webir

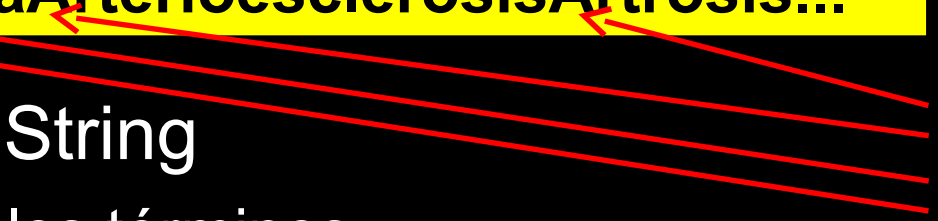
Clase 7

Temas

- Compresión de índices
  - Vocabulario
  - Lista de postings
- Relevancia
- Modelo de espacio vectorial

Compresión de Vocabulario

... ArbolArteriaArterioesclerosisArtrosis...



- Diccionario en un sólo String
  - Lista continua de todos los términos
  - Reducción de hasta 60%
- Por Bloques

... 5Arbol7Arteria17Arterioesclerosis8Artrosis...

... 9Elemental8Elemento8Eliminar...

...

... 5Zorro6Zorzal8Zozobrar...

- Codificación Frontal – Front Coding
  - 11.2 → 5.9

... 8Automata8Automate9automatic10Automation...

... 8Automat*a1¤e2¤ic3¤ion...

Compresión de la Lista de Postings

- A veces es más “económico” representar el espacio entre documentos < 20 bits
- Representación de largo variable - compresión
  - Por bytes
    - El primer bit de cada byte indica “continuación”
      - 1 si es el último byte y 0 en caso contrario
    - Se concatenan 7 bits de cada byte
  - Por bits
    - Códigos γ (y δ) = *largo* y *offset*
    - *offset* es el número en representación binaria sin el 1 más significativo
    - *largo* es el largo del *offset* en código unario

Código Variable en bits – Código γ

- Ventajas
  - Universalidad
  - Sin prefijos comunes – sin delimitadores
  - Sin parámetros que ajustar
- Desventajas
  - Decodificación costosa
  - No siempre coinciden las unidades con palabras/unidades en memoria
  - Las operaciones de bajo nivel se aplican a las unidades de memoria

0	
1	0
2	10,0
3	10,1
4	110,00
9	1110,001
13	1110,101
24	11110,1000
511	111111110,11111111

Relevancia

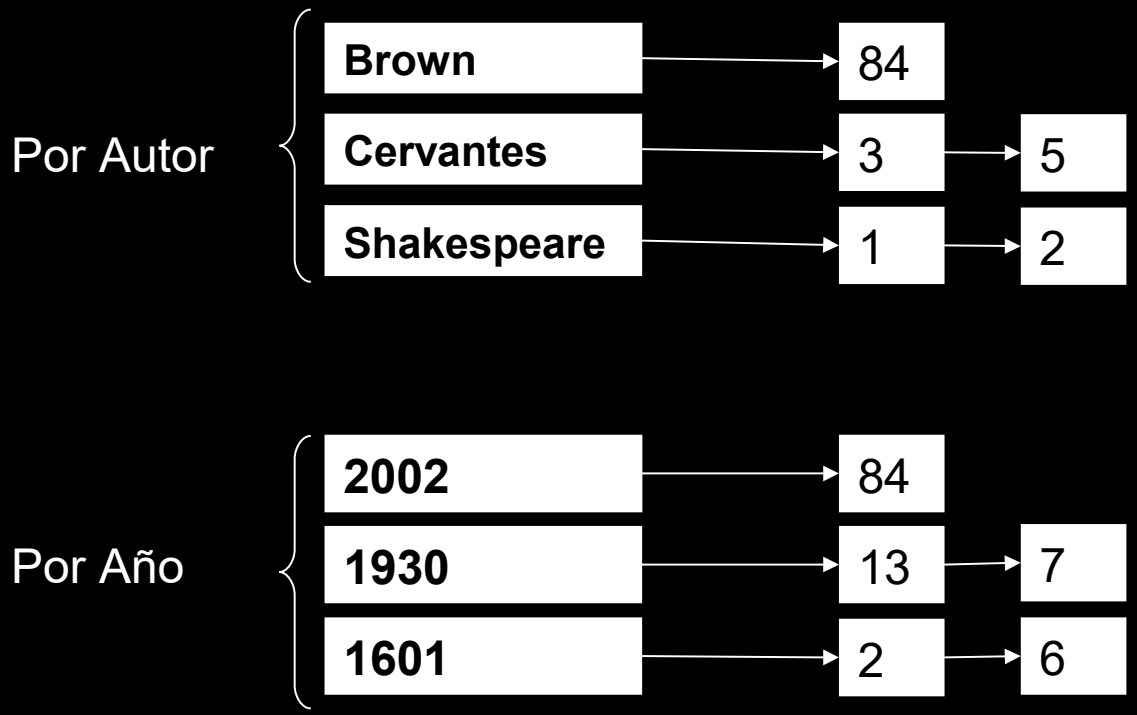
- Cuando los documentos que contienen las palabras son demasiados
- Ordenar los resultados de acuerdo a su relevancia - ranking
- Calcular un valor llamado puntuación - document score
- Puntuación asociada a parejas consulta-documento

Relevancia

- Índices paramétricos por zonas - metadatos
- Asignar relevancia a los términos basada en estadísticas como las frecuencias de aparición
- Modelo de espacio vectorial para representar consultas y documentos

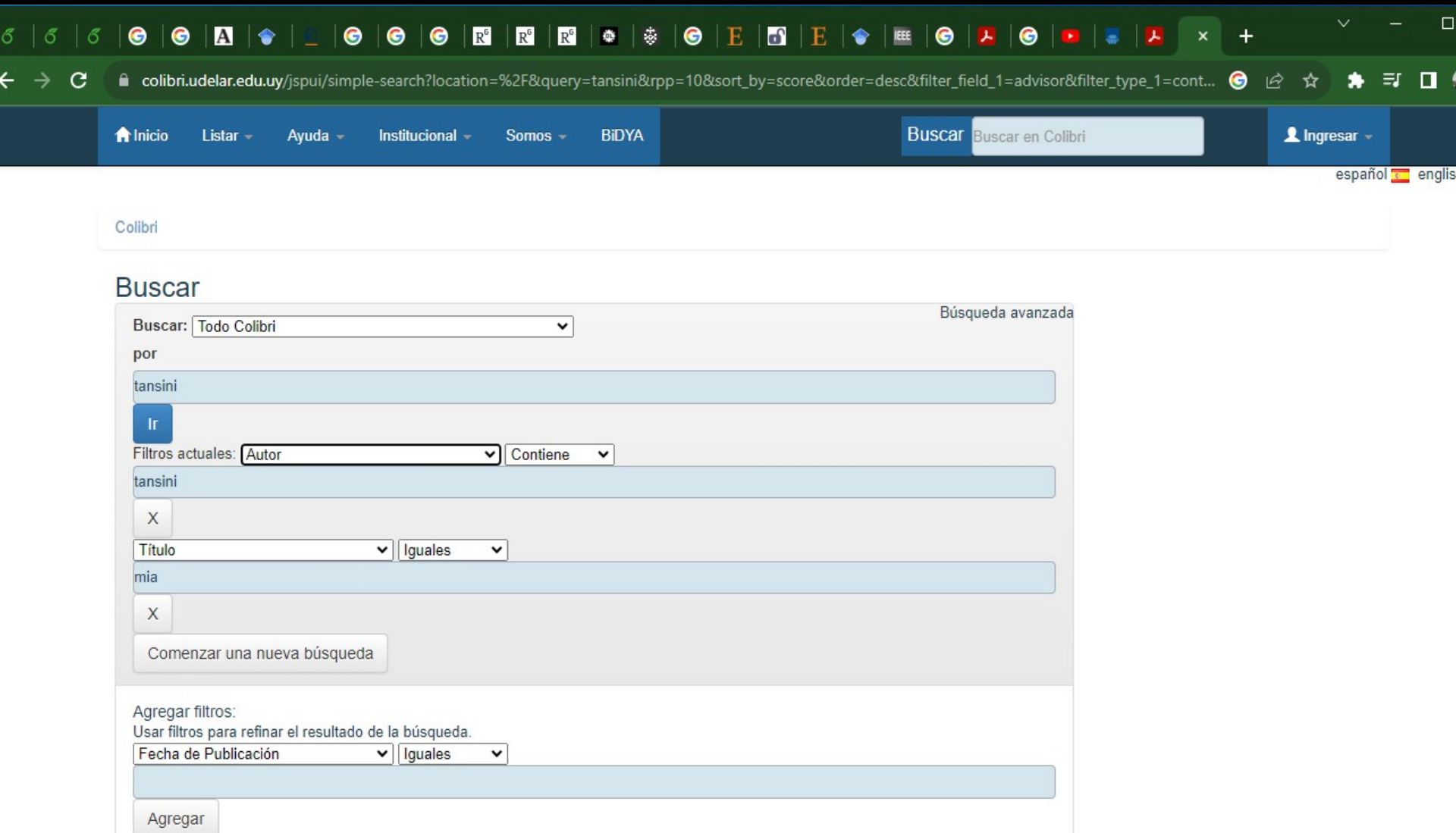
Relevancia

- Metadatos
- Campos
  - Autores
  - Título
  - Fecha
  - Idioma
- Indices paramétricos
  - Por campos
- Árboles B
  - rangos de fechas por ej



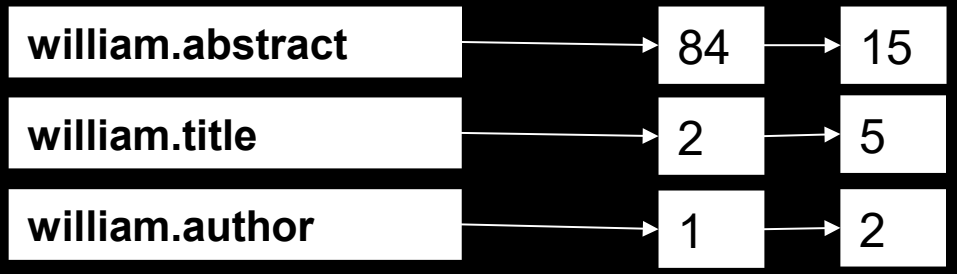
- Obras de Shakespeare de 1601

Relevancia – Búsquedas Paramétricas



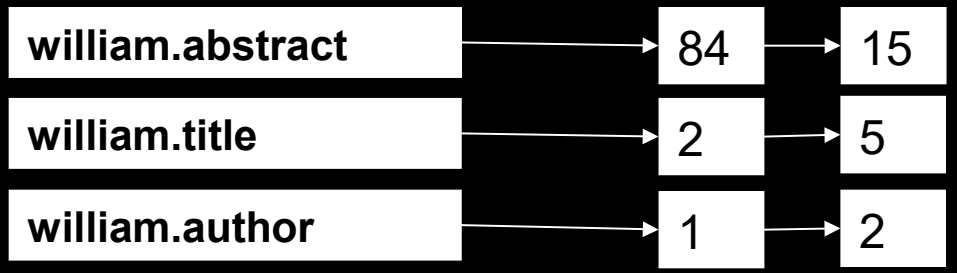
Relevancia

- Zonas
- Campos con texto cualquiera
  - Título
  - Abstract
- Indices paramétricos
  - Codificadas en el vocabulario
- Crece mucho el vocabulario
  - Para tener en memoria
  - ¿Alternativas?



Relevancia

- Zonas
- Campos con texto cualquiera
  - Título
  - Abstract



- Indices paramétricos
  - Codificadas en el vocabulario
  - Codificadas en los postings

- Reduce tamaño diccionario
- Puntuación por zonas

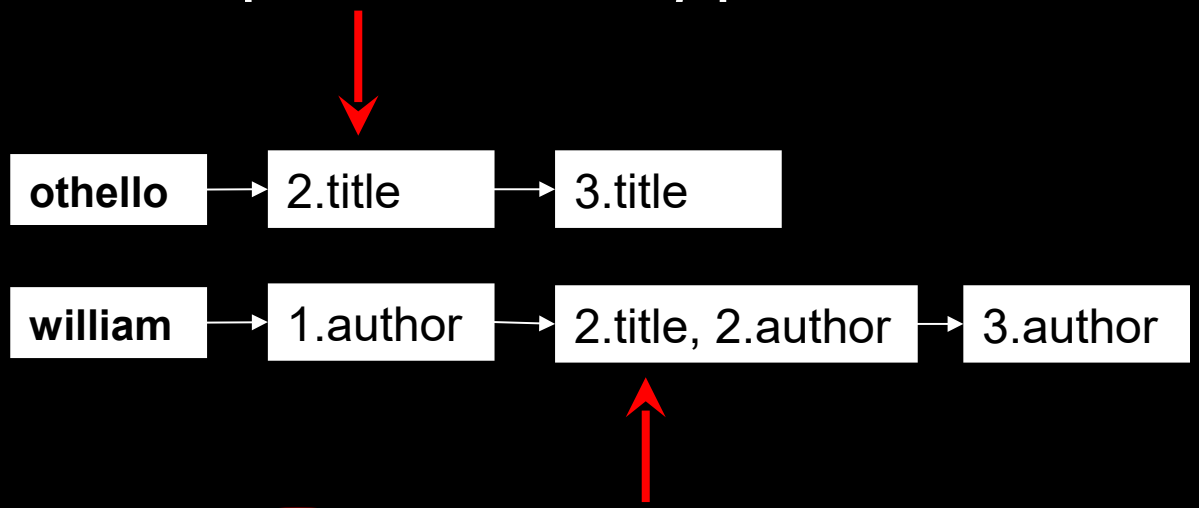
Relevancia – Puntuación por Zonas

- Consulta booleana
- Puntuación asociada a las parejas $(q, d) \in [0,1]$
- Combinación lineal de los puntajes de cada zona
  - l zonas
  - Coeficientes $g_1, g_2, \ldots, g_l \in [0,1]$, tales que $\Sigma_{i=1..l}\, g_i = 1$
  - $s_i\,(q,d) \in \{0,1\}$ representa la pertenencia de una/algunas/todas las palabras de q al documento d en la zona i (puntaje)

    $\rightarrow \Sigma_{i=1..l}\, g_i * s_i(q,d)$ (puntuación por zonas)
- Recuperación booleana ordenada (ranqueada)

Relevancia – Puntuación por Zonas

- Ejemplo
- Consulta $q = q_1$ *AND* q_2 = *othello AND william*
- $s_i(q,d) = 1$ si todas las palabras de q pertenecen a la zona i de d
- $i = 1, 2$



- $g_1 = g_2 = 0.5$

$\rightarrow \sum_{i=1..I} g_i * s_i(q,d2) = g_1 * s_1(q,d2) + g_2 * s_2(q,d2) = 0.5*1 + 0.5*0 = 0.5$

Relevancia – Puntuación por Zonas

- ¿Cómo elegir los coeficientes g_i?
- Manualmente – experto
- Usuario - ideal
- Mediante aprendizaje automático
  - Algunos ejemplos etiquetados manualmente con la relevancia, tuplas (d,q,r)
  - $r \in$ *{relevante, no relevante}*
  - Se “aprenden” los coeficientes g_i
  - Ajustar los datos a una función lineal
  - Difícil conseguir los ejemplos etiquetados

Relevancia – Puntuación por Zonas

- Caso simple = problema de optimización
- 2 zonas, por ejemplo *título* y *cuerpo (body)*
- Suponemos que existe una constante *g* tal que

  $Score(d,q) = g*s_T(d,q) + (1-g)*s_B(d,q)$

  para todo (d,q)
- $s_T(d,q)$ y $s_B(d,q)$ funciones booleanas

Relevancia – Puntuación por Zonas

- *Score(d,q) = g*s_T(d,q) + (1-g)*s_B(d,q) para todo (d,q)*
- $s_T(d,q)$ y $s_B(d,q)$ funciones booleanas que podemos calcular
- Ejemplos etiquetados o de entrenamiento, ternas: $\Phi_j = (d_j, q_j, r(d_j, q_j))$
  - *$r \in$ {relevante, no relevante} = {1, 0}*

Relevancia – Puntuación por Zonas

Ejemplo casos de entrenamiento

Example	DocID	Query	s_T	s_B	Judgment
Φ_1	37	linux	1	1	Relevant
Φ_2	37	penguin	0	1	Non-relevant
Φ_3	238	system	0	1	Relevant
Φ_4	238	penguin	0	0	Non-relevant
Φ_5	1741	kernel	1	1	Relevant
Φ_6	2094	driver	0	1	Relevant
Φ_7	3191	driver	1	0	Non-relevant

Relevancia – Puntuación por Zonas

- Se comparan ambos resultados *Score* y *Φ*
- Relevate vale *1*, no relevante *0*

s_T	s_B	Score
0	0	0
0	1	$1-g$
1	0	g
1	1	1

- Error para cada ejemplo a minimizar

$$\varepsilon(g,\Phi) = (r(d_j, q_j) - Score(d_j, q_j))^2$$

- Error total a minimizar $\Sigma_j \varepsilon(g,\Phi)$

Relevancia – Puntuación por Zonas

s_T	s_B	Score
0	0	0
0	1	$1-g$
1	0	g
1	1	1

- *Minimizar el error total* en el intervalo [0,1]
- *Definimos n_{01r} como el número de ejemplos relevantes de Φ con: $s_T(d_j,q_j) = 0$ y $s_B(d_j,q_j)= 1$*
- *De forma similar los otros n_{01r} n_{01n} n_{10r} n_{01n} ...*
- *Error total = $(n_{01r} + n_{10n})g^2 + (n_{10r} + n_{01n})(1-g)^2 + n_{00r} + n_{11n}$*

Derivando resp. g

Igualando a 0

$\rightarrow (n_{10r} + n_{01n})/(n_{10r} + n_{10n} + n_{01r} + n_{01n})$

valor óptimo para g

Relevancia – Puntuación por Zonas

- ¿Es necesario que todas las funciones de puntaje (s_i) sean iguales?

  $\rightarrow \Sigma_{i=1..l}\, g_i * s_i = 1$

- Pertenencia de una palabra a la zona
- Pertenencia de todas las palabras a la zona
- Pertenencia de al menos la mitad de las palabras a la zona

Relevancia – Frecuencia de Términos

- Usar la frecuencia con que aparecen los términos en los documentos/zonas para determinar la relevancia
- Asignar un *peso* a cada documento para cada término
- $tf_{t,d}$ representa la frecuencia de aparición del término en el documento
- Los documentos son “bolsas” de términos
  - No se toma en cuenta el orden
- ¿Todas las palabras valen lo mismo? ¿Stop words?

Relevancia – Frecuencia en Documentos

- Para disminuir el peso de los términos muy frecuentes
- ¿Inverso de la frecuencia de aparición en la colección?
  - Total de ocurrencias en la colección
- Mejor: número de documentos en que aparece el término df_t
  - Depende menos de la homogeneidad o heterogeneidad de la colección
  - Comportamientos y escalas diferentes
- Inverso de la frecuencia de aparición en los documentos

$$idf_t = log\ (N/df_t)$$

  - N es el número total de documentos en la colección

Relevancia – Frecuencia en Documentos

- Reuters
- 806 791 documentos

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

Relevancia – tf-idf

- Combinación $tf\text{-}idf_{t,d} = tf_{t,d} * idf_t$
  - Peso del término *t* en el documento *d*

- Más alto cuando *t* aparece muchas veces en unos pocos documentos
- Bajo cuando *t* aparece pocas veces en algunos documentos o aparece en muchos documentos
- Más bajo cuando *t* aparece muchas veces en la colección

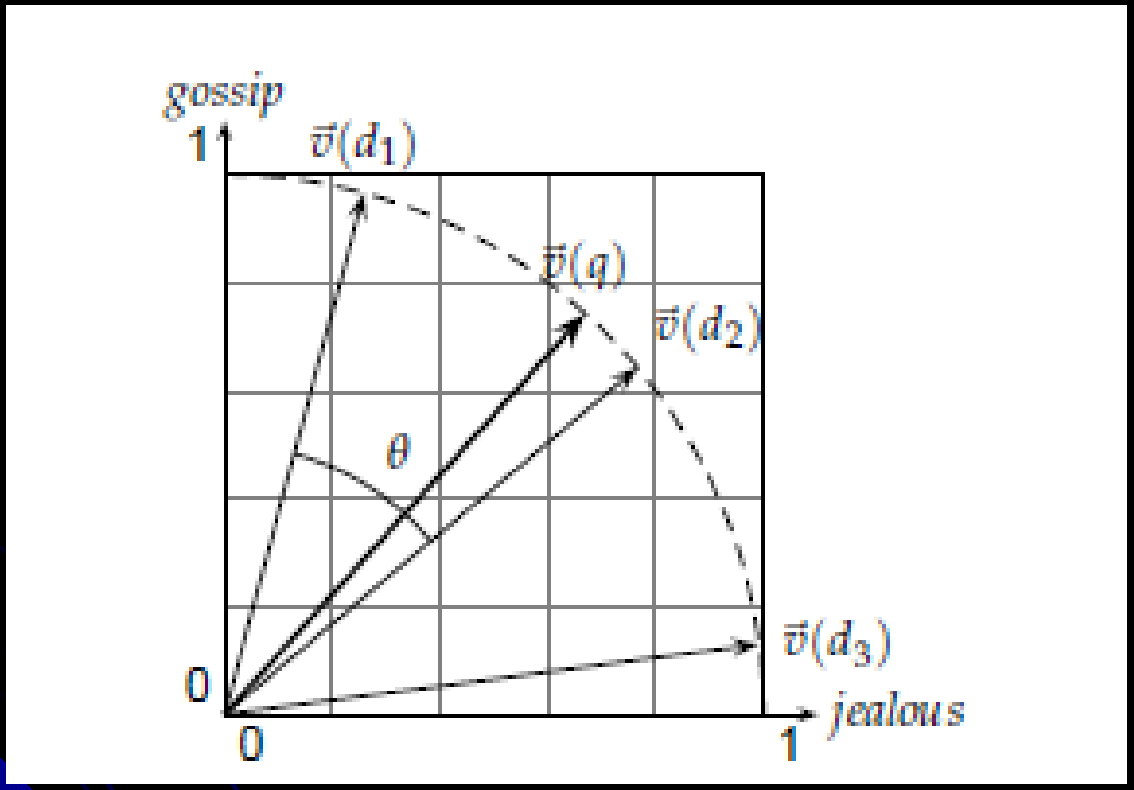
Modelo de Espacio Vectorial

- Cada documento es un vector de términos
- Las componentes son los términos
- Peso de cada componente $tf\text{-}idf_{t,d}$ o 0

- Overlap measure
  $Score(q,d) = \sum_{t \in q} tf\text{-}idf_{t,d} =$
  $\sum_{t \in q} tf_{t,d} * idf_t = \sum_{t \in q} tf_{t,d} * log\ (N/df_t)$

- ¿Cuál es el *Score* de un término que aparece en todos los documentos? ¿Stop list? $df_t = N$

Modelo de Espacio Vectorial



Modelo de Espacio Vectorial

- Cada documento es un vector de términos
- Representar las consultas como vectores en el mismo espacio vectorial

- Distancia entre vectores
  - Magnitud de la diferencia
    - Aunque sean similares si uno es mucho más largo que el otro no se verán como parecidos
    - Distribuciones relativas iguales pero distribuciones absolutas no
  - Similitud en coseno

Modelo de Espacio Vectorial

- Vector para cada doc.
  - Cada componente corresponde a un termino del diccionario
- Similitud en coseno

$$sim(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{\left|\vec{V}(d_1)\right|\left|\vec{V}(d_2)\right|} = \frac{\vec{V}(d_1)}{\left|\vec{V}(d_1)\right|} \cdot \frac{\vec{V}(d_2)}{\left|\vec{V}(d_2)\right|} = \vec{v}(d_1) \cdot \vec{v}(d_2)$$

- Producto interno

$$\vec{x} \cdot \vec{y} = \sum_{i=1}^{M} x_i y_i$$

- Norma euclídea

$$\left|\vec{x}\right| = \sqrt{\sum_{i=1}^{M} x_i^2}$$

Modelo de Espacio Vectorial

- Computar la similitud en coseno de la consulta con TODOS los documentos
  - Tomar los *K* más cercanos
- Costos muy elevados de calcular el producto interno con TODOS los documentos de la colección

Modelo de Espacio Vectorial Optimización

- $\vec{V}(q)$ no está normalizado tiene coordenadas = 1 para los términos que están en la consulta q
- Importa el orden "relativo" de los documentos:

$$\vec{V}(q)\cdot\vec{v}(d_1) \geq \vec{V}(q)\cdot\vec{v}(d_2) \Leftrightarrow \vec{v}(q)\cdot\vec{v}(d_1) \geq \vec{v}(q)\cdot\vec{v}(d_2)$$

- Alcanza con considerar:

$$sim(q,d) = \vec{V}(q)\cdot\vec{v}(d)$$

- Se debe recorrer la lista de términos de q, de la misma forma que se hacía en el modelo booleano

Modelo de Espacio Vectorial Optimización

- Tener N/df_t precalculado para c/término y $tf_{t,d}$ en cada posting
- Utilizar un Heap para obtener los *K* documentos más cercanos a la consulta
  - Construcción con *2J* operaciones
    - J es la cantidad de documentos con similitud mayor que 0
  - Se obtienen los *K* documentos más cercanos, cada uno en *log J* operaciones

Modelo de Espacio Vectorial Optimización

- Recuperación “inexacta” de *K* documentos más cercanos a la consulta - **menor tiempo**
- Encontrar un subconjunto **A** con muchos documentos con puntuación similar a los *K* documentos más cercanos a la consulta
  - Calcular similitud en coseno sólo para estos documentos
  - Seleccionar de **A** los *K* documentos más cercanos a la consulta
- Reglas basadas en tener para cada término $tf_{t,d}$ y idf_t precalculados

Modelo de Espacio Vectorial Recuperación “Inexacta”

Eliminación a nivel de índice

- No considerar documentos que no tengan al menos un término en común con la consulta
- Considerar documentos que tengan términos de la consulta con idf_t que supere un umbral μ
  - En general tienen listas de postings reducidas
  - idf_t bajo indica que son palabras muy comunes que no ayudan a diferenciar los documentos
  - ”Intendencia de Montevideo” → ”Intendencia Montevideo”
- Considerar documentos que tengan
  - Todos los términos de la consulta
  - Más de la mitad
  - 70%

Modelo de Espacio Vectorial Recuperación "Inexacta"

Lista de Documentos Favoritos – "top docs"

- Para cada término
- Precalcular para cada término del diccionario los *r* documentos con mayor puntaje
- Ante una consulta específica se toma la unión de los documentos de las listas de favoritos de los términos de la consulta
  - Calcular similitud en coseno sólo para estos documentos
  - Devolver los *K* documentos más cercanos a la consulta
- *r*?
  - Debería ser que $r>>K$ y $K<<N$
  - Se puede tener distintos *r* para distintos términos

Modelo de Espacio Vectorial Recuperación "Inexacta"

Lista de Documentos Favoritos + Calidad de Documentos

- Calidad de los documentos $g(d) \in [0,1]$

- El puntaje para una consulta puede ser por ejemplo

$$puntaje(q,d) = g(d) + \vec{v}(q) \cdot \vec{v}(d)$$

- Precalcular para cada término del diccionario los *r* documentos con mayor puntaje *g(d)* + $tf\text{-}idf_{t,d}$
- Ante una consulta específica se toma la unión de los documentos de las listas de favoritos de los términos de la consulta

Modelo de Espacio Vectorial Recuperación "Inexacta"

Impact Ordering

- Ordenar las listas de posting por de cada término t en orden decreciente de $tf_{t,d}$
- Para cada término considerar sólo una porción inicial de la lista de postings
  - Un número fijo *r*
  - Hasta que no supere un umbral μ
- Orden diferente en las listas de postings para cada término – recorridas costosas

Modelo de Espacio Vectorial Recuperación "Inexacta"

Agrupamiento y Eliminación - Cluster Pruning

- Agrupar los documentos por similitud
- Proceso
  - Elegir $\sqrt{N}$ documentos al azar, líderes
  - Para cada documento *d*, calcular su líder más cercano y lo asignamos a dicho grupo
  - Debería ser que para cada líder hay $N/\sqrt{N}=\sqrt{N}$ seguidores
- Dada una consulta q, encontrar el líder más cercano - calcular similitud en coseno sólo para los documentos de ese grupo para encontrar los *K* más cercanos

Sistema de RI