Modelo Bag-Of-Words (BOW)

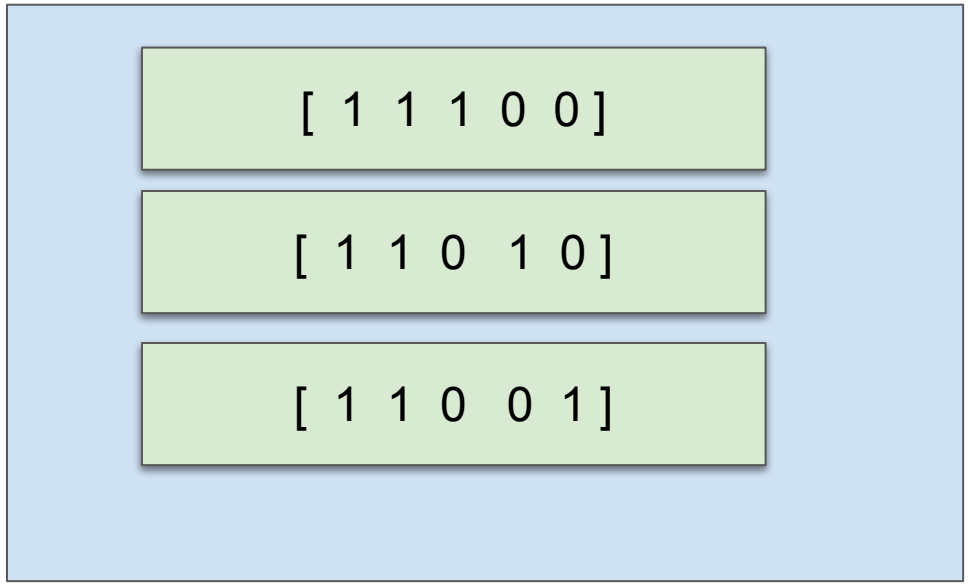
Willian Massami Watanabe

Contexto

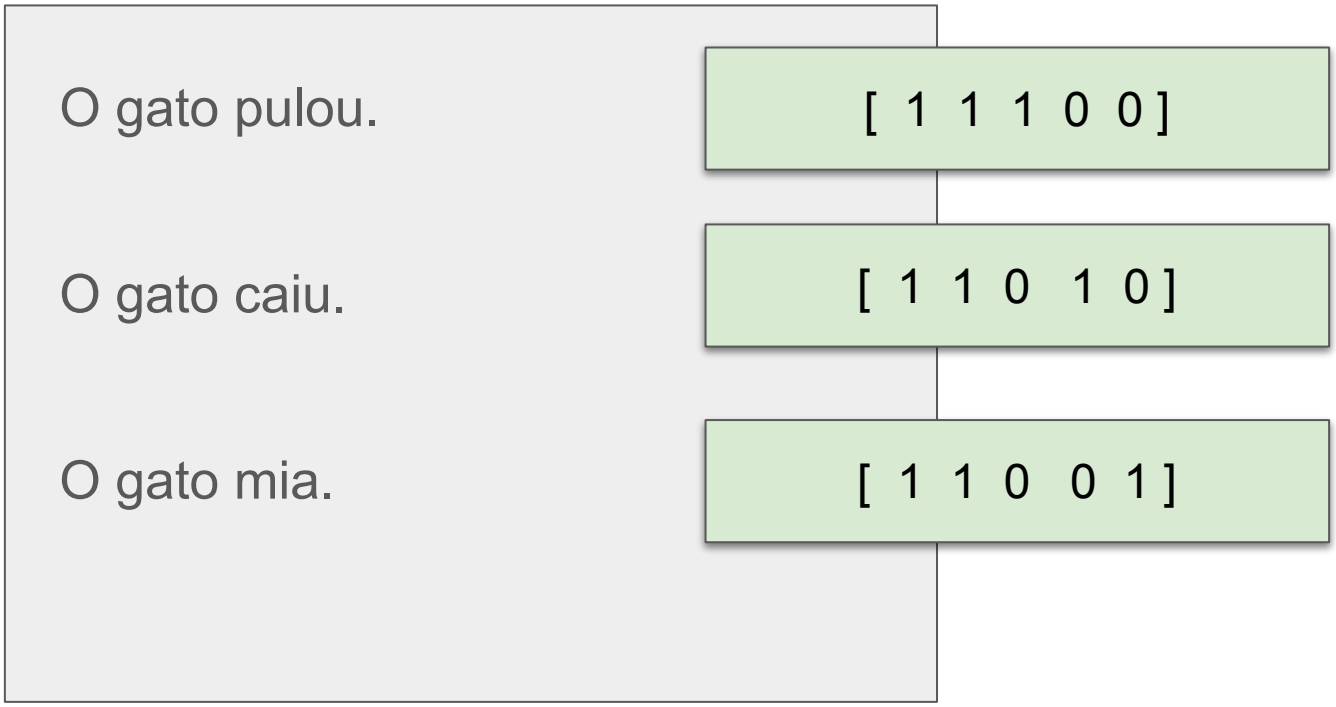
O gato pulou.

O gato caiu.

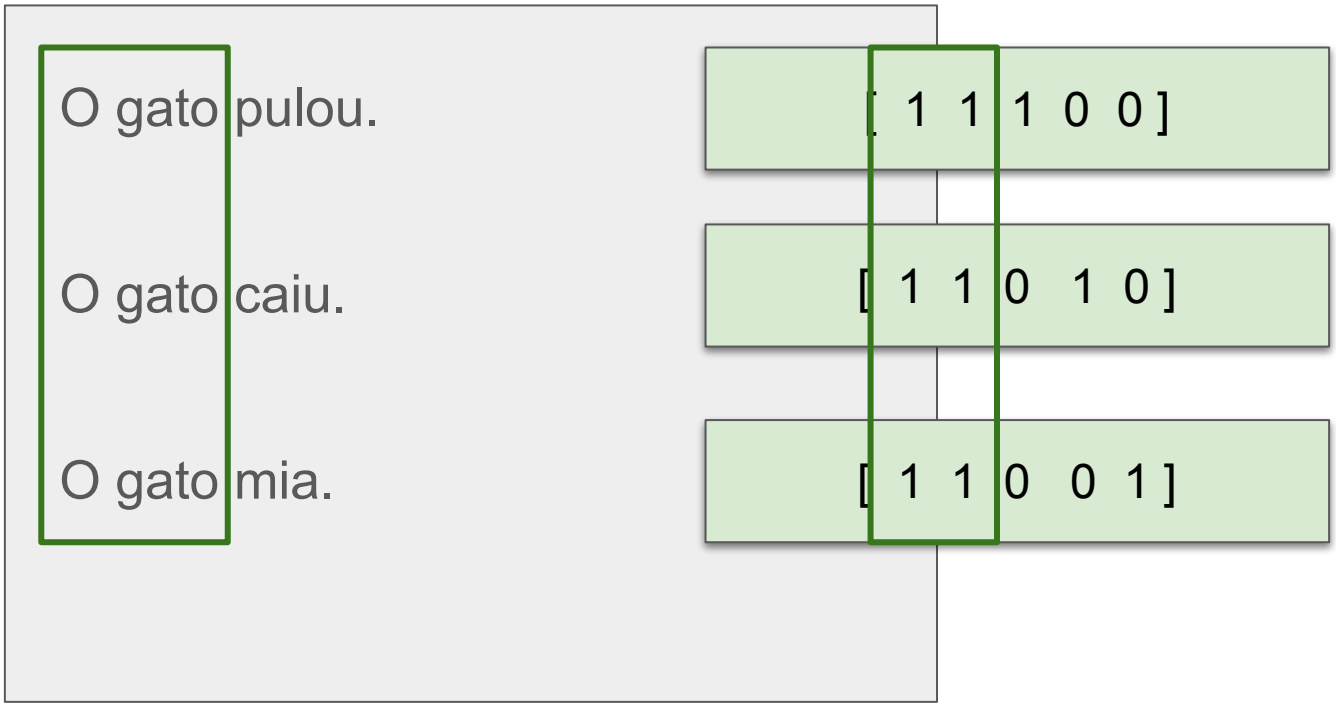
O gato mia.



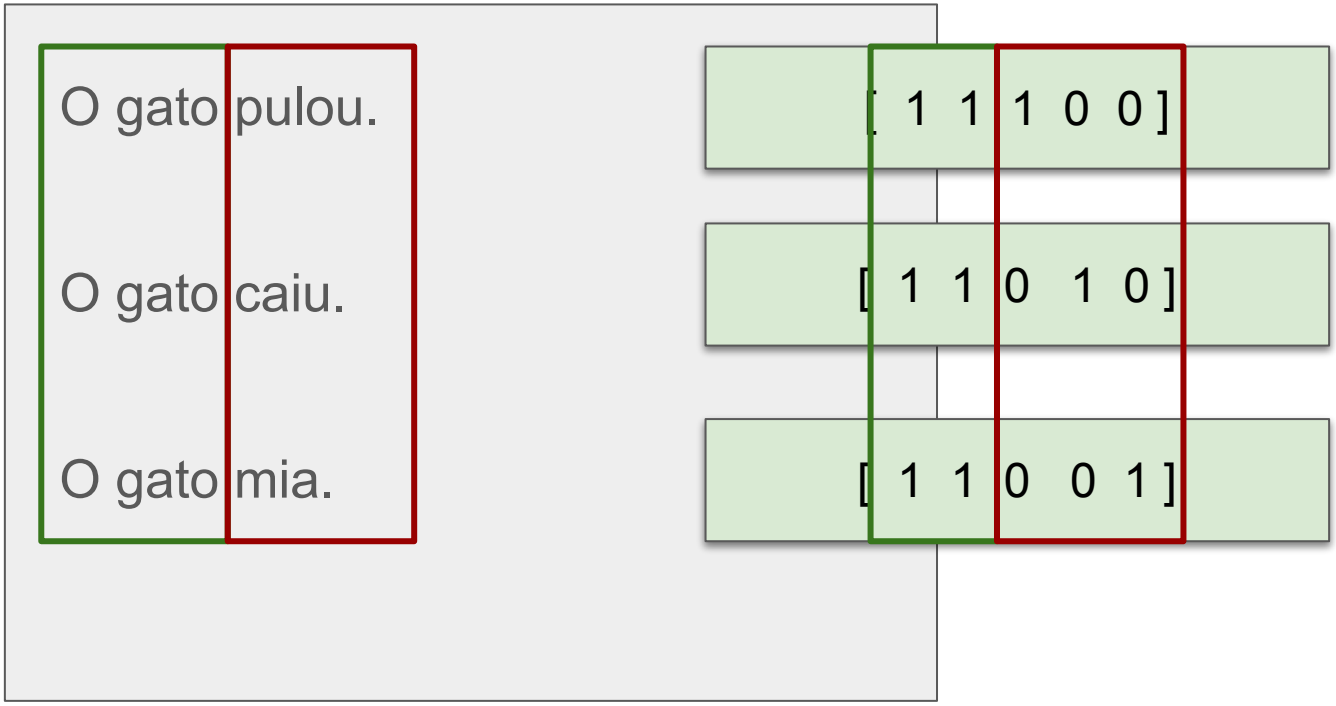
Contexto



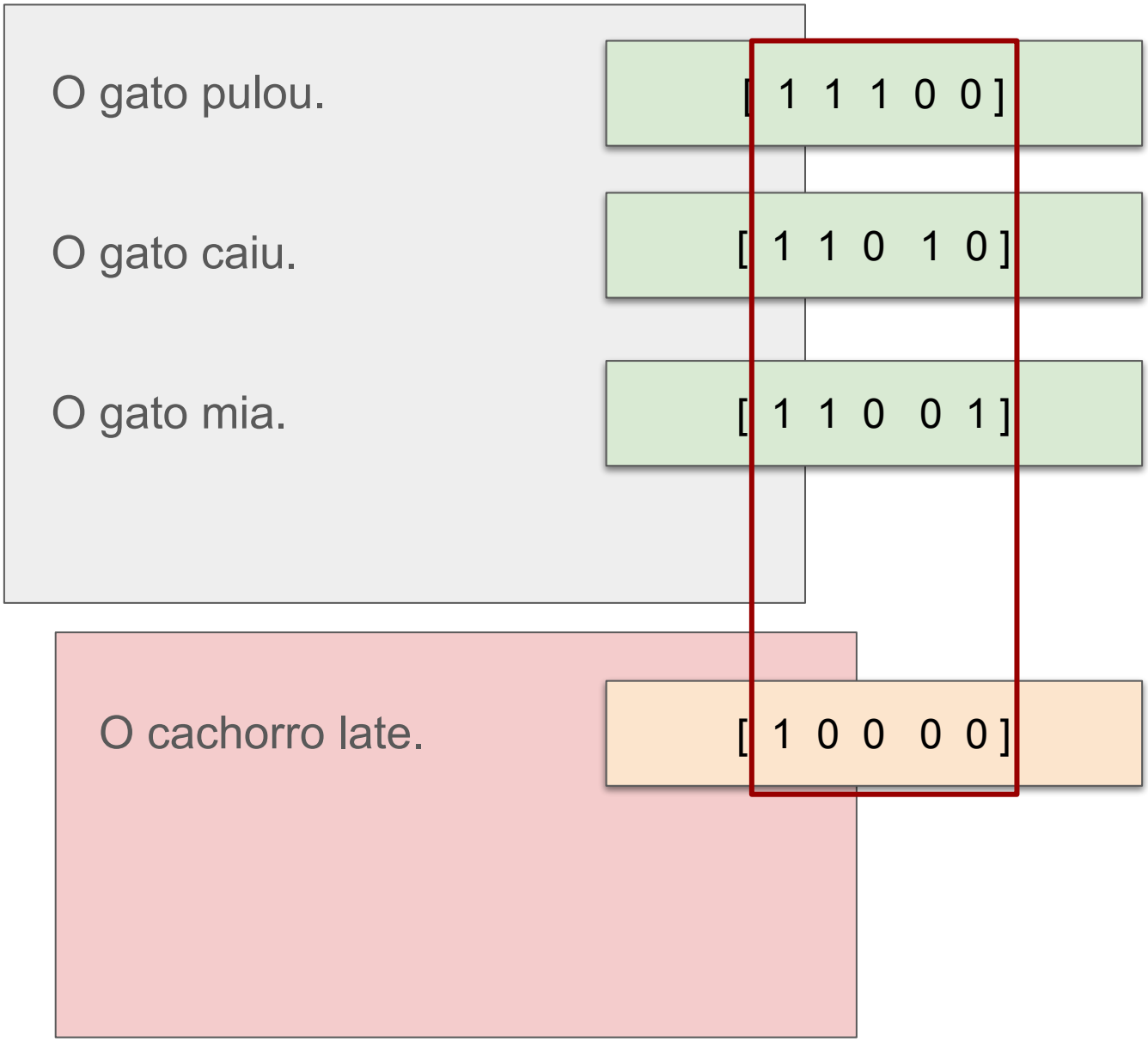
Contexto



Contexto



Contexto


O gato pulou.
[1 1 1 0 0]
O gato caiu.
[1 1 0 1 0]
O gato mia.
[1 1 0 0 1]
O cachorro late.
[1 0 0 0 0]

Definição

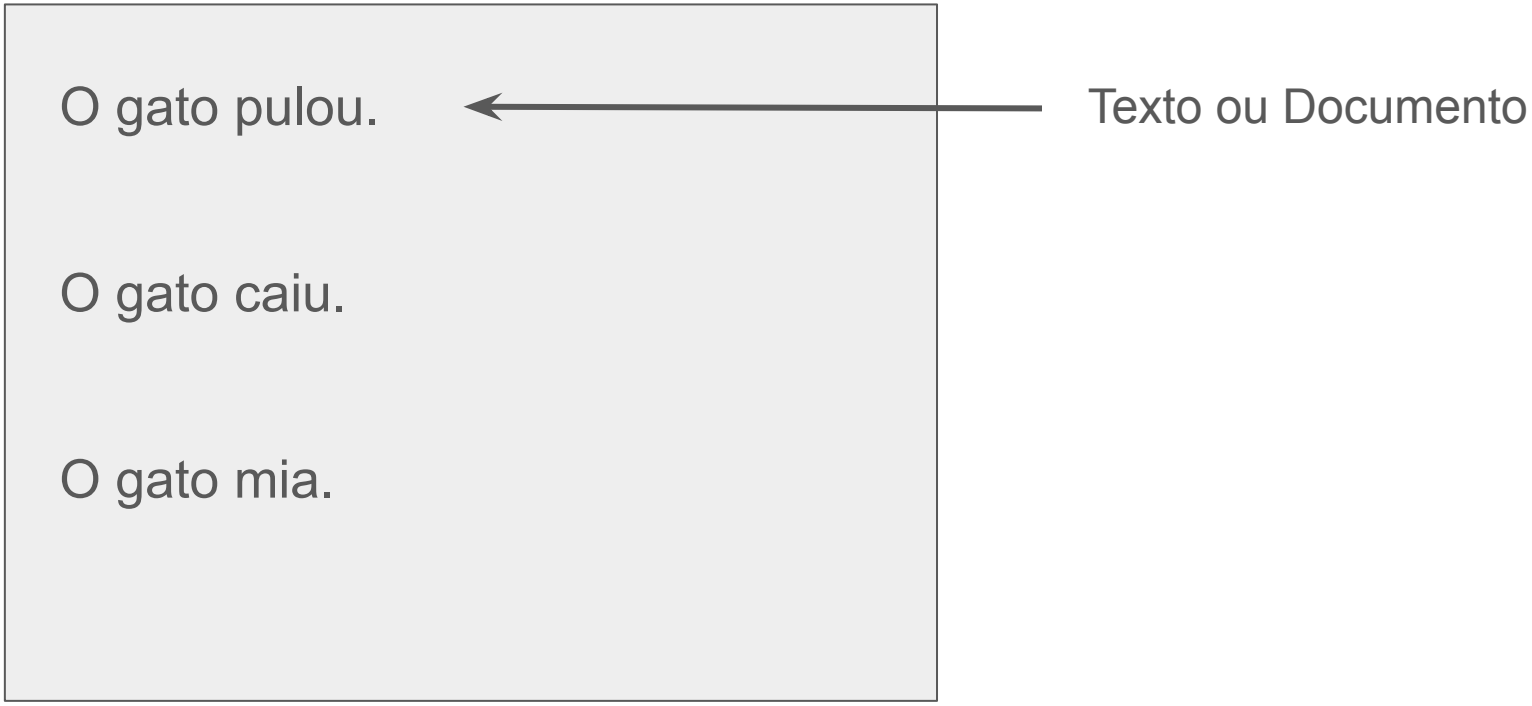
O "Bag of Words" (BOW) é uma representação simplificada utilizada em processamento de linguagem natural (PLN) e recuperação de informações (RI) para representar textos como uma coleção desordenada de palavras. Ele ignora a ordem das palavras e captura a multiplicidade, ou seja, registra a contagem de cada palavra no texto.

Esse modelo é frequentemente usado em métodos de classificação de documentos, onde a frequência de cada palavra é usada como um recurso para treinar um classificador.

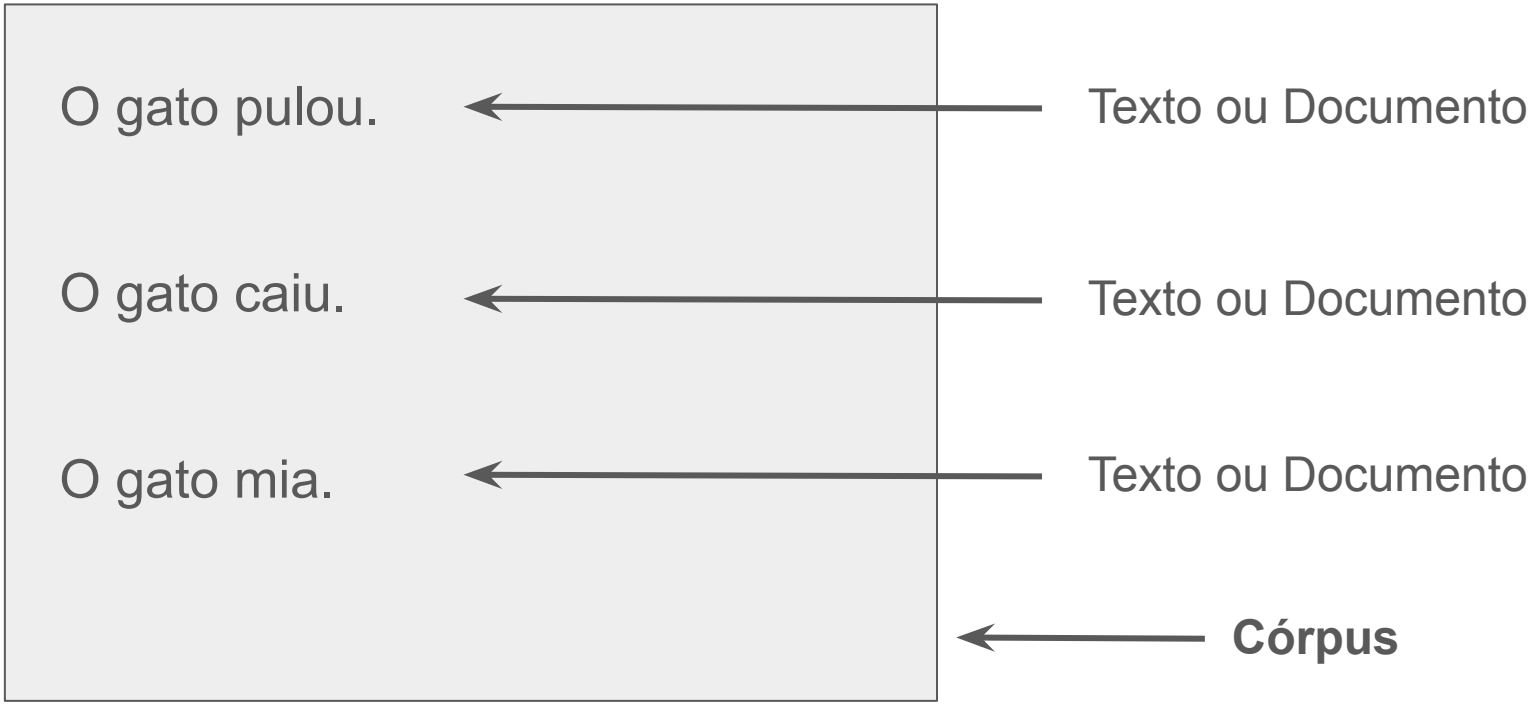
O gato pulou.

O gato caiu.

O gato mia.


O gato pulou.
O gato caiu.
O gato mia.
Texto ou Documento


O gato pulou.
Texto ou Documento
O gato caiu.
Texto ou Documento
O gato mia.
Texto ou Documento

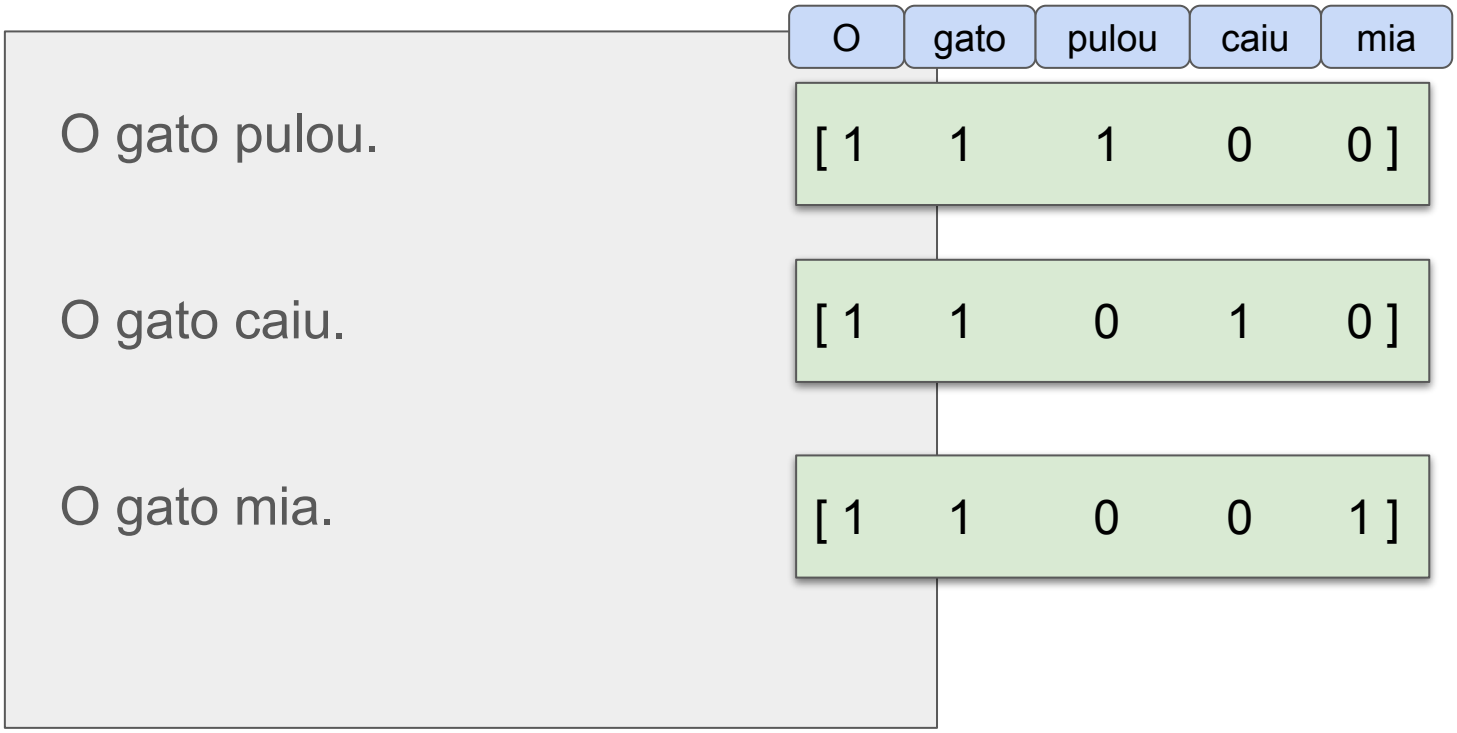

O gato pulou.
Texto ou Documento
O gato caiu.
Texto ou Documento
O gato mia.
Texto ou Documento
Córpus

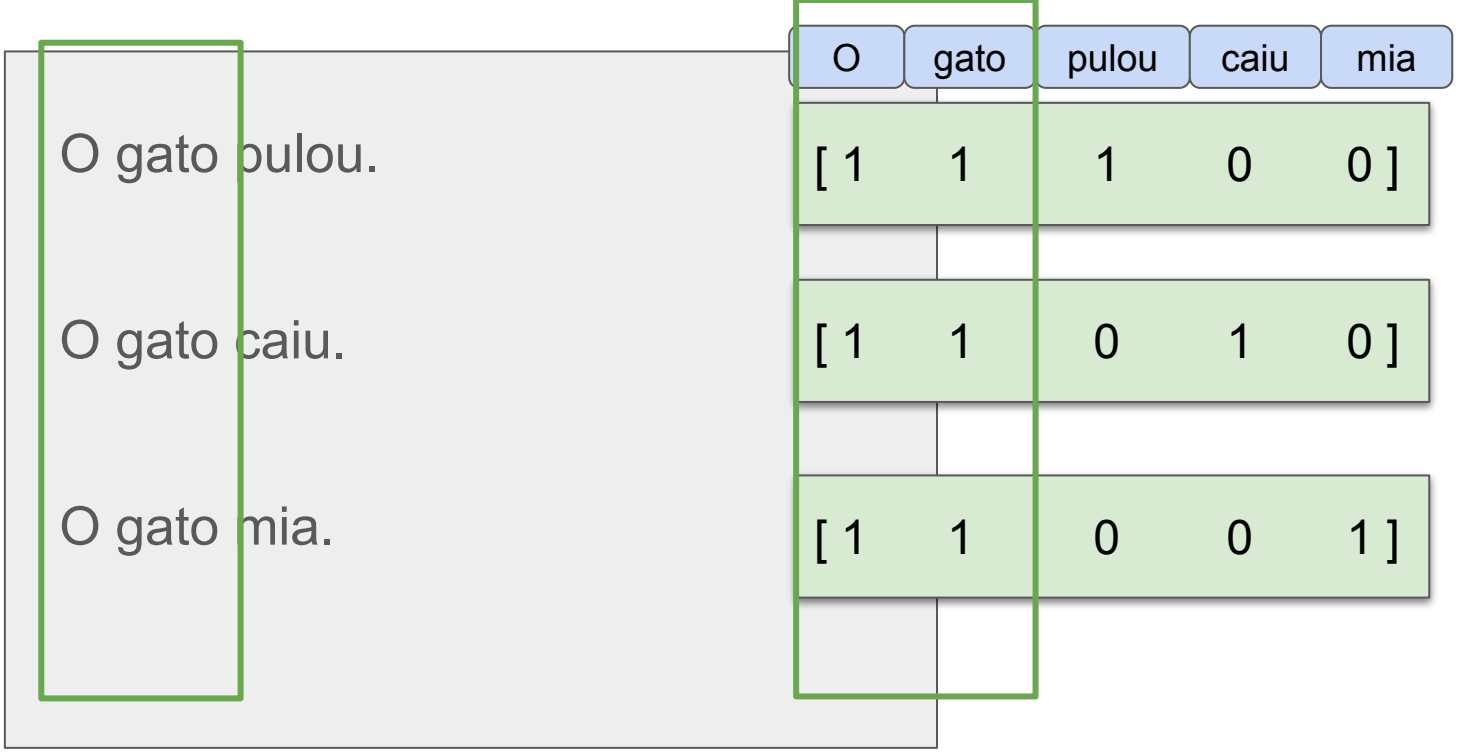
O	gato	pulou	caiu	mia

O gato pulou.

O gato caiu.

O gato mia.


O
gato
pulou
caiu
mia
O gato pulou.
[1 1 1 0 0]
O gato caiu.
[1 1 0 1 0]
O gato mia.
[1 1 0 0 1]

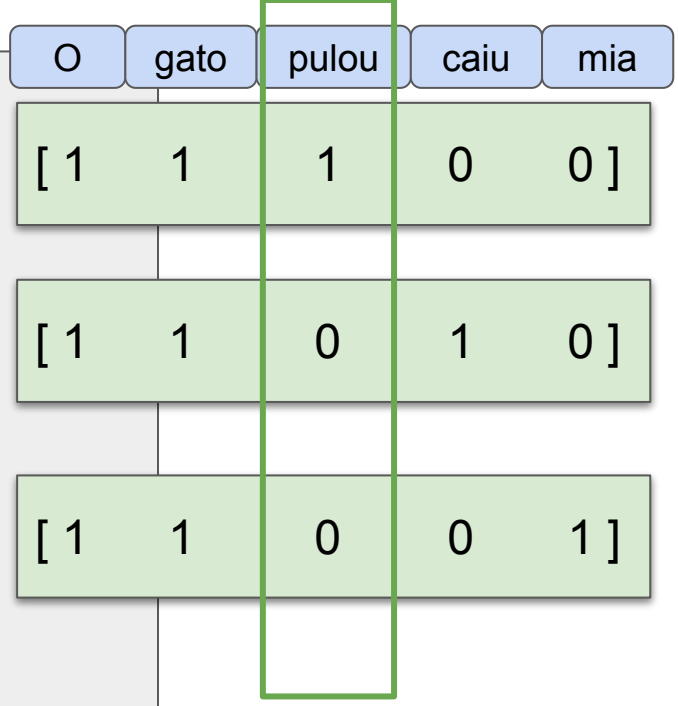


Dicionário de termos

O gato pulou.

O gato caiu.

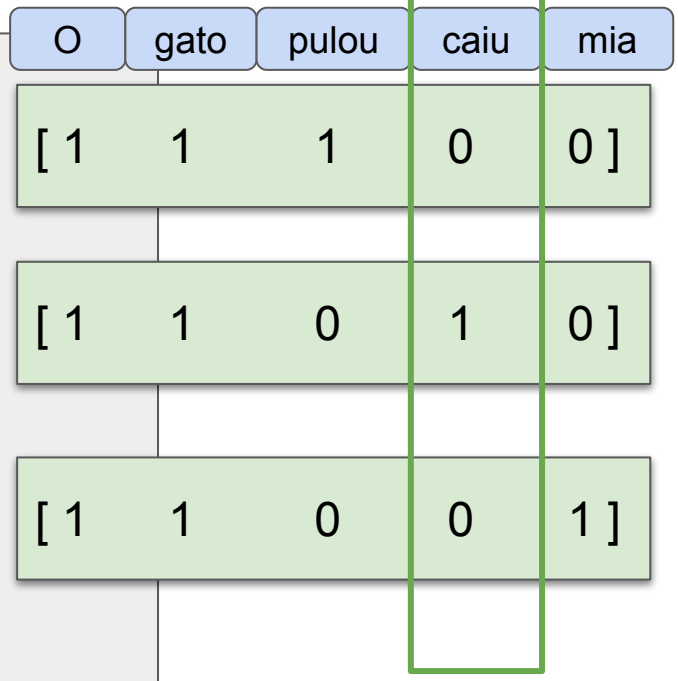
O gato mia.

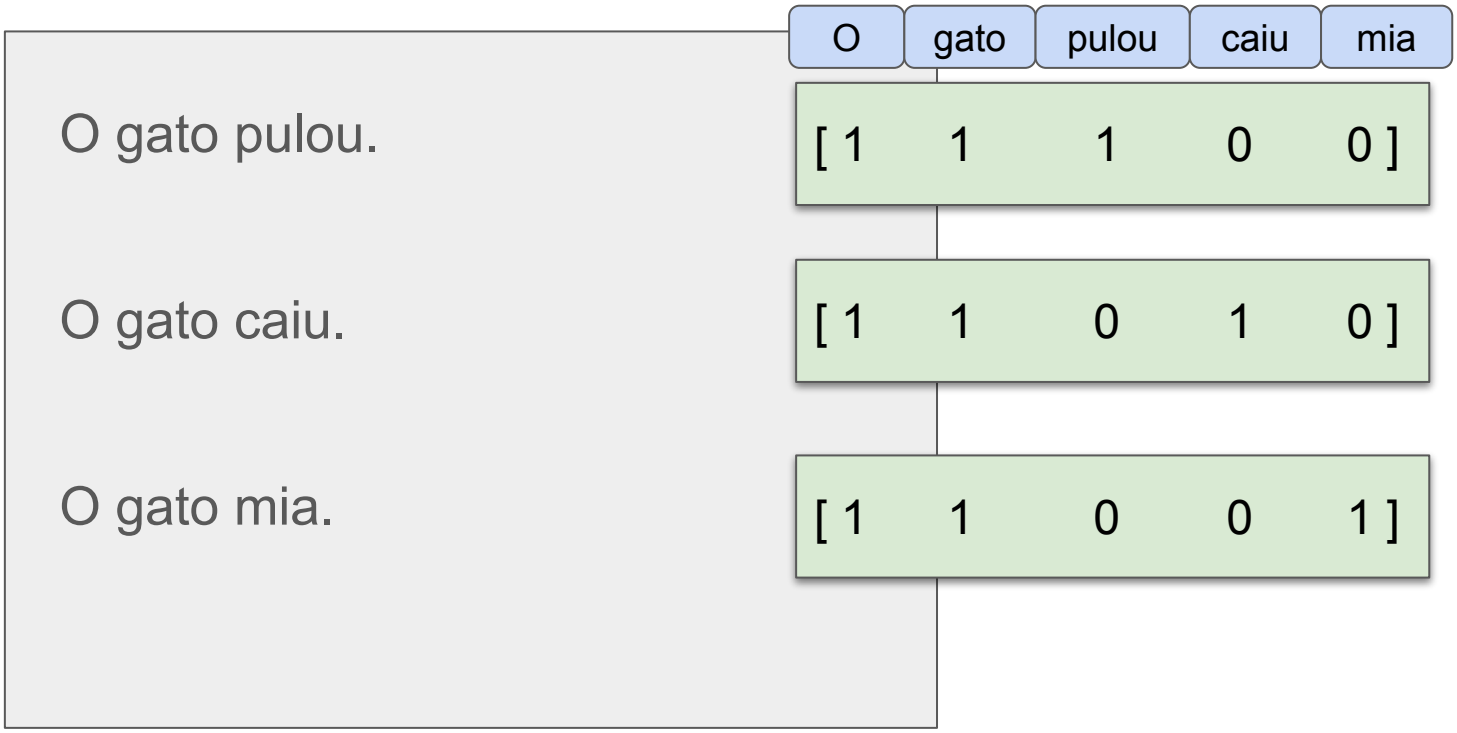


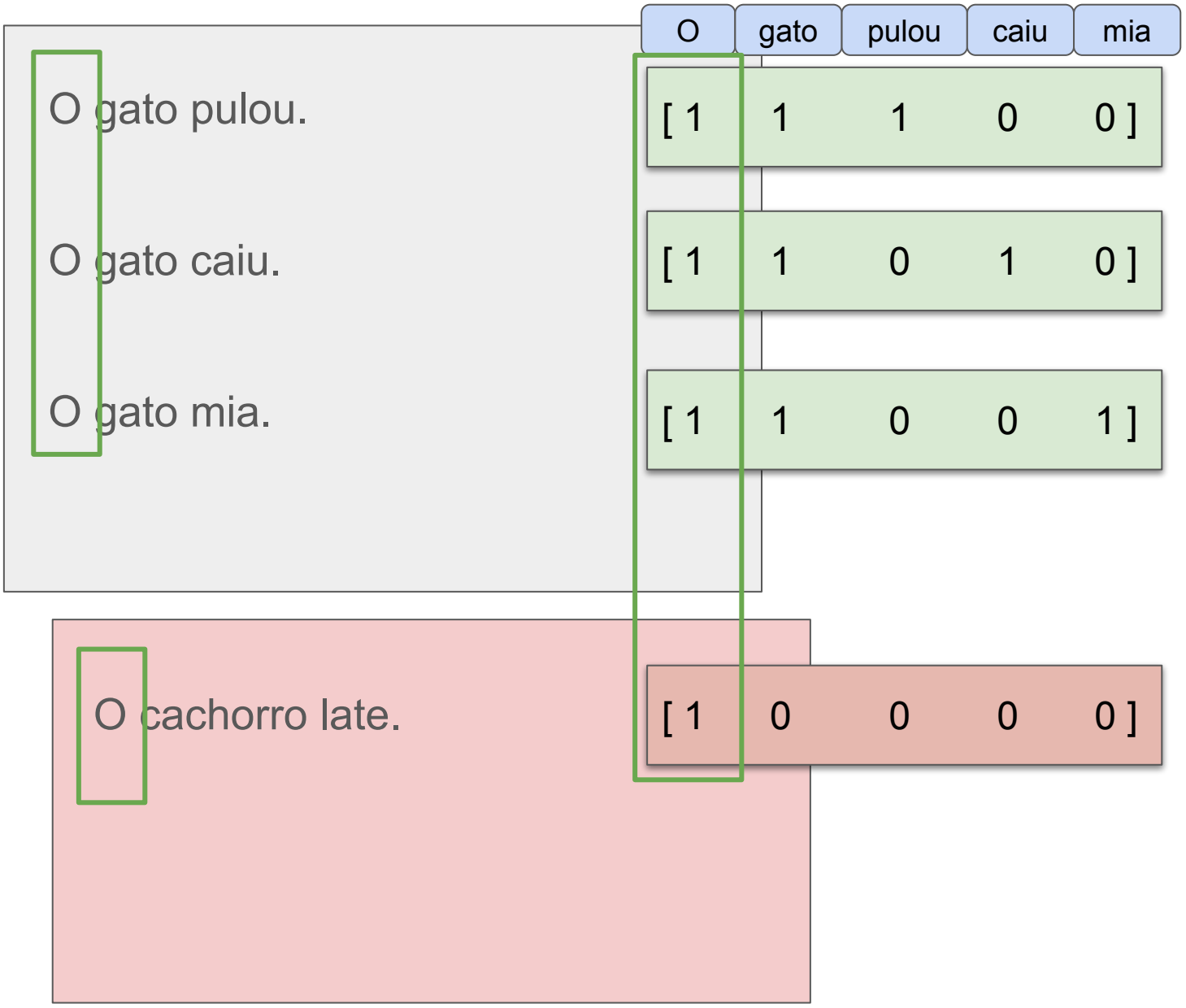
O gato pulou.

O gato caiu.

O gato mia.




O
gato
pulou
caiu
mia
Dicionário de termos
O gato pulou.
[1 1 1 0 0]
O gato caiu.
[1 1 0 1 0]
O gato mia.
[1 1 0 0 1]
Ajuste do modelo (fit)

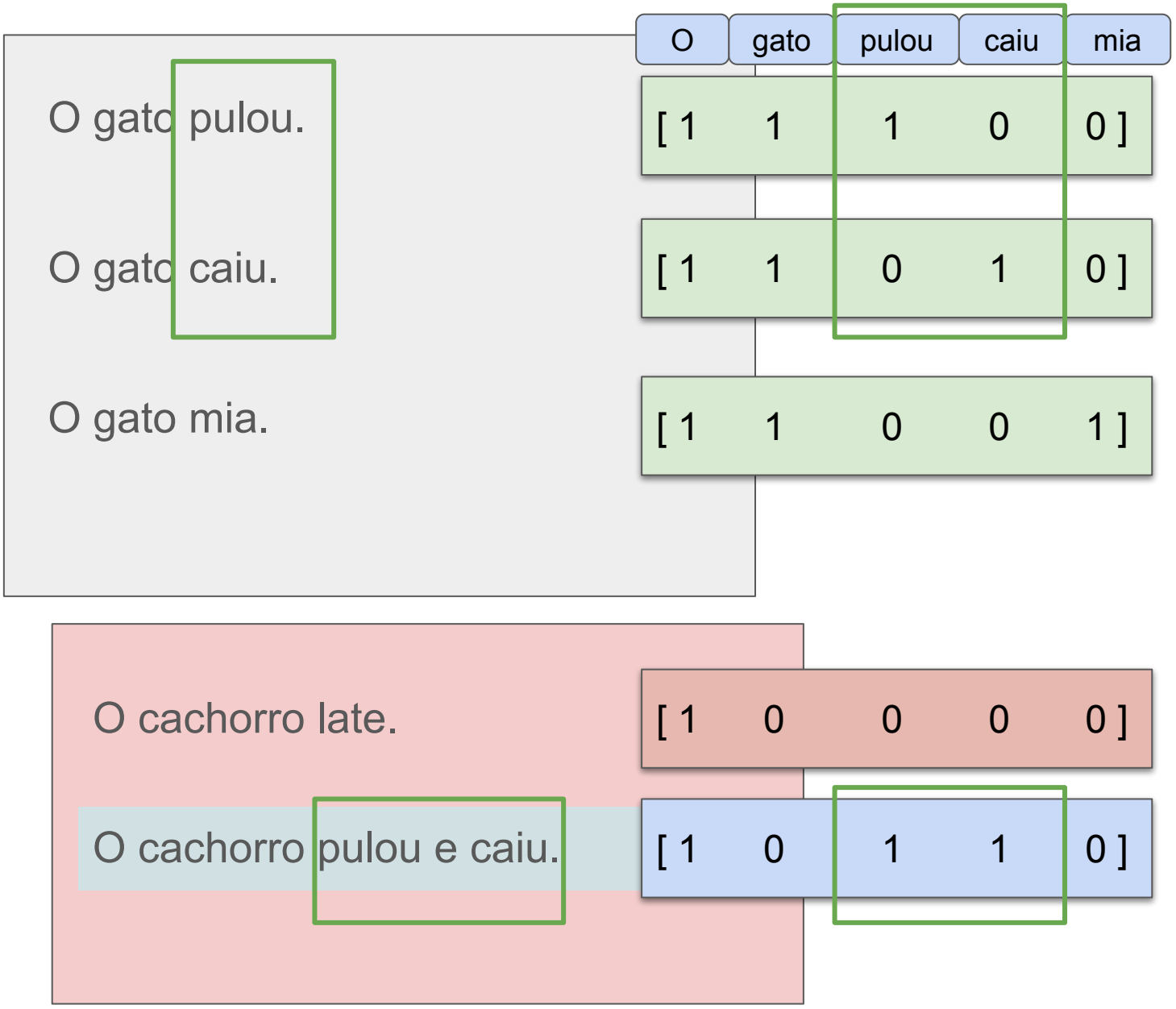


Dicionário de termos

Ajuste do modelo (fit)

Aplicação do modelo (transform)

Aplicação do modelo (transform)


O
gato
pulou
caiu
mia
Dicionário de termos
O gato pulou.
O gato caiu.
O gato mia.
[1 1 1 0 0]
[1 1 0 1 0]
[1 1 0 0 1]
O cachorro late.
O cachorro pulou e caiu.
[1 0 0 0 0]
[1 0 1 1 0]
Aplicação do modelo (transform)

Comentários sobre o exemplo

- No exemplo, os termos aparecem apenas 1 vez. Mas o modelo Bag-Of-Words, como ilustrado, pode ser utilizado com termos que aparecem múltiplas vezes em um mesmo documento;
- A representação dos documentos foi realizada como um vetor numérico e pode ser utilizada para identificar similaridade e distância entre documentos (distância do cosseno ou euclidiana, por exemplo).

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

Aplicação do modelo (transform)

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

Aplicação do modelo (transform)

	ação	de	gear	jogo	top	um
0	1	1	0	0	1	1

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

Ajuste do modelo (fit)

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

Aplicação do modelo (transform)

	ação	de	gear	jogo	top	um
0	1	1	0	0	1	1

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

Aplicação do modelo (transform)

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

```python
from sklearn.feature_extraction.text import CountVectorizer

texts = ['Top Gear é um jogo de ação.']
vectorizer = CountVectorizer()
vectorizer.fit(texts)

print(vectorizer.vocabulary_)
```

```
{'top': 4, 'gear': 2, 'um': 5, 'jogo': 3, 'de': 1, 'ação': 0}
```

```python
import pandas as pd

result = vectorizer.transform(['Top Gun foi um filme interessante de ação.'])

df = pd.DataFrame(result.toarray(), columns=vectorizer.get_feature_names_out())
print(df)
```

```
   ação  de  gear  jogo  top  um
0     1   1     0     0    1   1
```

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

- Palavras com menos de 2 caracteres são descartadas, por terem baixa representatividade na configuração inicial da API Vectorizer;
- Os termos são todos convertidos em letras minúsculas;
- Termos não existentes no dicionário são descartados da análise.

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = [ Mineração de Dados foi interessante.',
          'Perceptron é o começo de Redes Neurais ']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)
```

```
result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais	perceptron	redes
0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	1	0	0	1	0
2	0	0	0	0	1	0	0	0	0
3	0	2	4	0	0	2	0	0	0

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais	perceptron	redes
0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	1	0	0	1	0
2	0	0	0	0	1	0	0	0	0
3	0	2	4	0	0	2	0	0	0

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']
vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais	perceptron	redes
0	0	0	0	0	0	1	0	0	0
1	0	0	0	0	1	0	0	1	0
2	0	0	0	0	1	0	0	0	0
3	0	2	4	0	0	2	0	0	0

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = CountVectorizer(binary=True)

texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais	perceptron	redes
0	0	1	1	0	0	1	0	0	0

sklearn.feature_extraction.text **CountVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = CountVectorizer(binary=True)

texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

começo	dados	de	foi	interessante	mineração	neurais	perceptron	redes
0	0	1	1	0	0	1	0	0	0

TF-IDF - Term Frequency - Inverse Document Frequency

- É uma medida utilizada para avaliar a importância de uma palavra em um documento, em relação ao conjunto de documentos disponíveis.
- TF (Term Frequency) é a porcentagem de ocorrências do termo em um documento, dividida pela porcentagem de ocorrências do termo em todos os documentos.
- IDF (Inverse Document Frequency) é a raiz natural do logaritmo do número total de documentos, dividido pelo número de documentos que contêm o termo.

TF - Term Frequency

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

onde t é um termo é um termo do dicionário de termos e d é um documento

TF - Term Frequency

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

onde t é um termo é um termo do dicionário de termos e d é um documento

A bolinha de queijo caiu de ponta. *documento 1*

TF - Term Frequency

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

onde t é um termo é um termo do dicionário de termos e d é um documento

A bolinha de queijo caiu de ponta. *documento 1*

$$tf(t, d) = 1 / 6 = 0,16$$ *para t = bolinha no documento 1*

TF - Term Frequency

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

onde t é um termo é um termo do dicionário de termos e d é um documento

A bolinha de queijo caiu de bonta. *documento 1*

$tf(t, d) = 1 / 6 = 0,16$ *para t = bolinha no documento 1*

$tf(t, d) = 2 / 6 = 0,33$ *para t = de no documento 1*

TF - Term Frequency

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

onde t é um termo é um termo do dicionário de termos e d é um documento

PLN não voa. *documento 2*

tf(t, d) = 1 / 3 = 0,33 *para t = PLN no documento 2*

tf(t, d) = 1 / 3 = 0,33 *para t = não no documento 2*

tf(t, d) = 1 / 3 = 0,33 *para t = voa no documento 2*

IDF - Inverse Document Frequency

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos e
D é um córpus (conjunto de documentos)

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos e
D é um córpus (conjunto de documentos)

João não gosta de futebol.	*documento 1*
Nadar de óculos é interessante.	*documento 2*
Tênis é um esporte de raquete.	*documento 3*

$$1 + \log \frac{1 + \text{número de documentos em D}}{1 + df(t, D)}$$

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos e D é um córpus (conjunto de documentos)

João não gosta de futebol.	*documento 1*
Nadar de óculos é interessante.	*documento 2*
Tênis é um esporte de raquete.	*documento 3*

$$idf(t, d) = \log(3/(1+1)) = 0{,}405$$ *para t = João*

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos e D é um córpus (conjunto de documentos)

João não gosta de futebol.	*documento 1*
Nadar de óculos é interessante.	*documento 2*
Tênis é um esporte de raquete.	*documento 3*

$idf(t, d) = \log(3/(1+1)) = 0{,}405$ *para t = João*

$idf(t, d) = \log(3/(1+1)) = 0{,}405$ *para t = Nadar*

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos e D é um córpus (conjunto de documentos)

João não gosta de futebol.	*documento 1*
Nadar de óculos é interessante.	*documento 2*
Tênis é um esporte de raquete.	*documento 3*

$idf(t, d) = \log(3/(1+1)) = 0{,}405$ *para t = João*

$idf(t, d) = \log(3/(1+1)) = 0{,}405$ *para t = Nadar*

$idf(t, d) = \log(3/(3+1)) = -0{,}287$ *para t = de*

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

onde t é um termo é um termo do dicionário de termos, D é um córpus (conjunto de documentos) e d é um documento de D

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

Documento

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

Córpus

onde t é um termo é um termo do dicionário de termos, D é um córpus (conjunto de documentos) e d é um documento de D

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

Documento

$$df(t, D) = \text{número de documentos que contém o termo t em D}$$

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

Córpus

onde t é um termo é um termo do dicionário de termos, D é um córpus (conjunto de documentos) e d é um documento de D

$$tfidf(t, D) = tf(t, d) * idf(t, D)$$

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
```

```
result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.408248	0.816497	0.0	0.000000	0.408248	0.0

sklearn.feature_extraction.text TfidfVectorizer

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.408248	0.816497	0.0	0.000000	0.408248	0.0

sklearn.feature_extraction.text TfidfVectorizer

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.408248	0.816497	0.0	0.000000	0.408248	0.0

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.408248	0.816497	0.0	0.000000	0.408248	0.0

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
from sklearn.feature_extraction.text import TfidfVectorizer

vectorizer = TfidfVectorizer(use_idf=False)
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$tf(t, d) = \frac{\text{número de vezes que t aparece em d}}{\text{número de termos em d}}$$

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.408248	0.816497	0.0	0.000000	0.408248	0.0

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text TfidfVectorizer

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$idf(t, D) = \log \frac{\text{número de documentos em D}}{1 + df(t, D)}$$

```
[1.40546511 1.40546511 1.         1.40546511 1.40546511 1.40546511
 1.40546511 1.40546511 1.40546511]
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$idf(t, D) = 1 + \log \frac{1 + \text{número de documentos em D}}{1 + df(t, D)}$$

```
[1.40546511 1.40546511 1.         1.40546511 1.40546511 1.40546511
 1.40546511 1.40546511 1.40546511]
```

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)

result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

$$idf(t, D) = 1 + \log \frac{1 + \text{número de documentos em D}}{1 + df(t, D)}$$

```
1.40546511 1.40546511 1.         1.40546511 1.40546511 1.40546511
 1.40546511 1.40546511 1.40546511]
```

sklearn.feature_extraction.text TfidfVectorizer

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)
```

```
result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.498446	0.709297	0.0	0.000000	0.498446	0.0

sklearn.feature_extraction.text **TfidfVectorizer**

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)
```

$$tfidf(t, D) = tf(t, d) * idf(t, D)$$

```
result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.498446	0.709297	0.0	0.000000	0.498446	0.0

sklearn.feature_extraction.text TfidfVectorizer

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.feature_extraction.text

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.']

vectorizer.fit(texts)
print(vectorizer.vocabulary_)
print(vectorizer.idf_)
```

$$tfidf(t, D) = tf(t, d) * idf(t, D)$$

```
result = vectorizer.transform([
    'Aprendizado Profundo é parecido com Mineração?',
    'Perceptron é interessante mas não é legal.',
    'Outra disciplina interessante seria Cloud.',
    'Mineração de Dados usa conceitos de Mineração de Dados
])
```

	começo	dados	de	foi	interessante	mineração	neurais
0	0.0	0.000000	0.000000	0.0	0.000000	1.000000	0.0
1	0.0	0.000000	0.000000	0.0	0.707107	0.000000	0.0
2	0.0	0.000000	0.000000	0.0	1.000000	0.000000	0.0
3	0.0	0.498446	0.709297	0.0	0.000000	0.498446	0.0

Aplicações

- Recuperação de Informação Textual
- Sistemas de Recomendação
- Classificação de Documentos

Aplicações

- Recuperação de Informação Textual
- Sistemas de Recomendação
- Classificação de Documentos

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante.',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
print(result_df[['perceptron', 'redes', 'neurais']]
      .sort_values(by=['redes', 'neurais', 'perceptron'], ascending=False))
```

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
print(result_df[['perceptron', 'redes', 'neurais']]
      .sort_values(by=['redes', 'neurais', 'perceptron'], ascending=False))
```

	perceptron	redes	neurais
1	0.402446	0.490779	0.490779
3	0.402446	0.000000	0.000000
0	0.000000	0.000000	0.000000
2	0.000000	0.000000	0.000000
4	0.000000	0.000000	0.000000
5	0.000000	0.000000	0.000000

	aprendizado	cloud	com	começo	conceitos	dados	de
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.464964	0.392555
1	0.000000	0.000000	0.000000	0.490779	0.000000	0.000000	0.339772
2	0.402446	0.000000	0.490779	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
4	0.000000	0.472493	0.000000	0.000000	0.000000	0.000000	0.000000
5	0.205337	0.000000	0.000000	0.000000	0.250407	0.410674	0.693439

```
print(result_df[['perceptron', 'redes', 'neurais']]
      .sort_values(by=['redes', 'neurais', 'perceptron'], ascending=False))
```

	perceptron	redes	neurais
1	0.402446	0.490779	0.490779
3	0.402446	0.000000	0.000000
0	0.000000	0.000000	0.000000
2	0.000000	0.000000	0.000000
4	0.000000	0.000000	0.000000
5	0.000000	0.000000	0.000000

```
vectorizer = TfidfVectorizer()
texts = ['Mineração de Dados foi interessante ',
         'Perceptron é o começo de Redes Neurais.',
         'Aprendizado Profundo é parecido com Mineração?',
         'Perceptron é interessante mas não é legal.',
         'Outra disciplina interessante seria Cloud.',
         'Mineração de Dados usa conceitos de Mineração de

result = vectorizer.fit_transform(texts)
```

	perceptron	redes	neurais
1	0.402446	0.490779	0.490779
3	0.402446	0.000000	0.000000
0	0.000000	0.000000	0.000000
2	0.000000	0.000000	0.000000
4	0.000000	0.000000	0.000000
5	0.000000	0.000000	0.000000