# Data Mining

Daniela Costa
Ítalo Francyles
Sidney Melo
Thacyla Lima

# Data Mining

- Refere-se à **mineração** ou **descoberta de novas informações** em termos de padrões ou regras com base em **grande quantidade de dados**

- Não é bem integrada aos sistemas de gerenciamento de banco de dados

- Utiliza técnicas de áreas como **aprendizado de máquina, estatística, redes neurais e algoritmos genéticos**

# Mineração de dados

- Fase de descoberta de conhecimento
- Resultado da mineração pode descobrir os tipos de informação nova:
  - Regras de associação
  - Padrões sequenciais
  - Árvores de classificação
- Mineração de dados precisa ser precedida por uma preparação significativa de dados, antes de gerar informações úteis
- Resultados da mineração de dados podem ser informados em diversos formatos: listagens, saídas gráficas, tabelas de resumo ou visualizações

# Mineração de dados - Objetivos

- Executada com alguns **objetivos finais**:
  - **Previsão, identificação, classificação e otimização**

## Previsão

- **Análise de transações de compra para prever o que os consumidores comprarão sob certos descontos**
- **Quanto volume de vendas uma loja gerará em determinado período**

## Identificação

- **Intrusos tentando quebrar um sistema podem ser identificados pelos programas executados, arquivos acessados e tempo de CPU por sessão**
- **Autenticação é uma forma de identificação. Envolve uma comparação de parâmetros contra um bando de dados**

# Mineração de dados - Objetivos

## Classificação

- Pode particionar os dados de modo que diferentes classes ou categorias possam ser identificadas com base em combinações de parâmetros
- Os clientes em um supermercado podem ser categorizados em compradores que buscam desconto, com pressa, regulares leais e ligados a marcas conhecidas

## Otimização

- Otimizar o uso de recursos limitados, como tempo, espaço, dinheiro ou materiais
- Maximizar variáveis de saída como vendas ou lucros sob determinado conjunto de restrições

# Tipos de conhecimento descobertos durante a mineração de dados

- A mineração de dados enfoca o **conhecimento indutivo**
- Conhecimento descoberto durante a mineração de dados é descrito:
  - **Regras de associação** - correlacionam a presença de um itemset com outra faixa de valores para um conjunto de variáveis diverso
  - **Hierarquias de classificação** - O objetivo é trabalhar partindo de um conjunto existente de eventos ou transações para criar uma hierarquia de classes.
  - **Padrões sequenciais** - Uma sequência de ações ou eventos é buscada.

# Tipos de conhecimento descobertos durante a mineração de dados

- **Padrões dentro de série temporal** - As similaridades podem ser detectadas dentro de posições de uma série temporal de dados, que é uma sequência de dados tomados em intervalos regulares, como vendas diárias ou preços de ações de fechamento diário.
- **Agrupamento** - Determinada população de eventos ou itens pode ser particionada em conjuntos de elementos ‘semelhantes’.

# Regras de Associação

- Uma regra de associação tem a forma X => Y, onde X = {x1,x2, .. . ,xn}, e Y = {y1,y2, …, yn};
- X e Y = conjuntos de itens;
- $x_i$ e $j_i$ sendo itens distintos para todo i e y;
- X U Y (união) = itemset. Essa associação indica que, se um cliente compra X, ele provavelmente também comprará Y.
- LHS => RHS igual à X => Y
- Para que uma regra de associação seja de interesse, é necessário que a mesma satisfaça alguma medida de interesse (suporte e confiança).

# Medidas de Interesse

- Suporte ou Prevalência da Regra: dada uma regra X => Y, a sua medida de suporte representa a frequência com que um itemset específico ocorre no banco de dados. Se essa frequência for baixa, significa que não podemos afirmar que os itens no conjunto X U Y ocorrem juntos, a relevância para essa regra é baixa.
- Confiança ou Força da Regra: Representa a probabilidade de que os itens de Y sejam comprados desde que os itens de X também sejam. A confiança de uma regra X => Y é calculada como o suporte(X U Y)/ suporte(X).

# Medidas de Interesse

(1) leite => suco;

(2) pão => suco;

Suporte (1) = 2/4 = 50%; Confiança (1) = ⅔ = 66.7%

Suporte (2) = ¼ = 25 %; Confiança (2) = ½ = 50%

| Id_transação | Hora | Itens_comprados |
|---|---|---|
| 101 | 6:35 | leite, pão, biscoito, suco |
| 792 | 7:38 | leite, suco |
| 1130 | 8:05 | leite, ovos |
| 1735 | 8:40 | pão, biscoito, café |

Exemplo de transações

# Regras de Associação

- A meta na mineração de regras de associação é gerar todas as possíveis regras que excedam as especificações dos usuários garantindo suporte e confiança acima do limite definido.
- Propriedades dos algoritmos para geração das regras de associação:
  - Propriedade do fechamento por baixo: um subconjunto de um itemset grande precisa também ser grande (isto é, cada subconjunto de um conjunto de itens grande excede o suporte mínimo exigido).
  - Antimonotonicidade: um superconjunto de um itemset pequeno é também pequeno (implicando que ele não tem suporte suficiente).

# Algoritmo Apriori

**Entrada:** banco de dados de $m$ transações, $D$, e um suporte mínimo, *mins*, representado como uma fração de $m$.

**Saída:** itemsets frequentes, $L_1, L_2, ..., L_k$

**Início** /* etapas ou instruções são numeradas para aumentar a legibilidade */

1. Calcule suporte($i_j$) = conta($i_j$)/$m$ para cada item individual, $i_1, i_2, ..., i_n$ fazendo a varredura do banco de dados uma vez e contando o número de transações em que o item $i_j$ aparece (ou seja, conta($i_j$));
2. O 1-itemset frequente candidato, $C_1$, será o itemset $i_1, i_2, ..., i_n$;
3. O subconjunto de itens contendo $i_j$ de $C_1$ onde suporte($i_j$) >= mins torna-se o 1-itemset frequente, $L_1$;
4. $k = 1$;
   termina = false;
   **repita**
   1. $L_{k+1}$ = ;
   2. Crie o ($k$+1)-itemset frequente candidato, $C_{k+1}$, combinando membros de $L_k$ que têm $k$–1 itens em comum (isso forma os ($k$+1)--itemsets frequentes candidatos ao estender seletivamente os $k$-itemsets frequentes em um item);
   3. Além disso, apenas considere como elementos de $C_{k+1}$ aqueles $k$+1 itens tais que cada subconjunto de tamanho $k$ apareça em $L_k$;
   4. Faça a varredura do banco de dados uma vez e calcule o suporte para cada membro de $C_{k+1}$; se o suporte para um membro de $C_{k+1}$ >= mins, então acrescente o membro em $L_{k+1}$;
   5. Se $L_{k+1}$ for vazio, então termina = true, se não, $k = k + 1$;

   **até que termina;**

**Fim;**

# Algoritmo Apriori

**Entrada: mins = 0.5**
**Início**
$C_1$ = {leite, pão, suco,biscoito, ovos, café}
$L_1$ = {leite, pão, suco, biscoito}
k = 1
**1º loop**
$C_2$ = ~~{leite, pão}~~, {leite, suco}, ~~{leite, biscoito}~~, ~~{pão, suco}~~, {pão, biscoito}, ~~{suco, biscoito}~~
$L_2$ = {leite, suco}, {pão, biscoito}

**2º loop**
$C_3$ = {~~leite, suco, pão}~~, ~~{leite, suco, biscoito}~~, ~~{leite, pão, biscoito}~~, ~~{suco, pão, biscoito}~~
$L_3$ = vazio;
termina = true;

**Saída: $L_1$ = {leite, pão, suco, biscoito}**
**$L_2$ = {leite, suco}, {pão, biscoito}**

| Id_transação | Hora | Itens_comprados |
|---|---|---|
| 101 | 6:35 | leite, pão, biscoito, suco |
| 792 | 7:38 | leite, suco |
| 1130 | 8:05 | leite, ovos |
| 1735 | 8:40 | pão, biscoito, café |

# Algoritmo de Amostragem

- Neste algoritmo, é selecionada da base de dados uma pequena amostra, sobre a qual determinam-se os itemsets frequentes da mesma. Se os itemsets frequentes da amostra formarem um superconjunto dos itemsets frequentes para o banco de dados inteiro, então os itemsets frequentes de fato são determinados fazendo a varredura do restante do banco de dados a fim de calcular os valores de suporte exatos para os itemsets do superperconjunto.
- Borda negativa: contém os itemsets mais próximos que também poderiam ser frequentes. Por exemplo, considerando um conjunto X que não está contido nos itemsets frequentes. Se todos os subconjuntos de X estiverem contidos no conjunto de itemsets frequentes, então X estaria na borda negativa.

# Algoritmo de Amostragem

Exemplo:

I = {A, B, C, D}

Itemsets frequentes: S = {{A}, {B},{C}, {D}, {AB} {AC},{BC}, {AD}, {CD}, {ABC}}.

Borda negativa = {{BD}, {ACD}}.

# Algoritmo de árvore de padrão frequente (FP)

- Estrutura de dados utilizada para minerar conjuntos frequentes
- Menor custo que algoritmo A Priori
- Suporte: quantidade de transações que contém o itemset.

# Algoritmo de árvore de padrão frequente (FP)

- 1ª varredura
  - São encontrados os 1-itemsets frequentes e seus respectivos suportes.
  - São organizados em ordem decrescente em uma tabela de cabeçalho contendo um link (ponteiro que será usado para associar cada item à nós na árvore FP).
  - Criação da árvore FP com rótulo nulo na raiz.

| Itens_comprados |
| --- |
| leite, pão, biscoito, suco |
| leite, suco |
| leite, ovos |
| pão, biscoito, café |

| Item | Suporte | Link |
| --- | --- | --- |
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |

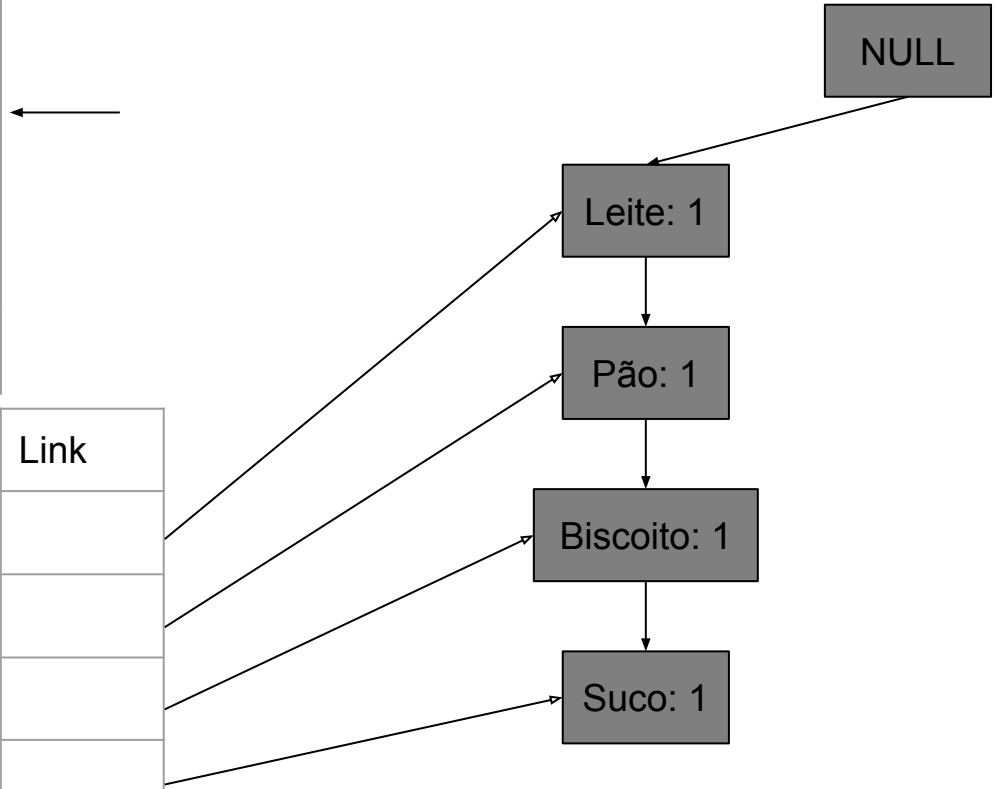
# Algoritmo de árvore de padrão frequente (FP)

- 2ª varredura
  - Para cada transação T, os 1-itemsets frequentes de T são inseridos ordenadamente na árvore FP a partir da raiz, seguindo o algoritmo:
    - Se o nó atual N possui um filho tal que o item = cabeça de T (primeiro elemento de T)
      - incremente o contador associado ao nó N em 1.
    - Senão
      - Crie outro nó M com contagem 1, vincule M a N e à tabela de cabeçalho do item.
    - Se a cauda não for vazia, repita a etapa anterior usando como lista ordenada apenas a cauda de T.

# Algoritmo de árvore de padrão frequente (FP)

| Transações T |
| --- |
| leite, pão, biscoito, suco |
| leite, suco |
| leite |
| pão, biscoito |

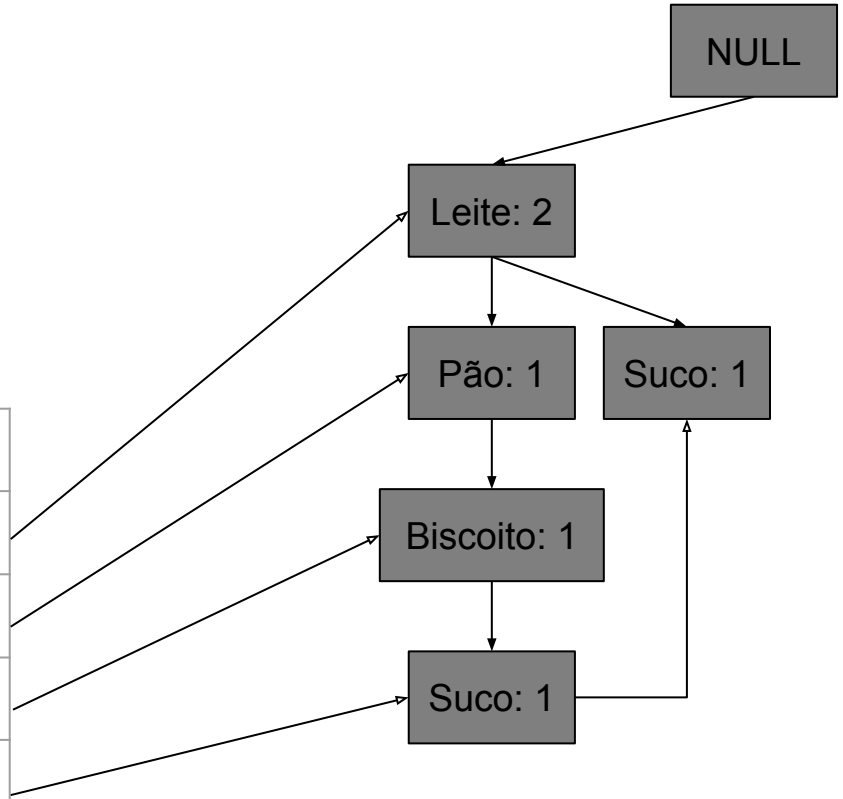
| Item | Suporte | Link |
| --- | --- | --- |
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de árvore de padrão frequente (FP)

| Transações T |
| --- |
| leite, pão, biscoito, suco |
| leite, suco |
| leite |
| pão, biscoito |

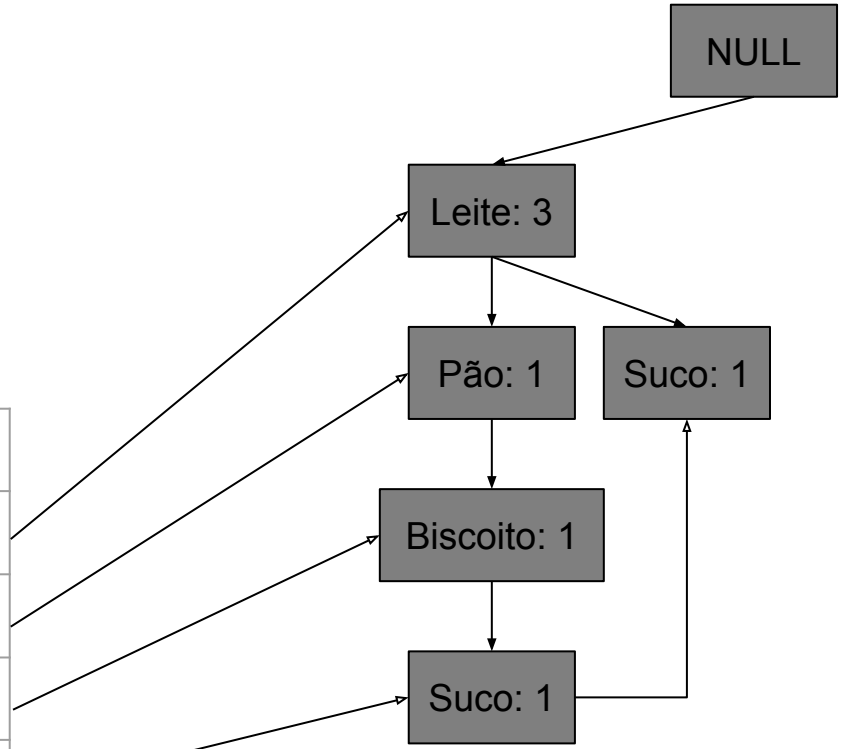
| Item | Suporte | Link |
| --- | --- | --- |
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de árvore de padrão frequente (FP)

| Transações T |
| --- |
| leite, pão, biscoito, suco |
| leite, suco |
| leite |
| pão, biscoito |

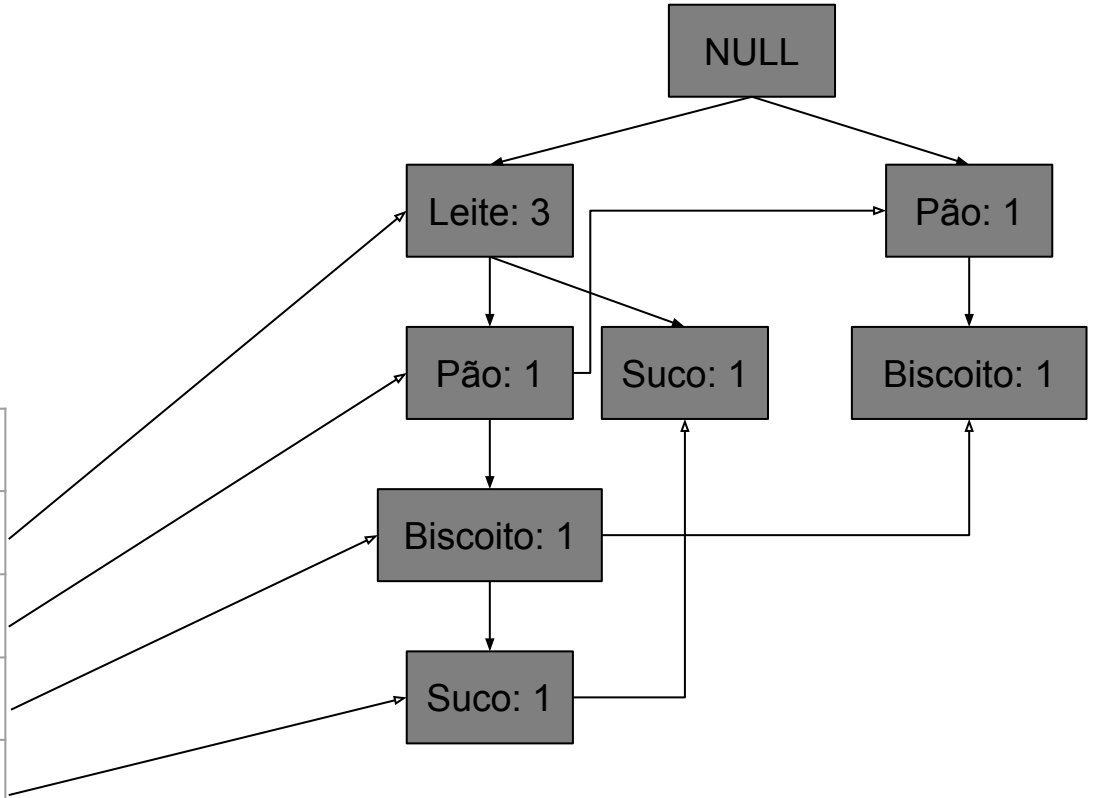
| Item | Suporte | Link |
| --- | --- | --- |
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de árvore de padrão frequente (FP)

| Transações T |
| --- |
| leite, pão, biscoito, suco |
| leite, suco |
| leite |
| pão, biscoito |

| Item | Suporte | Link |
| --- | --- | --- |
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de crescimento FP

- Base de padrão condicional
  - Conjunto de caminhos de prefixo para um 1-itemset frequente (sufixo)
- Arvore FP condicional
  - Construída a partir de padrões na base de padrão condicional

# Algoritmo de crescimento FP

Input: Árvore FP, suporte_mínimo e alfa

Begin

se *árvore* contém um único caminho *P* então

para cada combinação *beta* dos nós no caminho

gera padrão (*beta* U *alfa*) com suporte = suporte mínimo do padrão

# Algoritmo de crescimento FP

senão

para cada item *i* no cabeçalho da árvore faça

gera padrão *beta* = (*i* U *alfa*) com suporte = i.suporte;

constrói base de padrão condicional de beta;

constrói árvore FP condicional de *beta, árvore_beta*;

se *árvore_beta* não está vazia

crescimento-FP(árvore_beta, beta);

fim_para;

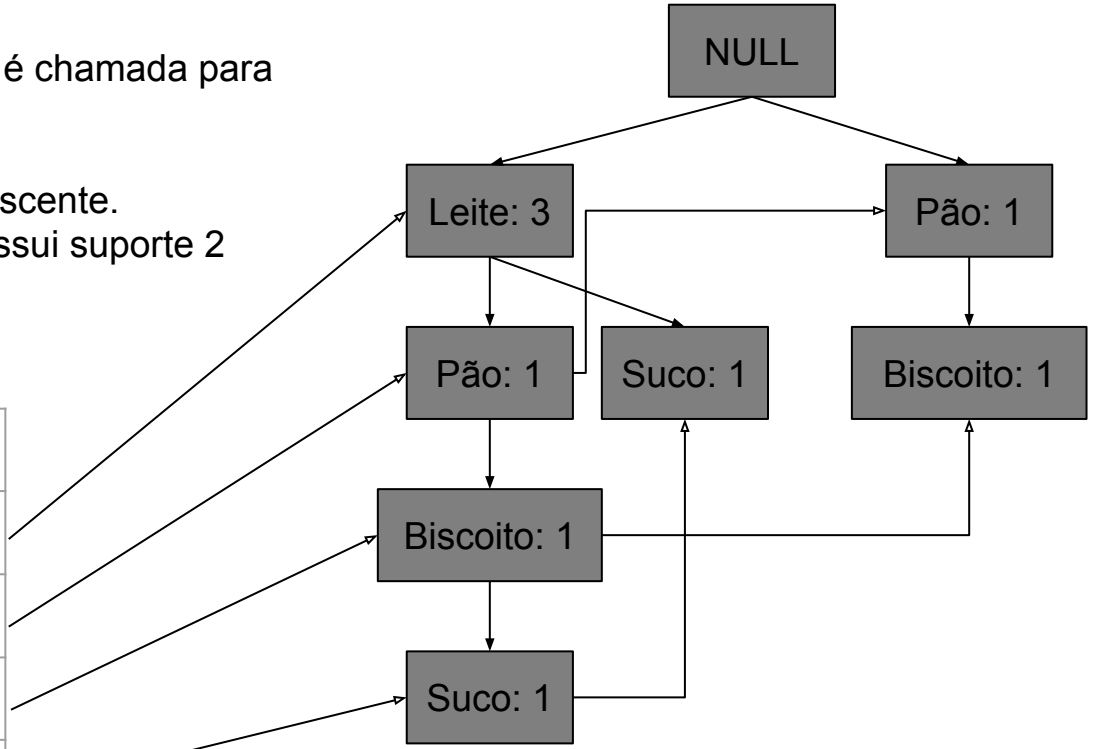
Fim;

# Algoritmo de crescimento FP

A função procedimento-crescimento é chamada para FP e NULL (*a*).

Os itens são visitados em ordem crescente.
O primeiro é, portanto, suco, que possui suporte 2

| Item | Suporte | Link |
|---|---|---|
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |

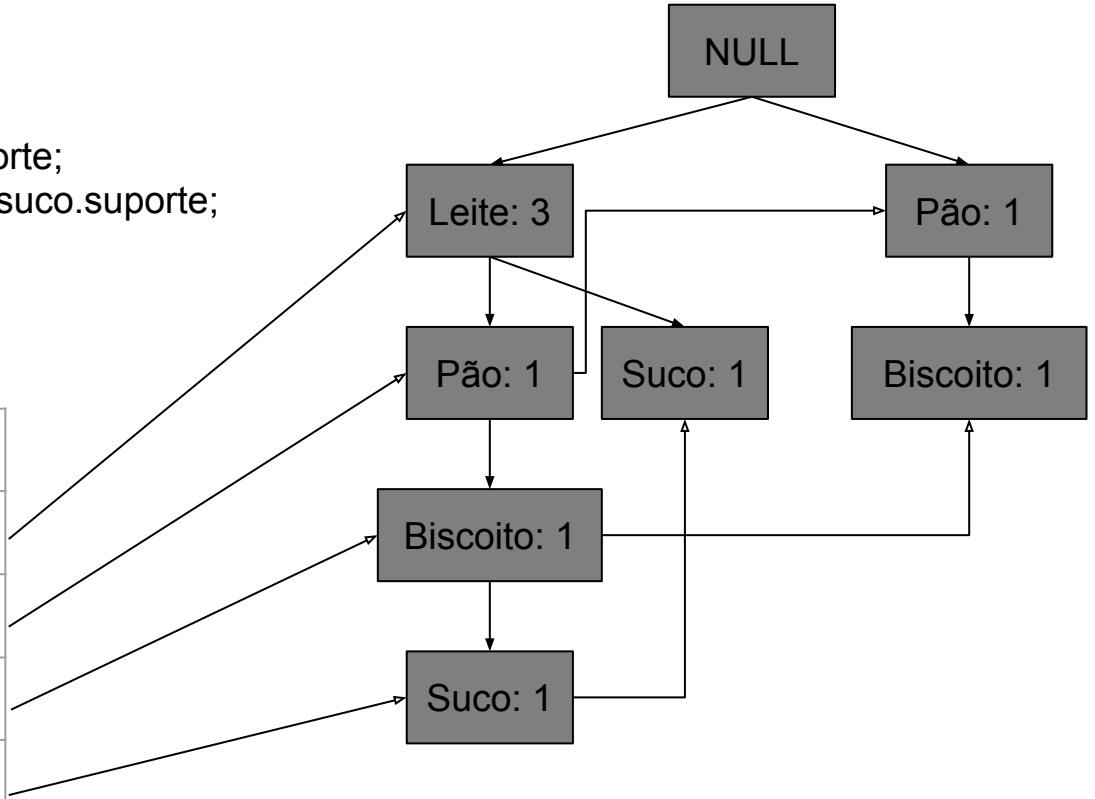


# Algoritmo de crescimento FP

i = suco; alfa = NULL; suporte = 2;

beta = (i U alfa) com suporte = i.suporte;
beta = (suco U NULL) com suporte=suco.suporte;
beta = (suco) com suporte = 2;

| Item | Suporte | Link |
|---|---|---|
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de crescimento FP

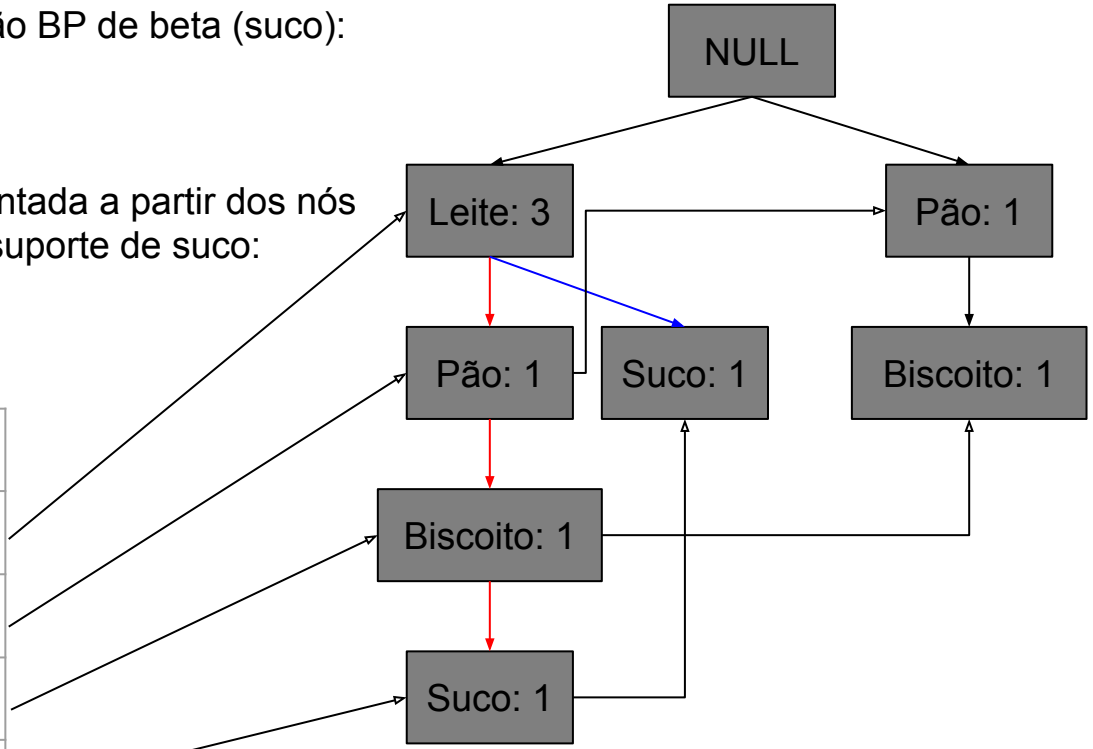
Agora, construímos a base de padrão BP de beta (suco):

{leite, pão, biscoito : 1} e {leite : 1}

A árvore FP condicional é então montada a partir dos nós cujo suporte é superior ou igual ao suporte de suco:

*arvore_FP_suco* = {leite : 2}

| Item | Suporte | Link |
|---|---|---|
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |

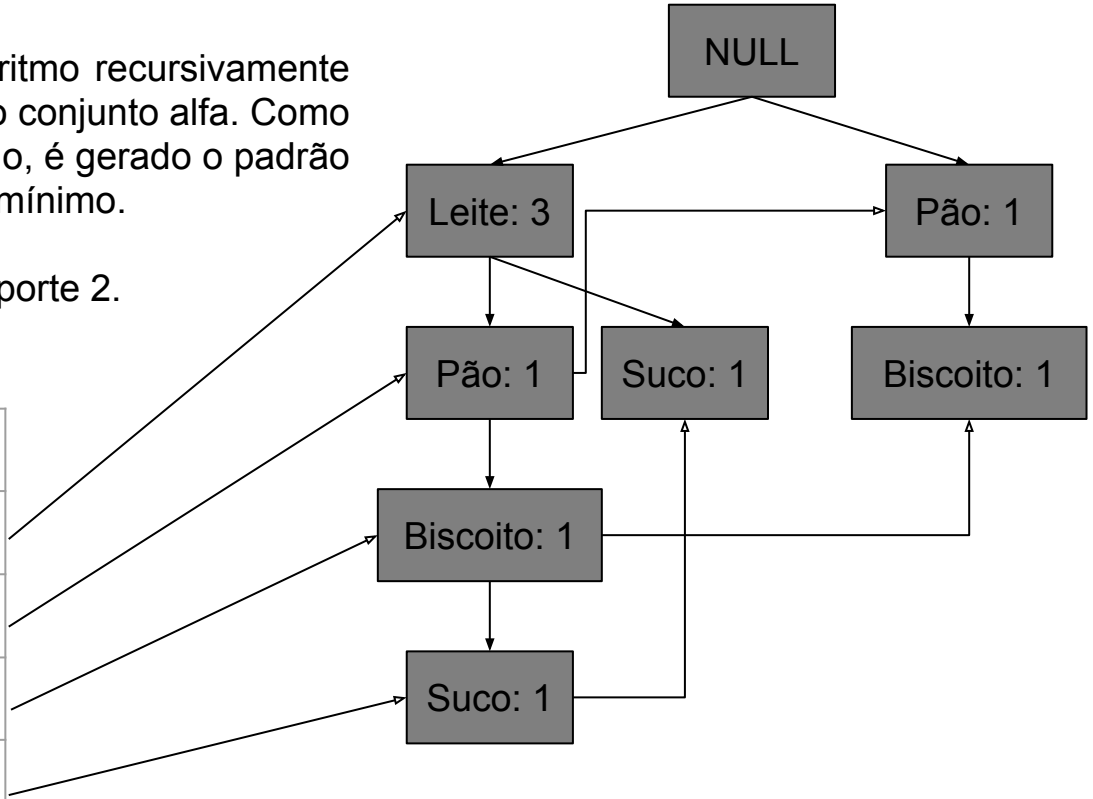


# Algoritmo de crescimento FP

Agora, é necessário chamar o algoritmo recursivamente para a *arvore_FP_suco* e *suco* como conjunto alfa. Como essa árvore possui um único caminho, é gerado o padrão (*beta* U *alfa*) com suporte = suporte mínimo.

Ou seja: padrão {leite, suco} com suporte 2.

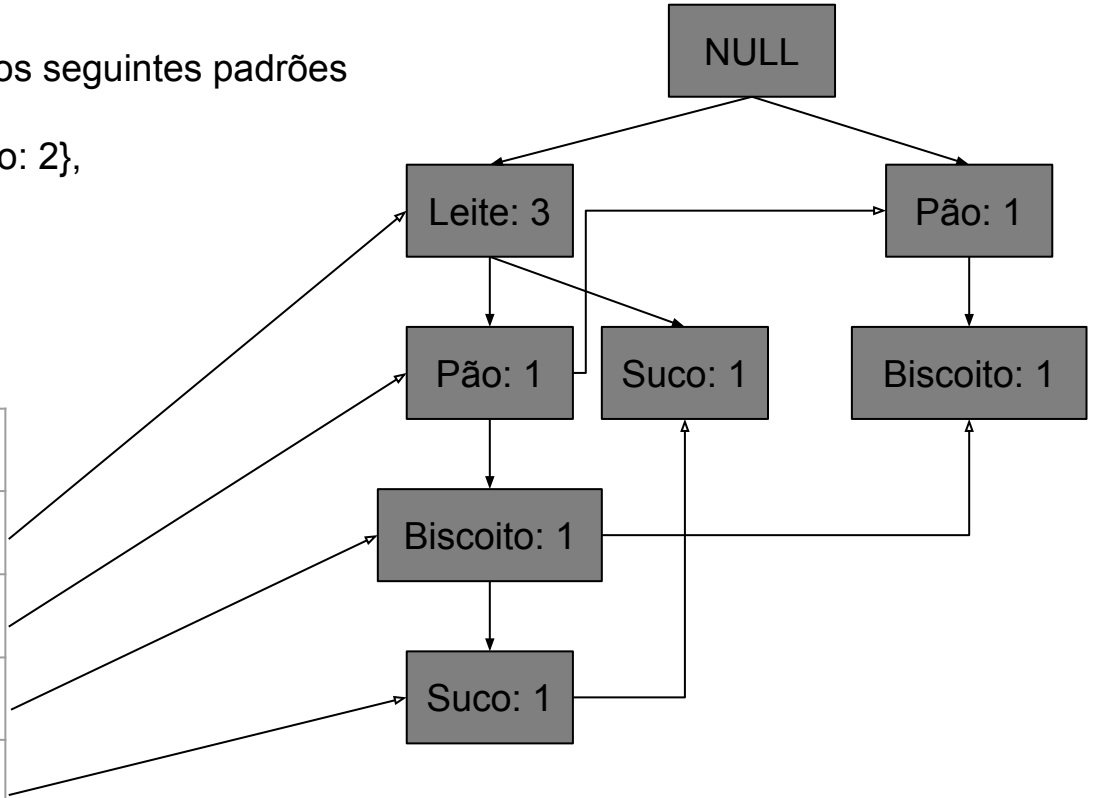
| Item | Suporte | Link |
|---|---|---|
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Algoritmo de crescimento FP

Ao final do algoritmo, são formados os seguintes padrões frequentes:
{{leite: 3}, {pão: 2}, {biscoito: 2}, {suco: 2},
{leite, suco : 2}, {pão, biscoito: 2}}

| Item | Suporte | Link |
|---|---|---|
| Leite | 3 | |
| Pão | 2 | |
| Biscoito | 2 | |
| Suco | 2 | |



# Classificação e Aprendizado Supervisionado

- Dado um conjunto de dados especificados, estes são previamente divididos em subconjuntos seguindo certos critérios. Cada subconjunto é chamado de classe.
- Aprendizado Supervisionado:
  - Dado um modelo de dados previamente classificados, este será útil para classificar novos dados de entrada.
  - Fases
    - Modelagem
    - Treinamento
    - Testes

# Classificação e Aprendizado Supervisionado

- Aprendizado Supervisionado: Algoritmo KNN (K-nearest neightboors)
    - Consiste em classificar um objeto com base na proximidade deste dentro do espaço de características com outros previamente classificados.
    - Vantagem: Simplicidade
    - São necessários:
        - um conjunto de dados de treinamento,
        - definir uma métrica para calcular a distância entre os exemplos de treinamento
        - indicar o valor de K (o número de vizinhos mais próximos)
- Exemplo: Uma aplicação que prediz se um indivíduo com 48 anos tem condições de pedir $142.000 ao banco como empréstimo

# Classificação e Aprendizado Supervisionado

| ID | Idade (X) | Empréstimo ($) (Y) | Classificação | Distância |
|---|---|---|---|---|
| 1 | 25 | $40.000 | Não | 102000 |
| 2 | 35 | $60.000 | Não | 82000 |
| 3 | 45 | $80.000 | Não | 62000 |
| 4 | 20 | $20.000 | Não | 122000 |
| 5 | 35 | $120.000 | Não | 22000 |
| 6 | 23 | $95.000 | Sim | 47000 |
| 7 | 40 | $62.000 | Sim | 80000 |
| 8 | 60 | $100.000 | Sim | 42000 |
| 9 | 48 | $220.000 | Sim | 78000 |
| 10 | 33 | $150.000 | Sim | 8000 |
| 11 | 48 | $142.000 | ? | |

# Classificação e Aprendizado Supervisionado

- Métrica Utilizada: Distância Euclidiana
- K = 3
- Conclusões:
  - Os vizinhos mais próximos do indivíduo 11 são os indivíduos 10, 5 e 8, cujas distâncias são, respectivamente, 8000, 22000 e 42000.
  - Dois pertencem à classe daqueles que podem retirar empréstimos.
  - Conclui-se então que este indivíduo 11 de 48 anos está habilitado a pedir um empréstimo de $142.000 ao banco.

# Agrupamento e Aprendizado Não Supervisionado

- Aprendizado não supervisionado: Dados são divididos sem uma amostra de treinamento.
- Agrupamento:
  - Agrupa-se dados seguindo padrões de características.
  - Os registros de um mesmo grupo são similares e que todos os grupos são disjuntos.
  - Pode ser usada para reduzir a dimensão de um conjunto de dados, reduzindo uma ampla gama de objetos à informação do centro do seu conjunto.
  - Caracteriza-se por ser bastante complexa em termos computacionais e para ser realizada com eficácia, é preciso escolher uma heurística hábil.

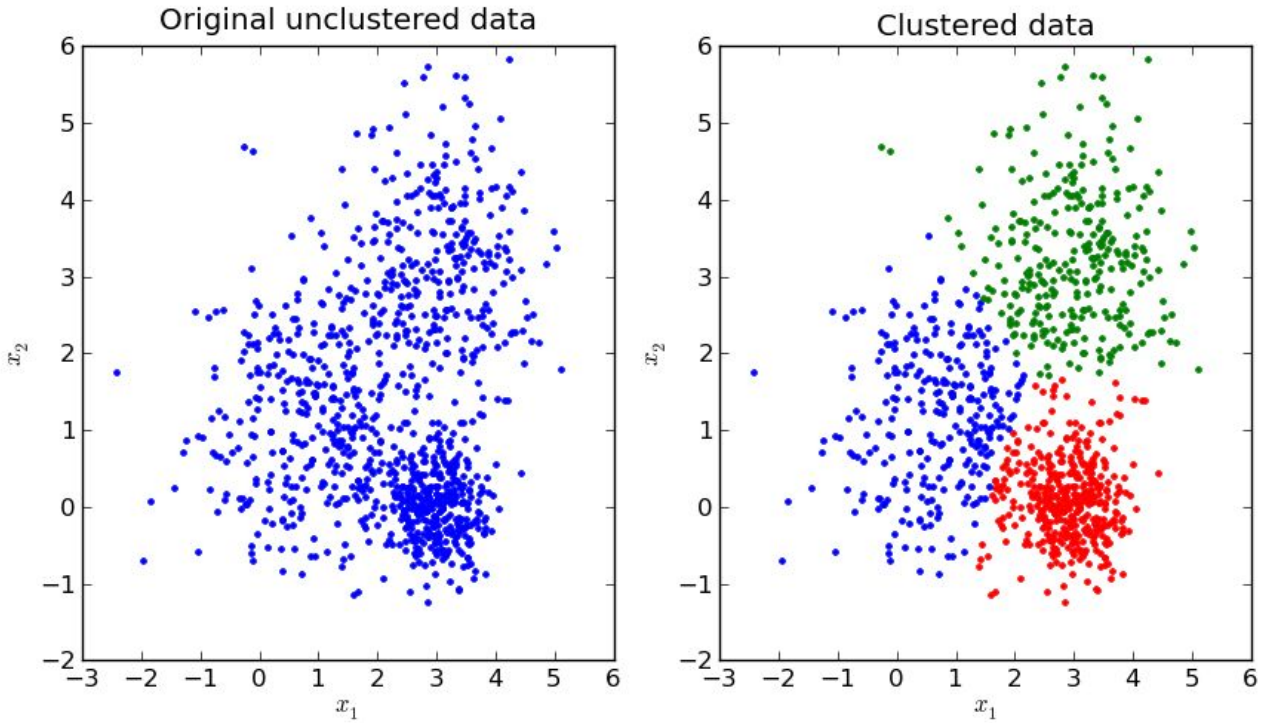
# Agrupamento e Aprendizado Não Supervisionado

- Heurística de Agrupamento: K-means
  - Heurística de agrupamento não hierárquico
  - Busca minimizar a distância dos elementos a um conjunto de k centros dado por X = {x1,x2,...,xk} de forma iterativa.
  - A função a ser minimizada então, é dada por:
  - funcao

# Agrupamento e Aprendizado Não Supervisionado

- Heurística de Agrupamento: K-means
  - Passos:
    i. Escolher k distintos valores para centros dos grupos (possivelmente, de forma aleatória)
    ii. Associar cada ponto ao centro mais próximo
    iii. Recalcular o centro de cada grupo
    iv. Repetir os passos 2-3 até nenhum elemento mudar de grupo
  - Este algoritmo depende do parâmetro K

# Agrupamento e Aprendizado Não Supervisionado



# Aplicações da Mineração de Dados

- Nível estratégico empresarial (tomada de decisão)
- Marketing (perfis de clientes, recomendação, análise de sentimento)
- Finanças (análise de crédito, performance de investimentos)
- Saúde (diagnóstico)