

# INTRODUCCIÓN A LA CIENCIA DE DATOS

---



UNIVERSIDAD DE MONTEVIDEO

---

# NATURAL LANGUAGE PROCESSING

---

# Objetivos

- Entender qué es **text mining**.
- Aprender a pre-procesar textos.
- Aprender a comparar documentos.
- Compilar un modelo *bag of words*.
- Identificar tópicos.
- Construir bi-gramas, tri-gramas, ..., n-gramas.

## Extraer conocimiento:



En 2011, Watson compitió contra competidores humanos legendarios y los sobrepasó! Watson tenía acceso a 200 millones de páginas de contenido estructurado como no estructurado, consumiendo 4 terabytes de almacenamiento de disco incluyendo todo el contenido disponible de Wikipedia.

## Detectar patrones:



Entre 1978 y 1995, el terrorista matemático americano Ted Kaczynski mató a 3 personas e hirió a otras 23. En 1995 publicó en el NY Times su manifiesto “*Industrial Society and Its Future*”. Su hermano David reconoció su estilo de escritura lo que llevó a su posterior captura por el FBI.

## Realizar búsquedas:



Google recibe más de 63.000 búsquedas por segundo en cualquier día. Este promedio se traduce en al menos 2 trillones de búsquedas anuales, 3.8 millones de búsquedas por minuto, 228 millones de búsquedas por hora o 5.6 billones de búsquedas por día.

## Clasificar textos:



Los algoritmos de clasificación de textos mantienen alejado el spam de nuestro inbox!

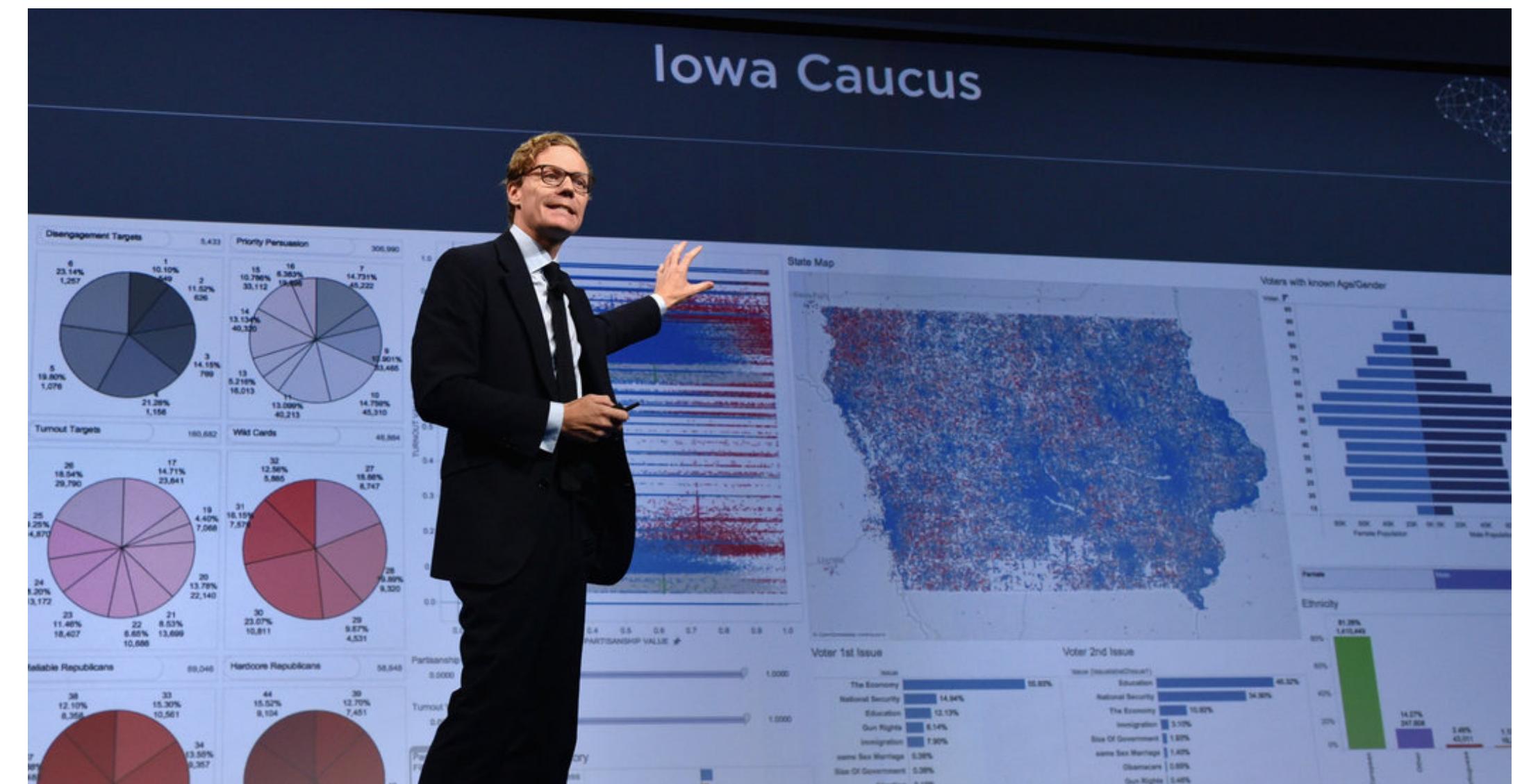
# Text mining

## Análisis de sentimientos:



Existen numerosas librerías y diccionarios que nos permiten evaluar los sentimientos encontrados en un texto. Esta es una tarea muy común dentro del Procesamiento de Lenguaje Natural.

## Campañas políticas: 😞



Cambridge Analytica fue una consultora política británica que se vió involucrada en el escándalo “Facebook-Analytica data scandal”, influenciando globalmente cientos de elecciones políticas.

Cómo extraemos atributos de textos?

1. Tokenization.
2. Case folding.
3. Removal of short and stop words.
4. Stemming.
5. Lemmatization.

# Tokenization

**Tokenization:** la aislación de unidades semejantes a palabras de un texto.

- Es la base para todo el procesamiento de texto posterior.
- La precisión de esta etapa afecta todo el procesamiento de más alto nivel.
- Tokenización por medio de: espacios, comas, puntos, etc.

La tokenización es ambigüa:

- State-of-the-art
- Résumé vs. Resume
- Computer science, San Francisco, Natural language processing, etc (noun phrases).
- Otros idiomas como el alemán o el holandés combinan palabras individuales para formar palabras compuestas. Por ejemplo, en un buscador alemán al buscar *Hund* debería devolvernos una búsqueda relacionada como *Windhund*. Pero una búsqueda como *Blau* no debería devolvernos una búsqueda relacionada como *Blauzungekrankheit*.
- Algunos lenguajes asiáticos no tienen espacios (ejemplo: Thai, Lao, o Khmer).
- Algunos lenguajes tienen morfologías muy complejas: Uygarlastiramadiklarimizdanmissinizcasina (turco).
- No hay un tokenizador que servirá en todos los escenarios. Es bueno añadir una tabla de referencia para palabras particulares.

# Case folding

**Case folding:** reducir todas las letras de una palabra a minúscula.

- Un algoritmo distingue las letras mayúsculas de las minúsculas, siendo caractéres distintos, por lo que a efectos prácticos debemos reducir todo a minúscula.

Esta técnica suele funcionar para la mayoría de los casos:

- “**Miren**, tómense su tiempo y **miren** la letra del parcial antes de comenzar.”

En ciertos casos no funciona:

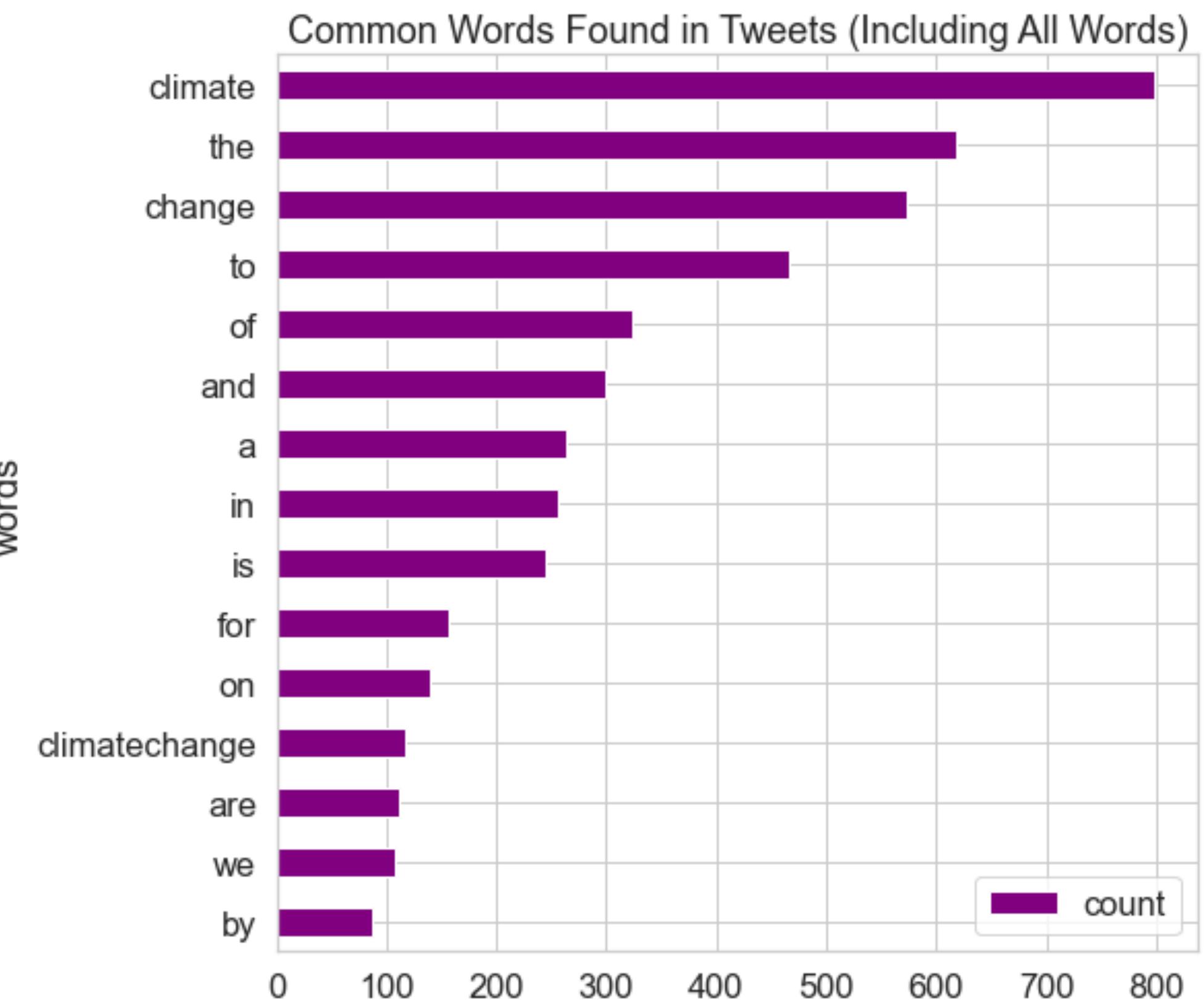
- General Motors ≠ general motors.
- The Who ≠ the who.

Suele solucionarse llevando a minúscula solo las palabras que poseen mayúscula al comienzo de una oración/frase y no llevando a minúscula palabras que poseen mayúscula en el medio de una oración/frase.

# Removal of stop words

**Removal of stop words:** típicamente consiste en remover palabras que poseen menos de 3 caractéres.

- Usualmente, en casi todo lenguaje, las palabras más utilizadas son palabras de poca longitud como pronombres y artículos. Esto conlleva a que éstas palabras aparezcan con mucho mayor frecuencia respecto al resto.
- Ejemplo: yo, tú, el, las, eso, ...
- Muchas veces se añaden a esta lista palabras adicionales que se usan con gran frecuencia en el contexto del texto con el que se trabaja. Por ejemplo, si tuviésemos un texto de la segunda guerra mundial quizá agregaríamos la palabra “batalla”.



# Stemming

**Stemming:** reducir una palabra a su “tallo” (stem) o raíz.

- Casi todas las palabras tienen una raíz en común de la cual se derivan. Por ejemplo, la raíz “corr-” del verbo “correr”:
- **Verbo:** correr, corrió, corriendo.
- **Número:** corredor, corredores.
- **Género:** corredor, corredora.

Dependiendo del lenguaje, existen reglas gramaticales para obtener las raíces de las palabras como la que se puede apreciar en la imagen.

Existen buenos foros de discusión respecto a ésta temática al igual que herramientas:

- <https://snowballstem.org/texts/introduction.html>
- <http://textanalysisonline.com>

Existen complicaciones dependiendo el contexto del habla. En redes sociales, donde se habla más informalmente, esto puede ser un problema:

- “OMG! This is sooooo LOOOVELY, sooo gooood, noooo waaay!!! 😍😍😍”

## Step 1a

SSES	->	SS	caresses	->	caress
IES	->	I	ponies	->	poni
SS	->	SS	ties	->	ti
S	->		caress	->	caress
			cats	->	cat

## Step 1b

(m>0) EED	->	EE	feed	->	feed
(*v*) ED	->		agreed	->	agree
(*v*) ING	->		plastered	->	plaster
			bled	->	bled
			motoring	->	motor
			sing	->	sing

# Lemmatization

**Lemmatization:** obtener la palabra que engloba todas las formas derivadas.

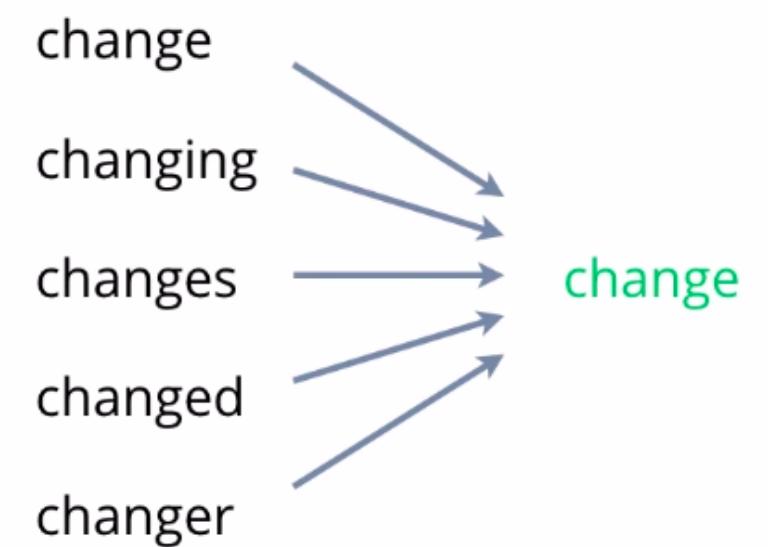
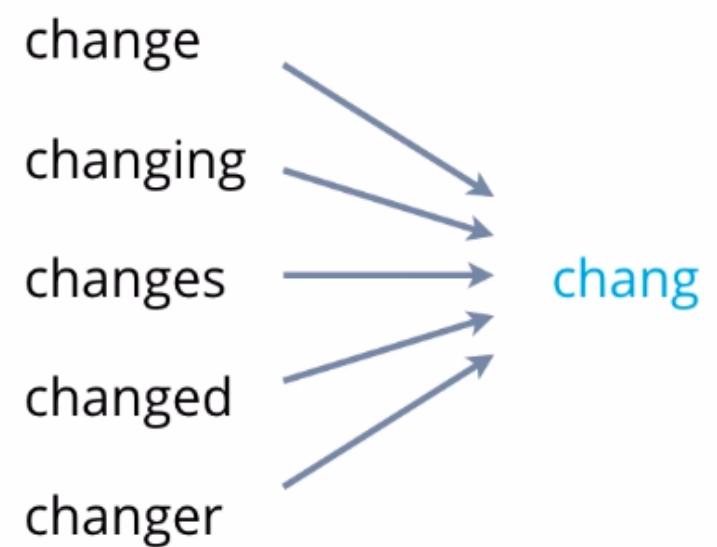
- Más complejo que stemming.
- Toma en consideración el contexto.
- Usa algún tipo de diccionario.
- Implica agrupar todas las palabras derivadas a una base común.

Por un lado, Stemming usa algún algoritmo heurístico para cortar palabras a una raíz común, siendo correcto, eficiente y fácil de implementar la mayoría de los casos. En cambio, Lemmatization hace un análisis morfológico de las palabras, devolviendo una palabra, diccionario o forma de la misma, siendo más lento al usar el vocabulario y contexto.

Ejemplos:

- “better” → lemma: “good”. Esto no lo captura stemming.
- “walking” → lemma: “walk”, stem: “walk”.
- “meeting” → dos significados: “reunión” o “reuniendo”. Lemmatization tiene en cuenta el contexto.

## Stemming vs Lemmatization



# NLP Word features

- **Root words:** una palabra o parte de una palabra que sirve de base para formar nuevas palabras mediante la adición de un prefijo o sufijo.
- **Part Of Speech Tags (POSTagging):** es el proceso de etiquetar o marcar una palabra en un texto con el fin de asociarla a una parte del habla.
- **Named entities:** un objeto del mundo real, como puede ser una persona, una organización, una ubicación, un producto, etc.



contentSkip to site indexPoliticsSubscribeLog InSubscribeLog InToday's PaperAdvertisementSupported ORG byF.B.I. Agent Peter Strzok PERSON , Who Criticized Trump PERSON in Texts, Is FiredImagePeter Strzok, a top F.B.I. GPE counterintelligence agent who was taken off the special counsel investigation after his disparaging texts about President Trump PERSON were uncovered, was fired. CreditT.J. Kirkpatrick PERSON for The New York TimesBy Adam Goldman ORG and Michael S. SchmidtAug PERSON . 13 CARDINAL , 2018WASHINGTON CARDINAL — Peter Strzok PERSON , the F.B.I. GPE senior counterintelligence agent who disparaged President Trump PERSON in inflammatory text messages and helped oversee the Hillary Clinton PERSON email and Russia GPE investigations, has been fired for violating bureau policies, Mr. Strzok PERSON 's lawyer said Monday DATE .Mr. Trump and his allies seized on the texts — exchanged during the 2016 DATE campaign with a former F.B.I. GPE lawyer, Lisa Page — in PERSON assailing the Russia GPE investigation as an illegitimate "witch hunt." Mr. Strzok PERSON , who rose over 20 years DATE at the F.B.I. GPE to become one of its most experienced counterintelligence agents, was a key figure in the early months DATE of the inquiry.Along with writing the texts, Mr. Strzok PERSON was accused of sending a highly sensitive search warrant to his personal email account.The F.B.I. GPE had been under immense political pressure by Mr. Trump PERSON to dismiss Mr. Strzok PERSON , who was removed last summer DATE from the staff of the special counsel, Robert S. Mueller III PERSON . The president has repeatedly denounced Mr. Strzok PERSON in posts on



# Language corpora

**Corpus:** un conjunto de documentos, público y estático (usualmente anotados).

- Suelen usarse para análisis estadísticos y testeo de hipótesis.
- Pueden ser pequeños (~50.000 palabras) o enormes (millones de palabras).
- Distintos corpus corresponden a distintos contextos del lenguaje. Por ejemplo, puede haber un corpus de e-mails empresariales (habla formal), un corpus de tweets (habla informal), discursos políticos (lenguaje político), corpus científico, etc.
- Referencia para benchmarking de algoritmos.

Nos ayudan a descubrir patrones como por ejemplo:

- Cuáles son las palabras más utilizadas en un contexto?
- Qué sentimientos predominan en un contexto?
- Qué tiempos verbales son más utilizados?
- Qué preposiciones se encuentran previo a ciertos verbos?
- Qué sustantivos, adjetivos y verbos son más empleados en un contexto?
- Qué cantidad de palabras un hablante nativo utiliza en el día a día?



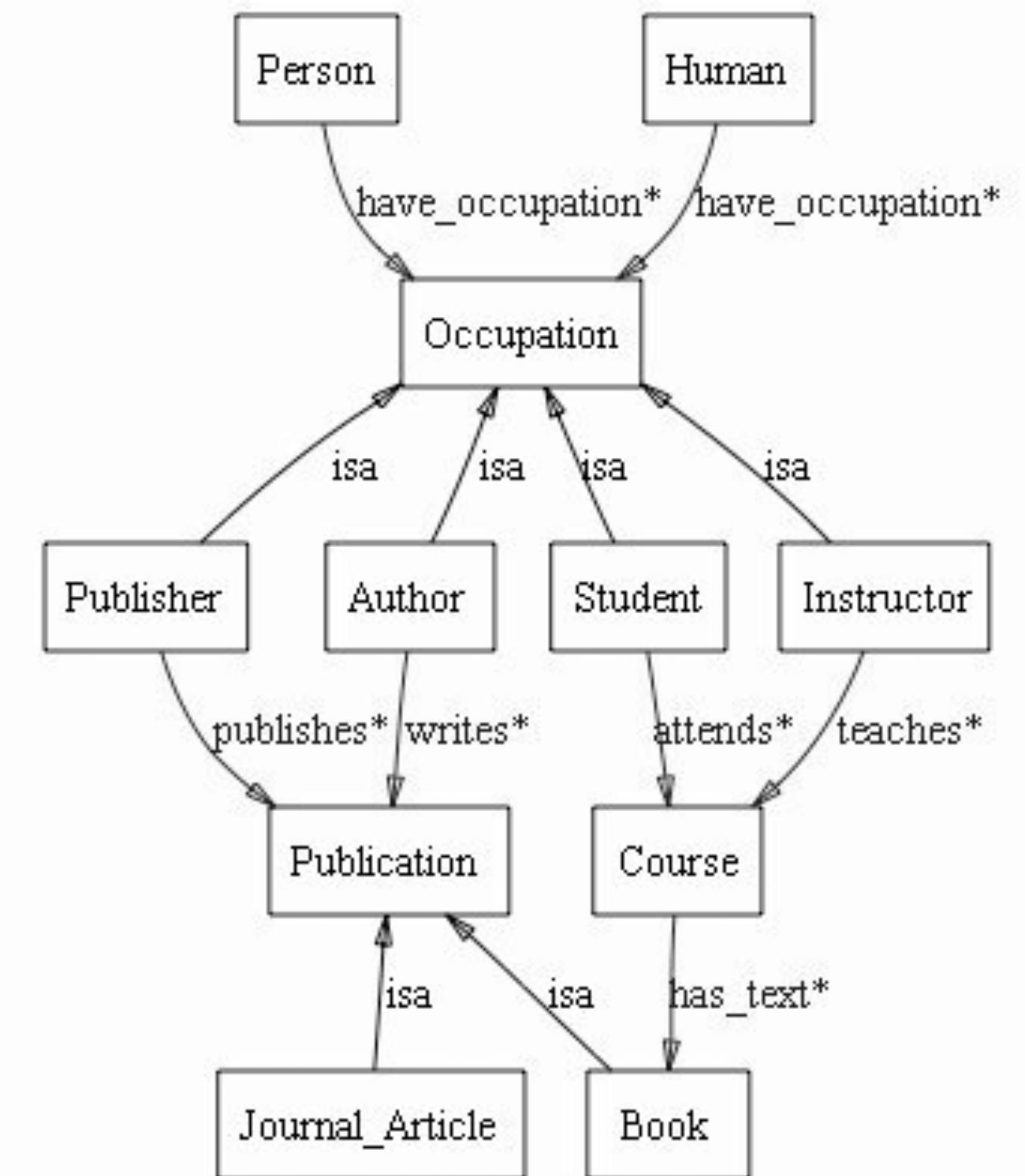
# Word feature: Information content

- **Information Content (IC)**: métrica que denota la importancia de un término en un corpus.

Combina tanto:

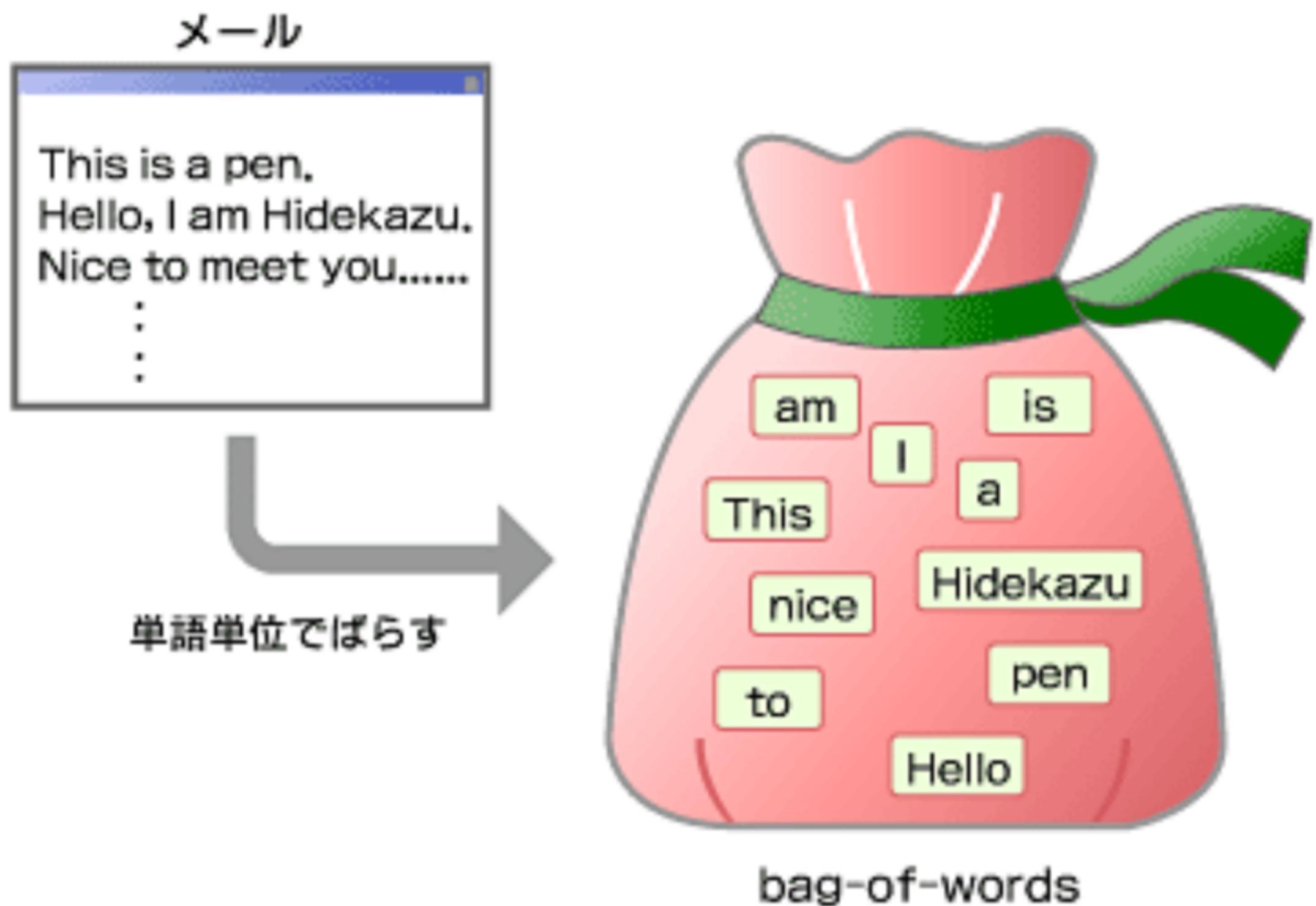
1. Conocimiento de su estructura jerárquica dentro de una ontología (cómo entidades son agrupadas en categorías básicas). Ej: clases en Object Oriented Programming.
2. Estadísticas del empleo de una palabra en un texto a partir de un corpus.

- Ejemplo: IC(collar) > IC(joya) pues “joya” es más genérico y por ende menos “interesante”.
- Problemas: los textos fuera del corpus reciben IC = 0.



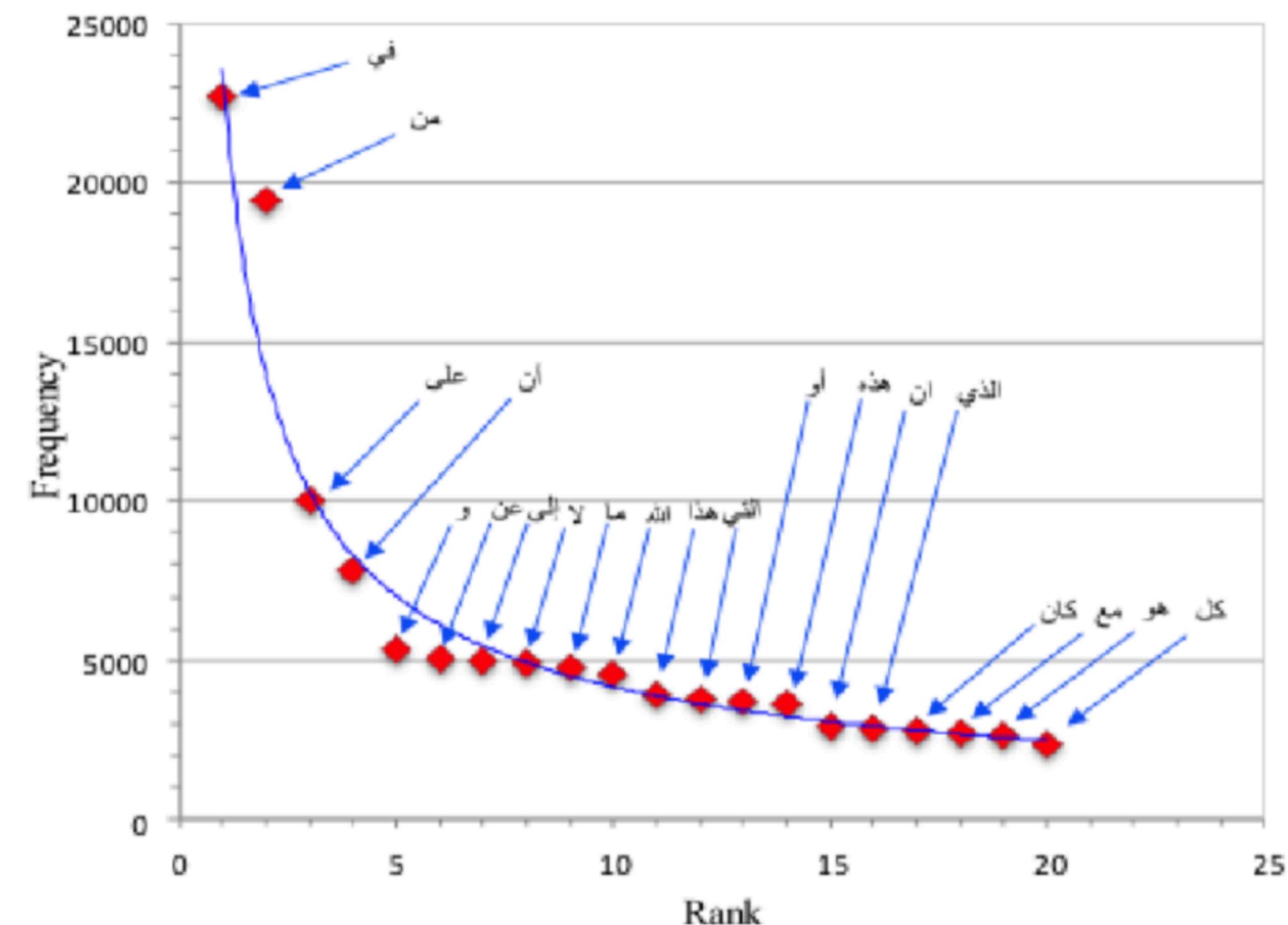
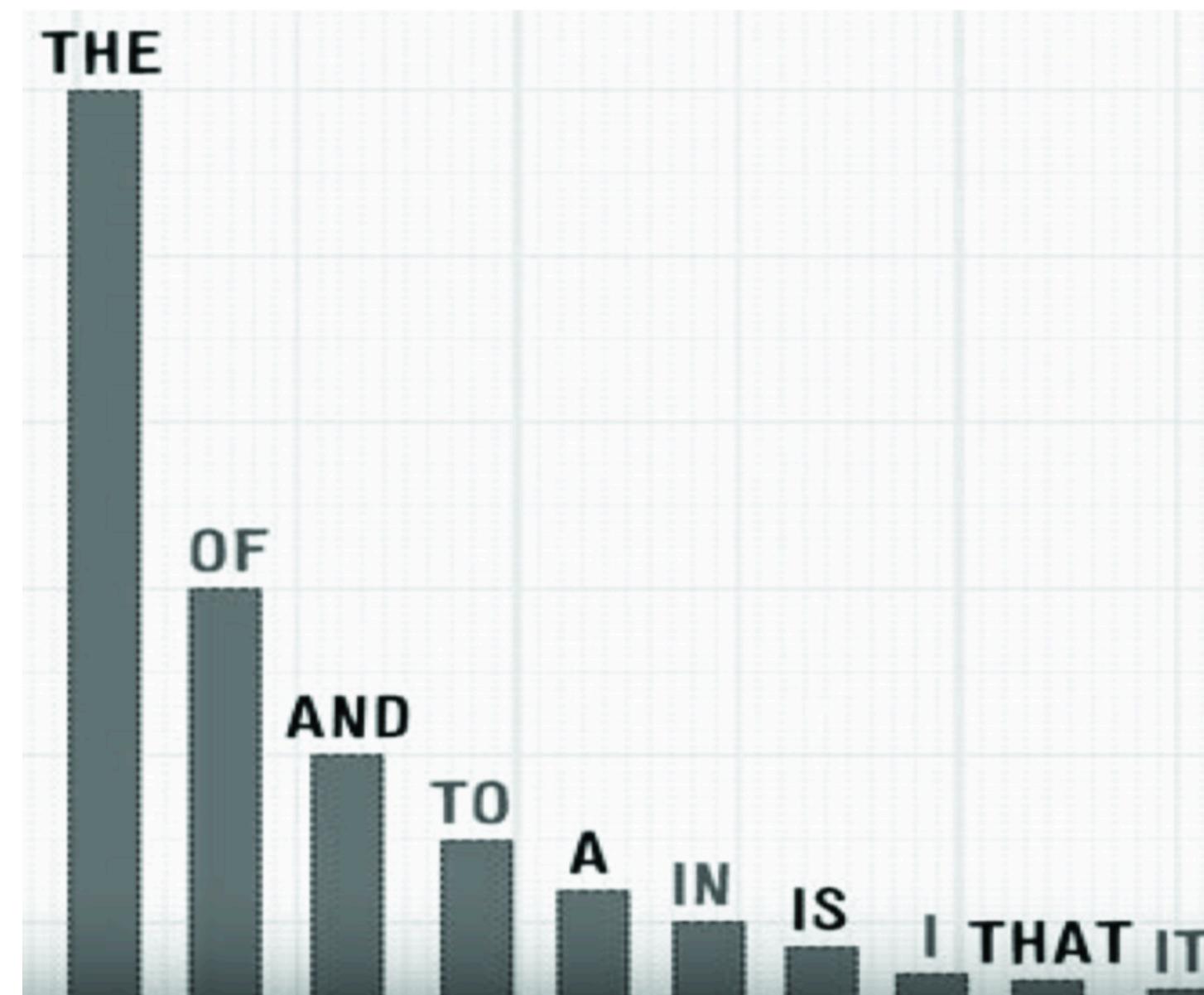
# Bag of Words (BOW)

- Un conjunto de palabras que ignora el orden, el contexto, la jerarquía, etc.
- Cada palabra es considerada un término o token.
- “El perro muerde al hombre” vs. “El hombre muerde al perro”.
- BOW = {“el” , “muerde”, “perro”, “hombre”, “al”}



# Term Frequency

- **Term Frequency:** el peso de cada *término* en un *documento*.
  - Es proporcional al número de ocurrencias de ese término en el documento.
  - $$TF(term, doc) = \frac{\# \text{ocurrencias del término en el documento}}{\text{cantidad de palabras del documento}}$$
  - A grandes rasgos, sigue la Ley de Zipf: el i-ésimo término tiene una frecuencia aproximada de  $1/i$
  - Más intuitivamente, el primer término tendrá el doble de ocurrencias que el segundo, el triple que el tercero y así sucesivamente.
  - Es por esto que a veces trabajamos con  $TF = \log(TF)$



# Term Frequency x Inverse Document Frequency (TF.IDF)

- **Term Frequency Inverse Document Frequency:** estadístico numérico que refleja la importancia de una palabra dentro de un documento a su vez dentro de un corpus.
- Contempla qué tan importante es una palabra dentro de un documento si:
  1. Qué tanto aparece en un documento (TF)
  2. Qué tan rara vez aparece en otros documentos del corpus (IDF)

Por ejemplo, si tuviésemos una colección de libros de historia, la palabra “Guerra” tendrá seguramente un TF muy grande pues es esperable que aparezca frecuentemente en un libro de historia. Sin embargo, es posible que también aparezca **en muchos** libros de historia por lo que su IDF será extremadamente bajo, resultando en TF.IDF bajo.

En cambio, si encontramos la palabra “Guerra” en un libro de matemática en una colección de tomos de matemática seguramente tenga un TF.IDF alto.

# Term Frequency x Inverse Document Frequency (TF.IDF)

Existen muchas maneras de calcular el TF.IDF

- **N** = número de documentos.
- **Document Frequency: DF (term, docs)** = cantidad de documentos en los cuales el término aparece.
- DF suele también escribirse de la siguiente manera:  $|\{d \in D : t \in d\}|$
- **Inverse Document Frequency: IDF(term, docs)** =  $\log \frac{N}{|\{d \in D : t \in d\}|}$
- **TF.IDF(term, doc, docs)** = TF(term, doc) \* IDF(term, docs)

Variants of term frequency (tf) weight

weighting scheme	tf weight
binary	0, 1
raw count	$f_{t,d}$
term frequency	$f_{t,d} / \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Variants of inverse document frequency (idf) weight

weighting scheme	idf weight ( $n_t =  \{d \in D : t \in d\} $ )
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left( \frac{N}{1 + n_t} \right) + 1$
inverse document frequency max	$\log \left( \frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t} \right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

**Vector Space Model:** la representación de un conjunto de documentos como vectores en un espacio común.

- Cada término  $t$  de un diccionario es considerado como una dimensión.
- Un documento  $d$  puede ser representado por el peso de cada término del vocabulario:

$$V(d) = (w(t_1, d), w(t_2, d), \dots, w(t_n, d))$$

Nótese que la frecuencia puede ser utilizada como peso. Ejemplo: representación de 3 documentos  $d_1$ ,  $d_2$  y  $d_3$  usando frecuencia.

**Pregunta:** cómo calculamos la similitud entre documentos?

	$d_1$	$d_2$	$d_3$
affection	115	58	20
jealous	10	7	11
gossip	2	0	6

**Similitud entre vectores:** un primer approach puede ser el producto interno.

$$\text{Producto interno: } \mathbf{V}(d_1) \cdot \mathbf{V}(d_2)$$

Qué hay de la longitud de vectores? Documentos más largos serán representados por vectores más largos, pero eso no implica que tengan más relevancia. Como todas las entradas de los vectores son no negativas entonces el hecho de tener más palabras solo incrementaría la similitud si usaramos el producto interno

**Normalización Euclídea:** podemos normalizar el vector documento para que tenga longitud 1 y evitar este último problema.

$$\mathbf{v}(d) = \frac{\mathbf{V}(d)}{\|\mathbf{V}(d)\|} \quad \text{donde } \|\mathbf{V}(d)\| = \sqrt{\sum_{i=1}^n x_i^2}$$

## Ejemplo:

- $\text{Sim}(d_1, d_2) = \mathbf{v}(d_1) \cdot \mathbf{v}(d_2)$

- $\mathbf{v}(d_1) \cdot \mathbf{v}(d_2) = \sum_{i=1}^n d_{1i} d_{2i}$

- **Normalizando por longitud:**  $\mathbf{v}(d_i) = \frac{\mathbf{v}(d_i)}{\|\mathbf{v}(d)\|}$

- **Distancia Euclídea:**  $\|\mathbf{v}(d)\| = \sqrt{\sum_{i=1}^n x_i^2}$

vocabulary	d1	d2	d3
1: affection	115	58	20
2: jealous	10	7	11
3: gossip	2	0	6

$$\|\mathbf{v}(d_1)\| = \sqrt{115^2 + 10^2 + 2^2}$$

$$\|\mathbf{v}(d_{11})\| = \frac{115}{\sqrt{115^2 + 10^2 + 2^2}} = 0.996$$

$$\|\mathbf{v}(d_{12})\| = \frac{10}{\sqrt{115^2 + 10^2 + 2^2}} = 0.087$$

$$\|\mathbf{v}(d_{13})\| = \frac{2}{\sqrt{115^2 + 10^2 + 2^2}} = 0.017$$

## Ejemplo:

- $\text{Sim}(d_1, d_2) = \mathbf{v}(d_1) \cdot \mathbf{v}(d_2)$

- $\mathbf{v}(d_1) \cdot \mathbf{v}(d_2) = \sum_{i=1}^n d_{1i} d_{2i}$

- **Normalizando por longitud:**  $\mathbf{v}(d_i) = \frac{\mathbf{v}(d_i)}{\|\mathbf{v}(d)\|}$

- **Distancia Euclídea:**  $\|\mathbf{v}(d)\| = \sqrt{\sum_{i=1}^n x_i^2}$

vocabulary	$d1$	$d2$	$d3$
1: affection	115	58	20
2: jealous	10	7	11
3: gossip	2	0	6

$$\|\mathbf{v}(d_2)\| = \sqrt{58^2 + 7^2 + 0^2}$$

$$\|\mathbf{v}(d_{21})\| = \frac{58}{\sqrt{58^2 + 7^2 + 0^2}} = 0.993$$

$$\|\mathbf{v}(d_{22})\| = \frac{7}{\sqrt{58^2 + 7^2 + 0^2}} = 0.120$$

$$\|\mathbf{v}(d_{23})\| = \frac{0}{\sqrt{58^2 + 7^2 + 0^2}} = 0$$

## Ejemplo:

- $\text{Sim}(d_1, d_2) = \mathbf{v}(d_1) \cdot \mathbf{v}(d_2)$

- $\mathbf{v}(d_1) \cdot \mathbf{v}(d_2) = \sum_{i=1}^n d_{1i} d_{2i}$

- **Normalizando por longitud:**  $\mathbf{v}(d_i) = \frac{\mathbf{v}(d_i)}{\|\mathbf{v}(d)\|}$

- **Distancia Euclídea:**  $\|\mathbf{v}(d)\| = \sqrt{\sum_{i=1}^n x_i^2}$

vocabulary	d1	d2	d3
1: affection	115	58	20
2: jealous	10	7	11
3: gossip	2	0	6

$$\|\mathbf{v}(d_3)\| = \sqrt{20^2 + 11^2 + 6^2}$$

$$\|\mathbf{v}(d_{31})\| = \frac{20}{\sqrt{20^2 + 11^2 + 6^2}} = 0.847$$

$$\|\mathbf{v}(d_{32})\| = \frac{11}{\sqrt{20^2 + 11^2 + 6^2}} = 0.466$$

$$\|\mathbf{v}(d_{33})\| = \frac{6}{\sqrt{20^2 + 11^2 + 6^2}} = 0.254$$

Cada nuevo documento  $n$  es representado usando vectores de la misma manera

- $\text{sim}(n, d) = \mathbf{v}(n) \cdot \mathbf{v}(d)$
- Siendo  $n = \langle \text{jealous}, \text{gossip} \rangle$

	$d_1$	$d_2$	$d_3$	$n$
affection	115	58	20	0
jealous	10	7	11	1
gossip	2	0	6	1

Obtenemos:

- $\text{sim}(d_n, d_1) = 0.074$
- $\text{sim}(d_n, d_2) = 0.085$
- $\text{sim}(d_n, d_3) = 0.509$

Útiles para:

1. Búsquedas
2. Clustering
3. Descubrir los temas principales de un corpus

Modelos estadísticos que:

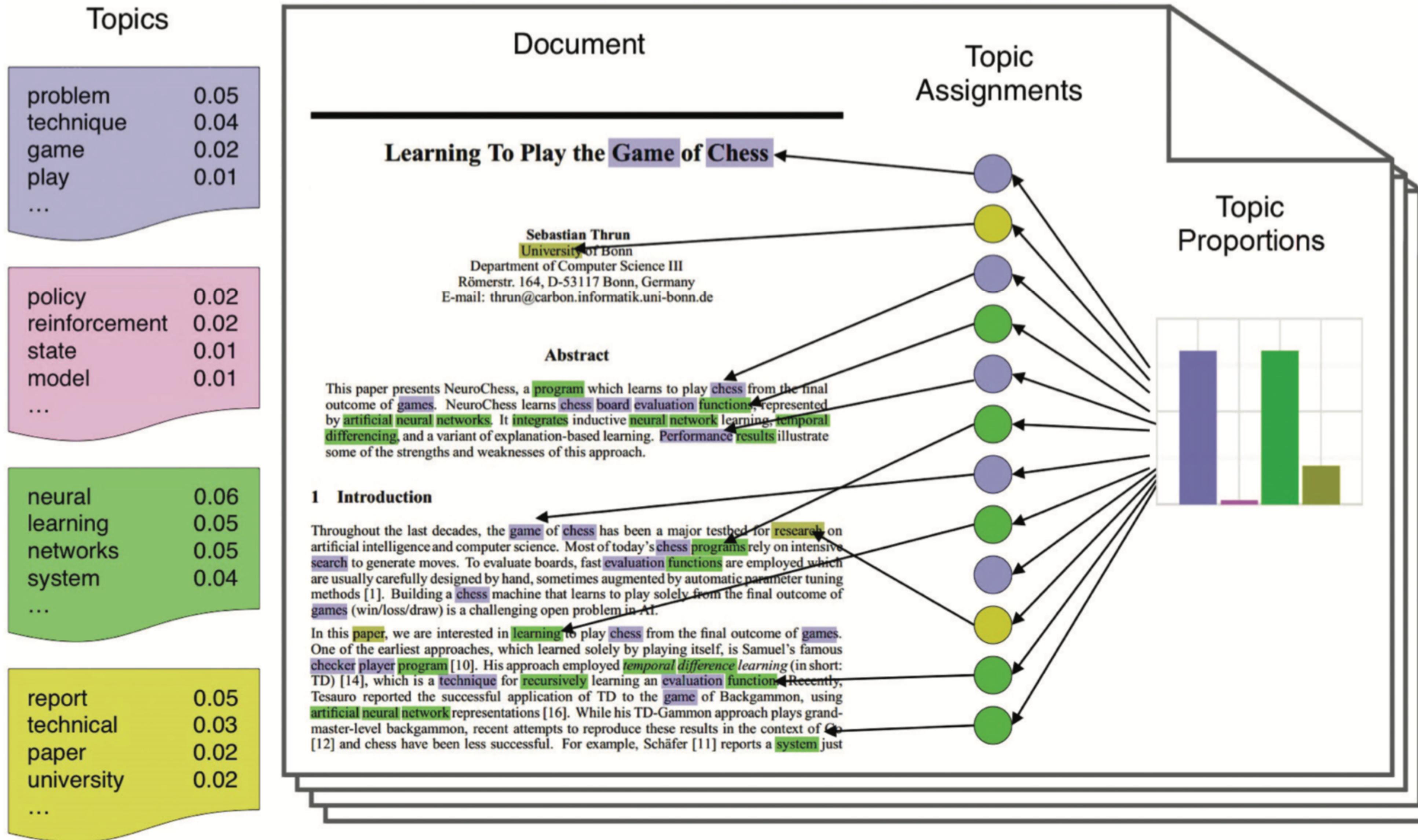
1. Determinan los temas tratados en un documento/corpus.
2. Útiles para organizar, buscar y resumir.

Basado en la **hipótesis de distribución**, es decir, palabras con significados similares aparecerán en textos similares.

**Topic:**

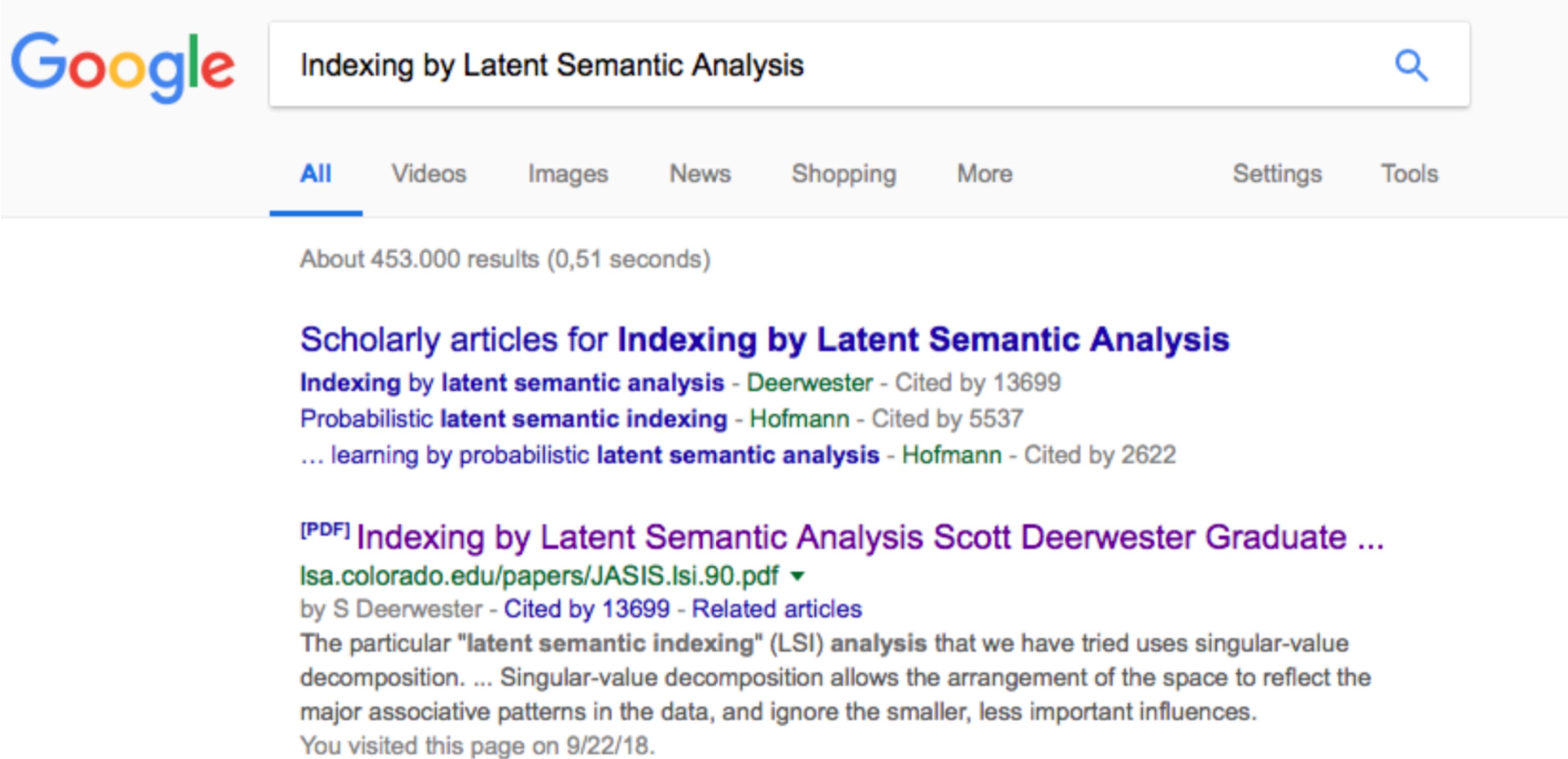
- Cluster de palabras que frecuentemente ocurren juntas y comparten el mismo tema.
- Formalmente, una distribución dentro de un vocabulario fijo de palabras. Diferentes tópicos tendrán distribuciones diferentes de palabras.
- Pueden emerger por sí solos en un análisis de textos.

# Topic Models example



- Un documento es representado como una mezcla de tópicos latentes.
- El número de tópicos latentes  $K$  está predefinido.
- Un documento puede pertenecer a varios tópicos.
- Los tópicos no tienen por qué estar normalmente distribuídos.
- LDA es uno de los modelos de tópicos más simples.
- Ampliamente implementado.

- Hay muchos artículos de investigación en la literatura sobre Latent Semantic Analysis



A screenshot of a Google search results page. The search query "Indexing by Latent Semantic Analysis" is entered in the search bar. The "All" tab is selected, and the results show approximately 453,000 results found in 0,51 seconds. The top result is a scholarly article titled "Scholarly articles for Indexing by Latent Semantic Analysis". Below it are three other links related to latent semantic analysis. The snippet for the first result discusses the use of singular-value decomposition in LSI analysis.

Google Indexing by Latent Semantic Analysis

All Videos Images News Shopping More Settings Tools

About 453.000 results (0,51 seconds)

Scholarly articles for Indexing by Latent Semantic Analysis

[Indexing by latent semantic analysis](#) - Deerwester - Cited by 13699

[Probabilistic latent semantic indexing](#) - Hofmann - Cited by 5537

[... learning by probabilistic latent semantic analysis](#) - Hofmann - Cited by 2622

[PDF] [Indexing by Latent Semantic Analysis Scott Deerwester Graduate ...](#)  
[Isa.colorado.edu/papers/JASIS.Isi.90.pdf](http://lsa.colorado.edu/papers/JASIS.Isi.90.pdf) ▾  
by S Deerwester - Cited by 13699 - Related articles  
The particular "latent semantic indexing" (LSI) analysis that we have tried uses singular-value decomposition. ... Singular-value decomposition allows the arrangement of the space to reflect the major associative patterns in the data, and ignore the smaller, less important influences.  
You visited this page on 9/22/18.

# Latent Dirichlet Allocation

## Titles:

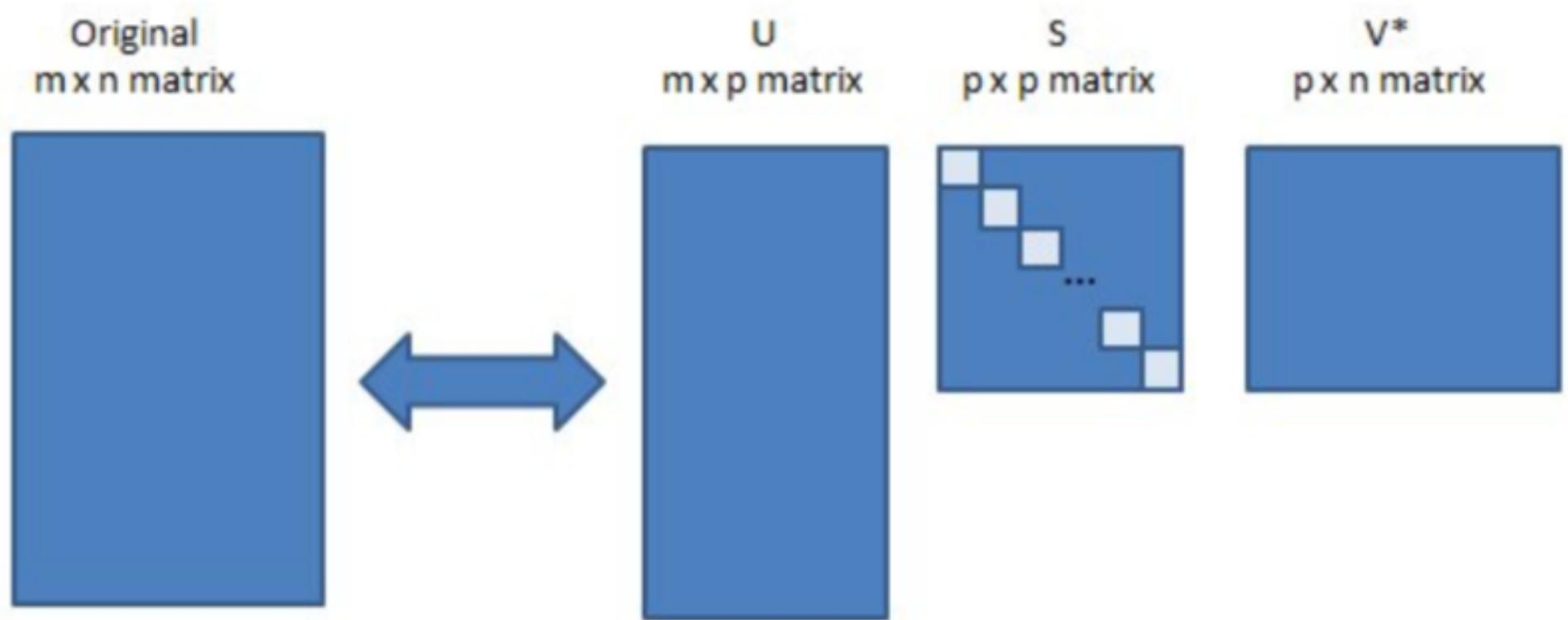
- c1: *Human machine interface* for Lab ABC *computer applications*
  - c2: A survey of *user* opinion of *computer system response time*
  - c3: The *EPS user interface* management system
  - c4: *System and human system* engineering testing of *EPS*
  - c5: Relation of *user-perceived response time* to error measurement
- 
- m1: The generation of random, binary, unordered *trees*
  - m2: The intersection *graph* of paths in *trees*
  - m3: *Graph minors IV: Widths of trees and well-quasi-ordering*
  - m4: *Graph minors: A survey*

Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

- Query: “Human computer interaction”.
- Utilizando técnicas simples de matching esta búsqueda nos devolvería los documentos c1, c2 y c4, ya que comparten uno o más términos de la query.
- Sin embargo, los documentos c3 y c5, los cuales son también relevantes, no son devueltos al no compartir ningún término en común.

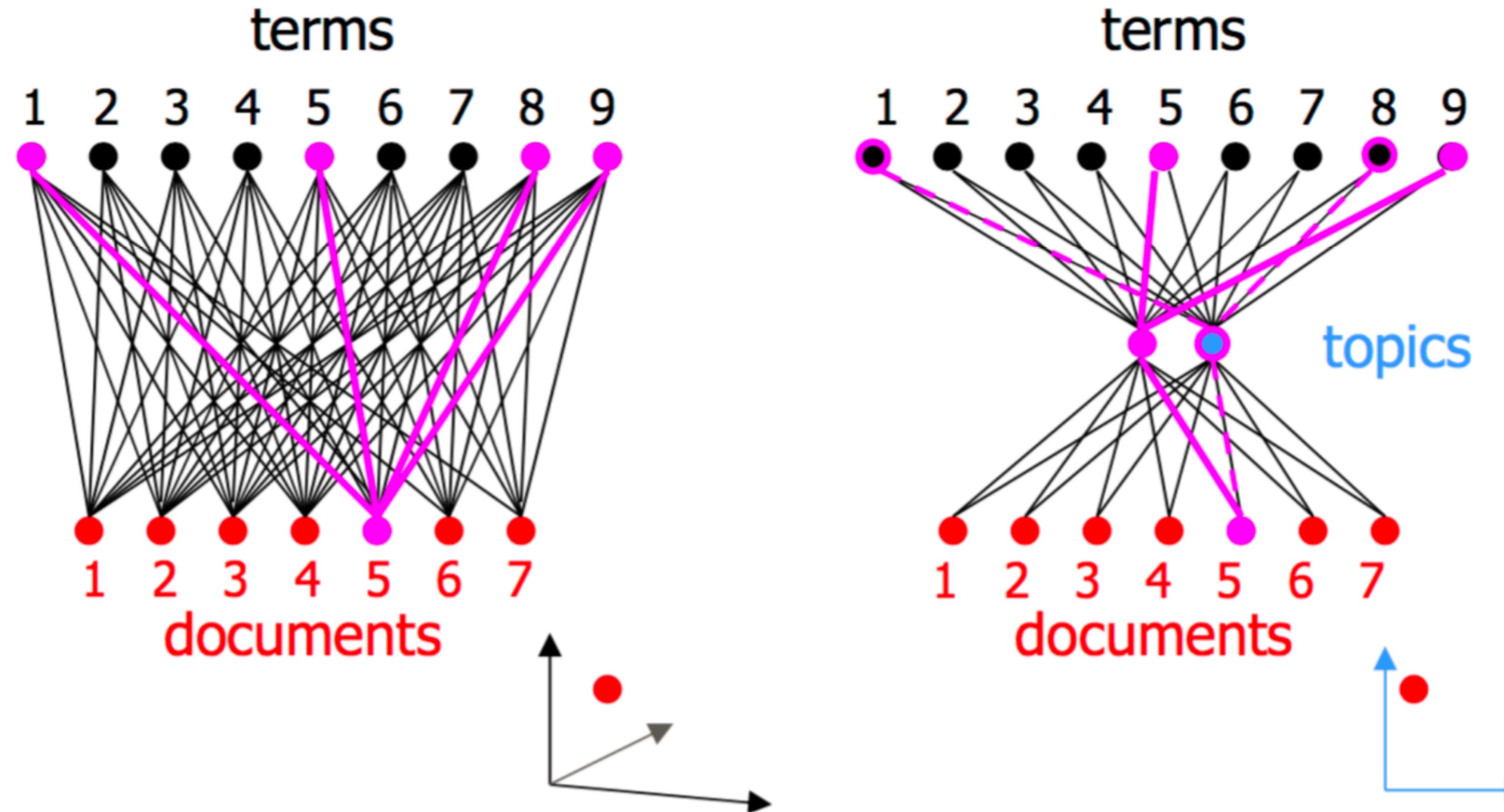
# Latent Semantic Analysis

- Crear una matriz de términos  $\mathbf{M}$ . Las filas corresponden a documentos y las columnas a los términos. Las celdas se rellenan con el número de ocurrencias de cada término/TFIDF/etc.
- Aplicar Descomposición en Valores Singulares (SVD) a  $\mathbf{M}$ .
- Reducir la dimensionalidad de la matriz sigma  $\Sigma$  en SVD.



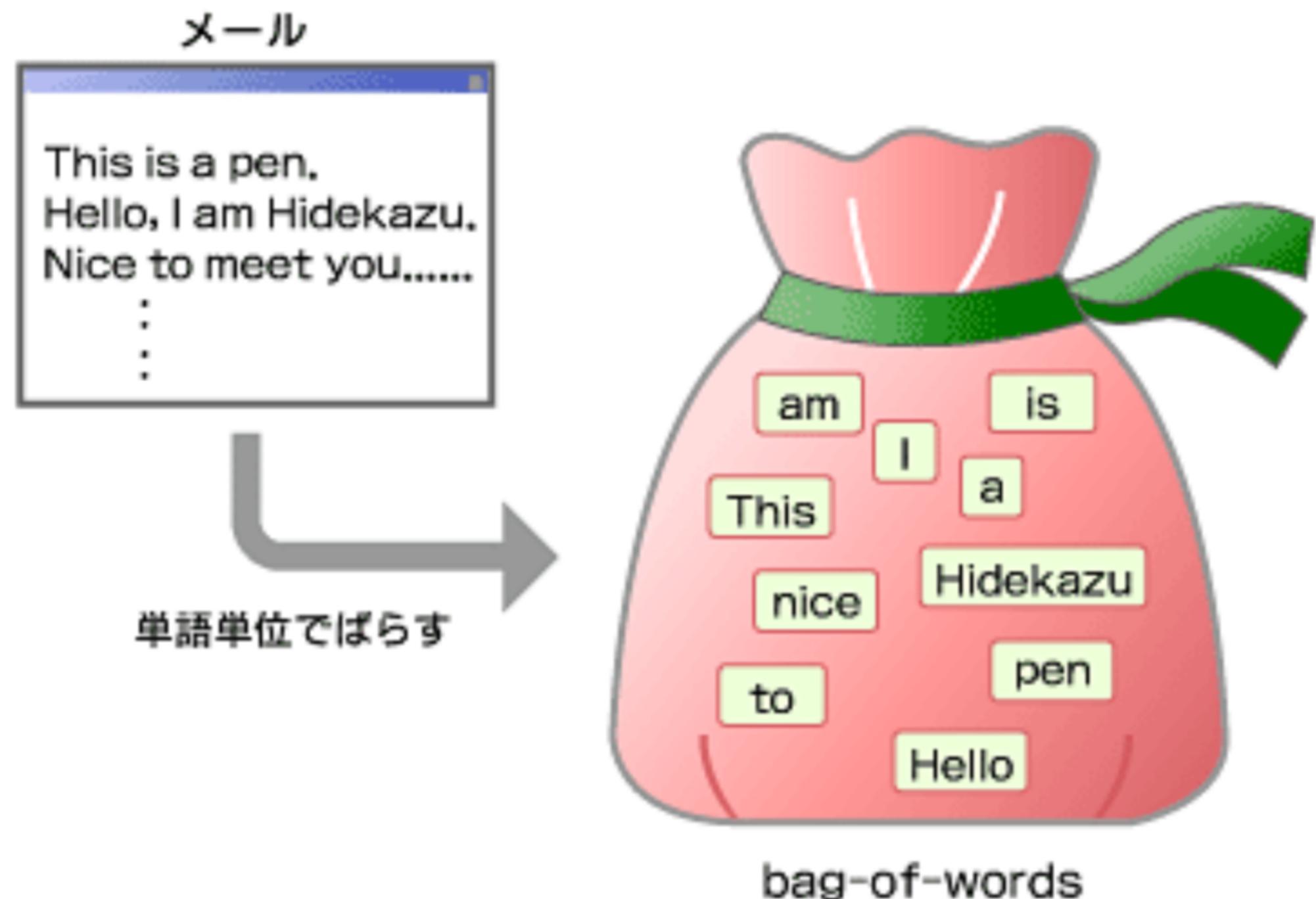
- Crear una nueva matriz  $\mathbf{M}'$  basada en la forma reducida de  $\Sigma$ .
- Ahora podremos:
  1. Realizar búsquedas como hicimos en Vector Space Model.
  2. Hacer clusters de vectores documento para descubrir tópicos.

# Latent Semantic Analysis



# Limitaciones de Bag of Words

- No capta bien frases o expresiones multi-palabra como: *a priori, San Juan, dulce de leche, etc.*
- No contempla bien posibles faltas de ortografía o derivaciones de palabras.



- **N-gramas:** dado un corpus, un n-grama es una secuencia consecutiva de **n** palabras de ese corpus.
- Pueden usarse tanto con letras como palabras o caracteres.
- Existen bi-gramas, tri-gramas, cuatri-gramas, ...
- Nuestra BOW crece considerablemente.

- the cat that sat on **the sofa** also sat on the mat.
- the cat that sat on the **sofa also** sat on the mat.
- the cat that sat on the sofa **also sat** on the mat.

Cabe preguntarnos:

- Son todas las combinaciones relevantes?
- Son las más frecuentes mejores?

- **Collocations:** combinaciones recurrentes de palabras que co-ocurren con más frecuencia que por el azar, muchas veces sin tener un significado compuesto.

Ejemplos:

- Café fuerte.
- Buenas prácticas.
- Regla general.

A qué nos referimos cuando decimos “con más frecuencia que por el azar”? Queremos comparar que la probabilidad de que dos palabras co-ocurran corresponde efectivamente a un **suceso** y no a una **sorpresa**.

Usamos reglas de asociación para determinar si dos palabras co-ocurren por mera chance o de una manera significativa: Chi-cuadrado ( $\chi^2$ ), Log-likelihood ratio (LL), Pointwise mutual information (PMI).