Indexar

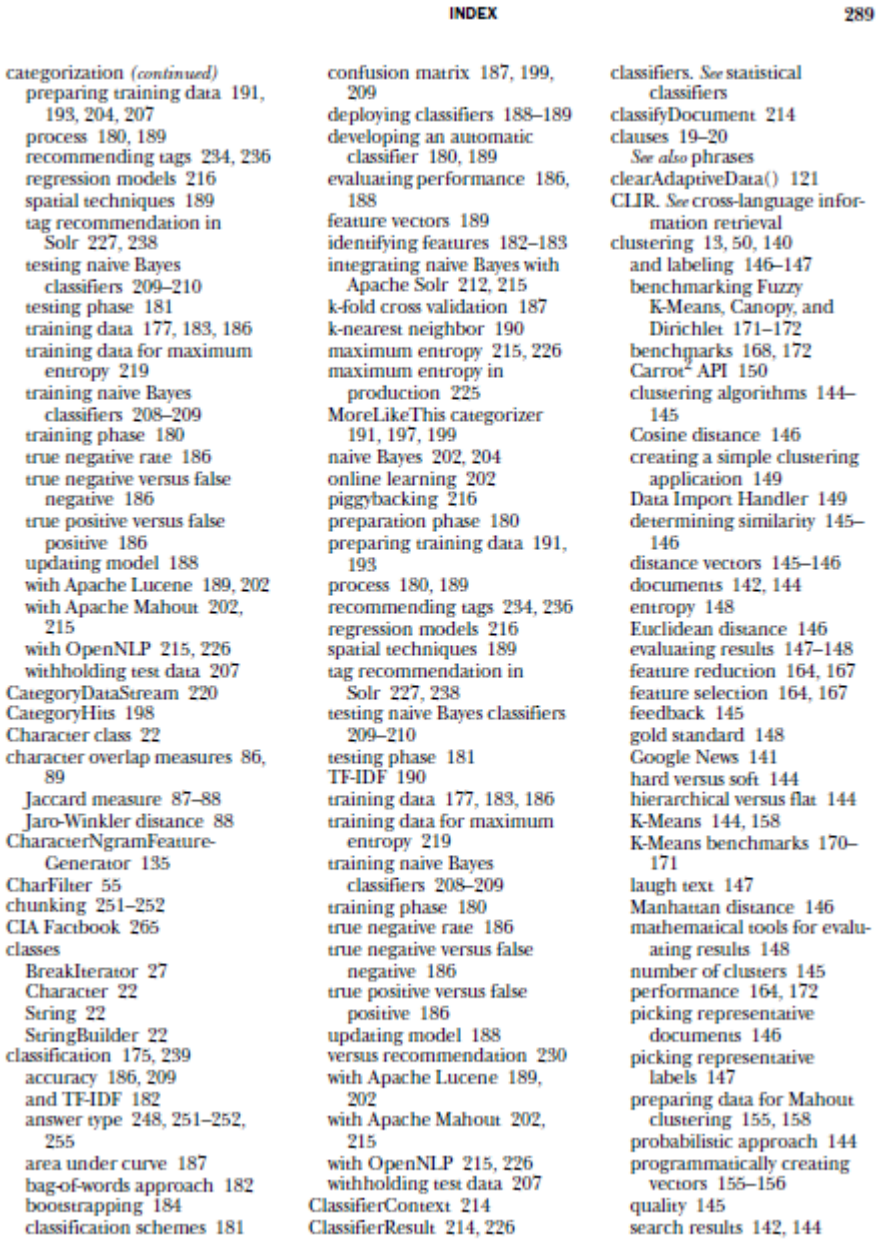
Procesar el Texto
de los Documentos

# Antes de indexar

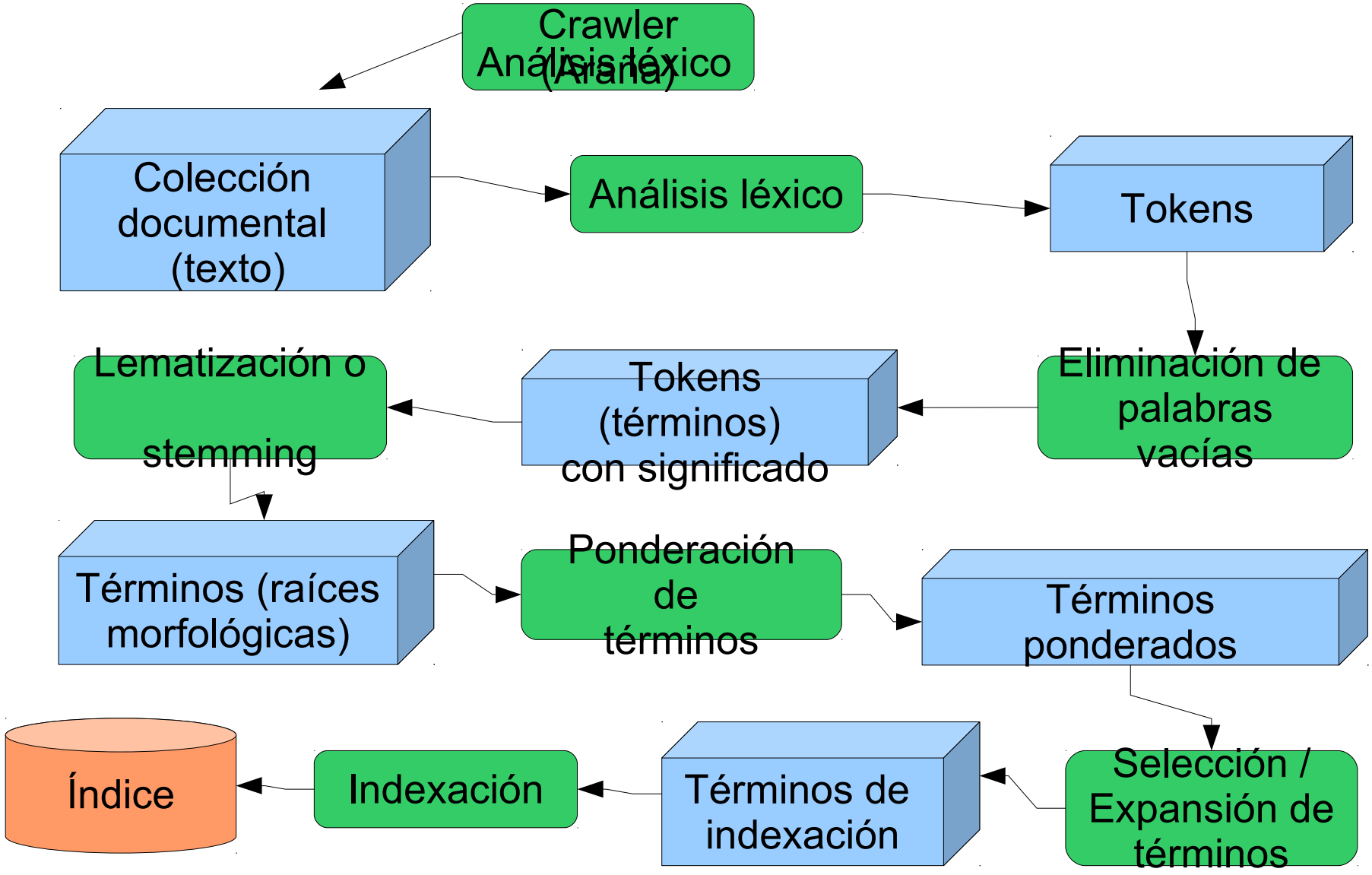
- La primera tarea, antes de crear el índice, es conocer cómo se va a buscar
- ¿ Qué tipo de información es importante sobre los documentos que vamos a considerar.?

# Indexar

- ¿ El título, fechas, frases, etc. ?
- ¿ Hay elementos en el documento que no se indexan (p.e. Etiquetas html)?



# El proceso general de la indexación



# Pasos del procesamiento documental

Partimos de una colección documental en local

1. Análisis Sintáctico
2. Análisis léxico (tokenizing).
3. Eliminación de palabras vacías (stop  words).
4. Segmentación (Stemming) o lematización.
5. Ponderación de términos (weighting).
6. Selección de los mejores términos.
7. Construcción del índice.

# 1 Analizador sintáctico (parser)

- Cual es el formato del documento?
  - pdf/word/excel/html?


En qué lenguaje está escrito?

- Que juego de caracteres utiliza?
  - (CP1252, UTF-8, ...)

•Cada una de estas cuestiones es un problema de clasificación

se suelen hacer de forma heurística, metadatos, mediante reglas … si hay muchos 'the' => Inglés

# 2. Análisis léxico (tokenization)

- Proceso por el cual el texto (secuencia de caracteres) queda separado en secuencias de tokens (términos).
- Los tokens son agrupaciones de caracteres con un significado colectivo.
  - *“En resolución, don Quijote se enfrascó tanto en su lectura, que se le ... claro, y los días de turbio en turbio; y así, del poco dormir y del mucho leer, se le secó el celebro, de ... juicio”.*

# 2. Análisis léxico

- Identificación de las tokens (secuencia de caracteres candidata a ser indexada, tras un preproceso)
- Espacios en blanco, Dígitos, Guiones, Signos de puntuación, Letras mayúsculas al comienzo, ...

*En resolución, don Quijote se enfrascó tanto en su lectura, que se le ... claro, y los días de turbio en turbio; y así, del poco dormir y del mucho leer, se le secó el celebro, de ... juicio".*

# 2. Análisis léxico

- ¿Qué hacemos con...?
- Números:
  - Normalmente ignorarlos.
  - Pero... podemos encontrarnos con fechas (2006, 450 A.C.,...) o con otros que representen informaciones relevantes al dominio.
  - Mezclados con letras pueden tener sentido (por ejemplo, CC123, B52, H2O).
- Mayúsculas y minúsculas:
  - Se pasan a minúsculas.
    - Zapatero == zapatero

# 2. Análisis léxico

- ¿Qué hacemos con...?
- Guiones:
  - Tratados como separadores:
    - *State-of-art == state of art*.
  - Se ignoran:
    - *on-line → online*.
  - Mantenerlos: Algoritmo de *Knuth-Morris-Pratt.*
- Signos de puntuación:
  - Normalmente los ignoramos.
    - Microsoft.Net -> Microsoft Net

# 2. Análisis léxico

- Problema:
- Si obtenemos 2 tokens, (dividimos los guiones) entonces las consultas con 1 sólo termino emparejarán
  - “San Francisco” -> q = {francisco} OK
- Si consideramos un sólo token, entonces las consultas relativas a un termino no emparejan
  - “El quijote” -> q = quijote NO

# 2. Análisis léxico

*En resolución, don Quijote se enfrascó tanto en su lectura, que se le ... claro, y los días de turbio en turbio; y así, del poco dormir y del mucho leer, se le secó el celebro, de ... juicio".*

*en resolución don quijote se enfrascó tanto en su lectura que se le ... claro y los días de turbio en turbio y así del poco dormir y del mucho leer se le secó el celebro de ... juicio*

# 3. Eliminación de palabras vacías

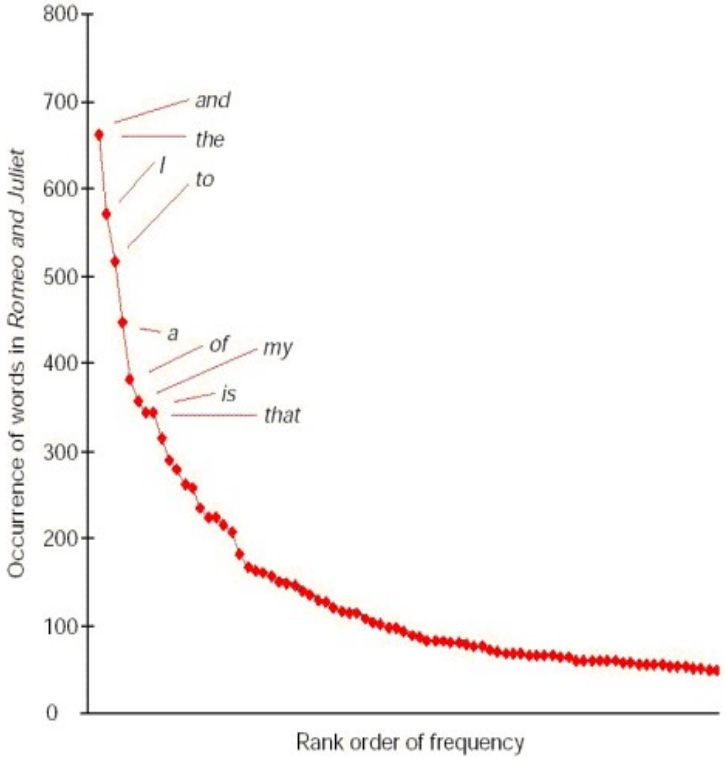
- Eliminar del diccionario aquellas palabras más comunes
  - Tienen poca semántica (artículos, determinantes, pronombres, preposiciones,...)
  - Son palabras muy frecuentes (más del 80%) y poco útiles para la recuperación:
    - ahorramos espacio y ganamos eficacia recuperadora.
- La tendencia hoy día es no hacerlo.
  - Se indexan y se determina su uso en el proceso de búsqueda

# Ley de Zipf

- Analiza la distribución de los términos en un lenguaje, colección o documento.
  - distribución altamente sesgada
  - Un conjunto pequeño de palabras son muy frecuentes, y otras muchas raramente ocurren
  - p.e. Quijote:
    - Las 5 palabras más frecuentes en el Quijote son: “que,de, y, la, a”, representando el 20% del total.
    - Un 10% de ellas ocurren 1 o 2 veces.

# Ley de Zipf:

- Idea: contar
  - Cuantas veces un término ocurre en el texto
    - Sobre todos los docs de la colección
- Ordenar (ranking) éstos según frecuencia de aparición

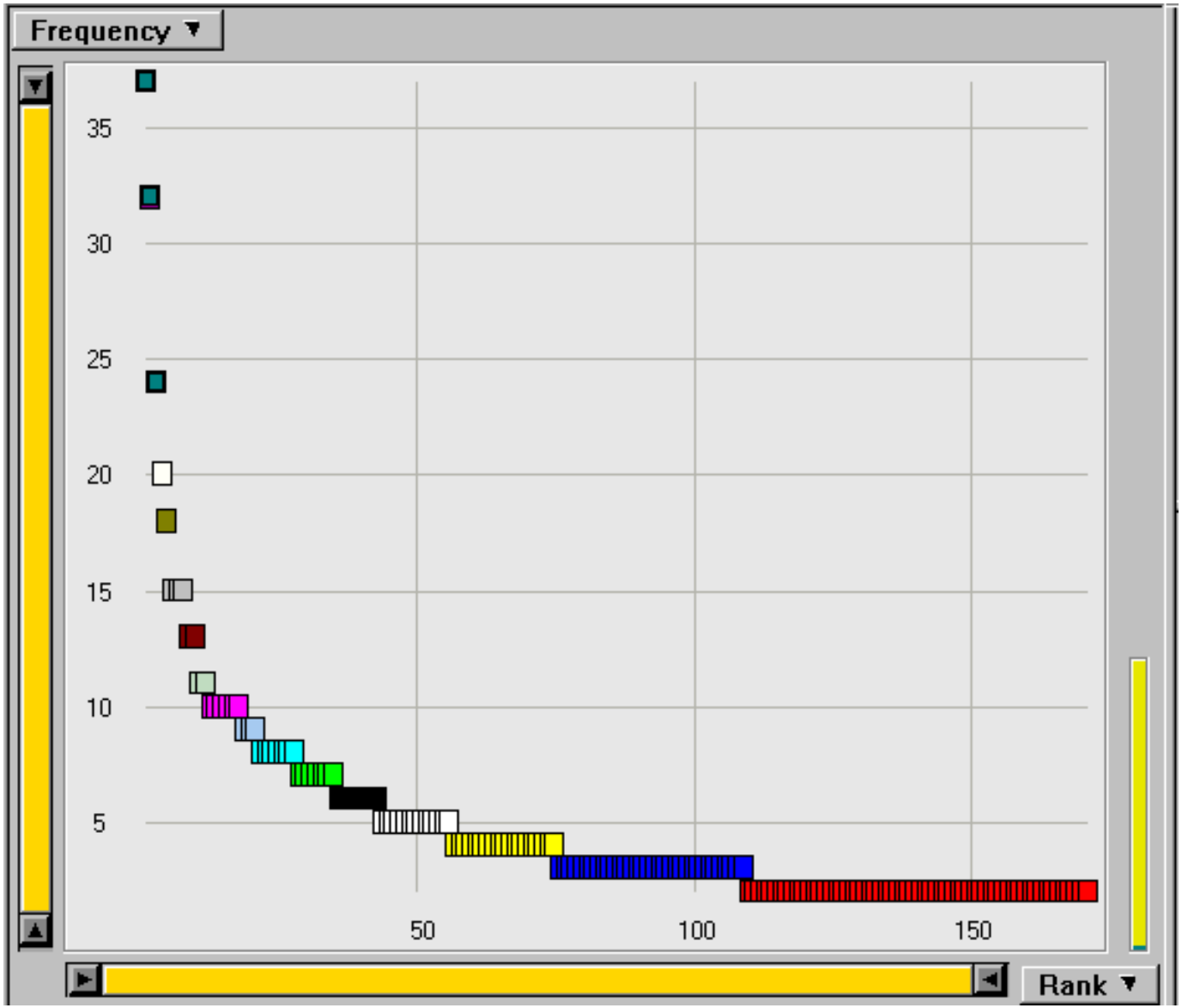


# Ejemplo: Términos más y menos frecuentes

| Rank | Freq | Term |
|---|---|---|
| 1 | 37 | system |
| 2 | 32 | knowledg |
| 3 | 24 | base |
| 4 | 20 | problem |
| 5 | 18 | abstract |
| 6 | 15 | model |
| 7 | 15 | languag |
| 8 | 15 | implem |
| 9 | 13 | reason |
| 10 | 13 | inform |
| 11 | 11 | expert |
| 12 | 11 | analysi |
| 13 | 10 | rule |
| 14 | 10 | program |
| 15 | 10 | oper |
| 16 | 10 | evalu |
| 17 | 10 | comput |
| 18 | 10 | case |
| 19 | 9 | gener |
| 20 | 9 | form |
| 150 | 2 | enhanc |
| 151 | 2 | energi |
| 152 | 2 | emphasi |
| 153 | 2 | detect |
| 154 | 2 | desir |
| 155 | 2 | date |
| 156 | 2 | critic |
| 157 | 2 | content |
| 158 | 2 | consider |
| 159 | 2 | concern |
| 160 | 2 | compon |
| 161 | 2 | compar |
| 162 | 2 | commerci |
| 163 | 2 | clause |
| 164 | 2 | aspect |
| 165 | 2 | area |
| 166 | 2 | aim |
| 167 | 2 | affect |

# Curva

| Rank | Freq | |
|---|---|---|
| 1 | 37 | system |
| 2 | 32 | knowledg |
| 3 | 24 | base |
| 4 | 20 | problem |
| 5 | 18 | abstract |
| 6 | 15 | model |
| 7 | 15 | languag |
| 8 | 15 | implem |
| 9 | 13 | reason |
| 10 | 13 | inform |
| 11 | 11 | expert |
| 12 | 11 | analysi |
| 13 | 10 | rule |
| 14 | 10 | program |
| 15 | 10 | oper |
| 16 | 10 | evalu |
| 17 | 10 | comput |
| 18 | 10 | case |
| 19 | 9 | gener |
| 20 | 9 | form |



# Distribución de Zipf

- Los aspectos importantes:
  - Unos pocos elementos *son muy frecuentes*
  - Un número medio de elementos tiene una frecuencia intermedia
  - Muchos elementos ocurren *muy rara vez*
  - Long tail (tamaño ciudades, películas vistas, etc.)

# Distribución de Zipf

- El producto de la frecuencia de las palabras (f) por su posición en el ranking (r) es aproximadamente constante

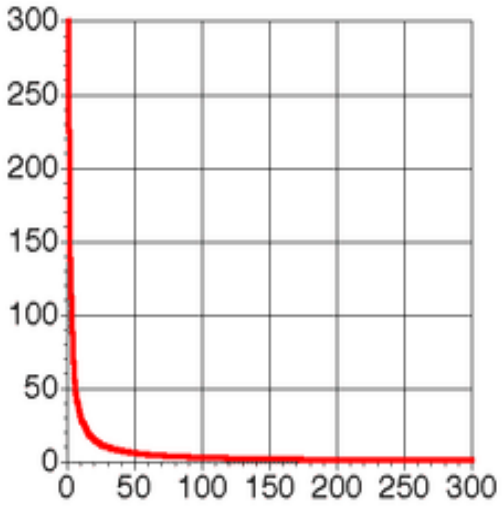
$$f*r=K \quad , \quad K \simeq N/10$$

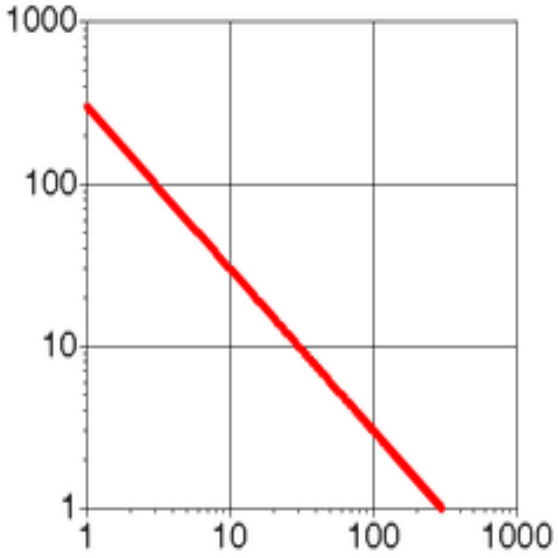
- Otra forma de verlo es:
  - El término más común ocurre K veces
  - El segundo término ocurre K/2 veces
  - El tercer término ocurre K/3 veces
  - ...

# Ley de Zipf

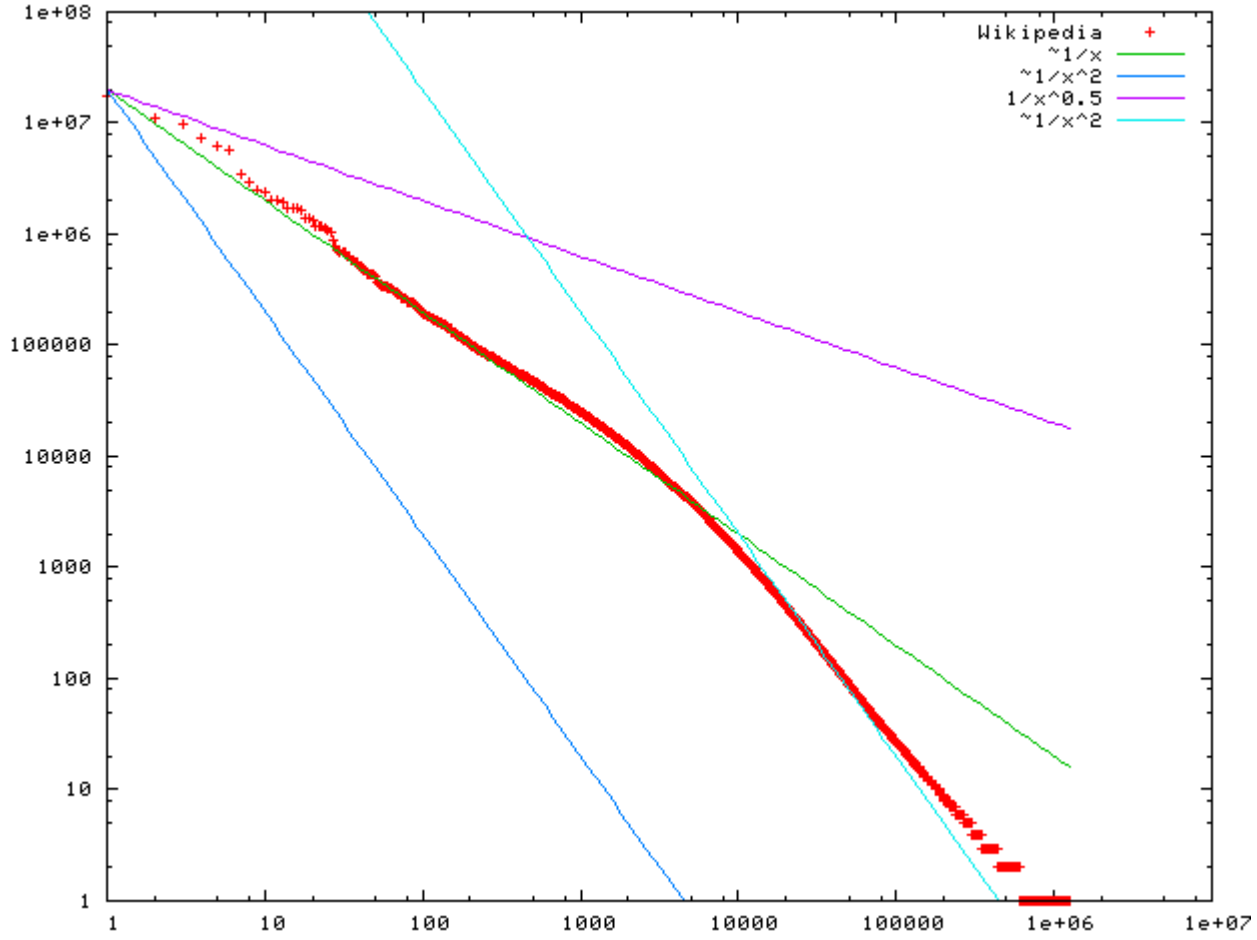
| Palabra | Frecuenc. | Posición | f * r |
|---|---|---|---|
| the | 3332 | 1 | 3332 |
| and | 2972 | 2 | 5944 |
| a | 1775 | 3 | 5235 |
| he | 877 | 10 | 8770 |
| be | 294 | 30 | 8820 |
| there | 222 | 40 | 8880 |
| one | 172 | 50 | 8600 |
| friends | 10 | 800 | 8000 |
| | | | |

# Distribución de Zipf (lineal y en escala log-log)



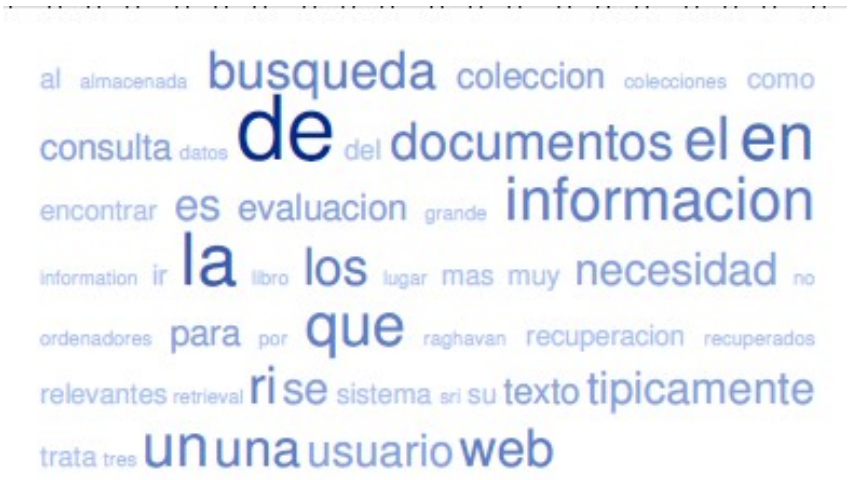
A plot of word frequency in Wikipedia (November 27, 2006). The plot is in log-log coordinates. x is rank of a word in the frequency table; y is the total number of the word's occurrences. Most popular words are "the", "of" and "and", as expected. Zipf's law corresponds to the middle linear portion of the curve, roughly following the green (1/x) line, while the early part is closer to the magenta (1/"x^0.5") line while the later part is closer to the cyan (1/"(k+x)^2.0") line. These lines correspond to three distinct parameterizations of the Zipf-Mandelbrot distribution.

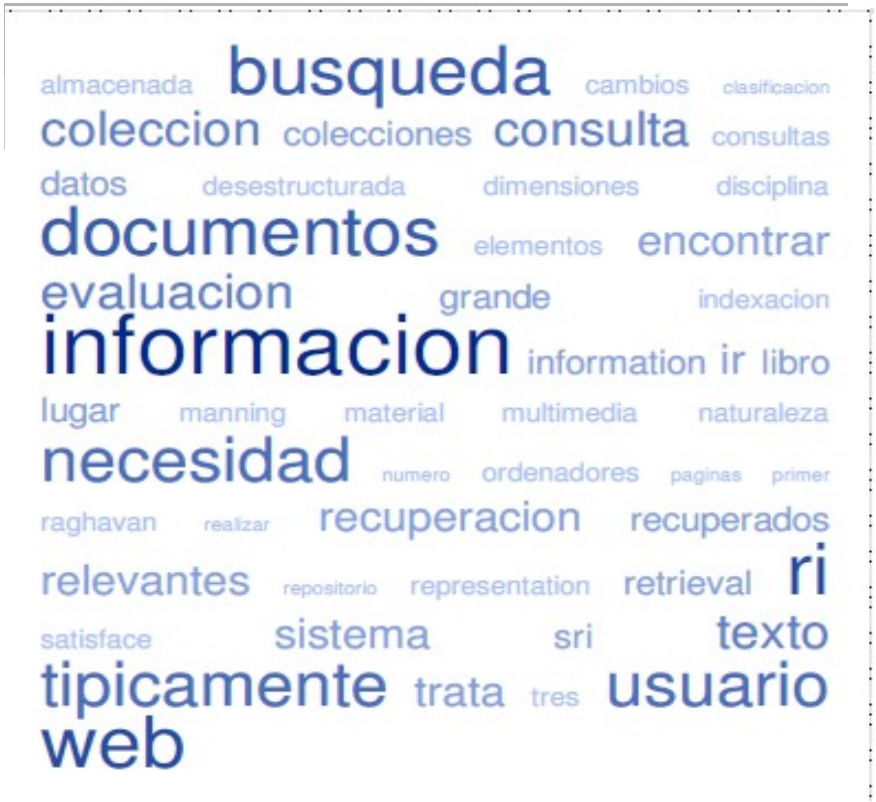


# Ley de Zipf => Aplicación a IR.

- Dado que la distribución de las palabras está muy sesgada,
  - Palabras que son demasiado frecuentes, palabras vacías, se pueden descartar
  - Una palabra vacía no tiene significado en lenguaje natural
    - Ejemplos de palabras vacías
      - En Inglés: a, the, by
      - En castellano: una, el, por, y
  - Afortunadamente, son las palabras más frecuentes
    - Por lo tanto, descartamos la mitad de las palabras que aparecen en un texto
    - Tamaño de los índices se reducen un 40%

# Nube de Palabras
# Sin / Con stop words



# 3. Eliminación de palabras vacías

- Dependen del idioma.
- Y del dominio:
  - Podemos incluir palabras muy comunes que aportan poco en colecciones concretas: “ordenador” en una colección de documentos sobre informática).

# 3. Eliminación de palabras vacías

**En español:**

*un, una, unas, unos, uno, sobre, todo, también, tras, otro, algún, alguno, alguna, algunos, algunas, ser, es, soy, eres, somos, sois, estoy, esta, estamos, estáis, están, como, en, para, atrás, porque, por qué, estado, estaba, ante, antes, siendo, ambos, pero, por, poder puede, puedo, podemos, podéis, pueden fui, fue, fuimos, fueron, ...*

**En inglés:**

*a, about, above, across, after, again, against, all, almost, alone, along, already, also, although, always, among, an, and, another, any, anybody, anyone, anything, anywhere, are, area, areas, around, as, ask, asked, asking, asks, at, away, b, back, backed, backing, backs, be, because, became, ...*

# 3. Eliminación de palabras vacías

- Consulta: “¿Ser o no ser?” , “the who” :-)
- Tendencia:
  - Lista de palabras vacía pequeña para colecciones generales o usuarios poco experimentados.
  - Más completa para dominios muy definidos y los usuarios están entrenados en el uso de esas colecciones.

# 3. Eliminación de palabras vacías

*En resolución, don Quijote se enfrascó tanto en su lectura, que se le ... claro, y los días de turbio en turbio; y así, del poco dormir y del mucho leer, se le secó el celebro, de ... juicio".*

*resolución don quijote enfrascó tanto*
*lectura ... claro días turbio*
*turbio poco dormir mucho*
*leer secó celebro ... juicio*

# 4. Segmentación y lematización

Métodos usados para reducir el tamaño del vocabulario: La ráiz de la palabra contiene el significado del término, los prefijos y sufijos son modificadores sintácticos

En lugar de indexar todas las palabras, se buscan sus “representantes” morfológicos, raíces o lemas.

biblioteca, bibliotecas, bibliotecario, bibliotecarios, bibliotecaria, bibliotecarias, Biblioteconomía y Bibliotecología.

Se aglutinan en biblioteca

# 4. Segmentación y lematización

- Ventajas:
  - Reducción del vocabulario → eficiencia y ahorro de espacio.
  - Aumenta el número de documentos recuperados.
- Desventajas:
  - Se pierde información sobre la palabra completa.
  - Es crítico para la calidad de la recuperación, tenemos que tener un compromiso entre recall/precision

poco/no segmentación — segmentación fuerte

-recall +precision ⟷ +recall -precision

# 4.1. Segmentación (Stemming)

- Segmentación: Usa reglas heurísticas que suelen implicar eliminar afijos
  Ejemplos en inglés:

{*connect*ed, *connect*ing *connect*ion,..}→ ***connect***

{*comput*er, *comput*ational, *comput*ation} →***comput***

for example compressed
and compression are
both
accepted as equivalent to
compress.

compres
exampl compres
accept equival compres.

# 4.1. Segmentación (Stemming)

- El análisis morfológico necesario para extraer la raíz es dependiente del idioma y puede llegar a ser muy complejo (como en el español).
- Lo habitual es que los métodos automáticos de segmentación eliminen los afijos de manera iterativa y “ciega”.

# 4.1. Segmentación (Stemming)

- Dos tipos básicos:
  - Basados en diccionarios: usa una lista de palabras relacionadas
  - Basados en algoritmos: Usan un algoritmo para determinar las palabras relacionadas
- Algoritmos
  - sufijos: eliminar la ‘s’ final del plural
    - gatos→ gato  ... cats → cat,
      - Falso negativos:  supplies → supplie
      - Falso positivo: ups → up

# 4.1.Stemming por Eliminación de afijos

Algoritmos comunes en inglés:

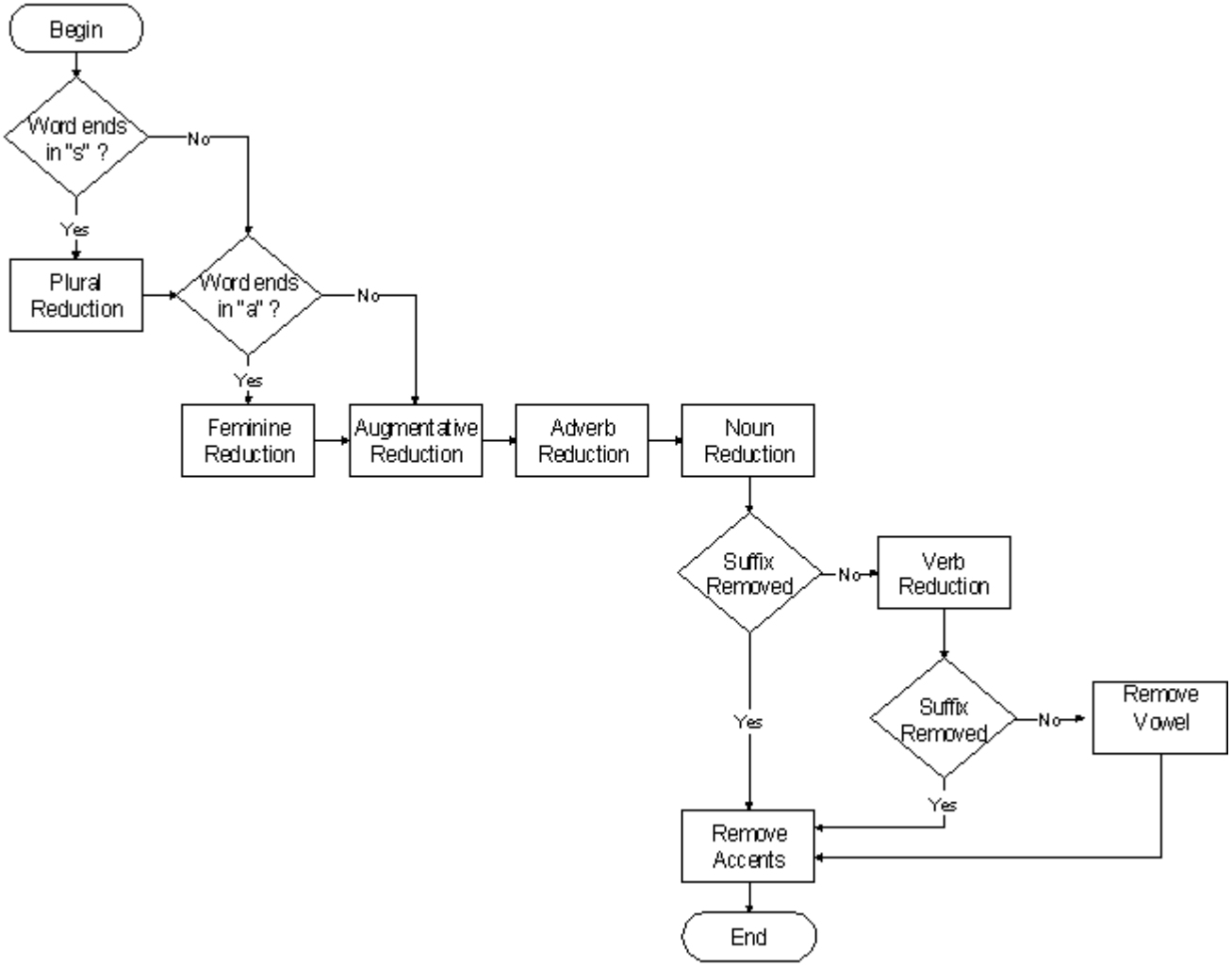
Lovins, Slaton, Dawson,

**Porter, Krovetz, snowball**

# 4.1.1. Eliminación de afijos

- Algoritmo de Porter:
- Originalmente diseñado para inglés.
- Utiliza una lista de sufijos y prefijos para eliminarlos.
- Aplica una serie de reglas a los afijos de las palabras.
- Puede dar lugar a raíces que no son reconocidas como raíces en el idioma.
- Ambigüedad en ciertos casos y errores en otros.

# 4.1.1. Eliminación de afijos



# Porter algorithm

(Porter, M.F., 1980, An algorithm for suffix stripping, *Program*, **14**(3) :130-137)

- Step 1: plurals and past participles
  - SSES -> SS  caresses -> caress
  - (*v*) ING ->  motoring -> motor
- Step 2: adj->n,  n->v,  n->adj, ...
  - (m>0) OUSNESS -> OUS callousness -> callous
  - (m>0) ATIONAL -> ATE  relational -> relate
- Step 3:
  - (m>0) ICATE -> IC  triplicate -> triplic
- Step 4:
  - (m>1) AL ->  revival -> reviv
  - (m>1) ANCE ->  allowance -> allow
- Step 5:
  - (m>1) E ->  probate -> probat
  - (m > 1 and *d and *L) -> single letter  controll -> control

# Porter algorithm

## (En castellano: http://snowball.tartarus.org/algorithms/spanish/stemmer.html)

- Step 0: Pronombres
  - SELOS, LAS, .. después de [iendo,ando,ar,er,ir] →”” por ejemplo, haciéndolas ->haciendo
- Step 1 : Eleminación de sufijos
  - (m>0) ismo, able -> “” considerable -> considera
  - (m>0) logía,logías -> log tecnología -> tecnolog
- Step 3: Elimanación de sufijos de verbos
  - (m>0) ería, arais, -> “” quería → quer

Etc. …

# Ejemplo:

- En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

- en un lug de la mancha, de cuy nombr no quier acordarme, no ha much tiemp que viv un hidalg de los de lanz en astillero, adarg antigua, rocin flac y galg corredor.

***Sample text:*** Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

***Lovins stemmer:*** such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

***Porter stemmer:*** such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

***Paice stemmer:*** such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

**Figure 2.8:** A comparison of three stemming algorithms on a sample text.

# 4.2. Lematización

- Transformación de la palabra a su forma base, lema al que pertenece.

  Es más precisa que la segmentación:
  - Utilizan un vocabulario controlado para determinar el lema
  - Realizan un análisis morfológico de los términos

  Necesitan de mas recursos
  Disciplinas de NLP (proc. leng. nat) y Lingüística computacional

# 4.2. Lematización

- Por ejemplo:

Formas verbales a infinitivo:

{Fui, van, iremos} → ir

Femeninos a masculinos:

{gata} → gato

Casa, casero, casucha, casona -> casa.

*the boy's cars are different colors*

*the boy car be different color*

# Qué hemos conseguido

Como resultado cada documento se representa mediante una secuencia de keywords (términos de indexación)

*En resolución, don Quijote se enfrascó tanto en su lectura, que se le ... claro, y los días de turbio en turbio; y así, del poco dormir y del mucho leer, se le secó el celebro, de ... juicio".*

*Pasa a*

*resoluc don quijote enfrasc lectura claro dia turbio dormir leer sec celebro juicio*

# Qué buscamos

Asociarle a cada uno de los términos de indexación un peso que refleje la importancia del término en el documento

$$D_1 = \{(t_1, w_1), (t_2, w_2), \ldots\}$$

*resoluc 0.2, don 0.4, quijote 0.4, enfrasc 0.8, lectura 0.4, claro 0.7, dia 0.3 .....*

# 5. Ponderación de términos

- Asignación de un peso a cada término que nos indique la importancia del mismo.
- Capacidad de discriminación de documentos relevantes y no relevantes.
- Esos pesos serán utilizados por los modelos de recuperación para obtener la salida.

# 5. Ponderación de términos

- Esquemas de ponderación:
  - Dependen del modelo de recuperación.
    - Ejemplos: espacio vectorial y probabilístico.
  - Muchas variantes.
  - Su elección dependen de la evaluación de la recuperación.

# 6. Selección de términos

- En algunos casos, no es suficiente filtrar las palabras vacías de significado y aplicar segmentación / lematización.
- Se necesita reducir más el vocabulario, e identificar los mejores términos para la recuperación.
- Se aplica un proceso de *Selección de términos*.

# 6. Selección de términos

- Métodos basados en la frecuencia de aparición:
  - Identificar los términos con una frecuencia de aparición media (basándonos en las ideas de Luhn).
  - Identificar los límites inferior y superior.

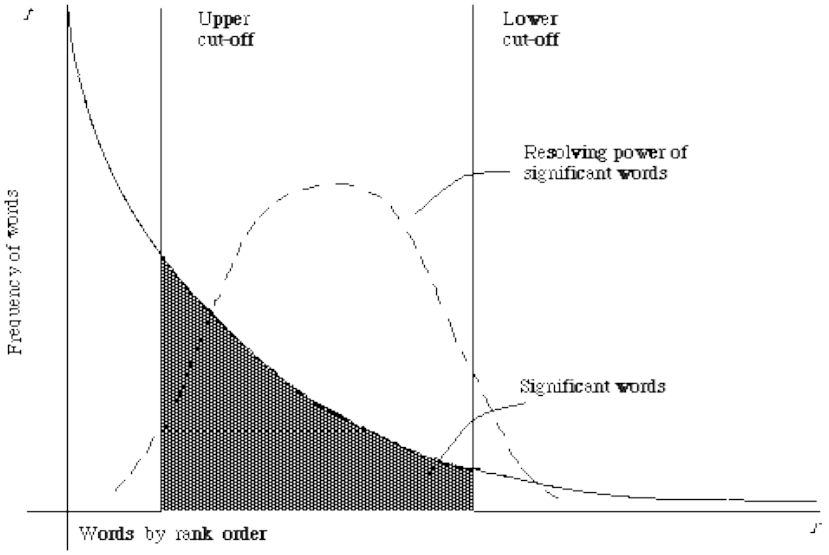



*Figure 2.1. A plot of the hyperbolic curve relating f, the frequency of occurrence and r, the rank order (Adapted from Schultz[14] page 120)*

# 6. Selección de términos

- Método del valor de discriminación del término (Salton):
  - Mide el grado con el que un término es capaz de discriminar documentos de la colección.
  - Cuanto mayor capacidad de discriminación, mayor la calidad del término como término de indexación.

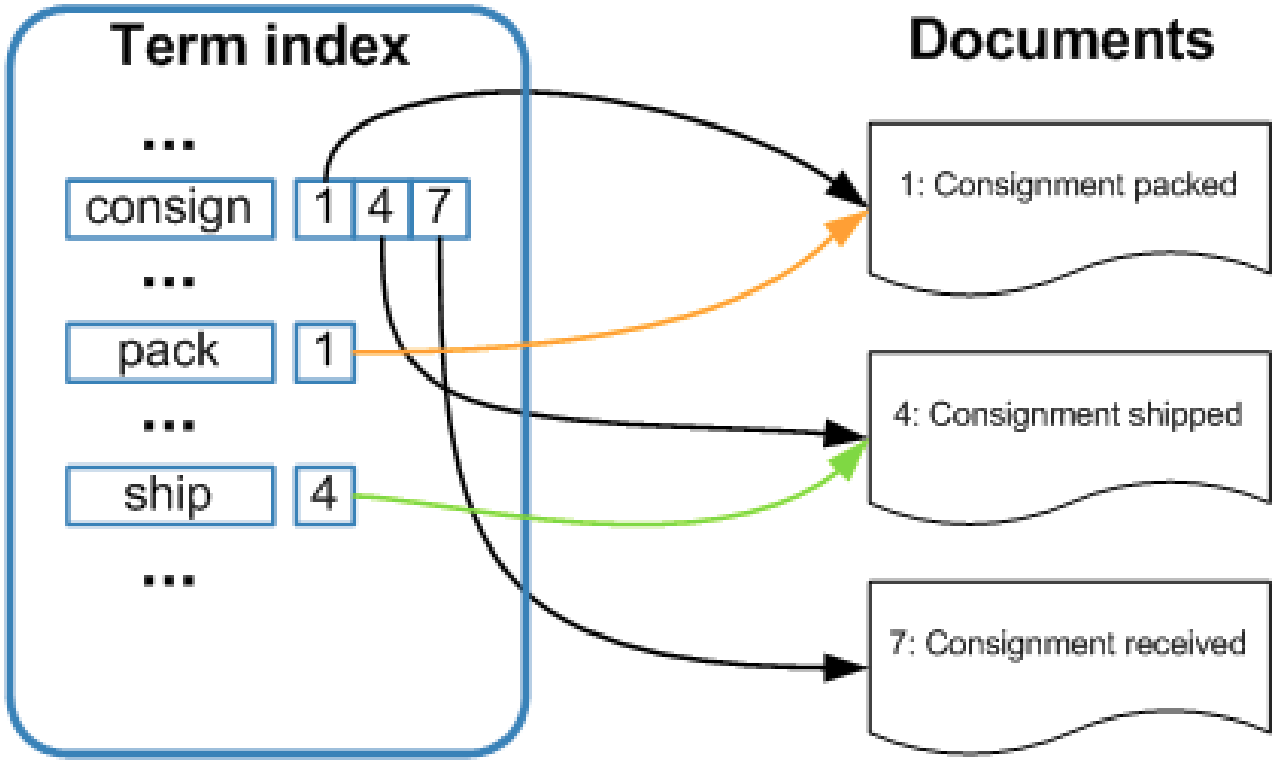
# 6. Selección de términos

- Cálculo del valor de discriminación (DV):
  1) Calcular la similitud media entre documentos (según espacio vectorial, p.e.)
  2) Para cada término, $t_k$, de la colección:
     1) Eliminarlo temporalmente de la colección.
     2) Calcular de nuevo la similitud media entre documentos.
     3) **$DV(t_k)$ = (similitud media sin $t_k$) - (similitud media con $t_k$).**
     4) Seleccionar aquellos con DV > 0.

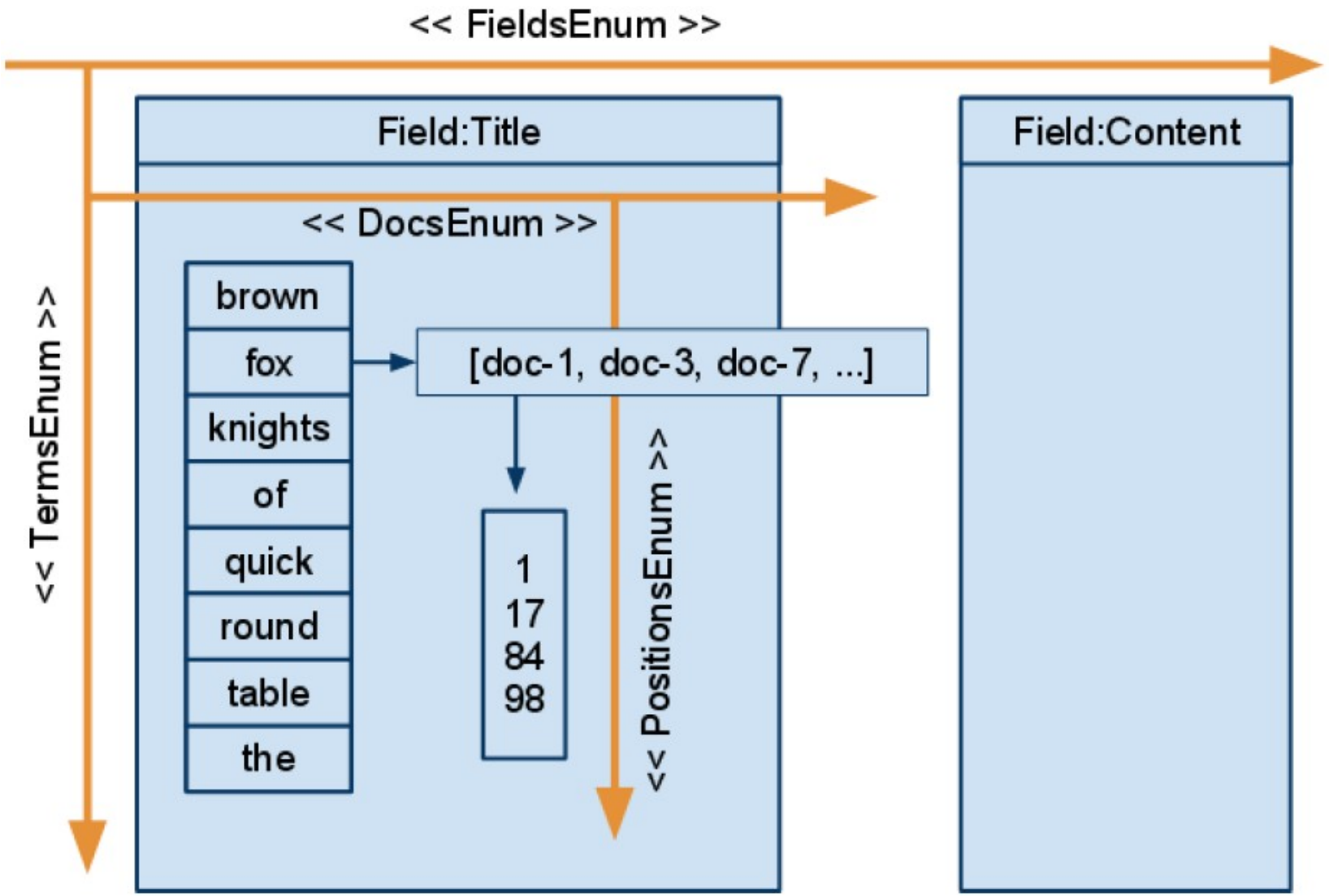
# Resumen: Manipular Texto

- Realizar el análisis léxico - procesamiento de textos en tokens
  - Muchos  problemas: la normalización, lematización
- Stemming reduce el número de elementos
  - Algortimo Porter más común
- Eliminar palabras vacías mejora el rendimiento
  - Lo que queda son los términos que se indexan
- El texto sigue una distribución power-low
  - La palabras con más poder están en el medio y en la cola de la distribución

# 7. Crear el índice



# 7. Índice Lucene



# Papel del índice en las búsquedas (AND)

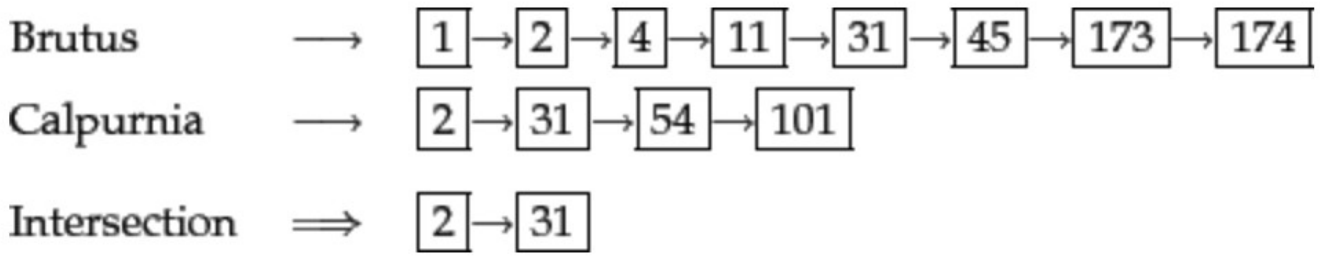



**Figure:** Intersecting the postings lists for Brutus and Calpurnia from Figure 1.3 .

```
INTERSECT(p1, p2)
 1  answer ← ⟨ ⟩
 2  while p1 ≠ NIL and p2 ≠ NIL
 3  do if docID(p1) = docID(p2)
 4        then ADD(answer, docID(p1))
 5             p1 ← next(p1)
 6             p2 ← next(p2)
 7        else if docID(p1) < docID(p2)
 8                then p1 ← next(p1)
 9                else p2 ← next(p2)
10  return answer
```

**Figure 1.6:** Algorithm for the intersection of two postings lists $p_1$ and $p_2$.