MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

DEPARTMENT OF INFORMATION & COMMUNICATION TECHNOLOGY

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL

CERTIFICATE

This is to certify that Ms./Mr. …………………...………………………………………………. Reg.No.…..……………………… Section: …………………Roll No:…………………..has satisfactorily completed the lab exercises prescribed for Data Mining Lab [ICT 3242] of 6th Semester B. Tech. (IT) Degree at MIT, Manipal, in the academic year …………..

Date: ……...................................

Signature of the faculty

CONTENTS

Course Objectives

- To get acquainted with various pre-preprocessing techniques for data mining
- To gain skills for constructing a data warehouse
- To apply data mining and predictive analysis techniques on the pre-processed data
- To provide a data mining solution to a real-world problem

Course Outcomes

At the end of this course, students will be able to

- Identify suitable pre-processing techniques for various datasets
- Demonstrate the construction of data warehouse
- Apply suitable data mining technique on pre-processed data
- Develop data mining applications for a large data set

INSTRUCTIONS TO THE STUDENTS

Pre- Lab Session Instructions

1. Students should carry the Lab Manual Book and the required stationery to every lab session
2. Be on time and follow the institution dress code
3. Must sign in the log register provided
4. Make sure to occupy the allotted seat and answer the attendance
5. Adhere to the rules and maintain the decorum

In- Lab Session Instructions

- Follow the instructions on the allotted exercises
- Show the program and results to the instructors on completion of experiments
- Prescribed textbooks and class notes can be kept ready for reference if required

General Instructions for the exercises in Lab

- Implement the given exercise individually and not in a group.
- The programs should meet the following criteria:
  - Programs should be interactive with appropriate prompt messages, error messages if any, and descriptive messages for outputs.
  - Comments should be used to give the statement of the problem.
  - Statements within the program should be properly indented.
- Plagiarism (copying from others) is strictly prohibited and would invite severe penalty in evaluation.
- In case a student misses a lab, he/ she must ensure that the experiment is completed before the next evaluation with the permission of the faculty concerned.
- Students missing out lab on genuine reasons like conferences, sports or activities assigned by the Department or Institute will have to take **prior permission** from the HOD to attend **additional lab** (with other batch) and complete it **before** the student goes on leave. The student could be awarded marks for the write-up for that day provided he submits it during the **next immediate** lab.
- Students who fall sick should get permission from the HOD for evaluating the lab records. However, attendance will not be given for that lab.
- Students will be evaluated only by the faculty with whom they are registered even though they carry out additional experiments in another batch.
- Presence of the student during the lab end semester exams is mandatory even if the student assumes he has scored enough to pass the examination.
- Minimum attendance of 75% is mandatory to write the final exam.
- If the student loses his book, he/she will have to rewrite all the lab details in the lab record.
- Questions for lab tests and examinations are not necessarily limited to the questions in the manual but may involve some variations and / or combinations of the questions.

THE STUDENTS SHOULD NOT

- Bring mobile phones or any other electronic gadgets to the lab.
- Go out of the lab without permission.

LAB NO. 1 **Date**

Talend Open Studio for Data Integration

Data Integration

Data integration is the process of combining data from several different sources in a unified view, making it more actionable and valuable to those accessing it. Successful data integration combines speed and integrity at scale — so that organizations can meet the demands of their business today, with data they can trust, and keep up with new innovations and the exponential growth in data. There is no universal approach to data integration. However, integration solutions generally involve a few common elements, including a network of data sources, a master server, and clients accessing data from the master server. In a typical data integration process, the client sends a request to the master server for data. The master server then intakes the needed data from internal and external sources. The data is extracted from the sources, then combined in a cohesive, unified form. This is served back to the client

Talend

Talend is an ETL tool for Data Integration. It provides software solutions for data preparation, data quality, data integration, application integration, data management and big data. Talend has a separate product for all these solutions. Data integration and big data products are widely used. This tutorial helps you to learn all the fundamentals of Talend tool for data integration and big data with examples.

ETL

ETL (or Extract, Transform, Load) is a process of data integration that encompasses three steps — extraction, transformation, and loading. In a nutshell, ETL systems take large volumes of raw data from multiple sources, converts it for analysis, and loads that data into your warehouse. Let's cover the three primary ETL steps.

Extraction

In the first step, extracted data sets come from a source (e.g., Salesforce, Google AdWords, etc.) into a staging area. The staging area acts as a buffer between the data warehouse and the source data. Since data may be coming from multiple different sources, it's likely in various formats, and directly transferring the data to the warehouse may result in corrupted data. The staging area is used for data cleansing and organization.
A big challenge during the data extraction process is how your ETL tool handles structured and unstructured data. All of those unstructured items (e.g., emails, web pages, etc.) can be difficult to extract without the right tool, and you may have to create a custom solution to assist you in transferring unstructured data if you chose a tool with poor unstructured data capabilities.

Transformation

The data cleaning and organization stage is the transformation stage. All of that data from multiple source systems will be normalized and converted to a single system format — improving data quality and compliance. ETL yields transformed data through these methods:

- Cleaning
- Filtering
- Joining
- Sorting
- Splitting
- Deduplication
- Summarization

Loading

Finally, data that has been extracted to a staging area and transformed is loaded into your data warehouse. Depending upon your business needs, data can be loaded in batches or all at once. The exact nature of the loading will depend upon the data source, ETL tools, and various other factors.

Working with projects

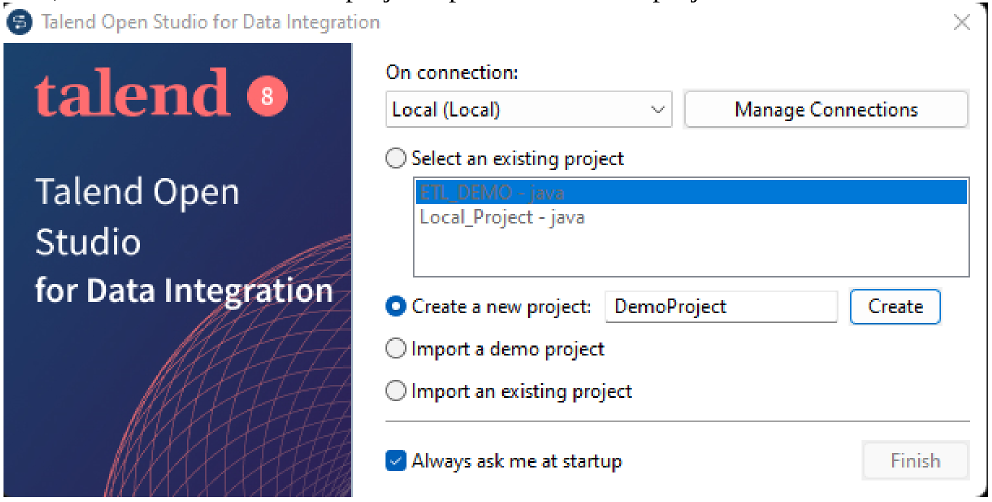
Creating a project

A project is the highest physical structure for storing all different types of items. Once you launch your Talend Studio and before you start a Business Model, a data integration Job, a Route, or any other tasks, you need first create or import a project.

Creating a project at initial Studio launch

Procedure

1. Launch Talend Studio and connect to a local repository.
2. On the login window, select the Create a new project option and enter a project name in the field.



Talend Open Studio for Data Integration

On connection: Local (Local)

( ) Select an existing project
- ETL_DEMO - java
- Local_Project - java

(●) Create a new project: DemoProject

( ) Import a demo project

( ) Import an existing project

☑ Always ask me at startup

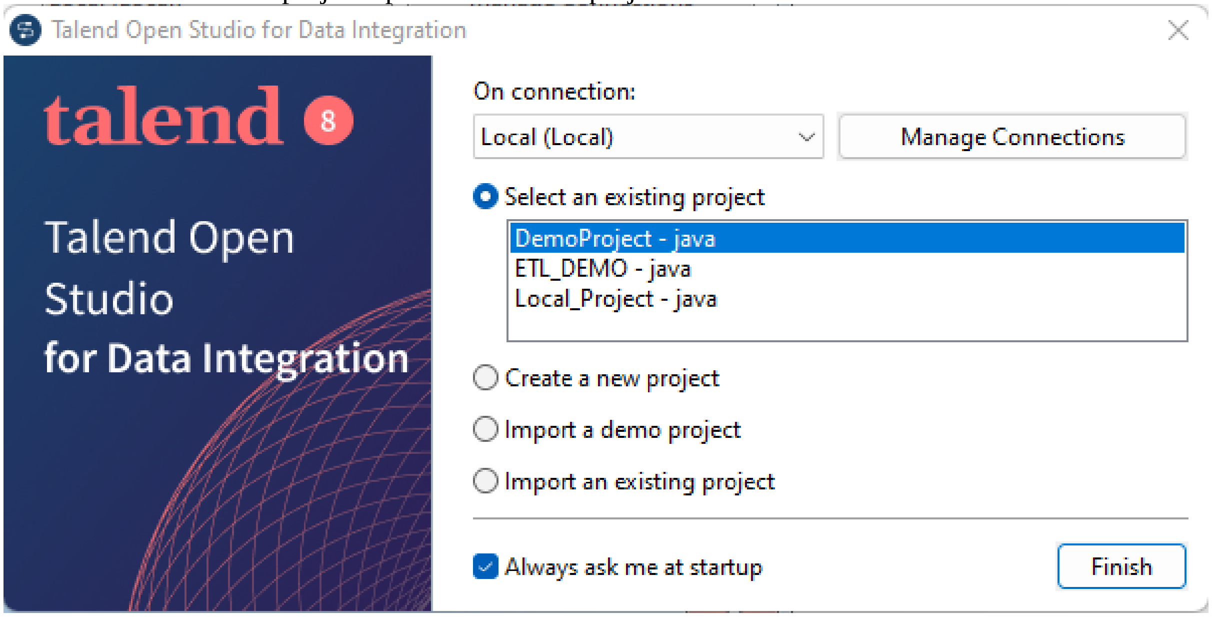
Note:
- A project name is case insensitive
- A project name must start with an English letter and can contain only letters, numbers, the hyphen (-), and the underscore (_)
- The hyphen (-) character is deemed as the underscore (_)

3. Click Finish to create the project and open it in the Studio.

Creating a new project after initial Studio launch

About this task

To create a new local project after the initial startup of the Studio, do the following: Procedure 1. On the login window, select the Create a new project option and enter a project name in the field.



Talend Open Studio for Data Integration

On connection: Local (Local)

(●) Select an existing project
- DemoProject - java
- ETL_DEMO - java
- Local_Project - java

( ) Create a new project

( ) Import a demo project

( ) Import an existing project

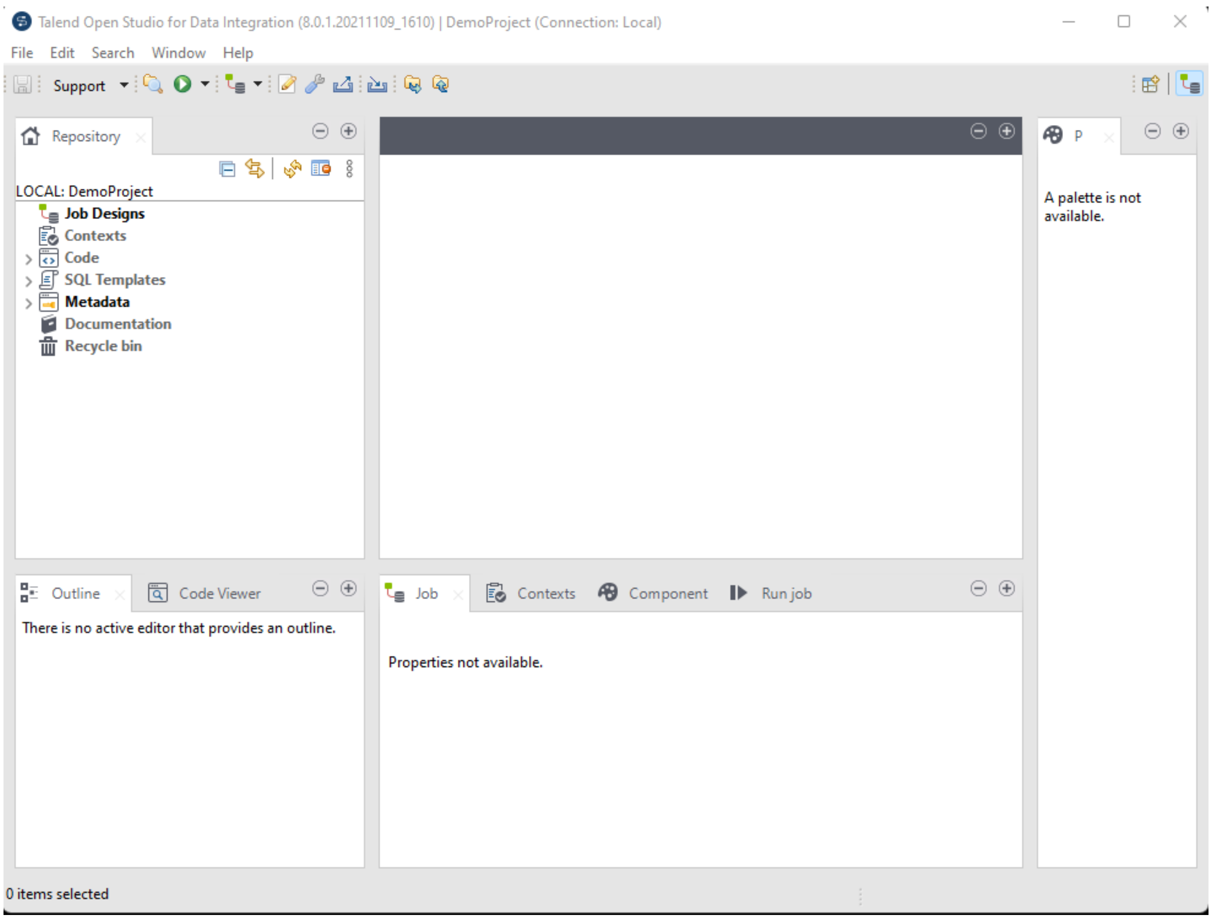
☑ Always ask me at startup

2. Click Create to create the project. The newly created project is displayed on the list of existing projects.
3. Select the project on the list and click Finish to open the project in the Studio. Later, if you want to switch between projects, on the Studio menu bar, use the combination File > Switch Project or Workspace

Designing Jobs

What is a Job design?

A Job Design is the runnable layer of a business model. It is a graphical design, of one or more components connected together, that allows you to set up and run dataflow management processes. A Job Design translates business needs into code, routines and programs, in other words it technically implements your data flow. The Jobs you design can address all of the different sources and targets that you need for data integration processes and any other related process.
When you design a Job in Talend Studio, you can:
• put in place data integration actions using a library of technical components.
• change the default setting of components or create new components or family of components to match your exact needs.
• set connections and relationships between components in order to define the sequence and the
nature of actions.
• access code at any time to edit or document the components in the designed Job.
• create and add items to the repository for reuse and sharing purposes (in other projects or Jobs or
with other users).


Talend Open Studio for Data Integration (8.0.1.20211109_1610) | DemoProject (Connection: Local)

A palette is not available.

There is no active editor that provides an outline.

Properties not available.

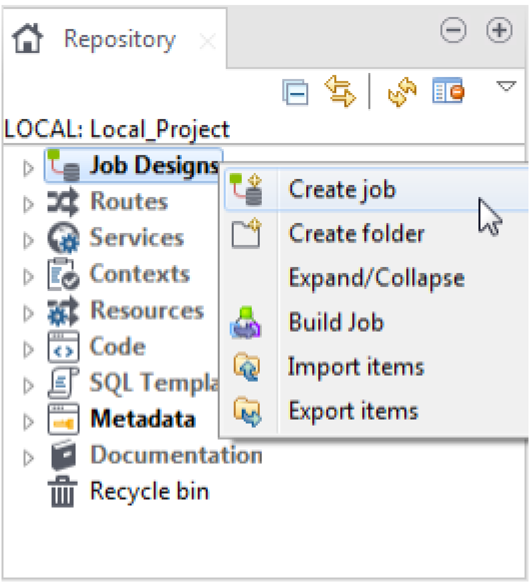
0 items selected

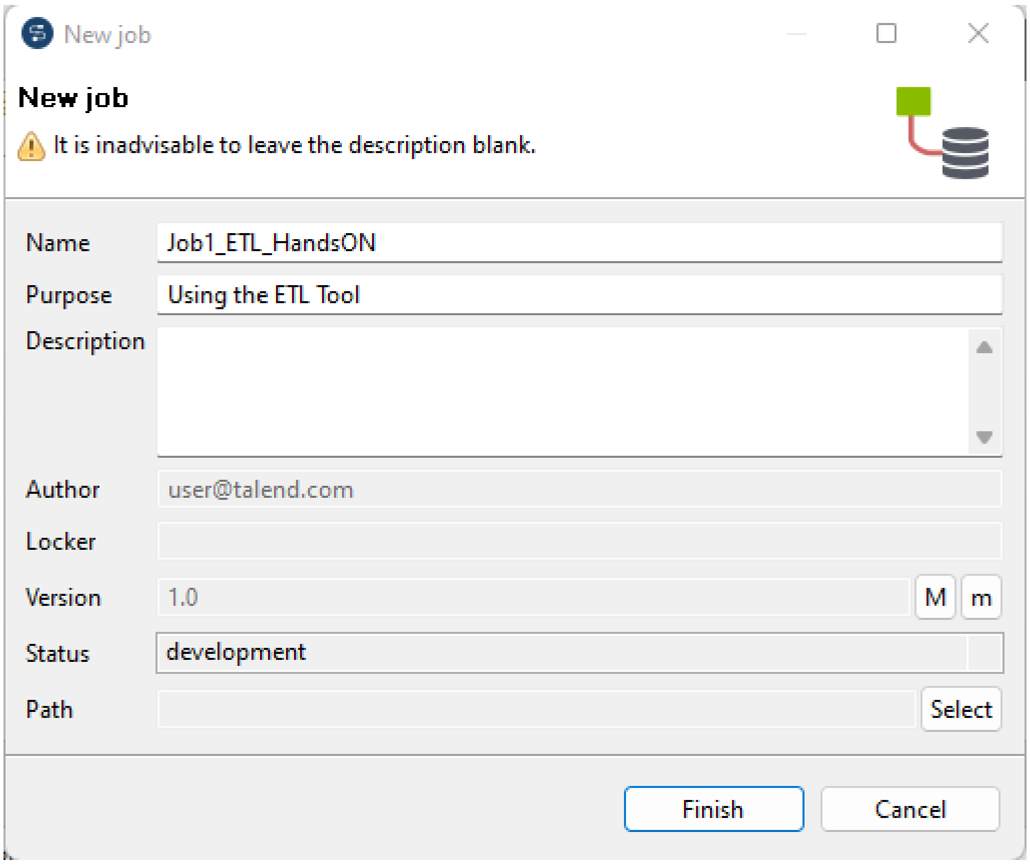
Creating a Job

Talend Studio enables you to create a Job by dropping different technical components from the Palette onto the design workspace and then connecting these components together.

About this task

To create the example Job described in this section, proceed as follows:
Procedure
In the Repository tree view of the Integration perspective, right-click the Job Designs node and select Create job from the contextual menu.




New job

New job

It is inadvisable to leave the description blank.

Name: Job1_ETL_HandsON

Purpose: Using the ETL Tool

Description:

Author: user@talend.com

Locker:

Version: 1.0

Status: development

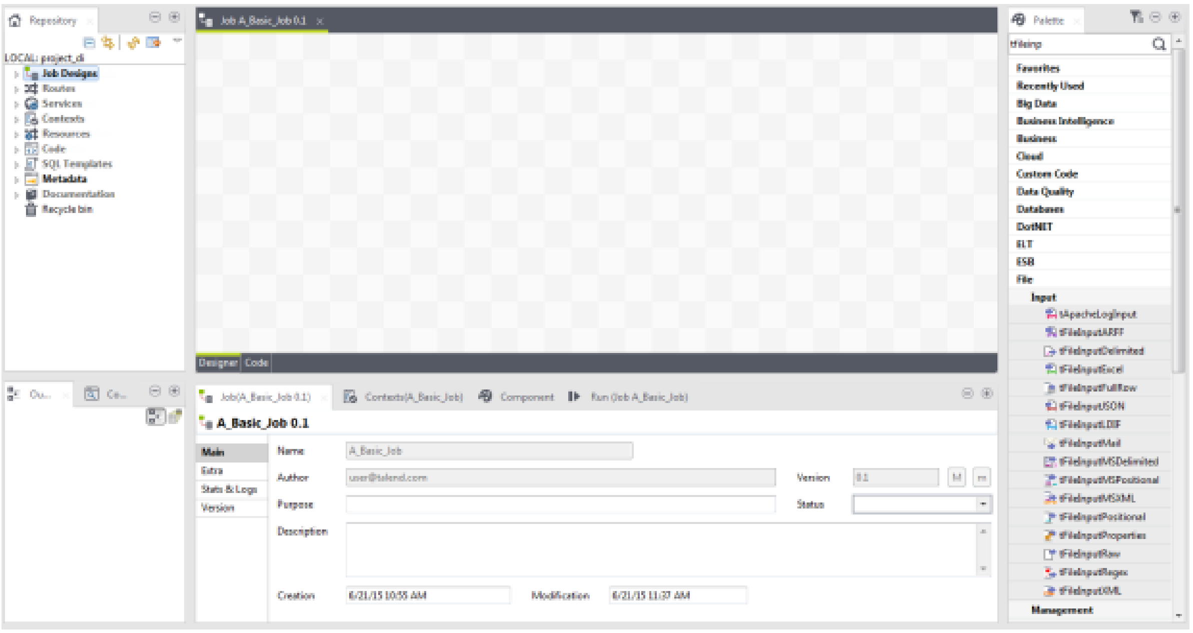
Path:

Fill the Job properties as shown in the previous screenshot.
The fields correspond to the following properties:

Field	Description
Name	the name of the new Job. Note that a message comes up if you enter prohibited characters.
Purpose	Job purpose or any useful information regarding the Job use.

Field	Description
Description	Job description containing any information that helps you describe what the Job does and how it does it.
Author	a read-only field that shows by default the current user login.
Locker	a read-only field that shows by default the login of the user who owns the lock on the current Job. This field is empty when you are creating a Job and has data only when you are editing the properties of an existing Job.
Version	a read-only field. You can manually increment the version using the **M** and **m** buttons. For more information, see Managing Job versions on page 112.
Status	a list to select from the status of the Job you are creating.
Path	a list to select from the folder in which the Job will be created.

3. An empty design workspace opens up showing the name of the Job as a tab label.



A_Basic_Job 0.1

Name: A_Basic_Job
Author: user@talend.com
Version: 0.1
Purpose:
Status:
Description:
Creation: 6/21/15 10:55 AM
Modification: 6/21/15 11:37 AM

Results

The Job you created is now listed under the Job Designs node in the Repository tree view.
You can open one or more of the created Jobs by simply double-clicking the Job label in the Repository tree view.

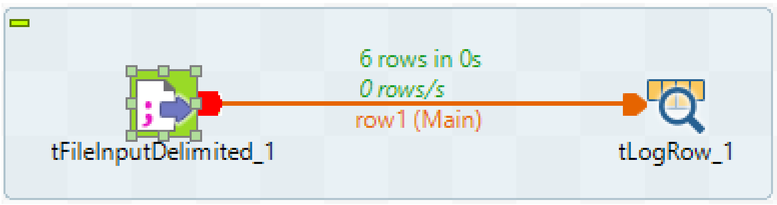
Introduction to Components:

1. tFileInputDelimited

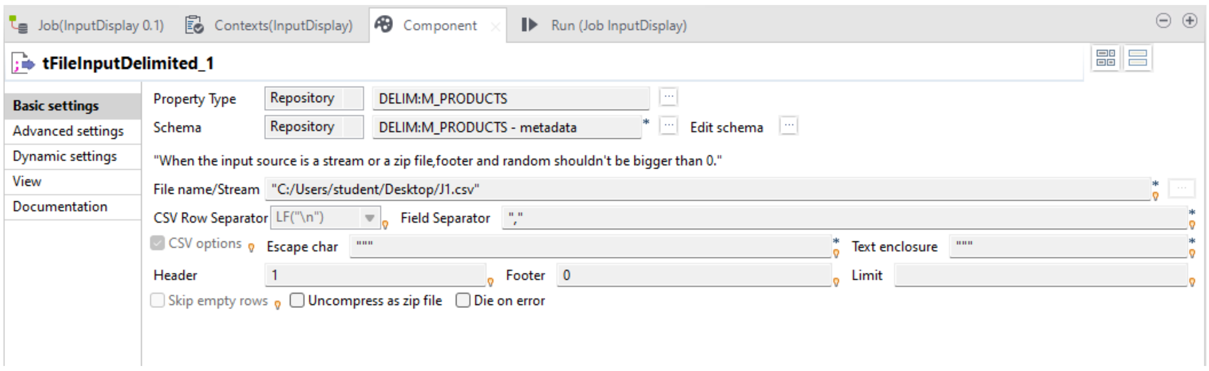
   Reads a delimited file row by row to split them up into fields and then sends the fields as defined in the schema to the next component.

   Configuring the components

   Procedure



1. Select the tFileInputDelimited component again, and define its Basic settings:


Job(InputDisplay 0.1) | Contexts(InputDisplay) | Component | Run (Job InputDisplay)

tFileInputDelimited_1

Basic settings
Advanced settings
Dynamic settings
View
Documentation

Property Type: Repository | DELIM:M_PRODUCTS
Schema: Repository | DELIM:M_PRODUCTS - metadata | Edit schema
"When the input source is a stream or a zip file,footer and random shouldn't be bigger than 0."
File name/Stream: "C:/Users/student/Desktop/J1.csv"
CSV Row Separator: LF("\n") | Field Separator: ","
CSV options | Escape char: """ | Text enclosure: """
Header: 1 | Footer: 0 | Limit:
Skip empty rows | Uncompress as zip file | Die on error

Fill in a path to the file in the **File Name** field. This field is mandatory.

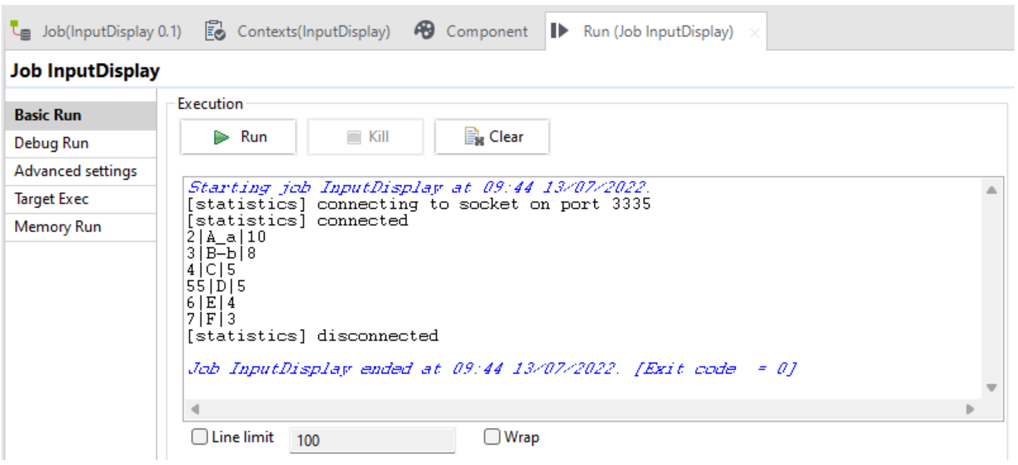
1. Define the **Row separator** allowing to identify the end of a row. Then define the **Field separator** used to delimit fields in a row.
2. In this scenario, the header and footer limits are not set. And the **Limit** number of processed rows is set on 50.
3. Set the **Schema** as remotely managed (**Repository**) to define the data to pass on to the **tLogRow** component.
4. You can load and/or edit the schema via the **Edit Schema** function.
5. Enter the encoding standard the input file is encoded in. This setting is meant to ensure encoding consistency throughout all input and output files.
6. Select the **tLogRow** and define the **Field separator** to use for the output display.
7. Select the **Print schema column name in front of each value** check box to retrieve the column labels in the output displayed.

Saving and executing the Job

Procedure

1. Press **Ctrl+S** to save your Job.
2. Go to **Run** tab, and click on **Run** to execute the Job.

   The file is read row by row and the extracted fields are displayed on the **Run** log as defined in both components **Basic settings**.



```
Starting job InputDisplay at 09:44 13/07/2022.
[statistics] connecting to socket on port 3335
[statistics] connected
2|A_a|10
3|B-b|8
4|C|5
55|D|5
6|E|4
7|F|3
[statistics] disconnected

Job InputDisplay ended at 09:44 13/07/2022. [Exit code = 0]
```

2. tLogRow:

Displays data or results in the Run console to monitor data processed.

3. tFileOutputDelimited, tMap:

Outputs the input data to a delimited file according to the defined schema.

Writing data in a delimited file

This scenario describes a three-component Job that extracts certain data from a file holding information about clients, *customers*, and then writes the extracted data in a delimited file.

In the following example, we have already stored the input schema under the Metadata node in the Repository tree view.

Procedure

1. In the Repository tree view, expand Metadata and File delimited in succession and then browse to your input schema, customers, and drop it on the design workspace. A dialog box displays where you can select the component type you want to use.

2. Click tFileInputDelimited and then OK to close the dialog box. A tFileInputDelimited component holding the name of your input schema appears on the design workspace.

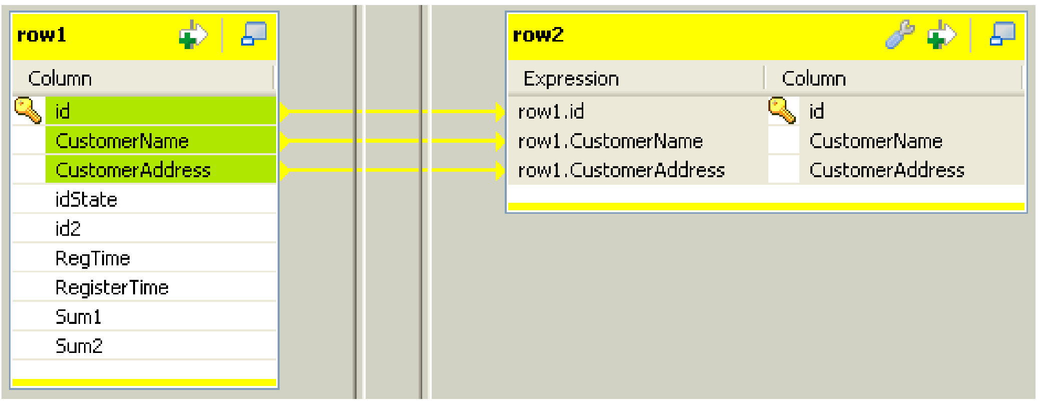
3. Drop a tMap component and a tFileOutputDelimited component from the Palette to the design workspace.

4. Link the components together using Row > Main connections.

5. Configure the input component. Refer tInputDelimited.

Configuring the mapping component :

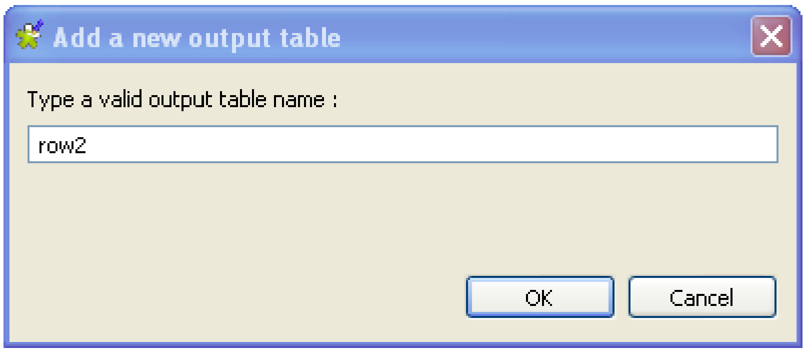
1. In the design workspace, double-click **tMap** to open its editor.



Column
id
CustomerName
CustomerAddress
idState
id2
RegTime
RegisterTime
Sum1
Sum2

Expression	Column
row1.id	id
row1.CustomerName	CustomerName
row1.CustomerAddress	CustomerAddress

2. In the **tMap** editor, click ✚ on top of the panel to the right to open the **Add a new output table** dialog box.



Add a new output table

Type a valid output table name :

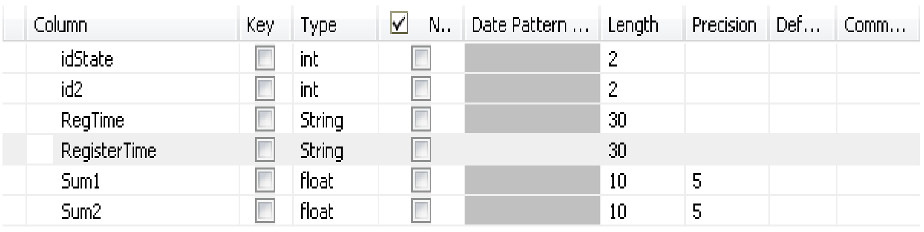
row2

3. Enter a name for the table you want to create, *row2* in this example.

4. Click **OK** to validate your changes and close the dialog box.

5. In the table to the left, *row1*, select the first three lines (*Id*, *CustomerName* and *CustomerAddress*) and drop them to the table to the right

6. In the **Schema editor** view situated in the lower left corner of the **tMap** editor, change the type of *RegisterTime* to **String** in the table to the right.



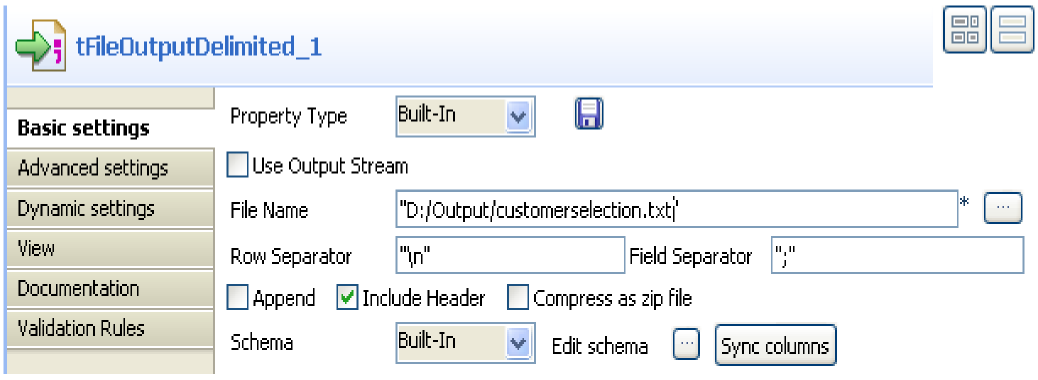
Column	Key	Type	✓ N..	Date Pattern ...	Length	Precision	Def...	Comm...
idState	☐	int	☐		2			
id2	☐	int	☐		2			
RegTime	☐	String	☐		30			
RegisterTime	☐	String	☐		30			
Sum1	☐	float	☐		10	5		
Sum2	☐	float	☐		10	5		

7. Click OK to save your changes and close the editor.

Configuring the output component

Procedure

1. In the design workspace, double-click **tFileOutputDelimited** to open its **Basic settings** view and define the component properties.



tFileOutputDelimited_1

Property Type: Built-In

☐ Use Output Stream

File Name: "D:/Output/customerselection.txt"

Row Separator: "\n" Field Separator: ";"

☐ Append ✓ Include Header ☐ Compress as zip file

Schema: Built-In Edit schema Sync columns

2. In the **Property Type** field, set the type to **Built-in** and fill in the fields that follow manually.

3. Click the **[...]** button next to the **File Name** field and browse to the output file you want to write data in, *customerselection.txt* in this example.

4. In the **Row Separator** and **Field Separator** fields, set "\n" and ";" respectively as row and field separators.

5. Select the **Include Header** check box if you want to output columns headers as well.

6. Click **Edit schema** to open the schema dialog box and verify if the recuperated schema corresponds to the input schema. If not, click **Sync Columns** to recuperate the schema from the preceding component.

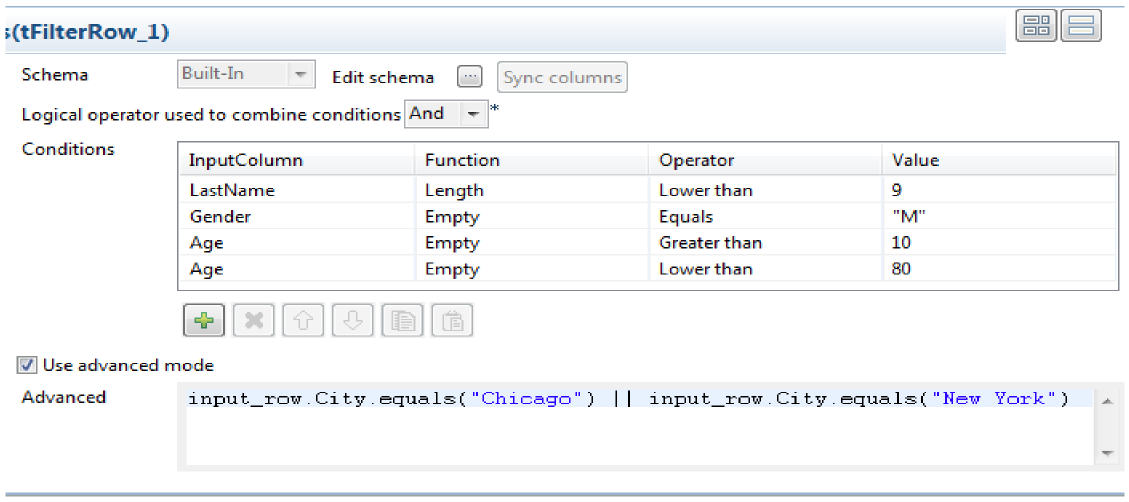
7. Press Ctrl+S **to save your Job, Press** F6 **or click** Run **on the** Run **tab to execute.**

4. tFilterRow:
Filtering a list of names through different logical operations

Based on the previous scenario, this scenario further filters the input data so that only those records of people from New York and Chicago are accepted. Without changing the filter settings defined in the previous scenario, advanced conditions are added in this scenario to enable both logical AND and logical OR operations in the same **tFilterRow** component.

Procedure

1. **Double-click the tFilterRow component to show its Basic settings view**.



(tFilterRow_1)

Schema: Built-In | Edit schema | Sync columns

Logical operator used to combine conditions: And

Conditions

InputColumn	Function	Operator	Value
LastName	Length	Lower than	9
Gender	Empty	Equals	"M"
Age	Empty	Greater than	10
Age	Empty	Lower than	80

☑ Use advanced mode

Advanced

```
input_row.City.equals("Chicago") || input_row.City.equals("New York")
```

2. Select the Use advanced mode **check box, and type in the following expression in the text field:**

```
input_row.City.equals("Chicago") || input_row.City.equals("New York")
```

This defines two conditions on the *City* column of the input data to filter records that contain the cities of Chicago and New York, and uses a logical OR to combine the two conditions so that records satisfying either condition will be accepted.

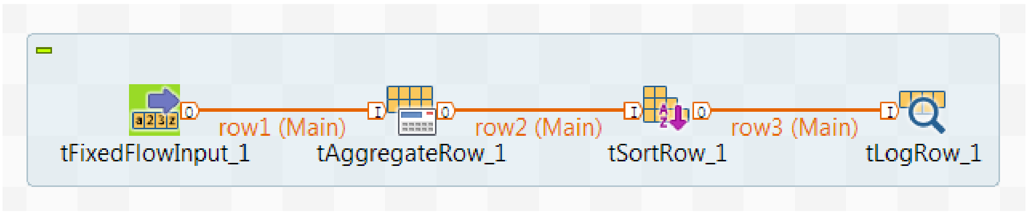
3. Press Ctrl+S **to save the Job and press** F6 **to execute it**

5. tAggregateRow, tSortRow: Aggregating values and sorting data

This example shows you how to use Talend components to aggregate the students'

Creating a Job for aggregating and sorting data

Create a Job to aggregate the students' comprehensive scores using the tAggregateRow **component, then sort the aggregated data using the** tSortRow **component, finally display the aggregated and sorted data on the console.**



Procedure

1. Create a new Job and add a **tFixedFlowInput** component, **tAggregateRow** component, a **tSortRow** component, and a **tLogRow** component by typing their names in the design workspace or dropping them from the **Palette**.

2. Link the **tFixedFlowInput** component to the **tAggregateRow** component using a **Row** > **Main** connection.

3. Do the same to link the **tAggregateRow** component to the **tSortRow** component, and the **tSortRow** component to the **tLogRow** component.

Configuring the Job for aggregating and sorting data

Configure the Job to aggregate the students' comprehensive scores using the **tAggregateRow** component and then sort the aggregated data using the **tSortRow** component.

Procedure

1. Double-click the **tFixedFlowInput** component to open its **Basic settings** view.

2. Click the [...] button next to **Edit schema** to open the schema dialog box and define the schema by adding two columns, *name* of String type and *score* of Double type. When done, click **OK** to save the changes and close the schema dialog box.

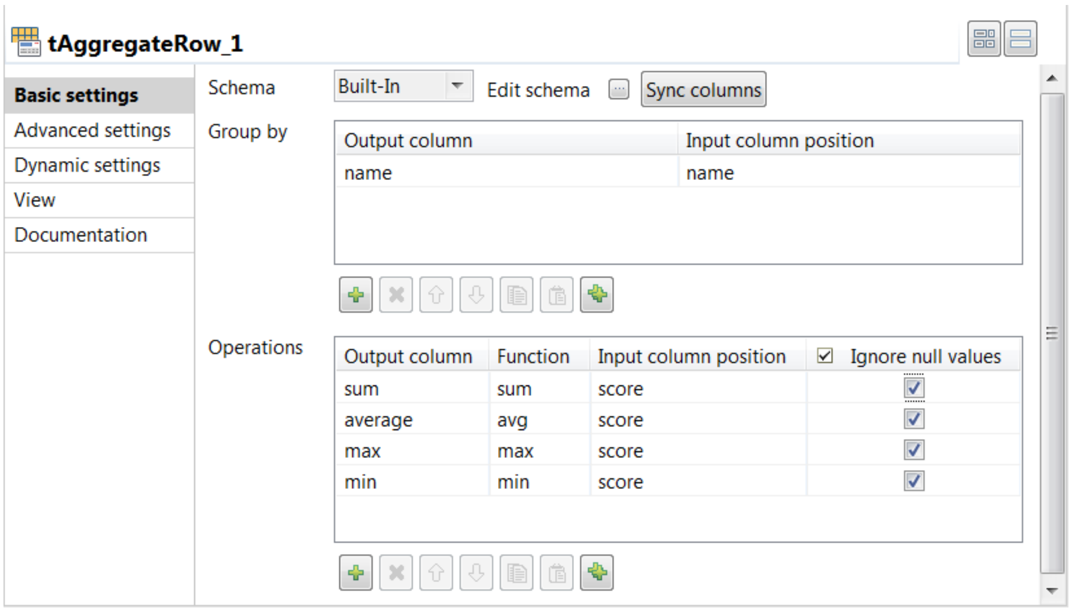
3. In the **Mode** area, select **Use Inline Content (delimited file)** and in the **Content** field displayed, enter the following input data:

```
4. Peter;92
```

```
5.  James;93
6.  Thomas;91
7.  Peter;94
8.  James;96
9.  Thomas;95
10. Peter;96
11. James;92
12. Thomas;98
13. Peter;95
14. James;96
15. Thomas;93
16. Peter;98
17. James;97
```

```
Thomas;95
```

18. Double-click the **tAggregateRow** component to open its **Basic settings** view.



tAggregateRow_1

Schema: Built-In   Edit schema   Sync columns

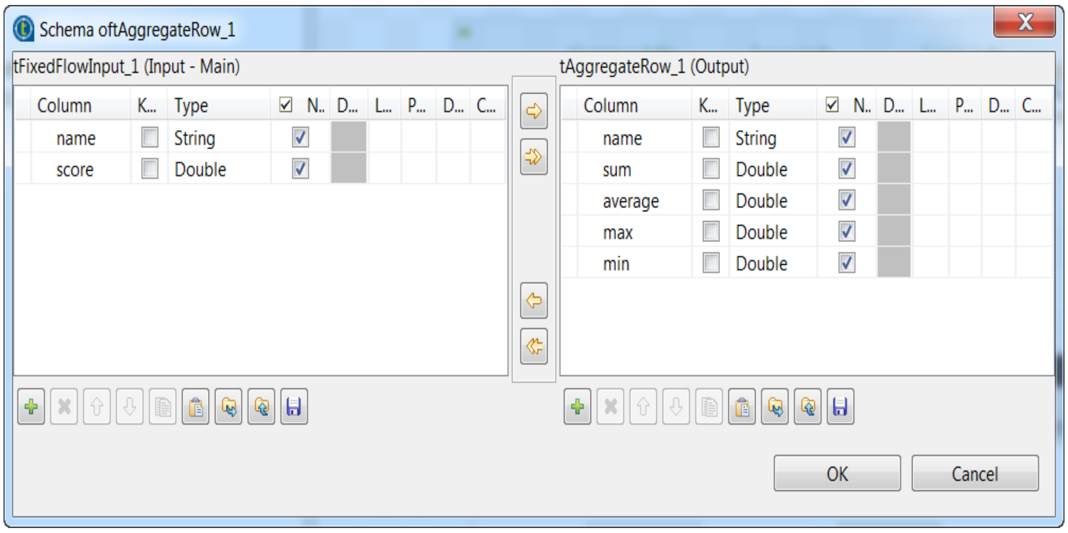
Group by

Output column	Input column position
name	name

Operations

Output column	Function	Input column position	☑ Ignore null values
sum	sum	score	☑
average	avg	score	☑
max	max	score	☑
min	min	score	☑

19. Click the ☐ button next to **Edit schema** to open the schema dialog box and define the schema by adding five columns, *name* of String type, and *sum*, *average*, *max*, and *min* of Double type.



Schema oftAggregateRow_1

tFixedFlowInput_1 (Input - Main)

Column	K...	Type	N..	D...	L...	P...	D...	C...
name	☐	String	✓					
score	☐	Double	✓					

tAggregateRow_1 (Output)

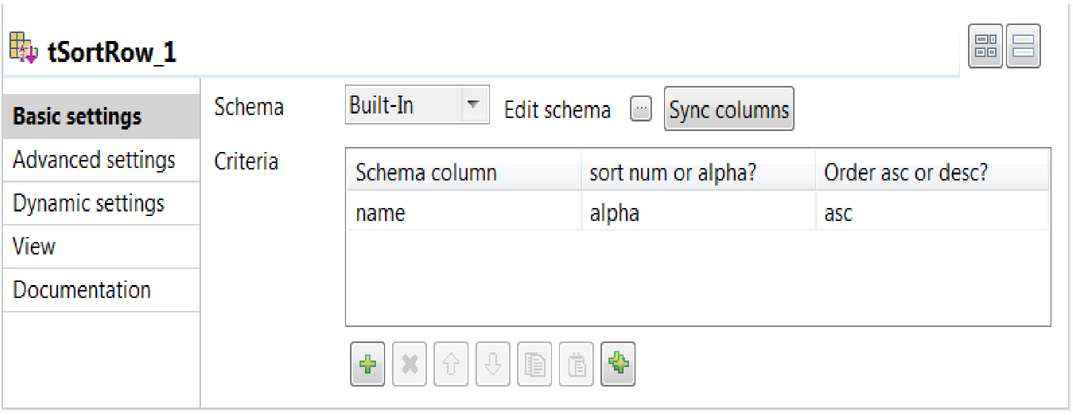
Column	K...	Type	N..	D...	L...	P...	D...	C...
name	☐	String	✓					
sum	☐	Double	✓					
average	☐	Double	✓					
max	☐	Double	✓					
min	☐	Double	✓					

When done, click **OK** to save the changes and close the schema dialog box.

20. Add one row in the **Group by** table by clicking the ➕ button below it, and select *name* from both the **Output column** and **Input column position** column fields to group the input data by the *name* column.

21. Add four rows in the **Operations** table and define the operations to be carried out. In this example, the operations are *sum*, *average*, *max*, and *min*. Then select *score* from all four **Input column position** column fields to aggregate the input data based on it.

22. Double-click the **tSortRow** component to open its **Basic settings** view.



tSortRow_1

Schema: Built-In | Edit schema | Sync columns

Criteria

Schema column	sort num or alpha?	Order asc or desc?
name	alpha	asc

23. Add one row in the **Criteria** table and specify the column based on which the sort operation is performed. In this example, it is the *name* column. Then select *alpha* from the **sort num or alpha?** column field and *asc* from the **Order asc or desc?** column field to sort the aggregated data in ascending alphabetical order.

24. Double-click the **tLogRow** component to open its **Basic settings** view, and then select **Table (print values in cells of a table)** in the **Mode** area for better readability of the result.

Executing the Job to aggregate and sort data

After setting up the Job and configuring the components used in the Job for aggregating and sorting data, you can then execute the Job and verify the Job execution result.

Procedure

1. Press **Ctrl + S** to save the Job.

2. Press **F6** to execute the Job.

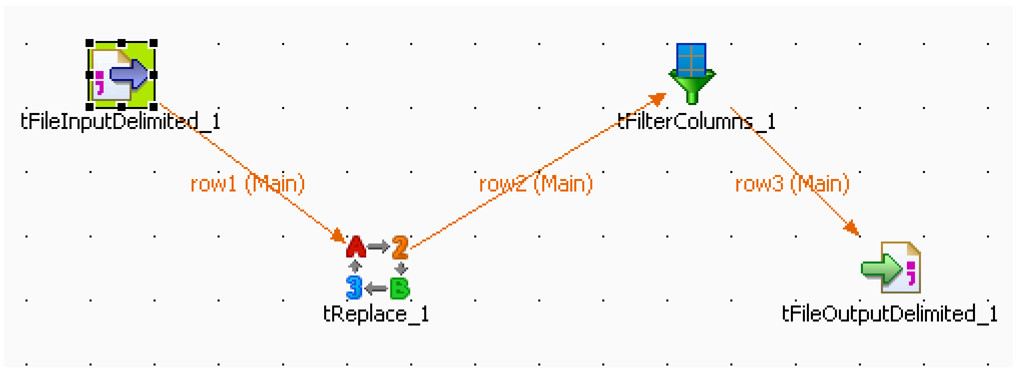
```
[statistics] connecting to socket on port 3914
[statistics] connected
.------+-----+-------+----+----.
|           tLogRow_1          |
|=-----+-----+-------+----+---=|
|name  |sum  |average|max |min |
|=-----+-----+-------+----+---=|
|James |474.0|94.8   |97.0|92.0|
|Peter |475.0|95.0   |98.0|92.0|
|Thomas|472.0|94.4   |98.0|91.0|
'------+-----+-------+----+----'

[statistics] disconnected
```

Results

As shown above, the students' comprehensive scores are aggregated and then sorted in ascending alphabetical order based on the student names.

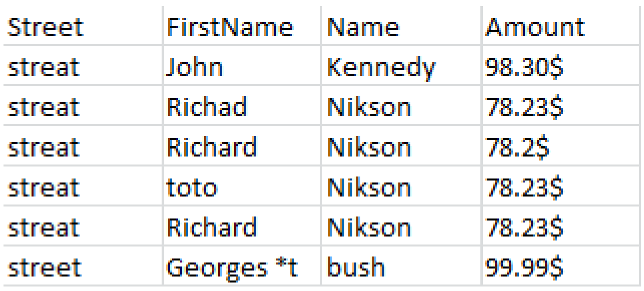
6. tReplace, tFilterColumns:

This Job searches and replaces various typos and defects in a csv file then operates a column filtering before producing a new csv file with the final output.



- The **File** is a simple csv file stored locally. The **Row Separator** is a carriage return and the **Field Separator** is a semi-colon. In the **Header** is the name of the column, and no **Footer** nor **Limit** are to be set.

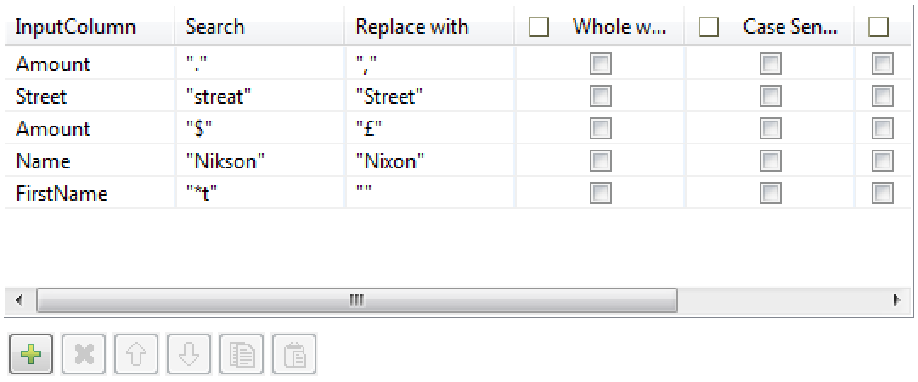
- The file contains characters such as: *t, . or Nikson which we want to turn into Nixon, and streat, which we want to turn into Street.

- 

Street	FirstName	Name	Amount
streat	John	Kennedy	98.30$
streat	Richad	Nikson	78.23$
streat	Richard	Nikson	78.2$
streat	toto	Nikson	78.23$
streat	Richard	Nikson	78.23$
street	Georges *t	bush	99.99$

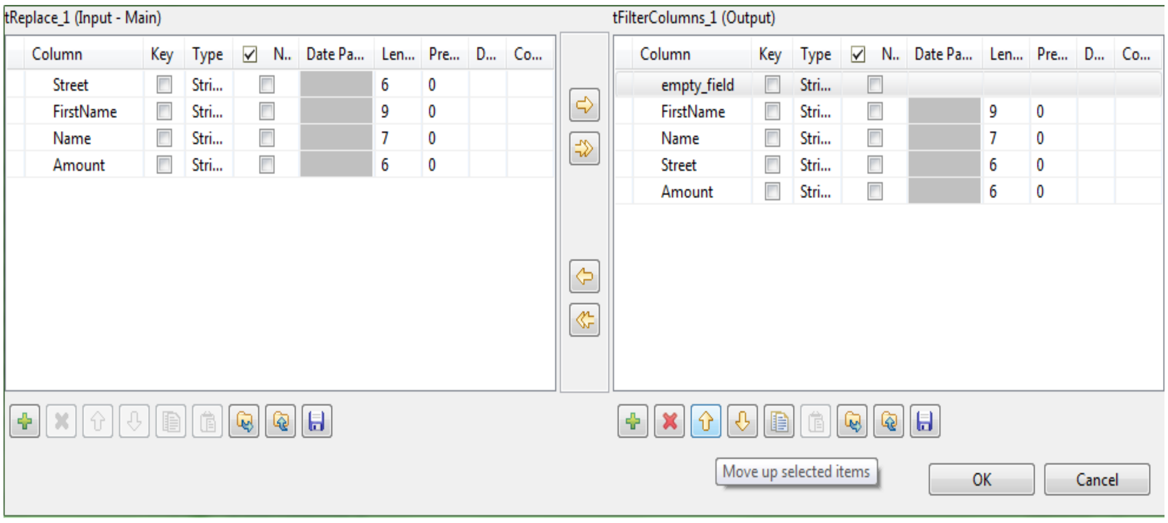
The schema for this file is built in also and made of four columns of various types (string or int).

- Now select the **tReplace** component to set the search & replace parameters.



InputColumn	Search	Replace with	☐ Whole w...	☐ Case Sen...	☐
Amount	"."	","	☐	☐	☐
Street	"streat"	"Street"	☐	☐	☐
Amount	"$"	"£"	☐	☐	☐
Name	"Nikson"	"Nixon"	☐	☐	☐
FirstName	"*t"	""	☐	☐	☐

- Select the next component in the Job, **tFilterColumn**.

- 

tReplace_1 (Input - Main)

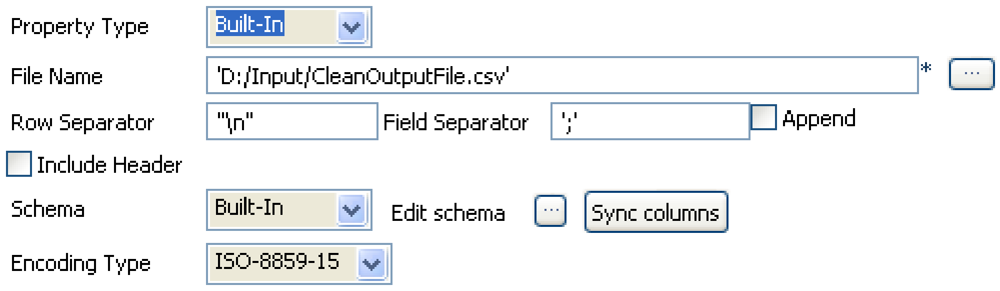
Column	Key	Type	✓ N..	Date Pa...	Len...	Pre...	D...	Co...
Street	☐	Stri...	☐		6	0		
FirstName	☐	Stri...	☐		9	0		
Name	☐	Stri...	☐		7	0		
Amount	☐	Stri...	☐		6	0		

tFilterColumns_1 (Output)

Column	Key	Type	✓ N..	Date Pa...	Len...	Pre...	D...	Co...
empty_field	☐	Stri...	☐					
FirstName	☐	Stri...	☐		9	0		
Name	☐	Stri...	☐		7	0		
Street	☐	Stri...	☐		6	0		
Amount	☐	Stri...	☐		6	0		

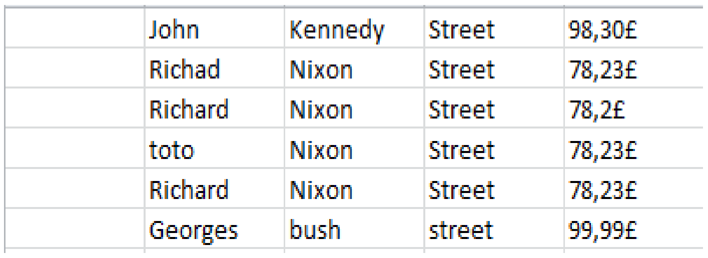
The **tFilterColumn** component holds a schema editor allowing to build the output schema based on the column names of the input schema. In this use case, add one new column named *empty_field* and change the order of the input schema columns to obtain a schema as follows: *empty_field*, *Firstname*, *Name*, *Street*, *Amount*.

- Click **OK** to validate.



Property Type	Built-In			
File Name	'D:/Input/CleanOutputFile.csv'			
Row Separator	"\n"	Field Separator	';'	☐ Append
☐ Include Header				
Schema	Built-In	Edit schema	...	Sync columns
Encoding Type	ISO-8859-15			

- Set the **tFileOutputDelimited** properties manually.
- The schema is built-in for this scenario, and comes from the preceding component in the Job.
- Save the Job and press **F6** to execute it.



	John	Kennedy	Street	98,30£
	Richad	Nixon	Street	78,23£
	Richard	Nixon	Street	78,2£
	toto	Nixon	Street	78,23£
	Richard	Nixon	Street	78,23£
	Georges	bush	street	99,99£

The first column is empty, the rest of the columns have been cleaned up from the parasitical characters, and *Nikson* was replaced with *Nixon*. The street column was moved and the decimal delimiter has been changed from a dot to a comma, along with the currency sign.

Lab Exercise:

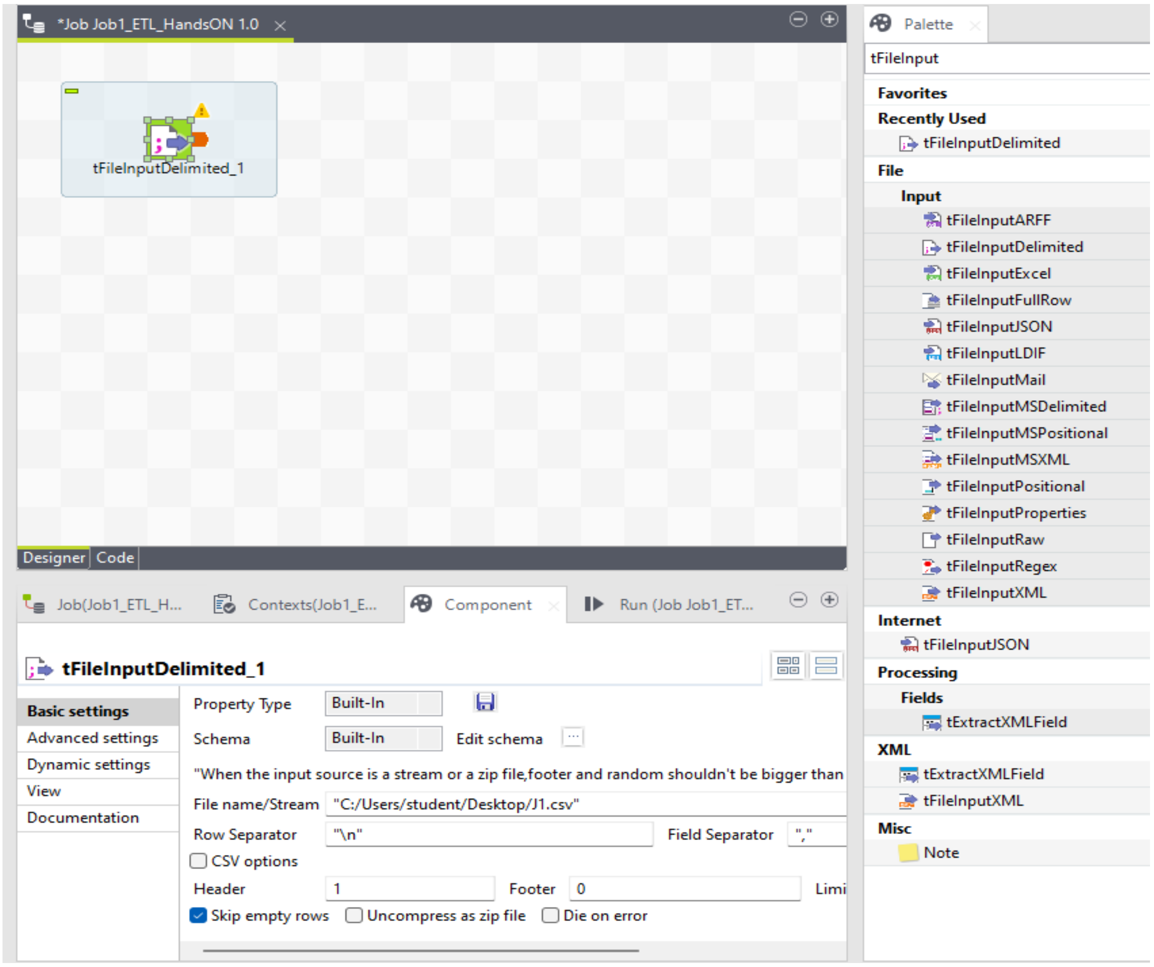
1. Explore and write the uses of the following components used for ETL
   a. tFileInputDelimited
   b. tLogRow:
   c. tFileOutputDelimited
   d. tMap:
   e. tAggregateRow
   f. tSortRow
   g. tReplace
   h. tFilterColumns
   i. tUniqrow
   j. tFileExists
2. Explore the options/components used for ETL in Talend.

LAB NO. 2 **Date**

Executing scenarios using TALEND

Adding components to the Job

Now that the Job is created, components have to be added to the design workspace, a tFileInputDelimited, a tLogRow, and a tFileOutputDelimited in this example.
There are several ways to add a component onto the design workspace. You can:
• find your component on the Palette by typing the search keyword(s) in the search field of the Palette and drop it onto the design workspace.
• add a component by directly typing your search keyword(s) on the design workspace.
• add an output component by dragging from an input component already existing on the design workspace.
• drag and drop a centralized metadata item from the Metadata node onto the design workspace, and then select the component of interest from the Components dialog box.



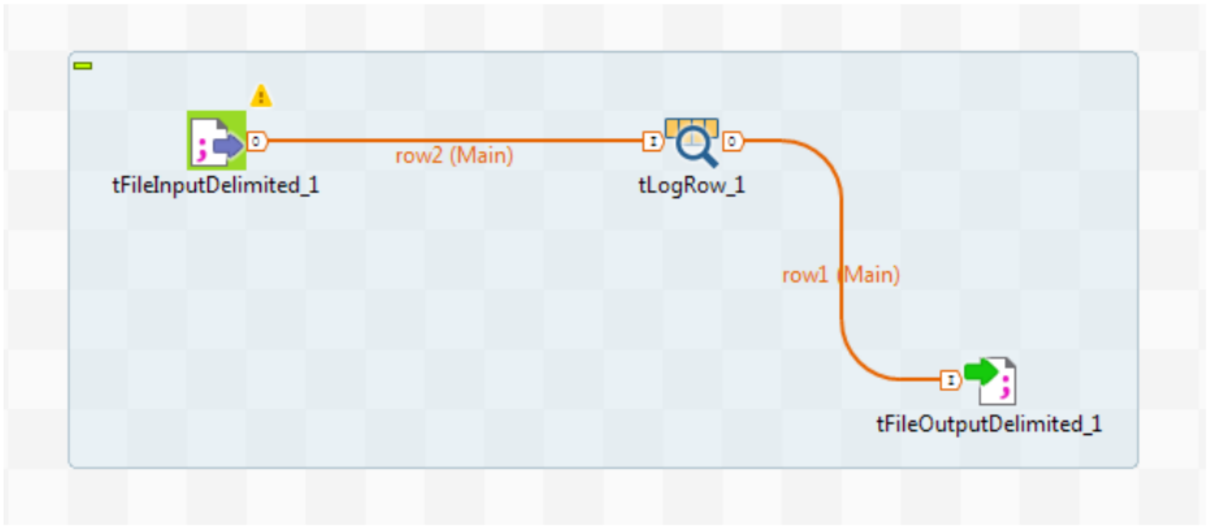
tFileInputDelimited_1

Property Type: Built-In
Schema: Built-In Edit schema
"When the input source is a stream or a zip file,footer and random shouldn't be bigger than
File name/Stream: "C:/Users/student/Desktop/J1.csv"
Row Separator: "\n" Field Separator: ","
☐ CSV options
Header: 1 Footer: 0 Limi
☑ Skip empty rows ☐ Uncompress as zip file ☐ Die on error

Connecting the components together

Now that the components have been added on the workspace, they have to be connected together. Components connected together form a subJob. Jobs are composed of one or several subJobs carrying out various processes.
In this example, as the tLogRow and tFileOutputDelimited components are already connected, you only need to connect the tFileInputDelimited to the tLogRow component.
To connect the components together, use either of the following methods:
Right-click and click again
Procedure

1. Right-click the source component, tFileInputDelimited in this example.
2. In the contextual menu that opens, select the type of connection you want to use to link the components, RowMain in this example.
3. Click the target component to create the link, tLogRow in this example.

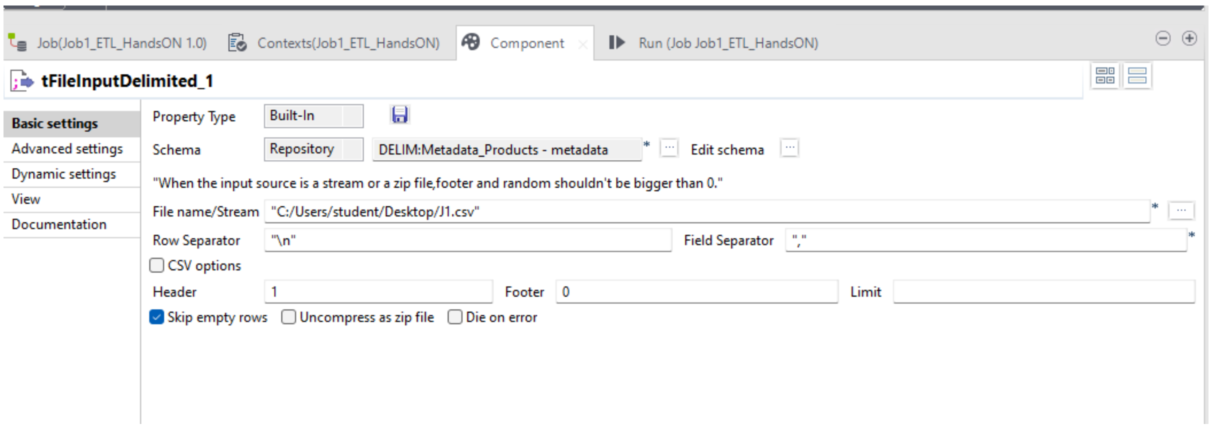


Configuring the components

Now that the components are linked, their properties should be defined.
Configuring the tFileInputDelimited component

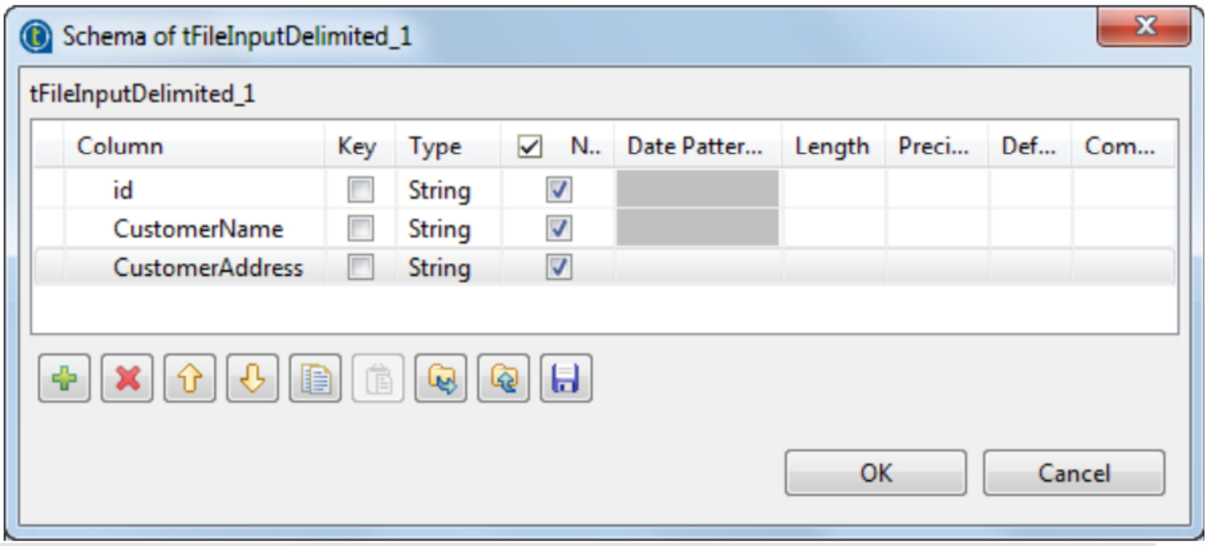
Procedure

1. Double-click the tFileInputDelimited component to open its Basic settings view



Property Type: Built-In
Schema: Repository — DELIM:Metadata_Products - metadata — Edit schema
"When the input source is a stream or a zip file,footer and random shouldn't be bigger than 0."
File name/Stream: "C:/Users/student/Desktop/J1.csv"
Row Separator: "\n" — Field Separator: ","
☐ CSV options
Header 1 Footer 0 Limit
☑ Skip empty rows ☐ Uncompress as zip file ☐ Die on error

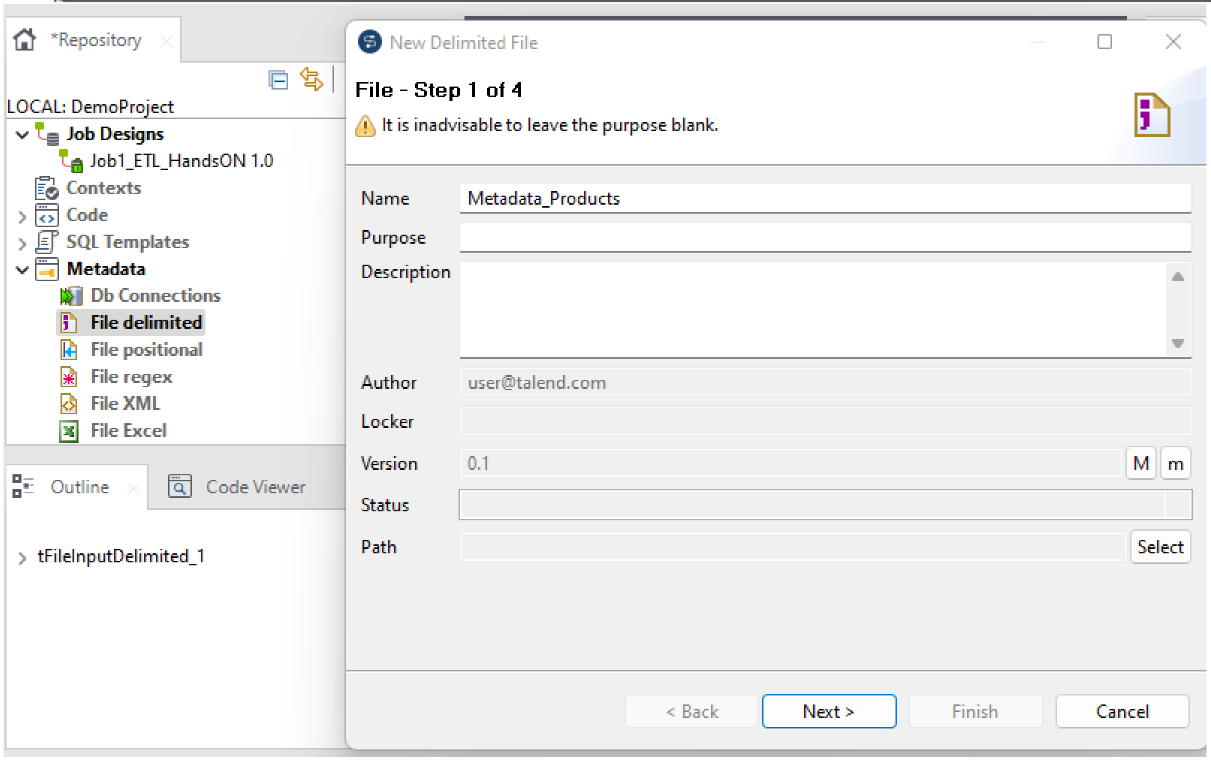
2. Click the [...] button next to the File Name/Stream field.
3. Browse your system or enter the path to the input file, customers.txt in this example.
4. In the Header field, enter 1.
5. Click the [...] button next to Edit schema.
6. In the Schema Editor that opens, click three times the [+] button to add three columns.
7. Name the three columns id, CustomerName and CustomerAddress respectively and click OK to close the editor.



Schema of tFileInputDelimited_1

tFileInputDelimited_1

Column	Key	Type	✓ N..	Date Patter...	Length	Preci...	Def...	Com...
id	☐	String	✓					
CustomerName	☐	String	✓					
CustomerAddress	☐	String	✓					

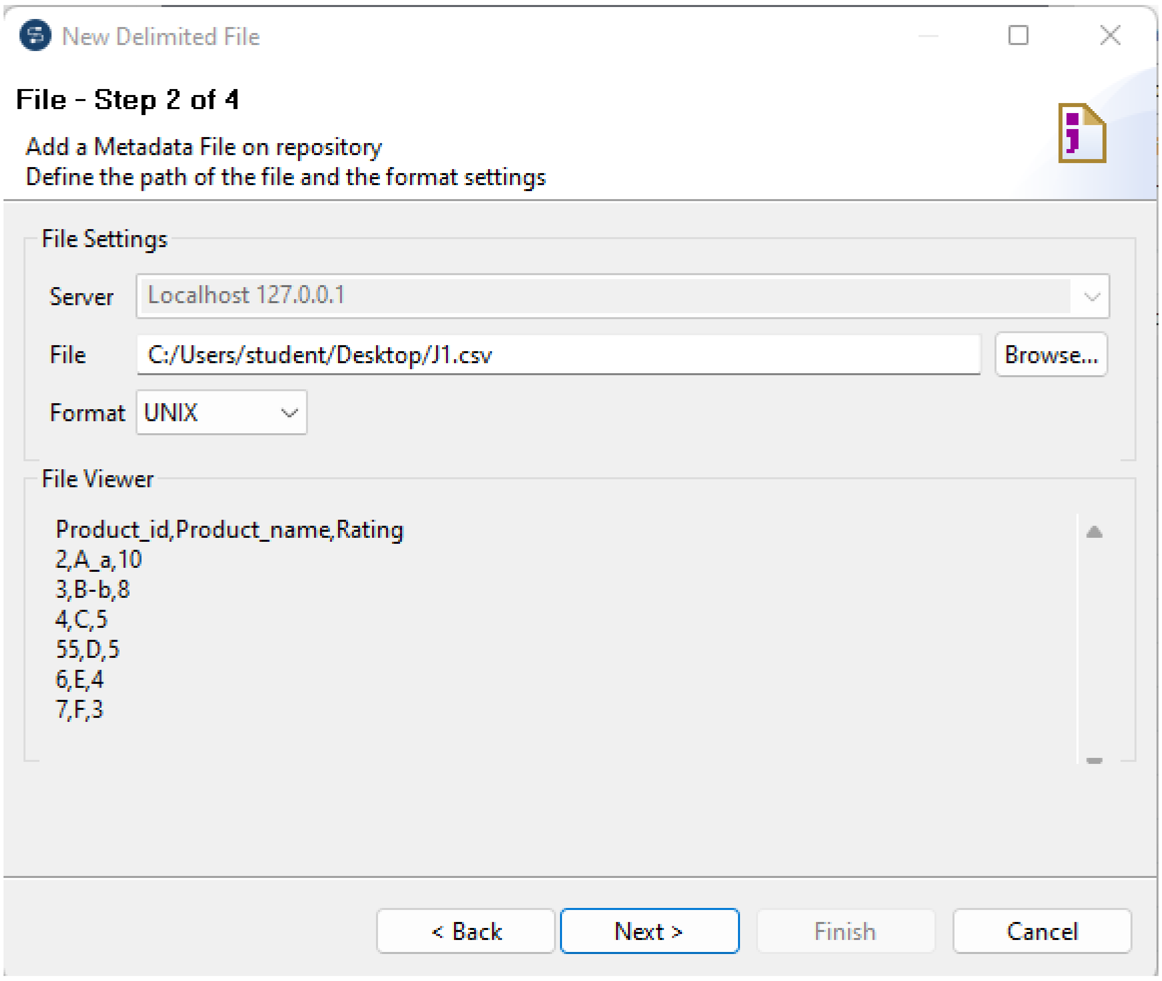


New Delimited File

File - Step 1 of 4

It is inadvisable to leave the purpose blank.

Name	Metadata_Products
Purpose	
Description	
Author	user@talend.com
Locker	
Version	0.1
Status	
Path	



File - Step 2 of 4

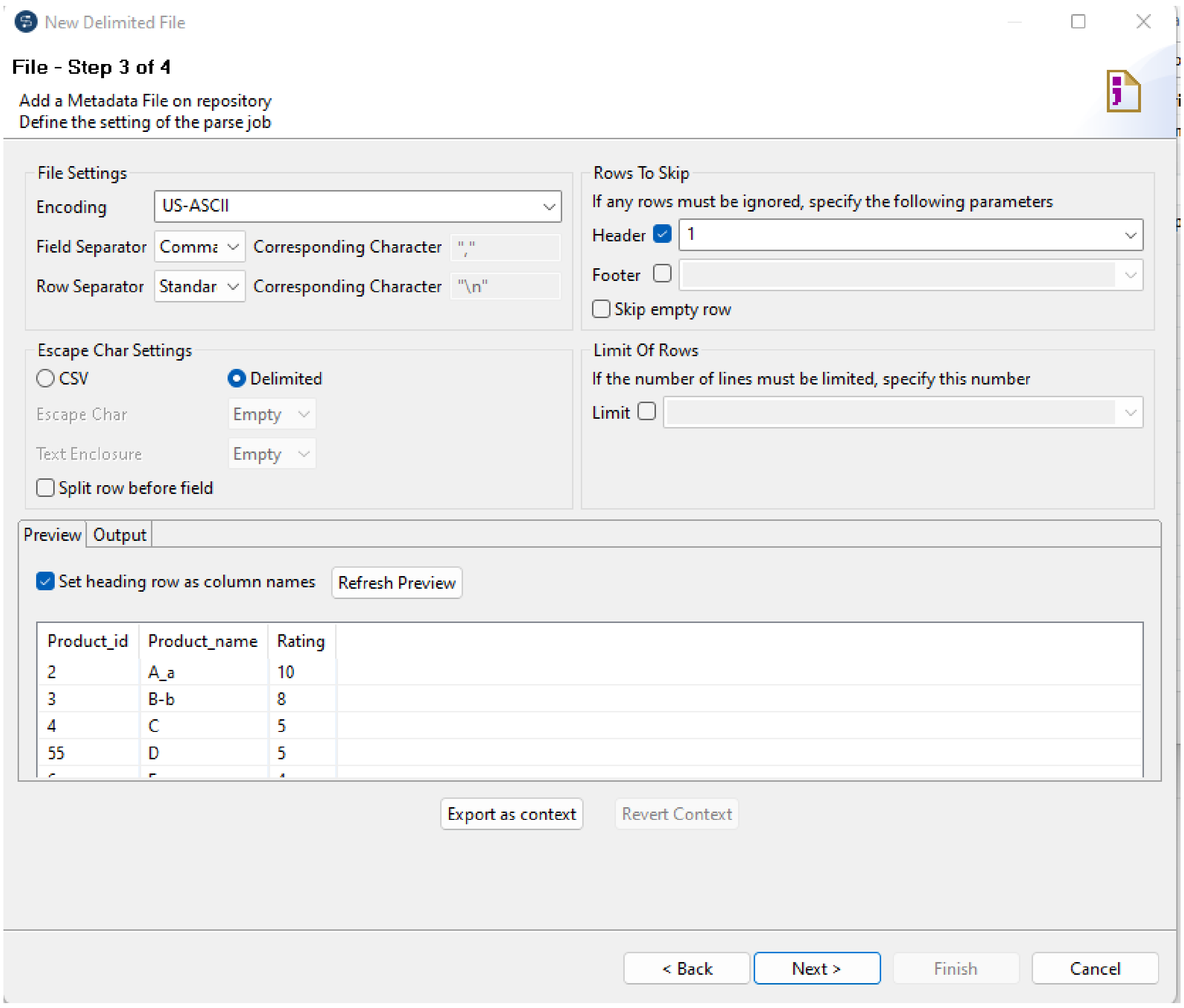
Add a Metadata File on repository
Define the path of the file and the format settings

File Settings

Server: Localhost 127.0.0.1
File: C:/Users/student/Desktop/J1.csv
Format: UNIX

File Viewer

```
Product_id,Product_name,Rating
2,A_a,10
3,B-b,8
4,C,5
55,D,5
6,E,4
7,F,3
```


New Delimited File

File - Step 3 of 4

Add a Metadata File on repository
Define the setting of the parse job

File Settings
Encoding: US-ASCII
Field Separator: Comma — Corresponding Character: ","
Row Separator: Standard — Corresponding Character: "\n"

Rows To Skip
If any rows must be ignored, specify the following parameters
Header ✓ 1
Footer ☐
☐ Skip empty row

Escape Char Settings
○ CSV ● Delimited
Escape Char: Empty
Text Enclosure: Empty
☐ Split row before field

Limit Of Rows
If the number of lines must be limited, specify this number
Limit ☐

Preview | Output

✓ Set heading row as column names — Refresh Preview

Product_id	Product_name	Rating
2	A_a	10
3	B-b	8
4	C	5
55	D	5

Lab Exercise:

1. Consider the hospital dataset with 19 attributes.
   a. Remove all the rows which do not have sample data of any patient.
   b. Select all the rows with provider number ‘10164’ and store in output.csv
   c.

2. Student (Regno, Name, Major, Bdate)
   Course (Course_ID, Cname, dept)
   Enroll (RegNo, Course_ID, marks)
   Book_Adoption (Course_ID, sem, ISBN)
   Text (book_ISBN, title, publisher, author)
   a. Display the course names taken by students of ICT department.
   b. Find whether ICT department has taken up any course which needs books with ISBN=1111.
   c. Find names of students who have scored more than 90 in the course of ‘Compilers’
   d. Find the departments which provides Major.
   e. Find the departments which does not use textbooks of ‘Wiley’ publisher.
   f. Display names of the students who have registered for courses for which more than 10 students have registered.
   g. Display the departments which uses more than 2 books of author ‘Ken’
   h. Find the department which has incorporated textbook by a single publisher.
   i. Find course which has maximum number of students.
   j. Find the total marks of each student in all courses.

Additional Exercise

Execute the following SQL queries using data flows in Infosphere
1.PERSON (driver _ id , name, address)
CAR (Regno, model, year)
ACCIDENT (report_number, date, location)
OWNS (driver-id, Regno)
PARTICIPATED (driver-id, Regno , report_number, damage)

i. Select registration number of cars which do not come in between the model year 2000 and 2010.
ii. Find driver names whose sum of damage amount of all his accidents is less than average
iii. damage amount of all accidents in the database.
iv. Find driver names whose sum of damage amount of all his accidents is less than average damage amount of all accidents in database.
v. Find driver ids' who have met with accident more than once but with different cars owned by him and damage amount is greater than Rs.50000.
vi. Find driver id of persons with name 'john'.

LAB NO: 3 **Date:**

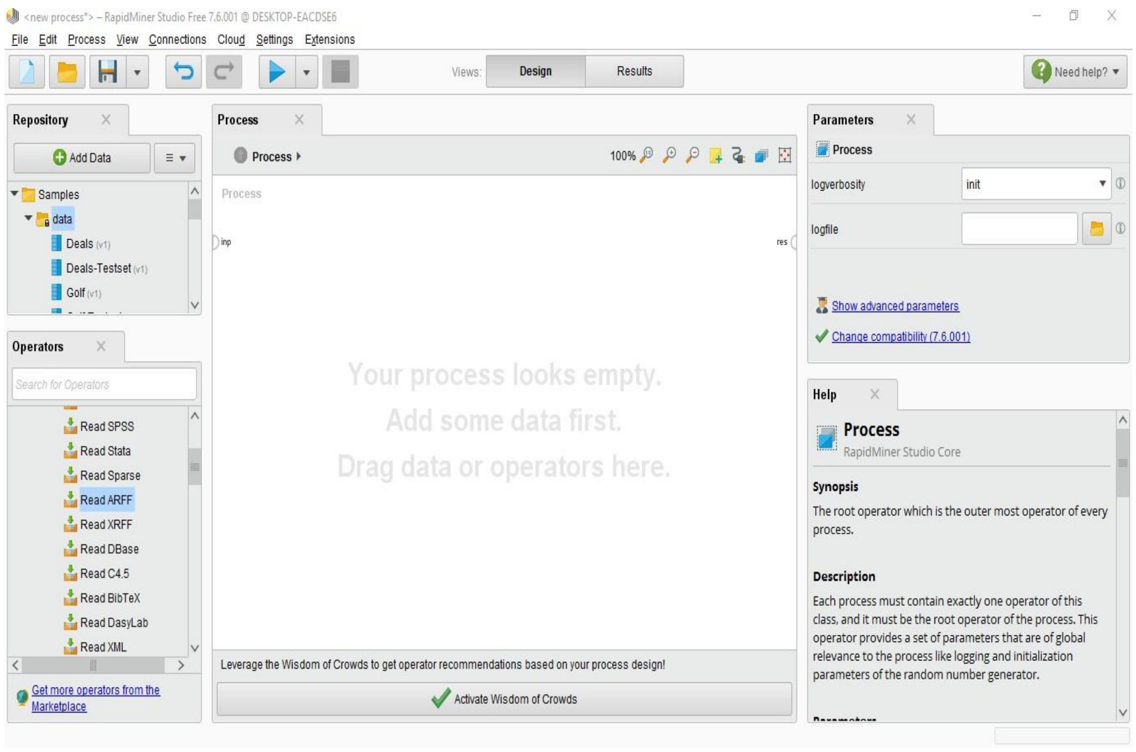
RAPID MINER OPERATORS

Objectives:

1. To understand the different operators for data access in Rapid miner.
2. To understand the different operators for data preprocessing in Rapid miner.

Introduction:

Rapid Miner Studio is open source and in process view of the Studio there are five panels. They are Repository, Operators, Process, Parameters and Help. Figure 4.1 shows the overall panels in process view of the tool. Rapid Miner Studio provides operators for Data Access, Blending, Cleansing, Modeling, Scoring, Validation and Utility.



Leverage the Wisdom of Crowds to get operator recommendations based on your process design!

Activate Wisdom of Crowds

logverbosity: init

logfile:

Synopsis

The root operator which is the outer most operator of every process.

Description

Each process must contain exactly one operator of this class, and it must be the root operator of the process. This operator provides a set of parameters that are of global relevance to the process like logging and initialization parameters of the random number generator.

Figure 4.1. Process View

This section provides details about various operators in Rapid Miner Studio such as Data Access operators and Data preprocessing operators.

1. Data Access Operators

This section has operators to

i. read and write data from and into File respectively.
ii. read, write and update Database
iii. obtain data from Web Applications

i. Reading from Files

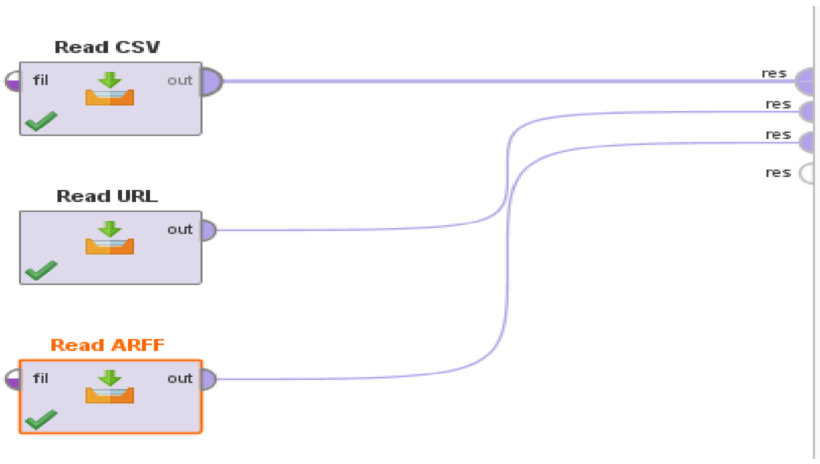


Figure 4.2. Example operators for reading from file

As shown in Figure 4.2 drag and drop Read CSV, Read URL and Read ARFF operators to the process panel and connect their out ports to res port. Set the configuration for each operator in parameter panel by clicking on respective operator icon. In order to see the result click on Execute button in top panel.

ii. Read-Write operation

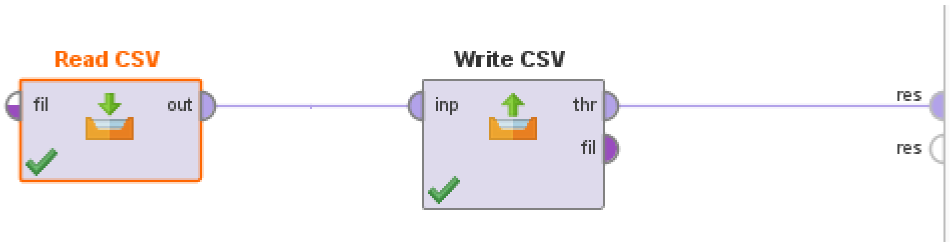


Figure 4.3. Read Write Operation

As shown in Figure 4.3, in order to write into a CSV file drag and drop a Read CSV and Write CSV operators to the Process Panel and then connect out port of Read CSV operator icon to inp port of Write CSV operator. The thr port of Write CSV should be connected to res port. The path of the file into which content has to be written should be given in parameter panel. Using repeated Write operator, it is possible to write into multiple files. Rapid miner also provides sample data set whose operators can be dragged and dropped on to process panel. Table 4.1 provides various port abbreviations and their meaning and description in each operator icon.

Table 4.1. Port Abbreviations

Port Abbreviation	**Meaning**	**Description**
Ass	*Association*	Association rules that have been discovered in a frequent item set
Att	*Attribute*	Attribute weights (in and out)

Ave	*Average*	Performance measures; estimate of performance using the model built on the complete delivered data set
Clu	*Cluster model*	Cluster model created when clustering an example set
Exa	*Example set*	Example set
Fil	*File*	File object
For	*Formula*	Formula result
Fre	*Frequent*	Frequent item or item sets for association rule learning
Gro	*Grouped*	Grouped models, attributes, items
Hie	*Hierarchical*	Hierarchical clustering model
Inp	*Input*	Input source, can take various objects
Ite	*Item sets*	Frequent item sets (groups of items that often appear together in the data)
Joi	*Join*	Join of the left and right example sets
Lab	*Labeled data*	Model that was given in input is applied on the example set and the updated example set is delivered from this port
Lef	*Left*	Left input port expecting an example set, which is used as the left example set for a join
Mat	*Matrix*	Correlations matrix of all attributes of the input example set
Mer	*Merged*	Merged example set
Mod	*Model*	Default model from this output port
Obj	*Object*	IO object
Ori	*Original*	Input example set is passed without changing to this port
Out	*Output*	Output port

Adding Twitter Data

Drag and drop the twitter icon on process panel. Connect the out port to res port as depicted in figure 4.4.

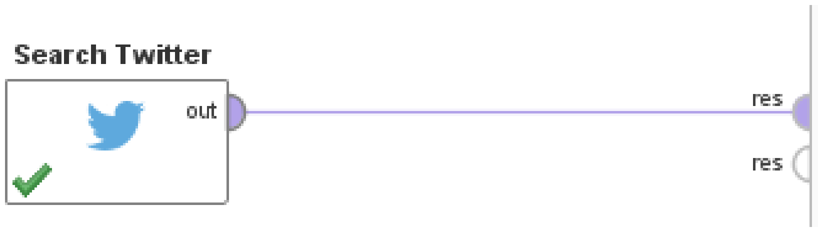


Figure 4.4. Adding Twitter data

In the parameter window, do the following to set the connection:

1. Create an access token for Rapid Miner to request twitter feed.
2. Click on twitter symbol against connection field.
3. Click Add Connection - Give a connection name and select connection type as twitter connection from the drop down. Click on create button.
4. Against access token field, paste the access token received (can be created in https://apps.twitter.com/app/) and click on button, Request access token as shown in Figure 4.5.

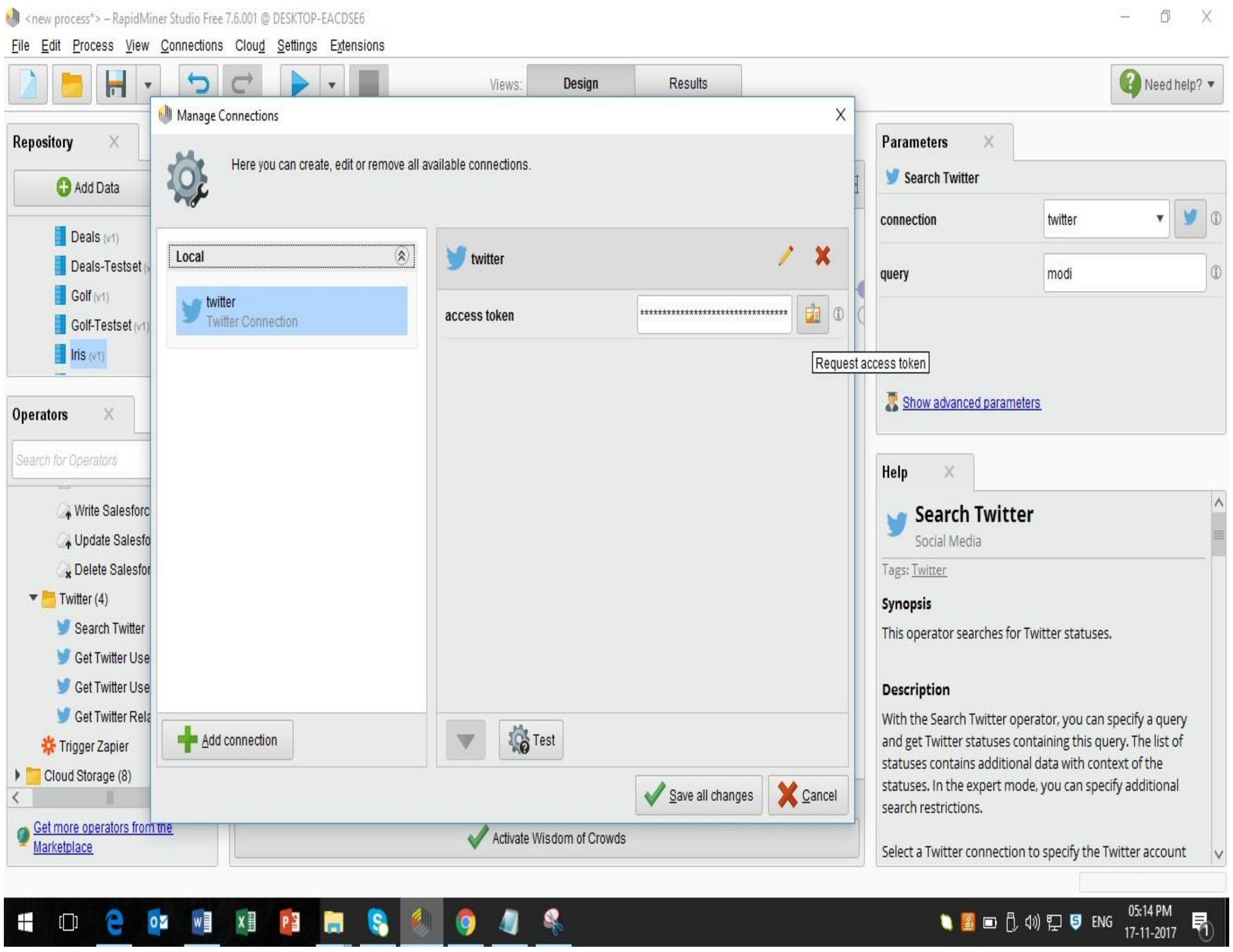
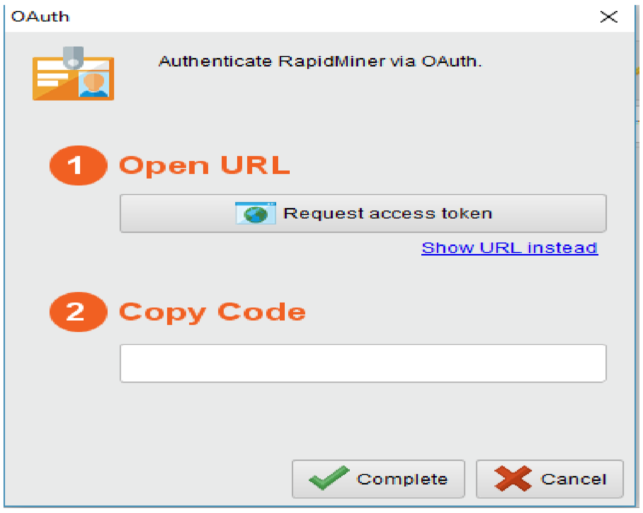

<new process*> – RapidMiner Studio Free 7.6.001 @ DESKTOP-EACDSE6
File Edit Process View Connections Cloud Settings Extensions
Views: Design Results
Need help?
Repository
Add Data
Deals (v1)
Deals-Testset
Golf (v1)
Golf-Testset (v1)
Iris (v1)
Operators
Search for Operators
Write Salesforc
Update Salesfo
Delete Salesfor
Twitter (4)
Search Twitter
Get Twitter Use
Get Twitter Use
Get Twitter Rela
Trigger Zapier
Cloud Storage (8)
Get more operators from the Marketplace
Manage Connections
Here you can create, edit or remove all available connections.
Local
twitter
Twitter Connection
twitter
access token
Request access token
Add connection
Test
Save all changes
Cancel
Activate Wisdom of Crowds
Parameters
Search Twitter
connection
twitter
query
modi
Show advanced parameters
Help
Search Twitter
Social Media
Tags: Twitter
Synopsis
This operator searches for Twitter statuses.
Description
With the Search Twitter operator, you can specify a query and get Twitter statuses containing this query. The list of statuses contains additional data with context of the statuses. In the expert mode, you can specify additional search restrictions.
Select a Twitter connection to specify the Twitter account
ENG
05:14 PM
17-11-2017

Figure 4.5. Setting access token

5. On click of the Request access token button the window shown in Figure 4.6 appears.


OAuth

Authenticate RapidMiner via OAuth.

1 Open URL

Request access token

Show URL instead

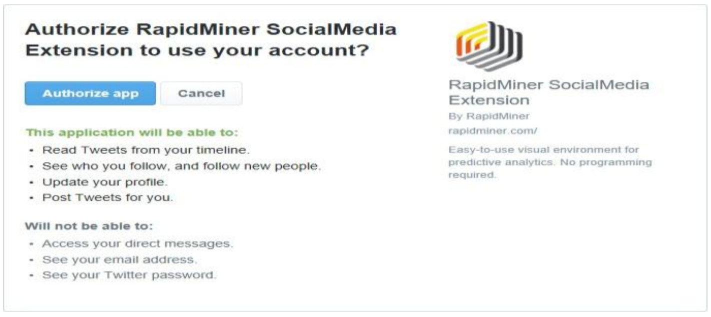
2 Copy Code

Complete

Cancel

Figure 4.6. Authentication

6. Click on Request access token the web page shown in Figure 4.7 appears.


Authorize RapidMiner SocialMedia Extension to use your account?

Authorize app Cancel

This application will be able to:
- Read Tweets from your timeline.
- See who you follow, and follow new people.
- Update your profile.
- Post Tweets for you.

Will not be able to:
- Access your direct messages.
- See your email address.
- See your Twitter password.

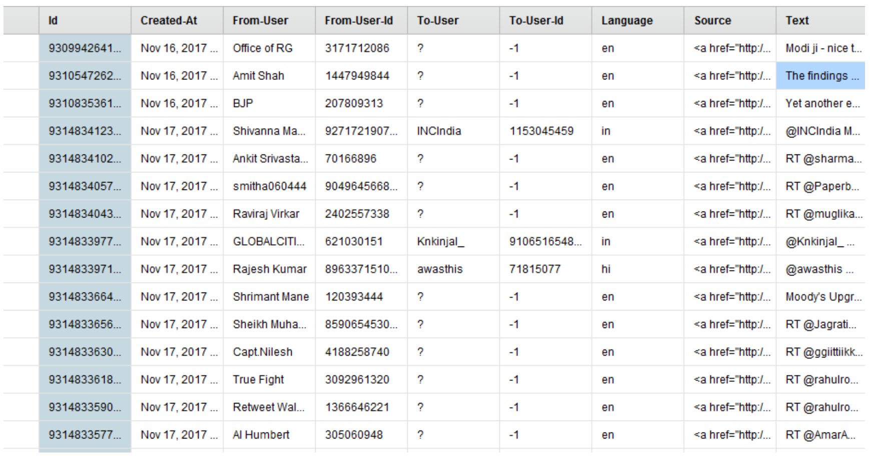
RapidMiner SocialMedia Extension
By RapidMiner
rapidminer.com/
Easy-to-use visual environment for predictive analytics. No programming required.

Figure 4.7. Authorization

7. Click on Authorize app.

8. Enter the PIN received in "COPY CODE" field. Click on complete.
9. Click on Test and followed by Save all changes.
10. Enter a query in query field. Example "Modi"
11. Execute.

The result of these steps is shown in Figure 4.8



Id	Created-At	From-User	From-User-Id	To-User	To-User-Id	Language	Source	Text
9309942641...	Nov 16, 2017 ...	Office of RG	3171712086	?	-1	en	<a href="http:/...	Modi ji - nice t...
9310547262...	Nov 16, 2017 ...	Amit Shah	1447949844	?	-1	en	<a href="http:/...	The findings ...
9310835361...	Nov 16, 2017 ...	BJP	207809313	?	-1	en	<a href="http:/...	Yet another e...
9314834123...	Nov 17, 2017 ...	Shivanna Ma...	9271721907...	INCIndia	1153045459	in	<a href="http:/...	@INCIndia M...
9314834102...	Nov 17, 2017 ...	Ankit Srivasta...	70166896	?	-1	en	<a href="http:/...	RT @sharma...
9314834057...	Nov 17, 2017 ...	smitha060444	9049645668...	?	-1	en	<a href="http:/...	RT @Paperb...
9314834043...	Nov 17, 2017 ...	Raviraj Virkar	2402557338	?	-1	en	<a href="http:/...	RT @muglika...
9314833977...	Nov 17, 2017 ...	GLOBALCITI...	621030151	Knkinjal_	9106516548...	in	<a href="http:/...	@Knkinjal_ ...
9314833971...	Nov 17, 2017 ...	Rajesh Kumar	8963371510...	awasthis	71815077	hi	<a href="http:/...	@awasthis ...
9314833664...	Nov 17, 2017 ...	Shrimant Mane	120393444	?	-1	en	<a href="http:/...	Moody's Upgr...
9314833656...	Nov 17, 2017 ...	Sheikh Muha...	8590654530...	?	-1	en	<a href="http:/...	RT @Jagrati...
9314833630...	Nov 17, 2017 ...	Capt.Nilesh	4188258740	?	-1	en	<a href="http:/...	RT @ggiittiikk...
9314833618...	Nov 17, 2017 ...	True Fight	3092961320	?	-1	en	<a href="http:/...	RT @rahulro...
9314833590...	Nov 17, 2017 ...	Retweet Wal...	1366646221	?	-1	en	<a href="http:/...	RT @rahulro...
9314833577...	Nov 17, 2017 ...	Al Humbert	305060948	?	-1	en	<a href="http:/...	RT @AmarA...

Figure 4.8. Result of Twitter data access

Database Connection in Rapid Miner

1. Drag and drop the Read Database operator on the process panel as shown in Figure 4.9.

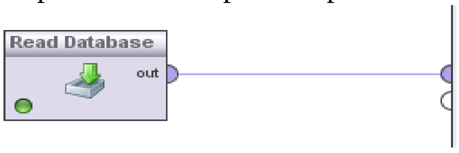
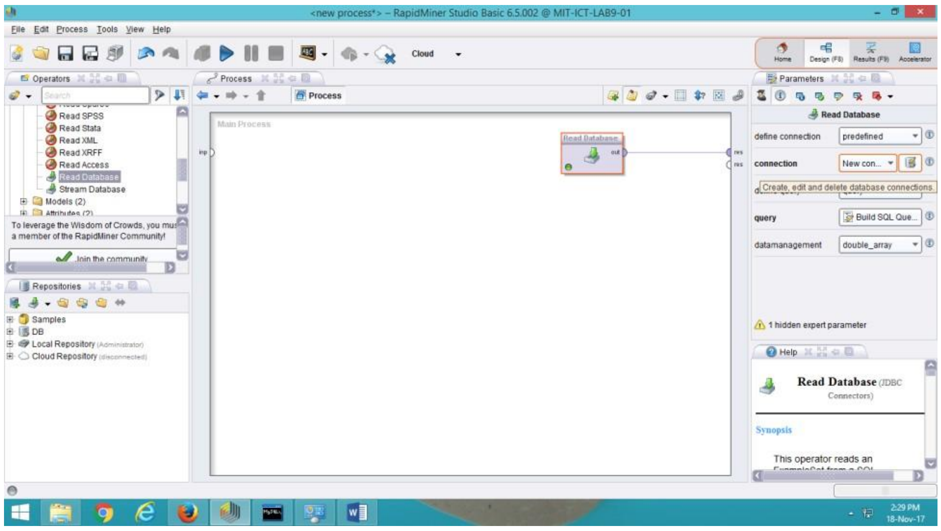


Figure 4.9. Data base access

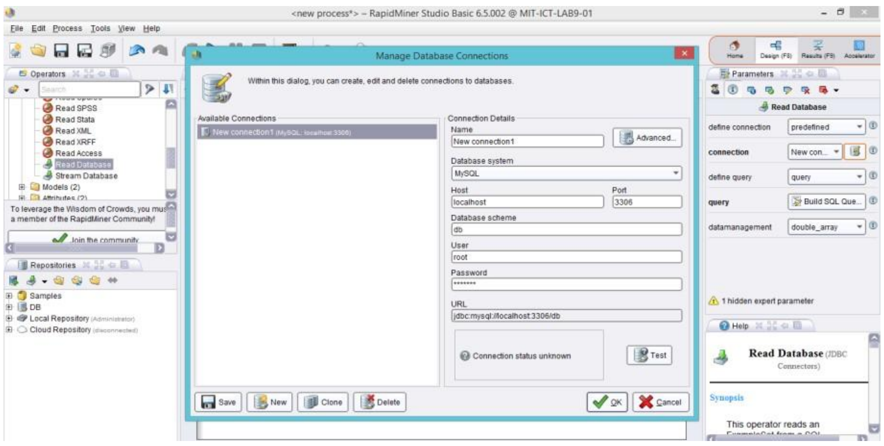
2. Do the following configuration in Property window:
   a) Set the define connection to “predefined”.
   b) Click on the button “create, edit, delete database connections, beside connection as shown in the screenshot below in Figure 4.10.


<new process*> – RapidMiner Studio Basic 6.5.002 @ MIT-ICT-LAB9-01
File Edit Process Tools View Help
Operators
Process
Parameters
Read Database
Create, edit and delete database connections.
1 hidden expert parameter
Help
Read Database (JDBC Connectors)
Synopsis
This operator reads an

define connection: predefined
connection: New con...
query: Build SQL Que...
datamanagement: double_array

Figure 4.10. Screen shot for setting parameters for DB connection

   c) A window (Figure 4.11) will appear in which the following information must be entered:
      - Connection name
      - Database system: MySql
      - Host: Localhost, Port:3306
      - Database scheme (The database name created in the backend)
      - User:root
      - Password:student

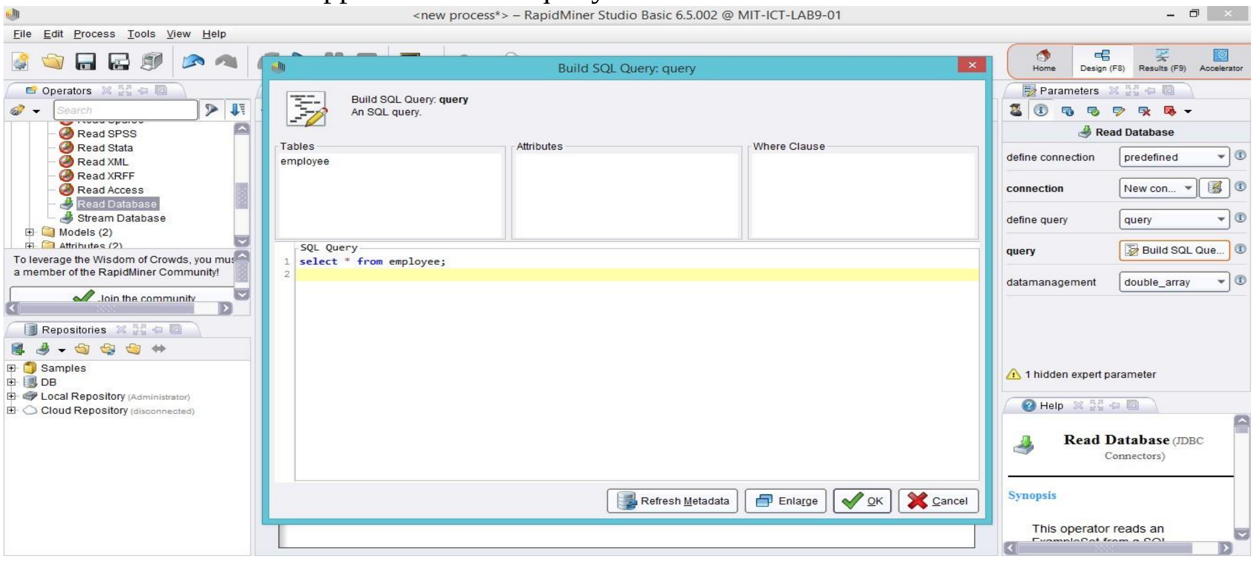

Manage Database Connections
Within this dialog, you can create, edit and delete connections to databases.
Available Connections
New connection1 (MySQL, localhost:3306)
Connection Details
Advanced...
Connection status unknown
Test
Save
New
Clone
Delete
OK
Cancel

Name: New connection1
Database system: MySQL
Host: localhost
Port: 3306
Database scheme: db
User: root
Password: *******
URL: jdbc:mysql://localhost:3306/db

Figure 4.11. Setting parameters for DB access

d) Click on “Test” button.

- Click on the query button in the property window. The window as shown in Figure 4.12 will appear where the query is to be entered.



```
select * from employee;
```

define connection	predefined
connection	New con...
define query	query
query	Build SQL Que...
datamanagement	double_array

Figure 4.12. Entering SQL query for DB Access

3.Execute the process to obtain the data required by the query.

2. Data Preprocessing Operators

I. Blending Operators

A. Attributes

i. Names & Roles

- **Rename:** Renames the attribute names.

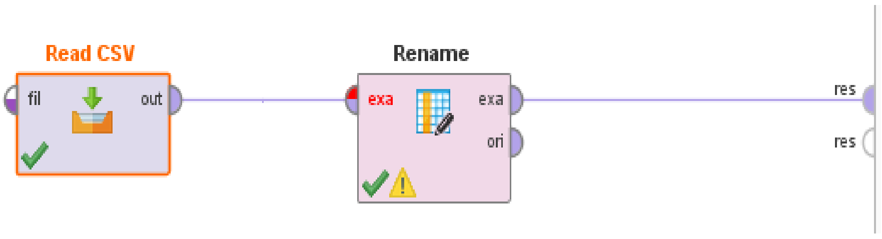


Figure 4.13. Renaming file

Figure 4.13 depicts the process diagram for renaming a file. In the parameter window of Rename operator specify the old attribute name in dataset file and the new name which user requires.

- **Rename by Generic Names**: This operator renames the selected attributes of the given ExampleSet to a set of generic names like att1, att2, att3 etc. Figure 4.14 shows the process diagram for usage of the Rename by Generic Names operator.

The Write Constructions operator writes all attributes of the ExampleSet given at the input port into the specified file. The path of the file is specified through the *attribute constructions* *file* parameter. Each line in the file holds the construction description of one attribute. The Read Constructions operator can be used for reading this file.

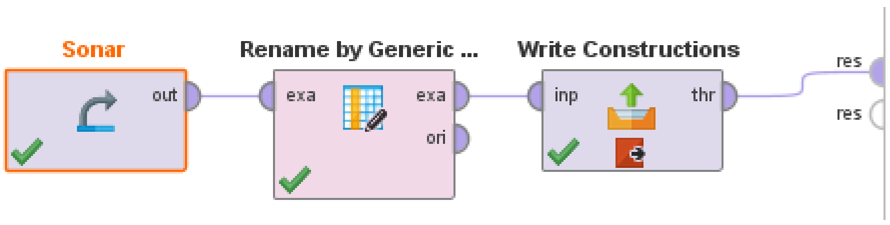


Figure 4.14. Rename by generic names

ii. **Types:** This operator changes the type of the selected numeric attributes to a binominal type. It also maps all values of these attributes to corresponding binominal values. Figure 4.15 shows the process diagram for numeric to binomial operator.

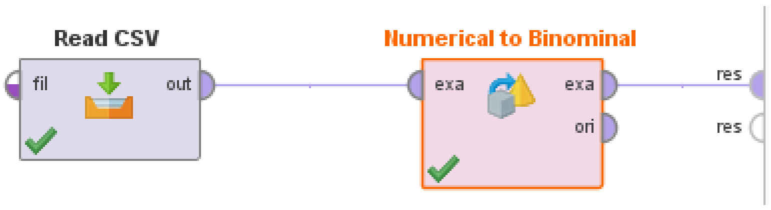
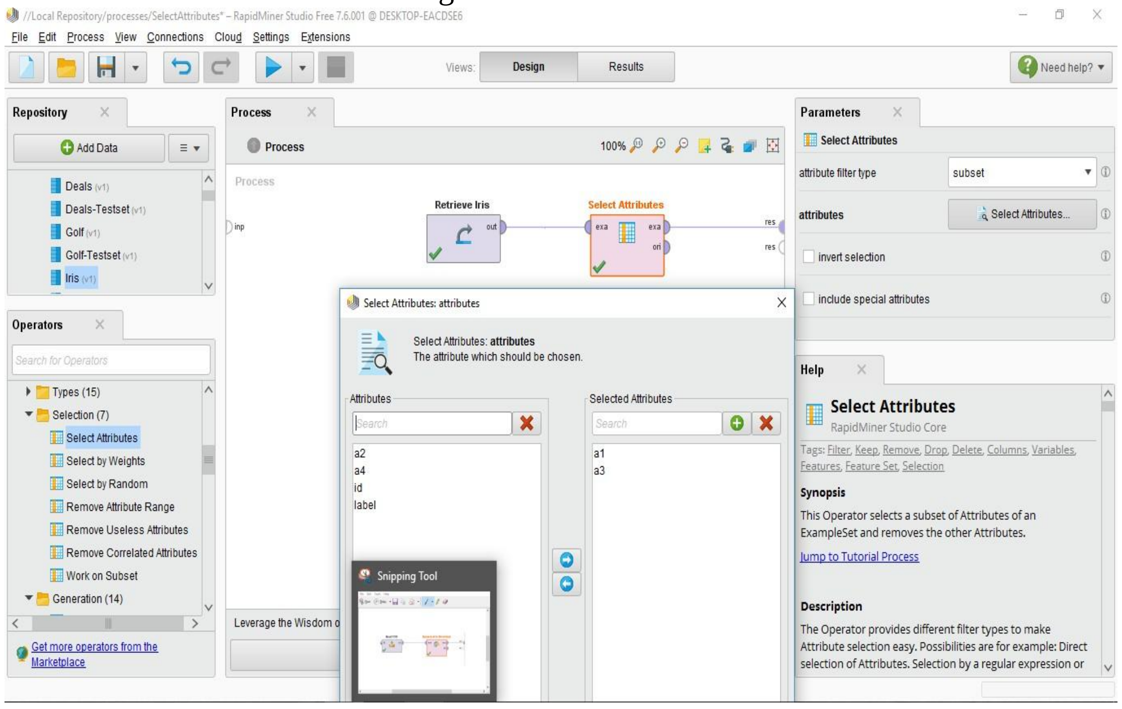


Figure 4.15. Types

iii. **Selection**

- **Select Attributes**: This Operator selects a subset of Attributes of an ExampleSet and removes the other Attributes. The Figure 4.16 demonstrates that a subset of attribute is selected in the parameter window and attributes which need to appear is selected in the "Select Attribute button" against "attributes" field.



Select Attributes

attribute filter type: subset

attributes: Select Attributes...

invert selection

include special attributes

Select Attributes: attributes

Select Attributes: **attributes**
The attribute which should be chosen.

Attributes	Selected Attributes
a2	a1
a4	a3
id	
label	

Select Attributes
RapidMiner Studio Core

Tags: Filter, Keep, Remove, Drop, Delete, Columns, Variables, Features, Feature Set, Selection

Synopsis
This Operator selects a subset of Attributes of an ExampleSet and removes the other Attributes.

Jump to Tutorial Process

Description
The Operator provides different filter types to make Attribute selection easy. Possibilities are for example: Direct selection of Attributes. Selection by a regular expression or

-

Figure 4.16. Setting parameters for selection

- **Remove correlated attributes:** This operator removes correlated attributes from an ExampleSet. The correlation threshold is specified by the user in the correlation field of parameter window. Figure 4.17 shows the process diagram for the operator.

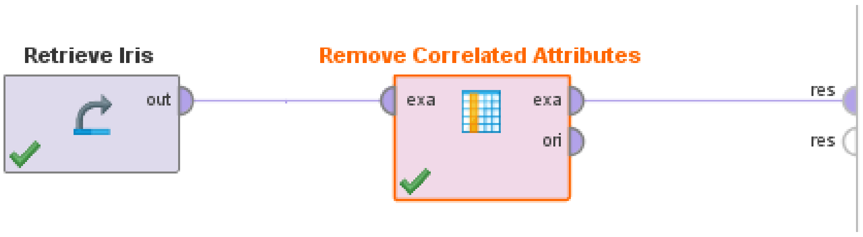


Figure 4.17. Removing correlated attributes

iv. Genertation

- **tf-idf:** This operator performs a TF-IDF filtering of the given ExampleSet. The usage of the operator is shown in process diagram of Figure 4.18.

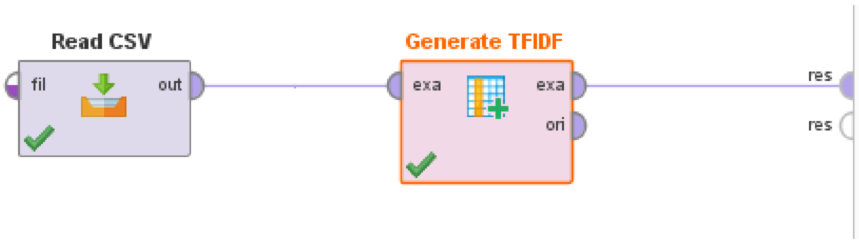
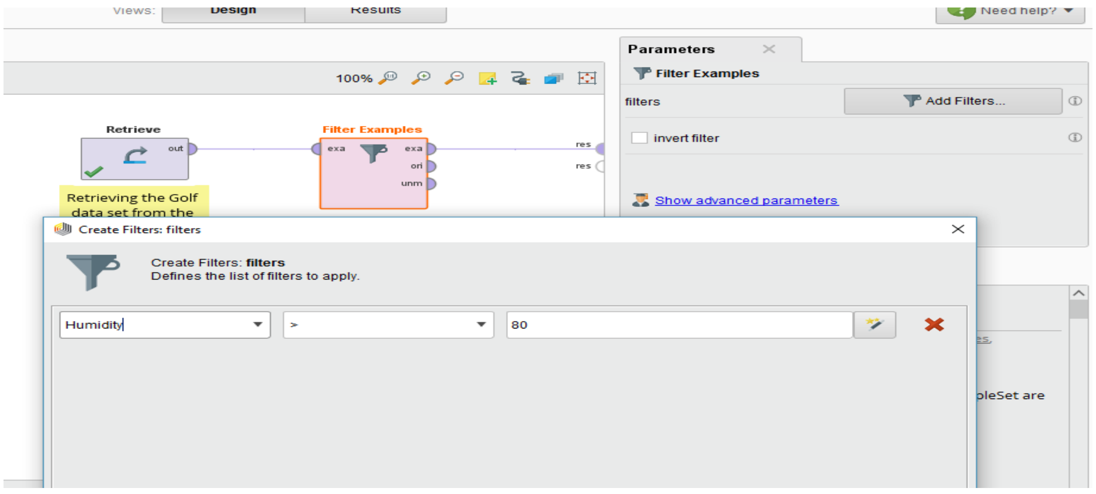


Figure 4.18. Generation of TFIDF

B. Examples

i) Filter Example operator

Filters examples based on a condition applied on an attribute. Figure 4.19 shows the parameter setting window where the user needs to enter attribute and values based on which dataset must be filtered.



Retrieving the Golf data set from the

Create Filters: filters

Create Filters: **filters**
Defines the list of filters to apply.

Humidity | > | 80

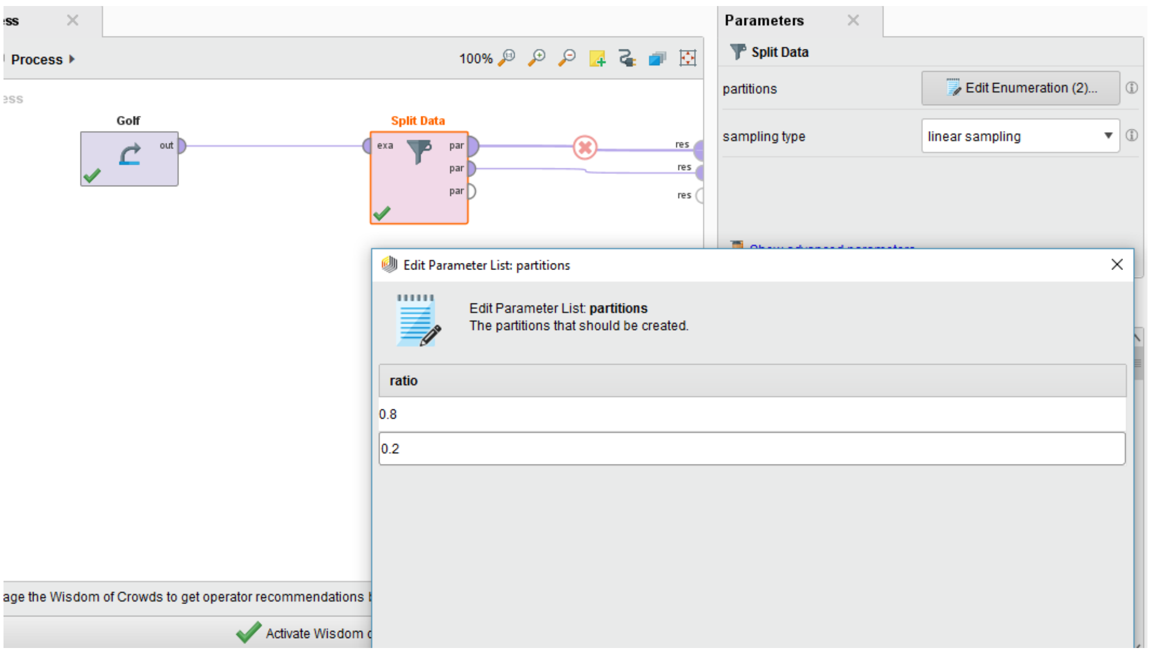
pleSet are

Figure 4.19. Filter operator parameter setting

ii) Sampling operator

- **split data:**

Splitting data set according to the ratio provided in the parameter panel. Figure 4.20 shows the parameter setting window where the ratios add to 1 and linear sampling is selected.



Parameters: Split Data

partitions: Edit Enumeration (2)...

sampling type: linear sampling

Edit Parameter List: partitions

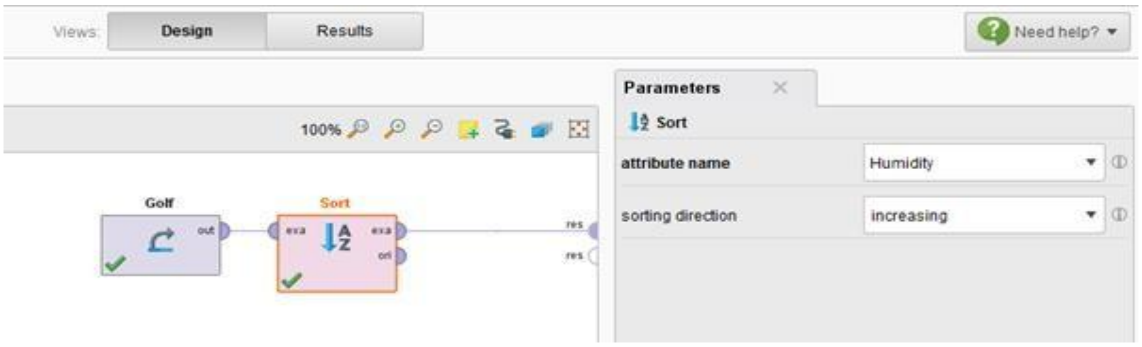
Edit Parameter List: partitions
The partitions that should be created.

ratio
0.8
0.2

age the Wisdom of Crowds to get operator recommendations

Figure 4.20. Sampling input data set.

ii)Sorting: Figure 4.21 depicts the process and parameter setting for the sort operator.



Parameters: Sort

attribute name: Humidity

sorting direction: increasing

Figure 4.21. Sorting

C. Tables

i)Rotation - Transpose: Figure 4.22 depicts the process diagram for transpose operator.

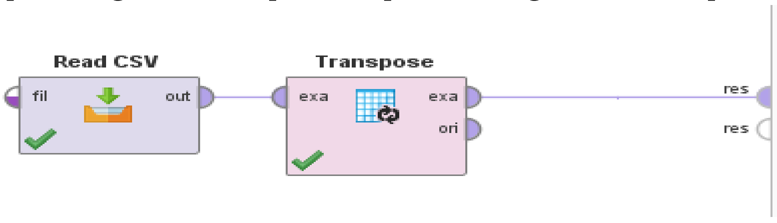


Figure 4.22. Finding Transpose

ii) Joins - Set minus operator: Figure 4.23 depicts the process diagram for Set Minus operator

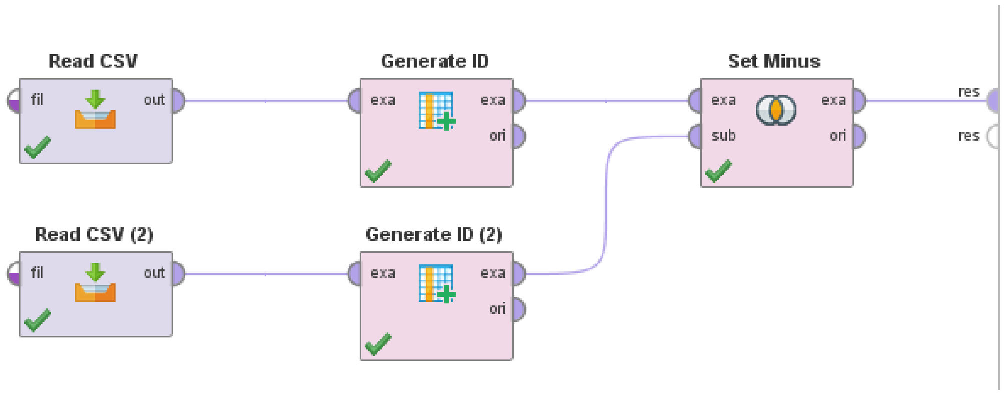
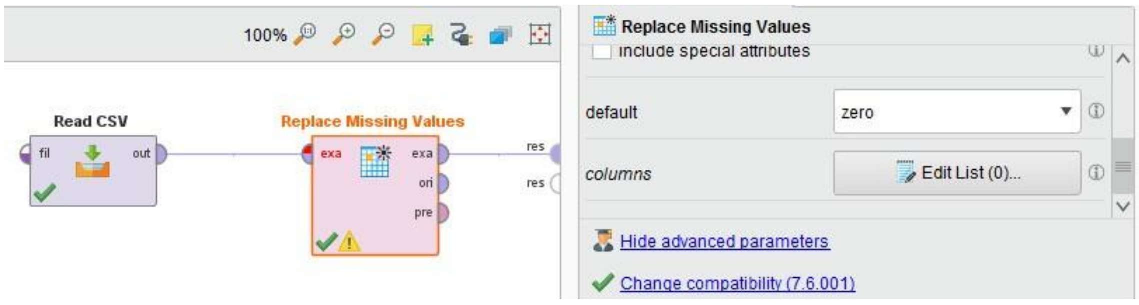


Figure 4.23. Subtraction of Data set.

1. Data Cleansing Operators:

i) Normalize: This operator normalizes the attribute values of the selected attributes.

ii) Replace Missing Values: This operator replaces missing values in examples of selected attributes by a specified replacement. Figure 4.24 shows the parameter setting where all missing values would be filled with zeros.



Replace Missing Values
include special attributes
default: zero
columns

Figure 4.24. Replacing missing values

iii) Remove Duplicates: This operator removes duplicate examples from an ExampleSet by comparing all examples with each other on the basis of the specified attributes. Two examples are considered duplicate if the selected attributes have the same values in them. Figure 4.25 shows the process diagram for remove duplicates.

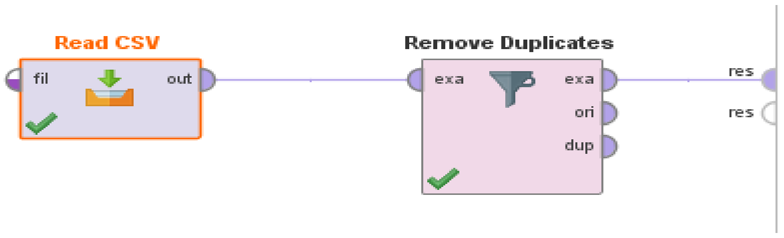
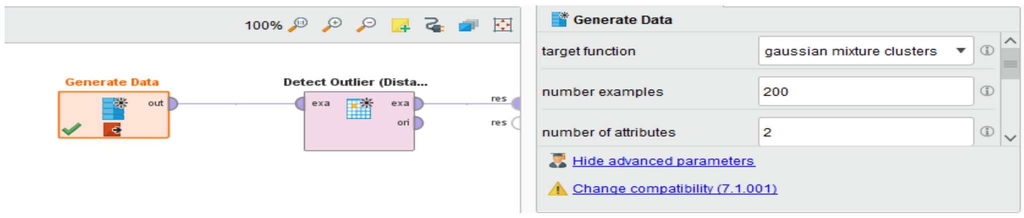


Figure 4.25. Removing duplicates

iv) Detect Outlier : This operator identifies *n* outliers in the given ExampleSet based on the distance to their *k* nearest neighbors. Figure 4.26 shows the variables *n* and *k* that can be specified through parameters as 200 and 2 respectively. The output of the outlier detection is shown in Figure 4.27..



Generate Data

target function: gaussian mixture clusters

number examples: 200

number of attributes: 2

Hide advanced parameters

Change compatibility (7.1.001)

Figure 4.26. Detection of outliers

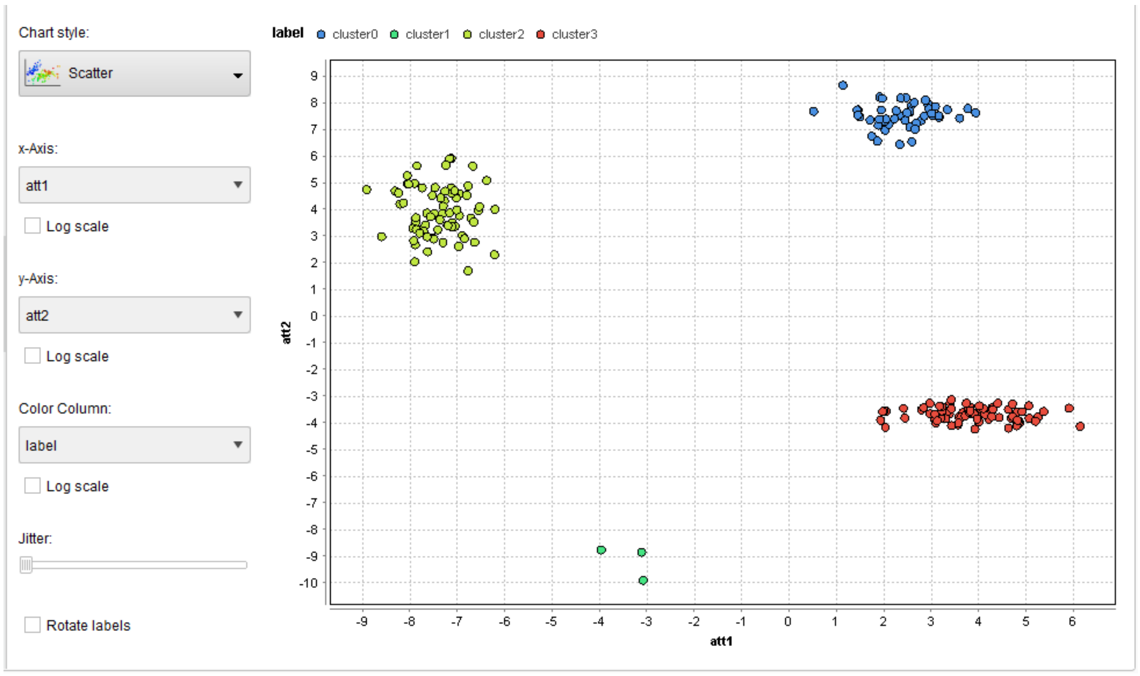


Figure 4.27. Result of outlier detection

v) Dimesionality Reduction(PCA) : This operator performs a Principal Component Analysis (PCA) using the covariance matrix. The user can specify the amount of variance to cover in the original data while retaining the best number of principal components. The user can also specify manually the number of principal components. Figure 4.28 shows the process diagram for PCA operator.

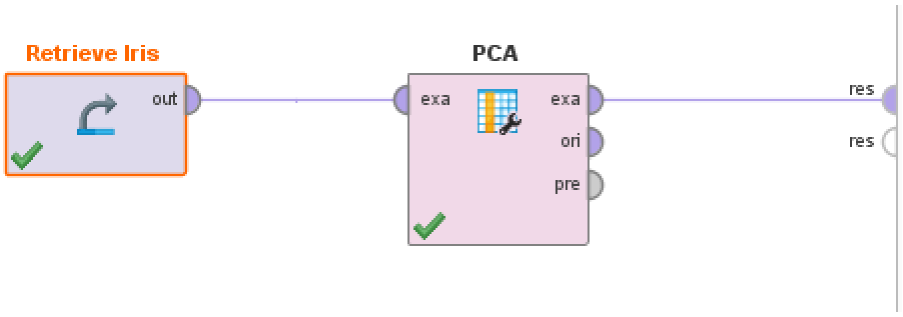


Figure 4.28. Dimensionality Reduction

Lab Exercises:

1. Explore all the other operators listed under operator panel for preprocessing of data. Draw the process diagram and list configuration parameters with sample input and output for the following.
   i. Read excel
   ii. Write Excel
   iii. Numerical to Binominal
   iv. Nominal to Binomial
   v. Select Attributes
   vi. Filter Example
   vii. Sample
   viii. Split Data
   ix. Sort
   x. Transpose
   xi. Intersect
   xii. Normalize
   xiii. Discretize by frequency
   xiv. Replace Missing Values
   xv. Remove Duplicates

2. Download the data set from UCI repository (https://archive.ics.uci.edu/ml/datasets/Car+Evaluation). The file is in csv format.
   i. Import the data to Rapid Miner with proper attribute names and attribute type.
   ii. Divide the dataset into 4 partitions based on the class label.
   iii. Considering the dataset with class label "unacc", What is the most common(maximum occurrence) value of attribute "buying".
   iv. Is there correlation between the attributes "doors" and "persons".
   v. Remove the attribute "maint".
   vi. Is it possible to apply transformation on attribute 'doors'? What can be done to apply transformation operator on this attribute?

3. Consider the dataset https://www2.stetson.edu/~jrasp/data/AMZN-KO.xls Which is daily returns for the stocks of two companies, Check whether the attributes are highly correlated.

Additional Exercise

1. Consider a data set with attributes RollNo, Score1, Score2. Perform the following operations.
   i. Select all attributes except RollNo and display the data.
   ii. How many missing values are there in each attribute?
   iii. Find statistics such as minimum and maximum in Score1 and Score2.
   iv. What is Z-transformation? Apply Z-transformation Score1 and Score2 and find the minimum and maximum value after transformation. Also find the mean and standard deviation.
   v. Check whether the distribution remains same for Score 1 and Score 2 after applying z-transformation
   vi. What is Range transformation? Apply range transformation and find minimum, maximum and average values.

LAB NO: 4 **Date:**

DATA VISUALIZATION AND MODELLING USING CLASSIFICATION

Objectives:

1. To visualize the dataset.
2. To explore association mining and clustering operators in rapid miner
3. To model the data set.

Data Visualization:

RapidMiner provides several plotters in order to visualize the properties of a data set. Each time an ExampleSet is part of the (intermediate) result, the user can select between data view, statistics view and charts view. The charts view provides several 2D and 3D charts, histogram charts, and other charts for high-dimensional data sets. Figure 5.1 shows pie chart for an example data set.

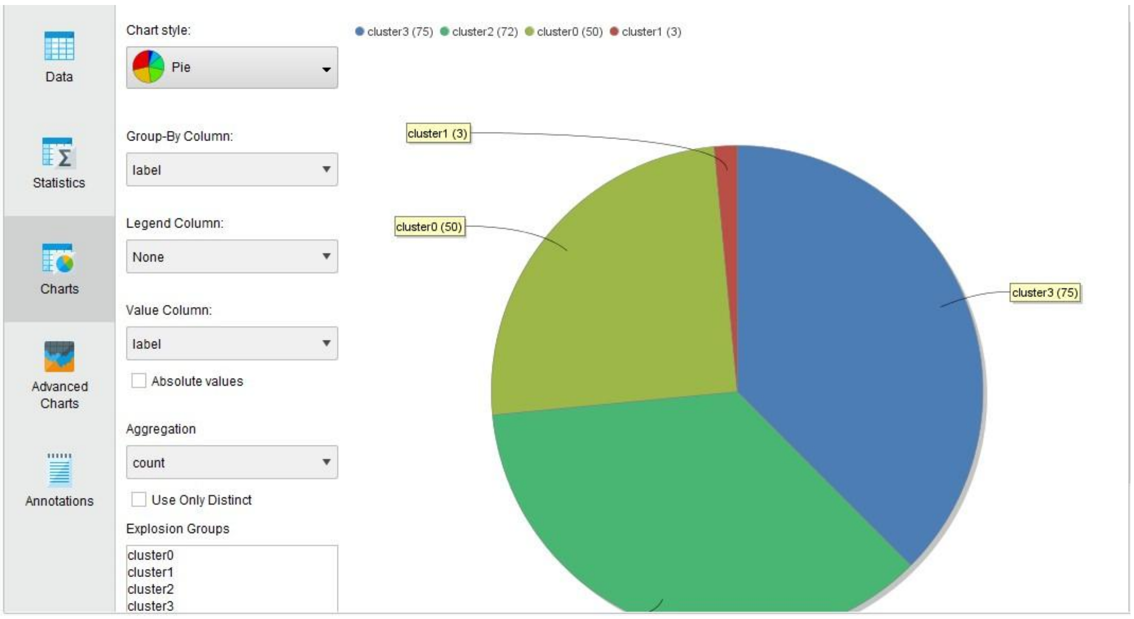


Chart style: Pie
Group-By Column: label
Legend Column: None
Value Column: label
Absolute values
Aggregation: count
Use Only Distinct
Explosion Groups: cluster0, cluster1, cluster2, cluster3

Figure 5.1. Pie chart for an example data set.

I. Modelling using Classification

1. Decision Tree Model creation: : Figure 5.2 shows the process diagram for modelling a decision tree and the Figure 5.3. shows the output obtained.

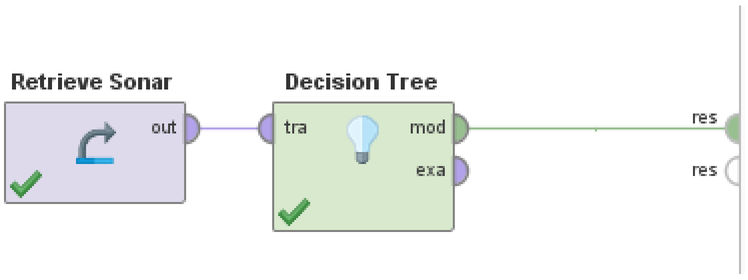


Figure 5.2. Decision tree model creation

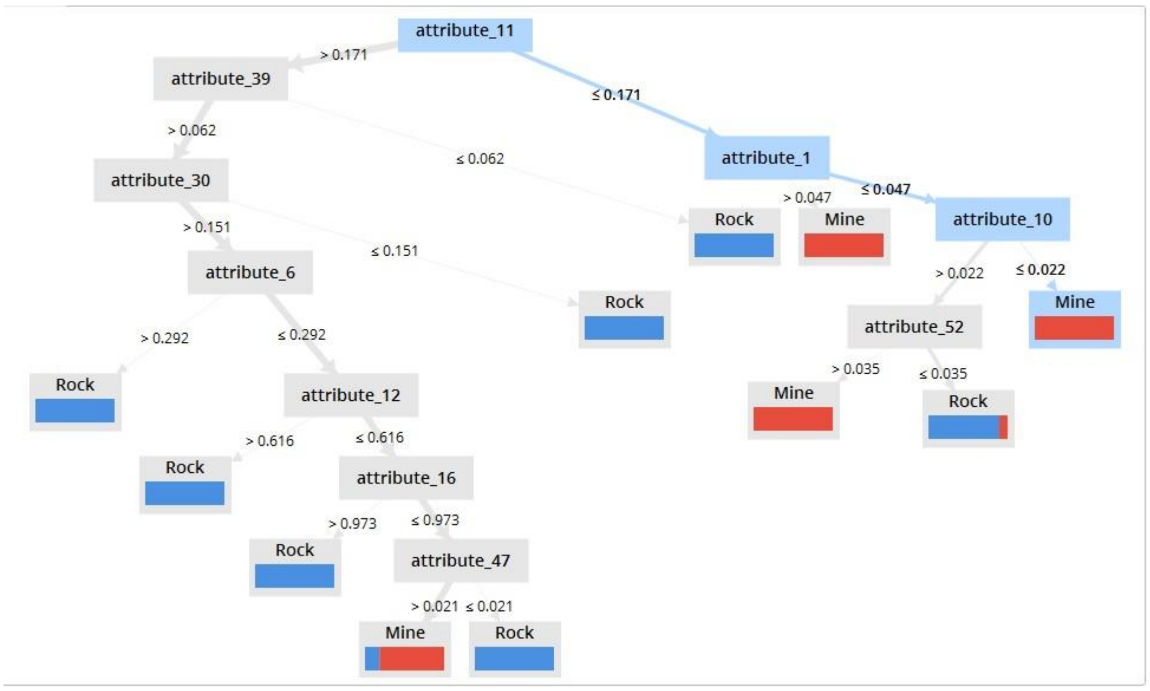


Figure 5.3. Decision Tree

1. **Criterion:** Selects the criterion on which Attributes will be selected for splitting.
2. **Pruning: P**runing the decision tree is optional which if selected confidence for pessimistic error calculation of pruning must be provided.
3. **minimal Maximal depth: gain** (optional) The gain of a node is calculated before splitting it. The node is split if its gain is greater than the minimal gain.
4. **minimal leaf size** (optional) The size of a leaf is the number of Examples in its subset. The tree is generated in such a way that every leaf has at least the *minimal leaf size* number of Examples.
5. **minimal size for split** (optional) The size of a node is the number of Examples in its subset. Only those nodes are split whose size is greater than or equal to the minimal size for split parameter.

Performance of the created model by applying on training data
Figure 5.4 shows the process diagram to create a model using Apply model operator. The performance of the model can be measured using Performance operator. Figure 5.5 shows sample output of Accuracy.

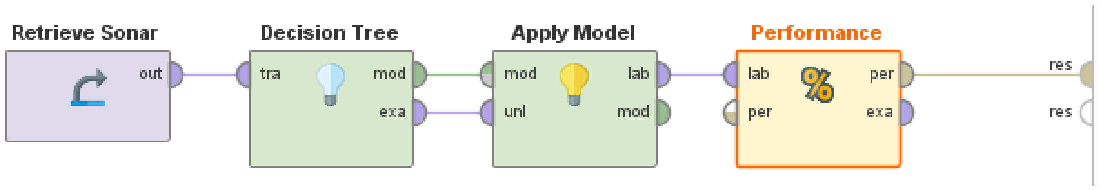
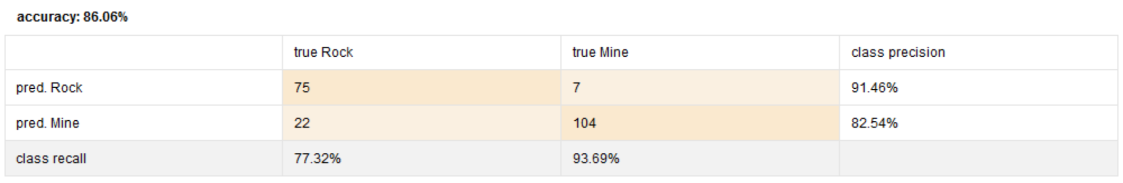


Figure 5.4. Applying model to training data

Performance on Test data

The Sonar dataset is split into two (80% for training and 20% for testing) subsets. Since the decision tree model has to be applied on training and test dataset, apply model operator is used on which Performance operator is applied as shown in the process diagram of Figure 5.6. Figure 5.7 shows the results, there would be two tabs, one for Performance on test data and other for Performance for Training Data.



accuracy: 86.06%

	true Rock	true Mine	class precision
pred. Rock	75	7	91.46%
pred. Mine	22	104	82.54%
class recall	77.32%	93.69%	

Figure 5.5. Accuracy of Training data

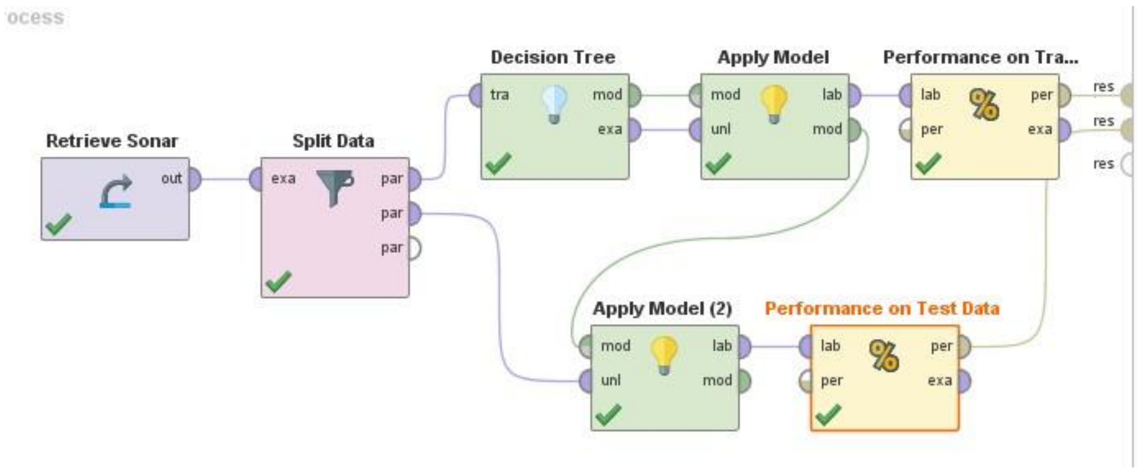
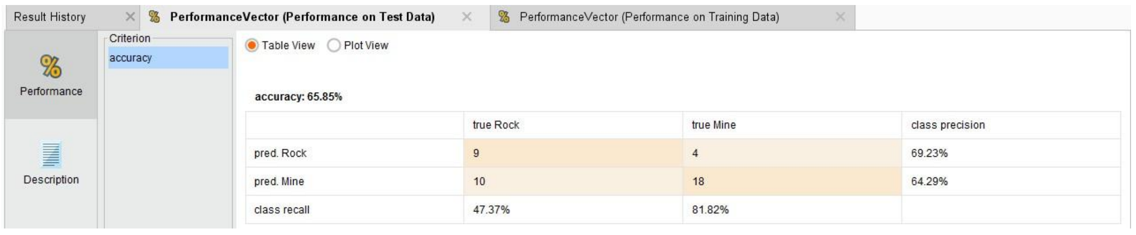


Figure 5.6. Performance on test data


Result History
PerformanceVector (Performance on Test Data)
PerformanceVector (Performance on Training Data)
Performance
Description
Criterion
accuracy
Table View
Plot View

accuracy: 65.85%

	true Rock	true Mine	class precision
pred. Rock	9	4	69.23%
pred. Mine	10	18	64.29%
class recall	47.37%	81.82%	

Figure 5.7. Accuracy on test data

Validation of the Result: Performance is obtained by applying cross validation operator as shown in process diagram of Figure 5.8. The results of validation are depicted in Figure 5.9.

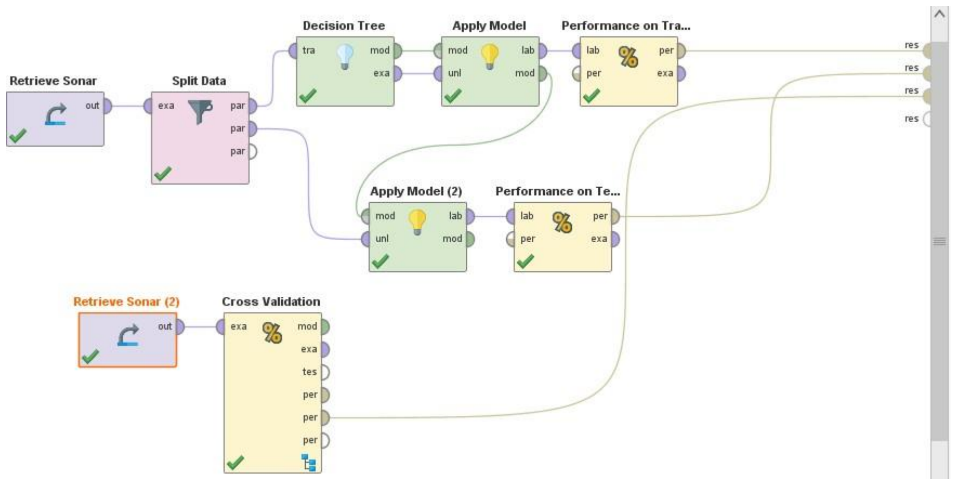
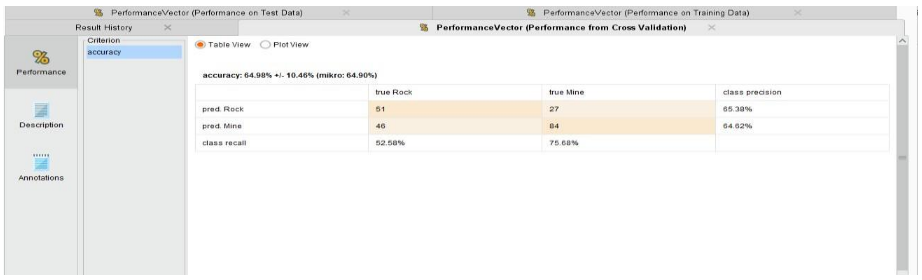


Figure 5.8. Cross Validation operator


PerformanceVector (Performance on Test Data)
PerformanceVector (Performance on Training Data)
Result History
PerformanceVector (Performance from Cross Validation)
Performance
Description
Annotations
Criterion
accuracy
Table View
Plot View

accuracy: 64.98% +/- 10.46% (mikro: 64.90%)

	true Rock	true Mine	class precision
pred. Rock	51	27	65.38%
pred. Mine	46	84	64.62%
class recall	52.58%	75.68%	

Figure 5.9. Result after cross validation

Prediction

The example for which the class has be predicted should be given in csv file as shown in process diagram of Figure 5.10.

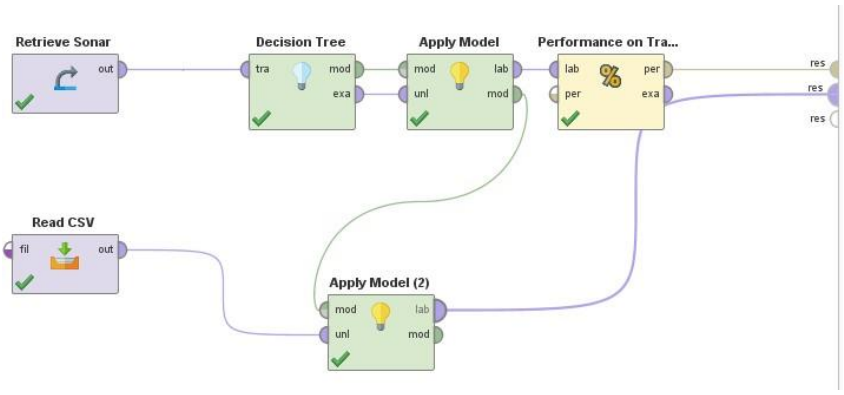


Figure 5.10. Prediction

II. Association Mining

FP-Growth:

This operator efficiently calculates all frequent itemsets from the given ExampleSet using the FP-tree data structure. It is compulsory that all attributes of the input ExampleSet should be binominal. In simple words, frequent itemsets are groups of items that often appear together in the data. It is important to know the basics of market-basket analysis
for understanding frequent itemsets.

This operator has two basic working modes:

- Finding at least the specified number of itemsets with highest support without taking the 'min support' into account. This mode is available when the find min number of itemsets parameter is set to true. Then this operator finds the number of itemsets specified in the min number of itemsets parameter. The min support parameter is ignored in this case.
- Finding all itemsets with a support larger than the specified minimum support. The minimum support is specified through the min support parameter. This mode is available when the find min number of itemsets parameter is set to false.

Create Association Rules

This operator generates a set of association rules from the given set of frequent itemsets. Association rules are created by analyzing data for frequent if/then patterns and using the criteria support and confidence to identify the most important relationships. Support is an indication of how frequently the items appear in the database. Confidence indicates the number of times the if/then statements have been found to be true. The frequent if/then patterns are mined using the operators like the FP-Growth operator as shown in process diagram of Figure 5.11. The Create Association Rules operator takes these frequent itemsets and generates association rules.

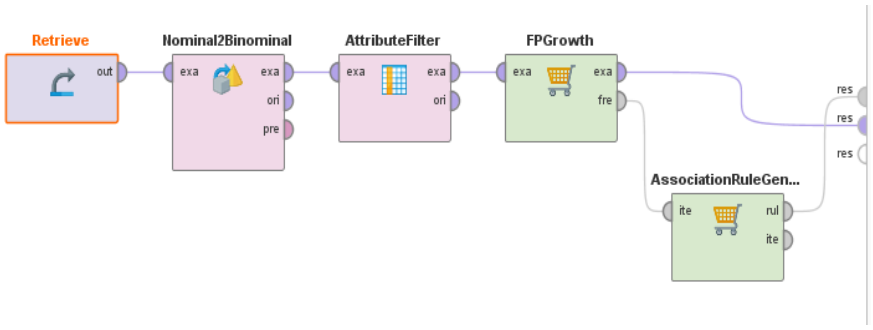
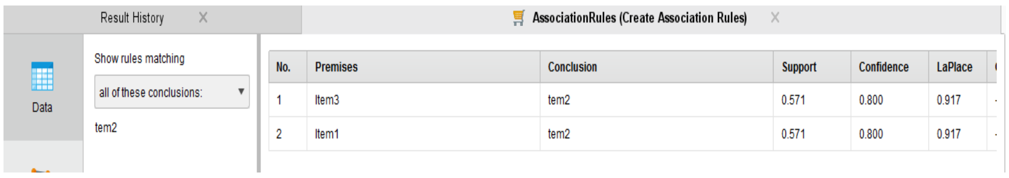


Figure 5.11. Association rule generation

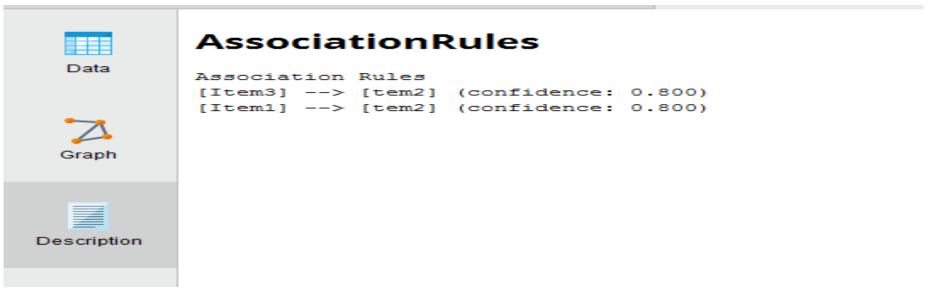
Apply Association rules:

This operator creates a new confidence attribute for each item occurring in at least one conclusion of an association rule. Then it checks for each example and for each rule, if the example fulfills the premise of the rule, which it does, if it covers all items in the premise. An example covers an item, if the attribute representing the item contains the positive value. If the check is positive, a confidence value for each item in the conclusion is derived. Which value is used, depends on the selected confidence aggregation method. There are two types: The binary choice will set a 1, for any item contained inside a fulfilled rule's conclusion. This is independent of how confident the rule was. Any aggregation choice will select the maximum of the previous and the new value of the selected confidence function. Figure 5.12 and 5.13 shows the output of association rules in data and description tab respectively.


Result History
AssociationRules (Create Association Rules)
Data
Show rules matching
all of these conclusions:
tem2

No.	Premises	Conclusion	Support	Confidence	LaPlace
1	Item3	tem2	0.571	0.800	0.917
2	Item1	tem2	0.571	0.800	0.917

Figure 5.12. Frequent items in data tab

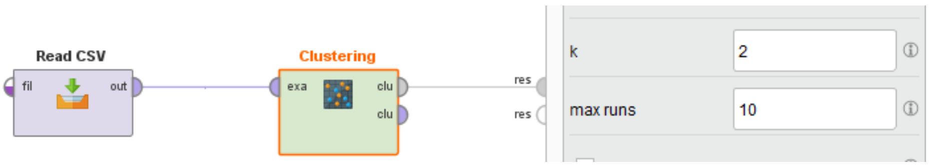


```
AssociationRules
Association Rules
[Item3] --> [tem2] (confidence: 0.800)
[Item1] --> [tem2] (confidence: 0.800)
```

Figure 5.13. Association Rules in Description tab

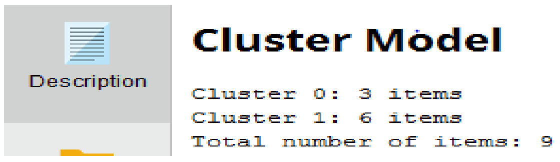
Clustering

K-Means: This operator performs clustering using the k-means algorithm as shown in process diagram of Figure 5.14. Clustering is concerned with grouping objects together that are similar to each other and dissimilar to the objects belonging to other clusters. Clustering is a technique for extracting information from unlabeled data. Figure 5.15 gives the overview of the cluster by specifying the number of items in each cluster. K-means clustering is an exclusive clustering algorithm i.e. each object is assigned to precisely oneof the set of clusters as shown in Figure 5.16.



k: 2
max runs: 10

Figure 5.14. K-Means operator applied on example data set



Cluster Model

```
Cluster 0: 3 items
Cluster 1: 6 items
Total number of items: 9
```

Figure 5.15. Resulting cluster details

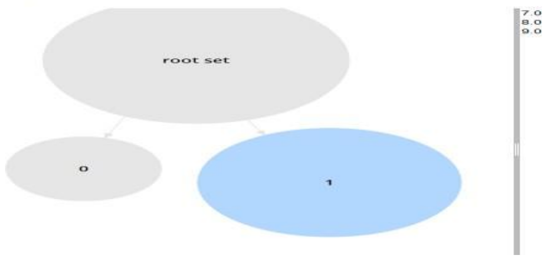


Figure 5.16. Clusters visualized on click of each circle data points

Lab Exercise

1. Apply the following visualizations on any two datasets and write the sample outputs and benefits of Scatter plot, Quartile plot, Deviation, Series, and Distribution plots
2. Retrieve iris data set from the Samples repository in rapid miner. Use different plots to visualize and explore the data set. Which attribute and value ranges(approximate) determine the type of iris flower?
3. Build a process to classify the iris dataset.
   i. Discretize all attributes of the iris data set by frequency into three bins.
   ii. Use split validation operator to generate training and test data set.
4. As inner operator of split validation use the ID3 to learn a decision tree and Performance (classification) operator to evaluate the accuracy of the model.
5. Use Decision Tree and k-NN operators for classifying iris dataset. Which gives best accuracy for the given dataset?

Additional Exercise

1. Apply Naïve Bayes operator for classification. Validate and predict the result as outlined above in manual.
2. Generate association rules for frequent itemsets obtained using FP-Growth operator.
3. Cluster the iris dataset using different algorithms and different parameter settings.
   a. Which algorithm and parameter setting reproduces the original division into three different species?
   b. If the data is normalized before applying the clustering will there be any change in cluster formed?

LAB NO. 5 **Date:**

MINI PROJECT – SYNOPSIS SUBMISSION

Objective

1. **To finalize the title of miniproject**

Students are required to submit the synopsis of the mini project. They are encouraged to select the topic based on the Scopus/WoS indexed paper in data mining area. They can also develop any application based on the data mining algorithms.

SAMPLE PROJECT SYNOPSIS
TITLE

Project Description:
Objectives:
Scope:
Software Requirements:

Submitted by

Name	Registration number	Roll Number	Semester & Branch	Section

Specifications to be followed:

1. Font size: 12 for regular text, font size: 14 for side headings and font size: 16 for title
2. Font Type: Times New Roman
3. Text alignment : Justified(Both Sides)

LAB NO. 6 **Date:**

APRIORI ALGORITHM

Objectives

1. To implement Apriori algorithm
2. To generate association rules based on frequent item sets

Apriori Algorithm

Agarwal and Srikant proposed the Apriori algorithm in 1994. Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time (a step known as *candidate generation)*, and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found. Apriori uses breadth-first search to count candidate item sets efficiently. This algorithm uses downward closure property, which states that, "Any subset of a frequent itemset must be frequent". It is called as apriori because it uses prior knowledge of frequent item set properties.

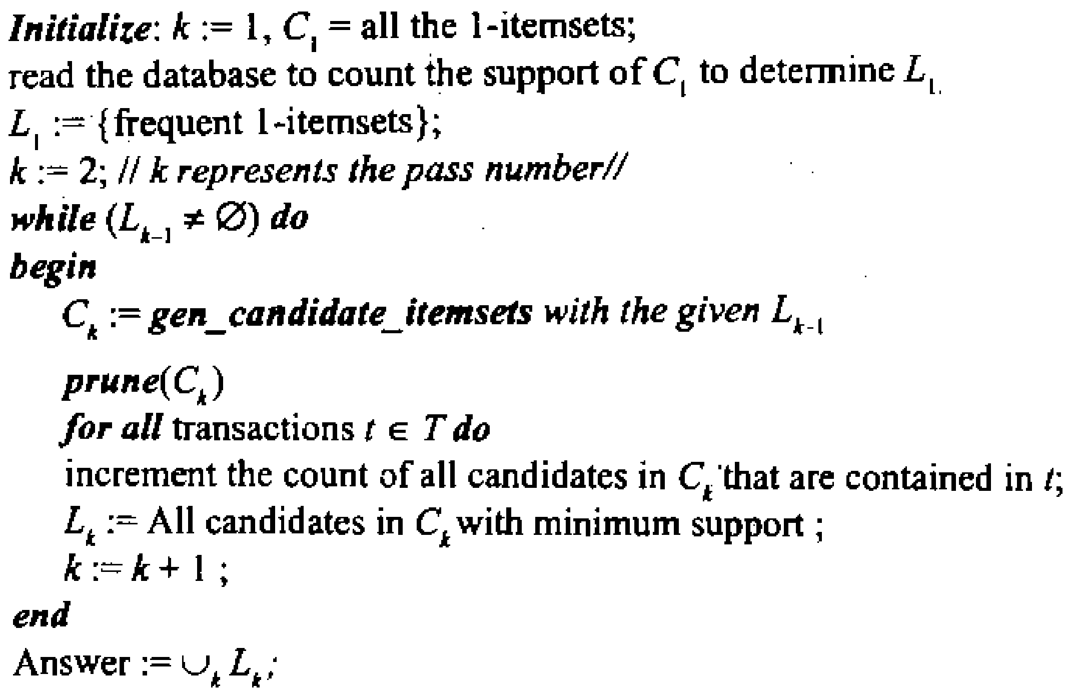
It uses level-wise search, where k-itemsets (an itemset containing k number of items is called as a k-itemset) are used to explore (k+1) itemsets to mine frequent itemsets from transactional database. First, the set of frequent 1-temset (L_1) is found. L_1 is used to find L_2, which is used to find L_3 and so on, until no more frequent k-itemsets can be found.

The candidate-gen function takes L_{k-1} and returns a superset (called the candidates) of the set of all frequent k-itemsets. It has two steps

- *join* step: Generate all possible candidate itemsets C_k of length k
- *prune* step: Remove those candidates in C_k that cannot be frequent.

Figure 7.1 shows the pseudocode of the algorithm.

```
prune($C_k$)

for all $c \in C_k$
for all $(k-1)$-subsets $d$ of $c$ do
      if $d \notin L_{k-1}$
      then $C_k = C_k \setminus \{c\}$


gen_candidate_itemsets with the given $L_{k-1}$ as follows:

$C_k = \varnothing$
for all itemsets $l_1 \in L_{k-1}$ do
for all itemsets $l_2 \in L_{k-1}$ do
      if $l_1[1] = l_2[1] \wedge l_1[2] = l_2[2] \wedge \ldots \wedge l_1[k-1] < l_2[k-1]$
      then $c = l_1[1], l_1[2] \ldots l_1[k-1], l_2[k-1]$
      $C_k = C_k \cup \{c\}$
```



```
Initialize: $k := 1$, $C_1$ = all the 1-itemsets;
read the database to count the support of $C_1$ to determine $L_1$
$L_1 := \{$frequent 1-itemsets$\}$;
$k := 2$; // k represents the pass number//
while ($L_{k-1} \neq \emptyset$) do
begin
    $C_k :=$ gen_candidate_itemsets with the given $L_{k-1}$
    prune($C_k$)
    for all transactions $t \in T$ do
    increment the count of all candidates in $C_k$ that are contained in $t$;
    $L_k :=$ All candidates in $C_k$ with minimum support ;
    $k := k + 1$ ;
end
Answer $:= \cup_k L_k$;
```

Figure 7.1 Apriori algorithm

Example

Consider a database consisting of 9 transactions as given in Table 7.1. Find the frequent itemsets using apriori algorithm. Assume the minimum support count as 2.

- Scan the database to get the support of each candidate item
  C_1 = { {A} – 6, {B} – 7, {C} – 6, {D} – 2, {E} – 2 }
- Determine L_1 from C_1
  L_1 = { {A}, {B}, {C}, {D}, {E} }

Table 7.1 Database

Transaction ID	List of Items
1	A, B, E
2	B, D
3	B, C
4	A, B, D
5	A, C
6	B, C
7	A, C
8	A, B, C, E
9	A, B, C

- Generate C_2 from L_1 using apriori join step
  C_2 = { {A,B}, {A,C}, {A,D}, {A,E}, {B,C}, {B,D}, {B,E}, {C,D}, {C,E}, {D,E}}
- Scan the database to get the support of each candidate item of C_2
  C_2 = { {A,B} – 4, {A,C} – 4, {A,D} – 1, {A,E} – 2, {B,C} – 4, {B,D} – 2, {B,E} – 2, {C,D} – 0, {C,E} – 1, {D,E} – 0 }

- Determine L_2 from C_2
  L_2 = { {A,B}, {A,C}, {A,E}, {B,C}, {B,D}, {B,E} }
- Generate C_3 from L_2 using apriori join step
  C_3 = { {A,B,C}, {A,B,E}, {A,C,E},{ B,C,D}, {B,C,E}, {B,D,E} }
- Prune C_3 using apriori prune step
  C_3 = { {A,B,C}, {A,B,E} }
- Scan the database to get the support of each candidate item of C3
  C_3 = { {A,B,C} – 2, {A,B,E} – 2 }
- Determine L_3 from C_3
  L_3 = { {A,B,C}, {A,B,E} }
- Generate C_4 from L_3 using apriori join step
  C_4 = { {A,B,C, E} }
- Prune C4 using apriori prune step
  C_4 = { }
  L_4 = { }

The algorithm stops, as L_4 is empty.
Answer: Frequent itemsets: {{A}, {B}, {C}, {D}, {E}, {A, B}, {A, C}, {A, E}, {B, C}, {B,D}, {B,E}, {A,B,C}, {A, B, E}}

Association Rules

An association rule is an implication of the form X ⇒ Y, where X & Y are transactions with set of items from a transactional database 'D'.

- The rule X ⇒ Y holds in 'D' with confidence c if c% of transactions in D that contain X also contain Y
- The rule X ⇒ Y has support s in D if s% of transactions in D contain X U Y

Find all rules that have support and confidence greater than user-specified minimum support and minimum confidence. The steps to generate association rules is as given below:

- For each frequent itemset *l*, generate all nonempty subsets of *l*.
- For every nonempty subset *s* of *l*, output the rule "*s*⇒ *l-s* " if (*support count*(*l*) / *support count*(*s*)) ≥ *min conf*, where *min conf* is the minimum confidence threshold.
- As the rules are generated from the frequent itemsets, each one automatically satisfies minimum support.

Example: Consider one of the frequent itemset {A, B, C} for the database given in Table 7.1 to generate association rule by considering the minimum confidence threshold as 75%

- *l* ={A, B, E} , Subsets of *l* ={{A, B},{A, E},{B, E},{A},{B},{E}}
- s = {A, B}, {A, B} ⇒ {E} , conf({A, B}⇒ {E})=2/4=50%
- s = {A, E}, {A, E} ⇒ {B} , conf({A, C} ⇒ {B})=2/2=100%
- s = {B, E}, {B, E} ⇒ {A} , conf({B, E} ⇒ {A})=2/2=100%
- s = {A}, {A}⇒ {B, E} , conf({A} ⇒ {B, E})=2/6=33%

- s = {B}, {B} ⇒ {A,E} , conf({B} ⇒ {A,E})=2/7=29%
- s = {E}, {E} ⇒ {A,B} , conf({E} ⇒{A,B})=2/2=100%

Association rules satisfying the minimum support and threshold are as follows:
{A, E} ⇒ {B}, {B, E} ⇒ {A}, {E} ⇒ {A, B}

Lab Exercise

1. Find the frequent itemsets by using the Apriori algorithm for a given transactional database and determine the association rules by considering suitable minimum support and confidence values

Additional Exercise

1. Implement partition based Apriori algorithm for a database stored as a file and find the association rules.

LAB NO: 7 **Date:**

K-MEANS ALGORITHM

Objective

1. To implement K-means algorithm for clustering

k-means algorithm

Clustering is a process of partitioning various objects into groups called as clusters, with the aim of having high intra-cluster similarity and low inter cluster similarity. It is an example of unsupervised learning. Clustering is a form of learning by observation, rather than learning by examples.

K-means is a well-known and commonly used partitioning method. The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intracluster similarity is high but the intercluster similarity is low. Cluster similarity is measured concerning the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or center of gravity.

The k-means algorithm proceeds as follows:
First, it randomly selects k of the objects, each of which initially represents a cluster mean or center. For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster. This process iterates until the criterion function converges. Typically, the square-error criterion is used, as given in equation 8.1.

$$E = \sum_{i=1}^{k} \sum_{p \in Ci} |p - mi|^2 \qquad (8.1)$$

where, E is the sum of the square error for all objects in the data set; p is the point in space representing a given object; and mi is the mean of cluster Ci (both p and mi are multidimensional). In other words, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This criterion tries to make the resulting k clusters as compact and as separate as possible. The k-means procedure is summarized in Figure 8.1
The algorithm attempts to determine k partitions that minimize the square-error function. It works well when the clusters are compact clouds that are well separated from one another. The method is relatively scalable and efficient in processing large data sets because the computational complexity of the algorithm is O(nkt), where n is the total number of objects, k is the number of clusters, and t is the number of iterations. Normally, k << n and t << n. The method often terminates at a local optimum.

The K-means method, however, can be applied only when the mean of a cluster is defined. The necessity for users to specify k, the number of clusters, in advance can be seen as a disadvantage. The K-means method is not suitable for discovering clusters with nonconvex shapes or clusters of very different size. It is sensitive to noise and outlier data points because a small number of such data can substantially influence the mean value.

Suppose that there is a set of objects located in space as depicted in the rectangle shown in Figure 8.2(a). Let k = 3; that is, the user would like the objects to be partitioned into three clusters. According to the algorithm in Figure 8.1, we arbitrarily choose three objects as the three initial cluster centers, where cluster centers are marked by a "+". Each object is distributed to a cluster based on the cluster center to which it is the nearest. Such a distribution forms silhouettes encircled by dotted curves, as shown in Figure 8.2(a). Next, the cluster centers are updated. That is, the mean value of each cluster is recalculated based on the current objects in the cluster. Using the new cluster centers, the objects are redistributed to the clusters based on which cluster center is the nearest. Such a redistribution forms new silhouettes encircled by dashed curves, as shown in Figure 8.2(b).

```
Algorithm: k-means. The k-means algorithm for partitioning, where each cluster's
  center is represented by the mean value of the objects in the cluster.

Input:
  ■ k: the number of clusters,
  ■ D: a data set containing n objects.

Output: A set of k clusters.

Method:
  (1) arbitrarily choose k objects from D as the initial cluster centers;
  (2) repeat
  (3)     (re)assign each object to the cluster to which the object is the most similar,
              based on the mean value of the objects in the cluster;
  (4)     update the cluster means, i.e., calculate the mean value of the objects for
              each cluster;
  (5) until no change;
```

Figure 8.1 K-means algorithm

.This process iterates, leading to Figure 8.2(c). The process of iteratively reassigning objects to clusters to improve the partitioning is referred to as iterative relocation. Eventually, no redistribution of the objects in any cluster occurs, and so the process terminates. The resulting clusters are returned by the clustering process.

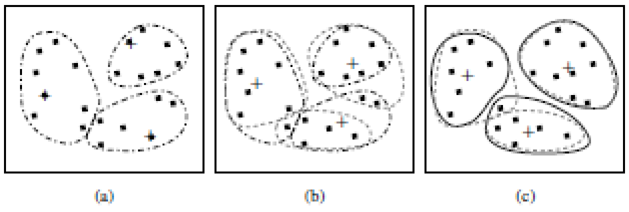


Figure 8.2 Clustering of a set of objects based on the k-means method

Lab Exercise

1. Implement K-means algorithm for a given dataset using Euclidean distance as a similarity measure.

Additional Exercise

1. Use Manhattan as a similarity measure to cluster the given dataset using K-means algorithm

Lab 8 **Date:**

DECISION TREE ID3 ALGORITHM FOR CLASSIFICATION

Objectives:

1. To understand the working of decision tree for classification
2. Implement ID3 algorithm to construct the decision tree.

Introduction:

ID3 builds a decision tree from a fixed set of examples. The resulting tree is used to classify future samples. The example has several attributes and belongs to a class (like yes or no). The leaf nodes of the decision tree contain the class name whereas a non-leaf node is a decision node. The decision node is an attribute test with each branch (to another decision tree) being a possible value of the attribute. ID3 uses information gain to help it decide which attribute goes into a decision node. The advantage of learning a decision tree is that a program, rather than a knowledge engineer, elicits knowledge from an expert.

How does ID3 decide which attribute is the best? A statistical property, called **information gain**, is used. Gain measures how well a given attribute separates training examples into targeted classes. The one with the highest information (information being the most useful for classification) is selected. In order to define gain, we first borrow an idea from information theory called entropy. Entropy measures the amount of information in an attribute.

Given a collection S of c outcomes, Entropy is calculated as given in equation 9.1

$$\text{Entropy}(S) = S\ -p(I)\log_2 p(I) \qquad (9.1)$$

where, p(I) is the proportion of S belonging to class I. S is over c. Log_2 is log base2. Note that S is not an attribute but the entire sample set. Gain(S, A) is information gain of example set S on attribute A is defined as given in equation 9.2.

$$\text{Gain}(S, A) = \text{Entropy}(S) - S\ ((|S_v| / |S|) * \text{Entropy}(S_v)) \quad (9.2)$$

where,

S is each value v of all possible values of attribute A

S_v = subset of S for which attribute A has value v

$|S_v|$ = number of elements in S_v

$|S|$ = number of elements in S

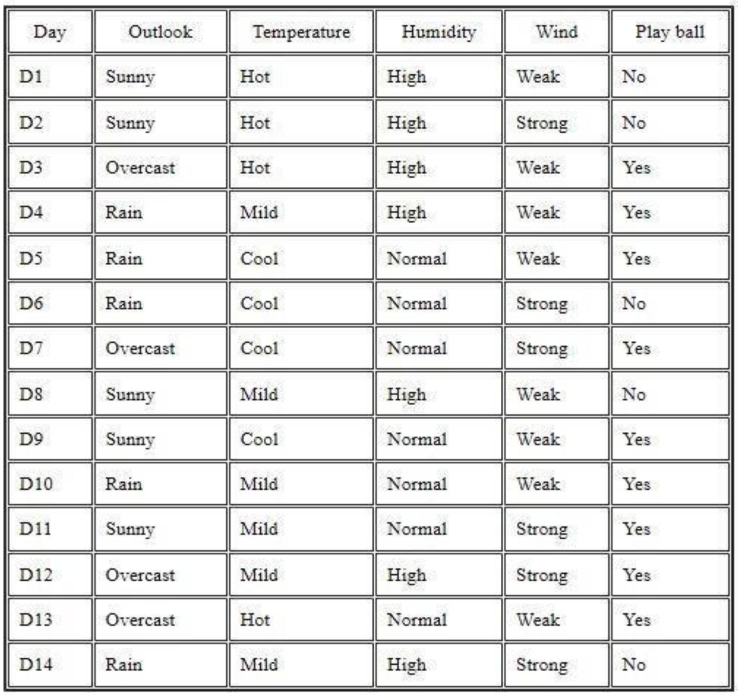
ID3 Algorithm:

```
ID3 (Examples, Target_Attribute, Attributes)
Create a root node for the tree
If all examples are positive, Return the single-node tree Root, with label = +.
If all examples are negative, Return the single-node tree Root, with label = -.
If number of predicting attributes is empty, then Return the single node tree Root, with label = most common value of the target attribute in the examples.
Otherwise Begin
        A ← The Attribute that best classifies examples.
        Decision Tree attribute for Root = A.
```

```
    For each possible value, $v_i$, of A,
        Add a new tree branch below Root, corresponding to the test A = $v_i$.
        Let Examples($v_i$) be the subset of examples that have the value $v_i$ for A
        If Examples($v_i$) is empty
            Then below this new branch add a leaf node with label = most common target value in the examples
        Else below this new branch add the subtree ID3 (Examples($v_i$), Target_Attribute, Attributes – {A})
End

Return Root
```

Consider the following example:

Table 9.1: Play ball dataset



Day	Outlook	Temperature	Humidity	Wind	Play ball
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Suppose Table 9.1 is a set of 14 examples in which one of the attributes is wind speed. The values of Wind can be *Weak* or *Strong*. The classification of these 14 examples are 9 YES and 5 NO. For attribute Wind, suppose there are 8 occurrences of Wind = Weak and 6 occurrences of Wind = Strong. For Wind = Weak, 6 of the examples are YES and 2 are NO. For Wind = Strong, 3 are YES and 3 are NO. Therefore,

Gain(S,Wind) = Entropy(S)-(8/14)*Entropy(S_{weak})-(6/14)*Entropy(S_{strong})
= 0.940 - (8/14)*0.811 - (6/14)*1.00
= 0.048

Entropy(S_{weak}) = - (6/8)*log2(6/8) - (2/8)*log2(2/8) = 0.811

Entropy(S_{strong}) = - (3/6)*log2(3/6) - (3/6)*log2(3/6) = 1.00

For each attribute, the gain is calculated and the highest gain is used in the decision node. This process goes on until all data is classified perfectly or we run out of attributes.

Lab Exercises

1. Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use data set given in table 1 for building the decision tree.
2. Classify new samples using the rules obtained from the decision tree in exercise1.

Additional Exercises

1. Construct the decision tree for the following dataset of mushroom

   https://archive.ics.uci.edu/ml/machine-learning-databases/mushroom/agaricus-lepiota.data

Lab 9 **Date:**

NAIVE BAYES CLASSIFIER

Objectives:

1. To understand the Naïve Bayesian classifier.
2. To build Naïve Bayes Gaussian classifier to classify data.
3. To build Multinomial Naïve Bayes Gaussian classifier to classify text.

Naïve Bayesian Gaussian Classifier

Introduction:

Naive Bayes is a supervised learning algorithm used for classification tasks. Hence, it is called as Naive Bayes Classifier. As other supervised learning algorithms, Naive Bayes uses features to make a prediction on a target variable. The key difference is that Naive Bayes assumes that features are independent of each other and there is no correlation between features

Bayes comes from the famous Bayes' Theorem of Thomas Bayes as shown in equation 10.1. To get a comprehensive understanding of Bayes' Theorem, we should talk about probability and conditional probability first.

$$P(A|B)=\frac{P(A).P(B)}{P(B)} \qquad (10.1)$$

Naive Bayes classifier calculates the probability of a class given a set of feature values (i.e. p(yi | x1, x2 , … , xn)). Input this into Bayes' theorem as given in equation 10.2.

$$P(yi|x1,x2,\dots.xn)=\frac{P(x1, x2, \dots . xn|yi).P(yi)}{P(x1,x2,\dots.xn)} \qquad (10.2)$$

p(x1, x2 , … , xn | yi) means the probability of a specific combination of features given a class label. To be able to calculate this, we need extremely large datasets to have an estimate on the probability distributionfor all different combinations of feature values. To overcome this issue, **naive bayes algorithm assumes that all features are independent of each other.** Furthermore, denominator (p(x1,x2, … , xn)) can be removed to simplify the equation because it only normalizes the value of conditional probability of a classgiven an observation (p(yi | x1,x2,… … , xn)).The probability of a class (p(yi)) is very simple to calculateas shown in equation 10.3.

$$P(yi)=\frac{number\ of\ observations\ with\ class\ yi}{number\ of\ all\ observations} \qquad (10.3)$$

Under the assumption of features being independent, **p(x1, x2 ,.. , xn | yi)** can be written as given in equation 10.4.

$$x1,x2,\dots.xn|yi)=P(x1|yi).P(x2|yi)\dots\dots.P(xn|yi) \qquad (10.4)$$

The conditional probability for a single feature given the class label (i.e. p(x1 | yi)) can be more easily estimated from the data. The algorithm needs to store probability distributions of features for each class independently. The type of distributions depends on the characteristics of features:

1. For binary features (Y/N, True/False, 0/1): Bernoulli distribution
2. For discrete features (i.e. word counts): Multinomial distribution
3. For continuous features: Gaussian (Normal) distribution

Multinomial Naive Bayes Classifier to Classify Text

Introduction:

Multinomial naive Bayes works similar to Gaussian naive Bayes, however the features are assumed to be multinomial distributed. In practice, this means that this classifier is commonly used when we have discrete data (e.g. movie ratings ranging 1 and 5).

For sentiment analysis, a Naive Bayes classifier is one of the easiest and most effective ways to hit the ground running for sentiment analysis.

Deriving the prior probability of a class is rather trivial, as it is simply the sum of all words in *doc* that are assigned to *c* divided by the number of words in *doc* as shown in equation 10.5.

$$\hat{P}(c) = \frac{N_c}{N_{doc}} \qquad (10.5)$$

How do we learn all of the probabilities of that make up each feature? The solution is again rather simple: for a given word *w* in words *W* from *d* we count how many of *w* belong in class *c*. We then divide this by all the words in *d* that belong to *c*. This gives us a probability for a word *w* given *c* as in equation 10.6.

$$\hat{P}(w_i|c) = \frac{count(w_i,c)+1}{\sum_{w\in V}(count(w,c)+1)} = \frac{count(w_i,c)+1}{(\sum_{w\in V} count(w,c))+|V|} \qquad (10.6)$$

Computing Error Rate, Accuracy, Precision and Recall from confusion matrix:

A confusion matrix is a technique for summarizing the performance of a classification algorithm. The number of correct and incorrect predictions are summarized with count values and broken down by each class. These numbers are then organized into a table, or a matrix as shown in Table 10.1.

Table 10.1: Confusion matrix for 2-class problem

		PREDICTED	
		Positive	**negative**
ACTUAL	**Positive**	True Positive	False negative
	negative	False Positive	True Negative

From confusion matrix one can get the following measures.

1. Accuracy (ACC) is calculated as the number of all correct predictions divided by the total number of the dataset as in equation 10.7.

Accuracy=(TP+TN)/(TP+TN+FP+FN)   (10.7)

2. Error rate (ERR) is calculated as the number of all incorrect predictions divided by the total number of the dataset as in equation 10.8.

$$\text{Error rate} = (FP+FN)/(TP+TN+FP+FN) \qquad (10.8)$$

3. Precision evaluates the fraction of correct classified instances among the ones classified as positive as in equation 10.9.

$$\text{Precision} = TP/(TP+FP) \qquad (10.9)$$

4. Recall is a metric that quantifies the number of correct positive predictions made out of all positive predictions that could have been made as in equation 10.10.

$$\text{Recall} = TP/(TP+FN) \qquad (10.10)$$

sklearn to fetch measures

sklearn can be used to fetch these measures using the following codes:

from sklearn import metrics

metrics.confusion_matrix

metrics.classification_report(predicted, expected)

Lab Exercises

1. Write a program to implement the naïve Bayesian classifier(Gaussian) for *Pima indians diabetes* training data set. Compute the accuracy of the classifier, considering few test data sets.
2. Use the naïve Bayesian(Multinomial) Classifier model to perform text classification task on 20newsgroups dataset.
3. Calculate the accuracy, precision, and recall for your data set on the confusion matrix obtained.

Additional Exercises

1. Write a program to implement the naïve Bayesian classifier(Gaussian) for scikit-learn wine training data set. Compute the accuracy of the classifier, considering few test data sets.
2. Use the naïve Bayesian(Multinomial) Classifier model to perform text classification task on Reuters News Dataset.

LAB NO. 10 **Date:**

MINI PROJECT – IMPLEMENTATION

Objective

1. **To implement the miniproject**

Students are required to implement the mini project.

LAB NO. 11 **Date:**

MINI PROJECT – PROGRESS

Objective

1. **To discuss the progress of miniproject implementation**

Students are required to show the progress of mini project implementation and discuss with faculty

LAB NO. 12 **Date:**

MINI PROJECT – PROGRESS

Objective

1. **To discuss the progress of miniproject result and analysis**

Students are required to show the progress of mini project result and analysis discuss with faculty

REFERENCES

1. Jiawei Han and Micheline Kamber, “Data Mining- Concepts and Techniques”, 3rd Edition, Morgan Kaufmann Publishers, 2011.

2. Arun K. Pujari, ”Data Mining Techniques”, University press, 2006.

3. G.K. Gupta,”Introduction to data mining with Case Studies”, Easter Economy edition, Prentice Hall of India,2006.

4. Pang-Ning Tan,Michael Steinbach and Vipin Kumar,”Introduction to Data Mining” Pearson Education,2007.

5. Rapid miner Documentation.