CIS

Texto a Vectores

Machine Learning
Fernando Teshima
2022-2

UNIVERSIDAD DE LIMA

TF-IDF

Data usual para ML

x1	y	x2	x3	x4
sqft	price	City	bedrooms	baths
3392	339000	Dublin	3	2.1
4100	899900	pleasanton	4	3
3200	448641	Clayton	5	4
1436	239999	Moraga	4	3
1944	377500	Antioch	3	2
1500	299900	Danville	3	2.5
1700	265000	El Dorado Hills	4	3
2507	449000	Shingle Springs	4	3
1580	439950	McKinleyville	3	2
1500	699888	Marina	4	2
2705	1250000	Roseville	3	2
1715	439000	Rocklin	4	3

¿Textos?

- Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Comments

Admin User - Wed, 2 Sep 2020, 7:47 PM

Note that the maxeditingtime for this site is set to 5 minutes!

Terri Teacher - Wed, 2 Sep 2020, 7:48 PM

I never comment on pages unless I have something to say.

Max Manager - Wed, 2 Sep 2020, 7:49 PM

Also note that no emails are sent from this site; the feature is disabled for reasons of spam prevention.

Stopping, Stemming y TF-IDF

Preprocesamiento de Texto

Eliminar Stop Words

- Palabras que no aportan mucho contenido al texto
- “el”, “la”, “y”, “a”, “algunos”, “ahora”, “entonces”, “este”

Son palabras frecuentes que aparecen repetidamente en textos y no nos dan mayor contexto del contenido ni la información que contiene.

Stemming

- Eliminar diferencias irrelevantes de las diferentes “versiones” de la misma palabra.
- Reducir la palabra a su raíz

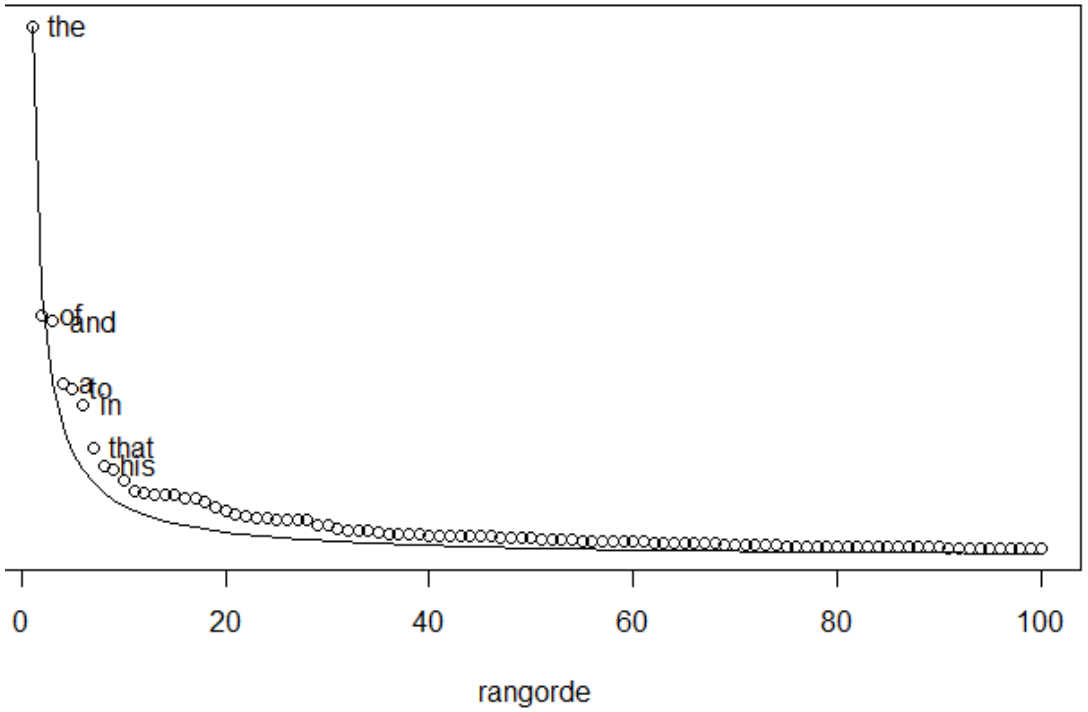
“bibliotecas”, “bibliotecario” -> “**bibliotec**”

Opcionalmente, explorar la relación semántica entre palabras. Si dos palabras tienen el mismo significado, tratarlas como la iguales.

Stemming

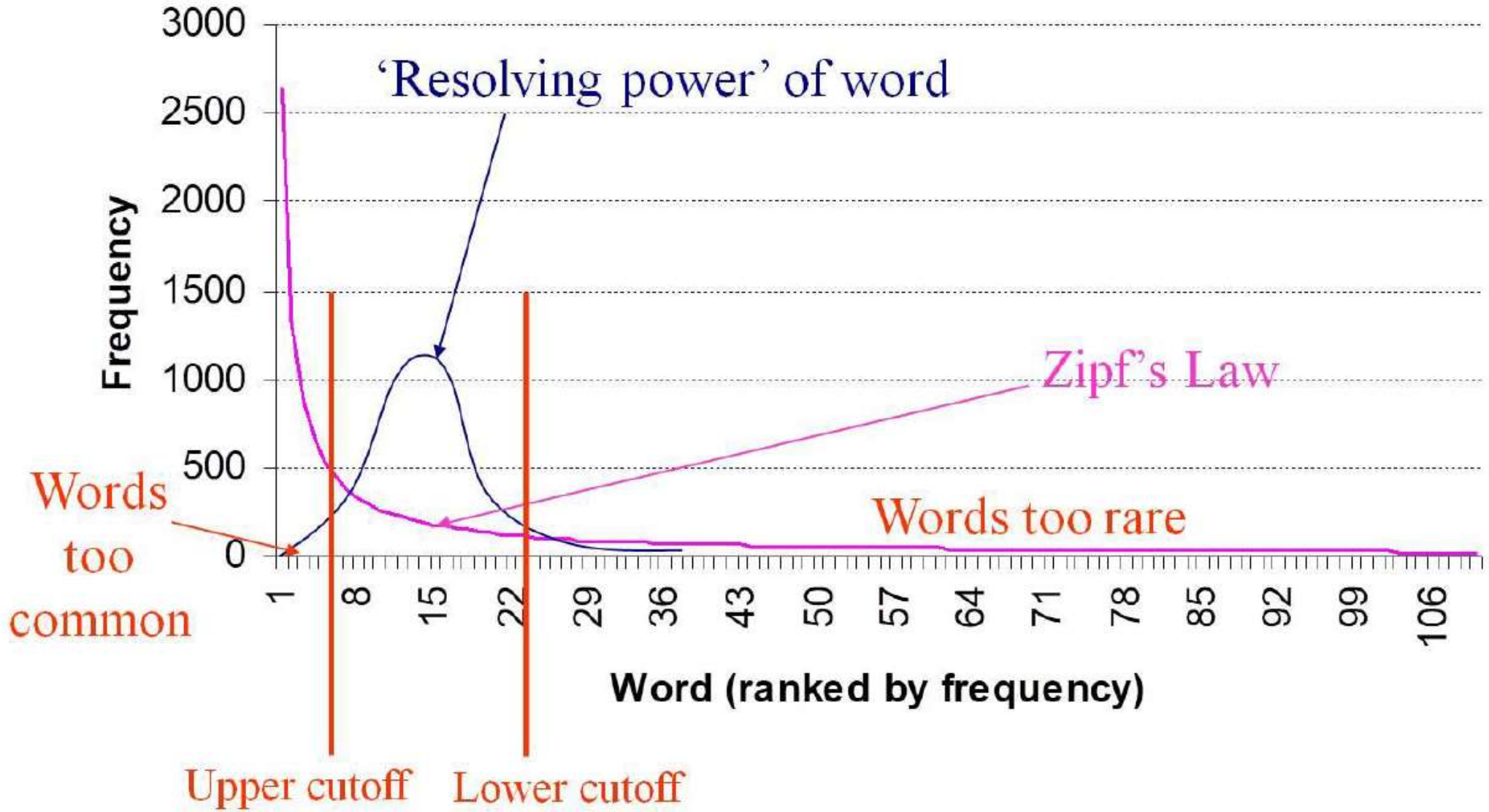
- **Idea general**: Si una consulta y un documento contienen diferentes formas de la misma palabra, entonces deben estar relacionadas.

Porqué Stemming



- Reducción del número de diferentes palabras en el corpus
- Poder de resolución de las palabras
- Ley de Zipf

- La implementación en código varía según lenguaje


3000
2500
2000
1500
1000
500
0
Frequency
'Resolving power' of word
Zipf's Law
Words too rare
Words too common
1
8
15
22
29
36
43
50
57
64
71
78
85
92
99
106
Word (ranked by frequency)
Upper cutoff
Lower cutoff

Ejemplo en inglés

Original first paragraph

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, `and what is the use of a book,' thought Alice `without pictures or conversation?'

Stop list 50 removed

alice beginning get very tired sitting sister bank having nothing do once twice peeped into book sister reading no pictures conversations what use book thought alice without pictures conversation

Stop list Brown removed

alice beginning tired sitting sister bank twice peeped book sister reading pictures conversations book alice pictures

IDF

Podemos determinar la importancia de cada término del “documento” usando Inverse Document Frecuency

Definido como

$$IDF(t) = \ln\left(\frac{ND}{ND_t}\right)$$

Donde

ND es el número total de documentos en el corpus

ND_t es el número de documentos que contienen el término $\boldsymbol{t}$

¿Por qué es útil IDF?

$$IDF(t) = \ln\left(\frac{ND}{ND_t}\right)$$

Caso 1: t aparece en todos los documentos

$$ND = ND_t$$

Por lo tanto

$$IDF(t) = 0$$

¿Por qué es útil IDF?

$$IDF(t) = \ln\left(\frac{ND}{ND_t}\right)$$

Caso 2: t ocurre de solo en algunos documentos

$$ND > ND_t$$

Por lo tanto

$$IDF(t) > 0$$

Nótese que $IDF(t)$ ignora la frecuencia de aparición de t

¿Y si aparece varias veces en el documento?

- Suponiendo una consulta q que consiste solamente del termino t
- Si el documento d_1 también consiste solo del término t
  - La coincidencia es perfecta

- Si el documento d_2 tiene 100 términos e incluye t
  - El número de ocurrencias es menor

TF-IDF (Term Frequency – Inverse Document Frequency)

- Considera la frecuencia de aparición del término

El peso $w_{t,d}$ del término t para el documento d es

$$w_{t,d} = f_{t,d} \cdot IDF(t)$$

Donde

$f_{t,d}$ es la frecuencia del término, el número de veces que ocurre t dentro de d

TF-IDF

$$IDF(t) = \ln\left(\frac{ND}{ND_t}\right)$$

$$w_{t,d} = f_{t,d} \cdot IDF(t)$$

Para que $w_{t,d}$ sea grande

$f_{t,d}$ debe ser grande, es decir t ocurre con frecuencia en d

$IDF(t)$ debe ser grande, es decir t aparece en pocos documentos

Consultas q

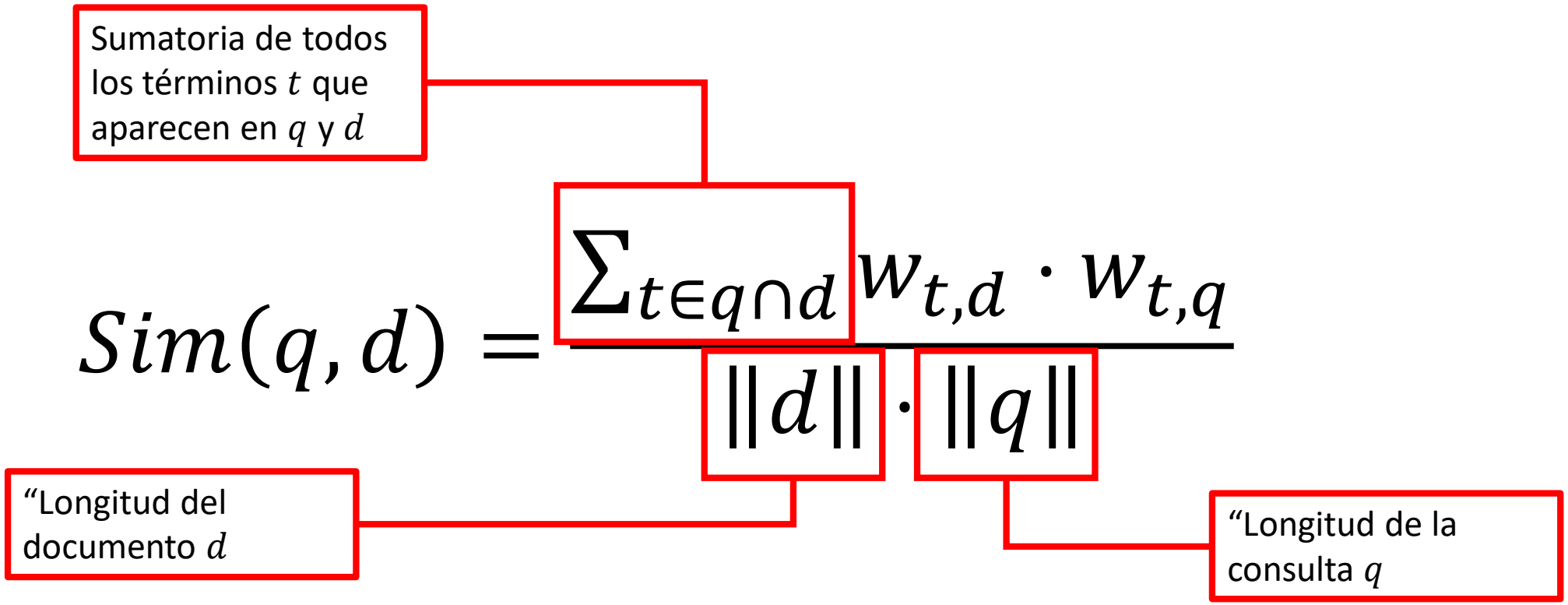
- Si la consulta q es larga, tiene varios términos y se repiten, se puede tratar q como un documento
- Los pesos se definen como

$$w_{t,q} = f_{t,q} \cdot IDF(t)$$

- Si q es pequeña, los pesos se definen como

$$w_{t,q} = IDF(t)$$

Similaridad TF-IDF



Longitud del documento

- Para cada término t en el documento d, se define el peso TF-IDF $w_{t,d}$
- La longitud del documento se define como

$$Len(d) = \|d\| = \sqrt{\sum_{t \in d} {w_{t,d}}^2}$$

Consideraciones prácticas

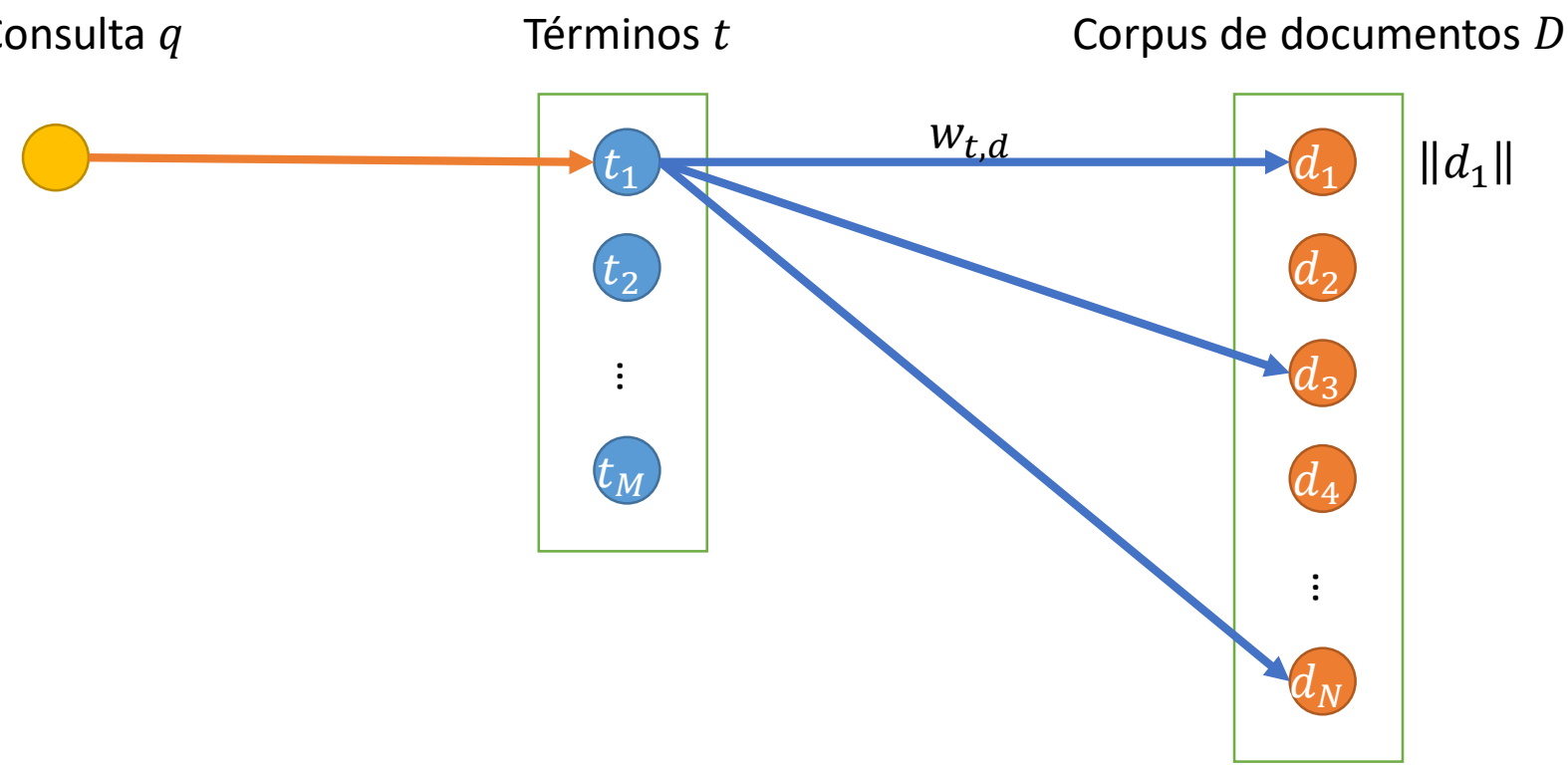
- Para consultas q
  - Calcular los pesos y la longitud no requiere de mucha computación.

- Para los documentos d,
  - La longitud $\|d\|$ puede ser calculada por adelantada
  - Los pesos $w_{t,d}$ también pueden ser calculados por adelantado
- Potencialmente se tiene gran número de cálculos.
- El tiempo para calcular todos los $Sim(q, d)$ puede ser enorme

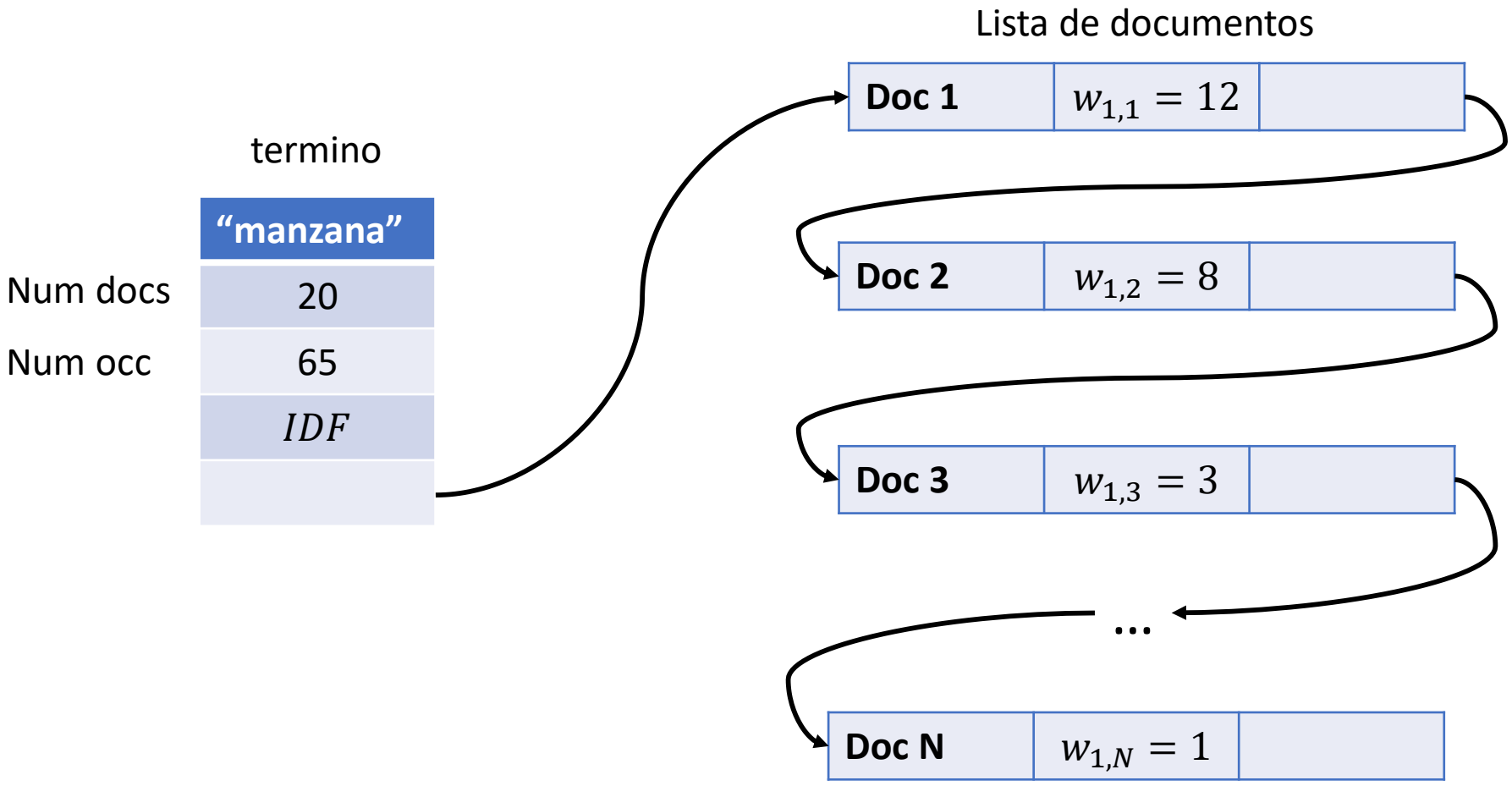
Consideraciones prácticas 2

- Si la consulta q tiene el término t y t no aparece en el corpus (lista de todos los documentos), no nos sirve t
- Identificar todos los documentos d que tienen t
- Puede demorar bastante
- Usamos una estructura llamada Document Index para agilizar el cálculo

Document Index



Document Index



Consideraciones

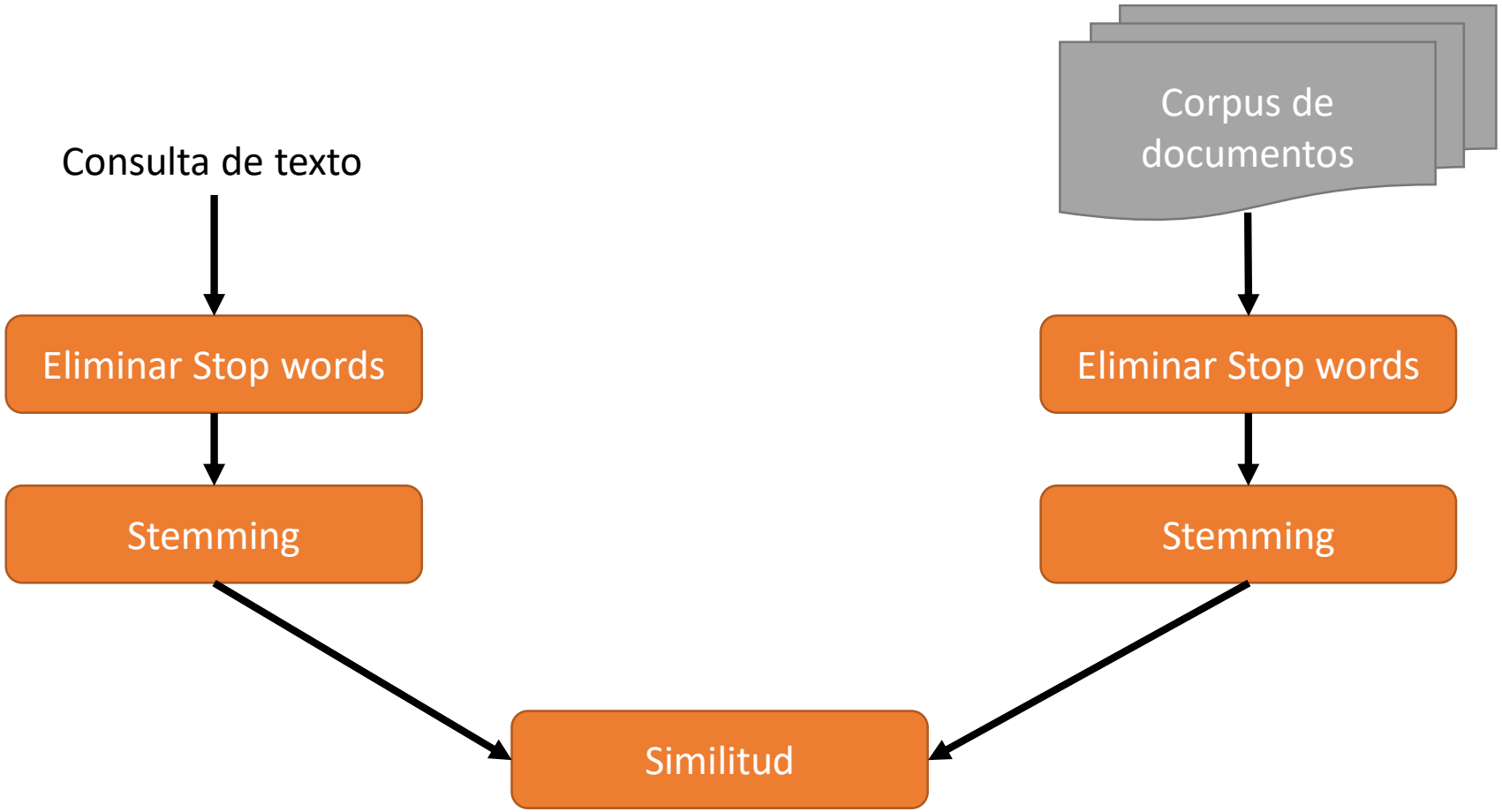
Ordenar los términos en orden descendiente según el IDF

Para cada término, ordenar los documentos en orden descendiente según el peso

Para cada término en la consulta

- Identificar el término en el índice
- Incrementar el puntaje de similitud del término por cada documento en la lista
- Opcionalmente parar cuando los pesos caen debajo de un umbral

Resumen del Proceso



Vectorizando documentos

Vectorizar Documentos

Supongamos que tenemos un set de documentos

$$D = \{d_1, d_2, \dots d_N\}$$

El Vocabulario $\boldsymbol{V}$ es el número de **diferentes** palabras del conjunto de documentos (corpus)

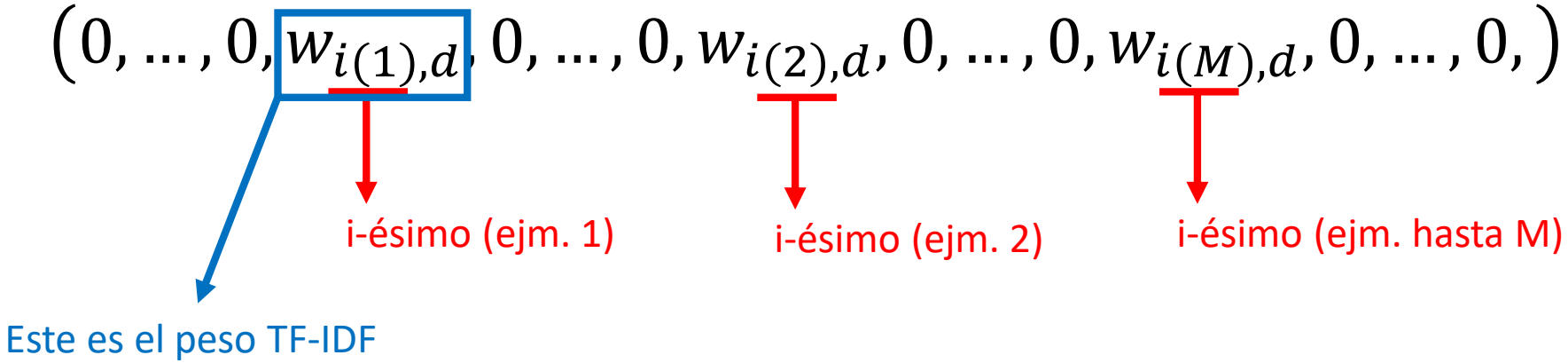
Ahora supongamos que un documento $\boldsymbol{d}$ contiene $\boldsymbol{M}$ términos diferentes $\{t_1, t_2, \dots t_M\}$

Y finalmente que el término t_i ocurre f_i veces

Vectorizar Documentos

- La representación vectorial $vec(d)$ del documento d es el vector de dimensión V:

$$(0, \dots, 0, w_{i(1),d}, 0, \dots, 0, w_{i(2),d}, 0, \dots, 0, w_{i(M),d}, 0, \dots, 0,)$$



- $vec(d)$ es el **vector documento** de d

¿Es único?

- ¿Es el mapeo de documentos a vectores uno-a-uno?
- ¿Si d_1 y d_2 son documentos, es verdad que $vec(d_1) = vec(d_2)$, si y solo si $d_1 = d_2$
- ¿Si λ es un escalar y $vec(d_1) = \lambda vec(d_2)$ que nos dice sobre d_1 y d_2?

Ejemplo

- $d_1 = El\ perro\ come\ la\ comida\ de\ perro \rightarrow perro\ come\ comida\ perro$
- $d_2 = El\ perro\ persigue\ al\ gato \rightarrow perro\ persigue\ gato$
- $d_3 = El\ gato\ quiere\ dormir\ en\ su\ cama \rightarrow gato\ quiere\ dormir\ cama$

- Vocabulario
- - cama, come, comida, dormir, gato, perro, persigue, quiere

Ejemplo

Palabra	d_1	d_2	d_3	nD	IDF	w_{t,d_1}	w_{t,d_2}	w_{t,d_3}
cama			1	1	1.1			1.1
come	1			1	1.1	1.1		
comida	1			1	1.1	1.1		
dormir			1	1	1.1			1.1
gato		1	1	2	0.4		1.1	1.1
perro	2	1		2	0.4	2.2	1.1	
persigue		1		1	1.1		1.1	
quiere			1	1	1.1			1.1

Ejemplo

$$vec(d_1) = \begin{bmatrix} 0 \\ 1.1 \\ 1.1 \\ 0 \\ 0 \\ 2.2 \\ 0 \\ 0 \end{bmatrix} \qquad vec(d_2) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1.1 \\ 1.1 \\ 1.1 \\ 0 \end{bmatrix} \qquad vec(d_3) = \begin{bmatrix} 1.1 \\ 0 \\ 0 \\ 1.1 \\ 1.1 \\ 0 \\ 0 \\ 1.1 \end{bmatrix}$$

Longitud del documento

- $vec(d) = \left(0, \dots, 0, w_{i(1),d}, 0, \dots, 0, w_{i(2),d}, 0, \dots, 0, w_{i(M),d}, 0, \dots, 0, \right)$
- $\|vec(d)\| = \sqrt{{w_{i(1),d}}^2 + {w_{i(2),d}}^2 + \cdots + {w_{i(M),d}}^2} = \|d\|$

Similitud de documentos

- Si dos documentos tiene las mismas palabras y en la misma proporción, entonces sus vectores apuntarán en la misma dirección
- Si tienen diferentes palabras, entonces apuntarán en diferentes direcciones
- Intuitivamente, mientras más diferentes sean los documentos, mayor será el ángulo entre ellas.

Similitud Coseno

$$\cos(\theta) = \frac{x_1 x_2 + y_1 y_2}{\|u\|\|v\|} = \frac{u \cdot v}{\|u\|\|v\|}$$

$$\cos(\theta) = \frac{vec(q) \cdot vec(d)}{\|q\|\|d\|} = \frac{\sum_{t \in q \cap d} w_{t,d} \cdot w_{t,q}}{\|d\| \cdot \|q\|} = Sim(q, d)$$