Trabalho Computacional 03
TIP8311 - Reconhecimento de Padrões
Clusterização em Dados Desconhecidos

Artur Rodrigues Rocha Neto - 431951
Mestrando em Engenharia de Teleinformática
artur.rodrigues26@gmail.com

7 de Dezembro de 2018

1 Introdução

Clusterização faz parte de diversos *pipelines* de predição. Ela envolve o conjunto de algoritmos responsáveis por agrupar objetos/entidades/amostras a partir de medidas de semelhança, ou seja, amostras de um dado grupo são mais parecidas entre si que entre amostras de um outro grupo. É uma técnica ditanão-supervisionado. Muitas tarefas de aprendizagem de máquina envolvem a exploração de dados que não possuem informação de classe *a priori*. As técnicas de clusterização e validação ajudam a revelar padrões entre as observações.

Neste trabalho, foi fornecido um conjunto de dados com dados que aborda o perfil de consumidores de uma cadeia de produção de uma empresa produtora de utensílios de uso geral. O objetivo é encontrar padrões nesses consumidores que possam ajudar a melhor a relação da empresa com estes, diminuindo custos e aumentando lucros.

2 Conjunto de Dados

Atributo	**Média**	**Mediana**	**Min**	**Max**	**Desvio**
atrib1	99.15	100.0	0.0	100.0	5.91
atrib2	4.2	3.5	0.0	72.67	3.66
atrib3	1439.95	350.5	38.22	117287.3	4785.7
atrib4	2.17	1.95	0.21	3.46	1.04
atrib5	2.23	2.25	0.27	3.46	1.01
atrib6	5014223.64	8001856.0	10.0	9995020.0	4557586.07

Tabela 1: Estatísticas gerais dos atributos

O conjunto de dados consta de $N = 1701$ amostras com $p = 6$ atributos cada. Os atributos receberam os nomes genéricos (`atrib1`, `atrib2`, ..., `atrib6`) apenas para facilitar as análises, já que um cabeçalho oficial não foi disponibilizado. A Figura 1 mostra as relações par-a-par entre os preditores. A distribuição de

alguns atributos é bastante enviesada, enquanto que os demais assemelham-se pouco a curvas normais. `atrib4` e `atrib5` apresentação uma forma relação linear entre si. A Tabela 1 traz as estatísticas gerais do conjunto. Alguns pontos que saltam aos olhos sao a presença apenas de dados positivos e alto valor de desvio padrão de `atrib6`. O mapa de calor na Figura 2 revela uma correlação absoluta baixa no conjunto, com exceção do par (`atrib4`, `atrib5`) que, como já visto na visão em pares, apresenta uma colinearidade quase total.

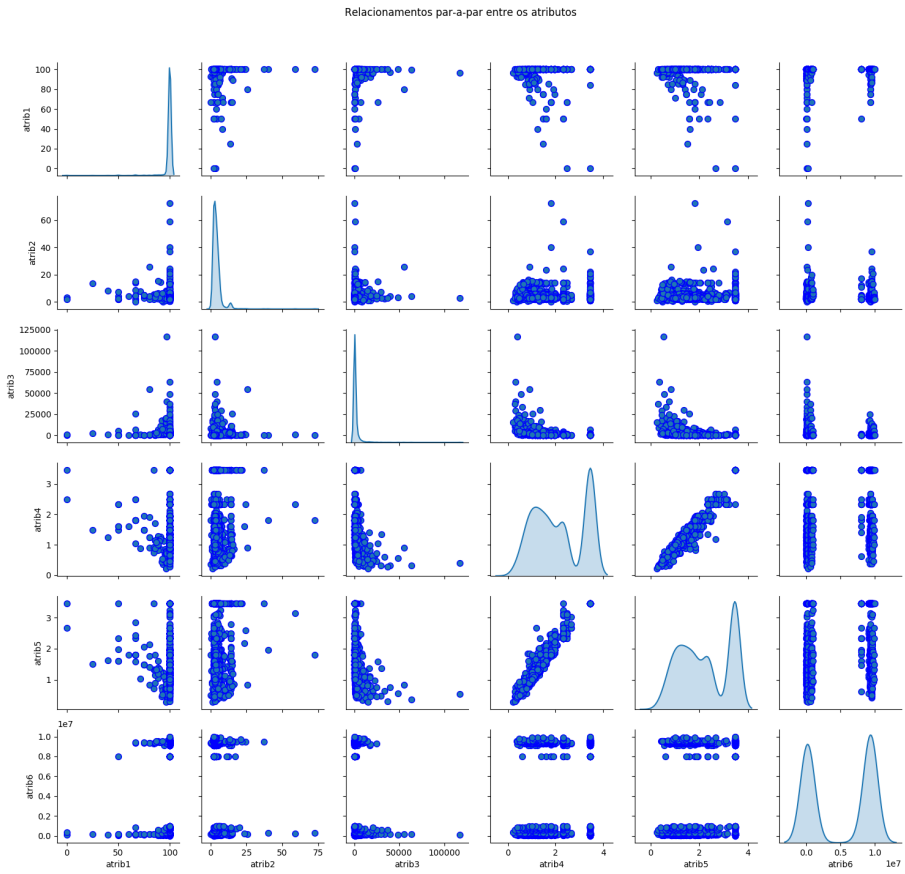

Figura 1: Relacionamento entre os dados do conjunto desconhecido

3 Metodologia

O objetivo deste trabalho é conseguir distinguir um dado número de agrupamentos no conjunto de dados. Para tanto, usaremos o algoritmo K-médias para estimar diversos cenários de clusterização. Para verificar a qualidade do agrupamento, serão calculados os índices de validação Calisnki-Harabasz, Davies-Bouldin e Dunn. Cada sugestão será então analisada em termos de média, mediana, mínimo e desvio padrão dos clusters associados. O ambiente de experimentação foi implementado em Python 3.5 com o auxílio da biblioteca de

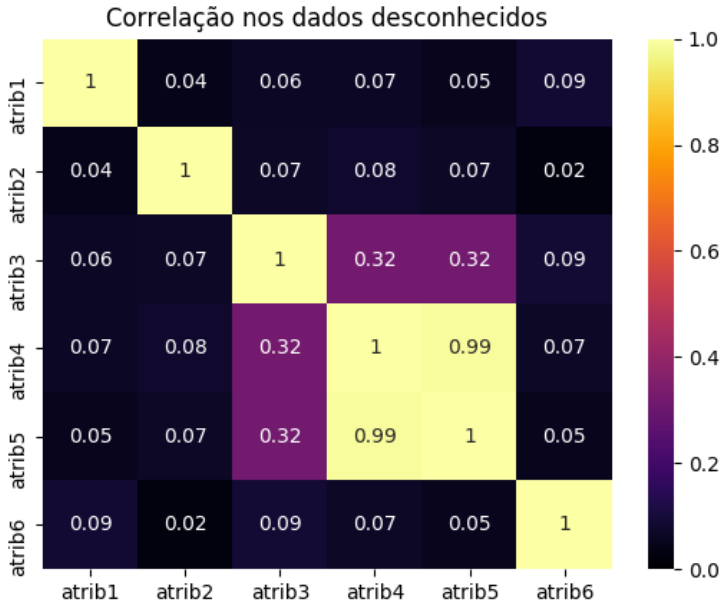

Figura 2: Correlação absoluta dos atributos

aprendizagem de máquina ***scikit-learn*** [1]. Todos os testes foram executados em uma máquina Debian GNU/Linux 9.5 com 8GB de RAM e processador i7-3770 @3.40GHz x4.

3.1 Algoritmo de Clusterização k-médias

O K-médias é um método de clusterização (ou agrupamento) que arranja massas de dados em n conjuntos bem separados e de igual variância. Seu funcionamento baseia-se na minimização de um critério chamado *soma das distâncias quadráticas*, do inglês *Squared Sum Distance* ou apenas SSD. O índice SSD é conhecido também como inércia: quanto menor a inércia de um ponto, menos esse ponto "se moveu"de uma interação a outra. A inércia, portanto, é o critério de convergência do K-médias. O algoritmo pode ser descrito como [2]:

Passo 1: Definir um valor para K.
Passo 2: Atribuir valores iniciais aos K protótipos.
Passo 3: Determinar a partição V_i do protótipo w_i, $i = 1, 2, ..., K$, unsando a Eq.1:

$$V_i = \{x \in \mathbb{R}^p | ||x - w_i|| < ||x - w_j||, \forall j \neq i\} \quad (1)$$

Passo 4: Calcular a nova posição do protótipo w_i como a média dos N_i objetos da partição V_i:

$$w_i = \frac{1}{N_i} \sum_{x \in V_i} x \quad (2)$$

Passo 5: Repetir os Passos 3 e 4 até a convergência do algoritmo.

A implementação do K-médias no pacote ***scikit-learn*** encontra-se na classe `sklearn.cluster.KMeans`.

3.2 Índice Calinski-Harabasz

Sobre o índice Calinski-Harabasz (CH), tamabém conhecido como Critério de Razão de Dispersão, pode ser usado na avaliação de modelos clusterizado. Quando maior for o CH, melhor será o agrupamento. Dado o número de clusters k, podemos calcular CH a partir da Eq.3, onde $tr(.)$ é o operador traço de matriz e B_k e W_k são, respectivamente, a matriz de dispersão intercluster (Eq.4) e a matriz de dispersão intracluser (Eq.5).

$$CH = \frac{tr(B_k)}{tr(W_k)} \times \frac{N-k}{k-1} \tag{3}$$

$$B_k = \sum_q n_q (c_q - c)(c_q - c)^T \tag{4}$$

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T \tag{5}$$

Quanto maior o valor do CH, melhor o agrupamento. Foi usada a função `sklearn.metrics.calinski_harabaz_score` para o cálculo do índice CH.

3.3 Índice Davies-Bouldin

O índice de Davies-Bouldin (DB) é outra métrica de avaliação de modelos clusterizados. Ao contrário do CH, o DB aponta melhores agrupamentos quanto menor for seu valor. A Eq.6 traz a fórmula do índice:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij} \tag{6}$$

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{7}$$

Onde:

- s_i: distância média entre cada ponto do cluster i e o centróide desse cluster
- d_{ij}: distância entre os centróides dos clusters i e j

Zero é o menor valor possível. Valores próximos de zero indicam bons agrupamentos. A função `sklearn.metrics.davies_bouldin_score` foi usada para calcular o índice DB.

3.4 Índice Dunn

O índice Dunn para um dado valor K é calculado como [2]:

$$DUNN = \frac{\min\limits_{i \neq j}\{\delta(V_i, V_j)\}}{\max\limits_{1 \leq l \leq K}\{\Delta(V_l)\}} \tag{8}$$

em que:

1. $\delta(V_i, V_j)$ denota uma medida de dissimilaridade entre as partições V_i e V_j:

$$\delta(V_i, V_j) = \min_{x \in V_i, y \in V_j}\{d(x, y)\} \tag{9}$$

2. $\Delta(V_l)$ é uma medida de dispersão dos dados da partição V_l:

$$\Delta(V_l) = \max_{x, y \in V_l}\{d(x, y)\}$$

Valores próximo de 1 indicam bons agrupamentos. O ***scikit-learn*** implementa o índice Dunn em `sklearn.metrics.silhouette_score`.

3.5 Normalização

De forma a melhorar a estabilidade de certos cálculos, um dos passos de pré-processamento mais simples e utilizado em *pipelines* de predição é a normalização, também conhecida como escalamento ou padronização [3].

Para normalizar um conjunto de dados, primeiro fazemos uma transformação de centralização, subtraindo de cada valor de um atributo k a média dos valores do mesmo $\bar{x_k}$. Depois, uma operação de escala é efetuada dividindo todos os valores de cada atributo x_k pelo desvio padrão σ_k. Ambos os passos podem ser descritos com uma fórmula, mostrada em Eq.10.

$$x_k^* = \frac{x_k - \bar{x_k}}{\sigma_k} \tag{10}$$

A transformação de normalização é implementada no ***scikit-learn*** pela classe `sklearn.preprocessing.StandardScaler`.

3.6 Remoção de Assimetria

Um distribuição de dados é dita simétrica quando a probabilidade de um evento ocorrer de um lado da distribuição é praticamente a mesmo que na região equidistante oposta. A assimetria de uma distribuição é cáculada usando a Eq.11, onde x é o atributo, n o número de valores e $\bar{x}$ é a média do atributo. Um distribuição possui *assimetria à direita* quando apresenta uma maior densidade de valores no lado esquerdo que do lado direito, e o seu valor é positivo. O análogo também serve para a *assimetria à esquerda* e o seu valor é negativo.

$$\begin{aligned} assimetria &= \frac{\sum(x_i - \bar{x})^3}{(n-1)v^{3/2}} \\ v &= \frac{\sum(x_i - \bar{x})^2}{(n-1)} \end{aligned} \tag{11}$$

Existe uma família de transformações que podem ser usadas para remover assimetria de dados. Aquela escolhida para esse trabalho foi a Yeo-Johnson [4], mostrada na Eq.12, onde λ é um parâmetro que define a transformação (pode ser estimado a partir de conjunto de treinamento [3]):

$$x^* = \begin{cases} ((x+1)^\lambda - 1)/\lambda & \text{se } \lambda \neq 0,\ x \geq 0 \\ \log(x+1)\text{se} & \text{se } \lambda = 0,\ x \geq 0 \\ -[(-x+1)^{2-\lambda} - 1]/(2-\lambda) & \text{se } \lambda \neq 2,\ x < 0 \\ -\log(-x+1) & \text{se } \lambda = 2,\ x < 0 \end{cases} \tag{12}$$

A transformação de remoção de assimetria é implementada no ***scikit-learn*** pela classe `sklearn.preprocessing.PowerTransformer`.

4 Resultados

Dois *scripts* Python foram criados: `recpad.py`, que agrega diversas funções utilitárias para classificação e visualização, e `tc3.py`, sequência do passo-a-passo de geração de resultados deste trabalho. Pedaços principais de código serão apresentados ao longo dos resultados. Para uma avaliação das implementações, ver arquivo `recpad.py`.

Foram testados 19 valores para K, de 2 a 20. O K-médias foi configurado para ser executado 50 vezes antes de acomodar o valor de SSD e o máximo de interações até a convergência foi definido como 2000.

```
from recpad import *

dataset = "data/datasetTC3.dat"
cols = ["atrib{}".format(n) for n in range(1, 7)]

df = pd.read_csv(dataset, header=None)
df.columns = cols
X = np.array(df)
X_trans = super_normalize(X)
X_trans = super_unskew(X_trans)
X_trans = pd.DataFrame(X_trans, columns=cols)
```

Listing 1: Cabeçalho do código trabalho03

4.1 Resultados Davies-Bouldin e Dunn

Os índices Davies-Bouldin e Dunn foram unânimes na escolha do número ótimo de grupos. Ambos indicaram $K = 2$ como a melhor escolha. A Figura 3 mostra a evolução das taxas de validação para a sequência de teste escolhida.

4.2 Resultados Calinski-Harabaz

Os primeiros resultados do índice Calinski-Harabaz não foram conclusivos. A Figura 4 revela que o valor de CH só aumenta quanto maior for o valor de K. Foram empregadas as transformações de normalização e remoção de assimetria para avaliar se, uma vez transformados, o dados revelariam a mesma tendência de agrupamento propostos pelos índices Davies-Bouldin e Dunn.

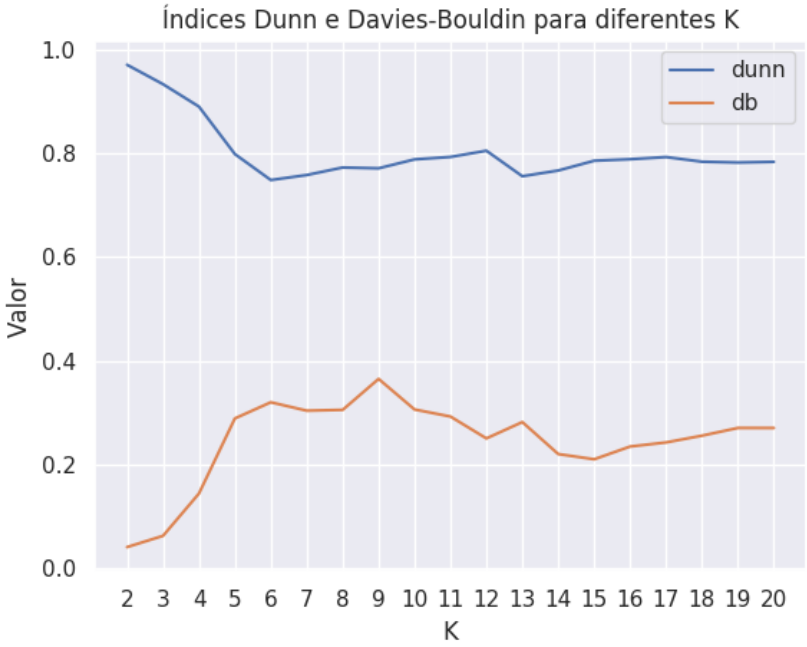

Figura 3: Resultado para índices Davies-Bouldin e Dunn

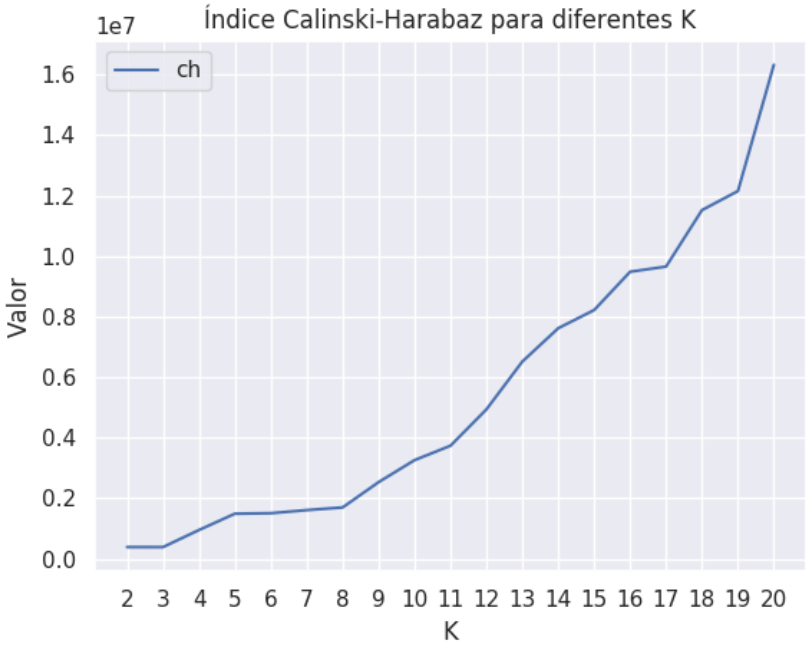

Figura 4: Resultado inicial para o índice Calinski-Harabaz

Após a aplicação de pré-processamento, constatou-se mais uma vez $K = 2$ como a melhor escolha de cluster. A Figura 5 mostra a curva de evolução, com pico em 2.

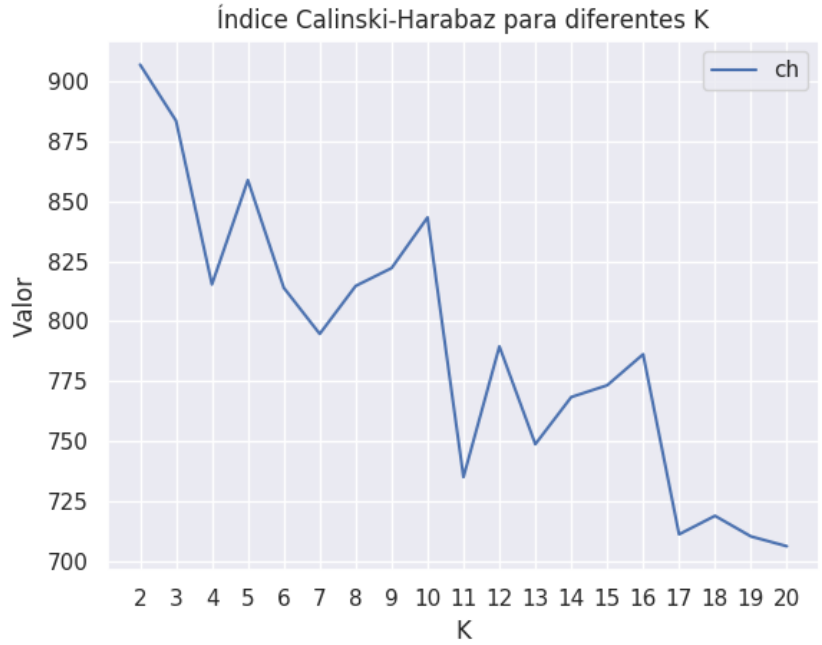

Figura 5: Resultado pós-transformações para o índice Calinski-Harabaz

```python
# funcao contida em recpad.py:
def clustering_kmeans(X, n):
    #km = KMeans(n_clusters=n, n_init=50, max_iter=2000, n_jobs=-1)
    km = MiniBatchKMeans(n_clusters=n, n_init=50, max_iter=2000)
    km = km.fit(X)
    labels = km.labels_

    ch = calinski_harabaz_score(X, labels)
    db = davies_bouldin_score(X, labels)
    dunn = silhouette_score(X, labels)

    return ch, db, dunn, labels, km.cluster_centers_

data = {"n" : [], "ch" : [], "db" : [], "dunn" : []}
for n in range(2, 21):
    ch, db, dunn, _, _ = clustering_kmeans(X, n)
    data["n"].append(n)
    data["ch"].append(ch)
    data["dunn"].append(dunn)
    data["db"].append(db)
    print("n={}, ch={}, db={}, dunn={}".format(n, ch, db, dunn))
```

Listing 2: Cálculo dos diversos índices

4.3 Análise Estatística

Com a escolha de $K = 2$ definida e justificada pelos três índices escolhidos, finalizamos agrupando algumas informações estatísticas do conjunto de dados particionado. A Tabela 2 mostra as estatísticas dos atributos após a clusterização. Os resultado foram semelhantes aos da Tabela 1, o que mostra que as classes foram encontrados em concordância com o perfil geral do conjunto de dados. A divisão final foi de 890 amostras em um grupo e 811 no outro.

Atributo	Média	Mediana	Min	Max	Desvio
atrib1	99.1	99.1	98.61	99.59	0.69
atrib2	4.09	4.09	4.08	4.1	0.01
atrib3	1446.55	1446.55	1001.09	1892.02	629.98
atrib4	2.14	2.14	2.07	2.22	0.11
atrib5	2.2	2.2	2.15	2.26	0.08
atrib6	4798539.41	4798539.41	242231.84	9354846.98	6443591.96

Tabela 2: Estatísticas gerais dos atributos clusterizados

```
ch, db, dunn, labels, centroids = clustering_kmeans(X, 2)
centroids = pd.DataFrame(centroids, columns=cols)
stats = data_stats(centroids)
stats.to_csv("data/tc3-analise-estatistica.csv")

uni, count = np.unique(labels, return_counts=True)
print("{} {}".format(uni, count))
```

Listing 3: Análise estatística

5 Conclusões

Nesse trabalho, praticou-se o uso de um algoritmo de clusterização para ajudar a encontrar perfis em um conjunto de dados. O uso de três distintos índices de validação, aliados a técnicas de pré-processamento e exploração de dados, se mostrou capaz de definir um particionamento satisfatório.

Referências

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[2] Guilherme de Alencar Barreto. Introdução à clusterização de dados. Slides da disciplina TIP8311 - Reconhecimento de Padrões, 2018.

[3] Max Kuhn and Kjell Johnson. *Applied predictive modeling*, volume 26. Springer, 2013.

[4] In-Kwon Yeo and Richard A Johnson. A new family of power transformations to improve normality or symmetry. *Biometrika*, 87(4):954–959, 2000.