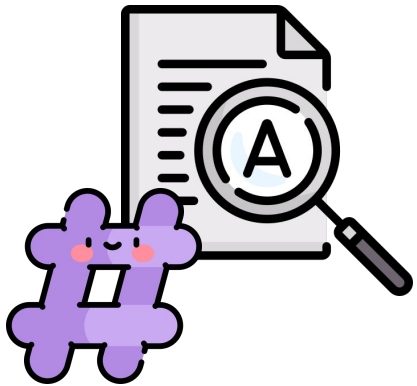
UNIFESP

Programa de Pós Graduação em Ciências da Computação
Análise de Algoritmos e Estruturas de Dados - Profa. Dra. Lilian Berton

Processamento de Texto e Redução de Dimensionalidade com *Feature Hashing*

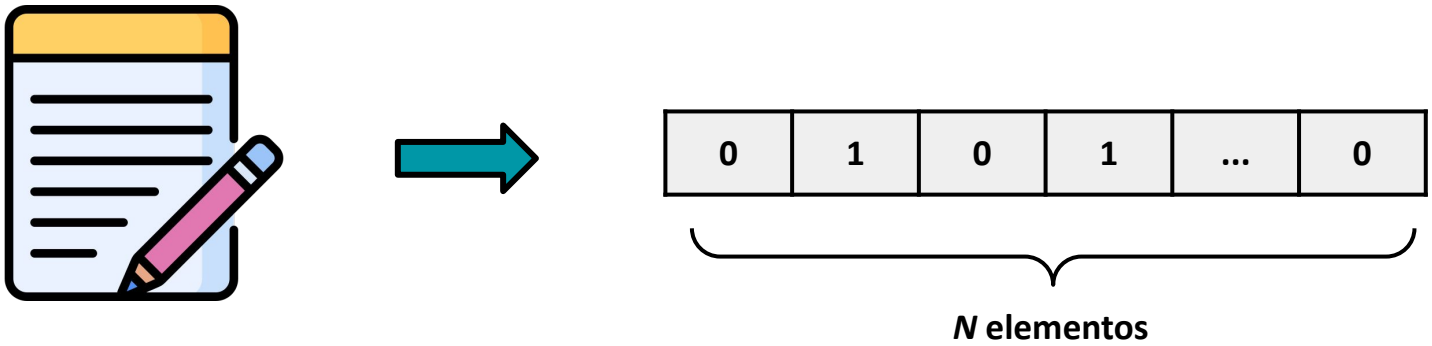


Willian Dihanster Gomes de Oliveira

31 de Maio de 2021
São José dos Campos - SP

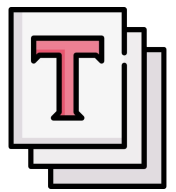
Introdução

- A utilização de **dados textuais em** algoritmos de ***Machine Learning*** emprega a **necessidade** de alguma **representação vetorial**.
  - Exemplo: contagem de palavras; redes neurais; *etc*.
- Em geral, retornam **bastante atributos**, o que pode gerar um **maior consumo de tempo** e de **armazenamento**. Técnicas de **redução de dimensionalidade** podem ser utilizadas.

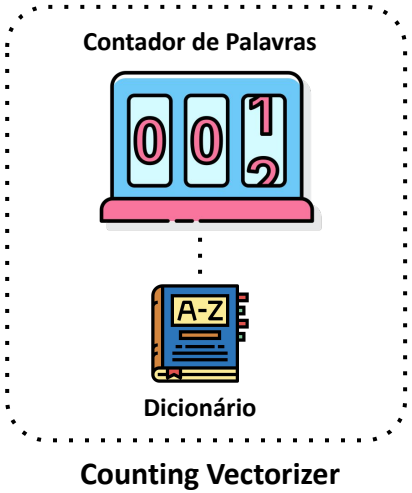


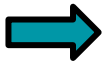
Counting Vectorizer

- **Contagem de palavras** em cada **documento** (pode binarizar: presença de palavra ou não).
- Necessita **salvar** o **vocabulário/dicionário de palavras** (mais memória).
- ***N*** = número de palavras únicas.



Documentos





Documentos Vetorizados

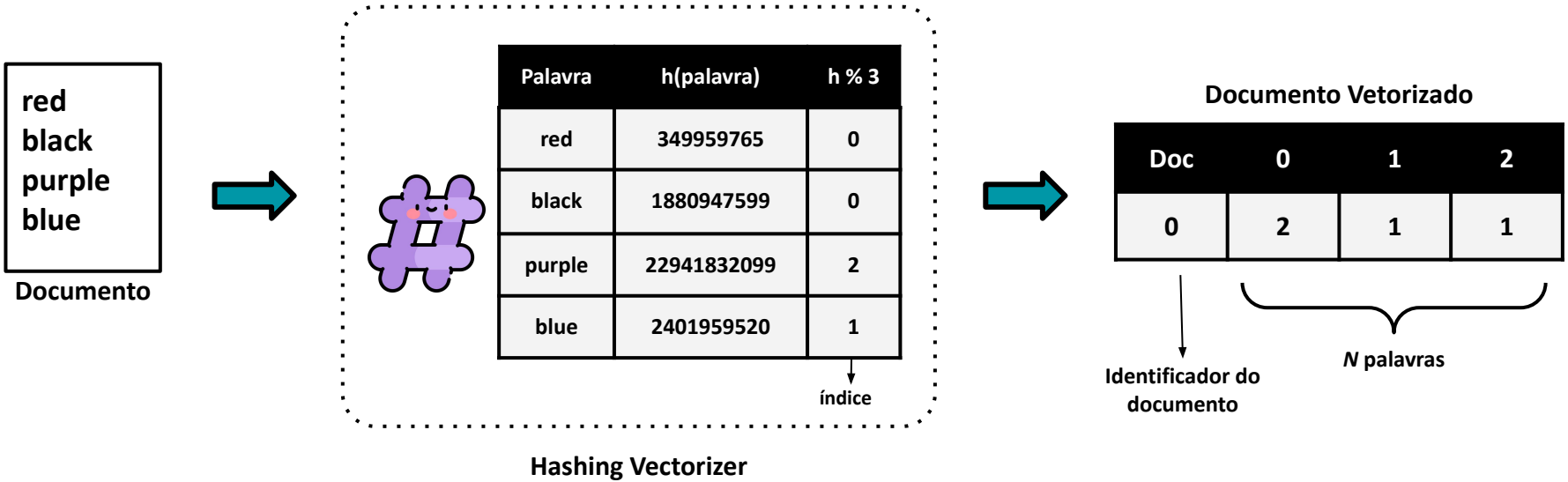
Doc	P1	P2	P3	...	Pn
0	1	3	5	...	1
...	...	...	...	...	...
m	1	4	10	...	1

Identificador dos documentos

N palavras

Hashing Vectorizer (*Feature Hashing*)

- **Mapeamento** de cada **palavra** a uma **posição** (coluna) via **função Hash (MurmurHash)**.
- Mais **rápido** (é **direto**) e **não necessita salvar vocabulário**.
- ***N*** = definido pelo usuário (pode ser utilizado para **reduzir dimensionalidade**).

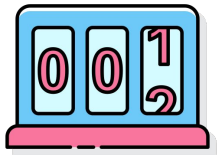


Exemplo

John gosta de assistir filmes.

Mary gosta de filmes também.

John também gosta de futebol.



Counting Vectorizer (todo o vocabulário)

Doc	john	gosta	de	assistir	filmes	mary	também	futebol
1	1	1	1	1	1	0	0	0
2	0	1	1	0	1	1	1	0
3	1	1	1	0	0	0	1	1

Hashing Vectorizer (*N*=5)

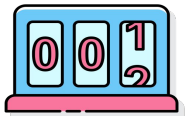
Doc	0	1	2	3	4
1	1	1	1	2	0
2	0	1	2	2	1
3	0	1	0	3	1

Exemplo

John gosta de assistir filmes.

Mary gosta de filmes também.

John também gosta de futebol.



Counting Vectorizer

Dicionário

Palavra	Cont
john	2
gosta	3
de	3
assistir	1
filmes	2
mary	1
também	2
futebol	1

Documentos Vetorizados

Doc	john	gosta	de	assistir	filmes	mary	também	futebol
1	1	1	1	1	1	0	0	0
2	0	1	1	0	1	1	1	0
3	1	1	1	0	0	0	1	1

Hashing Vectorizer

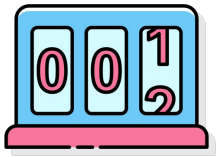
Exemplo de Cálculo da Hash

Palavra	h(palavra)	h%5
john	1663310373	3
gosta	3087871873	3
de	635560811	1
assistir	2736875975	0
filmes	1460040792	2
mary	640610058	3
também	1616290804	4
futebol	466864298	3

Documentos Vetorizados

Doc	0	1	2	3	4
1	1	1	1	2	0
2	0	1	2	2	1
3	0	1	0	3	1

Comparativo



Counting Vectorizer

Hashing Vectorizer

Em geral melhor acurácia

Mantém as palavras

Mais rápido

Usa menos memória

Necessário guardar vocabulário

Mais custoso

Pode gerar colisões

Impossível recuperar palavra “hasheada”

Experimentos

- **CountVectorizer** e **HashingVectorizer**, modelo **MultinomialNB** e ***dataset*** da biblioteca **sklearn**.
  - Dataset *fetch_20newsgroups* com textos sobre “eletronics” e “space”.
    - 1184 exemplos para treino (50% de cada classe) e 787 para teste.

'\n\n\nOk, here could be the first question or answer or something:\n\nQ: I want to copyprotect a **program** I wrote. How should I do it?\nA: You would be wise not to copyprotect that program. You see, those \n people that wants to get a cracked copy of your **program** will go to \n various length to crack your program, and some of those **crackers** \n are good, and know the common tricks.\n So, the copy protection wouldn\'t stop those.\n Ok, then. What about legitimate users? Copy protection can be a hassle\n for legitimate users, and can hinder them in their work, expecially\n if there is some "**key**" item that can get lost.\n So, the copy protection wouldn\'t help much of the legitimate users, but\n would make life somewhat of a misery for them.\n\n\n\n(This is my opinion, and I speak as a legitimate user :-)\nYou are of course free to have your opinion about this subject....\n\n\n'

Exemplo de texto sobre “eletronics”

'\n[Excellent discussion of DC-X landing techniques by Henry deleted]\n\n\nThe DC-X will not take of **horizontally**. It takes of **vertically**. \n\n\nFor several reasons. Vertical landings don\'t require miles of runway and limit\nnoise pollution. They don\'t require **wheels** or **wings**. Just turn on the engines\nand touch down. Of course, as Henry pointed out, vetical landings aren\'t quite\nthat simple.\n\n\nWell, to be blunt, yes. But at least you\'re learning.\n\n\nThe Soyuz **vehicles** use **parachutes** for the descent and then fire small **rockets**\njust before they hit the ground. **Parachutes** are, however, not especially\npractical if you want to reuse something without much effort. The landings\nare also not very comfortable. However, in the words of Georgy Grechko,\n"I prefer to have bruises, not to sink."\n\n'

Exemplo de texto sobre “space”

Resultados 1

- Usando todo o vocabulário no Counting Vectorizer.

Algoritmo	*N*	Acurácia	Tempo Médio (s)
Counting	17.984	0,91	0,36
Hashing	100	0,66	0,24
Hashing	1.000	0,80	0,25
Hashing	10.000	0,89	0,26
Hashing	15.000	0,91	0,26
Hashing	17.984	0,88	0,27

Hashing mais rápido, mesmo com *N* igual. Acurácia menor, mas bem parecida com *N* >= 10.000.

Resultados 2

- Limitando o número de palavras em ambos algoritmos.

Algoritmo	*N*	Acurácia	Tempo Médio (s)
Counting	1.000	0,87	0,31
Hashing	1.000	0,80	0,27
Counting	5.000	0,91	0,31
Hashing	5.000	0,81	0,25
Counting	10.000	0,91	0,33
Hashing	10.000	0,89	0,26

Hashing mais rápido em todos os testes. Counting melhor em acurácia, mas próximos, com *N* grande.

Hashing com pior acurácia, possivelmente pelas colisões.

Counting mais lento, porque mesmo limitando, é necessário calcular todo o vocabulário e pegar as *N* palavras mais frequentes.

Conclusões

- **Hashing Vectorizer** pode ser uma boa opção quando a **prioridade é tempo** e/ou **menos gastos de memória**, e é aceitável **colisões/errar mais** e quando **não é necessário** sabermos as **palavras utilizadas**.

- **Counting Vectorizer** pode ser mais indicado quando se busca **melhores resultados** (em termos de acurácia) e/ou se **saber qual palavra foi utilizada será útil**, além de ser possível aceitar o **tempo maior** e o uso de **memória** a mais.

Referências

Artigos Científicos:
WEINBERGER, Kilian et al. Feature hashing for large scale multitask learning. In: **Proceedings of the 26th annual international conference on machine learning**. 2009. p. 1113-1120.

Internet em geral:
https://kavita-ganesan.com/hashingvectorizer-vs-countvectorizer/#.YLAX7qhKjIV
https://scikit-learn.org/stable/auto_examples/model_selection/grid_search_text_feature_extraction.html#sphx-glr-auto-examples-model-selection-grid-search-text-feature-extraction-py

Obrigado!