Inteligencia Artificial

Aprendizaje automático

Departamento de Ciencias de la Computación
e Inteligencia Artificial

Universidad de Sevilla

Aprendizaje

- Definiciones de *aprendizaje*:
  - Cualquier cambio en un sistema que le permita realizar la misma tarea de manera *más eficiente* la próxima vez (*H. Simon*)
  - Modificar la representación del mundo que se está percibiendo (*R. Michalski*)
  - Realizar cambios útiles en nuestras mentes (*M. Minsky*)

Aprendizaje

- Aprendizaje automático: construir programas que mejoran automáticamente con la experiencia
- Ejemplos de tareas:
  - Construcción de bases de conocimiento a partir de la experiencia
  - Clasificación y diagnóstico
  - Minería de datos, descubrir estructuras desconocidas en grandes grupos de datos
  - Resolución de problemas, planificación y acción

Tipos de aprendizaje y paradigmas

- Tipos de aprendizaje
  - Supervisado
  - No supervisado
  - Con refuerzo
- Paradigmas
  - Aprendizaje por memorización
  - Clasificación (*Clustering*)
  - Aprendizaje inductivo
  - Aprendizaje por analogía
  - Descubrimiento
  - Algoritmos genéticos, redes neuronales

Ejemplo de aprendizaje

- Conjunto de entrenamiento
  - Ejemplos: días en los que es recomendable (o no) jugar al tenis
  - Representación como una lista de pares atributo–valor

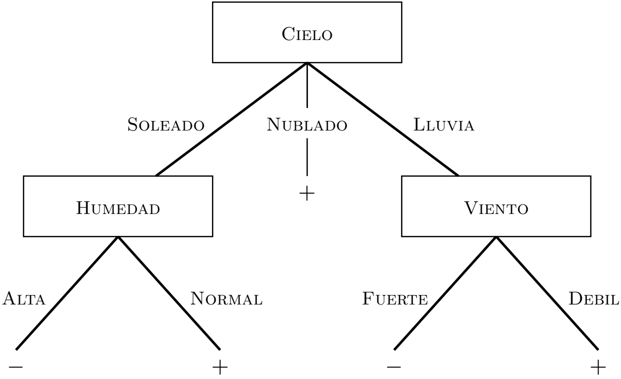
Ej.	Cielo	Temperatura	Humedad	Viento	JugarTenis
D_1	Soleado	Alta	Alta	Débil	-
D_2	Soleado	Alta	Alta	Fuerte	-
D_3	Nublado	Alta	Alta	Débil	+
D_4	Lluvia	Suave	Alta	Débil	+
...					

- Objetivo: Dado el *conjunto de entrenamiento*, *aprender* el concepto «Días en los que se juega al tenis»
  - Se trata de *aprendizaje supervisado*
- Problema: ¿Cómo expresar lo aprendido?
  - En este tema, veremos algoritmos para aprender *árboles de decisión*, *reglas*, *modelos probabilísticos*,...

Aprendizaje de árboles de decisión

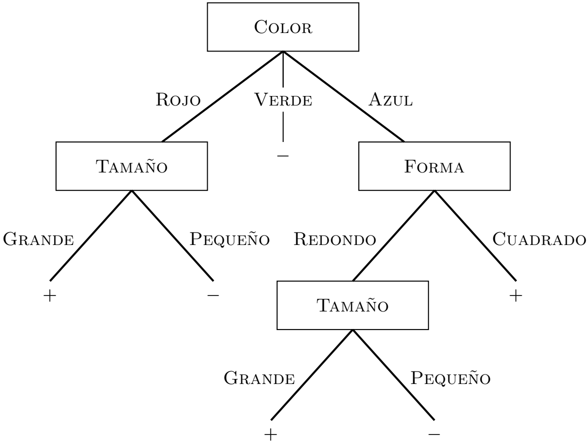
Árboles de decisión

- Ejemplos de árboles de decisión



Árboles de decisión

- Ejemplos de árboles de decisión



Árboles de decisión

- Árboles de decisión
  - Nodos interiores: atributos
  - Arcos: posibles valores del nodo origen
  - Hojas: valor de clasificación (usualmente $+$ ó $-$, aunque podría ser cualquier conjunto de valores, no necesariamente binario)
  - Representación de una función objetivo
- Disyunción de reglas proposicionales:

$$(\textsc{Cielo}{=}\textsc{Soleado} \wedge \textsc{Humedad}{=}\textsc{Alta} \rightarrow \textsc{JugarTenis}{=}-)$$
$$\vee\ (\textsc{Cielo}{=}\textsc{Soleado} \wedge \textsc{Humedad}{=}\textsc{Normal} \rightarrow \textsc{JugarTenis}{=}+)$$
$$\vee\ (\textsc{Cielo}{=}\textsc{Nublado} \rightarrow \textsc{JugarTenis}{=}+)$$
$$\vee\ (\textsc{Cielo}{=}\textsc{Lluvioso} \wedge \textsc{Viento}{=}\textsc{Fuerte} \rightarrow \textsc{JugarTenis}{=}-)$$
$$\vee\ (\textsc{Cielo}{=}\textsc{Lluvioso} \wedge \textsc{Viento}{=}\textsc{Debil} \rightarrow \textsc{JugarTenis}{=}+)$$

Aprendizaje de árboles de decisión

- Objetivo: aprender un árbol de decisión consistente con los ejemplos, para posteriormente clasificar ejemplos nuevos
- Ejemplo de conjunto de entrenamiento:

Ej.	Cielo	Temperatura	Humedad	Viento	JugarTenis
D_1	Soleado	Alta	Alta	Débil	-
D_2	Soleado	Alta	Alta	Fuerte	-
D_3	Nublado	Alta	Alta	Débil	+
D_4	Lluvia	Suave	Alta	Débil	+
...					

Algoritmo ID3

Algoritmo ID3

ID3(Ejemplos, Atributo-objetivo, Atributos)
1. Si todos los Ejemplos son positivos, devolver un nodo
   etiquetado con +
2. Si todos los Ejemplos son negativos, devolver un nodo
   etiquetado con -
3. Si Atributos está vacío, devolver un nodo etiquetado con el valor
   más frecuente de Atributo-objetivo en Ejemplos.
4. En otro caso:
4.1. Sea A el atributo de Atributos que MEJOR clasifica Ejemplos
4.2. Crear Árbol, con un nodo etiquetado con A.
4.3. Para cada posible valor v de A, hacer:
   * Añadir un arco a Árbol, etiquetado con v.
   * Sea Ejemplos(v) el subconjunto de Ejemplos con valor del
     atributo A igual a v.
   * Si Ejemplos(v) es vacío:
     - Entonces colocar debajo del arco anterior un nodo etiquetado
       con el valor más frecuente de Atributo-objetivo en Ejemplos.
     - Si no, colocar debajo del arco anterior el subárbol
       ID3(Ejemplos(v), Atributo-objetivo, Atributos-{A}).
4.4 Devolver Árbol

¿Cómo saber qué atributo clasifica mejor?

- Entropía de un conjunto de ejemplos D (resp. de una clasificación):
$$Ent(D) = -\frac{|P|}{|D|} \cdot log_2 \frac{|P|}{|D|} - \frac{|N|}{|D|} \cdot log_2 \frac{|N|}{|D|}$$
donde P y N son, respectivamente, los subconjuntos de ejemplos positivos y negativos de D
    - Notación: $Ent([p^+, n^-])$, donde $p = |P|$ y $n = |N|$
- Intuición:
    - Mide la ausencia de «homogeneidad» de la clasificación
    - Teoría de la Información: cantidad media de información (en bits) necesaria para codificar la clasificación de un ejemplo
- Ejemplos:
    - $Ent([9^+, 5^-]) = -\frac{9}{14} \cdot log_2 \frac{9}{14} - \frac{5}{14} \cdot log_2 \frac{5}{14} = 0.94$
    - $Ent([k^+, k^-]) = 1$ (ausencia total de homogeneidad)
    - $Ent([p^+, 0^-]) = Ent([0^+, n^-]) = 0$ (homogeneidad total)

Ganancia de información

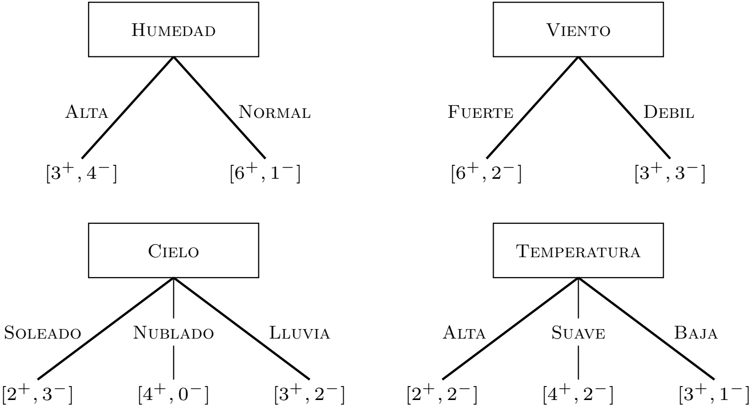
- Preferimos nodos con menos entropía (árboles pequeños)
- Entropía *esperada* después de usar un atributo A en el árbol:
  $$\sum_{v \in Valores(A)} \frac{|D_v|}{|D|} \cdot Ent(D_v)$$
  donde D_v es el subconjunto de ejemplos de D con valor del atributo A igual a v
- Ganancia de información *esperada* después de usar un atributo A:
  $$Ganancia(D, A) = Ent(D) - \sum_{v \in Valores(A)} \frac{|D_v|}{|D|} \cdot Ent(D_v)$$
- En el algoritmo ID3, en cada nodo usamos el atributo con mayor ganancia de información (considerando los ejemplos correspondientes al nodo)

Algoritmo ID3 (ejemplo 1)

- Conjunto de entrenamiento:

Ej.	Cielo	Temperatura	Humedad	Viento	JugarTenis
D_1	Soleado	Alta	Alta	Débil	-
D_2	Soleado	Alta	Alta	Fuerte	-
D_3	Nublado	Alta	Alta	Débil	+
D_4	Lluvia	Suave	Alta	Débil	+
D_5	Lluvia	Baja	Normal	Débil	+
D_6	Lluvia	Baja	Normal	Fuerte	-
D_7	Nublado	Baja	Normal	Fuerte	+
D_8	Soleado	Suave	Alta	Débil	-
D_9	Soleado	Baja	Normal	Débil	+
D_{10}	Lluvia	Suave	Normal	Débil	+
D_{11}	Soleado	Suave	Normal	Fuerte	+
D_{12}	Nublado	Suave	Alta	Fuerte	+
D_{13}	Nublado	Alta	Normal	Débil	+
D_{14}	Lluvia	Suave	Alta	Fuerte	-

Algoritmo ID3 (ejemplo 1)

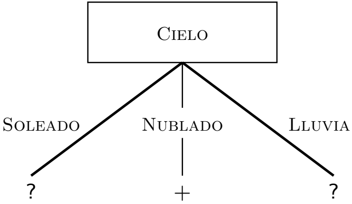


Algoritmo ID3 (ejemplo 1)

- Entropía inicial: $Ent([9^+, 5^-]) = 0.94$
- Selección del atributo para el nodo raíz:
  - $Ganancia(D,\textsc{Humedad}) =$
    $0.94 - \frac{7}{14} \cdot Ent([3^+, 4^-]) - \frac{7}{14} \cdot Ent([6^+, 1^-]) = 0.151$
  - $Ganancia(D,\textsc{Viento}) =$
    $0.94 - \frac{8}{14} \cdot Ent([6^+, 2^-]) - \frac{6}{14} \cdot Ent([3^+, 3^-]) = 0.048$
  - $Ganancia(D,\textsc{Cielo}) =$
    $0.94 - \frac{5}{14} \cdot Ent([2^+, 3^-]) - \frac{4}{14} \cdot Ent([4^+, 0^-])$
    $- \frac{5}{14} \cdot Ent([3^+, 2^-]) = 0.246$ (mejor atributo)
  - $Ganancia(D,\textsc{Temperatura}) =$
    $0.94 - \frac{4}{14} \cdot Ent([2^+, 2^-]) - \frac{6}{14} \cdot Ent([4^+, 2^-])$
    $- \frac{4}{14} \cdot Ent([3^+, 1^-]) = 0.02$
- El atributo seleccionado es $\textsc{Cielo}$

Algoritmo ID3 (ejemplo 1)

- Árbol parcialmente construido:



Algoritmo ID3 (ejemplo 1)

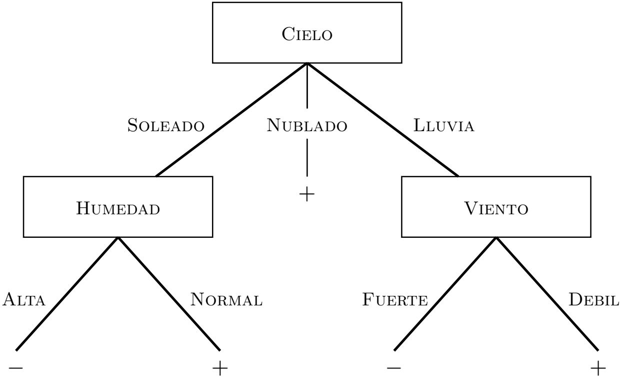
- Selección del atributo para el nodo $\text{Cielo}=\text{Soleado}$
- $D_{\text{Soleado}} = \{D_1, D_2, D_8, D_9, D_{11}\}$ con entropía $Ent([2^+, 3^-]) = 0.971$
  - $Ganancia(D_{\text{Soleado}}, \text{Humedad}) = 0.971 - \frac{3}{5} \cdot 0 - \frac{2}{5} \cdot 0 = 0.971$ (mejor atributo)
  - $Ganancia(D_{\text{Soleado}}, \text{Temperatura}) = 0.971 - \frac{2}{5} \cdot 0 - \frac{2}{5} \cdot 1 - \frac{1}{5} \cdot 0 = 0.570$
  - $Ganancia(D_{\text{Soleado}}, \text{Viento}) = 0.971 - \frac{2}{5} \cdot 1 - \frac{3}{5} \cdot 0.918 = 0.019$
- El atributo seleccionado es Humedad

Algoritmo ID3 (ejemplo 1)

- Selección del atributo para el nodo $\textsc{Cielo}$=$\textsc{Lluvia}$:
- $D_{\textsc{Lluvia}} = \{D_4, D_5, D_6, D_{10}, D_{14}\}$ con entropía $Ent([3^+, 2^-]) = 0.971$
  - $Ganancia(D_{\textsc{Lluvia}}, \textsc{Humedad}) = 0.971 - \frac{2}{5} \cdot 1 - \frac{3}{5} \cdot 0.918 = 0.820$
  - $Ganancia(D_{\textsc{Lluvia}}, \textsc{Temperatura}) = 0.971 - \frac{3}{5} \cdot 0.918 - \frac{2}{5} \cdot 1 = 0.820$
  - $Ganancia(D_{\textsc{Lluvia}}, \textsc{Viento}) = 0.971 - \frac{3}{5} \cdot 0 - \frac{2}{5} \cdot 0 = 0.971$ (mejor atributo)
- El atributo seleccionado es $\textsc{Viento}$

Algoritmo ID3 (ejemplo 1)

- Árbol finalmente aprendido:



Algoritmo ID3 (ejemplo 2)

- Conjunto de entrenamiento:

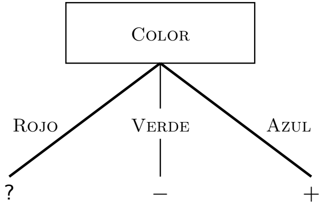
Ej.	Color	Forma	Tamaño	Clase
O_1	Rojo	Cuadrado	Grande	+
O_2	Azul	Cuadrado	Grande	+
O_3	Rojo	Redondo	Pequeño	-
O_4	Verde	Cuadrado	Pequeño	-
O_5	Rojo	Redondo	Grande	+
O_6	Verde	Cuadrado	Grande	-

Algoritmo ID3 (ejemplo 2)

- Entropía inicial en el ejemplo de los objetos, $Ent([3^+, 3^-]) = 1$
- Selección del atributo para el nodo raíz:
  - $Ganancia(D,\text{Color}) =$
    $1-\frac{3}{6}\cdot Ent([2^+, 1^-])-\frac{1}{6}\cdot Ent([1^+, 0^-])-\frac{2}{6}\cdot Ent([0^+, 2^-]) = 0.543$
  - $Ganancia(D,\text{Forma}) =$
    $1 - \frac{4}{6} \cdot Ent([2^+, 2^-]) - \frac{2}{6} \cdot Ent([1^+, 1^-]) = 0$
  - $Ganancia(D,\text{Tamaño}) =$
    $1 - \frac{4}{6} \cdot Ent([3^+, 1^-]) - \frac{2}{6} \cdot Ent([0^+, 2^-]) = 0.459$
- El atributo seleccionado es Color

Algoritmo ID3 (ejemplo 2)

- Árbol parcialmente construido:

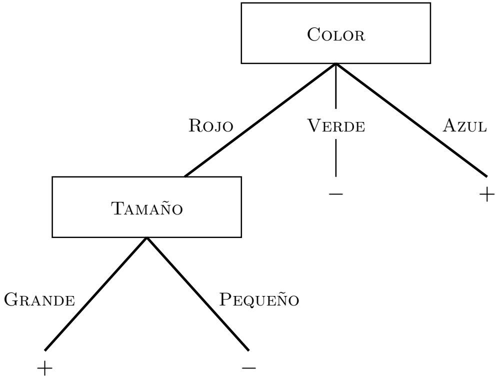


Algoritmo ID3 (ejemplo 2)

- Selección del atributo para el nodo $\textsc{Color}{=}\textsc{Rojo}$:
- $D_{\textsc{Rojo}} = \{O_1, O_3, O_5\}$ con entropía $Ent([2^+, 1^-]) = 0.914$
  - $Ganancia(D_{\textsc{Rojo}},\textsc{Forma}) =$
    $0.914 - \frac{1}{3} \cdot Ent([1^+, 0^-]) - \frac{2}{3} \cdot Ent([1^+, 1^-]) = 0.247$
  - $Ganancia(D_{\textsc{Rojo}},\textsc{Tamaño}) =$
    $0.914 - \frac{2}{3} \cdot Ent([2^+, 0^-]) - \frac{1}{3} \cdot Ent([0^+, 1^-]) = 0.914$
- El atributo seleccionado es $\textsc{Tamaño}$

Algoritmo ID3 (ejemplo 2)

- Árbol finalmente aprendido:



Búsqueda y sesgo inductivo

- Búsqueda local en un espacio de hipótesis
  - Espacio de todos los árboles de decisión
  - Un único árbol candidato en cada paso
  - Sin retroceso (peligro de óptimos locales), búsqueda en escalada
  - Decisiones tomadas a partir de conjuntos de ejemplos
- Sesgo inductivo
  - Se prefieren árboles más cortos sobre los más largos
  - Sesgo preferencial, implícito en la búsqueda
  - Principio de la navaja de Occam

Algunas cuestiones prácticas a resolver en aprendizaje automático

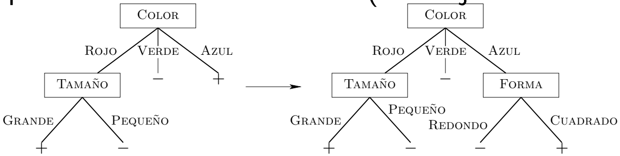
- Validar la hipótesis aprendida
  - ¿Podemos *cuantificar* la bondad de lo aprendido respecto de la explicación real?
- Sobreajuste
  - ¿Se ajusta *demasiado* lo aprendido al conjunto de entrenamiento?

Medida del rendimiento del aprendizaje

- Conjuntos de entrenamiento y prueba (*test*)
  - Aprender con el conjunto de entrenamiento
  - Medir el rendimiento en el conjunto de prueba:
    - proporción de ejemplos bien clasificados en el conjunto de prueba
- Repetición de este proceso
  - Curva de aprendizaje
  - Estratificación: cada clase correctamente representada en el entrenamiento y en la prueba
- Si no tenemos suficientes ejemplos como para apartar un conjunto de prueba: validación cruzada
  - Dividir en k partes, y hace k aprendizajes, cada uno de ellos tomando como prueba una de las partes y entrenamiento el resto. Finalmente hacer la media de los rendimientos.
  - En la práctica: validación cruzada, con $k = 10$ y estratificación

Sobreajuste y ruido

- Una hipótesis $h \in H$ *sobreajusta* los ejemplos de entrenamiento si existe $h' \in H$ que se ajusta peor que h a los ejemplos pero actúa mejor sobre la distribución completa de instancias.
- *Ruido:* ejemplos incorrectamente clasificados. Causa sobreajuste
- Ejemplo: supongamos que, por error, se incluye el ejemplo $<$AZUL,REDONDO,PEQUEÑO$>$ como ejemplo negativo
- El árbol aprendido en este caso sería (sobreajustado a los datos):



Sobreajuste y ruido

- Otras causas de sobreajuste:
  - Atributos que en los ejemplos presentan una aparente regularidad pero que no son relevantes en realidad
  - Conjuntos de entrenamiento pequeños
- Maneras de evitar el sobreajuste en árboles de decisión:
  - Parar el desarrollo del árbol antes de que se ajuste perfectamente a todos los datos
  - Podar el árbol *a posteriori*
- Poda *a posteriori*, dos aproximaciones:
  - Transformación a reglas, podado de las condiciones de las reglas
  - Realizar podas directamente en el árbol
  - Las podas se producen siempre que reduzcan el error sobre un conjunto de prueba

Podado de árboles

Algoritmo de poda para reducir el error

```
1. Dividir el conjunto de ejemplos en Entrenamiento y Prueba
2. Árbol=árbol obtenido por ID3 usando Entrenamiento
3. Continuar=True
4. Mientras Continuar:
 * Medida = proporción de ejemplos en Prueba correctamente
   clasificados por Árbol
 * Por cada nodo interior N de Árbol:
   - Podar temporalmente Árbol en el nodo N y sustituirlo por una
     hoja etiquetada con la clasificación mayoritaria en ese nodo
   - Medir la proporción de ejemplos correctamente clasificados en
     el conjunto de prueba.
 * Sea K el nodo cuya poda produce mejor rendimiento
 * Si este rendimiento es mejor que Medida, entonces
   Árbol = resultado de podar permanentemente Árbol en K
 * Si no, Continuar=Falso
5. Devolver Árbol
```

Otras cuestiones prácticas del algoritmo ID3

- Extensiones del algoritmo:
  - Atributos con valores contínuos
  - Otras medidas para seleccionar atributos
  - Otras estimaciones de error
  - Atributos sin valores
  - Atributos con coste
- Algoritmos C4.5 y C5.0 (Quinlan)

Clasificación mediante modelos probabilísticos: *naive* Bayes

Clasificadores *naive* Bayes

- Supongamos un conjunto de atributos $A_1, \ldots, A_n$ cuyos valores determinan un valor en un conjunto finito V de posibles «clases» o «categorías»
- Tenemos un conjunto de entrenamiento D con una serie de tuplas de valores concretos para los atributos, junto con su clasificación
- Queremos aprender un clasificador tal que clasifique nuevas instancias $\langle a_1, \ldots, a_n \rangle$
  - Es decir, el mismo problema en el tema de aprendizaje de árboles de decisión (pero ahora lo abordaremos desde una perspectiva probabilística).

Clasificadores *naive* Bayes

- Podemos diseñar un modelo probabilístico para un problema de clasificación de este tipo, tomando los atributos y la clasificación como variables aleatorias
- El valor de clasificación asignado a una nueva instancia $\langle a_1, \dots, a_n \rangle$, notado v_{MAP} vendrá dado por

$$\underset{v_j \in V}{argmax}\, P(v_j | a_1, \dots, a_n)$$

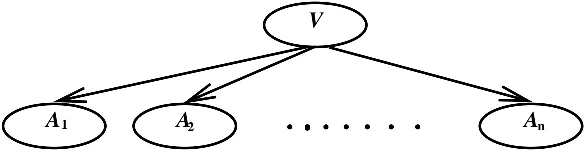
- Aplicando el teorema de Bayes podemos escribir

$$v_{MAP} = \underset{v_j \in V}{argmax}\, P(a_1, \dots, a_n | v_j) P(v_j)$$

- Y ahora, simplemente estimar las probabilidades de la fórmula anterior a partir del conjunto de entrenamiento
- Problema: necesitaríamos una gran cantidad de datos para estimar adecuadamente las probabilidades $P(a_1, \dots, a_n | v_j)$

Clasificadores *naive* Bayes

- Podemos simplificar el aprendizaje suponiendo que los atributos son (mútuamente) condicionalmente independientes dado el valor de clasificación (de ahí lo de «naive»)
- La situación se representa entonces por la red:



- En ese caso, tomamos como valor de clasificación:

$$v_{NB} = \underset{v_j \in V}{argmax}\, P(v_j) \prod_i P(a_i|v_j)$$

Estimación de probabilidades *naive* Bayes

- Para el proceso de aprendizaje, sólo tenemos que estimar las probabilidades $P(v_j)$ (probabilidades *a priori*) y $P(a_i|v_j)$ (probabilidades *condicionadas*). Son muchas menos que en el caso general.
- Mediante cálculo de sus frecuencias en el conjunto de entrenamiento, obtenemos estimaciones de *máxima verosimilitud* de esas probabilidades:

$$P(v_j) = \frac{\#(V = v_j)}{N} \qquad P(a_i|v_j) = \frac{\#(A_i = a_i, V = v_j)}{\#(V = v_j)}$$

donde N es el número total de ejemplos, $\#(V = v_j)$ es el número de ejemplos clasificados como v_j y $\#(A_i = a_i, V = v_j)$ es el número de ejemplos clasificados como v_j cuyo valor en el atributo A_i es a_i.

Clasificador *naive* Bayes: un ejemplo

- Vamos a aplicar el clasificador a un ejemplo ya conocido, usado en el tema de árboles de decisión:

Ej.	Cielo	Temperatura	Humedad	Viento	JugarTenis
D_1	Soleado	Alta	Alta	Débil	-
D_2	Soleado	Alta	Alta	Fuerte	-
D_3	Nublado	Alta	Alta	Débil	+
D_4	Lluvia	Suave	Alta	Débil	+
D_5	Lluvia	Baja	Normal	Débil	+
D_6	Lluvia	Baja	Normal	Fuerte	-
D_7	Nublado	Baja	Normal	Fuerte	+
D_8	Soleado	Suave	Alta	Débil	-
D_9	Soleado	Baja	Normal	Débil	+
D_{10}	Lluvia	Suave	Normal	Débil	+
D_{11}	Soleado	Suave	Normal	Fuerte	+
D_{12}	Nublado	Suave	Alta	Fuerte	+
D_{13}	Nublado	Alta	Normal	Débil	+
D_{14}	Lluvia	Suave	Alta	Fuerte	-

Clasificador *naive* Bayes: un ejemplo

- Supongamos que queremos predecir si un día soleado, de temperatura suave, humedad alta y viento fuerte es bueno para jugar al tenis
- Según el clasificador Naive Bayes:

$$v_{NB} = \underset{v_j \in \{+,-\}}{argmax}\, P(v_j)P(soleado|v_j)P(suave|v_j)P(alta|v_j)P(fuerte|v_j)$$

- Así que necesitamos estimar todas estas probabilidades, lo que hacemos simplemente calculando frecuencias en la tabla anterior:
  - $p(+) = 9/14$, $p(-) = 5/14$, $p(soleado|+) = 2/9$, $p(soleado|-) = 3/5$, $p(suave|+) = 4/9$, $p(suave|-) = 2/5$, $p(alta|+) = 3/9$, $p(alta|-) = 4/5$, $p(fuerte|+) = 3/9$ y $p(fuerte|-) = 3/5$

Clasificador *naive* Bayes: un ejemplo

- Por tanto, las dos probabilidades a posteriori son:
  - $P(+)P(soleado|+)P(suave|+)P(alta|+)P(fuerte|+) = 9/14 \cdot 2/9 \cdot 4/9 \cdot 3/9 \cdot 3/9 = 0.007$
  - $P(-)P(soleado|-)P(suave|-)P(alta|-)P(fuerte|-) = 5/14 \cdot 3/5 \cdot 2/5 \cdot 4/5 \cdot 3/5 = 0.0411$
- Así que el clasificador devuelve la clasificación con mayor probabilidad a posteriori, en este caso la respuesta es $-$ (no es un día bueno para jugar al tenis)

Detalles técnicos sobre las estimaciones

- Si alguna de las probabilidades es cercana a 0 y tenemos pocos ejemplos en el conjunto de entrenamiento, lo más seguro es que la estimación de esa probabilidad sea 0
- Esto plantea dos problemas:
  - La inexactitud de la propia estimación
  - Afecta enormemente a la clasificación que se calcule, ya que se multiplican las probabilidades estimadas y por tanto si una de ellas es 0, anula a las demás
- Una primera manera de abordar el problema: usar logaritmos de las probabilidades.
  - Los productos se transforman en sumas

$$v_{NB} = \underset{v_j \in V}{argmax}[log(P(v_j)) + \sum_i log(P(a_i|v_j))]$$

Suavizado

- Problema en la estimaciones:
  - Probabilidades nulas o casi nulas, por ausencia en el conjunto de entrenamiento de algunos valores de atributos en algunas categorías
  - Sobreajuste
- Idea: *suponer* que tenemos m ejemplos adicionales, cuyos valores se distribuyen teóricamente *a priori* de alguna manera.
- Estimación *suavizada* de una probabilidad, a partir de observaciones: $\frac{n'+m\cdot p}{n+m}$
  - n' y n: número de ejemplos favorables y totales observados
  - p: estimación *a priori* de la probabilidad que se quiere estimar.
  - m: *tamaño de muestreo equivalente*, indica el número de ejemplos adicionales (ficticios)

Suavizado aditivo (o de Laplace)

- Un caso particular de lo anterior se suele usar para la estimación de las probabilidades condicionales en *naive* Bayes:

$$P(a_i|v_j) = \frac{\#(A_i = a_i, V = v_j) + k}{\#(V = v_j) + k|A_i|}$$

  donde k es un número fijado y $|A_i|$ es el número de posibles valores del atributo $|A_i|$.
- Intuitivamente: se supone que además de los del conjunto de entrenamiento, hay k ejemplos en la clase v_j por cada posible valor del atributo A_i
- Usualmente $k = 1$, pero podrían tomarse otros valores
  - Elección de k: experimentando con los distintos rendimientos sobre un *conjunto de validación*

Aprendizaje basado en instancias: kNN

Clasificación mediante vecino más cercano

- Una técnica alternativa a construir el modelo probabilístico es calcular la clasificación directamente a partir de los ejemplos (*aprendizaje basado en instancias*)
- Idea: obtener la clasificación de un nuevo ejemplo a partir de las categorías de los ejemplos más «cercanos».
  - Debemos manejar, por tanto, una noción de «distancia» entre ejemplos.
  - En la mayoría de los casos, los ejemplos serán elementos de R^n y la distancia, la euclídea.
  - Pero se podría usar otra noción de distancia
- Ejemplo de aplicación: clasificación de documentos

El algoritmo k-NN

- El algoritmo k-NN (de k *nearest neighbors*):
  - Dado un conjunto de entrenamiento (vectores numéricos con una categoría asignada) y un ejemplo nuevo
  - Devolver la categoría mayoritaria en los k ejemplos del conjunto de entrenamiento más cercanos al ejemplo que se quiere clasificar

Distancias para k-NN

- Posibles distancias usadas para definir la «cercanía»:
  - Euclídea: $d_e(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$
  - Manhattan: $d_m(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{n}|x_i - y_i|$
  - Hamming: número de componentes en las que se difiere.
- La euclídea se usa cuando cada dimensión mide propiedades similares y la Mahattan en caso contrario; la distancia Hamming se puede usar aún cuando los vectores no sean numéricos.
- Normalización: cuando no todas las dimensiones son del mismo orden de magnitud, se normalizan las componentes (restando la media y dividiendo por la desviación típica)

Algunas observaciones sobre k-NN

- Elección de k:
  - Usualmente, basándonos en algún conocimiento específico sobre el problema de clasificación
  - También como resultado de pruebas en conjuntos más pequeños (conjuntos de validación)
  - Si la clasificación es binaria, preferiblemente impar, para intentar evitar empates (k=5, por ejemplo)
- Variante en kNN: para cada clase c, sumar la similitud (con el que se quiere clasificar) de cada ejemplo de esa clase que esté entre los k más cercanos. Devolver la clase que obtenga mayor puntuación.
  - Así un ejemplo cuenta más cuanto más cercano esté

Clustering

Clustering

- Como última aplicación del aprendizaje estadístico, trataremos técnicas de *agrupamiento* o *clustering*
- Se trata de dividir un conjunto de datos de entrada en subconjuntos (*clusters*), de tal manera que los elementos de cada subconjunto compartan cierto patrón o características a priori desconocidas
- En nuestro caso, los datos serán números o vectores de números y *el número de clusters nos vendrá dado*
- Aprendizaje *no supervisado*: no tenemos información sobre qué cluster corresponde a cada dato.
- Aplicaciones de clustering:
  - Minería de datos
  - Procesamiento de imágenes digitales
  - Bioinformática

Dos ejemplos

- *Color quantization*:
  - Una imagen digital almacenada con 24 bits/pixel (aprox. 16 millones de colores) se tiene que mostrar sobre una pantalla que sólo tiene 8 bits/pixel (256 colores)
  - ¿Cuál es la mejor correspondencia entre los colores de la imagen original y los colores que pueden ser mostrados en la pantalla?
- Mezcla de distribuciones:
  - Tenemos una serie de datos con el peso de personas de un pais; no tenemos información sobre si el peso viene de un varón o de una mujer, pero sabemos que la distribución de pesos es de tipo normal, y que en los hombres es distinta que en las mujeres
  - Atendiendo a los datos, ¿podemos *aprender* de qué dos distribuciones de probabilidad vienen?

Clustering basado en distancia

- Idea: dado el número k de grupos o *clusters*, buscar k puntos o *centros* representantes de cada cluster, de manera que cada dato se considera en el cluster correspondiente al centro que tiene a menor «distancia»
- Como antes, la distancia sería específica de cada problema:
  - Expresará la medida de similitud
  - La distancia más usada es la euclídea

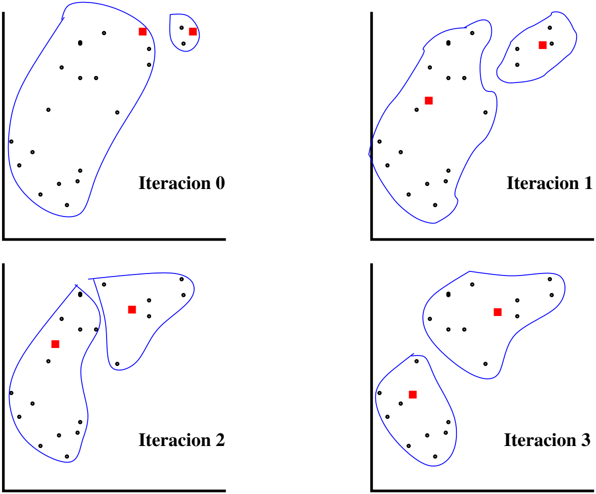
Un algoritmo clásico: k-medias

- Entrada: un número k de clusters, un conjunto de datos $\{x_i\}_{i=1}^N$ y una función de distancia
- Salida: un conjunto de k centros $m_1, \ldots, m_k$

k-medias(k,datos,distancia)

```
1. Inicializar m_i (i=1,...,k) (aleatoriamente o con algún
   criterio heurístico)
2. REPETIR (hasta que los m_i no cambien):
   2.1 PARA j=1,...,N, HACER:
       Calcular el cluster correspondiente a x_j, escogiendo,
       de entre todos los m_i, el m_h tal que
       distancia(x_j,m_h) sea mínima
   2.2 PARA i=1,...,k HACER:
       Asignar a m_i la media aritmética de los datos
       asignados al cluster i-ésimo
3. Devolver m_1,...,m_n
```

Idea gráfica intuitiva en el algoritmo de k-medias



Ejemplo en el algoritmo k-medias

- Datos sobre pesos de la población: 51, 43, 62, 64, 45, 42, 46, 45, 45, 62, 47, 52, 64, 51, 65, 48, 49, 46, 64, 51, 52, 62, 49, 48, 62, 43, 40, 48, 64, 51, 63, 43, 65, 66, 65, 46, 39, 62, 64, 52, 63, 64, 48, 64, 48, 51, 48, 64, 42, 48, 41
- El algoritmo, aplicado con $k = 2$ y distancia euclídea, encuentra dos centros $m_1 = 63.63$ y $m_2 = 46.81$ en tres iteraciones
- 19 datos pertenecen al primer cluster y 32 al segundo cluster

Cuestiones sobre el algoritmo k-medias

- Puede verse como un algoritmo de búsqueda local: encontrar los centros m_i que optimizan

  $$\Sigma_j \Sigma_i b_{ij} d(x_j, m_i)^2$$

  donde b_{ij} vale 1 si x_j tiene a m_i como el centro más cercano, 0 en otro caso.
  No garantiza encontrar el óptimo global.
- Inicialización: aleatoria o con alguna técnica heurística (por ejemplo, partir los datos aleatoriamente en k clusters y empezar con los centros de esos clusters)
- En la práctica, los centros con los que se inicie el algoritmo tienen un gran impacto en la calidad de los resultados que se obtengan
- Clusters vacíos: elegir un nuevo centro de manera aleatoria o usando alguna técnica heurística

Bibliografía

- Mitchell, T.M. *Machine Learning* (McGraw-Hill, 1997)
  - Caps. 3,6,8 y 10
- Russell, S. y Norvig, P. *Artificial Intelligence (A Modern Approach)* (3rd edition) (Prentice Hall, 2010)
  - Seccs. 18.1, 18.2, 18.3, 20.1 y 20.2
- Russell, S. y Norvig, P. *Inteligencia Artificial (Un enfoque moderno)* (segunda edición en español) (Pearson Education, 2004)
  - Seccs. 18.1, 18.2, 18.3, 18.8, 20.1, 20.2 y 20.4
- Witten, I.H. y Frank, E. *Data mining* (Second edition) (Morgan Kaufmann Publishers, 2005)
  - Cap. 3, 4, 5 y 6.