UNC CHARLOTTE
College of Computing and Informatics

Department of Computer Science

University of North Carolina Charlotte

North Carolina, USA

KDD FINAL PROJECT
Analysis on Movies Dataset

Instructor: Dr. ZByszek Ras

Project Team: (Group-9)

Nishitha Dinesh - 801308890

Sheshi Rekha Guntuka - 801312506

Rithika Bejjanki - 801311702

Chiranjeevi Rao Surineni - 801273485

Project Description and Requirements

The main goal of this project is to conduct action rule mining for the given movie rating Index from data sets with new extended features (four features) added to the original data set, as well as further analysis to determine what changes to the classification features are required to lower the index.

Extract a subset of 9,000 movies from the Movies Database using the "links_small.csv" file and the column C. Calculate the average ratings of all 9,000 movies based on the data from the "ratings_small" file, which has 10 possible values ranging from 0.5 to 5.

Add four new classification attributes to the decision table and explain why you believe they are correlated with customer ratings of the movies. Build classifiers using Orange or Weka from the decision table, both with and without the four additional attributes, and compare their performance using F-score.

It is expected that the classifier built from the decision table with the additional attributes will have a higher F-score. Finally, use Lisp-Miner or python to find several action rules that suggest how movie ratings could be improved.

Average Ratings of Movie

The average ratings in the Movies Dataset from Kaggle refer to the numerical value given by users to rate movies on a scale of 0.5 to 5. This value represents the user's overall opinion of the movie's quality, with higher ratings indicating a more positive opinion. The dataset includes ratings from a large number of users and covers a wide range of movies, making it a valuable resource for analyzing and understanding movie ratings.

The rating of each movie depends on its range from:

0.5-1

1-1.5

1.5-2

2-2.5

2.5-3

3.3.5

4-4.5

4.5-5

Features of the Dataset

Release_Date: Date when the movie was released.

Title: Name of the movie.

Overview: Brief summary of the movie.

Popularity: It is a very important metric computed by TMDB developers based on the number of views per day, votes per day, number of users marked it as "favorite" and "watchlist" for the data, release date and more other metrics.

Vote_Count: Total votes received from the viewers.

Vote_Average: Average rating based on vote count and the number of viewers out of 10.

Original_Language: Original language of the movies. Dubbed version is not considered to be original language.

Genre: Categories the movie it can be classified as.

Poster_Url: Url of the movie poster.

Production companies: Movies that are produced by few companies.

Status: Movie is released or not.

Tagline: Movie tagline.

Production countries: Movies that are produced by countries.

Extended Features

Release year

The "release year" column in the Movies Dataset from Kaggle indicates the year in which a movie was released. This column contains numerical values that correspond to the year in which each movie was originally released in theaters. This information can be useful in analyzing trends or patterns over time, such as changes in popular movie genres, technological advances in film production, or changes in societal attitudes or cultural norms. It can also be useful for creating recommendation systems based on a user's preferences for movies released during a particular time period.

Awards won

The "awards" column in the Movies Dataset from Kaggle provides information on the number of awards won by a movie. This column contains numerical values that represent the total number of awards that the movie has won, including any major or minor awards, such as Academy Awards, Golden Globes, or film festival awards. This information can be useful in analyzing the critical acclaim of a movie or in recommending highly acclaimed movies to users who enjoy award-winning films. It is important to note, however, that the value in this column may not provide a complete picture of the movie's quality, as the awarding of prizes can be subjective and influenced by a variety of factors.

Box Office Collection

The "box office collection" column in the Movies Dataset from Kaggle provides information on the total amount of money that a movie has earned at the box office. This column contains numerical values that represent the total worldwide gross revenue earned by the movie during its theatrical release. This information can be useful in analyzing the financial success of a movie, as well as in making recommendations to users who enjoy financially successful movies. It is important to note, however, that box office success does not necessarily correlate with critical acclaim or quality, as a movie's success can be influenced by a variety of factors such as marketing, release timing, and competition from other movies.

Duration

The "duration" column in the Movies Dataset from Kaggle provides information on the duration or length of a movie, typically measured in minutes. This column contains numerical values that represent the total length of the movie, including opening and closing credits. This information can be useful in analyzing trends or patterns in movie lengths, such as the average length of movies in a particular genre or time period. It can also be useful for creating recommendation systems based on a user's preference for shorter or longer movies.

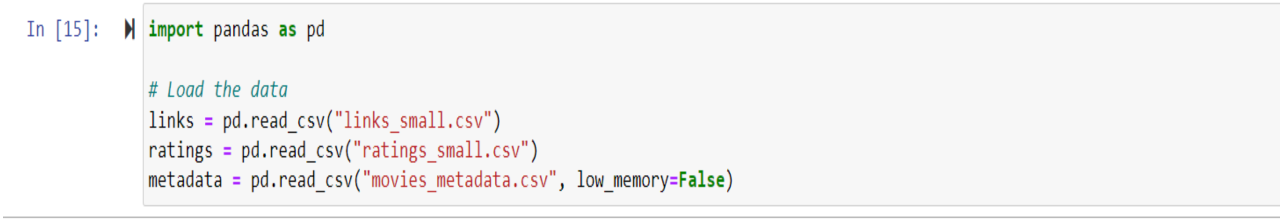
Extraction

The method for extracting columns from a movie dataset will depend on the format and structure of the dataset. However, in general, there are a few common approaches we can use.

In the Movies Dataset from Kaggle, the columns are extracted using the Comma-Separated Values (CSV) file format. This format stores data in a tabular form, where each row represents a record or observation, and each column represents a variable or attribute of that observation.

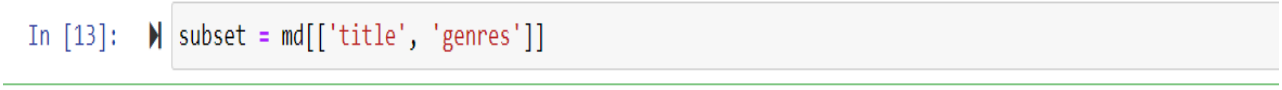
To extract specific columns from the dataset, we can load the CSV file into a programming environment or data analysis tool, such as Python, R, or Excel, and use functions or methods to select the desired columns. For example, in Python, we can use the pandas library to load the CSV file into a DataFrame object, and then use the DataFrame's indexing and selection methods to extract the desired columns.

To extract the "title" and "genres" or any other columns from the "movies_metadata.csv" file in the Movies Dataset, one could use the following Python code:



```
In [15]:  import pandas as pd

          # Load the data
          links = pd.read_csv("links_small.csv")
          ratings = pd.read_csv("ratings_small.csv")
          metadata = pd.read_csv("movies_metadata.csv", low_memory=False)
```



```
In [13]:  subset = md[['title', 'genres']]
```

Preprocessing

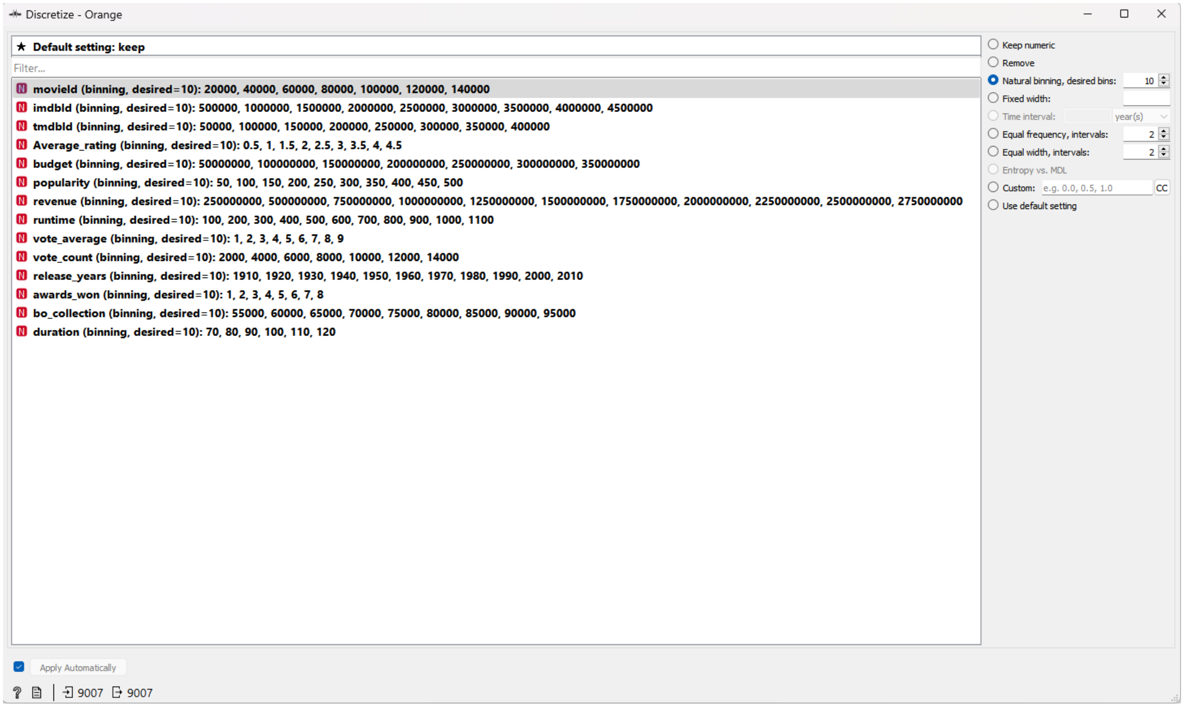
Orange is a data analysis and visualization tool that allows users to perform various tasks, including data preprocessing. Preprocessing refers to the process of cleaning, transforming, and preparing data before it can be used for analysis or modeling. In Orange, preprocessing can be done using the Preprocess tab, which includes various widgets to perform different tasks.

Some common data preprocessing and further tasks in Orange include:

Loading Data: Orange allows you to load data from various sources, including Excel, CSV, and SQL databases.

Discretize: The Discretize widget in Orange provides several methods for discretizing continuous variables, including Equal Width, Equal Frequency, and Entropy-based. These methods differ in how they group the data into intervals or bins.

To use the Discretize widget in Orange, simply select the continuous variable(s) you want to discretize and choose the desired discretization method. The widget will create a new discrete variable based on the selected method and replace the original continuous variable(s) in the data table.



Discretize - Orange

★ Default setting: keep

Filter...

N movieId (binning, desired=10): 20000, 40000, 60000, 80000, 100000, 120000, 140000
N imdbId (binning, desired=10): 500000, 1000000, 1500000, 2000000, 2500000, 3000000, 3500000, 4000000, 4500000
N tmdbId (binning, desired=10): 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000
N Average_rating (binning, desired=10): 0.5, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5
N budget (binning, desired=10): 50000000, 100000000, 150000000, 200000000, 250000000, 300000000, 350000000
N popularity (binning, desired=10): 50, 100, 150, 200, 250, 300, 350, 400, 450, 500
N revenue (binning, desired=10): 250000000, 500000000, 750000000, 1000000000, 1250000000, 1500000000, 1750000000, 2000000000, 2250000000, 2500000000, 2750000000
N runtime (binning, desired=10): 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000, 1100
N vote_average (binning, desired=10): 1, 2, 3, 4, 5, 6, 7, 8, 9
N vote_count (binning, desired=10): 2000, 4000, 6000, 8000, 10000, 12000, 14000
N release_years (binning, desired=10): 1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010
N awards_won (binning, desired=10): 1, 2, 3, 4, 5, 6, 7, 8
N bo_collection (binning, desired=10): 55000, 60000, 65000, 70000, 75000, 80000, 85000, 90000, 95000
N duration (binning, desired=10): 70, 80, 90, 100, 110, 120

○ Keep numeric
○ Remove
● Natural binning, desired bins: 10
○ Fixed width:
○ Time interval: year(s)
○ Equal frequency, intervals: 2
○ Equal width, intervals: 2
○ Entropy vs. MDL
○ Custom: e.g. 0.0, 0.5, 1.0 CC
○ Use default setting

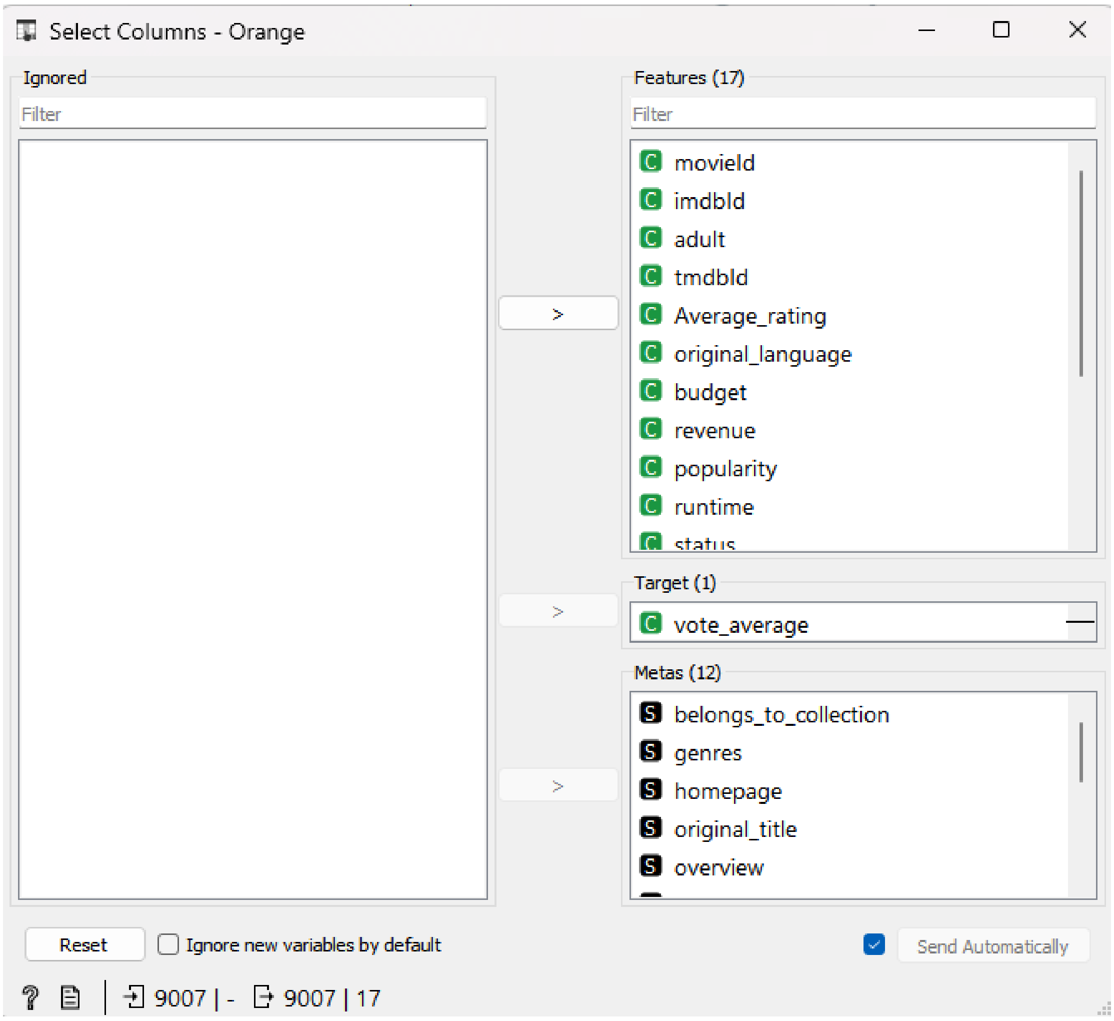
☑ Apply Automatically

9007 9007

Fig: Discretize widget in Orange

Columns Selection: Selecting columns is a common data manipulation task that can be done using the Select Columns widget in the Preprocess tab. This widget allows you to choose which columns to keep or remove from your data table.

To use the Select Columns widget, simply drag and drop it from the widget panel to the canvas. Next, connect the data table you want to manipulate to the input of the widget. The widget will display a list of all the columns in the data table, with checkboxes next to each column.



Select Columns - Orange

Ignored

Filter

Features (17)

Filter

- C movieId
- C imdbId
- C adult
- C tmdbId
- C Average_rating
- C original_language
- C budget
- C revenue
- C popularity
- C runtime
- C status

Target (1)

- C vote_average

Metas (12)

- S belongs_to_collection
- S genres
- S homepage
- S original_title
- S overview

Reset □ Ignore new variables by default ☑ Send Automatically

9007 | - 9007 | 17

Fig: Selecting Columns in Orange

Test and Score: The Test and Score widget is used to evaluate the performance of a predictive model or classifier. This widget allows you to split your data into training and testing sets, fit a model to the training data, and evaluate its performance on the testing data.

Once the data is split, the widget will fit the model to the training data and evaluate its performance on the testing data. The performance metrics used for evaluation will depend on the type of model and the target variable. For example, if you are predicting a binary target variable, the performance metrics may include accuracy, precision, recall, F1 score.

Overall, the Test and Score widget in Orange is a powerful tool for evaluating the performance of predictive models and classifiers.

Classifiers Using ORANGE On Original and Extended Dataset

KNN Classifier:

1. To use K-Nearest Neighbors (KNN) classifier in Orange using movies dataset, you can follow these steps:

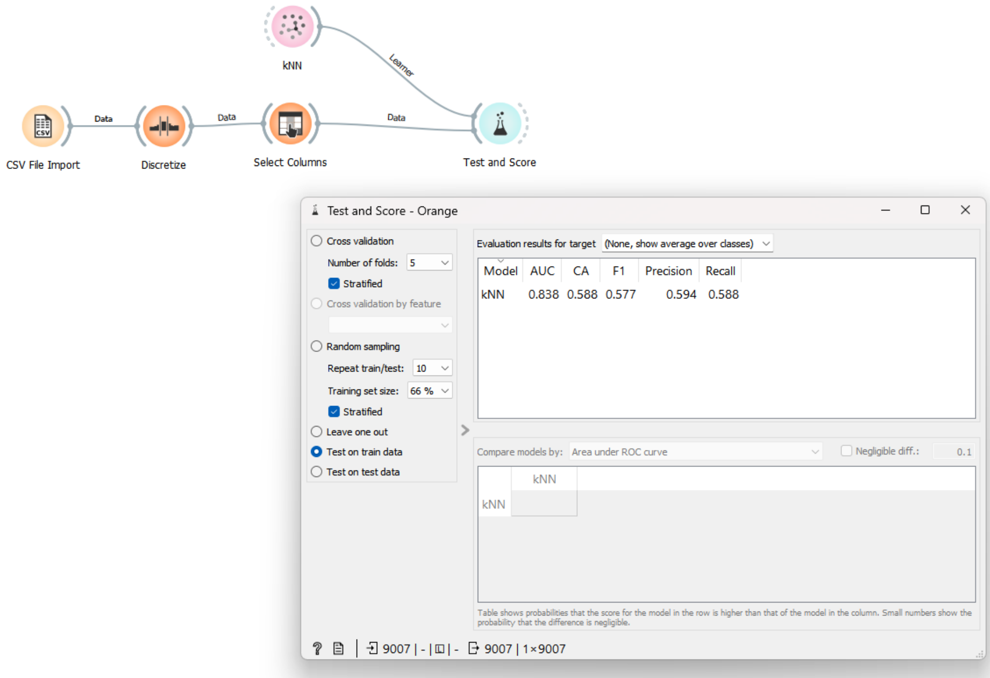
2. Import the movies dataset in Orange. The movies dataset contains information about movies, such as title, genre, actors, director, and user ratings.

3. Preprocess the data as needed, such as selecting relevant columns, discretizing continuous variables, and encoding categorical variables.

4. Split the data into training and testing sets using the Data Sampler widget. The Data Sampler widget allows you to specify the percentage or number of samples to use for the testing set.

5. Use the KNN widget to train a KNN classifier on the training data. The KNN widget allows you to specify the number of neighbors to consider and the distance metric to use for computing distances between samples.

6. Connect the trained KNN classifier and the testing data to the Test and Score widget. The Test and Score widget will evaluate the performance of the KNN classifier on the testing data and display various performance metrics, such as accuracy, precision, recall, and F1 score.



Test and Score - Orange

○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
kNN	0.838	0.588	0.577	0.594	0.588

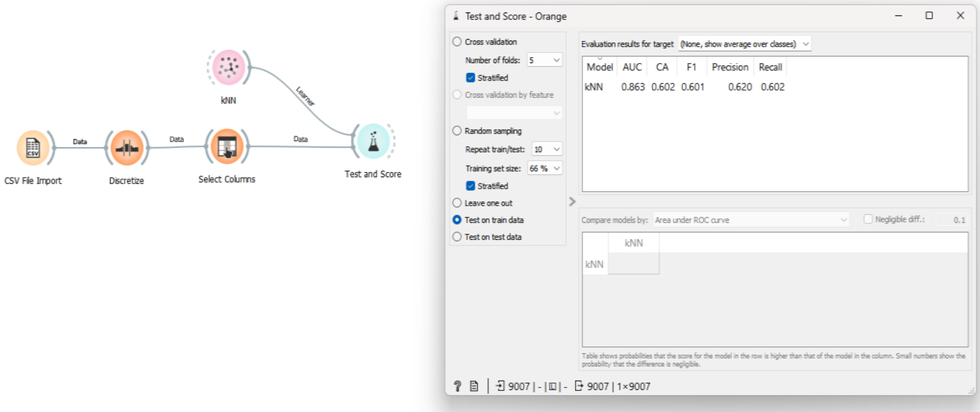
Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	kNN
kNN	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | | - 9007 | 1×9007

Fig: Test and Score of KNN Classifier for Original Dataset



Test and Score - Orange

○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
kNN	0.863	0.602	0.601	0.620	0.602

Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	kNN
kNN	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | | - 9007 | 1×9007

Fig: Test and Score of KNN Classifier for Extended Dataset

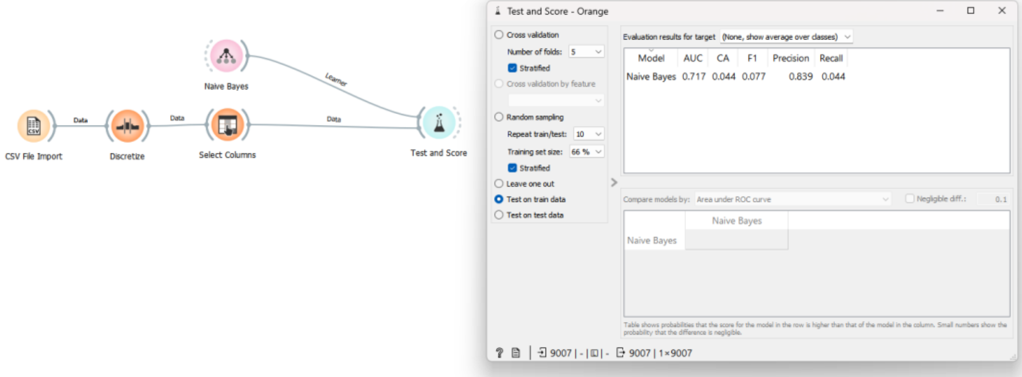
Naives Bayes Classifier:

To use Naive Bayes classifier in Orange using movies dataset, you can follow these steps:

1. Import the movies dataset in Orange. The movies dataset contains information about movies, such as title, genre, actors, director, and user ratings.

2. Preprocess the data as needed, such as selecting relevant columns, discretizing continuous variables, and encoding categorical variables.

3. Split the data into training and testing sets using the Data Sampler widget. The Data Sampler widget allows you to specify the percentage or number of samples to use for the testing set.
4. Use the Naive Bayes widget to train a Naive Bayes classifier on the training data. The Naive Bayes widget allows you to specify the type of Naive Bayes classifier to use, such as Gaussian, Multinomial, or Bernoulli.
5. Connect the trained Naive Bayes classifier and the testing data to the Test and Score widget. The Test and Score widget will evaluate the performance of the Naive Bayes classifier on the testing data and display various performance metrics, such as accuracy, precision, recall, and F1 score.



Test and Score - Orange

○ Cross validation
Number of folds: 5
☑ Stratified
○ Cross validation by feature
○ Random sampling
Repeat train/test: 10
Training set size: 66 %
☑ Stratified
○ Leave one out
● Test on train data
○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
Naive Bayes	0.717	0.044	0.077	0.839	0.044

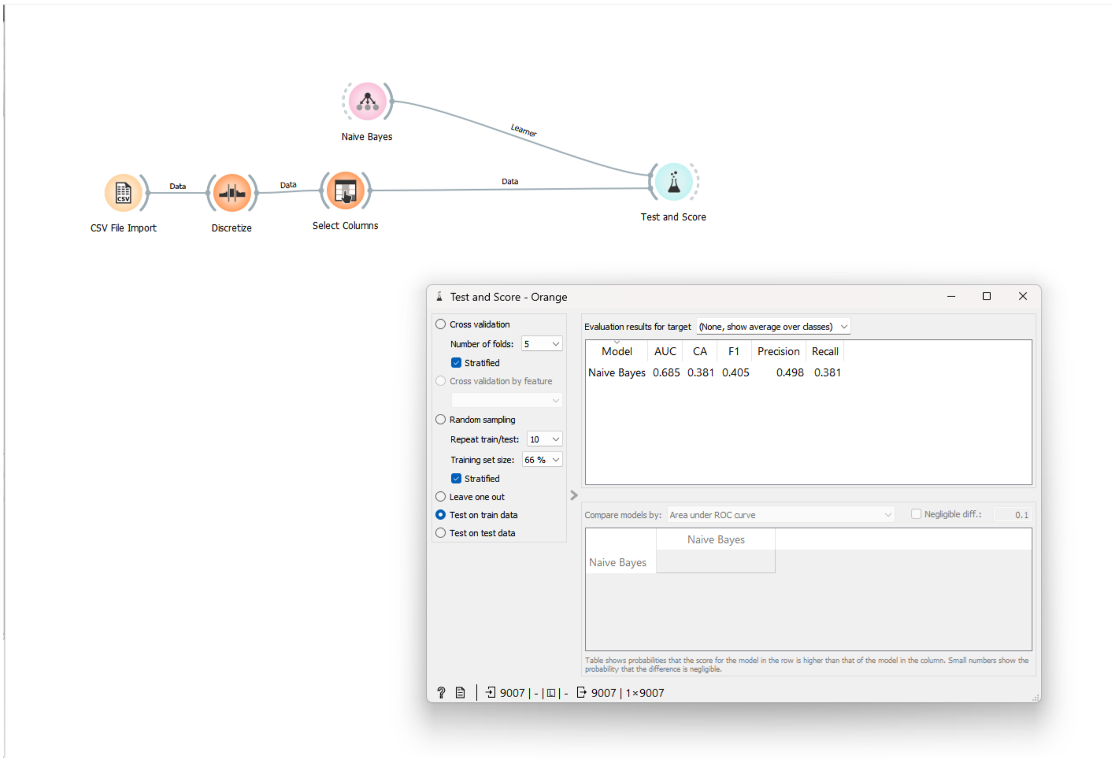
Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	Naive Bayes
Naive Bayes	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | | - 9007 | 1×9007

Fig: Test and Score of Naïve Bayes for Original Dataset



○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
Naive Bayes	0.685	0.381	0.405	0.498	0.381

Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	Naive Bayes
Naive Bayes	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | | - 9007 | 1×9007

Fig: Test and Score of Naïve Bayes for Extended Dataset

Stochastic Gradient Descent:

Stochastic Gradient Descent (SGD) classifier is a type of linear classifier in Orange, which is used for binary classification or multi-class classification tasks. The SGD classifier is a popular algorithm for large-scale learning tasks and works well with high-dimensional data.

In Orange, the SGD classifier can be used with the classification workflows, such as Test and Score, Predictions, and Cross-validation. The SGD classifier works by minimizing the loss function using stochastic gradient descent optimization. The loss function measures the difference between the predicted and actual class labels and the optimization algorithm tries to find the optimal weights for the linear classifier that minimize the loss function.

To use the SGD classifier in Orange, follow these steps:

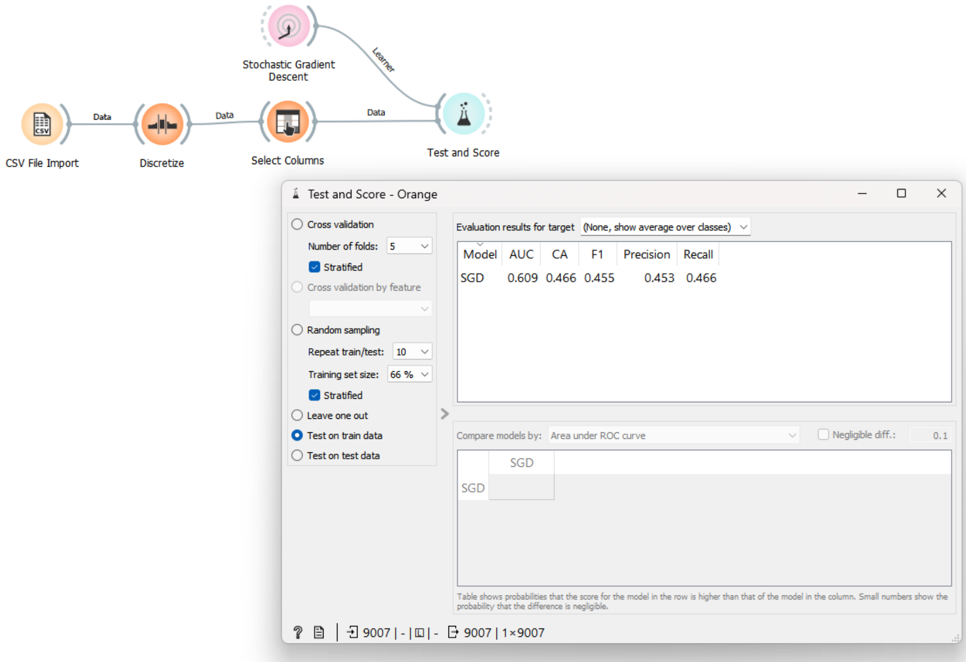
1. Import the dataset in Orange.

2. Preprocess the data by selecting relevant columns, encoding categorical variables, and discretizing continuous variables.

3. Use the SGD classifier widget in Orange and connect it to the preprocessed data.

4. Configure the SGD classifier widget by specifying the loss function, learning rate, penalty, and other parameters.

5. Connect the SGD classifier to the Test and Score widget to evaluate its performance on the test set.



○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
SGD	0.609	0.466	0.455	0.453	0.466

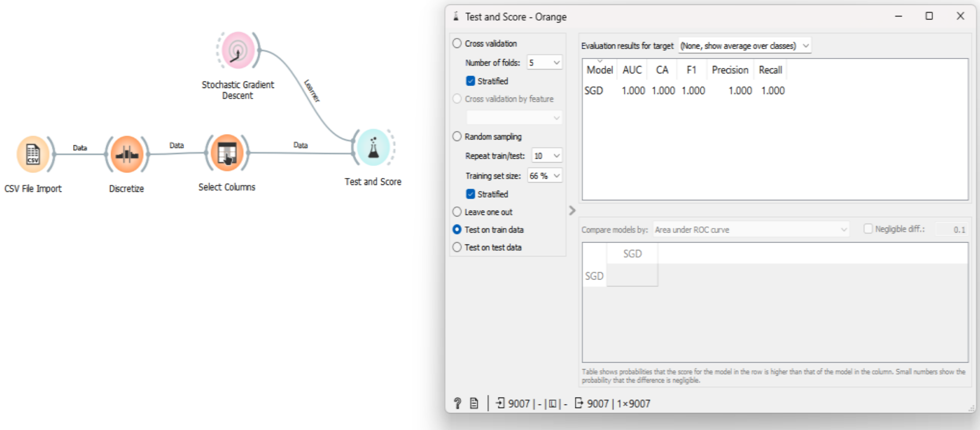
Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	SGD
SGD	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | - | 9007 | 1×9007

Fig: Test and Score of SGD for Original Dataset



○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
SGD	1.000	1.000	1.000	1.000	1.000

Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	SGD
SGD	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | - | 9007 | 1×9007

Fig: Test and Score of SGD for Extended Dataset

Neural Network:

Neural network is a machine learning algorithm used for classification and regression tasks in Orange. It is a type of deep learning algorithm that uses artificial neural networks to model complex relationships between inputs and outputs. Neural networks are particularly effective at recognizing patterns in data and can be used for a wide range of applications, including image recognition, speech recognition, and natural language processing.

In Orange, the Neural Network widget can be used to build, train, and evaluate a neural network model. Here are the general steps to use neural network in Orange:

1. Import the dataset in Orange.

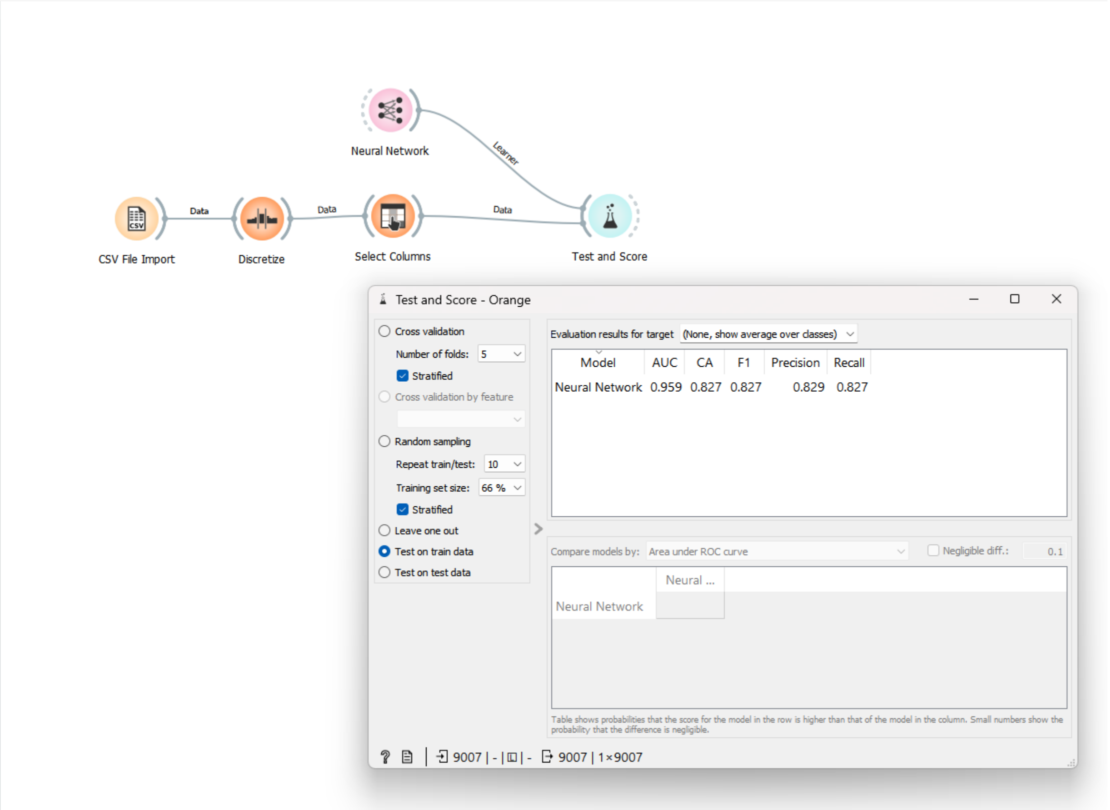
2. Preprocess the data by selecting relevant columns, encoding categorical variables, and discretizing continuous variables.

3. Use the Neural Network widget in Orange and connect it to the preprocessed data.

4. Configure the Neural Network widget by specifying the number of hidden layers, the number of neurons in each layer, the activation function, and other hyperparameters.

5. Train the neural network by clicking on the Train button.

6. Evaluate the performance of the neural network by connecting it to the Test and Score widget or Cross-validation widget. These widgets can be used to compute various performance metrics such as accuracy, precision, recall, F1-score.



○ Cross validation
Number of folds: 5
☑ Stratified
○ Cross validation by feature
○ Random sampling
Repeat train/test: 10
Training set size: 66 %
☑ Stratified
○ Leave one out
● Test on train data
○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
Neural Network	0.959	0.827	0.827	0.829	0.827

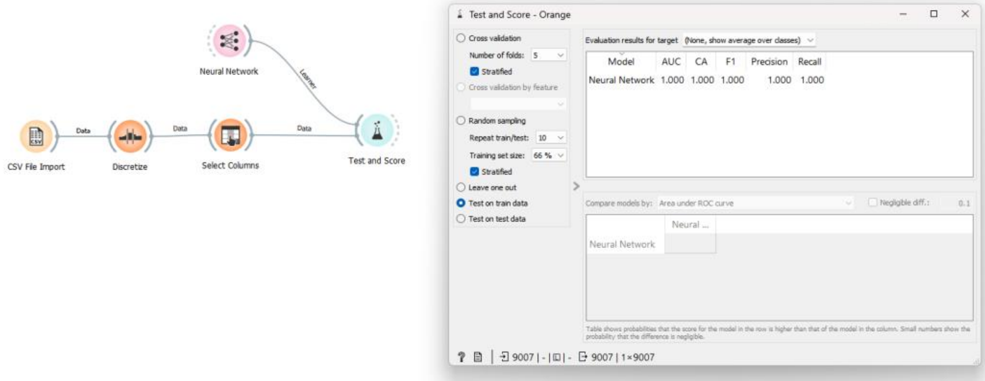
Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	Neural ...
Neural Network	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | - | 9007 | 1×9007

Fig: Test and Score of Neural Network for Original Dataset



○ Cross validation
Number of folds: 5
☑ Stratified
○ Cross validation by feature
○ Random sampling
Repeat train/test: 10
Training set size: 66 %
☑ Stratified
○ Leave one out
● Test on train data
○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
Neural Network	1.000	1.000	1.000	1.000	1.000

Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	Neural ...
Neural Network	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | - | 9007 | 1×9007

Fig: Test and Score of Neural Network for Extended Dataset

Screenshots

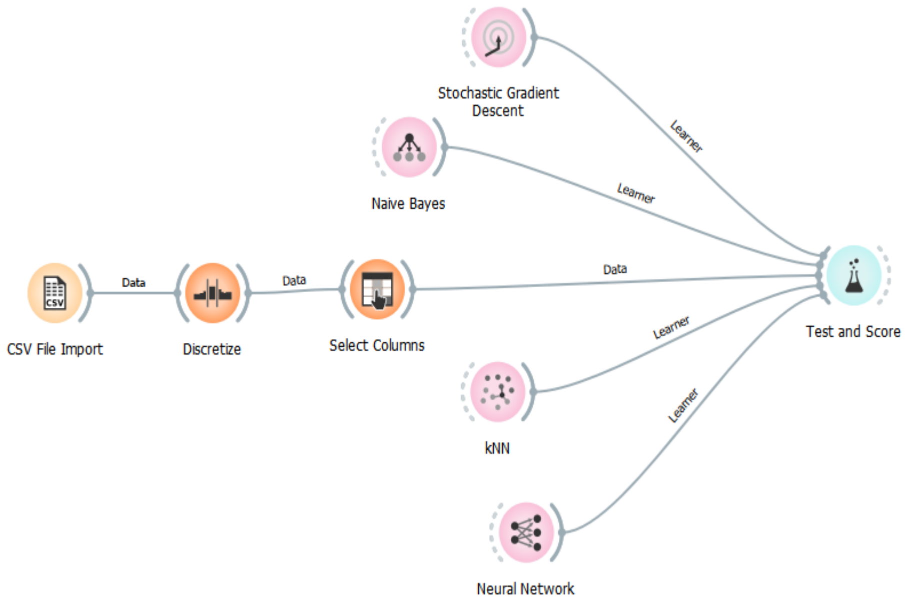
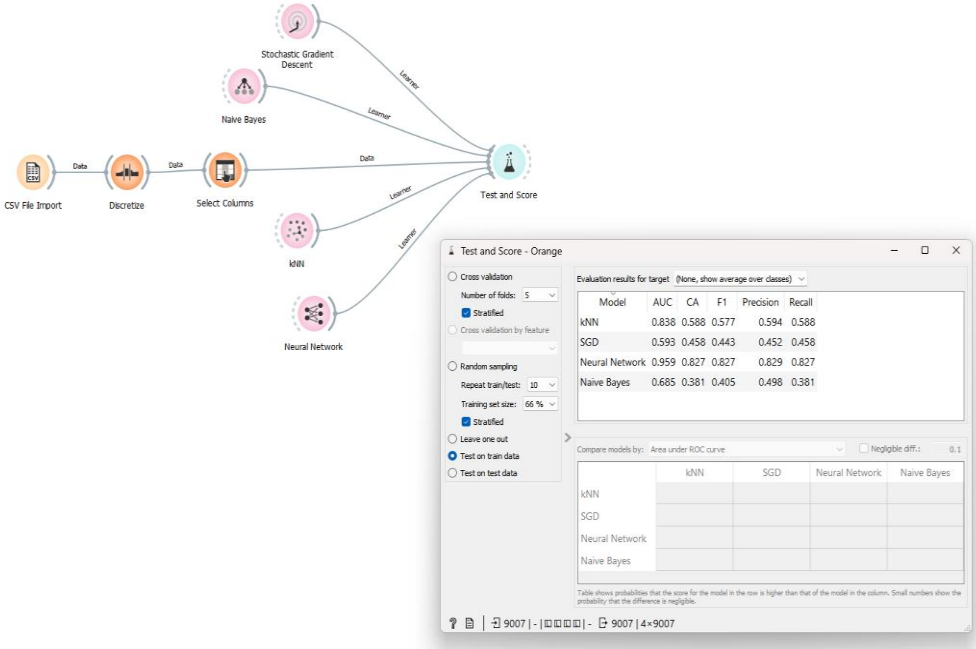


Fig: Analysis in Orange for Original Dataset

- Comparing F1-Scores of Original And Extended Dataset:



○ Cross validation
Number of folds: 5
☑ Stratified
○ Cross validation by feature
○ Random sampling
Repeat train/test: 10
Training set size: 66 %
☑ Stratified
○ Leave one out
● Test on train data
○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
kNN	0.838	0.588	0.577	0.594	0.588
SGD	0.593	0.458	0.443	0.452	0.458
Neural Network	0.959	0.827	0.827	0.829	0.827
Naive Bayes	0.685	0.381	0.405	0.498	0.381

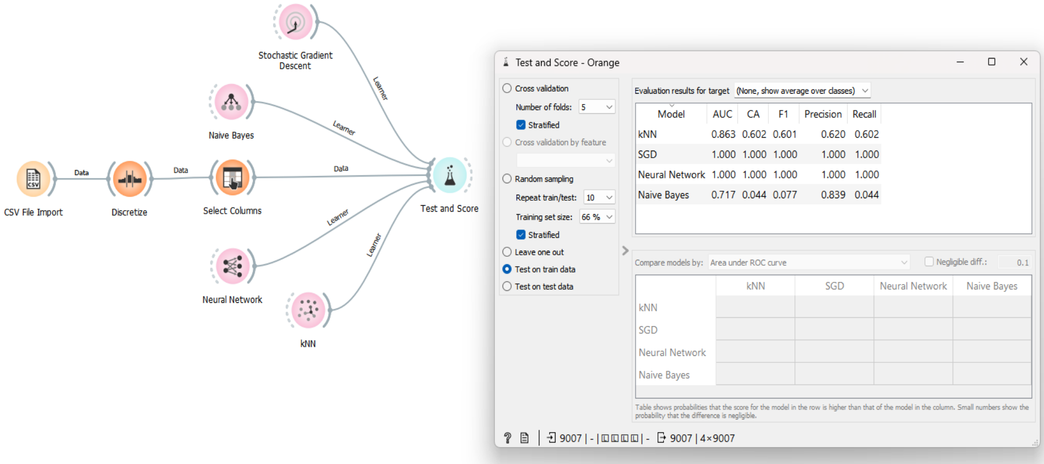
Compare models by: Area under ROC curve □ Negligible diff.: 0.1

	kNN	SGD	Neural Network	Naive Bayes
kNN				
SGD				
Neural Network				
Naive Bayes				

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | - 9007 | 4×9007

Fig: Classifiers Test and Score in Orange for Original Dataset



Test and Score - Orange

○ Cross validation

Number of folds: 5

☑ Stratified

○ Cross validation by feature

○ Random sampling

Repeat train/test: 10

Training set size: 66 %

☑ Stratified

○ Leave one out

● Test on train data

○ Test on test data

Evaluation results for target (None, show average over classes)

Model	AUC	CA	F1	Precision	Recall
kNN	0.863	0.602	0.601	0.620	0.602
SGD	1.000	1.000	1.000	1.000	1.000
Neural Network	1.000	1.000	1.000	1.000	1.000
Naive Bayes	0.717	0.044	0.077	0.839	0.044

Compare models by: Area under ROC curve ☐ Negligible diff.: 0.1

	kNN	SGD	Neural Network	Naive Bayes
kNN				
SGD				
Neural Network				
Naive Bayes				

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

9007 | - | | - 9007 | 4×9007

Fig: Classifiers Test and Score in Orange for Extended Dataset

Lisp miner

LispMiner is a data mining and knowledge discovery tool that provides a suite of data mining algorithms, including action rules mining. Action rules mining is a type of rule-based classification that generates rules based on the actions that users perform on a website or system. In the case of the movies dataset, action rules mining can be used to generate rules that identify patterns in user behavior or preferences.

Action Rules

Here are the general steps to use LispMiner action rules mining in Orange for the movies dataset:

1. Import the movies dataset into Orange.

2. Preprocess the data as needed, such as selecting relevant columns and discretizing continuous variables.

3. Use the Discretize widget to discretize the continuous variables in the dataset.

4. Use the LispMiner widget in Orange and connect it to the preprocessed data.

5. Configure the LispMiner widget by specifying the target variable, the minimum support and confidence values, and other parameters.

6. Run the LispMiner algorithm to generate the action rules.

Use the Results widget to explore the action rules generated by LispMiner. The Results widget displays the action rules in a tabular format and allows you to filter and sort the rules based on various criteria.

Some possible action rules that can be generated from the movie dataset are:

- If the genre is Action and the director is Christopher Nolan, then the rating is likely to be high.
- If the language is English and the release year is after 2000, then the movie is likely to be popular.
- If the runtime is longer than 2 hours and the genre is Drama, then the rating is likely to be high.

- If the genre is Horror and the rating is low, then the movie is likely to have poor critical reviews.
- If the cast includes Tom Hanks and the genre is Drama, then the movie is likely to have high audience ratings.

Action rules can be generated using an extended dataset, which includes additional columns or features that are not present in the original dataset. This is called attribute extension in LispMiner.

Attribute extension is a powerful feature of LispMiner that allows you to include additional attributes or variables in the mining process, which can potentially improve the quality of the generated rules. The extended dataset is created by joining the original dataset with one or more additional datasets or tables that contain the additional attributes or variables.

When using attribute extension in LispMiner to generate action rules, the extended dataset is used as input to the mining algorithm. The additional variables included in the extended dataset can be used to refine the search for action rules, and to generate more accurate and relevant rules.

For example, in the case of the movie dataset, additional variables such as movie budgets, production companies, or the number of Oscar awards won, could be added to the extended dataset. These additional variables could help to identify patterns and rules related to the financial success of movies, or the impact of critical acclaim on audience ratings.

Overall, attribute extension in LispMiner is a powerful tool for improving the accuracy and relevance of action rules generated from a dataset. By including additional variables in the mining process, you can gain deeper insights into the patterns and relationships within the data, and uncover hidden trends and correlations that may not be evident from the original dataset alone.

Analysis of Action Rule Mining

Action rule mining is a data mining technique that is used to identify patterns and relationships in a dataset, based on the occurrence of specific actions or events. In the case of the movie dataset, action rule mining can be used to identify interesting and useful patterns or rules that can help to explain the success or failure of movies, or to predict the performance of new movies.

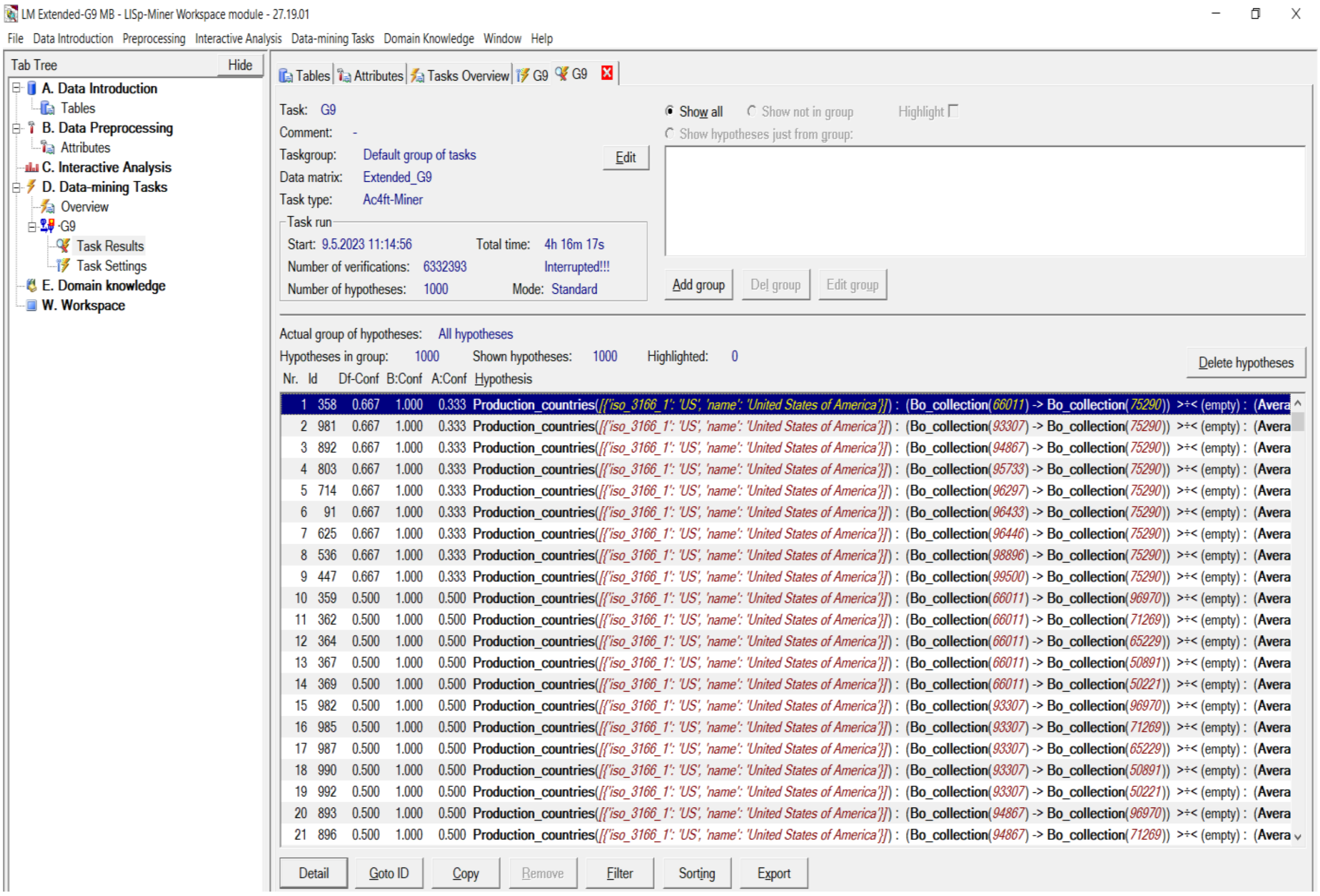
Here are some possible analyses that can be performed using action rule mining for the movie dataset:

- Identification of key factors influencing movie success: By generating action rules that associate specific attributes or variables with high or low movie ratings, action rule mining can help to identify the key factors that influence the success or failure of movies. For example, action rules may reveal that movies with certain genres, directors, or actors tend to perform better than others, or that critical acclaim is a more important factor than box office success in determining audience ratings.

- Prediction of movie performance: Action rule mining can also be used to predict the performance of new movies, based on the attributes or variables that are most strongly associated with high or low ratings. By analyzing past patterns and relationships in the dataset, action rule mining can generate rules that predict the likelihood of a movie being successful or not, based on its genre, cast, director, release date, or other attributes.

- Identification of hidden trends and correlations: Action rule mining can help to uncover hidden trends and correlations within the movie dataset that may not be evident from a simple analysis of the data. For example, action rules may reveal that movies with longer runtimes tend to be associated with higher ratings, or that movies released during certain months of the year tend to perform better than others.

- Refinement of marketing and distribution strategies: By identifying the factors that are most strongly associated with movie success, action rule mining can help movie studios and distributors to refine their marketing and distribution strategies. For example, action rules may suggest that movies with certain genres or themes should be released at certain times of the year, or that certain cast members or directors should be promoted more heavily in advertising campaigns.


LM Extended-G9 MB - LISp-Miner Workspace module - 27.19.01
File Data Introduction Preprocessing Interactive Analysis Data-mining Tasks Domain Knowledge Window Help
Tab Tree Hide
A. Data Introduction
Tables
B. Data Preprocessing
Attributes
C. Interactive Analysis
D. Data-mining Tasks
Overview
G9
Task Results
Task Settings
E. Domain knowledge
W. Workspace
Tables Attributes Tasks Overview G9 G9
Edit
Add group Del group Edit group
Delete hypotheses
Detail Goto ID Copy Remove Filter Sorting Export

Task: G9

Comment: -

Taskgroup: Default group of tasks

Data matrix: Extended_G9

Task type: Ac4ft-Miner

Task run

Start: 9.5.2023 11:14:56 Total time: 4h 16m 17s

Number of verifications: 6332393 Interrupted!!!

Number of hypotheses: 1000 Mode: Standard

● Show all ○ Show not in group Highlight □

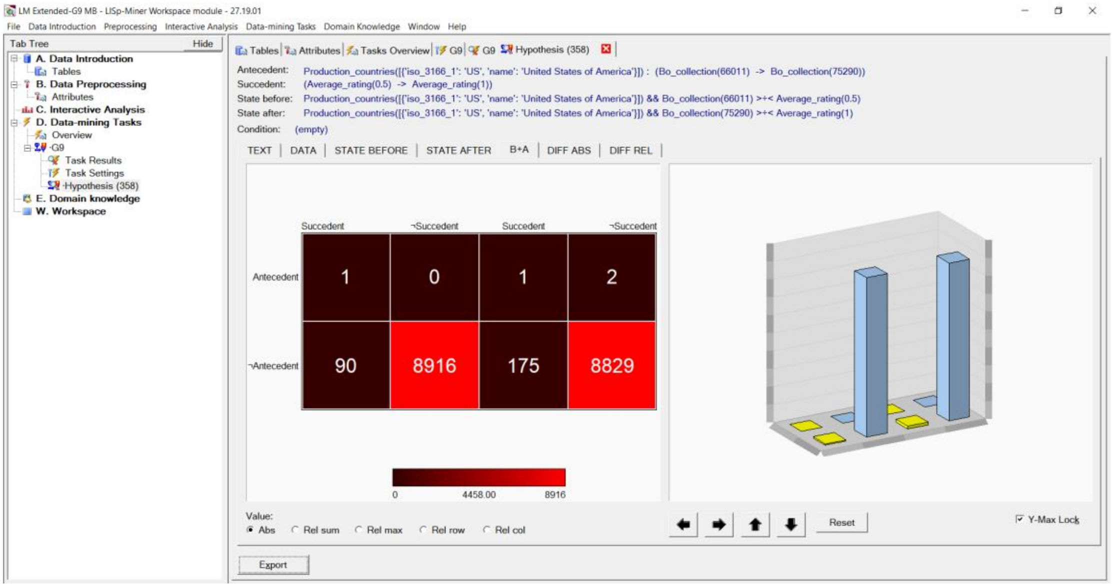
○ Show hypotheses just from group:

Actual group of hypotheses: All hypotheses

Hypotheses in group: 1000 Shown hypotheses: 1000 Highlighted: 0

Nr.	Id	Df-Conf	B:Conf	A:Conf	Hypothesis
1	358	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(75290)) >÷< (empty) : (Avera
2	981	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(75290)) >÷< (empty) : (Avera
3	892	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(94867) -> Bo_collection(75290)) >÷< (empty) : (Avera
4	803	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(95733) -> Bo_collection(75290)) >÷< (empty) : (Avera
5	714	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(96297) -> Bo_collection(75290)) >÷< (empty) : (Avera
6	91	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(96433) -> Bo_collection(75290)) >÷< (empty) : (Avera
7	625	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(96446) -> Bo_collection(75290)) >÷< (empty) : (Avera
8	536	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(98896) -> Bo_collection(75290)) >÷< (empty) : (Avera
9	447	0.667	1.000	0.333	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(99500) -> Bo_collection(75290)) >÷< (empty) : (Avera
10	359	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(96970)) >÷< (empty) : (Avera
11	362	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(71269)) >÷< (empty) : (Avera
12	364	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(65229)) >÷< (empty) : (Avera
13	367	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(50891)) >÷< (empty) : (Avera
14	369	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(50221)) >÷< (empty) : (Avera
15	982	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(96970)) >÷< (empty) : (Avera
16	985	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(71269)) >÷< (empty) : (Avera
17	987	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(65229)) >÷< (empty) : (Avera
18	990	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(50891)) >÷< (empty) : (Avera
19	992	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(93307) -> Bo_collection(50221)) >÷< (empty) : (Avera
20	893	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(94867) -> Bo_collection(96970)) >÷< (empty) : (Avera
21	896	0.500	1.000	0.500	Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(94867) -> Bo_collection(71269)) >÷< (empty) : (Avera

Fig: Analysis of Action Rules Using LispMiner


LM Extended-G9 MB - LISp-Miner Workspace module - 27.19.01
File Data Introduction Preprocessing Interactive Analysis Data-mining Tasks Domain Knowledge Window Help
Tab Tree Hide
A. Data Introduction
Tables
B. Data Preprocessing
Attributes
C. Interactive Analysis
D. Data-mining Tasks
Overview
G9
Task Results
Task Settings
Hypothesis (358)
E. Domain knowledge
W. Workspace
Tables Attributes Tasks Overview G9 G9 Hypothesis (358)
TEXT | DATA | STATE BEFORE | STATE AFTER | B+A | DIFF ABS | DIFF REL
0 4458.00 8916
Reset
Export

Antecedent: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(66011) -> Bo_collection(75290))

Succedent: (Average_rating(0.5) -> Average_rating(1))

State before: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(66011) >÷< Average_rating(0.5)

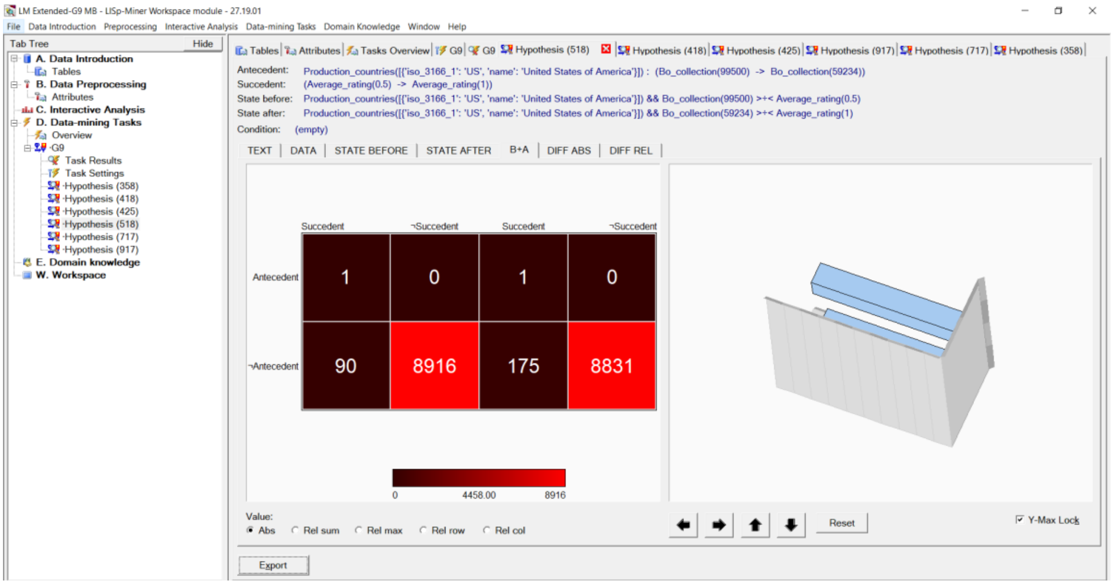
State after: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(75290) >÷< Average_rating(1)

Condition: (empty)

	Succedent	¬Succedent	Succedent	¬Succedent
Antecedent	1	0	1	2
¬Antecedent	90	8916	175	8829

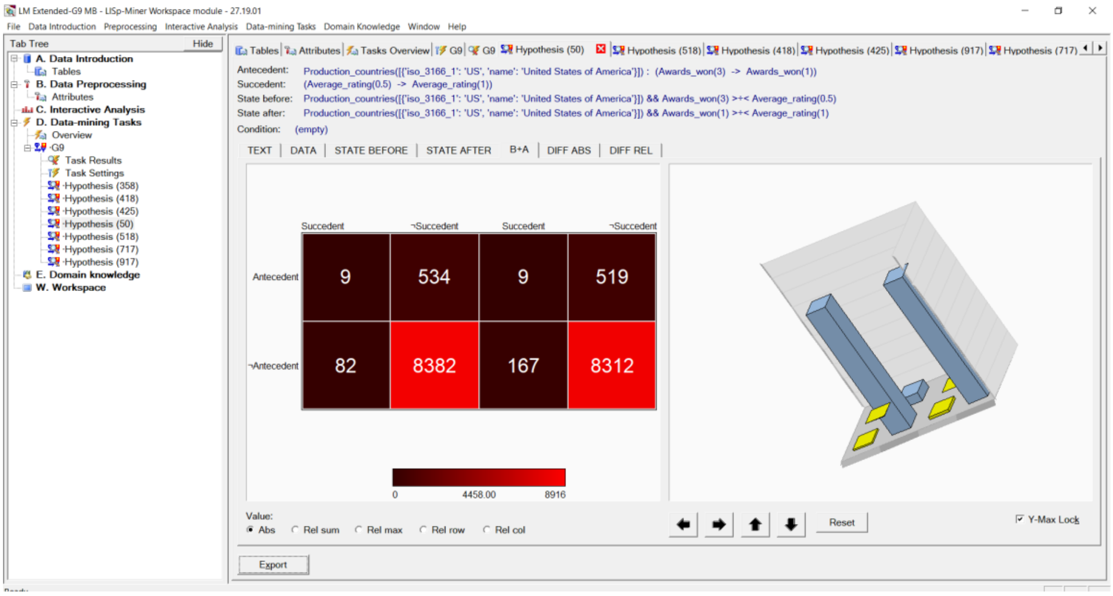
Value: ● Abs ○ Rel sum ○ Rel max ○ Rel row ○ Rel col

☑ Y-Max Lock


LM Extended-G9 MB - LISp-Miner Workspace module - 27.19.01
File Data Introduction Preprocessing Interactive Analysis Data-mining Tasks Domain Knowledge Window Help
Tab Tree Hide
A. Data Introduction
Tables
B. Data Preprocessing
Attributes
C. Interactive Analysis
D. Data-mining Tasks
Overview
G9
Task Results
Task Settings
Hypothesis (358)
Hypothesis (418)
Hypothesis (425)
Hypothesis (518)
Hypothesis (717)
Hypothesis (917)
E. Domain knowledge
W. Workspace
Tables | Attributes | Tasks Overview | G9 | G9 | Hypothesis (518) | Hypothesis (418) | Hypothesis (425) | Hypothesis (917) | Hypothesis (717) | Hypothesis (358)
TEXT | DATA | STATE BEFORE | STATE AFTER | B+A | DIFF ABS | DIFF REL
0
4458.00
8916
Value:
● Abs ○ Rel sum ○ Rel max ○ Rel row ○ Rel col
Reset
☑ Y-Max Lock
Export

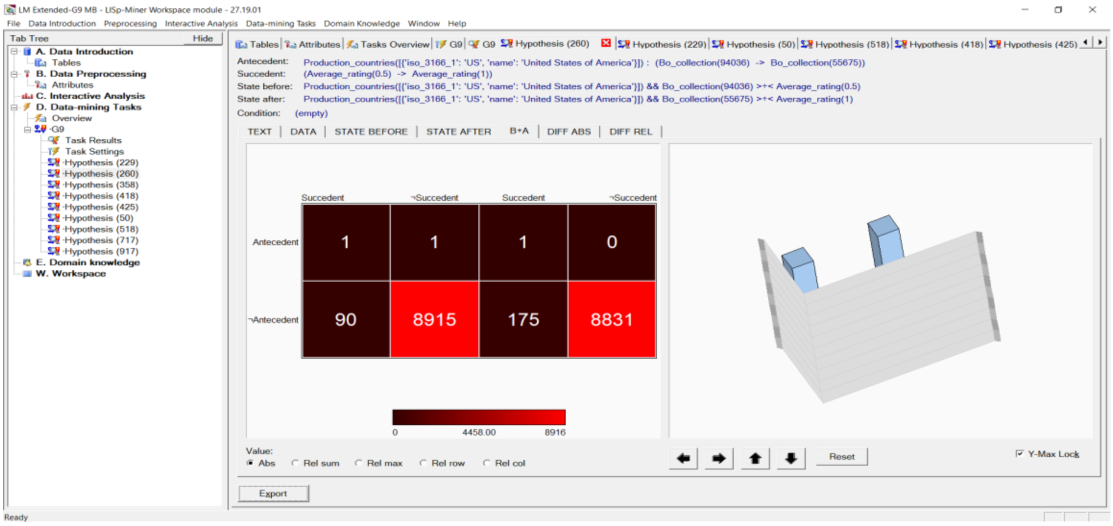
Antecedent: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(99500) -> Bo_collection(59234))
Succedent: (Average_rating(0.5) -> Average_rating(1))
State before: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(99500) >+< Average_rating(0.5)
State after: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(59234) >+< Average_rating(1)
Condition: (empty)

	Succedent	¬Succedent	Succedent	¬Succedent
Antecedent	1	0	1	0
¬Antecedent	90	8916	175	8831


LM Extended-G9 MB - LISp-Miner Workspace module - 27.19.01
File Data Introduction Preprocessing Interactive Analysis Data-mining Tasks Domain Knowledge Window Help
Tab Tree Hide
A. Data Introduction
Tables
B. Data Preprocessing
Attributes
C. Interactive Analysis
D. Data-mining Tasks
Overview
G9
Task Results
Task Settings
Hypothesis (358)
Hypothesis (418)
Hypothesis (425)
Hypothesis (50)
Hypothesis (518)
Hypothesis (717)
Hypothesis (917)
E. Domain knowledge
W. Workspace
Tables | Attributes | Tasks Overview | G9 | G9 | Hypothesis (50) | Hypothesis (518) | Hypothesis (418) | Hypothesis (425) | Hypothesis (917) | Hypothesis (717)
TEXT | DATA | STATE BEFORE | STATE AFTER | B+A | DIFF ABS | DIFF REL
0
4458.00
8916
Value:
● Abs ○ Rel sum ○ Rel max ○ Rel row ○ Rel col
Reset
☑ Y-Max Lock
Export

Antecedent: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Awards_won(3) -> Awards_won(1))
Succedent: (Average_rating(0.5) -> Average_rating(1))
State before: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Awards_won(3) >+< Average_rating(0.5)
State after: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Awards_won(1) >+< Average_rating(1)
Condition: (empty)

	Succedent	¬Succedent	Succedent	¬Succedent
Antecedent	9	534	9	519
¬Antecedent	82	8382	167	8312



Antecedent: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) : (Bo_collection(94036) -> Bo_collection(55675))
Succedent: (Average_rating(0.5) -> Average_rating(1))
State before: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(94036) >+< Average_rating(0.5)
State after: Production_countries([{'iso_3166_1': 'US', 'name': 'United States of America'}]) && Bo_collection(55675) >+< Average_rating(1)
Condition: (empty)

	Succedent	¬Succedent	Succedent	¬Succedent
Antecedent	1	1	1	0
¬Antecedent	90	8915	175	8831

Value:
● Abs ○ Rel sum ○ Rel max ○ Rel row ○ Rel col