Data Mining
Regras de Associação

Prof. Dr. Joaquim Assunção

DEPARTAMENTO DE COMPUTAÇÃO APLICADA
CENTRO DE TECNOLOGIA
UFSM
2019

www.inf.ufsm.br/~joaquim

Fair user agreement

Este material foi criado para a disciplina de Mineração de Dados - Centro de Tecnologia da UFSM.

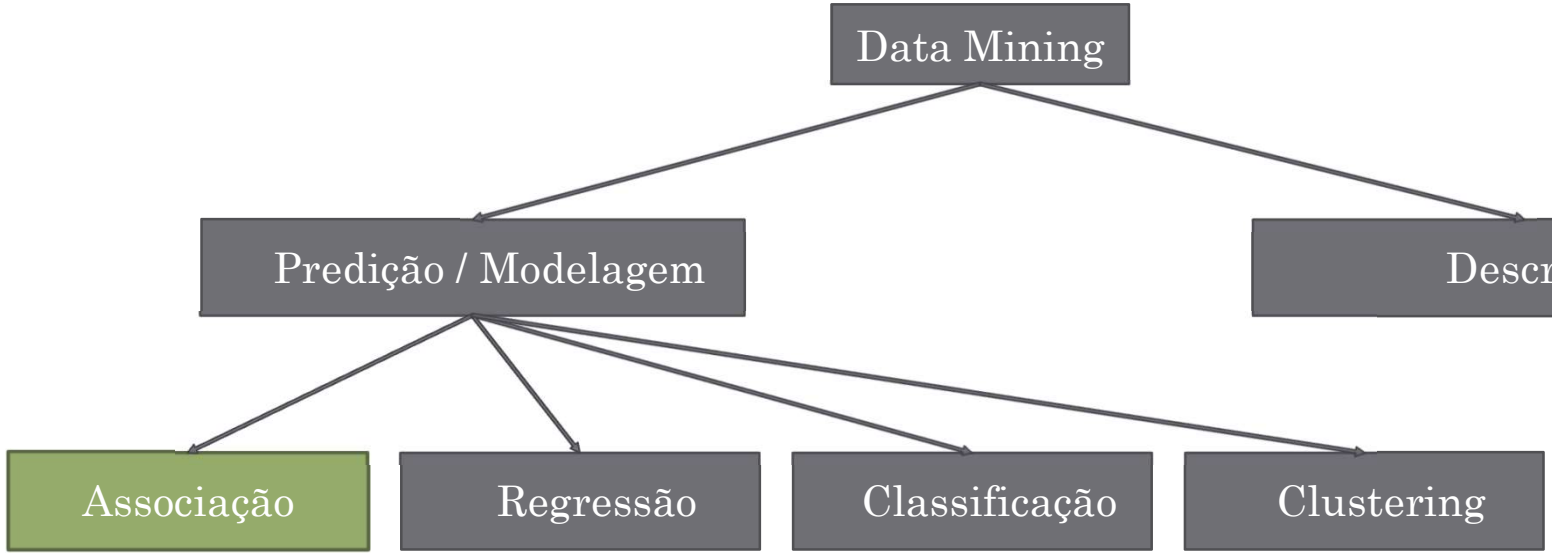
Você pode usar este material livremente*; porém, caso seja usado em outra instituição, **me envie um e-mail** avisando o nome da instituição e a disciplina.

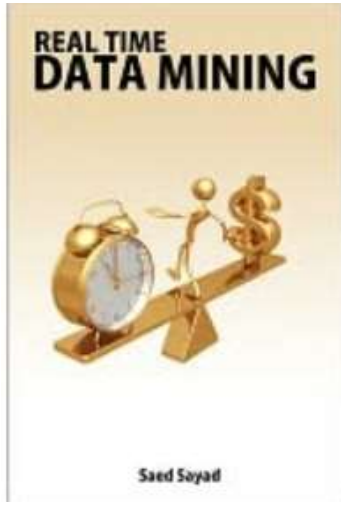
*Caso você queira usar algo desse material em alguma publicação, envie-me um e-mail com antecedência.

Prof. Dr. Joaquim Assunção.

joaquim@inf.ufsm.br

Mapa para Mineração de Dados*





***http://www.saedsayad.com/data_mining_map.htm**

Introdução

Em determinados casos uma ação está associada a outra com probabilidade *p.*

Exemplo: Quem compra queijo possivelmente comprará presunto.

Conjunto de dados

Os conjuntos de dados devem ser compostos por:

- Conjunto de transações **T;**
- Conjunto I com M atributos binários.
- Conjunto **T** é composto por N transações.

$$I = i_1, i_2, ..., i_M$$

$$T = t_1, t_2, ..., t_N$$

Problema

Dado um conjunto de transações, encontrar a probabilidade de ocorrência de um conjunto de itens dada a ocorrência de um outro conjunto de itens.

Mineração de dados no Wal-Mart

Dadas incontáveis notas de compras, quais itens tendem a ser comprados simultaneamente?

```
Compra 1: pão, ovos, queijo
Compra 2: arroz, feijão, sal
...
Compra N: pão, queijo, presunto
```

Mineração de dados no Wal-Mart

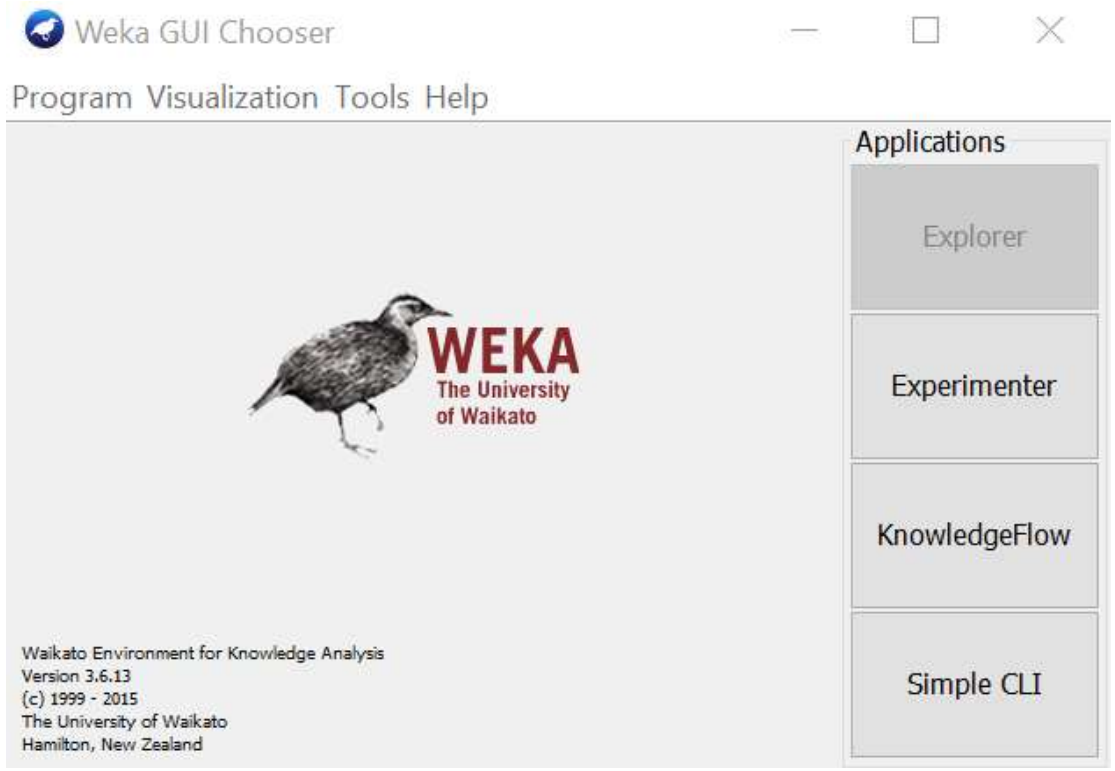
Lembre-se da definição: ***"A extração não trivial de informação implícita, previamente desconhecida e potencialmente útil"*** W. Frawley, G. Piatetsky-Shapiro e C. Matheus.

```
pão => queijo
...
pão, presunto => queijo, margarina
```

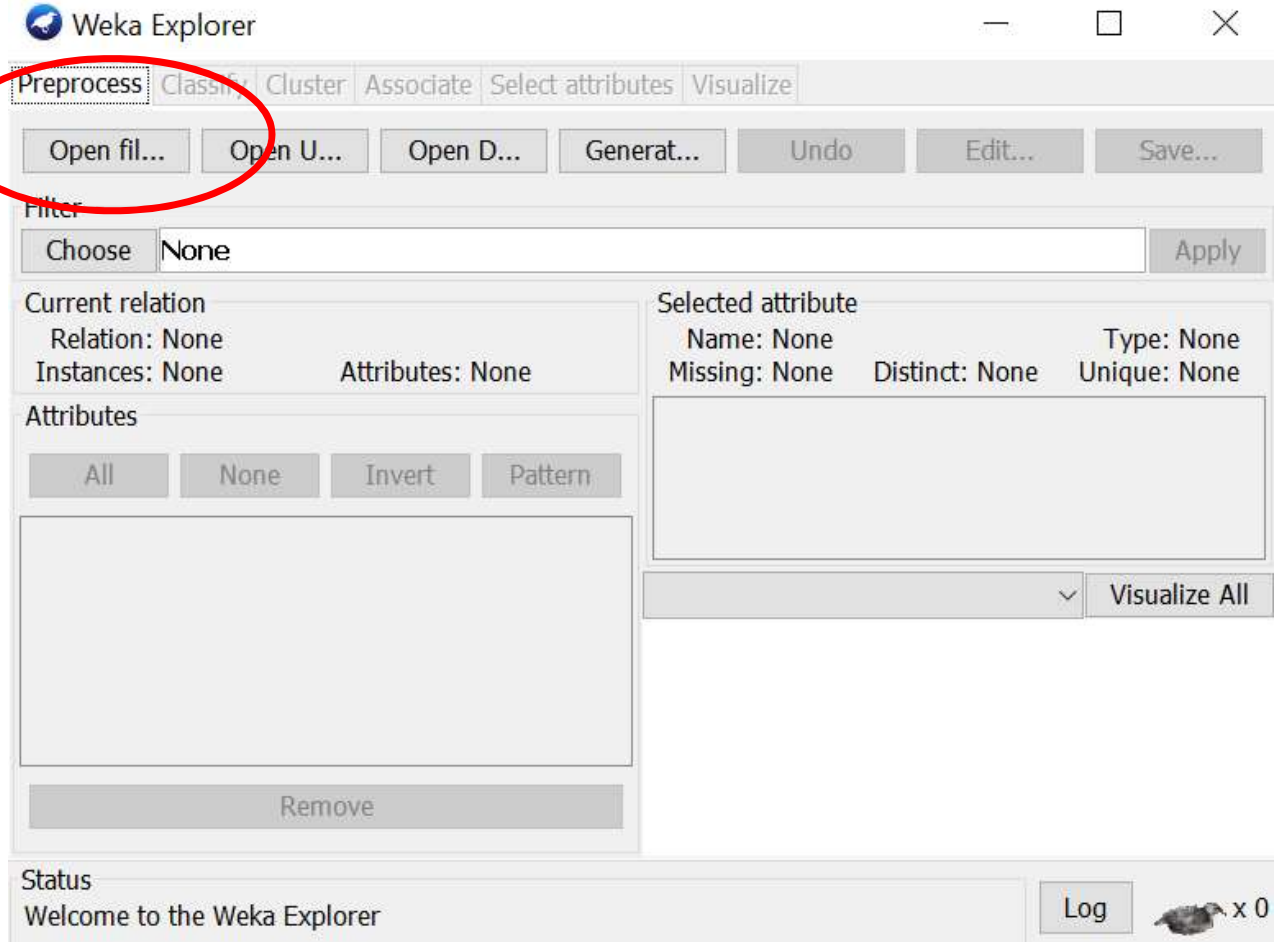
Mineração de dados no Wal-Mart

Fraldas e Cerveja.

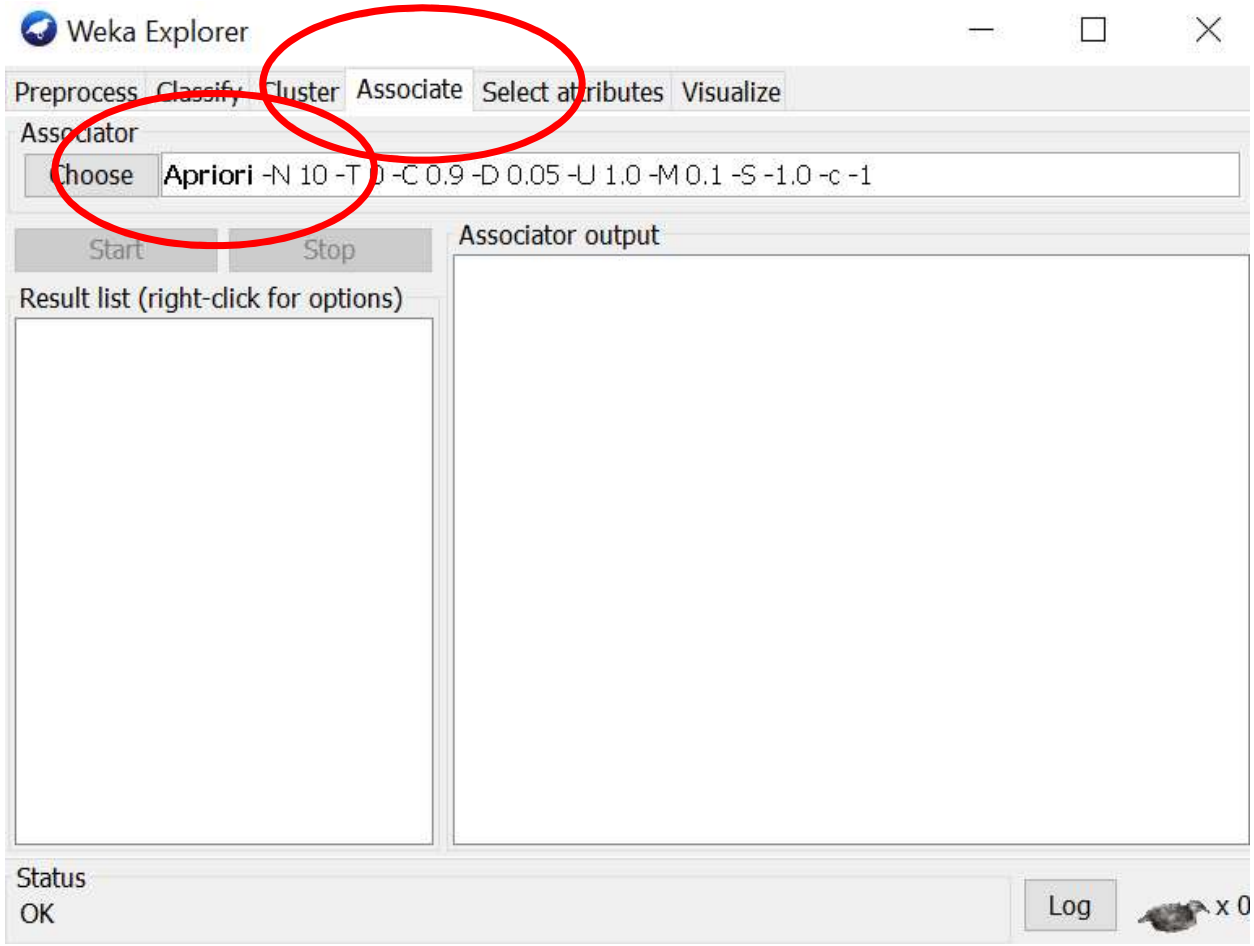
Weka


Weka GUI Chooser
Program Visualization Tools Help
WEKA
The University of Waikato
Applications
Explorer
Experimenter
KnowledgeFlow
Simple CLI
Waikato Environment for Knowledge Analysis
Version 3.6.13
(c) 1999 - 2015
The University of Waikato
Hamilton, New Zealand

Weka – *Open File*



Weka – *Associate*



Hands On!

- Abra o arquivo: `_ASSOC00.csv`

1. Rode o APRIORI. Encontrou algum padrão. Qual?
2. Altere o parâmetro da confiança para 0.6. O que aconteceu? Quantos padrões foram encontrados?

Suporte e Confiança

Suporte, ou cobertura, é o primeiro filtro para se obter as regras. Ao definirmos um suporte mínimo, o algoritmo verifica, logo de início, se o suporte é satisfeito. Somente se verdadeiro o item é adicionado ao subconjunto de itens frequentes.

$$suporte(A \Rightarrow B) = P(B|A) = \frac{A \cup B}{total(T)}$$

Suporte e Confiança

A confiança demonstra o quanto podemos confiar em uma implicação de acordo com o suporte dos termos.

A confiança é um bom indicador de quão forte é a associação entre os itens.

$$confianca(A \Rightarrow B) = P(B|A) =$$

$$\frac{suporte(A \Rightarrow B)}{suporte(A)} = \frac{A \cup B}{suporte(A)}$$

Entendendo ...

- Dado o seguinte conjunto de dados

A {0,1,0,1,0,0,1,0,0,0}
B {0,0,1,1,1,1,0,0,1,1}
C {1,0,0,0,1,1,1,1,0,0}
D {1,1,1,0,1,0,1,1,1,0}
E {0,0,1,1,0,1,1,0,1,1}
Objetivo <- c(0,1,1,1,0,0,1,0,1,1)

- Poderíamos gerar uma regra: `A --> Objetivo 0.3 1.00`

→0.3 é o suporte e 1 é a confiança.

→Todas as vezes que A apareceu, o objetivo foi cumprido (100%), mas isso só ocorreu em 3 dos 10 casos (30%)

Lift

Um segundo parâmetro é o lift que determina a dependência das regras. Quanto mais próximo de 0, mais dependente.

$$Lift(A \Rightarrow B) = (B|A) = \frac{A \cup B}{A * B}$$

Para a regra anterior temos $3/(3 * 6) = 1.666$

APRIORI com R

- Para termos um pouco mais de flexibilidade, podemos usar uma linguagem de programação.
- Podemos unir o pré-processamento com a execução de algoritmos.

Vamos a um exemplo...

Technical demo

- Rode o seguinte trecho de código:

```
a <- c(1,1,0,0,1)
b <- c(0,1,0,1,1)
c <- c(0,1,1,0,1)
goal <- c(1,0,1,0,1)
```

- No R, podemos usar `cbind()` "colar" colunas.

```
> myDF <- cbind(a,b,c,goal)
> myDF
     a b c goal
[1,] 1 0 0    1
[2,] 1 1 1    0
[3,] 0 0 1    1
[4,] 0 1 0    0
[5,] 1 1 1    1
```

Technical help

- Podemos usar a biblioteca `arules` para gerar regras de associação.

```
library(arules)
```

- Esta biblioteca possui uma implementação do APRIORI. Os escopo é:

```
apriori(data, parameter = NULL,
appearance = NULL, control = NULL)
```

Technical help

```
apriori(data, parameter = NULL,
appearance = NULL, control = NULL)
```

- Podemos usar a bibliotéca `arules` para gerar regras de associação.

```
library(arules)
```

- Dentro de `parameter` podemos usar o parâmetro target para especificar o tipo de associação a ser minerado. O APRIORI pode usar "`rules`". `parameter` sempre recebe uma lista. ...Use `list(target = "rules")`

Technical help

- A matriz de entrada também pode ser gerada através da função `read.transactions()`.
- `Por exemplo`, a linha a seguir carrega um *csv* contendo *n* colunas, onde `cols` é o número do ID da transação. `format="basket"` altera para o formato de entrada do algoritmo (APRIORI da biblioteca `arules`).

```
read.transactions(file="meuArquivo.csv", format="basket",sep=",",cols=1);
```

Hands On!

1. Rode o APRIORI para obter regras de associação no objeto myDF. A saída é um objeto que contém as especificações dos parâmetros e algumas estatísticas. Use `inspect()` para obter as regras dentro desse objeto.

Technical help

- A saída das regras possui 2 partes. A primeira (*lhs*) são os itens que implicam na saída (*rhs*).

Ex: `Pao, queijo` ➔ `presunto.` Pão e queijo é a parte `lhs` e presento é `rhs`

Controle a saída desejada usando os comandos `subset()` e `"%in%"`.

Ex: para o problema anterior podemos fazer:

```
Meu_subconjunto <- subset(regras, (lhs %in% "goal"))
```

Hands On!

2. Abra o arquivo "`_ASSOC00.csv`" no R. Use a biblioteca arules para gerar regras de associação.