
Modelo Vetorial

Tópicos Especiais em Recuperação de Informações

Prof^a. Solange Pertile

18/09/15

Fontes:

Prof. Viviane Moreira (UFRGS)

Prof. Jairo de Souza (UFJF)

Relembrando Modelo Booleano...

- O Modelo Booleano leva em consideração apenas se o termo da consulta esta presente no documento
- Mas...
 - Um documento que menciona o termo da consulta mais vezes deve estar mais relacionado a ela
 - Com isto, e possível atribuir escores aos documentos
 - Usando os escores, monta-se um ranking

Relembrando Modelo Booleano...

- Até agora lidamos com consultas Booleanas
- Documentos casam ou não casam.
- Boa para usuários especialistas com um entendimento preciso das suas necessidades e da coleção
- Boa para aplicações: aplicações podem facilmente processar milhares de resultados.
- Ruim para a maioria dos usuários
- Incapazes de escrever consultas Booleanas (ou são, mas acham muito trabalhoso)
- Não querem procurar em milhares de resultados
 - Principalmente quando se trata de busca na Web

Relembrando Modelo Booleano...

- Problemas consultas booleanas
- Consultas Booleanas resultam ou em poucos (=0) ou em muitos (milhares) resultados
- Precisa-se de muita habilidade para produzir consultas que gerem um número razoável de resultados.
 - AND muito poucos; OR demais

Modelo Vetorial

- ✓ Vector space model (VSM)
- ✓ Associa peso aos termos de indexação.
- ✓ Atribui escores aos documentos.
- ✓ Possibilita *ranking* dos resultados da consulta.

Modelo Vetorial

- ✓ Em modelos de RI baseados em ranking, o sistema retorna uma ordenação dos documentos na coleção em relação a uma consulta.
- ✓ Consultas em texto livre: Em vez de uma linguagem de consulta com operadores e expressões, a consulta é apenas uma ou duas palavras em linguagem natural.

Modelo Vetorial

- ✓ Atribuição de peso
 - ✓ Queremos retornar, em ordem de relevância, os documentos mais prováveis de satisfazer uma consulta.
 - ✓ Como podemos ordenar (ranquear) os documentos em uma coleção de acordo com uma consulta?
 - ✓ Atribuindo um peso – digamos em $[0, 1]$ – para cada documento.
 - ✓ Esse peso mensura quão bem o documento casa com a consulta.

Modelo Vetorial

- ✓ Atribuição de peso
 - ✓ Precisamos de uma forma de atribuir um peso para um par consulta/documento.
 - ✓ Vamos começar com consultas de um-termo.
 - ✓ Se o termo de consulta não ocorre no documento: peso deve ser 0.
 - ✓ Quanto mais frequente o termo de consulta no documento, maior o peso.

Peso dos Termos

- ***Term Frequency (tf)***

- Cada termo em um documento recebe um peso que depende do número de ocorrências do termo no documento

$$tf_{t,d} = \frac{freq_{t,d}}{\max_l} \text{ número de ocorrências do termo mais frequente em } d$$

Alguns termos tem mais importância do que outros

Peso dos Termos

- ***TF-Scaling***
- Mas usar tf puro não é uma boa ideia:
 - Um doc. com 10 ocorrências de um termo é mais relevante que um documento com 1 ocorrência do termo.
 - Mas não 10 vezes mais relevante.
- Relevância não aumenta proporcionalmente com tf.

$$wf_{t,d} = \begin{cases} 1 + \log_{10} tf_{t,d} & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$$

0 → 0
1 → 1
2 → 1.3
10 → 2
1000 → 4

Peso dos Termos

- **Frequência de Documentos**

- Termos raros são mais informativos que termos frequentes
- Lembre das stop words
- Considere um termo na consulta que seja raro na coleção (e.g., **agorafobia**).
- Um documento contendo esse termo é muito provável de ser relevante a consulta **agorafobia**.
- É razoável atribuir maior peso para termos raros como **agorafobia**.

Peso dos Termos

- **Frequência de Documentos**

- A frequência dos termos não é um indicador certo de relevância.
- Queremos um esquema que atribua maior peso aos termos raros em detrimento dos termos frequentes.

Peso dos Termos

- **Peso IDF**

- Seja df_t a freq. de documento para t : o número de documentos que contém t
- Df_t é uma medida inversa da informatividade de t
- $df_t \leq N$
- O idf (*inverse document frequency*) de t é definido por

$$idf_t = \log_2 \frac{N}{df_t}$$

Peso dos Termos

- **Peso IDF**

- Seja df_t a freq. de documento para t : o número de documentos que contém t
- Df_t é uma medida inversa da informatividade de t
- $df_t \leq N$
- O idf (*inverse document frequency*) de t é definido por

$$idf_t = \log_2 \frac{N}{df_t}$$

Peso dos Termos

Exemplo: 1 milhão de documentos
 $N = 1$ milhão

Termo	df	idf
Cachorro	1	
Gato	100	
Rato	1.000	

Qual seria o IDF de cada termo na coleção??

Existe um IDF para cada termo t em uma coleção.

$$idf_t = \log_2 \frac{N}{df_t}$$

idf é uma medida de quão informativo é um determinado termo

Peso dos Termos

Exemplo: 1 milhão de documentos
 $N = 1$ milhão

Termo	df	idf
Cachorro	1	6
Gato	100	4
Rato	1.000	3

Existe um IDF para cada termo t em uma coleção.

$$idf_t = \log_2 \frac{N}{df_t}$$

Peso dos Termos

Exemplo: 1 milhão de documentos
 $N = 1$ milhão

Termo	df	idf
calpurnia	1	
animal	100	
sunday	1,000	
fly	10,000	
under	100,000	
the	1,000,000	

$$idf_t = \log_2 \frac{N}{df_t}$$

Peso dos Termos

Exemplo: 1 milhão de documentos
 $N = 1$ milhão

Termo	df	idf
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

$$idf_t = \log_2 \frac{N}{df_t}$$

Peso dos Termos

- **Freq. Coleção vs. Freq. Documento**

- A frequência de coleção de *t* é o número de ocorrências de *t* na coleção
- Exemplo:

Palavra	Frequencia da Coleção	Frequência de Documento
<i>seguro</i>	10440	3997
<i>tentativa</i>	10422	8760

- Qual palavra é um melhor termo de busca (e deveria obter um maior peso)?

Peso dos Termos

- **Peso TF-IDF**

- O peso tf-idf de um termo é o produto do seu peso tf e seu peso idf.
- O melhor esquema de pesos conhecido da RI
- Nomes alternativos: tf.idf, tf x idf
- Aumenta com o número de ocorrências dentro de um documento.
- Aumenta com a raridade do termo na coleção.

$$w_{t,d} = tf_{t,d} \times \log_2(N / df_t)$$

$$w_{t,d} = tf \times idf$$

$freq_{t,d}$ = número de ocorrências do termo t no doc d

N = número de documentos na coleção

n_t = número de ocorrências do termo t na coleção

Exercícios

- Considere a tabela (a) de frequências para os 3 documentos denotados por Doc1, Doc2, Doc3 abaixo. Calcule os pesos tf-idf para os termos “car”, “auto”, “insurance” e “best”, para cada documento, usando os valores de idf na tabela (b) abaixo.

	Doc1	Doc2	Doc3
car	27	4	24
auto	3	33	0
insurance	0	33	29
best	14	0	17

Tabela (a)

term	df_t	idf_t
car	18,165	1.65
auto	6723	2.08
insurance	19,241	1.62
best	25,235	1.5

Tabela (b)

Matriz de incidência binária

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Matriz de Frequência termo-documento

- Considere o número de ocorrências de um termo em um documento:
 - Cada documento é um vetor de nr. de ocorrências de termos

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Documentos como vetores


- ✓ Tem-se um espaço n-dimensional
 - Os termos são os eixos
 - Documentos são vetores neste espaço
 - Os vetores são muito esparsos
 - Mesmo com θ o número de dimensões pode ser bem grande ($>50\ 000$)
 - Termos que não ocorrem no documento tem peso zero
 - O vetor de um documento j é definido por

$$d_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$$


peso

Consultas como vetores

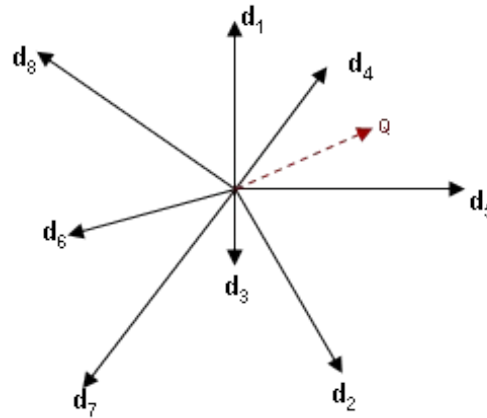
- ✓ **Ideia 1**: assim como os documentos, as consultas também são vetores no espaço
- ✓ **Ideia 2**: Ranquear os documentos pela proximidade com a consulta
- ✓ **proximidade** = similaridade de vetores
- ✓ **proximidade** \approx inverso da distância
- ✓ Termos que não ocorrem na consulta têm peso zero
- ✓ O vetor da consulta q é definido por

$$q_j = (w_{1,q}, w_{2,q}, \dots, w_{t,q})$$


peso

Ideia básica

- ✓ Documentos que estão próximos no espaço vetorial tem conteúdo similar



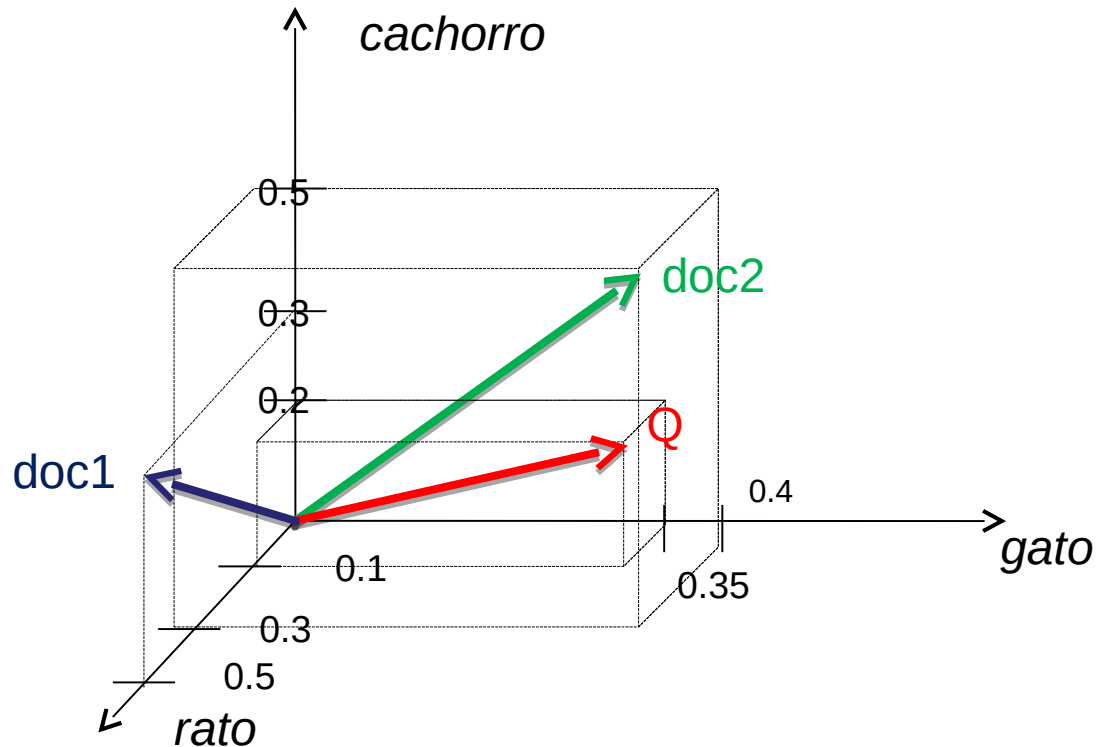
- ✓ Como computar a distância/proximidade entre docs?

Modelo Vetorial

✓ Portanto, um documento d_j e uma consulta de usuário q são representados como vetores com t dimensões:

	Cachorro	gato	rato
Q_i	0.2	0.35	0.1

	Doc1	Doc2
Cachorro	0.3	0.5
Gato	0.0	0.4
Rato	0.5	0.3



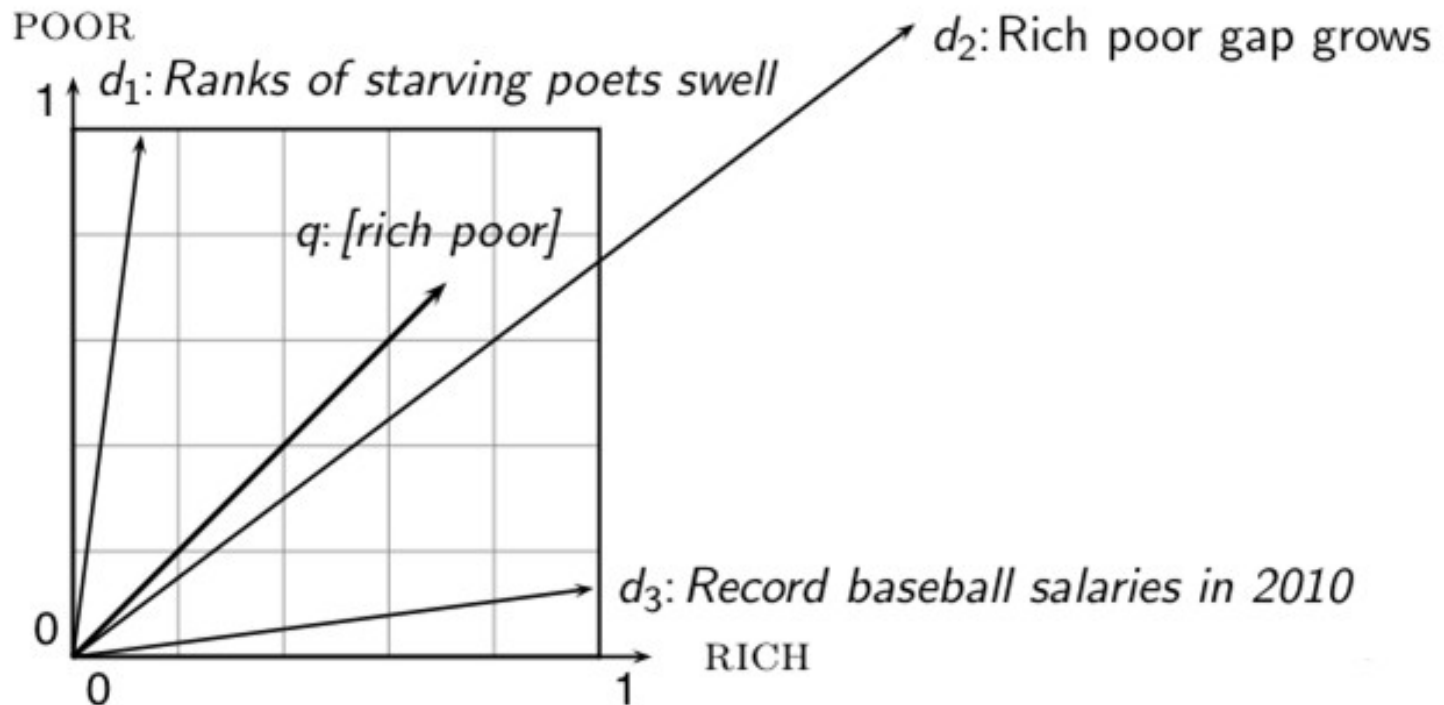
Grau de Similaridade

- Primeiro corte: distância entre dois pontos (distância entre os pontos extremos dos dois vetores)
- Distância Euclidiana?
- Utilizar a Distância Euclideana é uma má ideia . . .
- . . . Porque resulta em valores muito grandes para vetores de diferentes comprimentos.

Grau de Similaridade

✓ Porque distância é uma má ideia

- ✓ A Distância Euclidiana entre a consulta q e o documento d_2 é muito grande apesar de ambos terem uma distribuição similar de termos



Grau de Similaridade

- ✓ O grau de similaridade do documento d_j em relação à consulta q é dado à partir da correlação entre os vetores d_j e q ;
- ✓ Um meio de quantificar essa correlação é através do cálculo do cosseno entre os vetores d_j e q :

Por que cosseno?

Grau de Similaridade

✓ Por que cosseno?

- Classificar os documentos de acordo com o seu ângulo em relação à consulta
- Experimento: escolha um documento d e duplique seu conteúdo. Chame esse documento de d' .
- Apesar de d' ter o dobro do tamanho de d , eles representam “semanticamente” o mesmo conteúdo.
- O ângulo entre os dois documentos é 0, correspondendo a similaridade máxima . . .
- . . . mas a distância Euclidiana entre os dois pode ser muito grande.

Grau de Similaridade

- Grau de similaridade entre um determinado documento e uma consulta, no modelo vetorial, é dado por:

Produto interno dos vetores do documento e da consulta

$$sim(d_j, q) = \frac{\vec{d_j} \bullet \vec{q}}{|\vec{d_j}| \times |\vec{q}|} = \frac{\sum_{i=1}^t w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^t w_{i,j}^2} \times \sqrt{\sum_{i=1}^t w_{i,q}^2}}$$

Comprimento do vetor do documento j

Comprimento do vetor da consulta q

q_i é o peso (e.g. tf-idf) do termo i na consulta

d_i é o peso (e.g. tf-idf) do termo i no documento

Grau de Similaridade

✓ Cosseno de vetores normalizados

- Um vetor pode ser **normalizado** (fazer seu comprimento=1) se dividirmos cada um de seus componentes pelo **comprimento do vetor**.

$$\cos(\vec{q}, \vec{d}) = \vec{q} \cdot \vec{d} = \sum_i q_i \cdot d_i$$

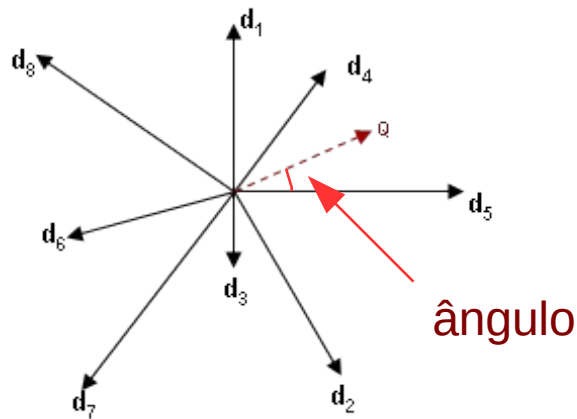
- Para vetores normalizados, o cosseno iguala-se ao produto interno.

Grau de Similaridade

- **O grau de similaridade ($\text{sim}(d_j, q)$) varia entre 0 e 1;**
 - Ao invés de adotar um critério binário, os documentos são ordenados com base no grau de similaridade;
 - Assim, um documento pode ser recuperado, mesmo que ele satisfaça a consulta apenas parcialmente.
- **Quanto mais próximo de 1, mais bem ranqueado será o documento d_j com relação a consulta q ;**
 - Valores próximos de 1 para $\cos(\theta)$ representam maior “proporcionalidade” entre os vetores d_j e q .

Grau de Similaridade

- As consultas são tratadas como pseudo-documentos.
- A similaridade entre um documento e uma consulta é calculada pelo cosseno do ângulo entre eles.
- O comprimento dos vetores não é levado em consideração, apenas suas direções.

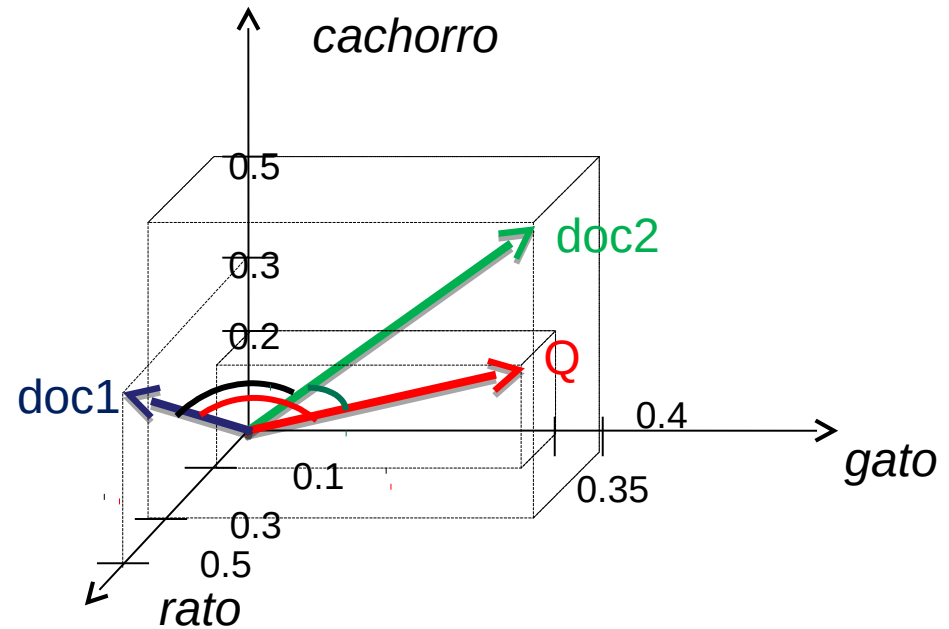
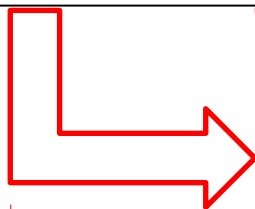


Grau de Similaridade

	Cachorro	gato	rato
Q_i	0.2	0.35	0.1

Termo	Doc1	Doc2
Cachorro	0.3	0.5
Gato	0.0	0.4
Rato	0.5	0.3

$$sim(d_j, q) = \frac{\sum_{i=1}^t w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^t w_{i,j}^2} \times \sqrt{\sum_{i=1}^t w_{i,q}^2}}$$



$sim(DOC1, Q) = \underline{0.45}$

$sim(DOC2, Q) = \underline{0.92}$

$sim(DOC1, DOC2) = \underline{0.73}$

Grau de Similaridade

	Doc1	Doc2	Doc3	
<i>affection</i>	115	58	20	-
<i>jealous</i>	10	7	11	
<i>gossip</i>	2	0	6	

← Frequência dos termos

	Doc1	Doc2	Doc3	
<i>affection</i>	0.996	0.993	0.847	-
<i>jealous</i>	0.087	0.12	0.466	
<i>gossip</i>	0.017	0.00	0.254	

← Normalização

Consulta q = jealous gossip → → Vetor normalizado da consulta q = (0, 0.707, 0.707)

$$\text{Sim}(\text{doc1}, q) = 0 \cdot 0.996 + 0.707 \cdot 0.087 + 0.707 \cdot 0.017 = 0.074$$

$$\text{Sim}(\text{doc2}, q) = 0 \cdot 0.993 + 0.707 \cdot 0.120 + 0.707 \cdot 0 = 0.084$$

$$\text{Sim}(\text{doc3}, q) = 0 \cdot 0.847 + 0.707 \cdot 0.466 + 0.707 \cdot 0.254 = 0.509$$

Grau de Similaridade

	Doc1	Doc2	Doc3	
<i>affection</i>	115	58	20	-
<i>jealous</i>	10	7	11	
<i>gossip</i>	2	0	6	

Frequência dos termos

	Doc1	Doc2	Doc3	
<i>affection</i>	0.996	0.993	0.847	-
<i>jealous</i>	0.087	0.12	0.466	
<i>gossip</i>	0.017	0.00	0.254	

Normalização

Consulta q = jealous gossip → → Vetor normalizado da consulta q = (0, 0.707, 0.707)

$$\text{Sim}(\text{doc1}, q) = 0 \cdot 0.996 + 0.707 \cdot 0.087 + 0.707 \cdot 0.017 = 0.074$$

$$\text{Sim}(\text{doc2}, q) = 0 \cdot 0.993 + 0.707 \cdot 0.120 + 0.707 \cdot 0 = 0.084$$

$$\text{Sim}(\text{doc3}, q) = 0 \cdot 0.847 + 0.707 \cdot 0.466 + 0.707 \cdot 0.254 = 0.509$$

Exercício

Quais documentos são mais similares entre si?

	Doc1	Doc2	Doc3
<i>affection</i>	0.996	0.993	0.847
<i>jealous</i>	0.087	0.120	0.466
<i>gossip</i>	0.017	0.000	0.254

$$\text{sim}(d_j, d_k) = \vec{d}_j \bullet \vec{d}_k = \sum_{i=1}^t w_{i,j} \times w_{i,k}$$

Processamento das consultas

- As consultas são sequências de palavras-chave sem conectores booleanos
- Calcula-se o cosseno entre a consulta e cada um dos documentos
- Os documentos são ordenados de forma decrescente, de acordo com seu cosseno com a consulta
- Os top-k documentos são recuperados

Processamento das consultas

- E preciso computar o cosseno entre a consulta e todos os documentos?

Processamento das consultas

- E preciso computar o cosseno entre a consulta e todos os documentos?
 - Não. Os documentos que não possuem nenhuma das palavras da consulta podem ser descartados.

Processamento das consultas

N= 6 documentos

Docs Termos	Tom e Jerry	Super Mouse	Garfield	Scooby Doo	PiuPiu e Frajola	Mônica
Cachorro	2	1	3	1	0	2
Casa	1	0	2	0	1	3
Gato	2	0	4	0	1	0
Menino	0	0	0	2	0	2
Passarinho	2	0	3	0	5	0
Rato	1	4	0	0	0	0

$$idf_t = \log_2 \frac{N}{df_t}$$

Termo	df	idf
Cachorro	5	0,2630
Casa	4	0,5850
Gato	3	1
Menino	2	1,5850
Passarinho	3	1
Rato	2	1,5850

Consulta



Termo	Q
Cachorro	1
Casa	0
Gato	1
Menino	0
Passarinho	0
Rato	1

Processamento das consultas

■ TF-IDF

$$w_t = tf * idf$$

Docs Termos	Tom e Jerry	Super Mouse	Garfield	Scooby Doo	PiuPiu e Frajola	Mônica
Cachorro	2	1	3	1	0	2
Casa	1	0	2	0	1	3
Gato	2	0	4	0	1	0
Menino	0	0	0	2	0	2
Passarinho	2	0	3	0	5	0
Rato	1	4	0	0	0	0

Docs Termos	Tom e Jerry	Super Mouse	Garfield	Scooby Doo	PiuPiu e Frajola	Mônica
Cachorro	0,5261	0,2630	0,7891	0,2630	0	0,5261
Casa	0,5850	0	1,1699	0	0,5850	1,7549
Gato	2	0	4	0	1	0
Menino	0	0	0	3,1699	0	3,1699
Passarinho	2	0	3	0	5	0
Rato	1,5850	6,3399	0	0	0	0

Termo	Q
Cachorro	0,2630
Casa	0
Gato	1
Menino	0
Passarinho	0
Rato	1,5850

Processamento das consultas

$$\text{sim}(Q, D_1) = \frac{0,5261 \times 0,2630 + 2 \times 1 + 1,5850 \times 1,5850}{\sqrt{0,5261^2 + 0,5850^2 + 2^2 + 2^2 + 1,5850^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,7366$$

$$\text{sim}(Q, D_2) = \frac{0,2630 \times 0,2630 + 6,3399 \times 1,5850}{\sqrt{0,2630^2 + 6,3399^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,8426$$

$$\text{sim}(Q, D_3) = \frac{0,7891 \times 0,2630 + 4 \times 1}{\sqrt{0,7891^2 + 1,1699^2 + 4^2 + 3^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,4279$$

Docs	$\text{sim}(D, Q_i)$
Tom e Jerry	0.7366
Super Mouse	0.8426
Garfield	0.4279
Scooby Doo	0.0115
PiuPiu e Frajola	0.1030
Mônica	0.0200

Ranking



Processamento das consultas

$$\text{sim}(Q, D_1) = \frac{0,5261 \times 0,2630 + 2 \times 1 + 1,5850 \times 1,5850}{\sqrt{0,5261^2 + 0,5850^2 + 2^2 + 2^2 + 1,5850^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,7366$$

$$\text{sim}(Q, D_2) = \frac{0,2630 \times 0,2630 + 6,3399 \times 1,5850}{\sqrt{0,2630^2 + 6,3399^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,8426$$

$$\text{sim}(Q, D_3) = \frac{0,7891 \times 0,2630 + 4 \times 1}{\sqrt{0,7891^2 + 1,1699^2 + 4^2 + 3^2} \times \sqrt{0,2630^2 + 1^2 + 1,5850^2}} = 0,4279$$

Docs	$\text{sim}(D, Q_i)$
Tom e Jerry	0.7366
Super Mouse	0.8426
Garfield	0.4279
Scooby Doo	0.0115
PiuPiu e Frajola	0.1030
Mônica	0.0200

Ranking



Rank	Doc
1	Super Mouse
2	Tom e Jerry
3	Garfield
4	PiuPiu e Frajola
5	Mônica
6	Scooby Doo

Vantagens em relação ao booleano

- Permite o ranking dos resultados
- Permite a comparação entre documentos
- Um índice do modelo vetorial pode ser usado para resolver consultas booleanas, mas o contrário não é verdade.

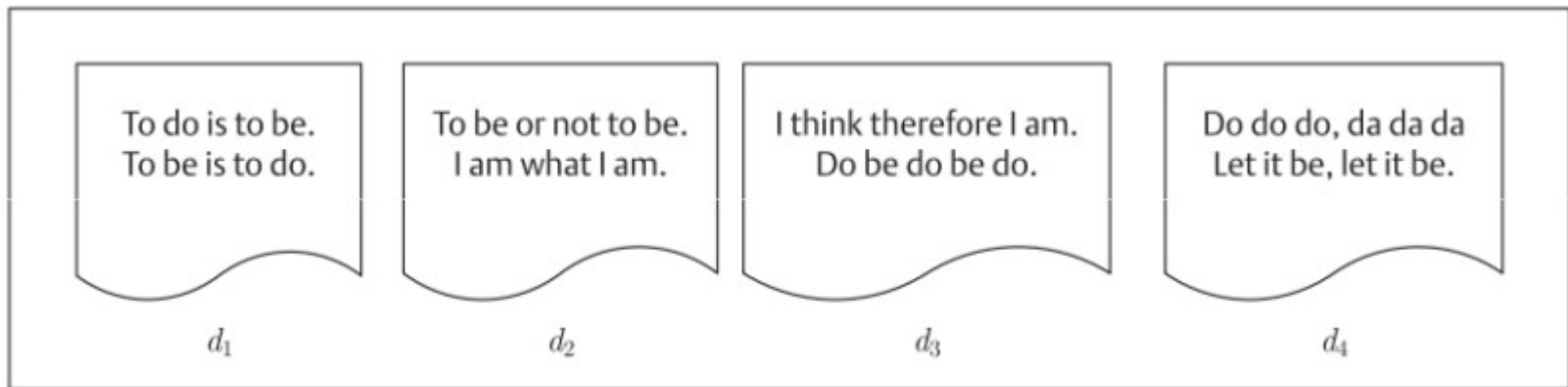
Exercício

1) Considere a tabela da frequência de termos para três documentos (doc1, doc2, doc3). Compute os valores de tf-idf (usando tf-scaling (log)).

<i>termo</i>	<i>doc1</i>	<i>doc2</i>	<i>doc3</i>
auto	3	33	0
best	14	0	17
car	27	4	24
insurance	0	33	29

Exercício

2) Considere a coleção de documentos abaixo:



e a consulta $q = \text{"to do"}$. Vamos calcular o grau de similaridade entre cada documento e a consulta. Apresente os documentos e ordem decrescente de similaridade.

Exercício

3. Quais documentos são mais similares entre si?

	Doc1	Doc2	Doc3
<i>affection</i>	0.996	0.993	0.847
<i>jealous</i>	0.087	0.120	0.466
<i>gossip</i>	0.017	0.000	0.254

$$\text{sim}(d_j, d_k) = \vec{d}_j \bullet \vec{d}_k = \sum_{i=1}^t w_{i,j} \times w_{i,k}$$