Procesamiento del lenguaje natural
Aplicaciones

Índice

- Recuperación de información
- Clasificación y agrupamiento de documentos

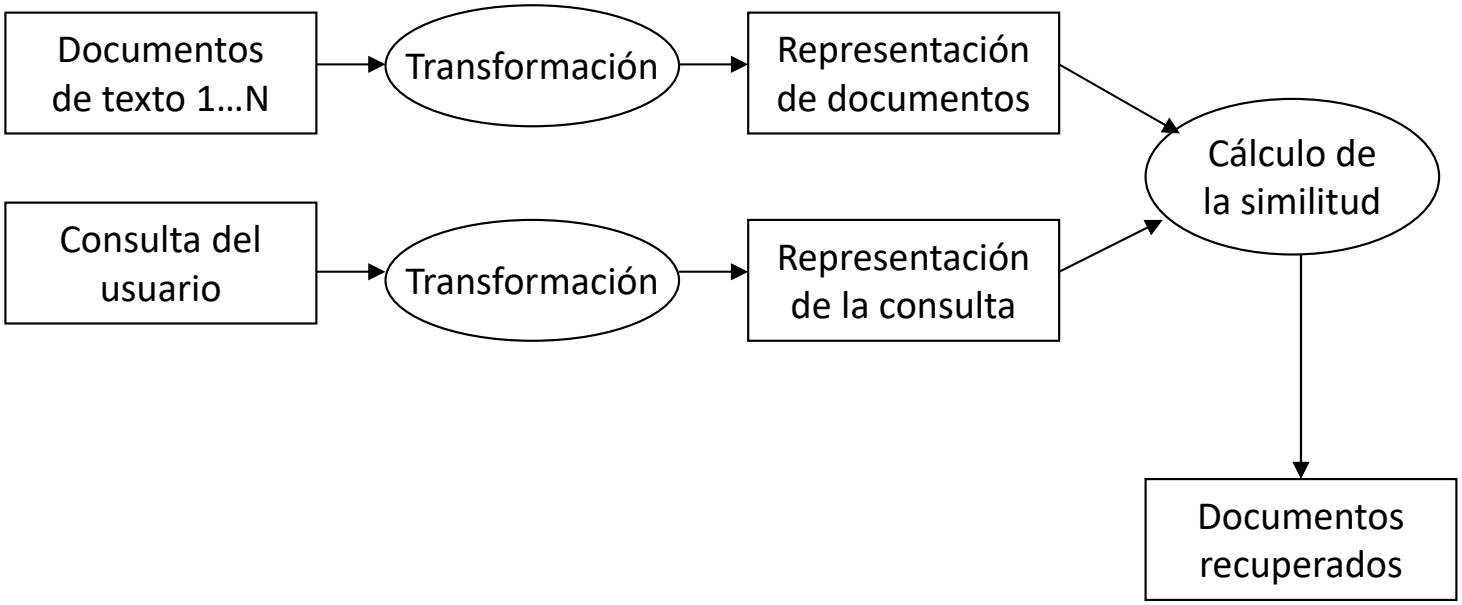
RECUPERACIÓN DE INFORMACIÓN

Recuperación de información

- La recuperación de información (*information retrieval)* consiste en encontrar los documentos relevantes asociados con una consulta
  - Es la tarea que hacen los motores de búsqueda que usamos comúnmente

- Aspectos a definir
  - Marco para modelar representaciones de documentos y consultas
  - Función de similitud entre la representación de un documento y una consulta
    - Genera un número real
    - Define un orden de los documentos con respecto a una consulta

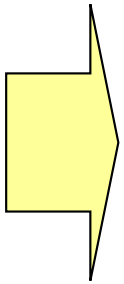
Modelo de IR

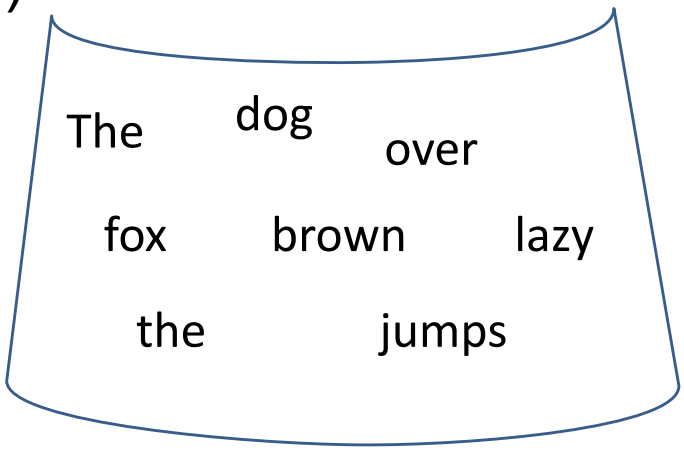

Documentos de texto 1…N
Transformación
Representación de documentos
Consulta del usuario
Transformación
Representación de la consulta
Cálculo de la similitud
Documentos recuperados

Cómo pasar de texto a datos: bolsa de palabras (bag of words)

- Se representa un documento como el conjunto de palabras que contiene, independientemente de su orden o su categoría sintáctica
- WordTokenizer (delimitadores de palabras)

The brown fox jumps over the lazy dog.

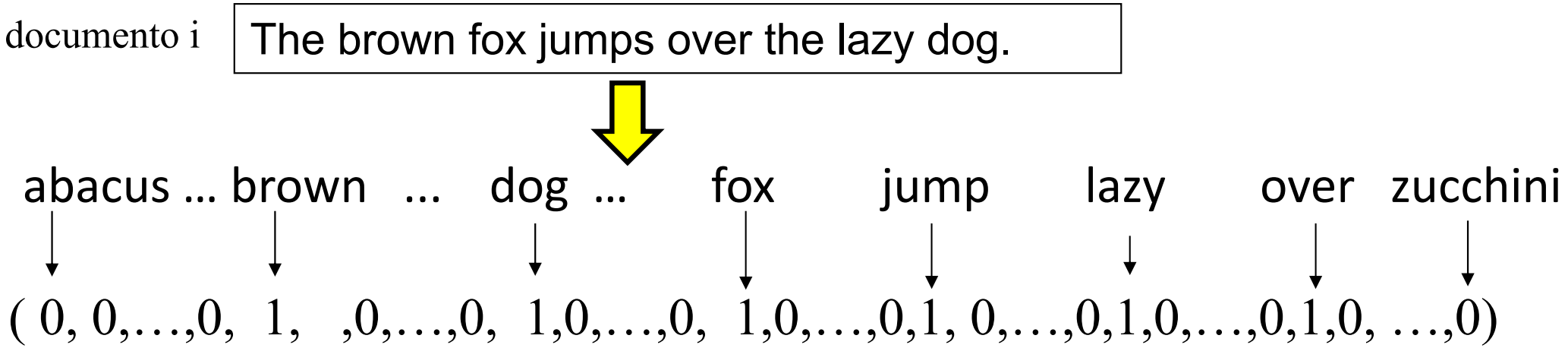




- Otros aspectos
  - Normalización
    - Minúsculas
    - Extractores de raíces (**stemmer**)
  - Eliminación de palabras vacías (**stop words**)
    - Artículos, preposiciones, adverbios…

Generación de las variables o atributos

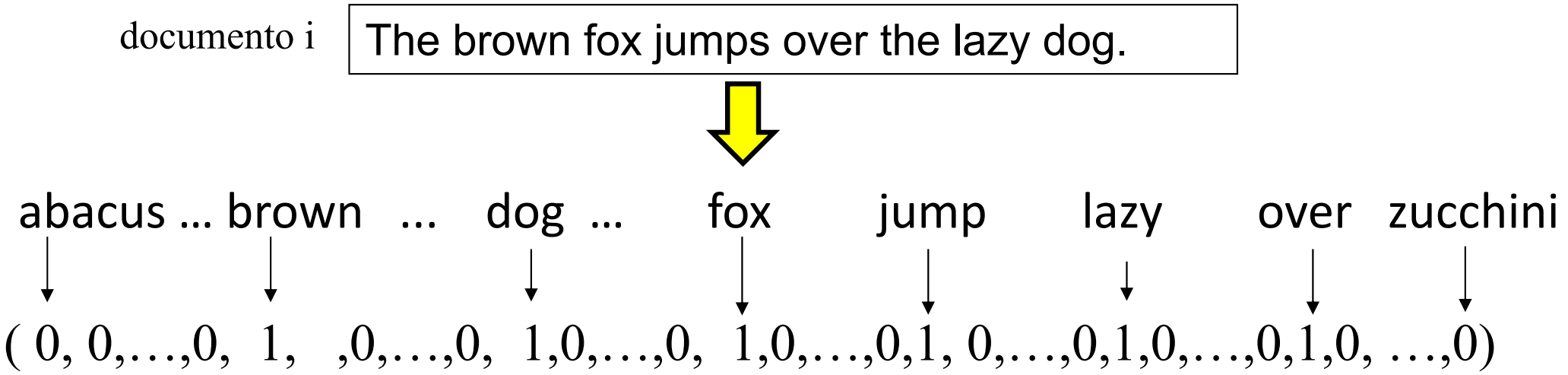
1. Enumerar todas las palabras en todos los documentos
2. Eliminar duplicados y ordenarlas
3. Convertir cada palabra en un valor
4. Crear un vector cuyo valor iésimo corresponde al término iésimo



Son vectores dispersos donde la mayoría de los valores es 0

Valores en el vector de palabras

- El valor de un término en el vector de palabras puede
  - Ser binario para indicar la presencia o ausencia del término
  - Ser la frecuencia de aparición del término en el documento
  - Ser la frecuencia con TF/IDF según los documentos del corpus
    - Esto hace que la importancia de una palabra dependa directamente de su frecuencia de aparición en el documento e inversamente a lo común que es en el corpus
    - Lo veremos a continuación



Modelos de IR

- Representación de documento y consulta
  - Conjunto de palabras clave => términos índice
    - Vocabulario $T = \{t_1, \dots t_M\}, \quad |T| = M$
    - El índice a menudo se construye agregando las bolsas de palabras de los documentos, pero podría construirse usando un diccionario o combinando ambas aproximaciones
  - Importancia de cada término índice => peso (weight)
    - Cuantifica la importancia para describir el contenido semántico del documento
    - Peso del término i en el documento j => $F(t_i, dj) = w_{ij}$
      - $D = \{d_1, \dots d_N\}, \quad |D| = N$
      - Usar la frecuencia de aparición tiene problemas, porque puede haber palabras que son muy frecuentes en todos los documentos y por tanto son poco relevantes → Usaremos TF-IDF
    - Los pesos de todos los términos de un documento son independientes entre sí
      - Es una simplificación, porque realmente muchos términos estarán correlacionados
    - Peso del término i en la consulta => $F(ti, c) = wi$

Peso asociado a cada término en el documento

- TF-IDF *Term frequency – Inverse document frequency*
- Asigna un peso a cada término t en un documento d

$$w_{t,d} = \underbrace{tf_{t,d}}_{\text{Frecuencia del término } t \text{ en el documento } d} \times \underbrace{\log\left(\frac{N}{df_t + 1}\right)}_{\text{Inversa de la frecuencia de aparición del término en los documentos (cuanto más frecuente, menos peso } w_{t,d})}$$

N es el número total de documentos

df_t es el número de documentos del corpus donde aparece el término t y se le suma 1 para evitar dividir por cero si no existe en ninguno

En el documento d, un término tendrá más peso si es muy frecuente en dicho documento y no aparece casi en el resto

Esto hace que términos que aparecen en todos los documentos no tengan mucho peso.

Documentos como vectores

- Cada documento se representa como un vector de valores tf x idf, un componente por cada término
- Por tanto tenemos un espacio vectorial
  - Los términos son las dimensiones
  - Los documentos son puntos (vectores) en este espacio
- Una colección de documentos se puede representar como una matriz término-documento
  - M términos como columnas
  - N documentos como filas

Vectores TF-IDF

- Cada documento es un vector de dimensión M y los pesos del vector son los pesos $w_{t,d}$
  - Ejemplo: Obras de Shakespeare

vector de términos →

documentos ↓

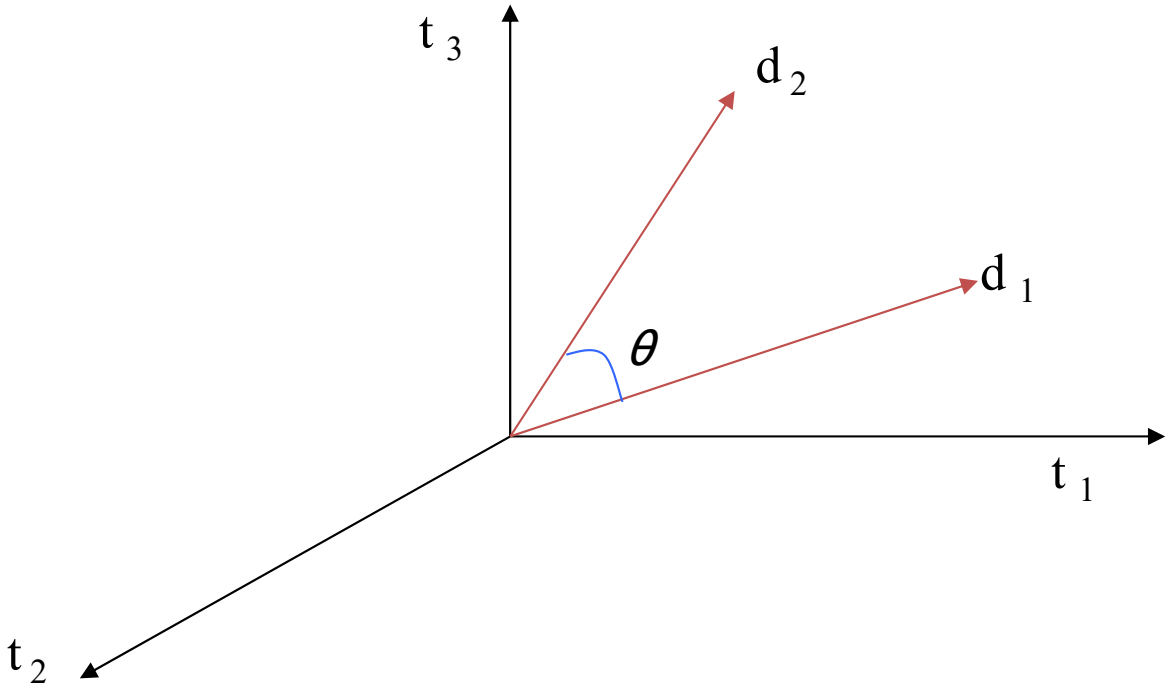
	Antony	Brutus	Caesar	Calpurnia	Cleopatra	mercy	worser
Antony and Cleopatra	13.1	3	2.3	0	17.7	0.5	1.2
Julius Caesar	11.4	8.3	2.3	11.2	0	0	0
The tempest	0	0	0	0	0	0.7	0.6
Hamlet	0	1	0.5	0	0	0.9	0.6
Othello	0	0	0.3	0	0	0.9	0.6
Macbeth	0	0	0.3	0	0	0.3	0

Similitud entre vectores: coseno

- La similitud entre los vectores d_1 y d_2 es reflejada por el coseno del ángulo que forman

Producto escalar de dos vectores

$$a \cdot b = \|a\|\|b\| \cos\theta$$



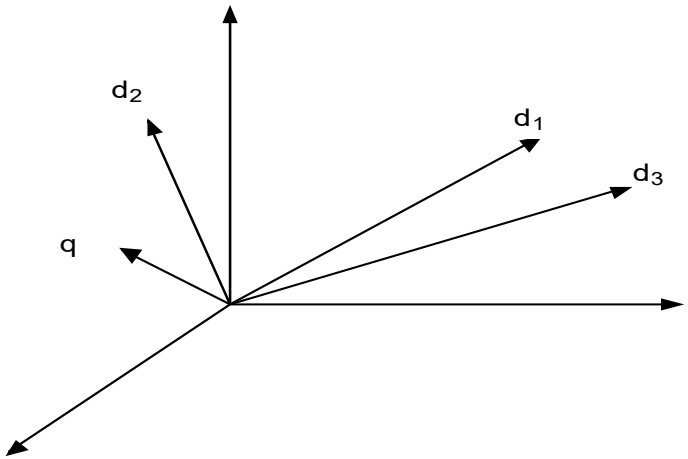
Medida de similitud del coseno

- La similitud del coseno se calcula como el coseno del ángulo que forman los dos vectores de términos
  - Si los vectores de términos son iguales, su ángulo es 0, y la similitud es la máxima, es decir, 1
  - El coseno se calcula a partir de la fórmula del producto escalar entre dos vectores
    - $a \cdot b = \|a\|\|b\| \cos\theta$
    - El producto escalar es la suma del producto de los vectores componente a componente

$$coseno(d_j, d_k) = \frac{d_j \cdot d_k}{|d_j||d_k|} = \frac{\sum_{i=1}^{M} w_{i,j} \cdot w_{i,k}}{\sqrt{\sum_{i=1}^{M} w_{i,j}^2} \sqrt{\sum_{i=1}^{M} w_{i,k}^2}}$$

IR con la similitud del coseno

Dada una consulta q y un conjunto de documentos d_i, el buscador devuelve la lista de documentos ordenada de mayor a menor según la similitud del coseno.



	algorithm	architecture	computer	logic	program
Documento1	2.23	5.34	2.45	0.00	0.00
Documento2	3.50	0.00	0.00	3.20	1.51
Documento3	0.00	4.76	3.23	0.00	2.31
Consulta	0.00	0.00	0.00	1.06	0.74

similitudes entre documentos d_i y consulta q:

$sim(d_1,q) = 0.0000$
$sim(d_2,q) = 0.7009$
$sim(d_3,q) = 0.2133$

Midiendo el rendimiento de un sistema de IR

- Si tenemos etiquetados los documentos relevantes para una consulta podemos medir la precisión y la exhaustividad
  - ¿Todos los documentos que recupera son relevantes? → Precisión
  - ¿Recupera todos los documentos relevantes? → Exhaustividad
- Estas métricas son las mismas que se usan para clasificación
  - Exhaustividad (recall) o Tasa de Verdaderos Positivos: $TVP = \frac{VP}{VP+FN} = \frac{VP}{P}$
  - Precisión o Valor Predictivo Positivo: $VPP = \frac{VP}{VP+FP}$
  - Medida F1: $F1 = 2 \cdot \frac{VPP \cdot TVP}{VPP+TVP} = \frac{2 \cdot VP}{2 \cdot VP+FP+FN}$
    - Es la media armónica de Precisión y Exhaustividad

		Documento relevante	
		1	0
Documento recuperado	1	*VP*	*FP*
	0	*FN*	*VN*

- En los buscadores típicamente se evalúa la calidad de un ranking, es decir, que los documentos más relevantes se devuelvan en las primeras posiciones
  - Para ello se mide la precisión en cada posición de la lista, esto equivale a relacionar la precisión y la exhaustividad
    - Porque cada posición de la lista equivale a un % de exhaustividad
    - Si la lista tiene 100 elementos, cada elemento supone un 1% de la exhaustividad
  - Con esto se puede pintar una curva y como resumen de ella se puede calcular la media de la precisión para todo el rango de exhaustividad (es decir, el área bajo la curva o la integral)

Ventajas e inconvenientes de la bolsa de palabras

Al representar texto mediante vectores de palabras estamos dejando de trabajar con texto propiamente dicho, esto es un arma de doble filo.

- Ventajas
  - Simplificación del problema que en muchos casos funciona
  - Permite efectuar consultas de manera sencilla
  - Permite trabajar textos con técnicas estadísticas y de aprendizaje automático

- Inconvenientes
  - No maneja la ambigüedad y la variabilidad léxica
    - Considera diferentes dos palabras sinónimas (casa y hogar)
    - Reconoce como iguales palabras polisémicas (ratón)
  - Los vectores resultantes son dispersos (muchos ceros) lo cual es un problema para muchas técnicas de análisis de datos
  - Requiere trabajar con corpus específicos si trabajamos en dominios especializados (con léxico propio), p.ej. medicina

Existen soluciones para muchos de estos problemas pero no las veremos aquí.

CLASIFICACIÓN Y AGRUPAMIENTO DE DOCUMENTOS

Clasificación y agrupamiento de documentos

- Categorización de textos (Text categorization)
  - Consiste en clasificar documentos en categorías predefinidas.
    - Clasificar correo en spam o no-spam.
    - Asignar etiquetas temáticas a noticias
    - Decir si una opinión es positiva o negativa
  - Enfoque supervisado
    - Aprendizaje de una función que asigne documentos a categorías
    - Requiere de un conjunto de documentos previamente etiquetado
      - (datos de entrenamiento)

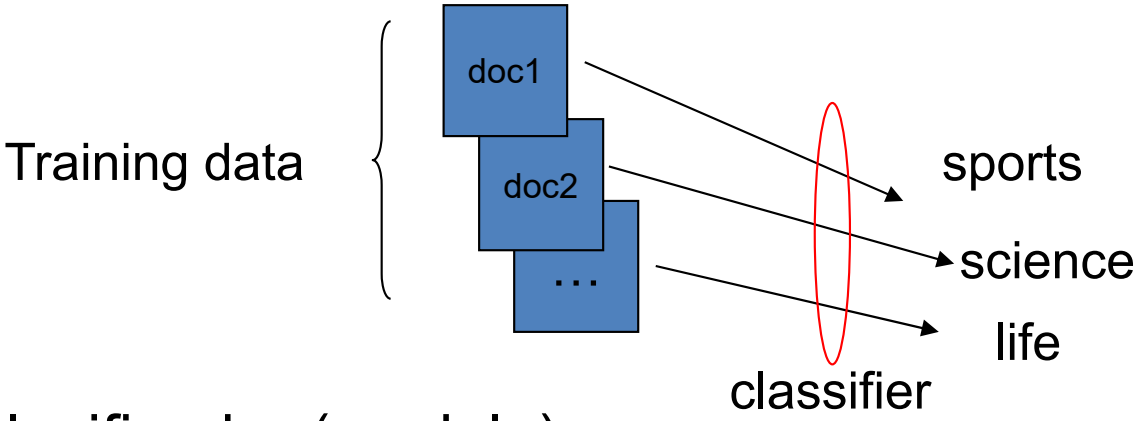
- Clustering de documentos
  - Agrupación de documentos con características similares
  - Enfoque no supervisado

Clasificación y agrupamiento de documentos

- Para realizar estas tareas se trabaja, como hemos visto en aprendizaje automático, con una matriz de datos
  - Las filas son los documentos
  - Las columnas, es decir las variables son las palabras (o la raíz de las mismas)
  - En las celdas tenemos
    - Valores binarios
    - Frecuencia de aparición (normalmente el valor relativo con respecto al número de palabras del documento)
    - TF/IDF

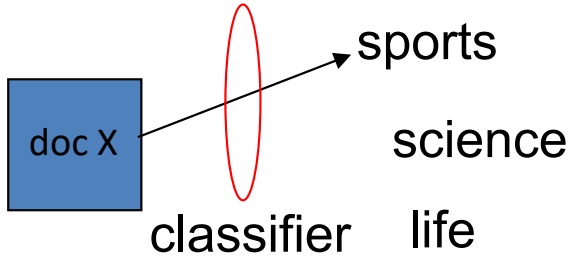
Categorización de texto

1. Construir un conjunto de entrenamiento clasificando cada documento
   - Si pueden pertenecer a varias clases es un problema de clasificación multi-etiqueta (multilabel)



2. Crear un clasificador (modelo)
   - Aplicando un algoritmo de aprendizaje automático a los datos de entrenamiento
     - Redes Neuronales, Support Vector Machine (SVM), Naïve Bayes (NB)

3. Clasificar nuevos documentos con el clasificador



Clasificadores para categorización de texto

- Los problemas de clasificación de texto sufren, a menudo, del problema de la maldición de la dimensionalidad
  - Gran cantidad de variables y, según el caso, incluso más que documentos
- Además, la matriz de datos resultante suele ser una matriz dispersa (con muchos valores a cero)
- Para evitarlo se deben usar métodos de clasificación que no se vean afectados por estas características del problema
- También se usan técnicas de reducción de la dimensionalidad (aprendizaje no supervisado) para reducir las variables a un número de factores
  - Estos factores a la postre constituyen los ejes "temáticos" o "conceptuales" del conjunto de documentos

Clasificación probabilística de documentos

- Para aproximarnos al problema vamos a suponer:
  - una variable de clase y con m categorías
  - el vector $\boldsymbol{x_i}$ de términos con d **variables binarias** que representan la presencia o no de un término i en el documento
    - Luego veremos cómo trabajar con la frecuencia del término

- Si quisiésemos aproximarnos al problema de clasificación utilizando la tabla de la distribución conjunta de las probabilidades, necesitaríamos especificar $2^{d+m-1}-1$ probabilidades para todas las combinaciones de valores de $\boldsymbol{x}$ e y, es decir, $p(y \wedge x_1 \wedge \cdots \wedge x_d)$
  - El teorema de Bayes puede ayudarnos a mitigar este problema

OJO: representaremos el vector de términos $\boldsymbol{x_i}$ en negrita
y cada término k de ese vector sin negrita $x_{k,i}$

Aproximación bayesiana al problema

- Según el teorema de Bayes, dado un vector de términos $\boldsymbol{x_i}$ para el que queremos determinar su clase y_j, tenemos que calcular la probabilidad condicionada cada clase y_j y optar por la clase más probable

$$P(y = y_j | \boldsymbol{x} = \boldsymbol{x_i}) = \frac{P(y = y_j)P(\boldsymbol{x} = \boldsymbol{x_i} | y = y_j)}{P(\boldsymbol{x} = \boldsymbol{x_i})}$$

- Las probabilidades de $P(\boldsymbol{x} = \boldsymbol{x_i})$ podemos obtenerlas a partir de los datos observados en los que se ha observado el vector de términos $\boldsymbol{x_i}$, aunque veremos que no son estrictamente necesarias
- Las probabilidades a priori $P(y = y_j)$ también se obtienen de los datos
  - Si n_j es el número de documentos de la clase $y = y_j$, $P(y = y_j) = {}^{n_j}/_{n}$
- Las probabilidades condicionales $P(\boldsymbol{x} = \boldsymbol{x_i} | y = y_j)$ a determinar son 2^d y además debemos tener en cuenta la dependencia

$$P(x_1 \wedge \cdots \wedge x_d | y) = P(x_1 | y)P(x_2 \wedge \cdots \wedge x_d | y \wedge x_1) = P(x_1 | y)P(x_2 | y \wedge x_1)P(x_3 \wedge \cdots \wedge x_d | y \wedge x_1 \wedge x_2) = \ldots$$
$$= P(x_1 | y)P(x_2 | y \wedge x_1) \ldots P(x_d | y \wedge x_1 \wedge \cdots \wedge x_{d-1})$$

  - La dependencia de las variables del vector $\boldsymbol{x}$, complica el problema

Ignorando la dependencia

- Si ignoramos las posibles dependencias entre las variables del vector $\boldsymbol{x}$ y asumimos que son independientes, es decir, que para todo $a \neq b$, tenemos que $P(x_a|y \wedge x_b) = P(x_a|y)$
  - Esto supone asumir que la co-ocurrencia de términos es totalmente independiente, lo que es radicalmente falso
- Sin embargo, los cálculos se simplifican enormemente ya que

$$P(x_1 \wedge \cdots \wedge x_d|y) = \prod_{k=1}^{d} P(x_k|y)$$

siendo x_k una variable binaria que indica la presencia del término k

- El clasificador naïve Bayes simplifica la realidad asumiendo que todas las variables utilizadas para clasificar son independientes
  - Esta hipótesis será en muchísimos casos incorrecta
    - De ahí que su nombre sea *naïve Bayes* (Bayes ingenuo) o maliciosamente *idiot Bayes* (Bayes idiota)
  - Sin embargo, pese a ello, el naïve bayes obtiene resultados tan buenos o mejores que otras técnicas de clasificación más sofisticadas

El clasificador Naïve Bayes

- Dado el vector binario de términos $\boldsymbol{x_i}$ para el que queremos determinar su clase (o categoría) y_j, tenemos que determinar

$$P(y = y_j | \boldsymbol{x} = \boldsymbol{x_i}) = \frac{P(y=y_j) \prod_{k=1}^{d} P(x_k=x_{ki}|y=y_j)}{P(\boldsymbol{x}=\boldsymbol{x_i})}$$

  - Para ello, no es necesario estimar $P(\boldsymbol{x} = \boldsymbol{x_i})$, porque al estar en el denominador es un valor constante para todas las posibles clases y_j
  - En ese caso estamos obteniendo una estimación de la verosimilitud de que $\boldsymbol{x_i}$ sea de clase y_j que es proporcional a la probabilidad

$$P(y = y_j | \boldsymbol{x} = \boldsymbol{x_i}) \propto P(y = y_j) \prod_{k=1}^{d} P(x_k = x_{ki} | y = y_j)$$

siendo $x_{ki} \in \{0,1\}$ un valor binario que indica la presencia o no del término x_k en el documento i

- Con Naive Bayes solo habría que especificar $md + m - 1$ probabilidades
  - Siendo m el número de clases y d el número de términos (variables binarias) del vector que define a los elementos

El clasificador Naïve Bayes

$$P(y = y_j | \boldsymbol{x} = \boldsymbol{x_i}) \propto P(y = y_j) \prod_{k=1}^{d} P(x_k = x_{ki} \,|y = y_j)$$

- La probabilidad de observar cada clase y_i es la frecuencia relativa de casos observados en el conjunto de entrenamiento en la clase

$$P(y = y_j) = frec(y = y_j)$$

- La probabilidad de observar el término x_k en el documento i condicionada que el documento sea de la clase y_i se calcula mediante frecuencias relativas como sigue

$$P(x_k = x_{ki} \,|y = y_j) = \frac{frec(x_k = x_{ki} \wedge y = y_j)}{frec(y = y_j)}$$

- Esta estimación da el óptimo según la estimación por máxima verosimilitud
  - Sin embargo, presenta problemas en muestras pequeñas que hay que corregir con métodos de alisado (p.ej Laplace o interpolación línea)

Ejemplo de clasificación con naïve Bayes

- Supongamos que debemos entrenar un conjunto de 220 emails para elaborar un filtro de SPAM y observamos las siguientes frecuencias

$$P(x_k = x_{ki} \,|y = y_j) = \frac{frec(x_k = x_{ki} \wedge y = y_j)}{frec(y = y_j)}$$

Emails de cada tipo

SPAM	No-SPAM
20	200

	SPAM	No-SPAM
millonario	5	5
Viagra	5	4
Nigeria	2	10
estimado	15	120
examen	2	50
ejercicio	4	40
clase	2	25

Prob. cond. de presencia de término

	SPAM	No-SPAM
millonario	0,25	0,025
Viagra	0,25	0,02
Nigeria	0,1	0,05
estimado	0,75	0,6
examen	0,1	0,25
ejercicio	0,2	0,2
clase	0,1	0,125

Prob. cond. de ausencia de término

	SPAM	No-SPAM
millonario	0,75	0,975
Viagra	0,75	0,98
Nigeria	0,9	0,95
estimado	0,25	0,4
examen	0,9	0,75
ejercicio	0,8	0,8
clase	0,9	0,875

Basta con calcular una tabla porque la otra es el complementario de la columna

Ejemplo de clasificación con naïve Bayes

$$P(y = y_j | \boldsymbol{x} = \boldsymbol{x_i}) \propto P(y = y_j) \prod_{k=1}^{d} P(x_k = x_{ki} \,|y = y_j)$$

$$P(x_k = x_{ki} \,|y = y_j) = \frac{frec(x_k = x_{ki} \wedge y = y_j)}{frec(y = y_j)}$$

Emails de cada tipo

SPAM	No-SPAM
20	200

Prob. cond. de presencia de término

	SPAM	No-SPAM
millonario	0,25	0,025
Viagra	0,25	0,02
Nigeria	0,1	0,05
estimado	0,75	0,6
examen	0,1	0,25
ejercicio	0,2	0,2
clase	0,1	0,125

Prob. cond. de ausencia de término

	SPAM	No-SPAM
millonario	0,75	0,975
Viagra	0,75	0,98
Nigeria	0,9	0,95
estimado	0,25	0,4
examen	0,9	0,75
ejercicio	0,8	0,8
clase	0,9	0,875

Nuevo email a clasificar

$\boldsymbol{x_N}$ ={millonario, Nigeria, estimado, clase}

$$P(y = SPAM|\boldsymbol{x} = \boldsymbol{x_N}) = \frac{20}{220}(0{,}25 \cdot 0{,}75 \cdot 0{,}1 \cdot 0{,}75 \cdot 0{,}9 \cdot 0{,}8 \cdot 0{,}1) = \mathbf{0,92 \cdot 10^{-4}}$$

$$P(y = No - SPAM|\boldsymbol{x} = \boldsymbol{x_N}) = \frac{200}{220}(0{,}025 \cdot 0{,}98 \cdot 0{,}05 \cdot 0{,}6 \cdot 0{,}75 \cdot 0{,}8 \cdot 0{,}125) = 0{,}501 \cdot 10^{-4}$$

Es más verosímil que el email recibido sea SPAM

Naive Bayes con variables cuantitativas

- Si x_k es una variable continua (por ejemplo, frecuencia o peso de la palabra) en lugar de binaria necesitamos una manera diferente para estimar $P(x_k = x_{ki} \,|y = y_j)$
- Normalmente se suele asumir que la variable x_k sigue una distribución normal cuya media y varianza dependen de y
  - OJO: este supuesto puede no ser muy acertado cuando hablamos de frecuencias de palabras en un corpus de documentos
- Durante el entrenamiento, para cada combinación de un atributo continuo x_k con un valor de clase $y = y_j$, se estimará su media μ_{kj} y su desviación típica σ_{kj} según los datos observados
- Durante la fase de clasificación se estimará la $P(x_k = x_{ki} \,|y = y_j)$ de un ejemplo concreto utilizando la función de distribución gaussiana de media μ_{kj} y desviación típica σ_{kj}

$$P(x_k = x_{ki} \,|y = y_j) = \frac{1}{\sigma_{kj}\sqrt{2\pi}} \exp\left(\frac{-(x_{ki} - \mu_{kj})^2}{2\sigma_{kj}^2}\right)$$

RECURSOS DE PLN

Enlaces

- Enlaces interesantes:
  - **Natural Language Processing by David Bamman (UC Berkeley):** diapositivas de un curso muy completo sobre PLN
    - https://people.ischool.berkeley.edu/~dbamman/nlp20.html
  - **NLTK**: librería de código abierto, escrita en Python, con herramientas avanzadas de PLN muy usada en investigación y docencia
    - http://www.nltk.org/
  - **Lucene**: librería de código abierto para implementar motores de búsqueda (buscadores) escrita en Java (y portada a múltiples lenguajes)
    - http://lucene.apache.org/
  - **Freeling**: librería de código abierto, escrita en C++, con herramientas avanzadas de PLN, desarrollada por la UPC
    - http://nlp.lsi.upc.edu/freeling/