

# Sistemas de Recuperación de Información

---

Aplicaciones del  
agrupamiento y  
de la clasificación  
en la recuperación  
de información  
en la Web

---

## **Autores**

*Laura Victoria Riera Pérez*

*Marcos Manuel Tirador del Riego*

**Resumen** Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum pretium libero non odio tincidunt semper. Vivamus sollicitudin egestas mattis. Sed vitae risus vel ex tincidunt molestie nec vel leo. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Maecenas quis massa tincidunt, faucibus magna non, fringilla sapien. In ullamcorper justo a scelerisque egestas. Ut maximus, elit a rutrum viverra, lectus sapien varius est, vel tempor neque mi et augue. Fusce ornare venenatis nunc nec feugiat. Proin a enim mauris. Mauris dignissim vulputate erat, vitae cursus risus elementum at. Cras luctus pharetra congue. Aliquam id est dictum, finibus ligula sed, tempus arcu.

**Palabras claves:** one, two, three, four

# Índice general

1. Introducción .....	1
2. Agrupamiento .....	1
2.1. Medidas de similitud .....	1
2.2. Agrupamiento particionado .....	2
2.3. Agrupamiento jerárquico .....	6
2.4. Aplicaciones a la RI .....	6
2.5. Ventajas .....	7
2.6. Desventajas .....	7
3. Clasificación .....	7
3.1. Problemas de la Recuperación de la Información .....	8
3.2. Naive Bayes .....	9
3.3. Feature Selection .....	10
3.4. K Nearest Neighbor .....	13
3.5. Evaluación de la clasificación .....	14
3.6. Ventajas y Desventajas .....	15
3.7. Aplicaciones a la RI .....	16
3.8. Ventajas .....	17
3.9. Desventajas .....	17
4. Agrupamiento vs. Clasificación .....	17
5. Ejemplos de aplicación .....	17
6. Conclusiones .....	18

## 1. Introduccin

## 2. Agrupamiento

Los algoritmos de agrupamiento, como su nombre lo indica, agrupan un conjunto de documentos en subconjuntos o clsteres. Son utilizados para generar una estructura de categoras que se ajuste a un conjunto de observaciones.

Los grupos formados deben tener un alto grado de asociacin entre los documentos de un mismo grupo, es decir, deben ser lo ms similares posibles, y un bajo grado entre miembros de diferentes grupos.

Es la forma ms comn de *aprendizaje no supervisado*, es decir no hay ningn experto humano que haya asignado documentos a clases. Es la distribucin y composicin de los datos lo que determinar la pertenencia al clster.

Si bien el agrupamiento a veces es denominada clasificacin automtica, esto no es estrictamente exacto ya que las clases formadas no se conocen antes de tratamiento, como implica la clasificacin, sino que se definen por los elementos que se les asignan.

La entrada clave para un algoritmo de agrupamiento es la medida de distancia. Diferentes medidas de distancia dan lugar a diferentes agrupamientos. En el agrupamiento de documentos, la medida de distancia es a menudo tambin la distancia euclidiana.

**Hipótesis de agrupamiento:** *Los documentos en el mismo grupo se comportan de manera similar con respecto a la relevancia para las necesidades de informacin.*

La hiptesis establece que si hay un documento de un grupo que es relevante a una solicitud de bsqueda, entonces es probable que otros documentos del mismo clster tambin sean relevantes.

Los algoritmos de agrupamiento pueden ser clasificados según la pertenencia a los grupos en *agrupamiento exclusivo o fuerte* (hard clustering) donde cada documento es miembro de exactamente un grupo; o *agrupamiento difuso o suave* (soft clustering) donde un documento tiene membresía fraccionaria en varios grupos.

También pueden clasificarse según el tipo de estructura impuesta sobre los datos como *agrupamiento particionado o plano* (flat clustering) o *agrupamiento jerárquico* (hierarchical clustering).

### 2.1. Medidas de similitud

#### ■ Coeficiente de Dice:

$$S_{D_i, D_j} = \frac{2 \sum_{k=1}^{max} (w_{ik} w_{jk})}{\sum_{k=1}^{max} w_{ik}^2 + \sum_{k=1}^{max} w_{jk}^2}$$

#### ■ Coeficiente de Dice para documentos representados como vectores binarios:

$$S_{D_i, D_j} = \frac{2C}{A + B}$$

■ **Coefficiente de Jaccard:**

$$S_{D_i, D_j} = \frac{\sum_{k=1}^{max} (w_{ik} w_{jk})}{\sum_{k=1}^{max} w_{ik}^2 + \sum_{k=1}^{max} w_{jk}^2 - \sum_{k=1}^{max} (w_{ik} w_{jk})}$$

■ **Coefficiente del coseno:**

$$S_{D_i, D_j} = \frac{\sum_{k=1}^{max} (w_{ik} w_{jk})}{\sqrt{\sum_{k=1}^{max} w_{ik}^2 \sum_{k=1}^{max} w_{jk}^2}}$$

## 2.2. Agrupamiento particionado

El *agrupamiento particionado* crea un conjunto de clsteres sin ninguna estructura explícita que los relacione entre sí.

### Medidas de evaluación

- **Pureza:** Para calcular la pureza, cada grupo se asigna a la clase que es más frecuente en el grupo, y luego se mide la precisión de esta asignación contando el número de documentos correctamente asignados y dividiendo por N.

Formalmente

$$purity(\Omega, \mathbb{C}) = \frac{1}{N} \sum_k \max_j |\omega_k \cap c_j|$$

donde  $\Omega = \{\omega_1, \omega_2, \dots, \omega_K\}$  es el conjunto de clústers y  $\mathbb{C} = \{c_1, c_2, \dots, c_J\}$  es el conjunto de clases. Se interpreta  $\omega_k$  como el conjunto de documentos en el clúster y  $c_j$  como el conjunto de documentos que pertenecen a esa clase. Malos agrupamientos tienen valores de pureza cercanos a 0, mientras que un agrupamiento perfecto tiene una pureza 1.

La alta pureza es fácil de lograr cuando el número de grupos es grande, en particular, la pureza es 1 si cada documento tiene su propio grupo. Sin embargo en este caso tendríamos la mayor cantidad de grupos posibles, por lo que nuestra solución no tendría calidad.

- **Información mutua normalizada:** Una medida que nos permite hacer la compensación entre calidad y pureza es la información mutua normalizada o *NMI* por sus siglas en inglés.

$$NMI(\Omega, \mathbb{C}) = \frac{I(\Omega, \mathbb{C})}{[H(\Omega) + H(\mathbb{C})]/2}$$

I es la información mutua:

$$I(\Omega, \mathbb{C}) = \sum_k \sum_j P(\omega_k \cap c_j) \log \frac{P(\omega_k \cap c_j)}{P(\omega_k)P(c_j)}$$

donde  $P(\omega_k)$ ,  $P(c_j)$  y  $P(\omega_k \cap c_j)$  son las probabilidades de que un documento sea del clúster  $\omega_k$ , clase  $c_j$ , y esté la intersección de  $\omega_k$  y  $c_j$ , respectivamente.

Para estimaciones de máxima verosimilitud de las probabilidades (es decir, la estimación de cada probabilidad es la correspondiente frecuencia relativa):

$$I(\Omega, \mathbb{C}) = \sum_k \sum_j |\omega_k \cap c_j| \log \frac{N |\omega_k \cap c_j|}{|\omega_k| |c_j|}$$

H es la entropía tal como se define en el Capítulo 5 (página 99): La característica de una distribución de probabilidad discreta  $P$  que determina sus propiedades de codificación (incluyendo si un código es óptimo)

$$H(\Omega) = - \sum_k P(\omega_k) \log P(\omega_k)$$

donde, de nuevo, la segunda ecuación se basa en estimaciones de máxima verosimilitud de las probabilidades

$I(\Omega, \mathbb{C})$  mide la cantidad de información por la cual nuestro conocimiento sobre las clases aumenta cuando se nos dice cuáles son los grupos.

El mínimo de  $I(\Omega, \mathbb{C})$  es 0 si el agrupamiento es aleatorio con respecto a pertenencia a clases. En ese caso, sabiendo que un documento está en un determinado cluster no nos da ninguna información nueva sobre cuál podrá ser su clase.

Se alcanza la máxima información mutua para un agrupamiento exacto que perfectamente recrea las clases, pero también si los grupos en exacto se subdividen en conglomerados más pequeños. En particular, un agrupamiento con  $K = N$  los grupos de documentos tienen MI máximo. Entonces MI tiene el mismo problema que la pureza: no penaliza cardinalidades grandes y por lo tanto no formaliza nuestro sesgo que, en igualdad de condiciones, menos grupos son mejores.

La normalización por el denominador  $[H(\Omega) + H(\mathbb{C})]/2$  soluciona este problema ya que la entropía tiende a aumentar con el número de clusters. Por ejemplo,  $H(\Omega)$  alcanza su log N máximo para  $K = N$ , lo que asegura que NMI es bajo para  $K = N$ . Debido a que NMI está normalizado, podemos usar para comparar agrupaciones con diferentes números de agrupaciones. Lo particular Se elige la forma del denominador porque  $[H(\Omega) + H(\mathbb{C})]/2$  es una parte superior estrecha atado en  $I(\Omega, \mathbb{C})$ . Por lo tanto, NMI es siempre un número entre 0 y 1.

- **Índice de frontera (Rand Index):** Una alternativa a esta interpretación de la información del agrupamiento es verlo como una serie de decisiones, una para cada uno de los  $N(N-1)/2$  pares de documentos de la colección. Queremos asignar dos documentos al mismo clúster si y sólo si son similares. Una decisión positiva verdadera (TP) asigna dos documentos similares al mismo grupo, una verdadera decisión negativa (TN) asigna dos documentos diferentes a diferentes grupos. Hay dos tipos de errores que podemos cometer. Una decisión de falso positivo (FP) asigna dos documentos similares al mismo clúster. Una decisión de falso negativo (FN) asigna dos

documentos similares a diferentes agrupaciones. El índice de Rand (IR) mide NDICE ALEATORIO Rhode Island el porcentaje de decisiones que son correctas. Es decir, es simplemente precisin.

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

El índice de Rand otorga el mismo peso a los falsos positivos y falsos negativos. Separar documentos similares a veces es peor que poner pares de documentos similares en el mismo grupo.

- **Medida F:** Podemos usar la medida F, vista anteriormente en conferencia, para penalizar los falsos negativos más fuertemente que los falsos positivos seleccionando un valor  $\beta > 1$ , dando así más peso a la recuperación.

$$P = \frac{TP}{TP + FP} \quad R = \frac{TP}{TP + FN}$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

## Algoritmos

- **K-means**

Es el algoritmo de agrupamiento plano más importante.

Su *objetivo* es minimizar la distancia euclidiana al cuadrado promedio de documentos de sus centros de conglomerados donde un centro de conglomerados se define como la media o centroide  $\mu$  de los documentos en un grupo  $\omega$ :

$$\vec{\mu}(\omega) \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$$

Se asume que los documentos se representan como de longitud normalizada vectores en un espacio de valor real de la manera habitual.

Usamos centroides para Rocchio clasificación. Aquí juegan un papel similar.

El grupo ideal en K-means es una esfera con el baricentro como su centro de gravedad. Idealmente, los grupos no deberían superponerse.

Nuestro desiderata para las clases en la clasificación de Rocchio fueron los mismos.

La diferencia es que no tenemos el conjunto de entrenamiento de verdad en agrupamiento para el cual sabemos qué documentos deben estar en el mismo grupo.

Una medida de qué tan bien los centroides representan a los miembros de su grupo es la suma residual de cuadrados o RSS, la distancia al cuadrado de cada vector desde su centroide sumado sobre todos los vectores:

$$RSS_k = \sum_{\vec{x} \in \omega_k} |\vec{x} - \vec{\mu}(\omega_k)|^2$$

$$RSS = \sum_{k=1}^K RSS_k$$

RSS es la *funcin objetivo* en K-means y nuestro objetivo es minimizarla. Ya que  $N$  es fijo, minimizar RSS es equivalente a minimizar el promedio al cuadrado distancia, una medida de qu tan bien los centroides representan sus documentos.

---

**Algorithm 1** K-Means

---

**Require:**  $\{\vec{x}_1, \dots, \vec{x}_N\}, K$

- 1:  $(\vec{s}_1, \dots, \vec{s}_K) \leftarrow \text{SelectRandomSeeds}(\{\vec{x}_1, \dots, \vec{x}_N\}, K)$
- 2: **for**  $k \leftarrow 1$  **to**  $K$  **do**
- 3:    $\vec{\mu}_k \leftarrow \vec{s}_k$
- 4: **while** stopping criterion has not been met **do**
- 5:   **for**  $k \leftarrow 1$  **to**  $K$  **do**
- 6:      $\omega_k \leftarrow \{\}$
- 7:   **for**  $n \leftarrow 1$  **to**  $N$  **do**
- 8:      $j \leftarrow \arg \min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$
- 9:      $\mu_j \leftarrow \omega_j \cup \{\vec{x}_n\}$  (reassignment of vectors)
- 10:   **for**  $k \leftarrow 1$  **to**  $K$  **do**
- 11:      $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$  (recomputation of centroids)
- 12: **return**  $\{\vec{\mu}_1, \dots, \vec{\mu}_N\}$

---



El primer paso de K-means es seleccionar como centros de conglomerados iniciales  $K$  al azar documentos seleccionados, las semillas. Luego, el algoritmo mueve los centros de los grupos en el espacio para minimizar el RSS. esto es iterativamente repitiendo dos pasos hasta que se cumpla un criterio de parada:

firma de documentos al clster con el centroide ms cercano; y recalculando cada centroide basado en los miembros actuales de su grupo.

Criterios de parada:

- Se ha completado un nmero fijo de iteraciones  $I$ . Esta condicin limita el tiempo de ejecucin del algoritmo de agrupamiento, pero en algunos casos la calidad de el agrupamiento ser deficiente debido a un nmero insuficiente de iteraciones.
- La asignacin de documentos a grupos (la funcin de particin  $\gamma$ ) no cambia entre iteraciones. Excepto en los casos con un mini local mal mam, esto produce un buen agrupamiento, pero los tiempos de ejecucin pueden ser inaceptablemente largo.
- Los centroides no cambian entre iteraciones. Esto es equivalente a  $\gamma$  no cambiando.
- Terminar cuando RSS cae por debajo de un umbral. Este criterio asegura que el agrupamiento tiene la calidad deseada despues de la terminacin. En la practica, nosotros necesita combinarlo con un lmite en el nmero de iteraciones para garantizar terminacin.
- Terminar cuando la disminucin de RSS cae por debajo de un umbral  $\zeta$ . Para  $\zeta$  pequeno, esto indica que estamos cerca de la convergencia. Una vez ms, tenemos que combinar con un lmite en el nmero de iteraciones para evitar tiempos de ejecucin muy largos.

### 2.3. Agrupamiento jerárquico

- Hierarchical agglomerative clustering
- Medidas de similitud:
  - Single link clustering
  - Complete link clustering
  - Centroid clustering
- Evaluación de calidad:
  - Group average link
  - Método de Ward
- Divisive clustering
- Cluster labeling
- Algoritmos:
  - Algoritmo HAC
  - Divisive Clustering

### 2.4. Aplicaciones a la RI

- Search result clustering
- Scatter-Gather
- Collection clustering
- Language modeling
- Cluster-based retrieval

## 2.5. Ventajas

Debido a que no es necesario identificar las clases antes del procesamiento, el análisis de conglomerados es útil para proporcionar estructura en grandes conjuntos de datos multivariados. Se ha descrito como una herramienta de descubrimiento porque tiene el potencial para revelar relaciones previamente no detectadas basadas en datos complejos.

Debido a que el análisis de conglomerados es una técnica de análisis multivariado que tiene aplicación en muchos campos, es con el apoyo de una serie de paquetes de software que a menudo están disponibles en la informática académica y otros. 21/03/2021 Recuperación de información: CAPÍTULO 16: ALGORITMOS DE CLÚSTER [orion.lcg.ufrj.br/Dr.Dobbs/books/book5/chap16.htm](http://orion.lcg.ufrj.br/Dr.Dobbs/books/book5/chap16.htm) 21/2 entornos.

## 2.6. Desventajas

## 3. Clasificación

El problema de clasificación en sentido general consiste en determinar dentro un conjunto de clases a cuál de ellas pertenece un objeto dado. En el marco de este documento estamos interesados en estudiar la clasificación de textos.

La forma más antigua de llevar a cabo la clasificación es manualmente. Por ejemplo, los bibliotecarios clasifican los libros de acuerdo a ciertos criterios, de modo que encontrar una información buscada no resulte una tarea de gigante dificultad. Sin embargo la clasificación manual tiene sus límites de escalabilidad.

Como alternativa podría pensarse el uso de *reglas* para determinar automáticamente si un texto pertenece o no a una clase determinada de documentos.

Ilustremos un ejemplo de regla aplicada automáticamente. Supongamos que un usuario necesita hacer una consulta en una página de noticias, por ejemplo, necesita tener actualidad sobre noticias relacionadas con las finanzas para tomar decisiones en su negocio. Al usuario entonces le podría interesar que de hacer una consulta en el sistema dicha consulta se mantenga ejecutando y le provea periódicamente las noticias relativas a finanzas.

A este tipo de consultas se le denomina consultas permanentes (o *standing queries* en inglés). Una consulta permanente es aquella que se ejecuta periódicamente en una colección a la cual nuevos documentos se adicionan en el tiempo. Toda consulta permanente se puede ver como un tipo de regla que se aplica a un sistema de clasificación que divide una colección en documentos que satisfacen la query y documentos que no.

Una regla captura una cierta combinación de palabras claves que identifican una clase. Reglas codificadas a mano pueden llegar a ser altamente escalable, pero crearlas y mantenerlas requiere un elevado costo en recursos humanos.

Existe, sin embargo, un enfoque adicional a los dos anteriores mencionados. Nos referimos al uso de *Aprendizaje de Máquinas*. En este enfoque el conjunto de reglas de clasificación, o en general, el criterio usado para clasificar, es aprendido de forma automática a partir de los datos de entrenamiento.

Al uso del Aprendizaje de Máquinas en la clasificación de textos se le conoce como clasificación estadística de texto (o en inglés *statistical text classification*) si el método de aprendizaje usado es estadístico.

Introduciremos a continuación la definición formal del problema de clasificación de textos, en el contexto del Aprendizaje de Máquinas.

**Definition 1.** Sea  $\mathcal{X}$  el espacio de documentos y  $\mathcal{C} := \{c_i \mid c_i \subset \mathcal{X}, i \in \{1, 2, \dots, n\}\}$  un conjunto fijo de clases (también llamadas categorías o etiquetas). Sea además  $D$  un conjunto entrenado de documentos clasificados  $(d, c) \in \mathcal{X} \times \mathcal{C}$ . El problema de la clasificación de textos consiste en encontrar, usando métodos o algoritmos de aprendizaje, una función clasificadora  $\gamma : \mathcal{X} \rightarrow \mathcal{C}$ , que mapee documentos a clases, que satisfaga que  $D \subset \gamma$ .

El aprendizaje que toma parte en la búsqueda de  $\gamma$  es llamado *aprendizaje supervisado* debido a que se necesita la ayuda de uno o varios expertos que creen el conjunto de entrenamiento  $D$ . Estos expertos son también quienes determinan el conjunto de clases en que se clasificarán los textos. Denotaremos el método de aprendizaje supervisado descrito por  $F$ , el cual actúa como una función que mapea un conjunto de datos de entrenamiento en una función clasificadora, o sea que  $F(D) = \gamma$ .

La definición dada en 1 implica que cada documento pertenece a una sola clase. Pero existe otro tipo de problemas que permiten que un documento pertenezca a más de una clase. Por ahora enfocaremos nuestra atención en el tipo de una clase.

### 3.1. Problemas de la Recuperación de la Información

El aprendizaje de máquinas es ampliamente usado en la actualidad para resolver una variedad de problemas de muchas esferas. La clasificación, de conjunto con la regresión, es una de las grandes tareas que tiene el aprendizaje de máquinas supervisado. Algunos problemas interesantes que resuelven la clasificación se encuentran dentro del marco de los Sistemas de Recuperación de la información. Anteriormente presentamos uno de ellos, que es la recuperación de consultas permanentes.

La organización del correo personal es otro de los ejemplos relacionados con RI que pueden ser resueltos mediante clasificación. En muchas ocasiones una persona tiene un número grande de correos en el buzón de entrada. A la hora de revisarlo sería deseable que pudiera dirigirse directamente a aquellos que son de interés para él. Para ello la organización automática del correo en carpetas podría ser una ventaja invaluable. Un ejemplo particular sería la carpeta de correo spam.

Otro problema que se pudiera resolver sería la clasificación de valoraciones sobre algo en positivas o negativas. Esto tendría numerosas aplicaciones prácticas como pudiera ser la selección de una película. Un usuario podría revisar la cantidad de comentarios negativos, antes de lanzarse a disponer de dos horas de su tiempo viendo un material audiovisual que no resultará muy placentero.

El uso más evidente que tiene la clasificación en la RI es la clasificación de documentos en tópicos. Esta clasificación facilitaría la implementación de un motor de búsqueda vertical, que permita al usuario restringir su búsqueda al tema deseado. Un SRI con semejante característica permite hacer una búsqueda más precisa en las consultas más especializadas por el usuario.

En RI es muy útil contar con un índice de los contenidos almacenados. En el proceso de creación del mencionado índice la clasificación puede jugar un importante papel. Por citar algunos ejemplos puede usarse para detectar el lenguaje de un documento, la segmentación y capitalización de las palabras y el codificado del documento.

### 3.2. Naive Bayes

Uno de los métodos más comunes de aprendizaje supervisado es el conocido como *Naive Bayes* (NB). Este es un método de aprendizaje probabilístico. La probabilidad de un documento  $d$  de pertenecer a una clase  $c$  se puede expresar como  $P(c | d)$ . La tarea del algoritmo es encontrar la mejor clase para cada documento  $d$ . Para ello NB establece que la clase más apropiada para un documento es la más probable, o sea

$$c_{map} = \arg \max_{c \in \mathcal{C}} P(c | d).$$

La clase escogida para  $d$  se denota por  $c_{map}$  debido a que este método de clasificación, de acuerdo a la clase más probable para un documento dado, es conocido como *maximum a posteriori* (MAP).

Sin embargo la probabilidad condicional  $P(c | d)$  es difícil de determinar. Haciendo uso del *Teorema de Bayes* la probabilidad anterior puede ser expresada como

$$P(c | d) = \frac{P(d | c)P(c)}{P(d)}.$$

El factor de normalización  $P(d)$  es usualmente ignorado ya que no aporta información a la hora de buscar la clase más apropiada para un documento  $d$ , ya que este tiene el mismo efecto en todos los candidatos. Este cálculo puede ser simplificado si lo expresamos en términos de los términos en los documentos. Supongamos que  $\{t_1, t_2, \dots, t_n\}$  son los términos que aparecen en  $d$ . Entonces tenemos que

$$c_{map} = \arg \max_{c \in \mathcal{C}} P(c)P(d | c) = \arg \max_{c \in \mathcal{C}} P(c) \prod_{1 \leq k \leq n} P(t_k | c),$$

donde  $P(t_k | c)$  es la probabilidad de que el término  $t_k$  aparezca en un documento de la clase  $c$ . Podemos considerar  $p(t_k | c)$  como una medida de qué tanto demuestra el término  $t_k$  que  $c$  es la clase correcta. El término  $P(c)$  es conocido como probabilidad previa (*prior probability*) y en caso de que la información

aportada por los términos no sea determinante en la selección podemos siempre escoger la clase con mayor valor de  $P(c)$ .

Para simplificar aún mas el cómputo podemos sustituir los valores anteriores por sus logaritmos. Esto reducirá el costo de hacer los cálculos y además los errores aritméticos, dado que la multiplicación se transforma en suma. La clase seleccionada sería entonces

$$c_{map} = \arg \max_{c \in \mathcal{C}} \left( \log(P(c)) + \sum_{1 \leq k \leq n} \log(P(t_k | c)) \right).$$

Solo nos queda ver como estimamos los parámetros  $P(c)$  y  $P(t_k | c)$ , dado que los valores reales no son posibles de calcular. Para la probabilidad previa podemos contar la frecuencia relativa de cada clase en  $D$ :

$$P(c) = \frac{N_c}{N},$$

donde  $N_c$  es el número de documentos en la clase  $c$  y  $N$  es el numero total de documentos. Procedemos de manera similar para la probabilidad específica de una palabra en una clase

$$P(t_k | c) = \frac{T_{c,t_k}}{T_c},$$

donde  $T_{c,t_k}$  indica la cantidad de veces que ocurre la palabra  $t_k$  en todos los documentos de la clase  $c$  y  $T_c$  es la cantidad total de palabras contando repeticiones en toda la clase  $c$ . Si embargo, aún tenemos un problema con estas fórmulas y es que estamos asignando probabilidad cero a todos las clases que no contengan a todas las palabras del documento a clasificar. Para evitar esto adicionamos por defecto una unidad a cada contador lo cual es conocido como *Laplace smoothing*

$$P(t | c) = \frac{T_{c,t} + 1}{T_c + |V|},$$

donde  $|V|$  es el número total de términos en el vocabulario.

Es importante destacar que en este método estamos obviando la posición de las palabras. Presentamos aquí los algoritmos para entrenar y clasificar usando BN que fueron textualmente copiados de 2009 Manning C. D., *Introduction to Information Retrieval*, página 260. [No recuerdo como citar esto correctamente. Also recordar que agregue unos paquetes arriba que no se si se agragan aqui o en otro de los .tex].

Podemos deducir de los algoritmos que la complejidad de ambos es lineal en el tiempo que toma escanear la información. Dado que esto hay que hacerlo al menos una vez, se puede decir que este método tiene complejidad temporal óptima. Dicha eficiencia hace que NB sea un método de clasificación tan usado.

### 3.3. Feature Selection

Un término con ruido (*noise feature*) es aquel que al pertenecer a la representación de los documentos, provoca un aumento del error de clasificación de

**Algorithm 2** TrainMultinomial**Require:** Set of classes  $\mathcal{C}$  and training set  $D$ .

---

```

1:  $V \leftarrow \text{ExtractVocabulary}(D)$ 
2:  $N \leftarrow \text{CountDocs}(D)$ 
3: for  $c \in \mathcal{C}$  do
4:    $N_c \leftarrow \text{CountDocsInClass}(D, c)$ 
5:    $\text{prior}[c] \leftarrow N_c/N$ 
6:    $\text{text}_c \leftarrow \text{ConcatenateTextOfAllDocsInClass}(D, c)$ 
7:   for  $t \in V$  do
8:      $T_{ct} \leftarrow \text{CountTokensOfTerm}(\text{text}_c, t)$ 
9:   for  $t \in V$  do
10:     $\text{condprob}[t][c] \leftarrow \frac{T_{c,t}+1}{\sum_{t'} (T_{c,t'}+1)}$ 
11: return  $V, \text{prior}, \text{condprob}$ 

```

---

**Algorithm 3** ApplyMultinomialNB**Require:**  $\mathcal{C}, V, \text{prior}, \text{condprob}, d$ 


---

```

1:  $W \leftarrow \text{ExtractTokensFromDoc}(V, d)$ 
2: for  $c \in \mathcal{C}$  do
3:    $\text{score}[c] \leftarrow \log \text{prior}[c]$ 
4:   for  $t \in W$  do
5:      $\text{score}[c] += \log \text{condprob}[t][c]$ 
6: return  $\arg \max_{c \in \mathcal{C}} (\text{score}[c])$ 

```

---

los datos. Por ejemplo, supongamos que tenemos una palabra que ocurre rara vez, pero que en el conjunto de entrenamiento ocurre siempre en la misma clase  $c$ . Entonces al clasificar un documento nuevo que contiene esta palabra, la misma provocará que el clasificador se incline en cierta medida por seleccionar esta a  $c$  como respuesta. Sin embargo, dado que la ocurrencia de esta palabra únicamente en  $c$  es accidental, claramente no aporta información suficiente para la clasificación y por tanto, al considerar lo contrario, aumenta el error.

Este es uno de los propósitos que tiene la selección de términos (*feature selection* (FS)). Esta consiste en reducir el vocabulario, considerado en la clasificación de textos solo un subconjunto del que aparece en el conjunto de entrenamiento. Nótese que al disminuir el tamaño del vocabulario aumenta la eficiencia de los métodos de entrenamiento y clasificación (aunque no es el caso de NB).

Selección de términos prefiere un clasificador más simple antes que uno más complejo. Esto es útil cuando el conjunto de entrenamiento no es muy grande.

En FS usualmente fijamos una cantidad  $k$  de vocablos por cada clase  $c$ , que serán los usados por el clasificador. Para seleccionar los  $k$  términos deseados establecemos un ranking entre los términos de la clase, haciendo uso de una función de medida de utilidad  $A(t, c)$ , y nos quedamos con los  $k$  mejor posicionados. El algoritmo básico consiste en para cada clase  $c$  iterar por todos los términos del vocabulario y computar su medida de utilidad para la clase; para finalmente ordenar los resultados y devolver una lista con los  $k$  mejores.

Presentaremos a continuación tres de los métodos de calcular  $A(t, c)$  más comunes.

#### ■ Información Manual.

Computar  $A(t, c)$  como el valor esperado de información mutua (*Mutual Information* (MI)), nos da una medida de cuánta información aporta, la presencia en  $c$  de un término dado, a tomar la decisión correcta de clasificación de un documento. Lo definimos como [\[Hay que poner aquí que esto se cogio del libro, pagina 272\]](#)

$$I(U_t; C_t) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} P(U_t = e_t, C_t = e_c) \log_2 \frac{P(U_t = e_t, C_t = e_c)}{P(U_t = e_t)P(C_t = e_c)},$$

donde  $U_t$  es una variable aleatoria que toma valor  $e_t = 1$  si el documento contiene el término  $t$  y  $e_t = 0$  en otro caso, y  $C$  es otra variable aleatoria que toma valor  $e_c = 1$  si el documento está en la clase  $c$  y  $e_c = 0$  en otro caso.

MI mide cuánta información un término contiene acerca de una clase. Por tanto, mantener los términos que están cargados de información, y eliminar los que no, contribuye a reducir el ruido y mejorar la precisión del clasificador.

#### ■ Selección Chi cuadrado $\chi^2$ .

En estadística se dice que dos eventos son independientes si el resultado de uno no afecta al resultado del otro. Esto se puede escribir formalmente como  $P(AB) = P(A)P(B)$ . En estadística el test  $\chi^2$  se usa para medir el grado de independencia de dos eventos. En FS podemos entonces considerar aplicar este test asumiendo como eventos la ocurrencia de los términos y la ocurrencia de las clases. Esto es [Esto tambien hay que poner de donde lo cogi \(pag 275\)](#)

$$\chi^2(D, t, c) = \sum_{e_t \in \{1,0\}} \sum_{e_c \in \{1,0\}} \frac{(N_{e_t e_c} - E_{e_t e_c})^2}{E_{e_t e_c}},$$

donde  $N$  es la frecuencia según  $D$ ,  $E$  es la frecuencia esperada y  $e_t$  y  $e_c$  se definen como en la medida anterior.

#### ■ Selección basada en frecuencia.

Esta medida consiste en priorizar los términos que son más comunes en la clase. Puede ser calculada de dos formas diferentes. La primera es cantidad de repeticiones de un término en los documentos de una clase, conocida como frecuencia en colección. La otra es frecuencia de documentos, y se calcula como la cantidad de documentos en la clase que contienen al término en cuestión.

Cuando son seleccionados varios miles de términos, entonces esta medida es bastante buena. Esta es preferible a otros métodos más complejos cuando se aceptan soluciones subóptimas.

### 3.4. K Nearest Neighbor

En el algoritmo de Naive Bayes representábamos los documentos como vectores booleanos de términos. Luego vimos que hay términos que no eran relevantes y que aportaban ruido, y lo solucionamos seleccionando para el clasificador solamente un subconjunto de todos los términos. Aún así estamos clasificando la relevancia de cada término de manera binaria en relevante o no relevante (que aporta ruido).

El método que presentamos en esta sección, así como otros similares, asignan a cada término cierto valor de importancia relativa al documento en que aparece. Para esto se cambia la representación de los documentos a vectores de  $\mathbb{R}^{|V|}$ , donde a cada componente corresponde cierto peso que se le asigna al término correspondiente a esta. Entonces, el espacio de documentos  $\mathcal{X}$  (dominio de  $\gamma$ ) es  $\mathbb{R}^{|V|}$ . A esta forma de representación de documentos se le conoce como modelo de espacio de vectores. La hipótesis básica para usar el modelo de espacio de vectores es la siguiente [citar adecuadamente : página 289](#)

**Hipótesis de contigüidad:** Documentos en la misma clase forman una región contigua y regiones de diferentes clases no se superponen.

Las decisiones de muchos clasificadores basados en espacio de vectores dependen de una noción de distancia. Pueden ser usadas por ejemplo similitud basado en el coseno (del ángulo formado entre los vectores) o distancia Euclideana. Por lo general no hay mucha diferencia entre usar una u otra de estas distancias.

La tarea de la clasificación en el modelo de espacio de vectores es determinar las fronteras entre los documentos pertenecientes a una u otra clase. Estas últimas son llamadas fronteras de decisión ya que dividen el espacio en diferentes poliedros, tales que si un documento pertenece a uno determinado, automáticamente sabemos de qué clase es.

En K Nearest Neighbor la frontera de decisión se determina localmente. En este asignamos cada documento a la misma clase que la mayoría de los  $k$  puntos más cercanos al documento. Basado en la hipótesis de contigüidad esperamos que el documento  $d$  pertenezca a la misma clase que aquellos más cercanos a él.

En  $kNN$  para subdividir el espacio de documentos en regiones, dado un  $k \in \mathbb{N}$  fijo, consideramos cada región como el conjunto de puntos para los cuales los  $k$  puntos más cercanos son los mismos. Estas regiones son poliedros convexos. Luego para cada una de estas regiones existe una clase a la que pertenecen todos sus puntos, que es aquella a la que pertenecen la mayoría de los documentos, ya clasificados, que están dentro de la región. En caso de que haya empate, la decisión de a qué clase asignar a un nuevo documento que pertenece a esta región del espacio, es tomada aleatoriamente entre las clases empatadas.

El parámetro  $k$  es usualmente seleccionado basado en el conocimiento que se posee sobre los problemas de clasificación similares al que se tiene. Otra forma de seleccionar  $k$  es usando conjuntos de documentos de prueba (ya clasificados) para ver que valor de  $k$  producen mejores resultados.

También hay variantes del método donde lo que se hace es calcular una similitud entre el documento  $d$  a clasificar y cada uno de los  $k$  más cercanos, usando, por ejemplo, el coseno entre los vectores. Luego se hace un ranking entre



las clases a las que pertenecen cada uno de los  $k$  puntos. Para ello se calcula para cada  $c$ , la suma de la similitud entre  $d$  y cada uno de los puntos que pertenecen a  $c$  y a la vez están entre los  $k$  mencionados. La siguiente función *score* produce el resultado deseado

$$score(c, d) = \sum_{d' \in S_k(d)} I_c(d') \cos(\vec{v}(d'), \vec{v}(d)),$$

donde  $S_k(d)$  es el conjunto de los  $k$  puntos más cercanos a  $d$  e  $I_c(d')$  es 1 o 0 en dependencia de si  $d'$  pertenece a la clase  $c$  o no. Finalmente se selecciona para  $d$  la clase  $c$  que más alto aparezca en el ranking. En ocasiones esta variante presenta mayor exactitud que la anterior.

---

**Algorithm 4** Train-kNN

---

**Require:** Set of classes  $\mathcal{C}$  and training set  $D$ .

- 1:  $D' \leftarrow PreProcess(D)$
  - 2:  $k \leftarrow Selectk(\mathcal{C}, D')$
  - 3: **return**  $D', k$
- 

---

**Algorithm 5** Apply-kNN

---

**Require:**  $\mathcal{C}, D', k, d$

- 1:  $S_k \leftarrow ComputeNearestNeighbors(D', k, d)$
  - 2: **for**  $c_j \in \mathcal{C}$  **do**
  - 3:      $p_j \leftarrow |S_k \cap c_j|/k$
  - 4: **return**  $\arg \max_j p_j$
- 

### 3.5. Evaluación de la clasificación

Una alta exactitud en la clasificación de los datos de entrenamiento no necesariamente se traduce en resultados correctos en los nuevos documentos introducidos. Puede suceder que el sistema resulte sobreentrenado (*over-fitting*) o subentrenado (*under-fitting*). El primero de los casos ocurre cuando el modelo se encuentra con muchos datos. En este caso el sistema aprende de los ruidos y de las entradas inexactas lo que provoca que el clasificador aprenda incorrectamente a clasificar los elementos de alguna clase. Por otro lado *under-fitting* ocurre cuando el sistema tiene información muy vaga sobre la frontera entre clases lo que provoca cierta aleatoriedad en la clasificación.

Para evaluar la labor de un clasificador se debe emplear un conjunto de documentos diferentes al conjunto entrenante el cual se llama conjunto de prueba. Este debe estar en todo momento aislado del conjunto de entrenamiento. Si se

juntan el clasificador puede aprender en su entrenamiento a reconocer los documentos del conjunto de prueba, y luego al evaluar el mismo, el desempeño del sistema será mucho más alto que el resultado real que tenga cuando se ponga en uso. Usualmente los datos que se tienen se dividen entre el conjunto entrenante y de prueba a razón de 4 : 1.

Las siguientes tres métricas pueden ser utilizadas para evaluar clasificadores de dos clases ( $c$  y  $\bar{c}$ ):

■ **Precisión**

$$Precision = \frac{\text{Número de documentos correctamente identificados como } c}{\text{Número de documentos identificados como } c}$$

■ **Recobrado**

$$recobrado = \frac{\text{Número de documentos correctamente identificados como } c}{\text{Número de documentos que pertenecen a } c}$$

■ **Medida  $F$**

$$F = 2 \times \frac{Precision \times Recobrado}{Precision + Recobrado}$$

El mismo resultado se puede ver usando la matriz de contingencia. Una matriz de contingencia muestra de todas las pruebas realizadas con el clasificador, cuántas resultaron en verdadero positivo, falso positivo, falso negativo y verdadero negativo.

En el caso que la cantidad de clases sea mayor que dos, podemos computar promedios entre las métricas anteriores viendo el clasificador como uno de dos clases aplicado a cada clase  $c$  (y su complemento). Existen dos promedios que son los más usados para esto. Macropromedio calcula simplemente la media entre las medidas anteriores para cada clasificador de dos clases. Microaverage primero adiciona las matrices de contingencia correspondientes a diferentes clases y luego aplica una de las métricas anteriores sobre la matriz resultante.

### 3.6. Ventajas y Desventajas

Ventajas:

- A diferencia de el aprendizaje no supervisado se puede tener una idea exacta sobre las clases de los objetos, dado que estas se crean basado en características externas de los mismos.
- La salida suele ser más precisa que en el ML supervisado
- El proceso es fácil de comprender (las acciones de la maquina se comprenden con precision a diferencia del no supervisado)
- es más facil de trabajar con el supervisado. (supongo que se refiera a más facil de implementar)

- Se evalúa y recibe retroalimentación para comprobar la correctitud.

Desventajas:

- Se necesita un especialista que determine cuáles serán las clases en que se desea clasificar
- Se requiere de un conjunto de datos entrenantes
- Por lo general no se resuelven tareas más complejas como en el ML supervisado
- Suelen requerir mucho tiempo de entrenamiento
- Puede predecir la salida incorrecta si los datos de prueba son diferentes de los datos de entrenamiento.

Cogido de:

-<https://www.crehana.com/blog/transformacion-digital/tipos-de-machine-learning/>  
 -<https://aprendeia.com/ventajas-y-desventajas-de-los-algoritmos-de-clasificacion-machine-learning/>

NOTA:

- Podría insertarse una foto de la matriz de contingencia(pagina 16 de classification.pdf)
- citas correctas

- Aprendizaje supervisado
- Problema que resuelve
- Rule-based classification
- Statistical classification
- Feature selection
- Medidas de evaluación:
  - Fitting
  - Precisión
  - Recobrado
  - Medida F (balanceada)
  - Classification accuracy
- Algoritmos:
  - Naive Bayes
  - K-Nearest Neighbours

### 3.7. Aplicaciones a la RI

- Standing queries
- Spam filtering

**3.8. Ventajas****3.9. Desventajas****4. Agrupamiento vs. Clasificación****5. Ejemplos de aplicación**

Ya hablamos en la epígrafe 3.1 acerca de la importancia de la clasificación en numerosos problemas de la sociedad. En especial nos centramos en ejemplificar algunos de los problemas específicos de la Recuperación de la Información en que es aplicable la clasificación. Trataremos en esta sección de ver la importancia de la clasificación en problemas más generales de la humanidad.

La bioinformática es una rama científica que ha adquirido un desarrollo impresionante en los últimos años. Esta se dedica al desarrollo de herramientas informáticas para aplicarlas en la gestión y análisis de datos biológicos. Algunas de las aplicaciones más importantes que tiene la clasificación, y en general, el aprendizaje automático, es precisamente en esta rama de la ciencia. Con el uso de sus algoritmos se permite automatizar la búsqueda de patrones en conjuntos de datos, que puedan ayudar a entender diferentes procesos biológicos que se manifiestan en los mismos. Debido a los grandes volúmenes de datos que se tienen ha crecido mucho en los últimos años la aplicación de la clasificación en esta disciplina.

Una de estas aplicaciones biológicas de las que hablabamos es la predicción de gen. Esta consiste en determinar que fracciones de una secuencia de ADN codifican proteínas. También se ve aplicado en la investigación de comunidades de microbios en nichos ecológicos. Así se puede obtener información taxonómica y metabólica de las comunidades estudiadas. Existen otras muchas aplicaciones en esta ciencia entre las que podemos mencionar la proteómica, los microarrays y la biología de sistemas.

Saliéndonos un tanto del marco de la bioinformática, pero no del todo lejos de la biología, encontramos algunas otras aplicaciones interesantes. Nos referimos al reconocimiento de algunos rasgos distintivos de una persona, como pueden ser, reconocimiento de rostros, huellas o voz. Este último lo tenemos al alcance de nuestras manos en asistentes virtuales como Siri o Google. Estas aplicaciones se entrenan para reconocer características distintivas de tu voz, mediante un proceso de aprendizaje supervisado, para luego poder diferenciar cuándo eres tú quien les está hablando.

El mercado financiero es una de los espacios de la vida del hombre donde más repercute la clasificación. Se puede emplear en mejorar el mercado digital, las ventas en línea, identificar el valor de vida útil de un cliente, la tasa de abandono, recomendaciones de productos y análisis de impacto de campañas de mercado.

En la seguridad informática también se evidencia la clasificación en acciones como detección de virus, enlaces maliciosos y fraude. Otras aplicaciones en que puede verse también son el Internet de las cosas (IoT), los motores de recomendación y la fijación de precios dinámicos de productos.

Leer: <https://futuroelectrico.com/aprendizaje-supervisado/>  
=====

## **6. Conclusiones**