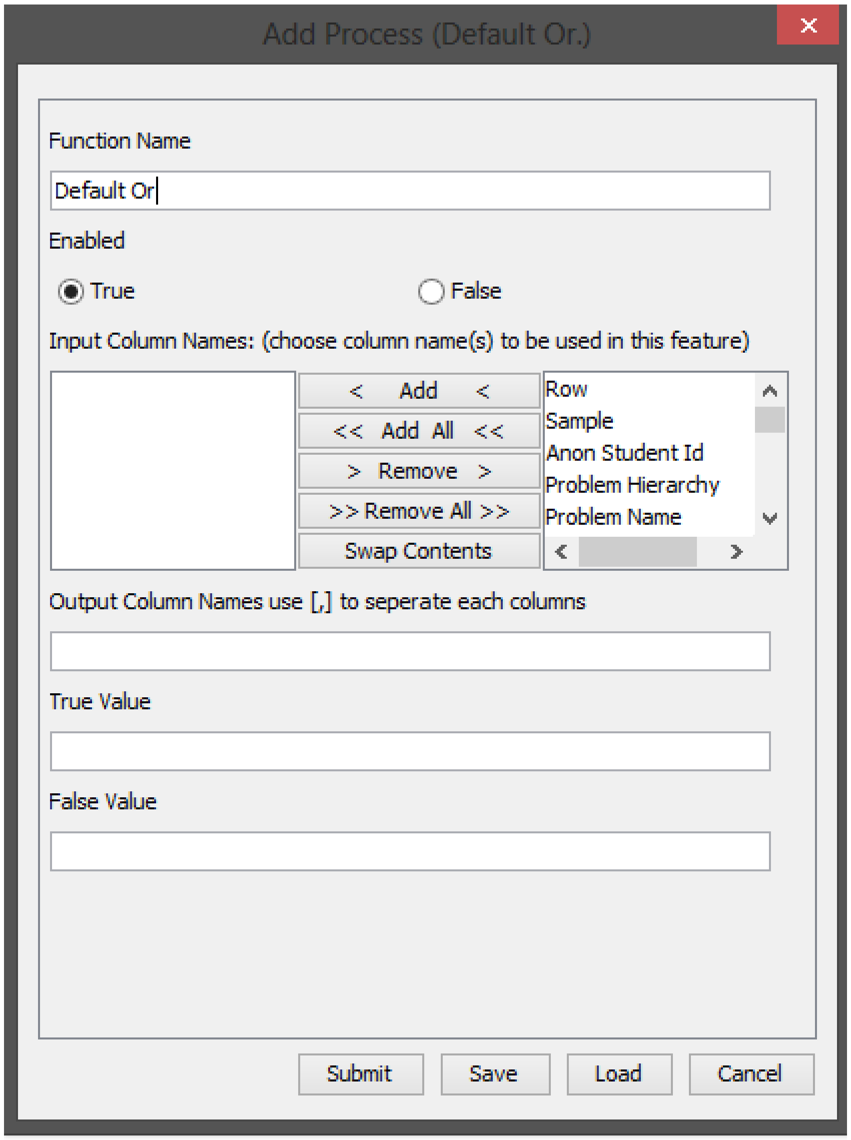
Figure 42: Default Minimum function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

***Sort Column* -** used for sorting the rows within
the same group.
***Group Column* -** Used for grouping rows with the same values for selected column.

- ## Default Or


Add Process (Default Or.)

Function Name

Default Or

Enabled

(● True) ○ False

Input Column Names: (choose column name(s) to be used in this feature)

Row
Sample
Anon Student Id
Problem Hierarchy
Problem Name

Output Column Names use [,] to seperate each columns

True Value

False Value

Figure 43: Default Or function window

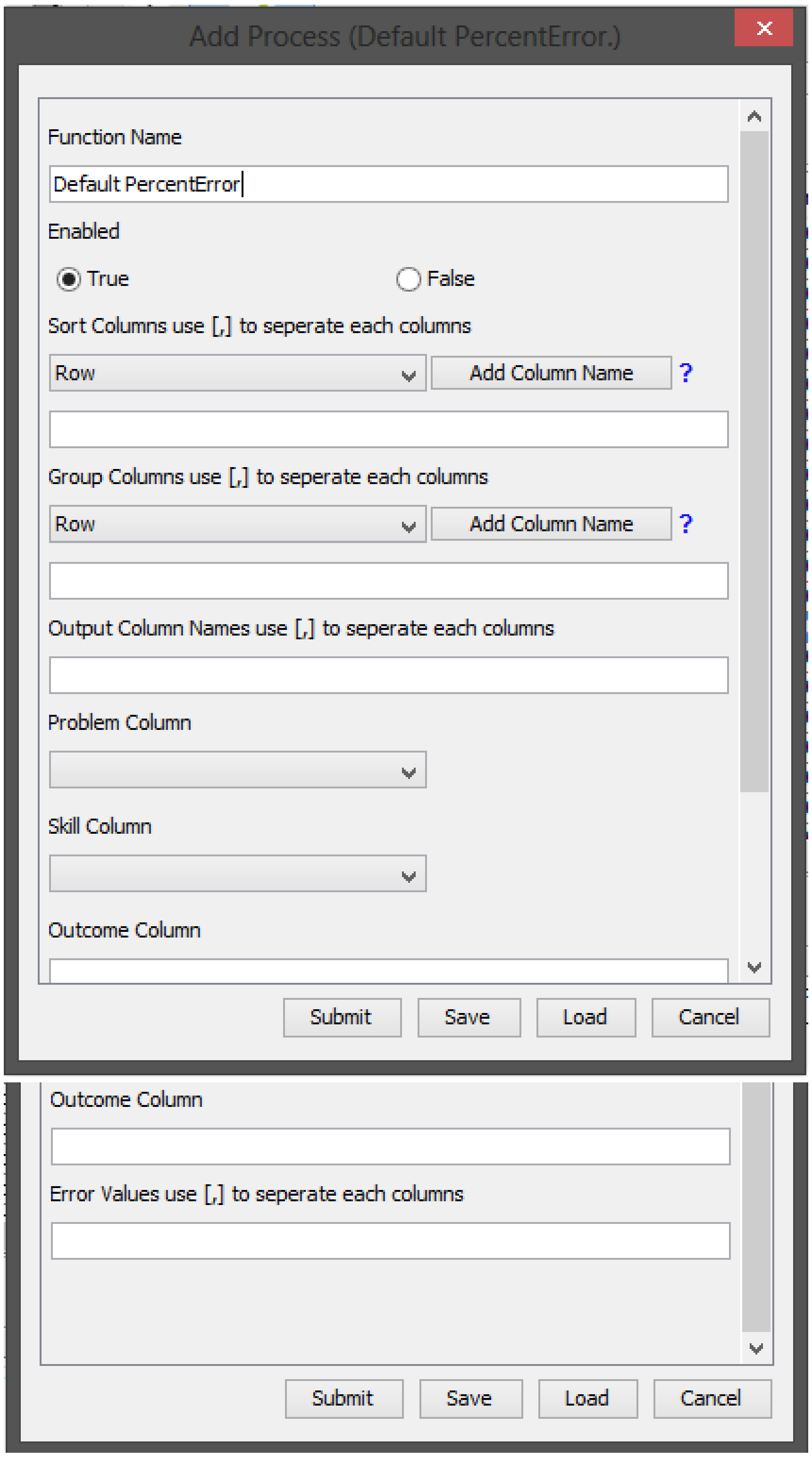
Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

True Value assigned to the result in the **Output Column Name** if operation returns a true.

False Value assigned to the result in the **Output Column Name** if operation returns a false.

- ## Default PercentError



Add Process (Default PercentError.)

Function Name

Default PercentError

Enabled

[✓ True] False

Sort Columns use [,] to seperate each columns

Row — Add Column Name ?

Group Columns use [,] to seperate each columns

Row — Add Column Name ?

Output Column Names use [,] to seperate each columns

Problem Column

Skill Column

Outcome Column

Error Values use [,] to seperate each columns

Figure 44: Default PercentError function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

***Sort Column* -** used for sorting the rows within
the same group.

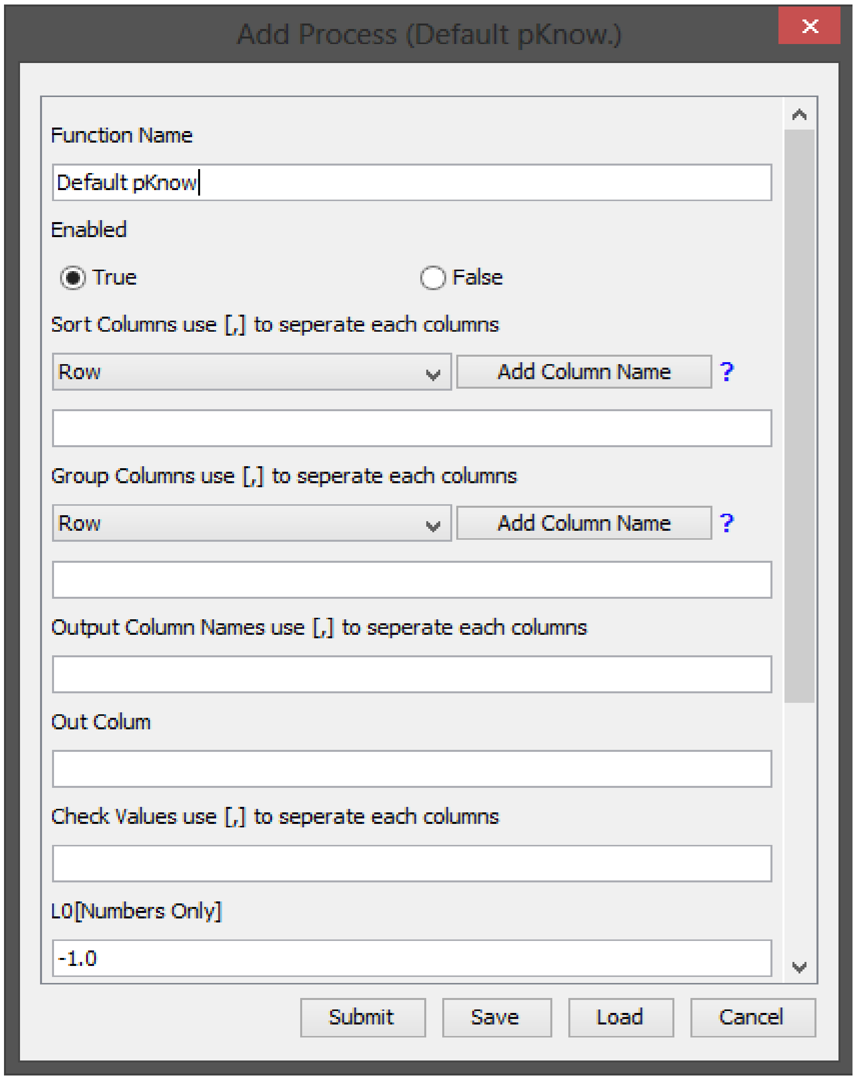
***Group Column* -** Used for grouping rows with the same values for selected column.

Problem Column – name of the column corresponding to the problem

Skill Column – name of the column specifying the skill

***Error Values* -** used to specify which values constitute an error for use by percentError.

• Default pKnow



Add Process (Default pKnow.)

Function Name

Default pKnow

Enabled

(● True) ○ False

Sort Columns use [,] to seperate each columns

Row | Add Column Name | ?

Group Columns use [,] to seperate each columns

Row | Add Column Name | ?

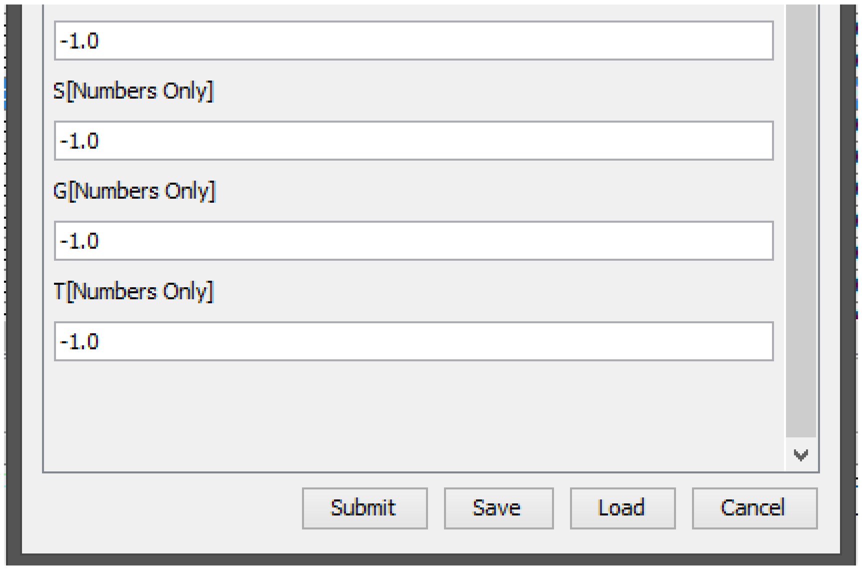
Output Column Names use [,] to seperate each columns

Out Colum

Check Values use [,] to seperate each columns

L0[Numbers Only]

-1.0



-1.0

S[Numbers Only]

-1.0

G[Numbers Only]

-1.0

T[Numbers Only]

-1.0

Figure 45: Default pKnow function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

***Sort Column* -** used for sorting the rows within
the same group.

***Group Column* -** Used for grouping rows with the same values for selected column.

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

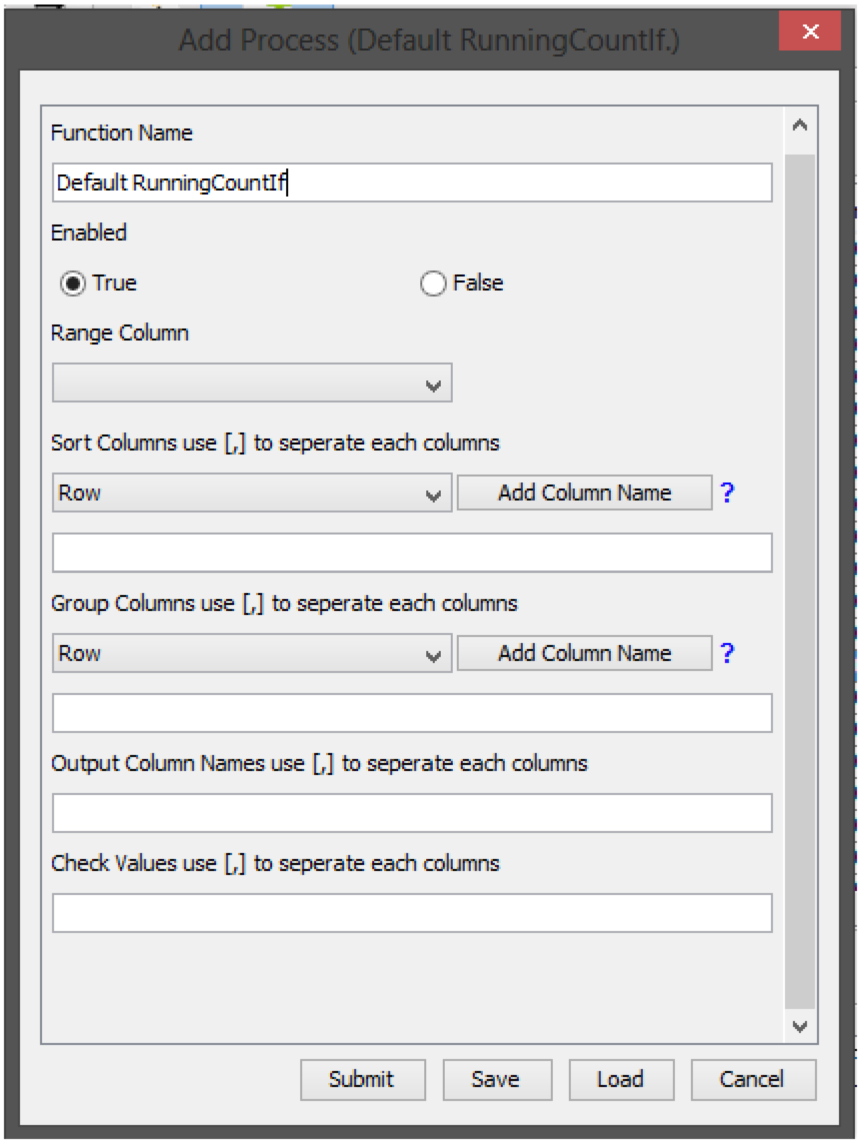
L0[Number Only] – probability that the skill is already known before the first instance in using the skill in problem solving.

S[Number Only] – probability that the student will commit a fault if the skill was already known beforehand

G[Number Only] – probability that the student will deduce the correct answer given that skill is not known.

T[Number Only] - probability that the skill will be learned at each opportunity to use the skill, regardless whether the answer is correct or incorrect.

- **Default RunningCountIf**


Add Process (Default RunningCountIf.)

Function Name

Default RunningCountIf

Enabled

(● True) ( ) False

Range Column

Sort Columns use [,] to seperate each columns

Row | Add Column Name | ?

Group Columns use [,] to seperate each columns

Row | Add Column Name | ?

Output Column Names use [,] to seperate each columns

Check Values use [,] to seperate each columns

Submit | Save | Load | Cancel

Figure 46: Default RunningCountIf function window

Parameters Needed:

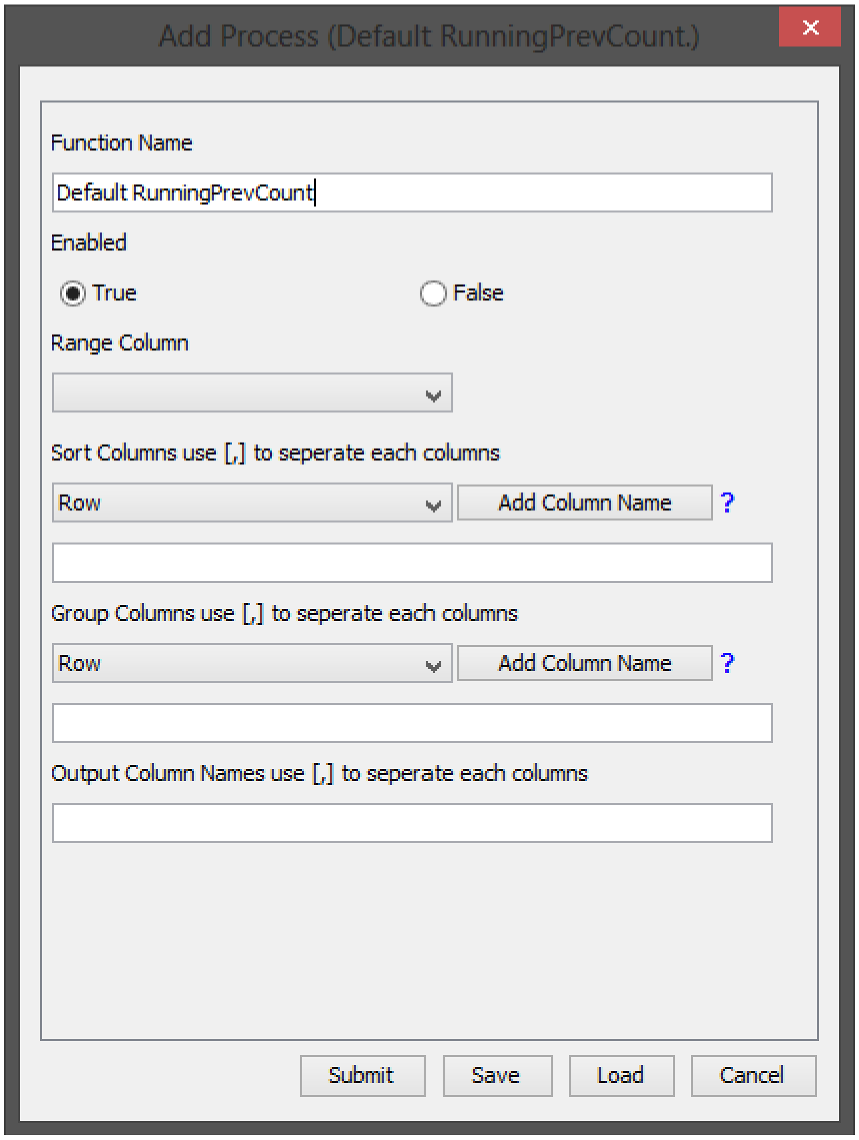
Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column **-** used for sorting the rows within
the same group.

***Group Column* -** Used for grouping rows with the same values for selected column.

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

- ## Default RunningPrevCount



Function Name

Default RunningPrevCount

Enabled

(● True) ○ False

Range Column

Sort Columns use [,] to seperate each columns

Row

Group Columns use [,] to seperate each columns

Row

Output Column Names use [,] to seperate each columns

Figure 47: Default RunningPrevCount window

Parameters Needed:

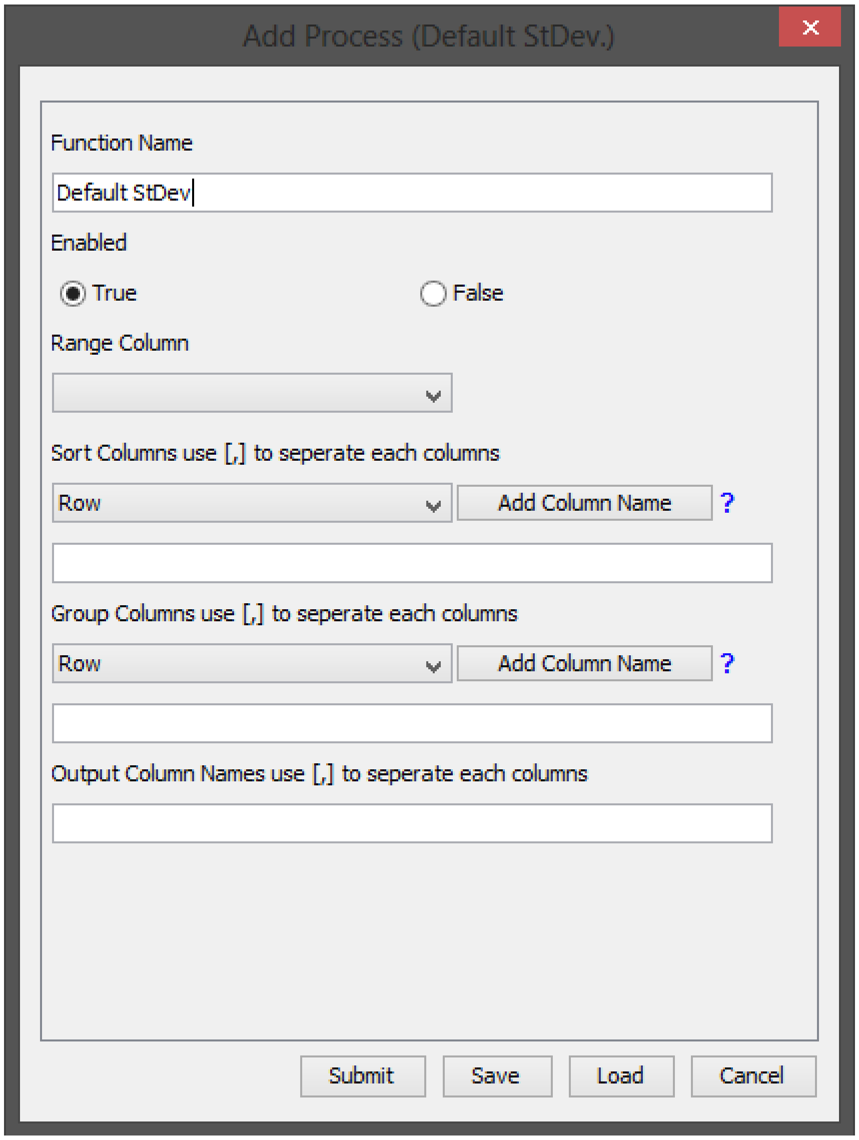
Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column - used for sorting the rows within the same group.

Group Column - Used for grouping rows with the same values for selected column.

Range Column - Range of values used for computation.

- ## Default StDev


Add Process (Default StDev.)

Function Name: Default StDev

Enabled: True [✓] False

Range Column:

Sort Columns use [,] to seperate each columns: Row

Group Columns use [,] to seperate each columns: Row

Output Column Names use [,] to seperate each columns:

Figure 48: Default StDev function window

Parameters Needed:

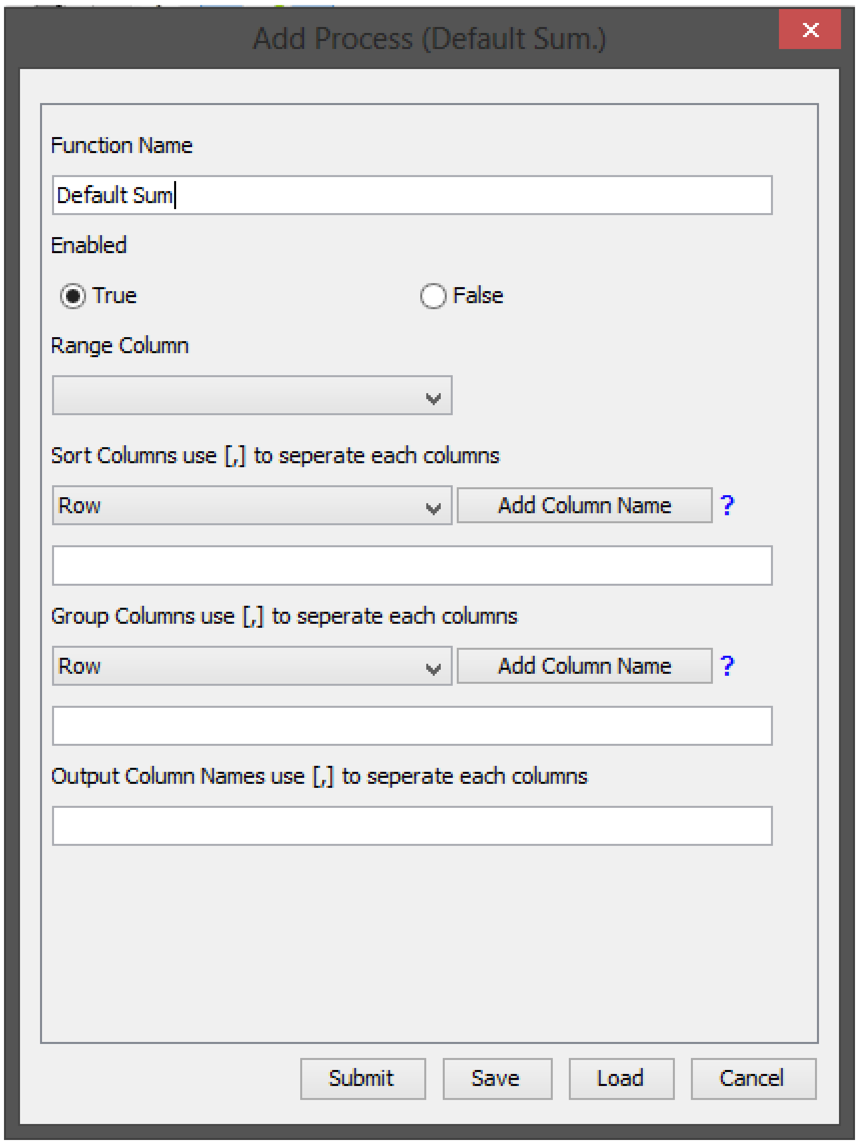
Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column - used for sorting the rows within the same group.

Group Column - Used for grouping rows with the same values for selected column.

Range Column - Range of values used for computation.

- ## Default Sum


Add Process (Default Sum.)

Function Name

Default Sum

Enabled

(● True) ○ False

Range Column

Sort Columns use [,] to seperate each columns

Row | Add Column Name | ?

Group Columns use [,] to seperate each columns

Row | Add Column Name | ?

Output Column Names use [,] to seperate each columns

Figure 49: Default Sum function window

Parameters Needed:

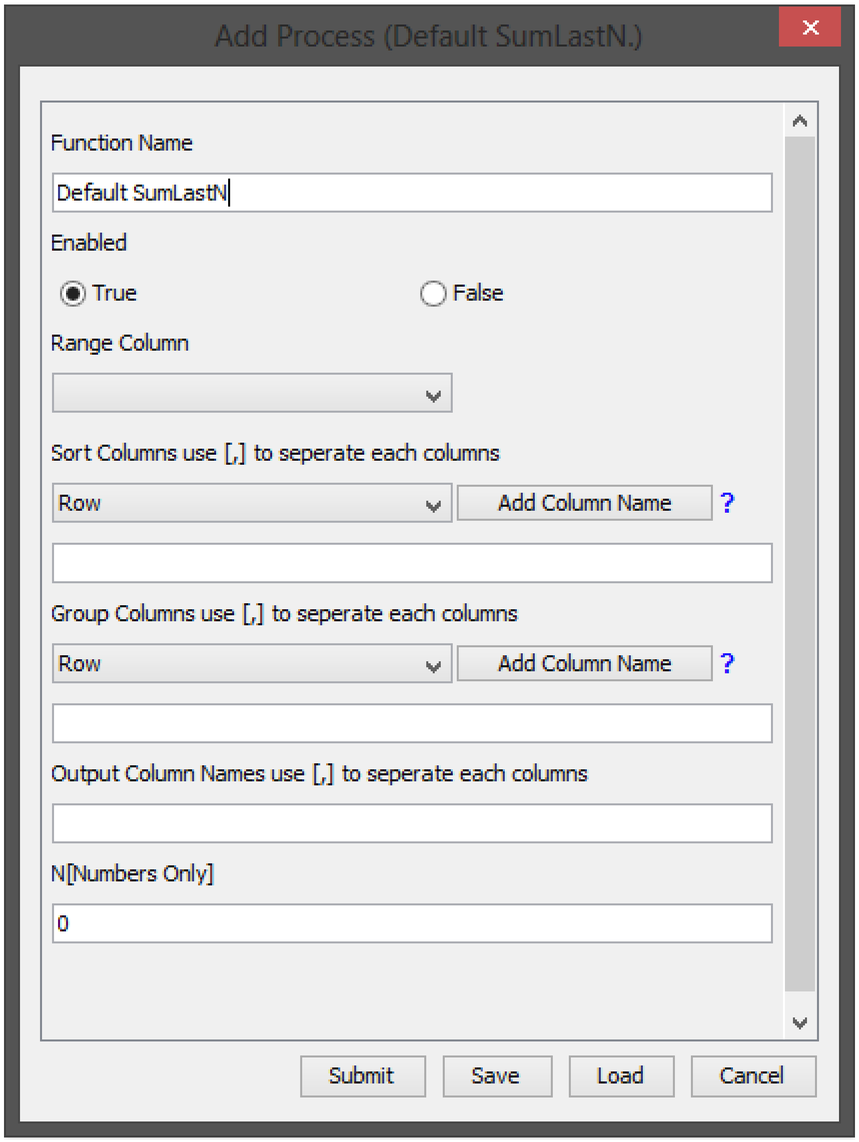
Enabled indicates whether to the selected feature will be used in the process or not.

Sort Column - used for sorting the rows within the same group.

Group Column - Used for grouping rows with the same values for selected column.

Range Column - Range of values used for computation.

- **Default SumLastN**


Add Process (Default SumLastN.)

Function Name: Default SumLastN

Enabled: [✓ True] False

Range Column:

Sort Columns use [,] to seperate each columns: Row

Group Columns use [,] to seperate each columns: Row

Output Column Names use [,] to seperate each columns:

N[Numbers Only]: 0

Figure 50: Default SumLastN function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

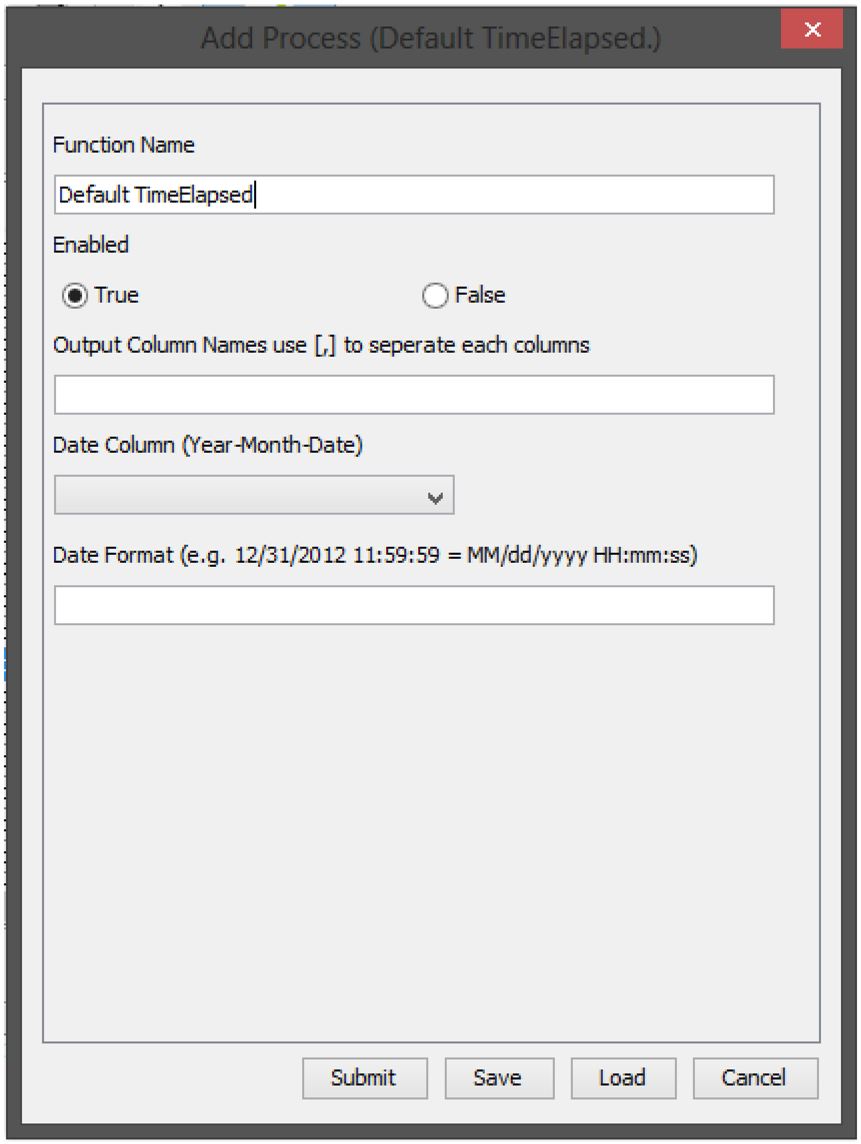
Sort Column - used for sorting the rows within the same group.

Group Column - Used for grouping rows with the same values for selected column.

Range Column - Range of values used for computation.

N[Numbers Only] if more elements in a group are found, only the last N items are kept for processing/start count every N rows.

- **Default TimeElapsed**



Function Name: Default TimeElapsed

Enabled: [✓ True] False

Output Column Names use [,] to seperate each columns

Date Column (Year-Month-Date)

Date Format (e.g. 12/31/2012 11:59:59 = MM/dd/yyyy HH:mm:ss)

Figure 51: Default TimeElapsed function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

Date Column's value is the date when the actions were taken/ time stamp.

Date Format is the format of the Date Column where:

M=month	H=hour
d=day	m=minutes
y=year	s=seconds

e.g. 31/12/12 11:59 = dd/MM/yy HH:mm
12/31/2012 11:59:59 = MM/dd/yyyy HH:mm:ss

- ## Default TimeSD

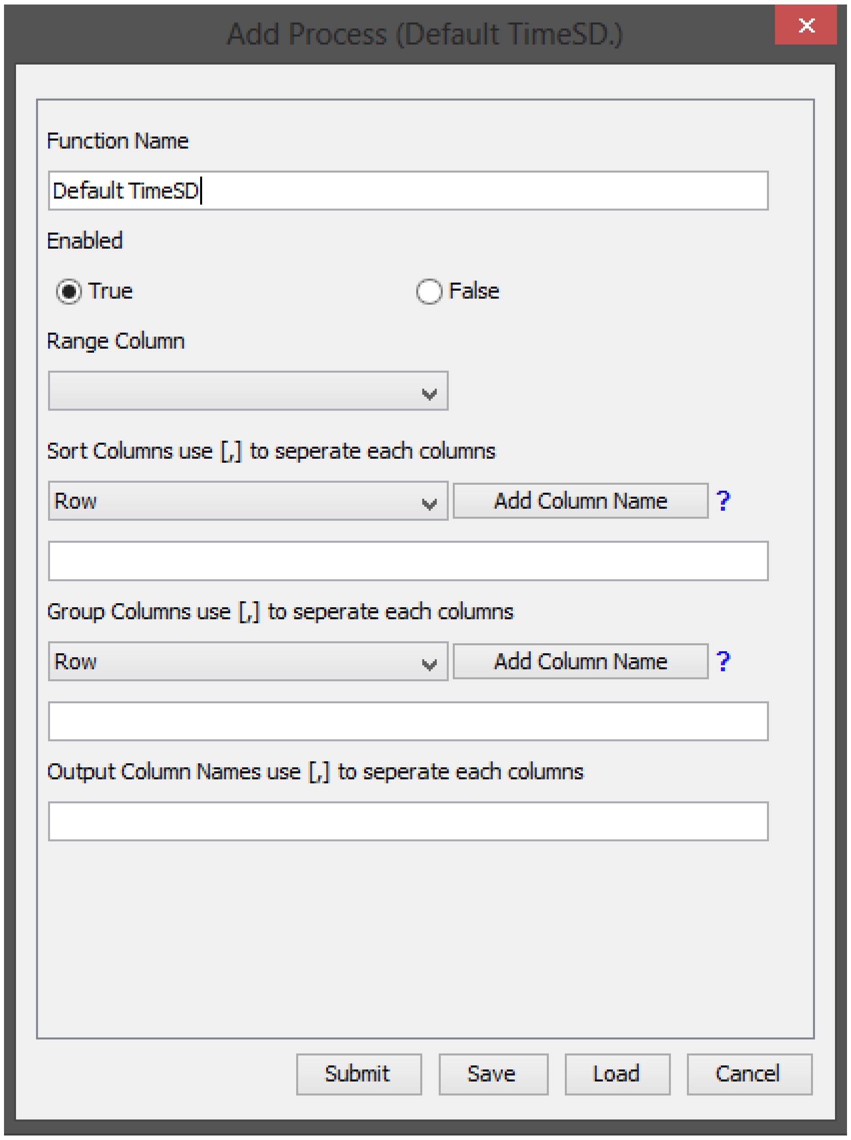

Add Process (Default TimeSD.)
Function Name
Default TimeSD
Enabled
True
False
Range Column
Sort Columns use [,] to seperate each columns
Row
Add Column Name
?
Group Columns use [,] to seperate each columns
Row
Add Column Name
?
Output Column Names use [,] to seperate each columns
Submit
Save
Load
Cancel

Figure 52: Default TimeSD function window

Parameters Needed:

Enabled indicates whether to the selected feature will be used in the process or not.

***Sort Column* -** used for sorting the rows within
the same group.

***Group Column* -** Used for grouping rows with the same values for selected column.

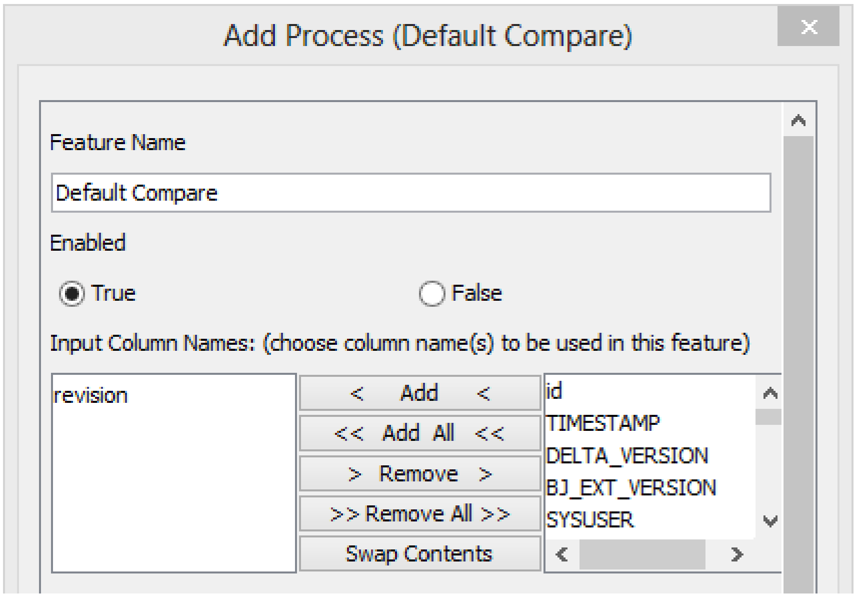
***Range Column* -** Range of values used for computation.

- ### Add Feature Buttons
  - #### Submit Button
    The submit button will execute the feature set by the user
  - #### Save Button
    The save button will save the user selected properties to a file to allow the same values to be used again later.
  - #### Load Button
    The load button allows the user to reload a template.
  - #### Cancel Button
    This cancels the selected feature and removes it from the process list.
- ### Add Feature Parameters

To add a new feature, the user will have to set several parameters. Depending on the operation that the user needs to perform, the user will have to supply a subset of the parameters listed below.

Input Column Names lists the selected values. The user can remove and/or add values to the columns.

Click one or multiple items and click **<Add<** to add the value(s) or click **<<Add All<<** to add all column name. Click **>Remove>** to delete one or multiple input column name or **>>Remove All>>** to remove all input column names.



Add Process (Default Compare)

Feature Name

Default Compare

Enabled

(● True) ○ False

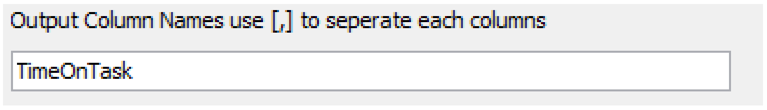
Input Column Names: (choose column name(s) to be used in this feature)

revision

id
TIMESTAMP
DELTA_VERSION
BJ_EXT_VERSION
SYSUSER

Figure 53: Sample add feature window

Output Column Names are columns added later in the Datagrid after the user-selected values have been processed. These columns will also be included in the **Required Columns** in the **Add Process Window** (Figure 54)**.**



Output Column Names use [,] to seperate each columns

TimeOnTask

Figure 54: Selection of column names

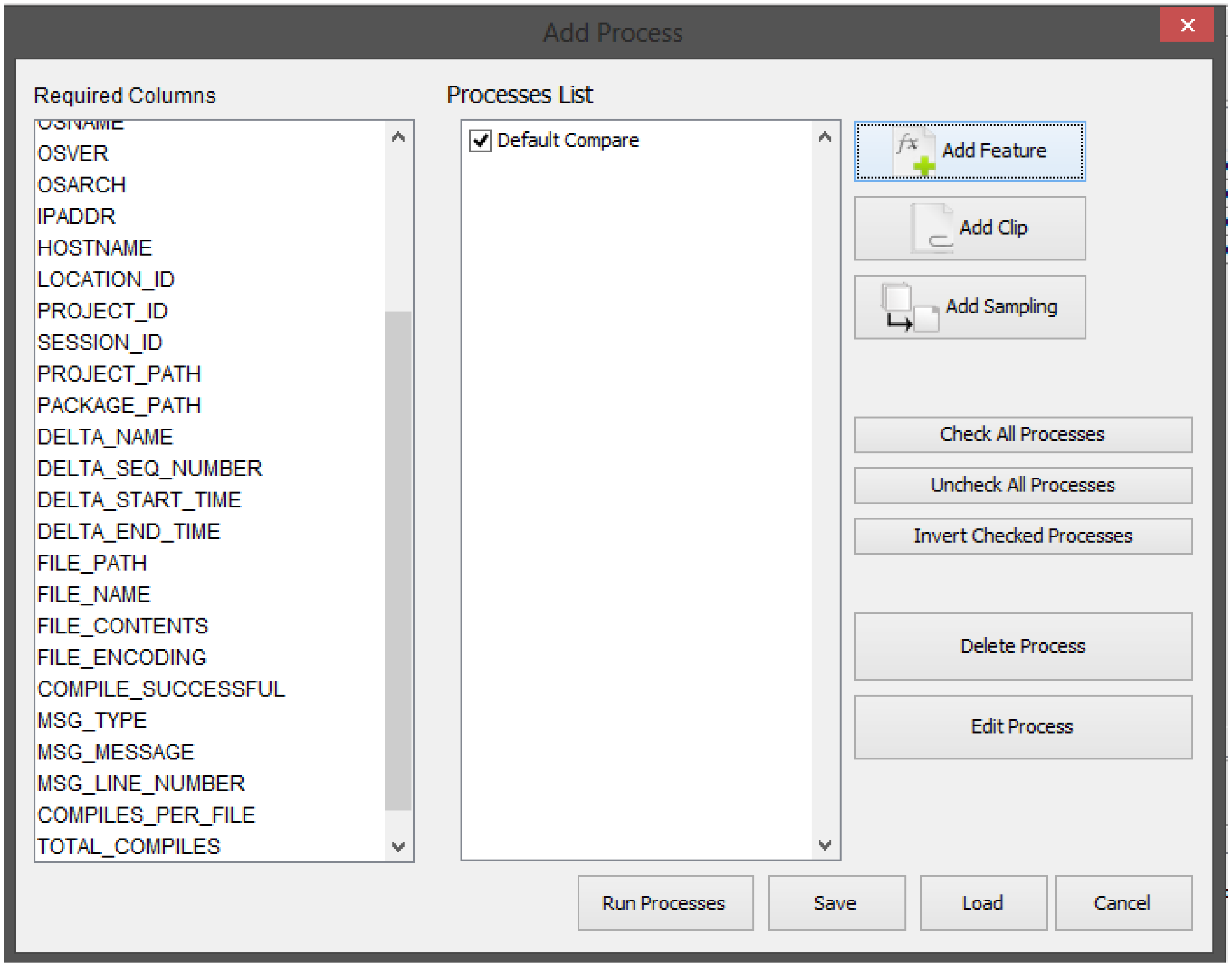
Feature Name is the name to be displayed in the Process List (see Figure 53).

Enabled indicates whether the selected feature will be used in the process or not. In Figure 31 the **Enabled** option was set to true. After submission, we now see that the feature is checked in the process list (see Figure 53).

True Value assigned to the result in the **Output Column Name** if operation returns a true. (see Figure 53).

False Value assigned to the result in the **Output Column**

Name if operation returns a false. (see figure 53).



OSNAME
OSVER
OSARCH
IPADDR
HOSTNAME
LOCATION_ID
PROJECT_ID
SESSION_ID
PROJECT_PATH
PACKAGE_PATH
DELTA_NAME
DELTA_SEQ_NUMBER
DELTA_START_TIME
DELTA_END_TIME
FILE_PATH
FILE_NAME
FILE_CONTENTS
FILE_ENCODING
COMPILE_SUCCESSFUL
MSG_TYPE
MSG_MESSAGE
MSG_LINE_NUMBER
COMPILES_PER_FILE
TOTAL_COMPILES

✓ Default Compare

Figure 55: Add Feature Window with updated column

Check Value is the value to be compared against the **Selected Input Column Names**. This value can either be a string or integer depending on the feature used.

Operation Type contains values from 1-6 that correspond to different operations. Strings or integers can be compared in this feature.

- *Example:* Compare feature was the selected feature. The Check Value will be compared to the Selected Column Name and the output will depend on what operation selected below.

1 - _Greater than_ operation
2 – _Greater than or Equal to_ operation
3 – _Less than_ operation
4 – _Less than or Equal to_ operation
5 – _Equal to_ operation
6 – _Starts with_ operation

Date Column's value should be in the Date (Year-Month-Date)format.

Time Column's value should be in the Time (Hour:Minute:Second.) format.

Date/Time Column's value should be in the Date and Time (Year-Month-Date Hour:Minute:Second) format.

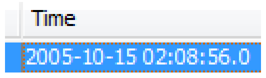

Time
2005-10-15 02:08:56.0

Figure 56: Time in (YYYY/MM/DD/HH/MM/SS)

All String checks if all the column values are strings, not numbers or any other type.

pKnowColumn's value should be the **pKnow** column. Calculate first the **pKnow** value using **pKnow** operation. Afterwards, use **pKnowDirect** with the **pKnow** value.

N[Numbers Only] if more elements in a group are found, only the last N items are kept for processing/start count every N rows??

***Range Column* -** Range of values used for computation.

***Group Column* -** Used for grouping rows with the same values for selected columns.

***Sort Column* -** used for sorting the rows within the same group.

Problem Column – name of the column corresponding to the problem

Skill Column – name of the column specifying the skill

Outcome Column – name of the column used by certain features

***Error Values* -** used to specify which values constitute an error for use by percentError.

L0[Number Only] – probability that the skill is already known before the first instance in using the skill in problem solving.

S[Number Only] – probability that the student will commit a fault if the skill was already known beforehand

G[Number Only] – probability that the student will deduce the correct answer given that skill is not known.

T[Number Only] - probability that the skill will be learned at each opportunity to use the skill, regardless whether the answer is correct or incorrect.

Attempt Column - Either of the two (depends on how it was used): "Is this the first attempt of the student to answer or get help on the problem step? ", or "How many attempts did they answer or ask for help on the problem step?"

Pre-defined functions

The system has 23 default operations available. Four parameters are common to all operations.

- Input Column Names
- Output Column Names
- Feature Name
- Enabled

Listed below are the current operations, their descriptions and parameters needed aside from the previously mentioned parameters.

Function	***Description(s)***	***Other Parameters Needed***
1. And	Executes a logical AND operation on the selection and returns the corresponding Boolean results.	- True Value - False Value
2. Compare	Compares if two values are identical. (Compare 1st selected Input Column Name with Check Values and its output is based on the Operation type used)	- Check Values - All Strings - Operation Type
3. Copy	Copy the values from a column (Values from Selected Input Column Name)	- None
4. CountIfLastN	Counts how many in the last n entries (including the current cell) are equal to a given value or values.	- Sort Columns - Group Columns - Range Columns - N[Numbers Only] - Check Values

5. CountLastN	Counts how many in the last n entries (including the current cell) are equal to the current cell.	- Sort Columns - Group Columns - Range Columns - N[Numbers Only]
6. Duration	Computes how many seconds the action took.	- Sort Columns - Group Columns - Date Column - Time Column - Date/Time Column
7. First Attempt	Determines if it is the first attempt.	- True Value - False Value - Group Columns - Date Column - Time Column - Date/Time Column
8. Inverse	Returns the inverse of a Boolean. If the column values equal the true value, return the false value instead and vice versa.	- True Value - False Value
9. ListUnique	Creates a new column with all the unique data from the selection.	- None
10. Maximum	Determines the maximum value in the selection provided.	- Sort Columns - Group Columns - Range Column
11. Mean	Computes the arithmetic mean of all the values in the selection.	- Sort Columns - Group Columns - Range Column
12. MeanCountIf	Computes the average number of entries that are equal to a given value or values, over all entries.	- Sort Columns - Group Columns - Range Column - Check Value
13. Minimum	Determines the minimum value in the selection provided.	- Sort Columns - Group Columns - Range Column
14. Or	Executes a logical OR operation and returns the corresponding Boolean results.	- True Value - False value

15. PercentError	Computes the percentage of past problems where errors were made on a skill.	- Sort Column - Group Colum - Problem Column - Skill Column - Outcome Column - Error Values
16. pKnow	Computes for the probability that the student knows the skill involved in an action.	- Sort Columns - Group Columns - Out Column - Check Values - L0[Numbers Only] - S[Numbers Only] - G[Numbers Only] - T[Numbers Only]
17. pKnowDirect	Checks if the current action is the student's first attempt on this problem step. If true, pknow-direct is equal to pknow; otherwise, pknow-direct is equal to -1.	- Attempt Column - pKnow Column - Check Value - False Value
18. RunningCountif	Computes the number of entries that are equal to a given value or values, up to the current cell, including the current cell.	- Sort Columns - Group Columns - Range Column - Check Value
19. RunningPrevCount	Computes the number of entries that are equal to the current cell, up to the cell before the current cell.	- Sort Columns - Group Columns - Range Column
20. StDev	Computes the standard deviation of a specified column.	- Sort Columns - Group Columns - Range Column
21. SumLastN	Computes the sum of the last n numbers in the selection specified.	- Sort Columns - Group Columns - Range Column - N[Numbers Only]
22. TimeSD	Computes time taken in terms of number of standard deviations from mean time.	- Sort Columns - Group Columns - Range Column

23. timeElapsed	Computes for the time interval per action in seconds (date of current row minus the date of the first row)	- Output Column - Date Column - Date Format

Figure 57: Function List

Submit Button will include the user-selected feature to the Process List.

Load Button will load available features.

Save Button will save the user-selected feature and add it to the directory of features for later use.

- ### Add Features in the Clip Level

In the clip-level, there are 5 features which can be imposed on the clips: mean, max, min, stdev, and listUnique. These features' functionalities are similar to the ones above. Clipped dataset are composed of a parent container and a dataset representing each clip. Non-clip level operations will append output columns to each of the enclosed clips; however, a clip-level operation will append output columns only to the parent container.

- ### Add Clipping

Allows user to set the desired clipping properties. The form applies the selected properties in the clipping form.

- ### Add Sampling

Allows user to set desired sampling properties. The form applies the sampling properties set in the sampling form.

- ### Cancel Button

Cancels and closes the Add Process form.

- ### Save Button

The system shall save all the properties set in the Processes List which are then checked into a process.xml file.

- ### Load Button

The system will load the all the configured processed list (process.xml) files available in the process directory upon clicking the load button.

Run Process Button

The system runs all checked processes in the process list. The system will display information feedback in the Status Bar on what process it is currently taking and throws an error dialogue when the system encounters an error.

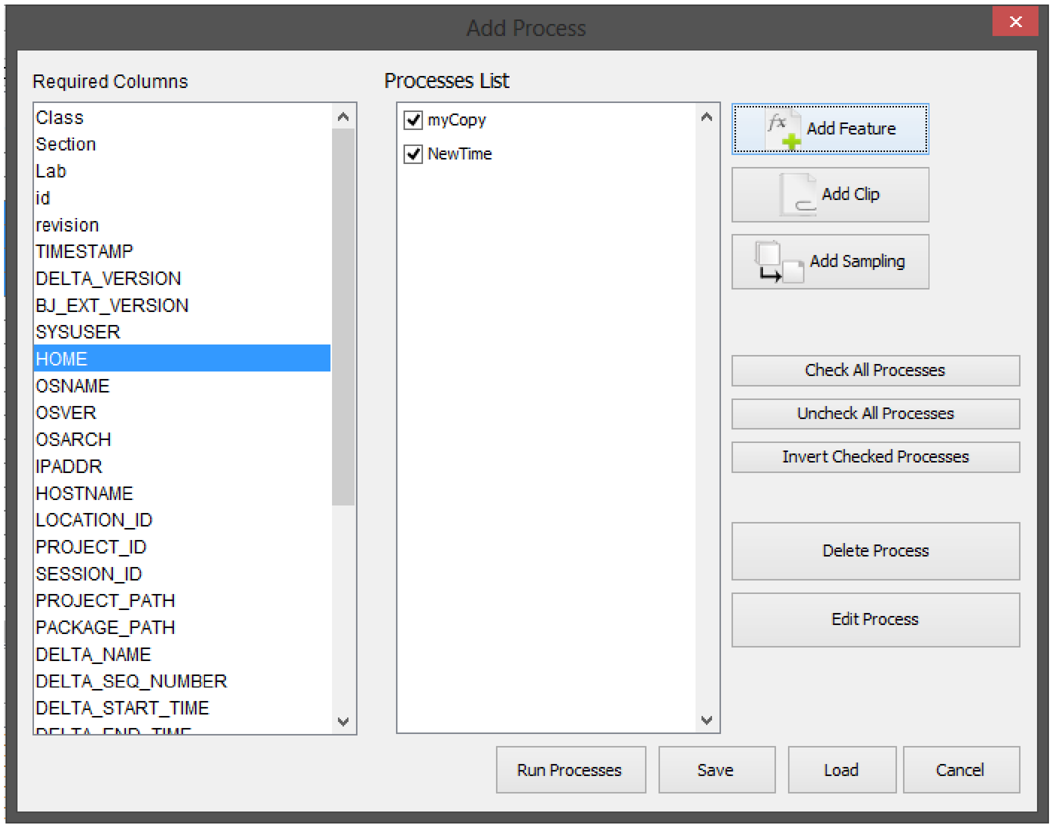
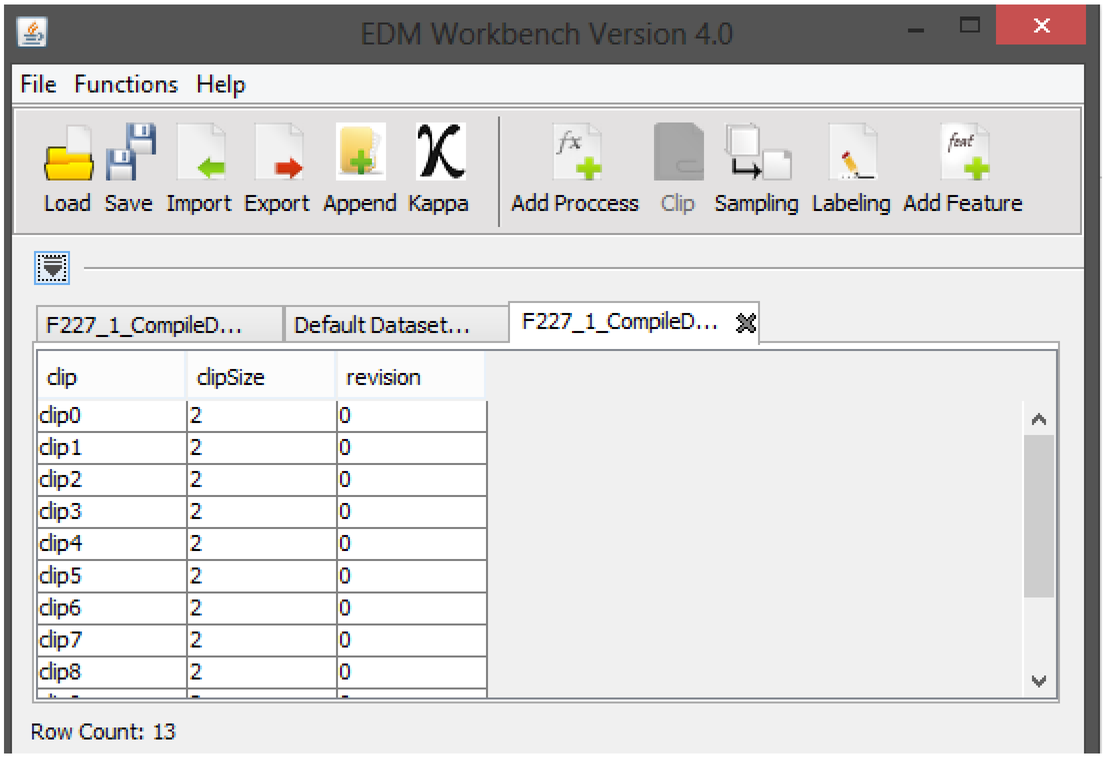


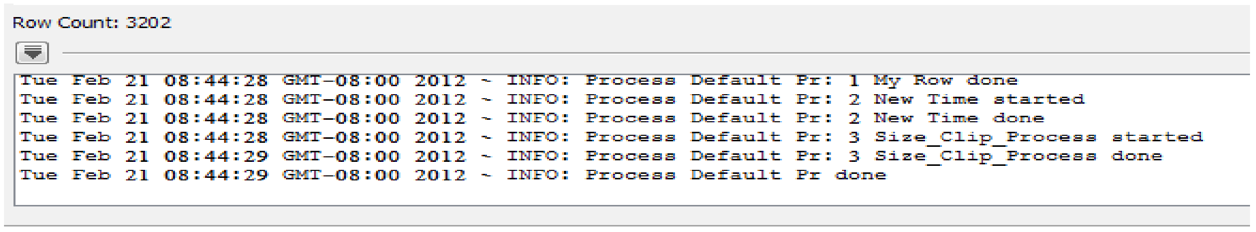
Figure 58: Sample System Process List



clip	clipSize	revision
clip0	2	0
clip1	2	0
clip2	2	0
clip3	2	0
clip4	2	0
clip5	2	0
clip6	2	0
clip7	2	0
clip8	2	0

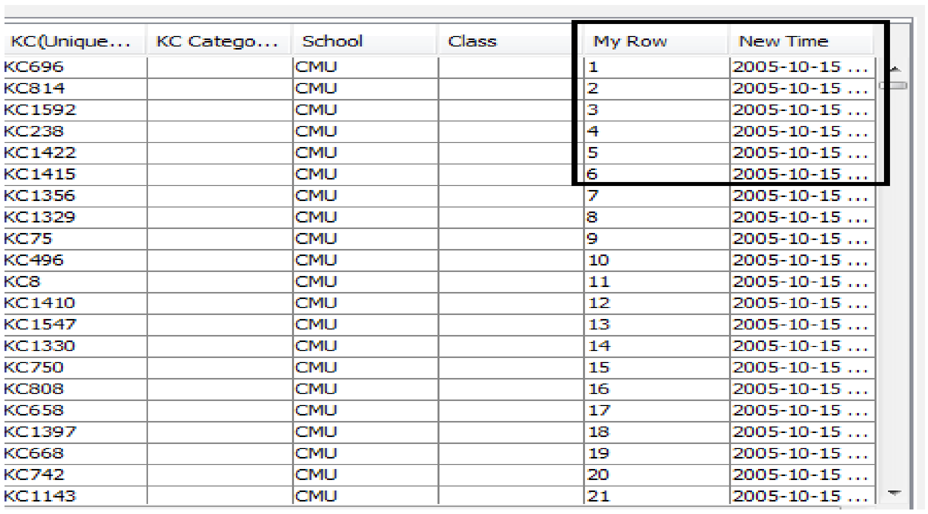
Row Count: 13

Figure 59: Sample Clipping display


Row Count: 3202

```
Tue Feb 21 08:44:28 GMT-08:00 2012 ~ INFO: Process Default Pr: 1 My Row done
Tue Feb 21 08:44:28 GMT-08:00 2012 ~ INFO: Process Default Pr: 2 New Time started
Tue Feb 21 08:44:28 GMT-08:00 2012 ~ INFO: Process Default Pr: 2 New Time done
Tue Feb 21 08:44:28 GMT-08:00 2012 ~ INFO: Process Default Pr: 3 Size_Clip_Process started
Tue Feb 21 08:44:29 GMT-08:00 2012 ~ INFO: Process Default Pr: 3 Size_Clip_Process done
Tue Feb 21 08:44:29 GMT-08:00 2012 ~ INFO: Process Default Pr done
```

Figure 60: Clipping feedback



KC(Unique...	KC Catego...	School	Class	My Row	New Time
KC696		CMU		1	2005-10-15 ...
KC814		CMU		2	2005-10-15 ...
KC1592		CMU		3	2005-10-15 ...
KC238		CMU		4	2005-10-15 ...
KC1422		CMU		5	2005-10-15 ...
KC1415		CMU		6	2005-10-15 ...
KC1356		CMU		7	2005-10-15 ...
KC1329		CMU		8	2005-10-15 ...
KC75		CMU		9	2005-10-15 ...
KC496		CMU		10	2005-10-15 ...
KC8		CMU		11	2005-10-15 ...
KC1410		CMU		12	2005-10-15 ...
KC1547		CMU		13	2005-10-15 ...
KC1330		CMU		14	2005-10-15 ...
KC750		CMU		15	2005-10-15 ...
KC808		CMU		16	2005-10-15 ...
KC658		CMU		17	2005-10-15 ...
KC1397		CMU		18	2005-10-15 ...
KC668		CMU		19	2005-10-15 ...
KC742		CMU		20	2005-10-15 ...
KC1143		CMU		21	2005-10-15 ...

Figure 61: Sample distil features

Labelling

Labelling is an operation that is usually performed after clipping and sampling. During labelling, the user assigns ground-truth labels to clips of data.

The user first specifies a subset of the clip columns that should be displayed. The user also specifies the labels that the observer or expert will use to characterize each clip. The expert or observer will have to select between three labels: Good, Not Bad, or Unsure. The circumstances under which an expert or observer labels a clip as "bad" changes depending on the data set, but typically indicate cases that are unfit for the user's purposes. "Unsure" clips can be separated for further analysis by other labellers.

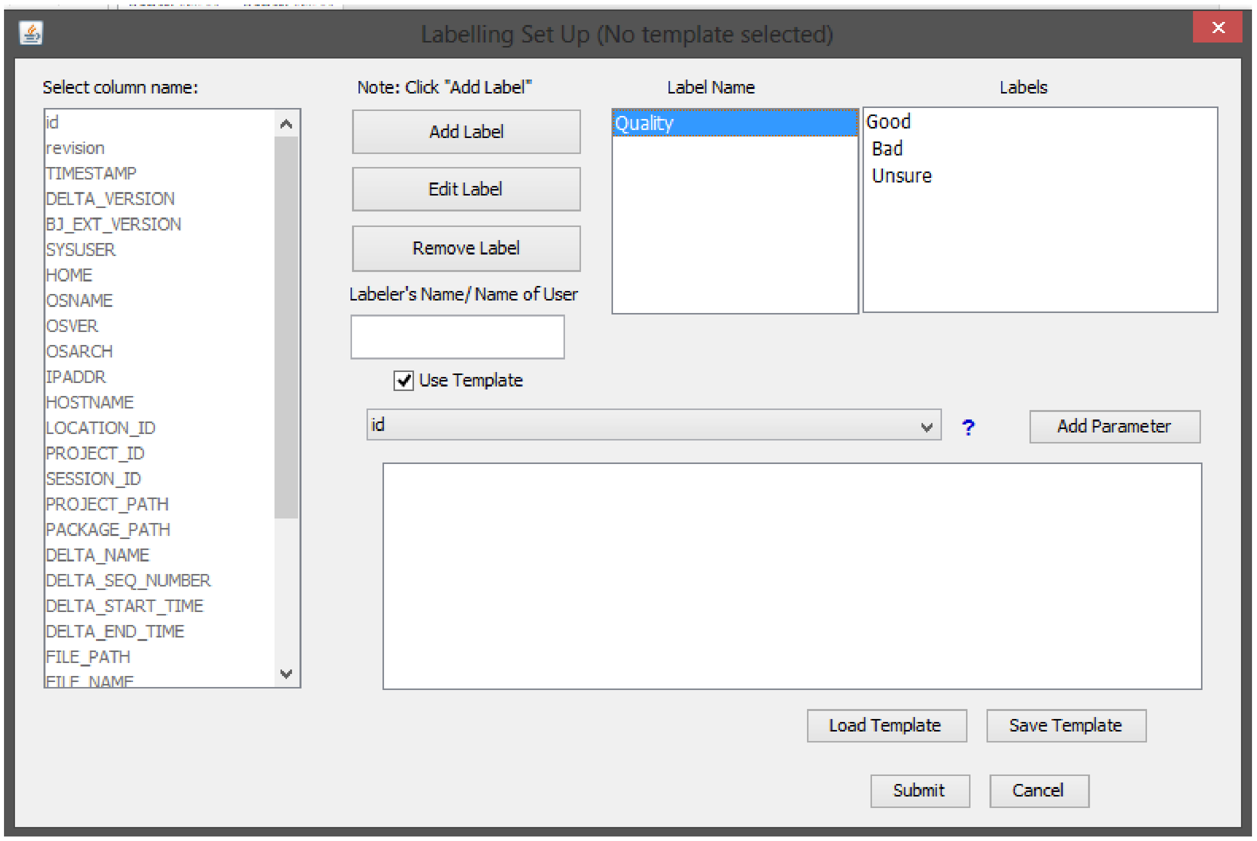
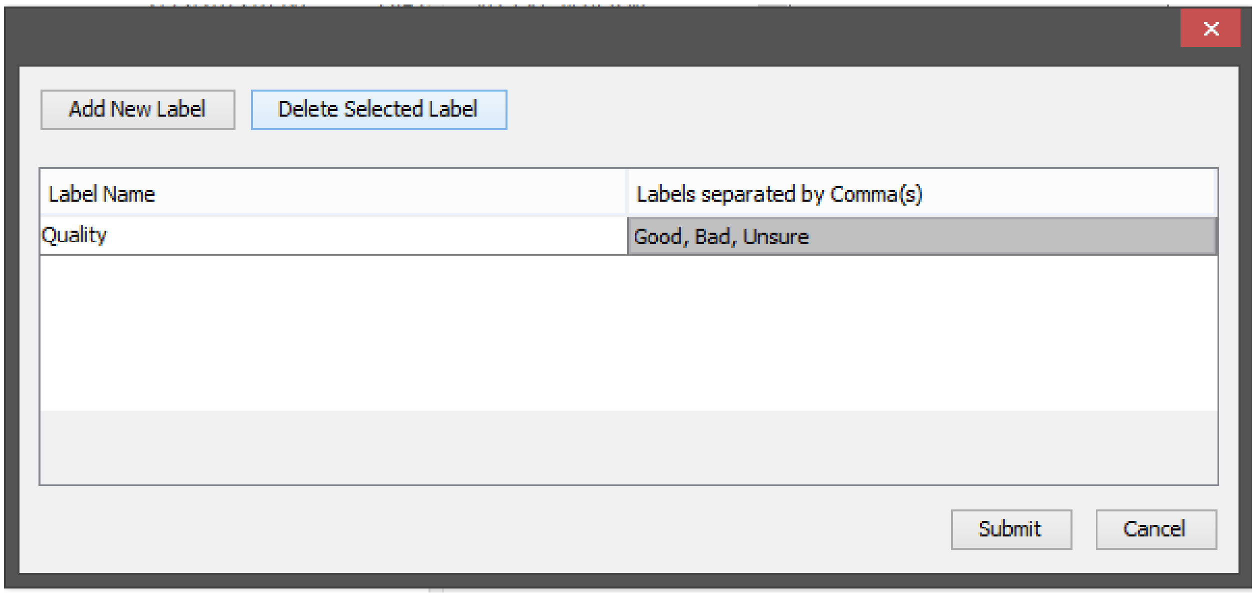


Figure 62: Labelling Window

A. Set-Up Labelling parameters



Label Name	Labels separated by Comma(s)
Quality	Good, Bad, Unsure

Figure 63: A sample Labelling window

1. **Label Name**
   Select Add Label in the Labelling window in order to add user-defined labels. Label name separates a label set from another.

2. **Labels separated by Comma(s)**
   Here, the user will be able to create labels for the data set as separated by commas.

○ Use Template

The template area specifies a "pretty print" of the text replay. The user supplies descriptive text and indicates where the fields should be inserted

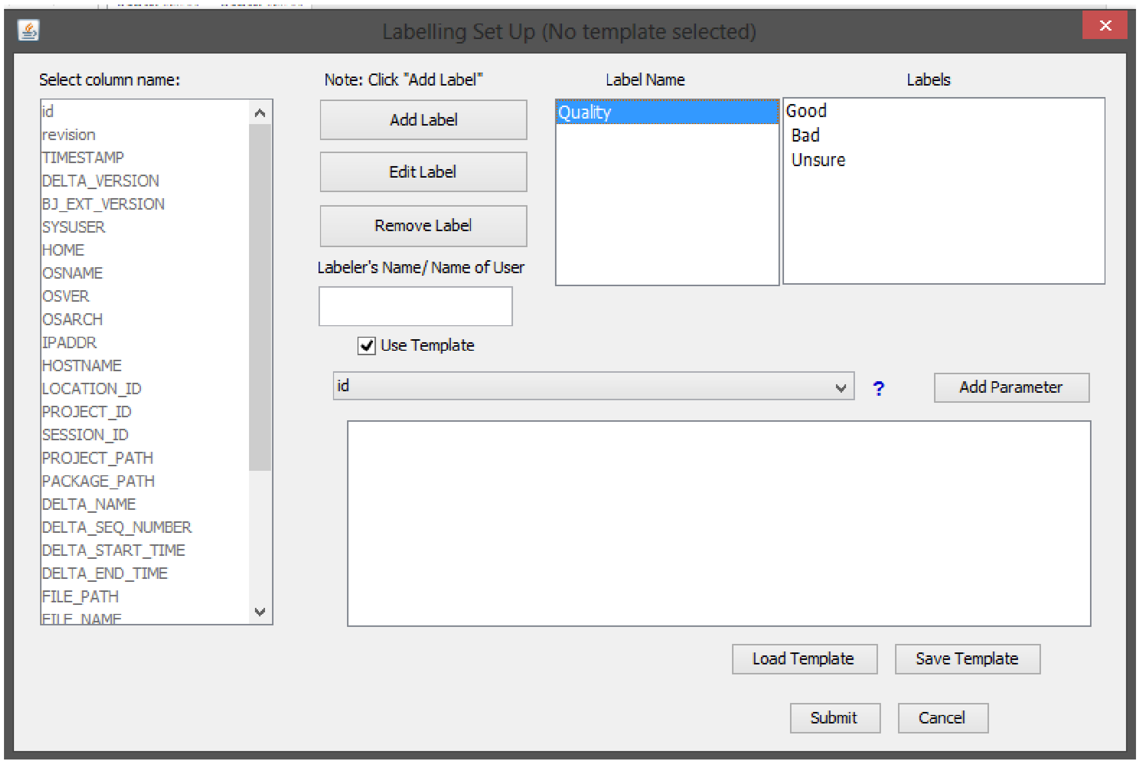


Figure 64: Parameter Addition

Note: The system will automatically select the parameter in the "Select Column Name" list from the textbox.

Multiple Labels

- Users can now (as of version 4) put multiple labels on a data set.

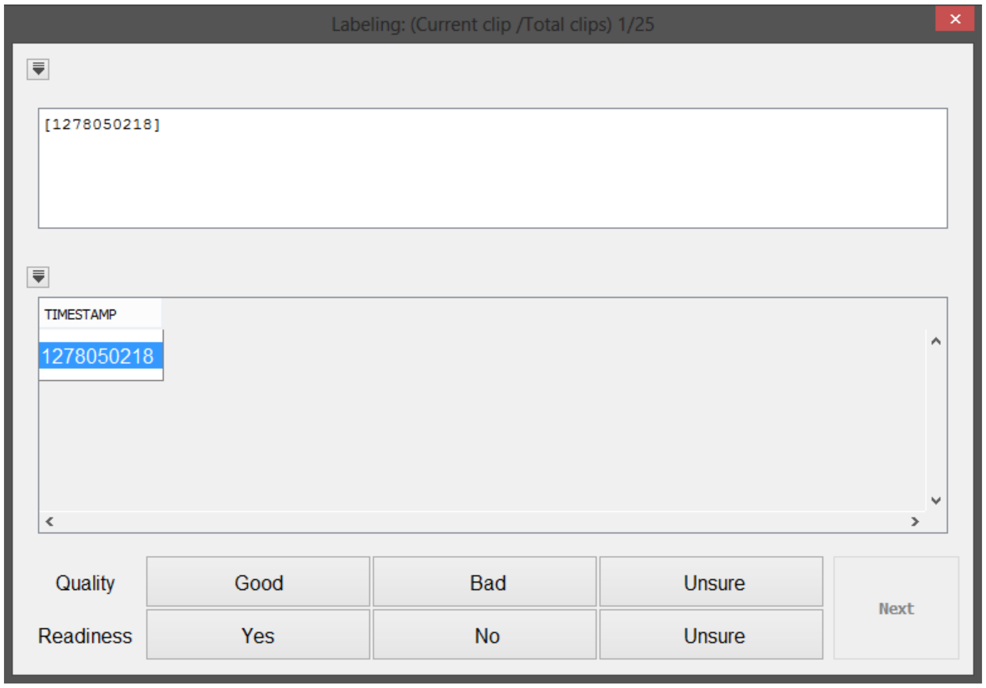


Figure 645: Multiple Labels

Labeller Name

- Users can keep track of labellers by identifying their names via the Labeller Name field. This is useful in keeping quality and standards when it comes to labelling datasets.

- **Labelling Button**
  - **Add Parameter Button**

    In constructing sentences, users can manually input the parameters by enclosing it in a bracket "[]" and with the correct spelling or by selecting a parameter from the dropdown list and then clicking on the Add Parameter button to insert the selected parameter.

  - **Save Template**

    The system allows the user to save the selected Labelling properties. A dialogue will be popped-up and will ask for a template name. The file will be saved as a Labelling.xml file.

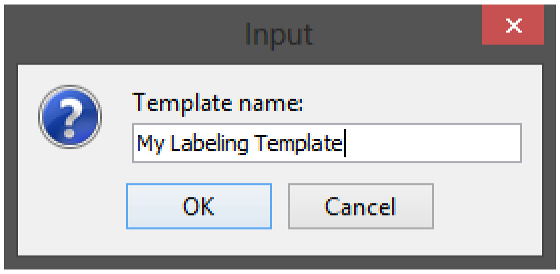


Figure 6: File Name input window

  - **Load Template**

    The user may select a template from the list of labelling templates displayed by the system. The system will then load the properties of the selected template to the labelling form.

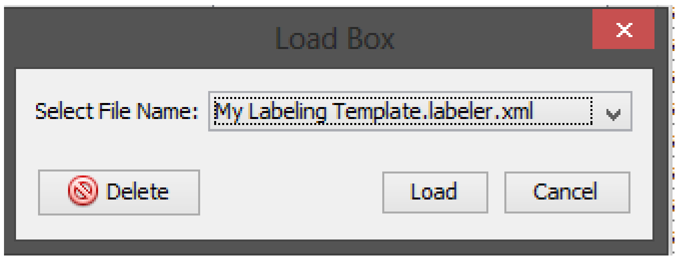
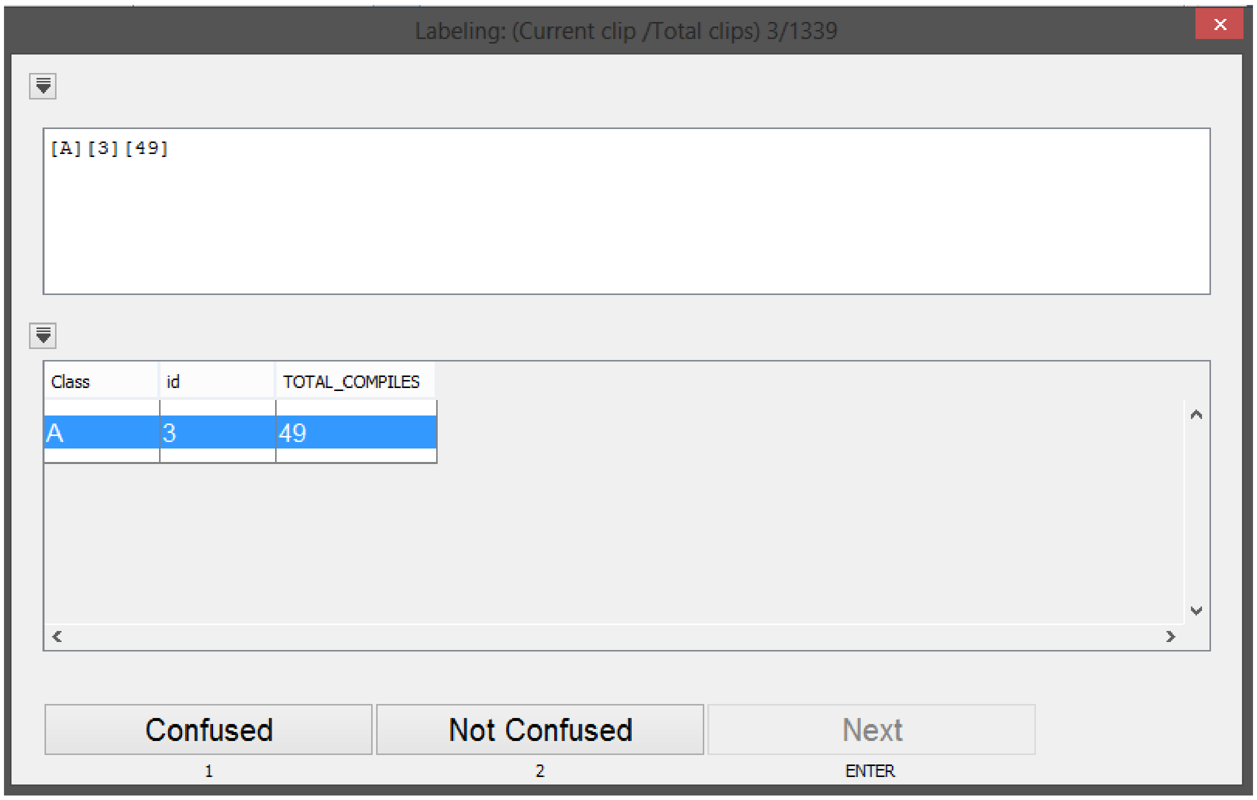


Figure 67: Labelling template loading window

B. Labelling the dataset

The Workbench then displays text replays of the clips together with the labelling options (Figure 3). A coder reads through the text replay and selects the label that best describes the clip. The labels are saved under a new column in the data set.

NOTE: Because a coder may have to label tens of thousands of clips [5], the coder may save his or her work and can continue the labelling process in a later session.



```
[A][3][49]
```

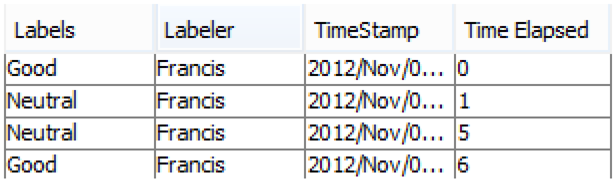
Class	id	TOTAL_COMPILES
A	3	49

Figure 68: Dataset labelling window

Note: In the above example, the user can press the number keys 1 and 2 as shortcut keys for the buttons "Confused and Not Confused" respectively. Press Enter to choose "Next" to go to the next row.

Labelling Time Elapsed

The GUI now displays how much time each labelling action took.

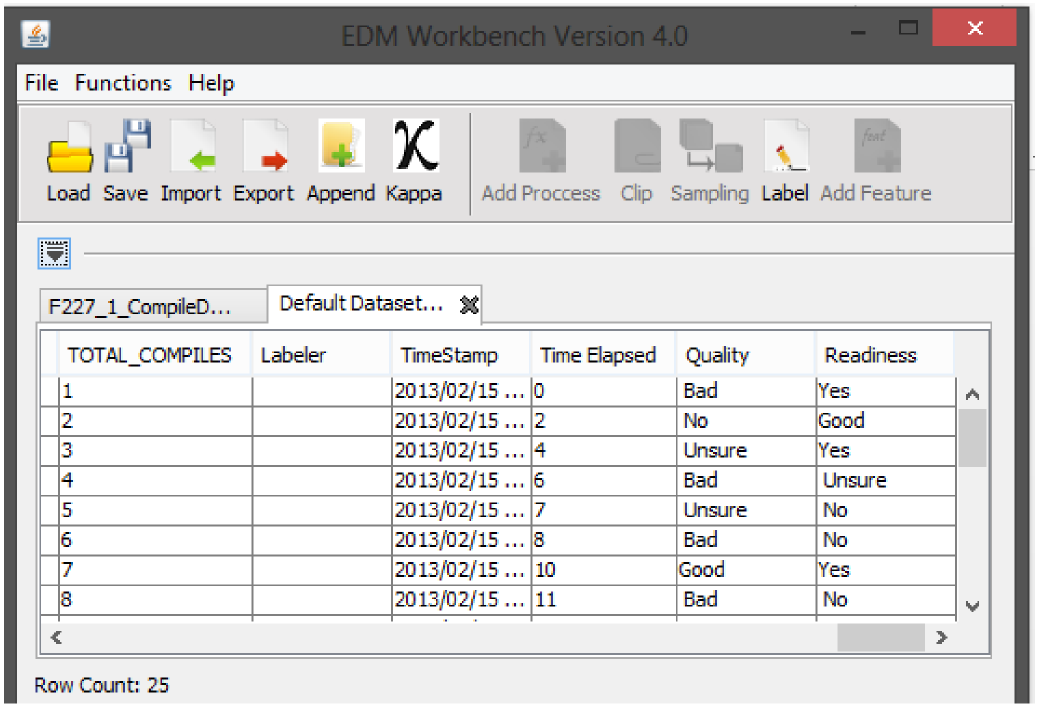


Labels	Labeler	TimeStamp	Time Elapsed
Good	Francis	2012/Nov/0...	0
Neutral	Francis	2012/Nov/0...	1
Neutral	Francis	2012/Nov/0...	5
Good	Francis	2012/Nov/0...	6

Figure 659: Time Elapsed Column for Labels

Labelling Output

As we can see in the figure 70 (below), the labels are shown with their corresponding timestamps and labeller. These column names are present for data organization.


EDM Workbench Version 4.0
File Functions Help
Load Save Import Export Append Kappa Add Process Clip Sampling Label Add Feature
F227_1_CompileD... Default Dataset...

TOTAL_COMPILES	Labeler	TimeStamp	Time Elapsed	Quality	Readiness
1		2013/02/15 ...	0	Bad	Yes
2		2013/02/15 ...	2	No	Good
3		2013/02/15 ...	4	Unsure	Yes
4		2013/02/15 ...	6	Bad	Unsure
5		2013/02/15 ...	7	Unsure	No
6		2013/02/15 ...	8	Bad	No
7		2013/02/15 ...	10	Good	Yes
8		2013/02/15 ...	11	Bad	No

Row Count: 25

Figure 70: Sample labelling output

- **Save**

  Saves the dataset in the current tab by clicking the Save button located either in **File** menu (Figure 6) or **Toolbar** (Figure 9). The system will ask for the directory and then save it in zip format.

  Note: Saving files will take time depending on the size of the dataset and speed of the computer.

- **Load**

  Loads EDM files by clicking the load button located either in the **File** menu (Figure 6) or **Toolbar** (Figure 9). Error dialogues will be displayed if any error is found with the specified directory or file.

  Note: The action button will be enabled depending on the file loaded.

- **Export**

  By clicking the export button located either in the **File** menu (Figure 6) or **Toolbar** (Figure 9), the system will save the current active tab into a CSV file or into another specified format. Users must specify the directory in which the file will be saved.

  Note: Exporting a file will take time depending on the dataset's size.

Note:

In this version, we replaced the term the erroneous "feature" with the more correct "operation". We apologize for the confusion this has caused and are undertaking measures to correct these in the next version.

References

[1] Alcala-Fdez, J., Sanchez, L., Garcia, S., de Jesus, M.J., Ventura, S., Garrell, J. M., Otero, J., Romero, C., Bacardit, J. & Rivas, V.M. (2009). KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing: A Fusion of Foundations, Methodologies and Applications*, *13*(3), 307-318. (1)

[2] Baker, R.S.J.d. (2007). Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. *Proceedings of ACM CHI 2007: Computer-Human Interaction*, 1059-1068. (3)

[3] Baker, R.S.J.d. & de Carvalho (2008). Labeling Student Behavior Faster and More Precisely with Text Replays. *1st International Conference on Educational Data Mining*, 38-47. *(5)*

[4] Corbett, A.T., & Anderson, J.R. (1995). Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction, 4*, 253-278. (7)

[5] de Vicente, A., Pain, H. (2002). Informing the detection of the students' motivational state: an empirical study. *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, 933-943. (8)

[6] McLaren, B.M., Scheuer, O., & Mikšátko, J. (2010). Supporting collaborative learning and e-Discussions using artificial intelligence techniques. *International Journal of Artificial Intelligence in Education (IJAIED) 20*(1), 1-46. (11)

[7] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. & Euler, T. (2006). YALE: Rapid Prototyping for Complex Data Mining Tasks. In *Proc. of the 12th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining (KDD 2006)*, (pp. 935-940), ACM Press. (12)

[8] Walonoski, J. & Heffernan, N.T. (2006). Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems. In Ikeda, Ashley & Chan (Eds.). *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. Springer-Verlag: Berlin. pp. 382-391. (14)

[9] Witten, I. H. & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, Second Edition, Morgan Kaufmann. (15)