INF0613 – Aprendizado de Máquina Não Supervisionado

Trabalho 1 - Regras de Associação

Kaleb Roncatti de Souza Nelson Gomes Brasil Junior

Neste primeiro trabalho vamos minerar Regras de Associação em uma base de dados que contém as vendas de uma padaria. A base de dados está disponível na página da disciplina no Moodle (arquivo `bakery.csv`).

Atividade 0 – Configurando o ambiente

Antes de começar a implementação do seu trabalho configure o *workspace* e importe todos os pacotes:

```
# Adicione os demais pacotes usados
# Bibliotecas usadas neste trabalho:
library(arules)

# Configurando ambiente de trabalho:
# setwd("")
```

Atividade 1 – Análise Exploratória da Base de Dados (*3,0 pts*)

Dado um caminho para uma base de dados, leia as transações e faça uma análise Exploratória sobre elas. Use as funções `summary`, `inspect` e `itemFrequencyPlot`. Na função `inspect` limite sua análise às 10 primeiras transações e na função `itemFrequencyPlot` gere um gráfico com a frequência relativa dos 30 itens mais frequentes.

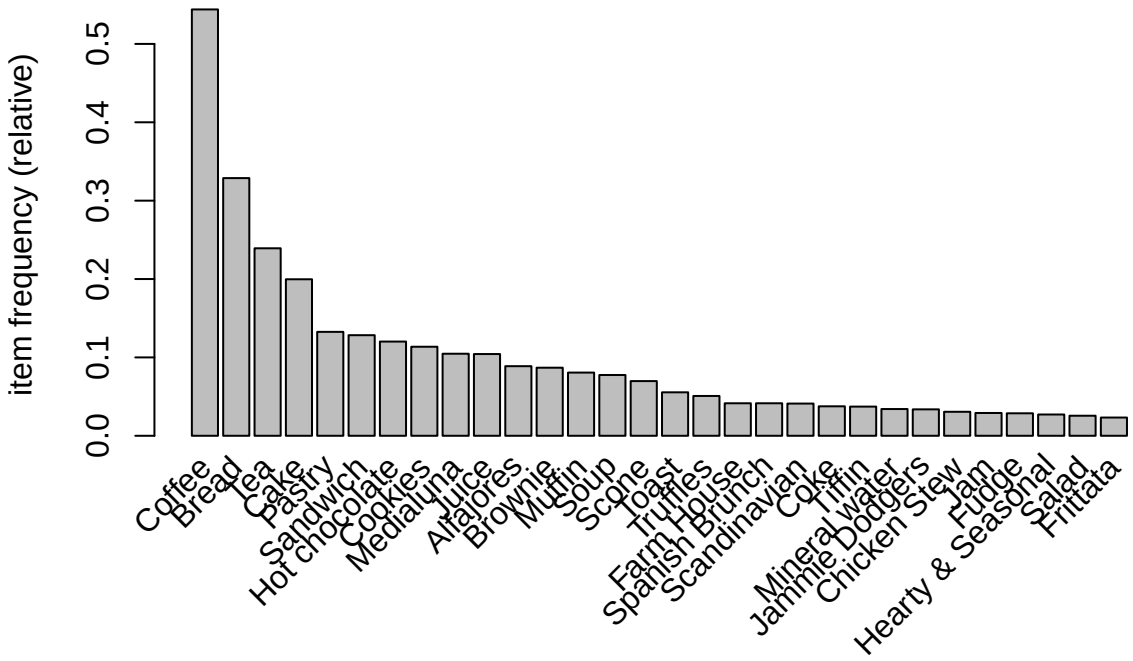
```
# Ler transações
bakery <- read.transactions(file.choose(),
                            format="basket",
                            sep=",")

# Visualizando transações
inspect(head(bakery, 10))
```

```
##      items
## [1]  {Coffee, Vegan mincepie}
## [2]  {Farm House, Muffin, Tea}
## [3]  {Bread, Ellas Kitchen Pouches, Jam, Juice, Muffin}
## [4]  {Bread, Juice, Salad, Sandwich}
## [5]  {Cake, Coffee, Sandwich, Smoothies, Soup}
## [6]  {Bread, Medialuna}
## [7]  {Chocolates, Coffee, Tea}
## [8]  {Alfajores, Brownie, Medialuna}
## [9]  {Alfajores, Coffee, Fudge}
## [10] {Bread, Pastry}
```

```r
# Sumário da base
summary(bakery)
```

```
## transactions as itemMatrix in sparse format with
##  2579 rows (elements/itemsets/transactions) and
##  91 columns (items) and a density of 0.0352
##
## most frequent items:
##  Coffee   Bread     Tea    Cake  Pastry (Other)
##    1403     848     617     515     342    4532
##
## element (itemset/transaction) length distribution:
## sizes
##    1    2    3    4    5    6    7    8    9   10
##   20  664 1041  591  189   52   15    4    2    1
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     2.0     3.0     3.2     4.0    10.0
##
## includes extended item information - examples:
##                     labels
## 1 Afternoon with the baker
## 2               Alfajores
## 3          Argentina Night
```

```r
# Analisando a frequência dos itens
itemFrequencyPlot(bakery, type = "relative", topN=30)
```



Análise

a) Descreva a base de dados discutindo os resultados das funções acima.

Resposta: Os itens acima representam itens de padaria no estilo "lista de compras". Basicamente, obtemos uma relação entre itens adquiridos/vendidos baseado em uma frequência na qual um determinado item aparece, e também nos interessa encontrar relações entre diferentes itens numa mesma compra/transação. A base de dados apresenta então transações para as quais cada uma pode apresentar de 1 até 10 itens de padaria. As funções que aplicamos acima, nos ajudam a:

- Ler e inspecionar os dados utilizando a biblioteca arule (agrupando os itens por chaves.
- Obter um sumário, com informações a respeito da distribuição/frequência destes itens.
- Obter uma visualização da contagem/frequência para cada um dos itens.
- Os itens mais comuns do conjunto em ordem decrescente de importância foram: Coffee, Bread, Tea, Cake e Pastry.
- Obtivemos uma densidade de 3.52%, significando que existem 3.52% de células não-zero na matriz.
- Temos 2579 transações (rows) e 91 tipos de itens (columns), resultando em 234689 células, das quais, pelo item anterior, 8261 são nulas.
- Na média, tivemos 3.2 itens por transação, e no máximo 10 itens foram adquiridos concomitantemente.

b) Ao gerarmos o gráfico de frequências, temos uma representação visual de uma informação já presente no resultado da função `summary`. Contudo, esse gráfico nos dá uma visão mais ampla da base. Assim podemos ver a frequência de outros itens em relação aos 10 mais frequentes. Quais informações podemos obter a partir desse gráfico (e da análise anterior) para nos ajudar na extração de regras de associação com o algoritmo `apriori`? Isto é, como a frequência dos itens pode afetar os parâmetros de configuração do algoritmo `apriori`?

Resposta: Um dos nossos principais objetivos com o conjunto de transações é gerar regras de associação. No entanto, dado `m` itens únicos num dado problema/conjunto de itens de compra, o número total de conjuntos de itens escala com 2^m e o número total de regras de associação, como visto em aula, se mostra como um valor extremamente elevado ($3^m - 2^{m+1} + 1$). Desta maneira, para não realizarmos buscas completas, um problema que se tornaria inviável de se realizar computacionalmente, desejamos reduzir os conjuntos de busca, e uma das maneiras de fazer isto é utilizando-se o algoritmo `apriori`, que possue como princípio o foco em conjuntos de itens que são frequentes através da propriedade anti-monotônica do suporte. Através do gráfico de frequência e do sumário que realizamos no exercício, conseguimos ter uma ideia de quais itens gerarão os valores mais relevantes de suporte. Quando queremos encontrar regras de associação, desejamos valores altos de suporte (e também confiança). O algoritmo `apriori`, começa por agrupar conjuntos de itens de tamanho $k = 1$ elevando-se o k até não conseguirmos encontrar mais regras cujo valor seja acima do valor de `threshold` mínimo de suporte e confiança, ou seja, os valores relevantes obtidos no gráfico de frequência e no sumário já representam os suportes para os itens **um a um**, e nos dá uma noção de quais itens terão maior relevância na geração das regras. Ademais, tal valor também pode nos ajudar na definição do limiar/`threshold` para aplicação do algoritmo, dado que sabemos os suportes para os itens um a um.

Atividade 2 – Minerando Regras (*3,5 pts*)

Use o algoritmo `apriori` para minerar regras na base de dados fornecida. Experimente com pelo menos *3 conjuntos* de valores diferentes de suporte e confiança para encontrar regras de associação. Imprima as cinco regras com o maior suporte de cada conjunto escolhido. Lembre-se de usar seu conhecimento sobre a base, obtido na questão anterior, para a escolha dos valores de suporte e confiança.

```
# Conjunto 1: suporte = 0.06   e confiança = 0.4
rules_set_1 <- apriori(bakery,
                       parameter=list(supp=0.06,
                                      conf=0.4))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.4    0.1    1 none FALSE            TRUE       5    0.06      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 154
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[91 item(s), 2579 transaction(s)] done [0.00s].
## sorting and recoding items ... [15 item(s)] done [0.00s].
```

```
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [9 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_set_1 <- sort(rules_set_1,
                    by = "support",
                    decreasing = TRUE)
inspect(head(rules_set_1, 5))
```

```
##     lhs           rhs      support confidence coverage lift  count
## [1] {}         => {Coffee} 0.5440  0.544      1.000    1.000 1403
## [2] {Bread}    => {Coffee} 0.1539  0.468      0.329    0.861  397
## [3] {Cake}     => {Coffee} 0.1117  0.559      0.200    1.028  288
## [4] {Tea}      => {Coffee} 0.1035  0.433      0.239    0.795  267
## [5] {Sandwich} => {Coffee} 0.0768  0.598      0.128    1.100  198
```

```
# Conjunto 2: suporte = 0.06    e confiança = 0.5
rules_set_2 <- apriori(bakery,
                       parameter=list(supp=0.06,
                                      conf=0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5    0.06      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 154
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[91 item(s), 2579 transaction(s)] done [0.00s].
## sorting and recoding items ... [15 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [7 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_set_2 <- sort(rules_set_2,
                    by = "support",
                    decreasing = TRUE)
inspect(head(rules_set_2, 5))
```

```
##     lhs               rhs      support confidence coverage lift count
## [1] {}             => {Coffee} 0.5440  0.544      1.000    1.00 1403
## [2] {Cake}         => {Coffee} 0.1117  0.559      0.200    1.03  288
```

```
## [3] {Sandwich}      => {Coffee} 0.0768  0.598      0.128    1.10  198
## [4] {Pastry}        => {Coffee} 0.0748  0.564      0.133    1.04  193
## [5] {Hot chocolate} => {Coffee} 0.0659  0.548      0.120    1.01  170
```

```
# Conjunto 3: suporte = 0.07    e confiança = 0.45
rules_set_3 <- apriori(bakery,
                       parameter=list(supp=0.07,
                                      conf=0.45))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##        0.45    0.1    1 none FALSE            TRUE       5    0.07      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 180
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[91 item(s), 2579 transaction(s)] done [0.00s].
## sorting and recoding items ... [14 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 done [0.00s].
## writing ... [5 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
rules_set_3 <- sort(rules_set_3,
                    by = "support",
                    decreasing = TRUE)
inspect(head(rules_set_3, 5))
```

```
##     lhs           rhs      support confidence coverage lift  count
## [1] {}         => {Coffee} 0.5440  0.544      1.000    1.000 1403
## [2] {Bread}    => {Coffee} 0.1539  0.468      0.329    0.861  397
## [3] {Cake}     => {Coffee} 0.1117  0.559      0.200    1.028  288
## [4] {Sandwich} => {Coffee} 0.0768  0.598      0.128    1.100  198
## [5] {Pastry}   => {Coffee} 0.0748  0.564      0.133    1.037  193
```

Análises

a) Quais as regras mais interessantes geradas a partir dessa base? Justifique.

Resposta: Antes de entrarmos nos detalhes de regras em específico, é interessante perceber que, para os 3 pares `(support, confidence)` escolhidos, `Coffee` está no item item consequente para absolutamente todas as 15 regras obtidas para os `thresholds` adotados. No que diz respeito à análise,

- **=> Coffee**: Para todos os 3 conjuntos, Coffee aparece como item consequente sem nenhum precedente com alto valor de suporte de confiança. Basicamente, o significado de tal elemnto nos diz que, a probabilidade de uma pessoa chegar na padaria e comprar café sem nenhum precedente de 54.4%. Embora o valor de lift trivialmente seja 1 dado a independência do evento precedente vazio e do evento consequente existente, é interessante saber que café é o item mais adquirido dentre todos os itens.
- **Cake => Coffee**: Para todos os três conjuntos, a regra apareceu com valores de suporte confiança relevantes. aproximadamente 56% das pessoas que levaram bolo também levaram café, possuindo um valor de lift ligeiramente superior a 1, mostrando uma levíssima dependência positiva entre os elementos.
- **Sandwich => Coffee**: Também para todos os três conjuntos de regras minerados, conseguimos observar valores de suporte e confiança relevantes, mas principalmente de confiança. Dentre as 15 regras obtidas, esta regra apresenta a maior confiança. Aproximadamente 60% das pessoas que levaram sanduíche também levaram café. Neste caso, também analisamos um lift superior a 1, ainda maior do que no caso da regra anterior, mostrando correlação positiva entre os elementos.
- **Bread => Coffee**: Apenas para um dos conjuntos analisados tal regra se demonstrou como relevante dada a ordenação por suporte mas foi deixada de fora nos outros conjuntos por causa s valores limiares/threshold de confiança adotados. Ainda assim, o suporte de tal regra se mostra como o segundo suporte mais elevado. No entanto, o valor de lift neste caso é inferior a 1, demonstrando que pão e café são na verdade negativamente dependentes, o que mostra que a regra não possue tanta relevância assim.
- **Pastry => Coffee**: Tal regra apareceu no top 5 de duas dos três conjuntos minerados. Neste caso, temos valores de suporte um pouco mais reduzidos mas o valor de confiança é de 56% e o lift também é positivo, demonstrando correlação positiva entre folheados e café.

Atividade 3 – Medidas de Interesse (*3,5 pts*)

Vimos na aula que, mesmo após as podas do algoritmo `apriori`, ainda temos algumas regras com características indesejáveis como redundâncias e dependência estatística negativa. Também vimos algumas medidas que nos ajudam a analisar melhor essas regras como o lift, a convicção e a razão de chances. Nesta questão, escolha um dos conjuntos de regras geradas na atividade anterior e o analise usando essas medidas. Compute as três medidas para o conjunto escolhido com a função `interestMeasure` e experimente ordenar as regras com cada uma das novas medidas.

Dica: para adicionar as medidas em um conjunto de regras qualquer, você pode utilizar o comando `cbind` e a função `quality`:

```
quality(regras) <- cbind(quality(regras), interestMeasure(regras, measure=c("conviction"),
                                          transactions = transacoes))
```

```
# Compute as medidas de interesse
quality(rules_set_2) <- cbind(quality(rules_set_2),
                              interestMeasure(rules_set_2,
                                              measure=c("conviction", "leverage", "oddsRatio"),
                                              transactions = bakery))

# Apresente as regras ordenadas por lift
inspect(sort(rules_set_2,
             by = "lift",
             decreasing = TRUE))
```

```
##     lhs                rhs      support confidence coverage lift count
## [1] {Sandwich}      => {Coffee} 0.0768  0.598      0.128    1.10  198
## [2] {Medialuna}     => {Coffee} 0.0609  0.581      0.105    1.07  157
## [3] {Cookies}       => {Coffee} 0.0655  0.577      0.114    1.06  169
## [4] {Pastry}        => {Coffee} 0.0748  0.564      0.133    1.04  193
## [5] {Cake}          => {Coffee} 0.1117  0.559      0.200    1.03  288
## [6] {Hot chocolate} => {Coffee} 0.0659  0.548      0.120    1.01  170
## [7] {}              => {Coffee} 0.5440  0.544      1.000    1.00 1403
##     conviction leverage oddsRatio
## [1] 1.13       0.006953 1.29
## [2] 1.09       0.003923 1.19
## [3] 1.08       0.003724 1.16
## [4] 1.05       0.002694 1.10
## [5] 1.03       0.003038 1.08
## [6] 1.01       0.000526 1.02
## [7] 1.00       0.000000  NaN
```

```
# Apresente as regras ordenadas por convicção
inspect(sort(rules_set_2,
             by = "conviction",
             decreasing = TRUE))
```

```
##     lhs                rhs      support confidence coverage lift count
## [1] {Sandwich}      => {Coffee} 0.0768  0.598      0.128    1.10  198
## [2] {Medialuna}     => {Coffee} 0.0609  0.581      0.105    1.07  157
## [3] {Cookies}       => {Coffee} 0.0655  0.577      0.114    1.06  169
## [4] {Pastry}        => {Coffee} 0.0748  0.564      0.133    1.04  193
## [5] {Cake}          => {Coffee} 0.1117  0.559      0.200    1.03  288
## [6] {Hot chocolate} => {Coffee} 0.0659  0.548      0.120    1.01  170
## [7] {}              => {Coffee} 0.5440  0.544      1.000    1.00 1403
##     conviction leverage oddsRatio
## [1] 1.13       0.006953 1.29
## [2] 1.09       0.003923 1.19
## [3] 1.08       0.003724 1.16
## [4] 1.05       0.002694 1.10
## [5] 1.03       0.003038 1.08
## [6] 1.01       0.000526 1.02
## [7] 1.00       0.000000  NaN
```

```
# Apresente as regras ordenadas por razão de chances
inspect(sort(rules_set_2,
             by = "oddsRatio",
             decreasing = TRUE))
```

```
##     lhs                rhs      support confidence coverage lift count
## [1] {Sandwich}      => {Coffee} 0.0768  0.598      0.128    1.10  198
## [2] {Medialuna}     => {Coffee} 0.0609  0.581      0.105    1.07  157
## [3] {Cookies}       => {Coffee} 0.0655  0.577      0.114    1.06  169
## [4] {Pastry}        => {Coffee} 0.0748  0.564      0.133    1.04  193
## [5] {Cake}          => {Coffee} 0.1117  0.559      0.200    1.03  288
## [6] {Hot chocolate} => {Coffee} 0.0659  0.548      0.120    1.01  170
## [7] {}              => {Coffee} 0.5440  0.544      1.000    1.00 1403
##     conviction leverage oddsRatio
```

```
## [1] 1.13       0.006953 1.29
## [2] 1.09       0.003923 1.19
## [3] 1.08       0.003724 1.16
## [4] 1.05       0.002694 1.10
## [5] 1.03       0.003038 1.08
## [6] 1.01       0.000526 1.02
## [7] 1.00       0.000000  NaN
```

Análise

a) Quais as regras mais interessantes do conjunto? Justifique.

Resposta: De maneira interessante, a ordenação pelas três medidas `lift`, `conviction` e `oddsRatio` resultou exatamente no mesmo ranking de regras mineradas. Se nos restringirmos ao top 5 de regras mais relevantes, veremos que, para todas as regras, os valores de suporte ficaram acima de 0.06 e os valores de confiança acima de 55%, significando que, quem comprou qualquer um dos precedentes das regras possui probabilidade acima de 55% de comprar o consequente. Vamos analisar as duas regras mais relevantes separadamente a seguir:

- **Sandwich => Coffee**: Mesmo antes de olharmos para as medidas de convicção e razão de chances, tínhamos identificado tal regra como forte. No que diz respeito ao suporte, vemos que o valor se mostra elevado (aproximadamente 7.6% de chance do conjunto aparecer em todas as transações) e a confiança se mostra como a maior observada em todas as nossas tratativas. Novamente, vemos que o valor de lift é o mais alto e se mostra como um valor positivo, demonstrando uma dependência estatística positiva e reafirmando a "força" da regra. Quando olhamos para a medida de convicção, vemos que o valor é maior que 1, demonstrando dependência estatística dado que a medida leva em consideração o suporte do antecedente e do consequente, o valor também não se demonstra como extremamente elevado, o que poderia indicar uma regra sem sentido/ilusória. Quando observamos o valor da métrica razão das chances, podemos ver um valor também acima de um, implicando em maiores chances de café ocorrer na presença de sanduíche, demonstrando-se como uma regra forte.
- **Medialuna => Coffee**: Não tínhamos obtido esta regra anteriormente justamente porque estávamos ordenando pelo suporte e apenas observando o top 5. Neste caso, embora o valor de suporte tenha um valor um pouco menor, ainda se mostra bastante relevante, e o valor de confiança também é destaque, dado que, 58.1% das pessoas que levaram medialuna/croissant também levaram café. No que diz respeito ao lift, observamos um valor positiva, o que mostra dependência estatística positiva, mostrando que a regra é forte. No que diz respeito à convicção, também observamos um valor positivo, nos levando à mesma conclusão da medida anterior. Por fim, analisando-se o valor da medida razão de chances, observamos um valor positivo, implicando novamente em maiores chances do café ocorrer na presença de medialuna/croissant.