Relatório de Análise: Clustering e Regras de Associação

1. Introdução

O presente relatório tem como objetivo analisar os clusters gerados pelo algoritmo de agrupamento *K-means* e relacioná-los com as regras de associação obtidas pelo algoritmo *Apriori*, utilizando a base de dados do cliente A. A análise busca compreender os padrões de comportamento dos clientes inadimplentes e identificar características relevantes para a tomada de decisão em processos de cobrança.

2. Metodologia

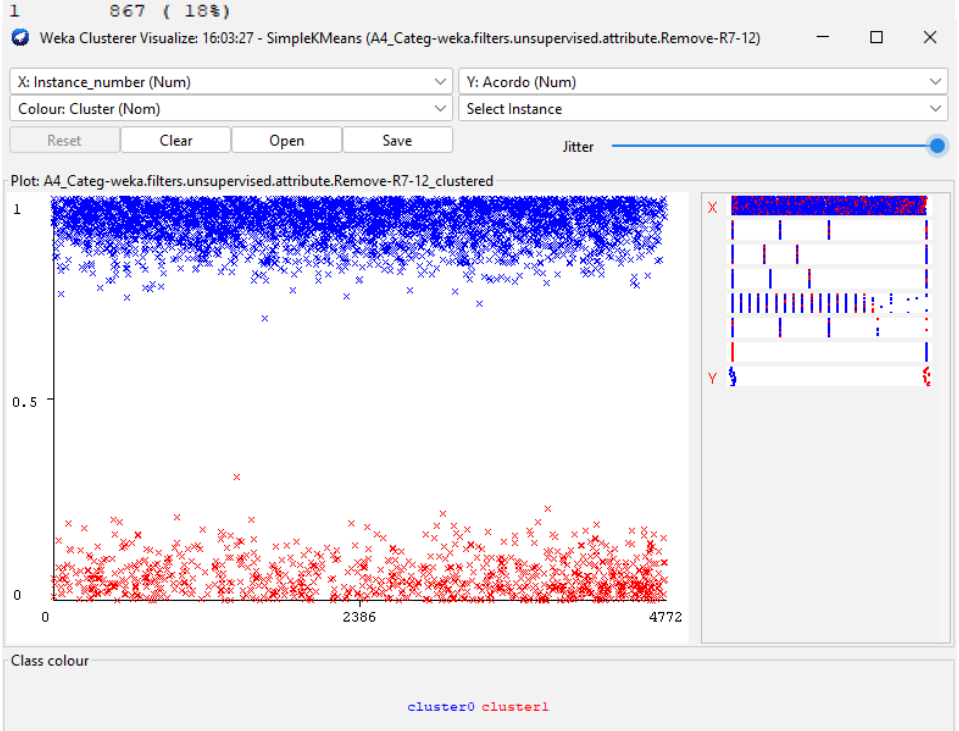
- **Base de dados utilizada:** histórico de 4.773 clientes devedores do cliente A.
- **Atributos considerados:** Idade, Atraso (dias de inadimplência), Valor do débito (da dívida), Contato Efetivo (número de tentativas de contato com o cliente), Efetivo (número de contatos efetivos) e Acordo (variável objetivo: se houve ou não acordo).
- **Pré-processamento:** para o algoritmo K-means, os atributos foram convertidos em faixas e codificados numericamente para otimizar a execução no Weka. Já para o algoritmo Apriori, os atributos foram também convertidos para uma descrição nominal. Todos os dados foram mantidos na mesma tabela e selecionados para a execução de cada algoritmo.
- **Algoritmos aplicados:**
  - *K-means*: variando o número de clusters de 2 a 2000, observando a formação dos grupos e o decaimento do erro RMS.
  - *Apriori*: geração de regras de associação entre atributos e a variável objetivo (Acordo).

3. Resultados – Clustering (K-means)

A aplicação do algoritmo K-means possibilitou a análise da base de dados em diferentes quantidades de grupos (clusters).

Com **2 clusters**, houve uma divisão clara entre clientes que fecharam acordo (3.906 instâncias) e clientes que não fecharam acordo (867 instâncias).

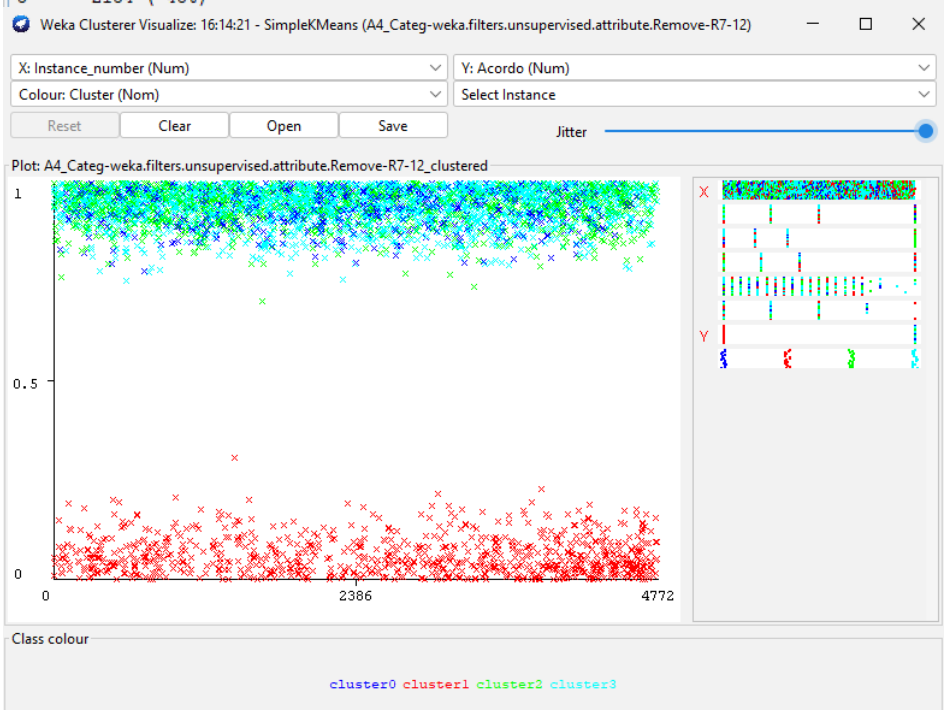
```
kMeans
======

Number of iterations: 10
Within cluster sum of squared errors: 1861.1424641830763

Initial starting points (random):

Cluster 0: 4,1,1,4,1,1
Cluster 1: 4,2,5,10,2,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1
               (4773.0)   (3906.0)    (867.0)
============================================
ï»¿Idade         1.7461     1.7087     1.9146
Atraso           2.4588     2.5507      2.045
Valor            1.4718     1.3525     2.0092
CONTATO          4.7591      4.649     5.2549
EFETIVO          1.1672     1.1792      1.113
Acordo           0.8184          1          0



Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       3906 ( 82%)
1        867 ( 18%)
```



Com **4 clusters**, três grupos distintos de clientes que fecham acordos e um grupo de clientes que não fecham.

Clusterer output

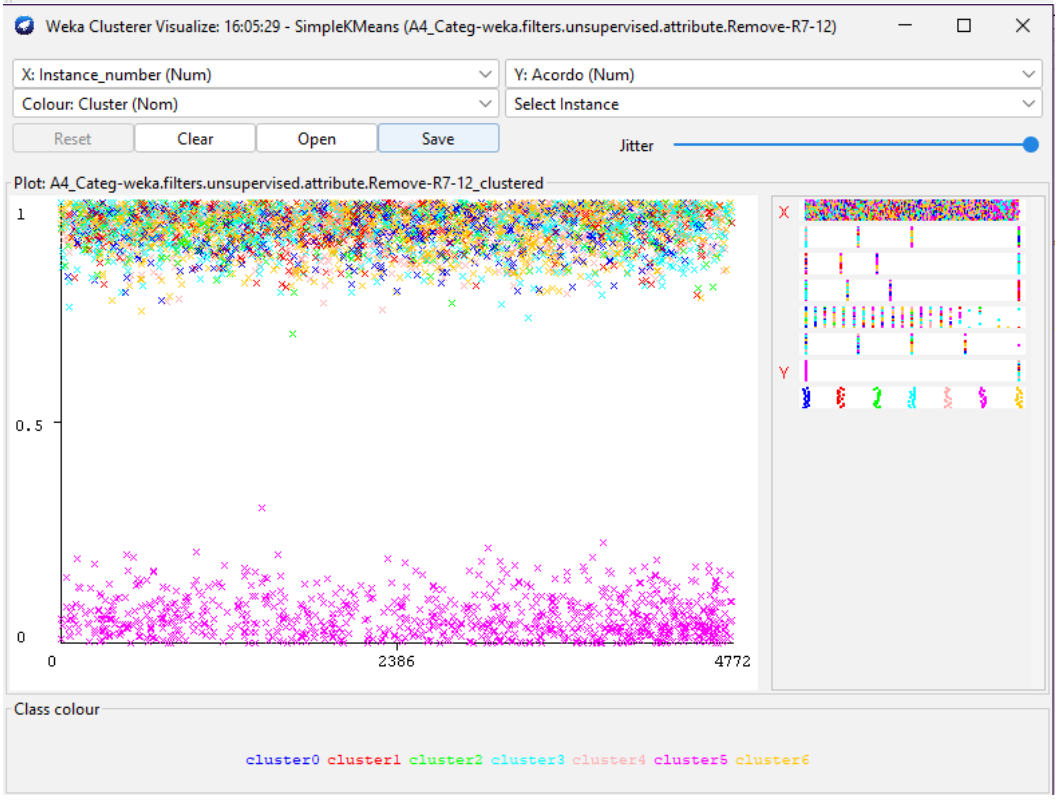
```
kMeans
======

Number of iterations: 12
Within cluster sum of squared errors: 1010.776154354433

Initial starting points (random):

Cluster 0: 4,1,1,4,1,1
Cluster 1: 4,2,5,10,2,1
Cluster 2: 4,2,2,2,2,1
Cluster 3: 1,2,1,2,1,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3
              (4773.0)    (665.0)    (867.0)   (1107.0)   (2134.0)
=============================================================================
ï»¿Idade        1.7461          4     1.9146     1.6603     1.0197
Atraso          2.4588     1.1955      2.045          6     1.1837
Valor           1.4718     1.9684     2.0092     0.4011     1.6542
CONTATO         4.7591     4.3835     5.2549     4.5718     4.7718
EFETIVO         1.1672      1.182      1.113     1.1879     1.1739
Acordo          0.8184          1          0          1          1



Time taken to build model (full training data) : 0.04 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       665 ( 14%)
1       867 ( 18%)
2      1107 ( 23%)
3      2134 ( 45%)
```



Com **7 clusters**, observou-se o melhor equilíbrio entre separação de grupos e erro RMS (712,12), indicando que esse número é adequado para representar a base. Nesse cenário, um cluster concentra majoritariamente clientes sem acordo, enquanto os demais representam perfis distintos de clientes que fecharam acordo.

Clusterer output

```
kMeans
======

Number of iterations: 14
Within cluster sum of squared errors: 712.1244016805499

Initial starting points (random):

Cluster 0: 4,1,1,4,1,1
Cluster 1: 4,2,5,10,2,1
Cluster 2: 4,2,2,2,2,1
Cluster 3: 1,2,1,2,1,1
Cluster 4: 1,2,2,4,1,1
Cluster 5: 1,2,1,8,1,1
Cluster 6: 2,2,1,10,2,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4          5          6
             (4773.0)    (534.0)    (384.0)    (236.0)    (869.0)    (759.0)    (867.0)   (1124.0)
=============================================================================================================
ï»¿Idade        1.7461          4     2.1432          4     1.0311     0.2688     1.9146     1.4867
Atraso          2.4588     1.2772     0.7969          6          6     0.8814      2.045     1.4911
Valor           1.4718     1.2247          5     0.4576     0.3751     1.2648     2.0092     1.1699
CONTATO         4.7591      4.161     5.8203     3.9958     4.7135     4.8577     5.2549      4.427
EFETIVO         1.1672     1.1685     1.2422     1.2246     1.1749     1.1291      1.113     1.1904
Acordo          0.8184          1          1          1          1          1          0          1



Time taken to build model (full training data) : 0.04 seconds
```



Para valores **acima de 7 clusters**, o modelo passa a dividir grupos de forma redundante, sem agregar informação significativa.

8 clusters:

```
kMeans
======

Number of iterations: 9
Within cluster sum of squared errors: 594.6409667274219

Initial starting points (random):

Cluster 0: 4,1,1,4,1,1
Cluster 1: 4,2,5,10,2,1
Cluster 2: 4,2,2,2,2,1
Cluster 3: 1,2,1,2,1,1
Cluster 4: 1,2,2,4,1,1
Cluster 5: 1,2,1,8,1,1
Cluster 6: 2,2,1,10,2,1
Cluster 7: 0,0,1,1,1,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4          5          6          7
               (4773.0)    (534.0)    (384.0)    (236.0)    (869.0)    (686.0)    (181.0)    (547.0)   (1336.0)
=============================================================================================================
ï»¿Idade        1.7461          4     2.1432          4     1.0311     1.9519     1.7735          2     0.5846
Atraso          2.4588     1.2772     0.7969          6          6     1.0015          6     1.2742     1.2335
Valor           1.4718     1.2247          5     0.4576     0.3751     2.3601     0.6796     1.2486     1.1916
CONTATO         4.7591      4.161     5.8203     3.9958     4.7135     5.3353     4.9503     4.5027     4.6407
EFETIVO         1.1672     1.1685     1.2422     1.2246     1.1749     1.1064     1.1381     1.1828     1.1587
Acordo          0.8184          1          1          1          1          0          0          1          1
```

9 clusters: erro RMS volta a aumentar

```
kMeans
======

Number of iterations: 23
Within cluster sum of squared errors: 674.3703620353895

Initial starting points (random):

Cluster 0: 4,1,1,4,1,1
Cluster 1: 4,2,5,10,2,1
Cluster 2: 4,2,2,2,2,1
Cluster 3: 1,2,1,2,1,1
Cluster 4: 1,2,2,4,1,1
Cluster 5: 1,2,1,8,1,1
Cluster 6: 2,2,1,10,2,1
Cluster 7: 0,0,1,1,1,1
Cluster 8: 1,6,0,2,1,1

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute    Full Data          0          1          2          3          4          5          6          7          8
               (4773.0)    (534.0)    (384.0)    (269.0)    (733.0)    (834.0)    (352.0)    (321.0)    (477.0)    (869.0)
=========================================================================================================================
ï»¿Idade        1.7461          4     2.1432          4     1.3083     1.8321          0          2     0.5744     1.0311
Atraso          2.4588     1.2772     0.7969          6     1.8472     1.8885     1.7131      0.785     0.2851          6
Valor           1.4718     1.2247          5     0.4498     0.8881     2.0731     0.8949     1.6293     1.6478     0.3751
CONTATO         4.7591      4.161     5.8203     4.0595     4.4557     5.2842     4.7273     4.5358     4.7736     4.7135
EFETIVO         1.1672     1.1685     1.2422      1.223     1.2033     1.1091      1.125     1.1558     1.1447     1.1749
Acordo          0.8184          1          1     0.8773          1          0          1          1          1          1
```

1472 clusters: erro RMS zera

Clusterer output

```
=== Run information ===

Scheme:       weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 1472 -A "weka.core.EuclideanDistance -R
Relation:     A4_Categ-weka.filters.unsupervised.attribute.Remove-R7-12
Instances:    4773
Attributes:   6
              Ï»¿Idade
              Atraso
              Valor
              CONTATO
              EFETIVO
              Acordo
Test mode:    evaluate on training data


=== Clustering model (full training set) ===


kMeans
======

Number of iterations: 2
Within cluster sum of squared errors: 0.0
```

4. Resultados – Regras de Associação (Apriori)

A análise de regras de associação foi realizada com o algoritmo Apriori no Weka, utilizando apenas os atributos categorizados da base de dados: *idade-cat*, *faixas-valor*, *faixas-atrasos*, *contatos*, *contatos-efetivos* e *Acordo-cat*. Para a execução, foram definidos parâmetros de suporte mínimo de 10% (ou seja, a regra deveria aparecer em pelo menos 477 registros da base) e confiança mínima de 90%.

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:     A4_Categ-weka.filters.unsupervised.attribute.Remove-R1-6
Instances:    4773
Attributes:   6
              idade-cat
              faixas-valor
              faixas-atrasos
              contatos
              contatos-efetivos
              Acordo-cat
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.1 (477 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 18

Size of set of large itemsets L(2): 48

Size of set of large itemsets L(3): 43

Size of set of large itemsets L(4): 13

Best rules found:

 1. contatos=Poucos Acordo-cat=Não 538 ==> contatos-efetivos=Um 507    <conf:(0.94)> lift:(1.1) lev:(0.01) [48] conv:(2.47)
 2. faixas-atrasos=0-15 dias contatos=Poucos 656 ==> contatos-efetivos=Um 596    <conf:(0.91)> lift:(1.06) lev:(0.01) [36] conv:(1.58)
 3. Acordo-cat=Não 867 ==> contatos-efetivos=Um 786    <conf:(0.91)> lift:(1.06) lev:(0.01) [46] conv:(1.55)
```

O resultado mostrou que as regras mais relevantes estão relacionadas principalmente ao número de contatos efetivos realizados pela central de cobrança. Algumas regras encontradas foram:

- Se contatos = Poucos e Acordo-cat = Não, então contatos-efetivos = Um (confiança: 94%, lift: 1.1).
- Se faixas-atrasos = 0–15 dias e contatos = Poucos, então contatos-efetivos = Um (confiança: 91%, lift: 1.06).
- Se Acordo-cat = Não, então contatos-efetivos = Um (confiança: 91%, lift: 1.06).

Essas regras significam que a probabilidade de um cliente não fechar acordo aumenta quando existe apenas um contato efetivo. Em outras palavras, clientes pouco contatados, mesmo aqueles com atraso pequeno, acabam tendo baixa taxa de acordo.

O valor de confiança indica a proporção de casos em que a regra se confirma (por exemplo, 94% significa que em 94% dos registros onde o cliente não fechou acordo e teve poucos contatos, ele de fato recebeu apenas um contato efetivo). Já o lift mostra a “força” da regra em relação ao acaso (valores acima de 1 indicam que a regra é útil, embora neste caso os lifts estejam próximos de 1, o que mostra que as regras são relevantes, mas não extremamente fortes).

Em síntese, o Apriori evidenciou que a efetividade do processo de contato é um fator decisivo para o fechamento de acordos. Esse resultado complementa a análise dos clusters, que também mostraram a separação clara entre clientes que fecham e não fecham acordo. Assim, a empresa pode direcionar estratégias de cobrança aumentando o número de contatos efetivos em determinados perfis de clientes, o que tende a elevar a taxa de sucesso das negociações.

5. Relação entre Clusters e Regras Apriori

- Regras ligadas ao número de contatos efetivos reforçam a interpretação de clusters com maior taxa de conversão, mostrando a importância do telemarketing ativo.
- Isso sugere que o cluster de "não acordo" não é apenas um grupo homogêneo de inadimplentes, mas sim um grupo caracterizado por pouca insistência da central de cobrança.
- Já os clusters de clientes com acordo podem estar associados aos casos em que houve mais de um contato efetivo, confirmando que maior esforço de cobrança aumenta a chance de negociação.

6. Conclusão

A análise integrada de clustering (K-means) e regras de associação (Apriori) mostrou-se fundamental para compreender os padrões de comportamento dos clientes inadimplentes.

O algoritmo K-means indicou que o número ótimo de clusters para representar a base é 7, sendo possível identificar grupos distintos de clientes, entre eles um cluster claramente formado por indivíduos que não fecham acordo.

As regras do Apriori complementaram essa descoberta ao evidenciar que a baixa efetividade de contatos está diretamente relacionada à ausência de acordos. Regras como "Acordo = Não ⇒ Contatos-efetivos = Um" (com mais de 90% de confiança) reforçam que clientes que não negociam quase sempre receberam apenas um contato efetivo.

Dessa forma, a integração dos dois métodos não apenas separou os grupos de clientes, mas também explicou a causa da baixa taxa de acordo em determinados clusters: a insuficiência de interações efetivas da central de cobrança.

Portanto, a análise sugere que estratégias de cobrança mais bem-sucedidas dependem do aumento do número de contatos efetivos, especialmente em perfis de clientes com maior potencial de negociação, permitindo otimizar recursos e elevar a taxa de conversão.

Relatório de Análise: gráfico de erro RMS com K-means

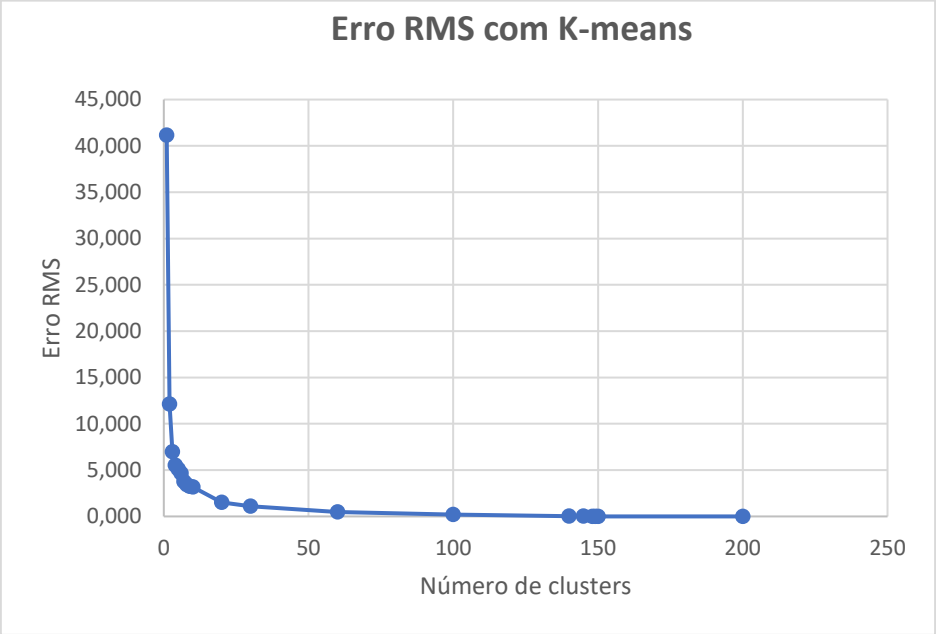
No segundo experimento, utilizou-se a base de dados IrisDataSet contida no arquivo *iris.csv*, amplamente conhecida em experimentos de aprendizado de máquina e clustering. Antes da execução do K-Means, o atributo referente às variedades originais das flores (Setosa, Versicolor e Virginica) foi removido, garantindo que o algoritmo formasse clusters sem referência às classes verdadeiras, de modo a avaliar sua capacidade de segmentação puramente baseada nos atributos numéricos.

O K-Means foi executado no Weka testando diferentes números de clusters (*k*), e o desempenho do modelo foi avaliado utilizando o erro RMS (Root Mean Square), representando a média do desvio quadrático entre os pontos de dados e os centróides de seus respectivos clusters. Valores menores de RMS indicam clusters mais bem ajustados aos dados.

A Tabela a seguir apresenta os valores de RMS obtidos para diferentes números de clusters:

Número de clusters	Erro RMS
1	41,16611042137320
2	12,127790750538
3	6,98221647378523
4	5,51693347204037
5	5,11488711601964
6	4,6711176350226
7	3,77280518191115
8	3,41471060336976
9	3,24721930944449
10	3,15584537509634
20	1,50332137087241
30	1,11469111181329
60	0,46963718602970
100	0,18912043594534
140	0,03832569597391
145	0,00978038629456
148	0,00298581297591
149	0,00000000000000
150	0,00000000000000
200	0,00000000000000

Observa-se que o erro RMS diminui à medida que aumenta o número de clusters, como esperado. No entanto, a partir de **k = 3 a 5**, a redução do RMS começa a se estabilizar, indicando um ponto de **diminuição marginal de retorno**. Este comportamento sugere que valores próximos de **3 clusters** já representam uma boa segmentação natural dos dados, o que é consistente com o conhecimento da base, que possui três variedades originais de flores.



O gráfico de erro RMS gerado ilustra claramente esta tendência: uma queda acentuada do erro até **k = 3**, seguida de uma redução gradual conforme o número de clusters aumenta, evidenciando que o modelo já atinge boa qualidade de clusterização mesmo sem utilizar o atributo de classe original.

Conclusão

O experimento demonstrou que o K-Means consegue identificar naturalmente a estrutura da base Iris mesmo sem o atributo de classe. O erro RMS caiu rapidamente ao aumentar o número de clusters até 3, indicando que três grupos representam bem a segmentação dos dados. A partir daí, aumentos adicionais no número de clusters proporcionam ganhos marginais, confirmando que 3 clusters são suficientes para uma boa clusterização. O gráfico de RMS reforça visualmente essa tendência, mostrando claramente o ponto em que o ajuste dos clusters se estabiliza.

Referências

- Material teórico fornecido pela disciplina (Estudo de Caso – K-means).
- Han, J.; Kamber, M.; Pei, J. *Data Mining: Concepts and Techniques*.
- Documentação do Weka