# Introduction

# Tools used:

Rapid Miner 4.6

## Why to do use any software for this data analysis?

I'm having 7,000 student records for which I've to extract some useful information for future predictions. For example I need information according to different patterns. E.g.

How many bachelor students are under 30?

Which school-city is most popular in era 2010-2011?

What is Student Major for students having age >30?

What are mostly student major, gender and school for students having age 30-37?

I can't do it manually. It would take much time for analysis of such patterns from a large domain if I do it by counting data or records myself. There are some algorithms that give me such information within few minutes. For applying that algorithms, for knowledge extraction from big data there are many algorithms and tools available and Rapid Miner is ranked 1 for such analysis that’s why I'm taking help from this software for ease in doing correct and precise knowledge discovery.

## Why Rapid Miner?

Data mining is an essential process where intelligent methods are applied in order to extract data patterns. Rapid Miner provides much more analysis steps (operators) than any other data Mining tool e.g. as compared to Weka and much more possibilities to combine them. It provides an additional set of about 400 operators for many aspects of Data Mining not covered by Weka. E.g. Preprocessing methods, validation and visualization techniques which are not available within Weka.

Rapid Miner 4.6



# Data Analysis and Knowledge Discovery

Data mining functionalities are used to specify the kind of patterns to be found in

data mining tasks.

In general, data mining tasks can be classified into two categories:

Descriptive and predictive. **Descriptive mining tasks characterize the general properties**

**of the data in the database**. Predictive mining tasks perform inference on the current data

in order to make predictions. By data mining system we can generate thousands of patterns or rules but only small fraction of the patterns potentially generated would actually be of interest to any given user. A pattern is interested if it is (1) *easily understood* by humans, (2) *valid* on new or test data with some degree of *certainty*, (3) potentially *useful*, and (4) *novel*. A pattern is also interesting if it validates a hypothesis that the user *sought to confirm*. An interesting pattern represents knowledge.

In current dataset Student data is having a large amount of Categorical Data, in order to analyze and for extracting of useful information I am going to apply following data mining approaches.

## Statistical Analysis

# Algorithms used for in-depth analysis of data

## What is frequent item-set mining?

Frequent pattern mining searches for recurring relationships in a given data set**.** Frequent itemset mining leads to the discovery of associations and correlations among items in large transactional or relational data sets.

A typical example of frequent itemset mining is market basket analysis. This process analyzes customer buying habits by finding associations between the different items that customers place in their “shopping baskets”. The discovery of such associations can help retailers develop marketing strategies by gaining insight into which items are frequently purchased together by customers. For instance, ***if customers are buying milk, how likely are they to also buy bread (and what kind of bread) on the same trip to the supermarket***? Such information can lead to increased sales by helping retailers do selective marketing and plan their shelf space.

Similarly,in given data set if I want to analyze Graduate student's behavior who belongs to college "Art, Humanities and Social sci" then what is the most student's citizenship, gender, Cert-Avg(High score) etc.

I would like to get such information by using frequent item-set generation that what is likelihood value of graduate students with respect to their residency, citizenship, diploma description and gender as well .

### Efficient and Scalable Frequent Item set Mining Methods

**Apriori**, is the basic algorithm for finding frequent itemsets**. Explanation of Apriori here:**

Apriori generates candidate sets whereas FPGrowth uses specialized data structures(no candidate sets).

### FPGrowth

It found frequent itemSet without candidate generation. It constructs a highly compact data structure (an *FP-tree*) to compress the original transaction database. Rather than employing the generate and-test strategy of Apriori-like methods, it focuses on frequent pattern (fragment) growth, which avoids costly candidate generation, resulting in greater efficiency.

### Generating Association Rules

Rule support and confidence are two measures of rule interestingness. Once the frequent itemsets from transactions in a database *D* have been found, it is straightforward to generate strong association rules from them (where *strong* association rules satisfy both minimum support and minimum confidence). This can be done using Equation for confidence, which is here:

***Confidence(A🡪 B*) = *P*(*B* | *A*) = *support count*(*A* U *B*) /*support count*(*A*) .**

## Decision Tree Analysis

Conducting analysis of decision making under uncertainty using decision trees serves several purposes.

* First, a decision tree is a visual representation of a decision situation (and hence aids communication).
* Second, the branches of a tree explicitly show all those factors within the analysis that are considered relevant to the decision (and implicitly those that are not).
* Third, and more subtly, a decision tree generally captures the idea that if different decisions were to be taken then the structural nature of a situation (and hence of the model) may have changed dramatically.
* Fourth, and arguably the most powerful, a decision tree allows for forward and backward calculation paths to happen and hence the choice of the correct decision to take (optimality of decision making, or optimal exercise if embedded real options) is made automatically.

## Cluster Analysis

Clustering partitions large data sets into groups according to their *similarity*. A cluster is a collection of data objects that are *similar* to one another within the same cluster and are *dissimilar* to the objects in other clusters.

Cluster analysis can be used as a stand-alone tool **to gain insight into the distribution of data, to observe the characteristics of each cluster**, and to focus on a particular set of clusters for further analysis

Additional advantages of such a clustering-based process are that it is adaptable to changes and helps single out useful features that distinguish different groups.

### Centroid-based clustering

### Density-based clustering

# Data Analysis using different algorithms:

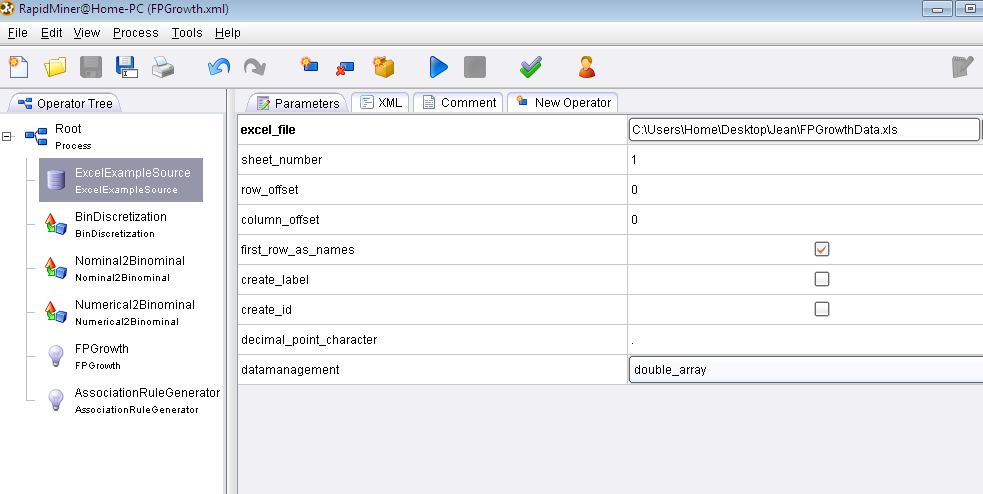
## FPGROWTH

### Input operators:

### Output:- Frequent Item sets

## Rule Association Mining

### Input Screen for "Rule Association Mining"



***Input Operators***

**Load File:**Its csv file here

**BinDiscretization:** where bin-size is 5

**Nominal2Binominal**

**FPGrowth:** Generates frequent itemsets

**AssociationRule..**

For Generation of association rules

Here I will put screen-shot for all input operators.

Then Output screen would be placed too.

## K-MEAN Cluster Analysis

## KMediods Cluster Analysis

## Performance Analysis

Here would be performance analysis(accuracy, precision and recall) of K-MEAN and KMediods Algorithms.

***Evaluation of clusterings using the validation operators ClusterCentroidEvaluator and ClusterDensityEvaluator***

### PerformanceVector [ClusterCentroidEvaluator]

An evaluator for centroid based clustering methods. The average within cluster distance is calculated by averaging the distance between the centroid and all examples of a cluster.

#### K=3

PerformanceVector:  
Avg. within centroid distance: 2.551  
Avg. within centroid distance\_cluster\_0: 2.029  
Avg. within centroid distance\_cluster\_1: 1.752  
Avg. within centroid distance\_cluster\_2: 7.913  
Davies Bouldin: 0.115

#### K=6

PerformanceVector:  
Avg. within centroid distance: 0.867  
Avg. within centroid distance\_cluster\_0: 0.847  
Avg. within centroid distance\_cluster\_1: 2.475  
Avg. within centroid distance\_cluster\_2: 0.430  
Avg. within centroid distance\_cluster\_3: 0.920  
Avg. within centroid distance\_cluster\_4: 0.000  
Avg. within centroid distance\_cluster\_5: 0.554  
Davies Bouldin: 0.076

#### Comment:

As K value is increased the items are nicely grouped (More Similar attributes forming a cluster)because centroid distance is decreasing by making more clusters.

### PerformanceVector [ClusterDensityEvaluator]

Input : Cluster Model, Similarity Measure

Output: Performance vector

#### K=3

##### Cluster Model

Cluster 0: 9 items  
Cluster 1: 11 items  
Cluster 2: 15 items  
Total number of items: 35

PerformanceVector:  
Avg. within cluster similarity: 11.880  
Avg. within cluster similarity for cluster 0: 8.788  
Avg. within cluster similarity for cluster 1: 10.670  
Avg. within cluster similarity for cluster 2: 14.624

#### K=6

PerformanceVector:  
Avg. within cluster similarity: 6.753  
Avg. within cluster similarity for cluster 0: 2.985  
Avg. within cluster similarity for cluster 1: 7.880  
Avg. within cluster similarity for cluster 2: 7.768  
Avg. within cluster similarity for cluster 3: 7.875  
Avg. within cluster similarity for cluster 4: 5.883  
Avg. within cluster similarity for cluster 5: 1.968

#### Comment:

## Decision Tree Analysis

# GLOSSARY

## BinDiscretization:

This operator discretizes all numeric attributes in the dataset into nominal attributes. This discretization is performed by simple binning, i.e. the specified number of equally sized bins is created and the numerical values are simply sorted into those bins e.g. in current data set it is applied for age (a numeric value attribute)

## Nominal2Binominal:

This operator maps the values of all nominal values to binary attributes. For example, if a nominal attribute with name "costs" and possible nominal values "low", "moderate", and "high" is transformed, the result is a set of three binominal attributes "costs = low", "costs = moderate", and "costs = high". Only one of the values of each attribute is true for a specific example, the other values are false.

## Numerical2Binominal:

It converts all numerical attributes to binary ones. If the value of an attribute is between the specified minimal and maximal value, it becomes false, otherwise true. If the value is missing, the new value will be missing. The default boundaries are both set to 0, thus only 0.0 is mapped to false and all other values are mapped to true.

## FPGrowth:

This operator calculates all frequent items sets from a data set by building a FPTree data structure on the transaction data base. From this FPTree all frequent item set are derived. A major advantage of FPGrowth compared to Apriori is that it uses only 2 data scans and is therefore often applicable even on large data sets.

Given data set is only allowed to contain binominal attributes, i.e. nominal attributes with only two different values. That’s why I used the preprocessing operators in order to transform this given dataset. The necessary operators are the discretization operators for changing the value types of numerical attributes to nominal and the operator Nominal2Binominal for transforming nominal attributes into binominal / binary ones.

This operator has two basic working modes: finding at least the specified number of item sets with highest support without taking the min\_support into account (default) or finding all item sets with a support large than min\_support.

# References

1. Data Mining Concepts and Techniques by Jiawei Han and Micheline Kamber
2. Perceptual edge: Quantitative vs. Categorical Data: A difference Worth Knowing