# 4 Soluções

A seguir serão descritas as diversas estratégias usadas para acelerar a deduplicação. Assim como a metodologia, modelagens, algoritmos e representações utilizadas na realização dos experimentos realizados ao longo da execução do trabalho.

## 4.1. Estratégias

Para equacionar o problema proposto e alcançar os objetivos traçados existem duas abordagens, a saber: com e sem particionamento. A seguir, apresentamos uma breve descrição.

### 4.1.1. Sem particionamento

Consideramos, para esta abordagem, o universo de registros completo sem realizar nenhum tipo de pré-processamento. Desta forma, temos como caminho inicial a comparação dois a dois de todos os registros contidos na base de dados.

A aplicação das regras para identificação de duplicatas entre todos os registros tem complexidade $\mathbf{O(n^2)}$, onde **n** é a quantidade de registros na base.

### 4.1.2. Com particionamento

Neste caso adicionamos uma etapa inicial de pré-processamento dos registros para segmentar o universo. Este pré-processamento da base reduz a complexidade teórica do processo para $\mathbf{O(Ckm^2)}$, onde **k** é o número de grupos formados, **C** é o numero de iterações do algoritmo de agrupamento e **m** é a quantidade de itens por grupo, sendo *k* e *m* muito menores que *n*.

A comparação dois a dois é, agora, realizada com exclusividade dentro de cada grupo, não havendo comparação entre registros de grupos diferentes.

## 4.2. Base verdade

Foi criada, para fins de comparação, uma base verdade a partir da aplicação dois a dois das regras de decisão estabelecidas para identificar duplicatas.

Esta comparação entre todos os registros gerou, como resultado final, o resultado ótimo, ou seja, o conjunto completo de registros duplicados de acordo com as regras pré-estabelecidas.

Além do resultado ótimo, a criação da base verdade forneceu o tempo total gasto para identificar as duplicatas. Esse tempo total, junto com o resultado ótimo, será utilizado para comparação com os resultados obtidos utilizando pré-processamento para segmentação do universo inicial.

## 4.3. Base de teste

A base de teste é o resultado da aplicação dois a dois das regras estabelecidas para identificação de registros duplicados exclusivamente dentro de cada um dos grupos gerados pelo pré-processamento. O resultado final é, portanto, a união dos resultados de cada grupo.

## 4.4. Erro

O objetivo deste trabalho é determinar o conjunto algoritmo-modelagem que agrupe os registros apresentando ganho significativo no tempo de processamento, não permitindo um erro superior a 5%.

O erro é descrito pela Figura 6.

$\varepsilon$ = $|\Delta|$ / |**Base de teste**|

onde

$\Delta$ = Base verdade $\cup$ Base de teste - Base verdade $\cap$ Base de teste

**Figura 6: Erro**

## 4.5. Estratégia vetorial

A seguir serão descritos os algoritmos e representações dos dados usadas para executar o pré-processamento de acordo com o apresentado anteriormente.

### 4.5.1. Agrupamentos

Os algoritmos que serão apresentados são bem difundidos e conhecidos no meio cientifico desta forma, poderemos mostrar o ganho real obtido com a inclusão desse tipo de pré-processamento ao processo de *data quality* e o potencial de melhoria existente com a exploração de novas técnicas e algoritmos.

#### 4.5.1.1. Algoritmos

São apresentados a seguir os três algoritmos para agrupamento estudados. Para cada um é fornecida uma descrição além do pseudocódigo.

##### 4.5.1.1.1. K-Means

Hartigan e Wong (1979) mostraram ao mundo o hoje notório *K-Means*, um dos mais bem sucedidos algoritmos de agrupamento de dados. Conhecido por sua simplicidade e eficiência, o *K-Means* é hoje utilizado nos mais diversos campos de estudo. Da mesma forma que o GNG, a ser apresentado adiante, utiliza-se de um espaço n-dimensional para modelar o conjunto de dados.

###### 4.5.1.1.1.1. Descrição

O *K-Means* posiciona os grupos no mesmo espaço e ainda da mesma forma que o GNG usando distância euclidiana como medida de proximidade ou confiança seja entre os dados ou os grupos. Para calcular a posição dos grupos, é estabelecido um centróide. O centróide é definido como o somatório de cada dimensão do espaço dividido pela quantidade de itens no grupo.

Para se associar um item a um grupo, é escolhido o grupo com a menor distância euclidiana até o dado. O algoritmo segue sendo executado iterativamente percorrendo todos os dados e todos os grupos fazendo as trocas de grupos quando necessário. O algoritmo pára quando a quantidade de trocas de grupo é menor do que um valor previamente estabelecido.

#### 4.5.1.1.1.2. Pseudocódigo

A Figura 7 apresenta o pseudocódigo do *K-Means*.

```
Inicia os k grupos aleatoriamente
Calcula o centróide de cada grupo
Enquanto houver modificações nos grupos
        Usa o centróide estimado para classificar os exemplos
        Para i de 1 to k
                Calcula o novo centróide
        fim_para
fim_enquanto
```

**Figura 7: Pseudocódigo para o *K-Means***

### 4.5.1.1.2. K-Medoid

Kaufman e Rousseeuw (1987) apresentaram uma variação do algoritmo *K-Means* chamada *K-Medoid* ou, como é comumente chamado em nosso idioma, *K-Medóide*. Esta técnica se mostra mais robusta e é menos sensível a existência de *outliers*.

Como exige muito tempo computacional para ser executada em sua versão original, dificultando a aplicação em grandes bases de dados, diversas abordagens foram sugeridas ao longo dos anos (Ester *et al*. 1995, Chu *et al*. 2002, Zhang e Couloigner 2005).

#### 4.5.1.1.2.1. Descrição

O *K-Medóide* é utilizado em uma grande variedade de áreas: psicologia e outras ciências sociais, biologia, estatística, reconhecimento de padrões, recuperação de informações, aprendizado de máquina e mineração de dados.

A diferença entre *K-Means* e *K-Medóide* está no centróide utilizado. No primeiro, o centróide é a média dos valores de todos os itens contidos no grupo, gerando uma posição no espaço que não necessariamente é correspondente a um item do conjunto de dados. No *K-Medóide* o centróide é definido como o dado com a menor distância para todos os outros. O *K-Medóide*, portanto, tem como centróide o dado mais próximo do centro calculado.

#### 4.5.1.1.2.2. Pseudocódigo

A Figura 8 apresenta o pseudocódigo do algoritmo *K-Medóide*.

```
Inicia os k grupos aleatoriamente
Escolhe o centróide de cada grupo
Enquanto houver modificações nos grupos
        Usa o centróide estimado para classificar os exemplos
        Para i de 1 to k
                Escolhe o elemento que minimiza o somatório da distância
                entre todos os elementos e atribui ao centróide
        fim_para
fim_enquanto
```

**Figura 8: Pseudocódigo para o K-Medóide**

### 4.5.1.1.3. GNG

Fritzke (1995) apresentou o GNG (*Growing Neural Gas*) que define uma rede neural incremental, agrupamento seus elementos em um grafo.

#### 4.5.1.1.3.1. Descrição

O grafo GNG é formado por *k* vértices onde cada vértice tem um vetor de pesos associados definindo sua posição no espaço e o conjunto de arestas que o liga a seus vizinhos. No agrupamento, novos vértices são introduzidos na rede até que um dado número máximo de vértices seja alcançado.

O processo de agrupamento de dados começa com dois vértices posicionados aleatoriamente no espaço e conectados por uma aresta.

Tanto os pesos, quanto a posição dos vértices são calculados iterativamente de forma gulosa. Para cada dado são determinados o vértice e vizinho mais próximos.

Neste momento estes dois vértices são conectados por uma aresta, caso ainda não haja uma entre eles. Para permitir uma maior mudança na topologia do grafo, cada aresta tem uma espécie de “idade” associada.

A cada etapa do aprendizado é somado um à idade das arestas conectadas ao vértice mais próximo do dado. Com esse mecanismo onde as arestas “envelhecem” identifica-se as inativas e as que excedem uma dada idade máxima. Desta forma são eliminadas arestas muito e nada visitadas.

Outro mecanismo que dá grande adaptabilidade ao grafo é o extermínio de vértices sem ligações com outros. Pela definição do algoritmo, a vizinhança de cada vértice é limitada a seus vizinhos topológicos.

Com isso, cada dado inserido em um vértice, o reposiciona no espaço e inclui o novo dado em seu conjunto. Adicionalmente é calculada sua nova posição e seus vizinhos são movidos por uma fração constante da distância.

A cada modificação é, portanto, reposicionada parte da topologia do conjunto de dados de cada vértice.

Não há conceito de *ranking* de vizinhança ou função de vizinhança. Todos os vizinhos topológicos são atualizados da mesma maneira. O algoritmo é muito competente para identificar rapidamente grupos, sendo usado hoje para estudos geológicos e genéticos dentre outros.

Como, porém, a topologia se move em bloco e os vértices interferem diretamente no posicionamento dos vizinhos, pode haver erro maior em dados cuja distribuição não é razoavelmente homogênea.

#### 4.5.1.1.3.2. Pseudocódigo

A Figura 9 apresenta o pseudocódigo do algoritmo GNG descrito acima.

1. Initialize the set $\mathcal{A}$ to contain two units $c_1$ and $c_2$ with reference vectors chosen randomly according to $p(\xi)$

$$\mathcal{A} = \{c_1, c_2\} \tag{5.25}$$

Initialize the connection set $\mathcal{C}$, $\mathcal{C} \subset \mathcal{A} \times \mathcal{A}$, to the empty set:

$$\mathcal{C} = \emptyset. \tag{5.26}$$

2. Generate at random an input signal $\xi$ according to $p(\xi)$.

3. Determine the winner $s_1$ and the second-nearest unit $s_2 (s_1, s_2 \in \mathcal{A})$ by

$$s_1 = \arg\min_{c \in \mathcal{A}} \|\xi - \mathbf{w}_c\| \tag{5.27}$$

and

$$s_2 = \arg\min_{c \in \mathcal{A} \setminus \{s_1\}} \|\xi - \mathbf{w}_c\|. \tag{5.28}$$

4. If a connection between $s_1$ and $s_2$ does not exist already, create it:

$$\mathcal{C} = \mathcal{C} \cup \{(s_1, s_2)\}. \tag{5.29}$$

Set the age of the connection between $s_1$ and $s_2$ to zero ("refresh" the edge):

$$\text{age}_{(s_1, s_2)} = 0. \tag{5.30}$$

5. Add the squared distance between the input signal and the winner to a local error variable:

$$\Delta \mathbf{E}_{s_1} = \|\xi - \mathbf{w}_{s_1}\|^2. \tag{5.31}$$

6. Adapt the reference vectors of the winner and its direct topological neighbors by fractions $\epsilon_b$ and $\epsilon_n$, respectively, of the total distance to the input signal:

$$\Delta \mathbf{w}_{s_1} = \epsilon_b (\xi - \mathbf{w}_{s_1}) \tag{5.32}$$

$$\Delta \mathbf{w}_i = \epsilon_n (\xi - \mathbf{w}_i) \qquad (\forall i \in N_{s_1}). \tag{5.33}$$

Thereby $N_{s_1}$ (see equation 2.5) is the set of direct topological neighbors of $s_1$.

7. Increment the age of all edges emanating from $s_1$:

$$\text{age}_{(s_1, i)} = \text{age}_{(s_1, i)} + 1 \qquad (\forall i \in N_{s_1}). \tag{5.34}$$

8. Remove edges with an age larger than $a_{max}$. If this results in units having no more emanating edges, remove those units as well.

9. If the number of input signals generated so far is an integer multiple of a parameter $\lambda$, insert a new unit as follows:

Determine the unit q with the maximum accumulated error:

$$q = \arg\max_{c \in \mathcal{A}} \mathbf{E}_c. \tag{5.35}$$

Determine among the neighbors of q the unit f with the maximum accumulated error:

$$f = \arg\max_{c \in N_q} \mathbf{E}_c. \tag{5.36}$$

Add a new unit r to the network and interpolate its reference vector from q and f.

$$\mathcal{A} = \mathcal{A} \cup \{r\}, \qquad \mathbf{w}_r = (\mathbf{w}_q + \mathbf{w}_f)/2. \tag{5.37}$$

Insert edges connecting the new unit r with units q and f, and remove the original edge between q and f:

$$\mathcal{C} = \mathcal{C} \cup \{(r, q), (r, f)\}, \qquad \mathcal{C} = \mathcal{C} \setminus \{(q, f)\}. \tag{5.38}$$

Decrease the error variables of q and f by a fraction $\alpha$:

$$\Delta \mathbf{E}_q = -\alpha \mathbf{E}_q, \qquad \Delta \mathbf{E}_f = -\alpha \mathbf{E}_f. \tag{5.39}$$

Interpolate the error variable of r from q and f:

$$\mathbf{E}_r = (\mathbf{E}_q - \mathbf{E}_f)/2. \tag{5.40}$$

10. Decrease the error variables of all units:

$$\Delta \mathbf{E}_c = -\beta \mathbf{E}_c \qquad (\forall c \in \mathcal{A}). \tag{5.41}$$

11. If a stopping criterion (e.g., net size or some performance measure) is not yet fulfilled continue with step 2.

**Figura 9: Pseudocódigo para o GNG original pelo autor.**

### 4.5.2. Representação

As modelagens apresentadas a seguir utilizam representação vetorial. O usual "saco de *tokens*" tem sua freqüência levada em consideração e, em alguns cenários, a freqüência é ponderada pela importância do *token*. São também incluídas na representação algumas características do registro tais como: quantidade de palavras, tamanho de cada palavra, tamanho do registro e quantidade de palavras.

#### 4.5.2.1. Completa

Na representação completa são incluídas no "saco de *tokens*" todas as classes com seus respectivos valores, representados pela freqüência.

| Registro: | IAN MONTEIRO NUNES | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Saco de *tokens* | | | | | | | | | | | Características do registro | | | | |
| A | E | I | O | U | M | N | T | R | S | Nº Palavras | Tamanho Registro | $Pal_1$ | $Pal_2$ | $Pal_3$ |
| 1 | 2 | 2 | 2 | 1 | 1 | 4 | 1 | 1 | 1 | 3 | 16 | 3 | 8 | 5 |

**Tabela 10: Exemplo de representação completa.**

#### 4.5.2.2. Tipos de *token*

Apresentamos a seguir os tipos de *token* utilizados no contexto deste trabalho. Para cada tipo, apresentamos um exemplo a partir da cadeia de caracteres "IAN NUNES".

##### 4.5.2.2.1. Caractere

Quando utilizamos os *tokens* por caractere, cada caractere do registro é um *token* e tem sua freqüência contabilizada. A Tabela 11 a seguir mostra sua aplicação na cadeia de caracteres utilizada como exemplo.

| Registro: | IAN NUNES | | | | |
|---|---|---|---|---|---|
| A | E | I | U | N | S |
| 1 | 1 | 1 | 1 | 3 | 1 |

**Tabela 11: Tokens por caractere.**

#### 4.5.2.2.2. Digrama

Quando utilizamos digramas como *tokens*, cada conjunto de dois caracteres em seqüência do registro é um *token* e tem sua freqüência contabilizada. A Tabela 12 a seguir mostra sua aplicação na cadeia de caracteres utilizada como exemplo.

| Registro: | IAN NUNES | | | | |
|---|---|---|---|---|---|
| IA | NA | NU | UM | NE | ES |
| 1 | 1 | 1 | 1 | 1 | 1 |

**Tabela 12: Tokens por digrama.**

#### 4.5.2.2.3. Trigrama

Quando utilizamos trigramas como *tokens*, cada conjunto de três caracteres em seqüência do registro é um *token* e tem sua freqüência contabilizada. A Tabela 13 a seguir mostra sua aplicação na cadeia de caracteres utilizada como exemplo.

| Registro: | IAN NUNES | | |
|---|---|---|---|
| IAN | NUN | UNE | NES |
| 1 | 1 | 1 | 1 |

**Tabela 13: Tokens por trigrama.**

### 4.5.2.3. Pesos

Na contagem da freqüência, normalmente o peso atribuído a cada ocorrência de um item é 1. Foi, porém, seguida uma abordagem de pesos variados para validar uma regra cognitiva que diz: as primeiras e últimas letras de cada palavra têm maior relevância na compreensão e produção de textos.

De acordo com essa regra os primeiros e últimos caracteres tiveram seus pesos modificados e inicialmente passaram a valer 2 enquanto os outros caracteres continuavam valendo 1.

A Tabela 14 a seguir exemplifica esta abordagem de pesos diferenciados para caracteres iniciais e finais.

| Registro: | NUNES | | | |
|---|---|---|---|---|
| **N** | **U** | **N** | **E** | **S** |
| 2 | 1 | 1 | 1 | 2 |

**Tabela 14: Pesos para caracteres inicial e final**

Seguindo a mesma linha foi implementada uma nova idéia que consiste em atribuir aos caracteres inicial e final o maior peso e diminuí-lo de forma constante até o centro de cada palavra.

A Tabela 15 a seguir exemplifica esta abordagem de pesos progressivos para os caracteres.

| Registro: | NUNES | | | |
|---|---|---|---|---|
| **N** | **U** | **N** | **E** | **S** |
| 3 | 2 | 1 | 2 | 3 |

**Tabela 15: Diferentes pesos até o centro**

#### 4.5.2.4. Tamanhos

Para contemplar as diversas variações possíveis nos registros textuais foram também inseridas características do registro na representação tais como: tamanhos de palavras, quantidade de palavras e o tamanho total dos registros.

#### 4.5.2.5. Heurísticas Lingüísticas

Foi desenvolvido um algoritmo baseado nos fonemas do português brasileiro como parte integrante deste trabalho. Este algoritmo teve como objetivo aumentar a acurácia dos agrupamentos de registros de um dado idioma.

Os fonemas apresentados na Tabela 16 são representados por letras e números do alfabeto romano (português brasileiro) e podem agrupar vários caracteres do alfabeto romano. O português brasileiro possui 34 fonemas, sendo 13 vogais, 19 consoantes e 2 semivogais.

| Tipo | Fonema | Características fonéticas | Exemplos |
|---|---|---|---|
| Vogais | Á | Aberta, frontal, oral, não arredondada. | **á**tomo, **a**rte |
| | Â | Semi-aberta, central, oral, não arredondada. | p**a**no, r**a**mo, l**a**nho |
| | Ã | Semi-aberta, central, nasal, não arredondada. | **an**tes, **am**plo, maç**ã**, **âm**bito, **ân**sia |
| | É | Semi-aberta, frontal, oral, não arredondada. | m**é**trica, p**e**ça. |
| | Ê | Semi-fechada, frontal, oral, não arredondada. | m**e**do, p**ê**ssego |
| | ẽ | Semi-fechada, frontal, nasal, não arredondada. | s**em**pre, **êm**bolo, c**en**tro, conc**ên**trico, t**êm**, tamb**ém**. |
| | Ó | Semi-aberta, posterior, oral, arredondada. | **ó**tima, **o**va. |
| | Ô | Semi-fechada, posterior, oral, arredondada. | r**o**lha, av**ô** |
| | Õ | Semi-fechada, posterior, nasal, arredondada. | **om**bro, **on**tem, c**ôm**puto, c**ôn**sul |
| | I | Fechada, frontal, oral, não arredondada. | **i**tem, silv**í**cola |
| | Ĩ | Fechada, frontal, nasal, não arredondada. | s**im**ples, s**ím**bolo, t**in**ta, s**ín**crono |
| | U | Fechada, posterior, oral, arredondada. | **u**va, **ú**tero |
| | Ũ | Fechada, posterior, nasal, arredondada. | alg**um**, pl**úm**beo, n**un**ca, ren**ún**cia, m**u**ito |
| Semivogais | Y | Oral, palatal, sonora | u**i**vo, m**ãe**, ár**e**a, t**êm**, tamb**ém**, viv**em** |
| | W | Oral, velar, sonora | a**u**tomático, móve**l**, p**ão**, freq**ü**ente, fal**am** |
| Consoantes | M | Nasal, sonora, bilabial | **M**arca |
| | N | Nasal, sonora, alveolar | **N**ervo |
| | Ñ | Nasal, sonora, palatal | Arra**nh**ado |
| | B | Oral, oclusiva, bilabial, sonora | **B**arco |
| | P | Oral, oclusiva, bilabial, surda | **P**ato |
| | D | Oral, oclusiva, linguodental, sonora | **D**ata |
| | T | Oral, oclusiva, linguodental, surda | **T**elha |
| | G | Oral, oclusiva, velar, sonora | **G**ato |
| | K | Oral, oclusiva, velar, surda | **C**arro, **q**uanto |
| | V | Oral, fricativa, labiodental, sonora | **V**ento |

| Tipo | Fonema | Características fonéticas | Exemplos |
|---|---|---|---|
| Consoantes | F | Oral, fricativa, labiodental, surda | **F**arelo |
| | Z | Oral, fricativa, alveolar, sonora | **z**ero, ca**s**a, e**x**alar |
| | S | Oral, fricativa, alveolar, surda | **s**eta, **c**ebola, espe**ss**o, e**xc**esso, a**ç**úcar, au**x**ílio, a**sc**eta |
| | J | Oral, fricativa, pós-alveolar, sonora | **g**elo, **j**arro |
| | X | Oral, fricativa, pós-alveolar, surda | **x**arope, **ch**uva |
| | R | Oral, vibrante, sonora, uvular. | **r**ato, ca**rr**oça |
| | R | Oral, vibrante, sonora, alveolar. | Va**r**iação |
| | Λ | Oral, lateral aproximante, sonora, palatal. | Cava**lh**eiro |
| | L | Oral, lateral aproximante, sonora, alveolar | **L**uz |

**Tabela 16: Fonemas do português brasileiro**

### 4.5.3. Experimentos

Os experimentos foram realizados com diferentes conjuntos de dados de origens diferentes. Todos os conjuntos de dados utilizados são de propriedade de empresas de diferentes setores da economia: financeiro, de transformação, e comércio varejista.

#### 4.5.3.1. *Corpus*

As bases de dados usadas para teste possuíam aproximadamente 50 mil registros, 100 mil registros e 200 mil registros. Sendo todos os dados oriundos de cadastros de pessoas e empresas, os atributos disponíveis comumente são os mesmos e apresentam características semelhantes.

Os experimentos têm como objetivo avaliar o desempenho dos algoritmos aplicados a um *framework* utilizando dados reais, que sejam de fato necessários no dia-a-dia de empresas. Sendo assim, as bases de dados escolhidas são representativas e têm as características necessárias para generalizar uma base corporativa relevante qualquer.

#### 4.5.3.2. Metodologia

Os experimentos foram realizados em duas etapas distintas. Durante a realização dos experimentos, as bases de dados de origens e tamanhos diferentes se comportaram de forma semelhante, não apresentando diferenças relevantes.

##### 4.5.3.2.1. Primeira etapa

Inicialmente foram realizados experimentos com os algoritmos *K-means* e GNG. Isto porque ambos trabalham com espaços n-dimensionais de forma semelhante e todos os experimentos poderiam ser realizados nas mesmas bases usando basicamente o mesmo *corpus* e pré-processamentos para adição de dimensões ou tratamento prévio dos dados permitindo comparações mais diretas.

As primeiras a serem testadas foram as bases simplificadas de 50 mil e 200 mil registros. Para estas, foi gerada uma "base verdade" com alguns milhões de registros para realizar a validação dos resultados. A geração desta "base verdade" foi feita através da aplicação do algoritmo de distância de edição dois-a-dois entre todos os elementos das bases de teste sendo o resultado dessas distâncias foi armazenado.

A base original foi, então, processada a fim de gerar os agrupamentos. Ao final do agrupamento a base gerada pelos conjuntos algoritmo-modelagem estudados, teve todos os seus elementos de cada grupo submetidos à mesma medida de distância de edição, gerando então uma base final.

A base final foi comparada com a base verdade gerando a taxa de erro dos agrupamentos que é relatada na Tabela 17 a seguir.

| Experimentos - Primeira rodada | Erro |
|---|---|
| *K-Means* | 30,02% |
| *K-Means* com dimensão extra com o tamanho do registro textual | 25,13% |
| *K-Means* com aplicação de algoritmo fonético | 27,27% |
| *K-Means* com aplicação de algoritmo fonético e dimensão extra com o tamanho do registro textual | 16,88% |
| GNG com aplicação de algoritmo fonético e trigramas | 19,08% |

| Experimentos - Primeira rodada | Erro |
|---|---|
| GNG com aplicação de algoritmo fonético | 22,31% |
| GNG com trigramas e peso 2 nas letras em extremidades de palavras | 23,74% |
| GNG com trigramas | 21,28% |
| GNG com aplicação de heurística para redução de alfabeto e trigramas | 19,71% |

**Tabela 17: Taxas de erro da primeira rodada da primeira etapa**

A evolução dos primeiros experimentos foi rápida e consistente em todos os algoritmos, indicando quais testes deveriam ser priorizados nos momentos seguintes.

Após diversos testes, a barreira de 15% parecia quase intransponível para as taxas de erro obtidas na comparação com a base verdade inicial. A base verdade foi porém, gerada de forma mais abrangente do que o domínio de estudo, indicando todas as duplicatas a partir da similaridade por distância de edição entre todos os itens do conjunto de dados original.

Desta forma, essa base serviu apenas para direcionar os próximos passos e indicava que bons resultados poderiam ser obtidos através de (a)uma base verdade específica para o domínio desejado e (b)algumas modificações na modelagem inicial. Foi então gerada uma nova base verdade (completa), sendo que dessa vez os modelos de agrupamento foram integrados ao *framework* utilizado para execução dos processos de *data quality*. Isso permitiu que o agrupamento tivesse sua avaliação a partir da identificação de duplicatas executada no *framework*.

Cada modelagem diferente gerou grupos de dados que somados representavam a base inteira, ou seja, identificar as duplicatas em todos os grupos separadamente é equivalente a identificar no conjunto inicial completo. Essa nova base verdade gerou os resultados apresentados na Tabela 18.

| Experimentos – Segunda rodada | Erro |
|---|---|
| *K-Modóide* permitindo o registro em mais de 1 grupo agrupando consoantes | 0,04% |
| *K-Modóide* | 0,47% |
| GNG | 0,30% |

**Tabela 18: Taxas de erro da segunda rodada da primeira etapa**

O objetivo inicial dos experimentos era atingir um percentual de erro suficientemente baixo para permitir que esses modelos sejam incorporados a processos de *data quality* sem perda de qualidade ou comprometimento do resultado. O real desafio era garantir uma taxa de erro baixa (inferior a 5%) e apresentar significativo ganho de tempo na execução.

Esta rodada mostrou claramente que o resultado obtido é muito parecido com o resultado esperado, ou seja, é possível reduzir o tamanho do problema inicial sem abrir mão da qualidade do resultado final.

Esses experimentos foram realizados ainda com a base simplificada. A base simplificada consiste apenas no primeiro nó das empresas e pessoas, e não possui mais nenhum dado para comparação ou geração de regras que modifiquem o conjunto de registros semelhantes.

#### 4.5.3.2.2. Segunda etapa

Na segunda etapa de experimentos os tempos de execução foram computados e as variações de parâmetros testadas anteriormente foram usadas e ampliadas. Essa ampliação tem como objetivo permitir que tenhamos um panorama completo do que pode ser realmente exeqüível com a melhor relação tempo-desempenho ao final dos experimentos.

Nesta etapa foram inseridos os dados completos nas bases de dados que, para este estudo, se limitam a nome, endereços, telefones, e emails. Os testes foram realizados com foco nos conjuntos algoritmo-modelagem mais bem sucedidos nos experimentos da primeira etapa.

Em todos os experimentos, todas as bases se comportaram de forma similar, indicando que aplicar os algoritmos a bases menores ou maiores não implica em perda ou ganho de confiança no resultado final. Todos os experimentos tiveram resultados bastante homogêneos em relação à quantidade de acertos e erros.

Com isso em mente, após a execução de testes redundantes nas diferentes bases, os experimentos passaram a ser realizados em apenas uma delas. A base de dados escolhida foi a de aproximadamente 50 mil registros que, por se comportar de forma similar às outras quando aplicada aos diferentes algoritmos, permite uma boa generalização dos resultados obtidos em todos os momentos do processo.

Analisamos a seguir os experimentos baseados nos resultados apresentados para os dados completos e com a validação completa, ou seja, no cenário onde os conjuntos algoritmo-modelagem realmente serão empregados.

Avaliamos todos os algoritmos separadamente e diversos cortes nos dados experimentais de cada um permitindo a observação de diversos detalhes quando passamos de uma modelagem para outra.

#### 4.5.3.3. Resultados

As seguintes colunas são utilizadas na demonstração dos resultados:

- **Modelo** – indicando a representação dos textos em dimensões de um espaço n-dimensional aplicada no experimento. (“Caractere” – cada letra é uma dimensão, “Digrama” – cada 2 letras formam uma dimensão, “Trigrama” – a cada três letras uma dimensão é formada);

- **Peso** – indicando qual o peso de cada conjunto de letras ou letra usada para criar as dimensões baseadas em seu posicionamento dentro da palavra. (“Normal” – todas as dimensões têm o mesmo peso, “Extremidades” – as primeiras e últimas dimensões de cada palavra ganham um peso maior, “Progressivo” – o peso muda progressivamente de acordo com a proximidade da extremidade da palavra);

- **Dimensões extra** – indica quais colunas foram adicionadas nessa modelagem especificamente, essas colunas sempre se referem à contagem de caracteres e palavras no campo em questão;

- **Tempo** – indica a duração em segundos do experimento;

- **Acurácia** – indica o percentual de acerto em relação à base-verdade.

#### 4.5.3.4. K-Means

A Tabela 19 a seguir contém os resultados para os experimentos com o algoritmo *K-Means* utilizando caracteres como *tokens*.

| Modelo Caracter | | | |
|---|---|---|---|
| **Peso** | **Dimensões extra** | **Tempo** | **Acurácia** |
| Progressivo | Caracteres por palavra | 591 | 96,19% |
| Progressivo | Nº de palavras | 576 | 96,08% |
| Progressivo | Nenhuma | 593 | 96,05% |
| Progressivo | Nº de caracteres e palavras | 853 | 95,90% |
| Normal | Caracteres por palavra | 584 | 95,76% |
| Extremidades | Caracteres por palavra | 590 | 95,73% |
| Normal | Nenhuma | 658 | 95,34% |
| Normal | Nº de caracteres | 571 | 95,25% |
| Extremidades | Nº de caracteres e palavras | 715 | 95,17% |
| Extremidades | Nº de caracteres | 563 | 95,14% |
| Extremidades | Nº de palavras | 746 | 95,03% |
| Normal | Nº de caracteres e palavras | 570 | 94,95% |
| Extremidades | Nenhuma | 591 | 94,95% |
| Normal | Nº de palavras | 502 | 93,19% |

**Tabela 19: Resultados para K-Means com caracteres como tokens**

Começando a analisar os resultados observamos que a modelagem por caracteres, com todas as variações não fonéticas possíveis, apresenta o melhor resultado quando usamos o peso progressivo e a quantidade de caracteres por palavra.

O pior desempenho é observado quando não utilizamos pesos e também usamos a quantidade de palavras.

A Tabela 20 a seguir contém os resultados para os experimentos com o algoritmo *K-Means* utilizando digramas como *tokens*.

| **Modelo Digrama** | | | |
|---|---|---|---|
| **Peso** | **Dimensões extra** | **Tempo** | **Acurácia** |
| Progressivo | Caracteres por palavra | 1032 | 97,48% |
| Extremidades | Nenhuma | 1388 | 97,37% |
| Extremidades | Caracteres por palavra | 1198 | 97,29% |
| Extremidades | Nº de palavras | 1234 | 97,28% |
| Progressivo | Nº de palavras | 1383 | 97,28% |
| Progressivo | Nº de caracteres | 1044 | 97,26% |
| Normal | Nenhuma | 1259 | 97,25% |
| Progressivo | Nº de caracteres e palavras | 1205 | 97,15% |
| Normal | Caracteres por palavra | 1200 | 97,14% |
| Progressivo | Nenhuma | 1604 | 97,14% |
| Normal | Nº de palavras | 1297 | 97,12% |
| Extremidades | Nº de caracteres | 1184 | 96,98% |
| Normal | Nº de caracteres e palavras | 1164 | 96,73% |
| Extremidades | Nº de caracteres e palavras | 1160 | 96,72% |
| Normal | Nº de caracteres | 1057 | 96,66% |

**Tabela 20: Resultados para K-Means com digramas como tokens**

Quando usamos a modelagem por digramas não fonética, percebemos uma grande homogeneidade nos resultados, do melhor para o pior cenário a variação é de aproximadamente 0,82%.

Novamente a mesma configuração, utilizando peso progressivo e a quantidade de caracteres por palavra, se mostra melhor que as demais, mesmo a diferença sendo pouco representativa.

O mesmo não ocorre no pior caso desse corte nos dados. O pior resultado é encontrado quando a modelagem não diferencia os pesos das dimensões e usa quantidade de caracteres como dimensão extra.

A Tabela 21 a seguir contém os resultados para os experimentos com o algoritmo *K-Means* utilizando trigramas como *tokens*.

| **Modelo Trigrama** | | | |
|---|---|---|---|
| **Peso** | **Dimensões extra** | **Tempo** | **Acurácia** |
| Progressivo | Nenhuma | 2584 | 98,51% |
| Progressivo | Nº de palavras | 2359 | 98,25% |
| Progressivo | Caracteres por palavra | 2628 | 98,22% |
| Normal | Nenhuma | 1426 | 98,17% |
| Progressivo | Nº de caracteres e palavras | 1958 | 97,93% |
| Progressivo | Nº de caracteres | 1971 | 97,91% |
| Normal | Caracteres por palavra | 1565 | 97,85% |
| Normal | Nº de palavras | 1462 | 97,83% |
| Extremidades | Nenhuma | 1999 | 97,79% |
| Extremidades | Nenhuma | 1788 | 97,79% |
| Extremidades | Nº de palavras | 1548 | 97,77% |
| Extremidades | Caracteres por palavra | 2227 | 97,57% |
| Normal | Nº de caracteres e palavras | 1795 | 97,15% |
| Extremidades | Nº de caracteres e palavras | 1531 | 97,12% |
| Extremidades | Nº de caracteres | 1363 | 97,04% |
| Normal | Nº de caracteres | 1660 | 97,01% |

**Tabela 21: Resultados para K-Means com trigramas como tokens**

A modelagem por trigramas se mostrou a mais competente e, assim como na modelagem por digramas, a diferença entre pior e melhor caso não é relevante mantendo-se em cerca de 1,50%. O percentual de acerto é extremamente alto e pouco pode ser melhorado em relação à acurácia do algoritmo.

Diferente dos anteriores, o melhor caso é sem a inclusão de dimensão extra. O peso progressivo, entretanto, ainda é o escolhido. Novamente para o pior caso ficou o peso dos caracteres normal e a quantidade de caracteres como dimensão extra.

#### 4.5.3.5. GNG

O GNG é um algoritmo que, assim como o *K-Means*, utiliza uma representação espacial-dimensional dos dados para executar o agrupamento. Segue as mesmas configurações e modelagens propostas para o *K-Means*, salvo pela inclusão da possibilidade de trabalhar com os dados "fonetizados" para comparar com o desempenho dos dados conforme adquiridos.

A tabela a seguir contém os resultados para os experimentos com o algoritmo GNG utilizando caracteres como *tokens*.

| Modelo | Peso | Fonético | Tempo | Acurácia |
|---|---|---|---|---|
| Caractere | Extremidades | Sim | 473 | 95,54% |
| Caractere | Nenhum | Sim | 464 | 95,39% |
| Caractere | Nenhum | Não | 535 | 94,99% |
| Caractere | Extremidades | Não | 461 | 94,57% |

**Tabela 22: Resultados para GNG com caracteres como tokens**

O melhor caso identificado nesta rodada foi com a fonetização dos registros e com a inclusão de pesos nas primeiras e ultimas dimensões geradas para cada palavra.

A modelagem por caractere apresenta um desempenho melhor quando a fonética é adicionada, porém, a diferença é inferior a 1,00% o que é pouco frente aos outros resultados obtidos.

A tabela a seguir contém os resultados para os experimentos com o algoritmo GNG utilizando digramas como *tokens*.

| Modelo | Peso | Fonético | Tempo | Acurácia |
|---|---|---|---|---|
| Digrama | Extremidades | Não | 1589 | 97,14% |
| Digrama | Nenhum | Não | 831 | 96,94% |
| Digrama | Nenhum | Sim | 1538 | 96,39% |
| Digrama | Extremidades | Sim | 936 | 96,12% |

**Tabela 23: Resultados para GNG com digramas como tokens**

A modelagem por digramas surpreende quando apresenta um resultado onde a fonetização fica com os últimos lugares. Adicionalmente a modelagem sem considerar fonemas tem desempenho bastante superior se considerarmos que na modelagem anterior a diferença entre elas foi aproximadamente 1,00% e agora a diferença se inverteu.

Isso mostra que nessas modelagens houve uma estagnação dos algoritmos que usaram os textos fonetizados, enquanto o texto normal permitiu uma evolução de mais de 2,10% na acurácia.

A tabela a seguir contém os resultados para os experimentos com o algoritmo GNG utilizando trigramas como *tokens*.

| Modelo | Peso | Fonético | Tempo | Acurácia |
|---|---|---|---|---|
| Trigrama | Nenhum | Não | 4068 | 98,92% |
| Trigrama | Extremidades | Não | 1829 | 96,84% |
| Trigrama | Extremidades | Sim | 2026 | 95,73% |
| Trigrama | Nenhum | Sim | 2054 | 95,40% |

**Tabela 24: Resultados para GNG com trigramas como tokens**

A melhor modelagem é, portanto, a que não leva em consideração nem os textos fonetizados nem pesos nas dimensões. Encontramos nesta modelagem por trigramas o mesmo cenário ocorrido na modelagem por digramas quando os textos fonetizados apresentaram uma involução em relação aos textos originais.

O resultado nos testes utilizando o texto original foi superior apresentando uma melhora na acurácia de quase 1,80%.

#### 4.5.3.6. K-Medóide

Uma versão diferente de K-Medóide é usada nesse trabalho onde o centróide é composto por três itens do universo o que gera a expectativa tanto de redução de *outliers* quanto de maior acurácia no resultado final.

A tabela a seguir contém os resultados para os experimentos com o algoritmo K-Medóide.

| Agrupa caracteres | Fonético | Centróides | Tempo | Acurácia |
|---|---|---|---|---|
| Vogais | Não | 3 | 8131 | 96,80% |
| Vogais e consoantes | Não | 3 | 7098 | 96,56% |
| Nenhum | Sim | 3 | 2108 | 96,50% |
| Nenhum | Não | 3 | 3729 | 96,49% |
| Todos os caracteres | Não | 3 | 8419 | 96,39% |
| Consoantes | Não | 3 | 7613 | 96,33% |
| Vogais | Sim | 3 | 5439 | 96,27% |
| Todos os caracteres | Sim | 3 | 5216 | 96,05% |
| Vogais e consoantes | Sim | 3 | 4225 | 95,91% |
| Consoantes | Sim | 3 | 4430 | 95,88% |

**Tabela 25: Resultados para K-Medóide**

É novamente surpreendente que quando os textos são fonetizados apresentam resultados piores, uma vez que intuitivamente ao fonetizar e incluir mais informações nas classes o resultado seria melhor. Nesse caso, o desempenho de todos é semelhante, a acurácia varia pouco de uma modelagem para outra, tendo apenas 0,92% de diferença entre o melhor e o pior.

Como não apresenta uma variação significativa entre os parâmetros, escolheremos o mais rápido para futuras comparações entre modelos.

## 4.6. Estratégia não-vetorial

A seguir serão descritos os algoritmos e representações dos dados usadas para executar o pré-processamento de acordo com o apresentado anteriormente.

### 4.6.1. Agrupamento

Para contrapor a estratégia anterior que é extremamente conservadora no que tange as técnicas utilizadas. O caminho seguido aqui é mais ousado já que o objetivo é propor um olhar diferente sobre os algoritmos e modificá-los gerando uma nova forma de agrupar registros textuais.

Esta estratégia mostrará que também é possível obter bons resultados através da inclusão de conceitos estatísticos simples e de uma nova medida de confiança para a proximidade entre os registros.

#### 4.6.1.1. Algoritmos

Como os dados agrupados são textuais é natural que uma abordagem puramente textual seja inserida para comparação. Essa abordagem utiliza apenas a comparação entre textos para agrupar os itens.

##### 4.6.1.1.1. Modóide

Como o problema em questão é puramente textual, uma abordagem não dimensional pode ser proposta de forma natural. Essa abordagem surgiu a partir da observação de que um centro estatístico pode ser gerado observando a freqüência dos registros, das palavras, das letras, ou ainda, de conjuntos de caracteres contidos nos registros.

Dessa forma, para a construção desse centro representante de cada grupo, a freqüência dos itens deve ser observada e para cada caso um centro diferente deve ser levado em consideração. Em todos os casos, a média da quantidade de palavras sempre foi assumida como a quantidade de palavras do registro construído que representasse o centro do grupo.

##### 4.6.1.1.2. Medóide

Uma opção é utilizar um medóide na sua definição original, o registro que minimiza a distância para todos os demais registros no *corpus*.

A descoberta de qual registro representa o medóide implica em calcular as distâncias entre todos os registros do grupo em questão combinados dois-a-dois. A identificação desse registro é, portanto, custosa demais. As aproximações abaixo têm como objetivo diminuir o tempo total utilizado para descobrir o medóide sem apresentar perda de acurácia nos agrupamentos.

#### 4.6.1.1.2.1. Pseudocodigo

O pseudocódigo para o *K-Modóide* é apresentado na Figura 10.

```
Inicia os k grupos aleatoriamente
Calcula o centróide de cada grupo
Enquanto houver modificações nos grupos
        Usa o centróide estimado para classificar os exemplos
        Para i de 1 to k
                Calcula o novo centróide
        fim_para
fim_enquanto
```

**Figura 10: Pseudocódigo para o K-Modóide**

### 4.6.2. Representação

Como a abordagem proposta é puramente textual, a representação dos registros é dada pelo texto do item. Essa abordagem exige que o centróide seja textual e construído a partir dos dados do domínio.

### 4.6.3. Centróide

O centróide usado no processo de agrupamento deve ser representado da mesma forma que os itens da base de dados. Abaixo são descritas algumas abordagens estatísticas para a construção de um centróide textual válido e representativo.

#### 4.6.3.1. Seleção estatística

#### 4.6.3.1.1. Moda posicional

Nessa abordagem é contabilizada a freqüência de cada caractere em cada posição dos itens da base.

Para a construção do centróide levamos em consideração a moda de cada caractere em cada posição e as médias dos tamanhos de cada palavra, assim como a média de palavras por registro.

O centróide é construído para cada grupo assim, quando os itens do grupo são inseridos ou excluídos, o centróide é modificado seguindo a mesma regra.

Para exemplificar tomemos os dados da Tabela 26 a seguir:

| Itens no Grupo |
|---|
| IAN NUNES |
| IVAN NUNES |
| IRAN MUNIZ |
| IGOR PERES |
| IANIRA NURBIN |
| JAN JANKOVIC |
| ANA KOVAC |

**Tabela 26: Dados para exemplo**

Com os dados da Tabela 26, a matriz de freqüência total do registro é apresentada na Tabela 27, a seguir.

| Geral | A | B | C | E | G | I | J | K | M | N | O | P | R | S | U | V | Z | _ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1ª Posição** | 1 | - | - | - | - | 5 | 1 | - | - | - | - | - | - | - | - | - | - | - |
| **2ª Posição** | 3 | - | - | - | 1 | - | - | - | - | 1 | - | - | 1 | - | - | 1 | - | - |
| **3ª Posição** | 3 | - | - | - | - | - | - | - | 3 | - | 1 | - | - | - | - | - | - | - |
| **4ª Posição** | - | - | - | - | - | 1 | - | - | - | 2 | - | - | 1 | - | - | - | - | 3 |
| **5ª Posição** | - | - | - | - | - | - | 1 | 1 | - | 1 | - | - | 1 | - | - | - | - | 3 |
| **6ª Posição** | 2 | - | - | - | - | - | - | - | 1 | 1 | 1 | 1 | - | - | 1 | - | - | - |
| **7ª Posição** | - | - | - | 1 | - | - | - | - | - | 2 | 1 | - | - | - | 2 | - | - | 1 |
| **8ª Posição** | 1 | - | - | 1 | - | - | - | 1 | - | 3 | - | - | 1 | - | - | - | - | - |
| **9ª Posição** | - | - | 1 | 2 | - | 1 | - | - | - | - | 1 | - | - | 1 | 1 | - | - | - |
| **10ª Posição** | - | - | - | - | - | - | - | - | - | - | - | - | 1 | 2 | - | 1 | 1 | - |
| **11ª Posição** | - | 1 | - | - | - | 1 | - | - | - | - | - | - | - | - | - | - | - | - |
| **12ª Posição** | - | - | 1 | - | - | 1 | - | - | - | - | - | - | - | - | - | - | - | - |
| **13ª Posição** | - | - | - | - | - | - | - | - | - | 1 | - | - | - | - | - | - | - | - |

**Tabela 27: Matriz de freqüência total**

A freqüência total do registro não deve, porém, ser levada em consideração, devendo ser considerada a freqüência de cada caractere para posição em cada palavra.

Ou seja, para o cálculo da primeira palavra do centróide, deve ser levada em consideração apenas a matriz de freqüência das primeiras palavras dos registros do grupo, para a geração da segunda palavra, a matriz das segundas palavras, e assim por diante.

Vemos na Tabela 28 a seguir a matriz de freqüências para a primeira palavra.

| 1ª Palavra | A | G | I | J | N | O | R | V | Moda |
|---|---|---|---|---|---|---|---|---|---|
| 1ª Posição | 1 | - | 5 | 1 | - | - | - | - | I |
| 2ª Posição | 3 | 1 | - | - | 1 | - | 1 | 1 | A |
| 3ª Posição | 3 | - | - | - | 3 | 1 | - | - | A/N |
| 4ª Posição | - | - | 1 | - | 2 | - | 1 | - | N |
| 5ª Posição | - | - | - | - | - | - | 1 | - | R |
| 6ª Posição | 1 | - | - | - | - | - | - | - | A |

**Tabela 28: Matriz de freqüência para a primeira palavra**

E na Tabela 29 a matriz de freqüências para a segunda palavra.

| 2ª Palavra | A | B | C | E | I | J | K | M | N | O | P | R | S | U | V | Z | Moda |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1ª Posição | - | - | - | - | - | 1 | 1 | 1 | 3 | - | 1 | - | - | - | - | - | N |
| 2ª Posição | 1 | - | - | 1 | - | - | - | - | - | 1 | - | - | - | 4 | - | - | U |
| 3ª Posição | - | - | - | - | - | - | - | - | 4 | - | - | 2 | - | - | 1 | - | N |
| 4ª Posição | 1 | 1 | - | 3 | 1 | - | 1 | - | - | - | - | - | - | - | - | - | E |
| 5ª Posição | - | - | 1 | - | 1 | - | - | - | - | 1 | - | - | 3 | - | - | 1 | S |
| 6ª Posição | - | - | - | - | - | - | - | - | 1 | - | - | - | - | - | 1 | - | O/V |
| 7ª Posição | - | - | - | - | 1 | - | - | - | - | - | - | - | - | - | - | - | I |
| 8ª Posição | - | - | 1 | - | - | - | - | - | - | - | - | - | - | - | - | - | C |

**Tabela 29: Matriz de freqüência para segunda palavra**

A partir das matrizes acima, para o nosso exemplo, o registro que representa o centro desse grupo é “IAAN NUNOSV”. Cada registro do exemplo tem em média 2 palavras onde a primeira tem, em média, aproximadamente 4 caracteres e a segunda, aproximadamente 6 caracteres.

#### 4.6.3.1.1.1. Moda do registro inteiro

Da mesma forma que no item anterior, a moda do registro também deve ser avaliada como *benchmark* para os outros métodos.

Seguindo a mesma idéia de utilizar a estatística simplificada para obter melhores resultados no agrupamento dos registros, nessa abordagem o registro com a maior freqüência é escolhido e selecionado como centro do grupo em questão. Assim como no caso anterior, a medida de confiança nessa modelagem é a distância de edição.

Essa modelagem é semelhante aos *Medóides*, porém, nesta não há a obrigação de minimizar a distância para os outros itens do grupo, apenas, a única restrição que deve ser respeitada é a moda dentre todos os registros.

Usemos como exemplo os dados da Tabela 30 a seguir.

| Itens no Grupo |
|---|
| IAN NUNES |
| IVAN NUNES |
| IRAN MUNIZ |
| IGOR PERES |
| IANIRA NURBIN |
| JAN JANKOVIC |
| ANA KOVAC |

**Tabela 30: Moda de todo o registro – dados originais**

Neste caso, como não há um único registro que represente a moda, foi necessário alterar o conjunto de dados para exemplificar melhor. Assim temos os registros da Tabela 31 a seguir.

| Itens no Grupo |
|---|
| IAN NUNES |
| IAN NUNES |
| IRAN MUNIZ |
| IGOR PERES |
| IANIRA NURBIN |
| JAN JANKOVIC |
| ANA KOVAC |

**Tabela 31: Moda de todo o registro – novos dados**

Na tabela acima, o registro “IAN NUNES” representa a moda dos dados no conjunto e é o centro do grupo.

#### 4.6.3.1.1.2. Moda de cada palavra e média de palavras

Seguindo a linha de modelos estatísticos simplificados, temos aqui um uma abordagem intermediária das duas anteriores. Aqui consideramos a moda de cada palavra do registro e, para formar o registro final, consideramos a média de palavras por registro. Ou seja, fazemos a moda estatística posicionalmente em relação às palavras do registro e não conforme citado anteriormente, utilizando a freqüência dos caracteres utilizados.

A Tabela 32 a seguir mostra os dados usados como exemplo.

| Itens no Grupo |
|---|
| IAN NUNES |
| IVAN NUNES |
| IRAN MUNIZ |
| IGOR PERES |
| IANIRA NURBIN |
| JAN JANKOVIC |
| ANA KOVAC |

**Tabela 32: Dados para exemplo**

Os registros têm, em média, duas palavras e para cada palavra a matriz de freqüências é mostrada na Tabela 33 a seguir.

| Primeira palavra | |
|---|---|
| IAN | 1 |
| IVAN | 1 |
| IRAN | 1 |
| IGOR | 1 |
| IANIRA | 1 |
| JAN | 1 |
| ANA | 1 |

| Segunda palavra | |
|---|---|
| NUNES | 2 |
| MUNIZ | 1 |
| PERES | 1 |
| NURBIN | 1 |
| JANKOVIC | 1 |
| KOVAC | 1 |

**Tabela 33: Matriz de freqüências**

Nas tabelas acima, os valores das freqüências de cada palavra estão apresentados. Nesse caso, não teríamos como escolher uma palavra do primeiro grupo, já no segundo grupo “NUNES” seria escolhido sem maiores problemas.

#### 4.6.3.1.2. Medóide

Uma opção é utilizar um medóide na sua definição original, o registro que minimiza a distância para todos os demais registros no *corpus*.

A descoberta de qual registro representa o medóide implica em calcular as distâncias entre todos os registros do grupo em questão combinados dois-a-dois. Conseqüentemente a identificação desse registro é custosa demais. As aproximações abaixo têm como objetivo diminuir o tempo total utilizado para descobrir o medóide sem apresentar perda de acurácia nos agrupamentos.

##### 4.6.3.1.2.1. Aleatório

Para selecionar um bom candidato a medóide o procedimento adotado nessa abordagem consiste em escolher *M* registros aleatoriamente e calcular qual o melhor candidato dentre esses *M* registros, e assumi-lo como medóide.

##### 4.6.3.1.2.2. Múltiplos

Já para selecionar *N* bons candidatos a medóide o procedimento adotado é extremamente semelhante ao utilizado no registro aleatório único. A diferença é que ao invés de escolher o melhor registro, os *N* melhores registros são escolhidos dentre os *M* registros aleatórios pré-selecionados.

#### 4.6.3.1.3. Combinando Medóide e Modóide

Com a junção das duas abordagens para geração do centróide e de suas respectivas variações com o objetivo de gerar grupos mais consistentes e com maior acurácia.

O objetivo dessa abordagem é descobrir se há melhora ao se combinar diversas modelagens e caso haja, quais delas geraram os melhores resultados. Um comparativo deve ser traçado mostrando o acréscimo no tempo necessário para agrupar os registros e deve-se avaliar se os resultados são promissores ou não.

#### 4.6.3.2. Medidas de confiança

O centro e os itens são agora representados por um registro textual. A medida de semelhança entre os registros é modificada. Nesse ponto, é feita a opção por não usar os algoritmos vetoriais e sim um algoritmo igualmente conhecido e estudado, o algoritmo que mede a distância de edição usando o método de Levenshtein.

Foram feitas algumas pequenas modificações no método de Levenshtein que permitem a medição percentual da proximidade entre dois registros.

Essa medida de semelhança é considerada uma das melhores maneiras de se comparar dois registros textuais e diferenciá-los precisamente. A partir deste fato é justificável o esforço para inserir essa medida de confiança nas avaliações expostas nesse trabalho, uma medida tão importante e confiável deve ser capaz de contribuir de forma relevante para os estudos em agrupamentos de registros textuais.

#### 4.6.3.3. Heurísticas Lingüísticas

Da mesma forma que na primeira estratégia apresentada, a utilização de fonemas para representar as palavras é uma possibilidade de melhora que deve ser explorada.

Foram seguidos rigorosamente os mesmos padrões e algoritmos usados anteriormente.

### 4.6.4. Experimentos

Os experimentos foram realizados com os mesmos conjuntos de dados já utilizados nos testes com algoritmos vetoriais. Todos os conjuntos de dados utilizados são de propriedade de empresas de diferentes setores da economia: financeiro, de transformação, e comércio varejista.

#### 4.6.4.1. Metodologia

Os experimentos que terão seus resultados expressos nesta seção foram obtidos na segunda etapa de experimentos apresentada anteriormente.

#### 4.6.4.2. Resultados

Apresentamos a seguir o resultado dos experimentos conduzidos com os algoritmos não vetoriais.

##### 4.6.4.2.1. K-Modóide

Esse algoritmo também testado é outra variação do *K-Means*, sendo usado um centróide baseado na moda estatística dos caracteres e palavras. A medida de distância é a da distância de edição entre textos.

A Tabela 34 a seguir apresenta os resultados obtidos com a utilização do K-Modóide com os diversos agrupamentos sendo ou não fonetizados.

| Fonético | Agrupa caracteres | Tempo | Acurácia |
|---|---|---|---|
| Não | Vogais | 2303 | 96,70% |
| Não | Nenhum | 1212 | 96,55% |
| Não | Todos os caracteres | 2103 | 96,19% |
| Sim | Nenhum | 821 | 96,12% |
| Não | Vogais e consoantes | 2601 | 96,08% |
| Não | Consoantes | 2920 | 96,05% |
| Sim | Consoantes | 1651 | 95,87% |
| Sim | Vogais e consoantes | 1779 | 95,77% |
| Sim | Vogais | 1240 | 95,74% |
| Sim | Todos os caracteres | 1324 | 95,71% |

**Tabela 34: Resultados obtidos pelo K-Modóide**

A melhor combinação de parâmetros novamente não inclui a fonetização dos textos e, como apresentado anteriormente, quando fonetizado o desempenho não alcança os melhores resultados. O agrupamento de vogais novamente apresenta bons resultados quando usamos a distância de edição e o agrupamento de vogais. Entretanto, a diferença entre as precisões do melhor e do pior caso é de aproximadamente 1,00% e, novamente, temos valores bastante homogêneos em todas as configurações. Para análises futuras usaremos o segundo melhor algoritmo devido à melhor relação acurácia/tempo apresentada.

#### 4.6.4.2.2. K-Modóide híbrido com K-Medóide

Neste momento, as duas abordagens que usam versões do *K-Means* modificadas são combinadas e temos o resultado apresentado na Tabela 35.

| Fonético | Centróides | Centróide Modal | Agrupa caracteres | Tempo | Acurácia |
|---|---|---|---|---|---|
| Não | 3 | Sim | Nenhum | 5121 | 96,92% |
| Não | 3 | Sim | Todos os caracteres | 10968 | 96,84% |
| Não | 3 | Sim | Vogais e consoantes | 8948 | 96,64% |
| Sim | 3 | Sim | Nenhum | 2521 | 96,46% |
| Não | 3 | Sim | Vogais | 10540 | 96,38% |
| Não | 3 | Sim | Consoantes | 9832 | 96,18% |
| Sim | 3 | Sim | Todos os caracteres | 6629 | 96,16% |
| Sim | 3 | Sim | Vogais | 6723 | 96,13% |
| Sim | 3 | Sim | Consoantes | 5239 | 95,96% |
| Sim | 3 | Sim | Vogais e consoantes | 5189 | 95,74% |

**Tabela 35: K-Modóide híbrido com K-Medóide**

A configuração mais simples supera as outras na acurácia. Em relação ao tempo gasto para executar a tarefa a versão fonética pode ser declarada vencedora pela segunda vez dentre todos os experimentos. Levando em consideração a relação acurácia/tempo a vencedora é a configuração mais simples utilizando os textos fonetizados.