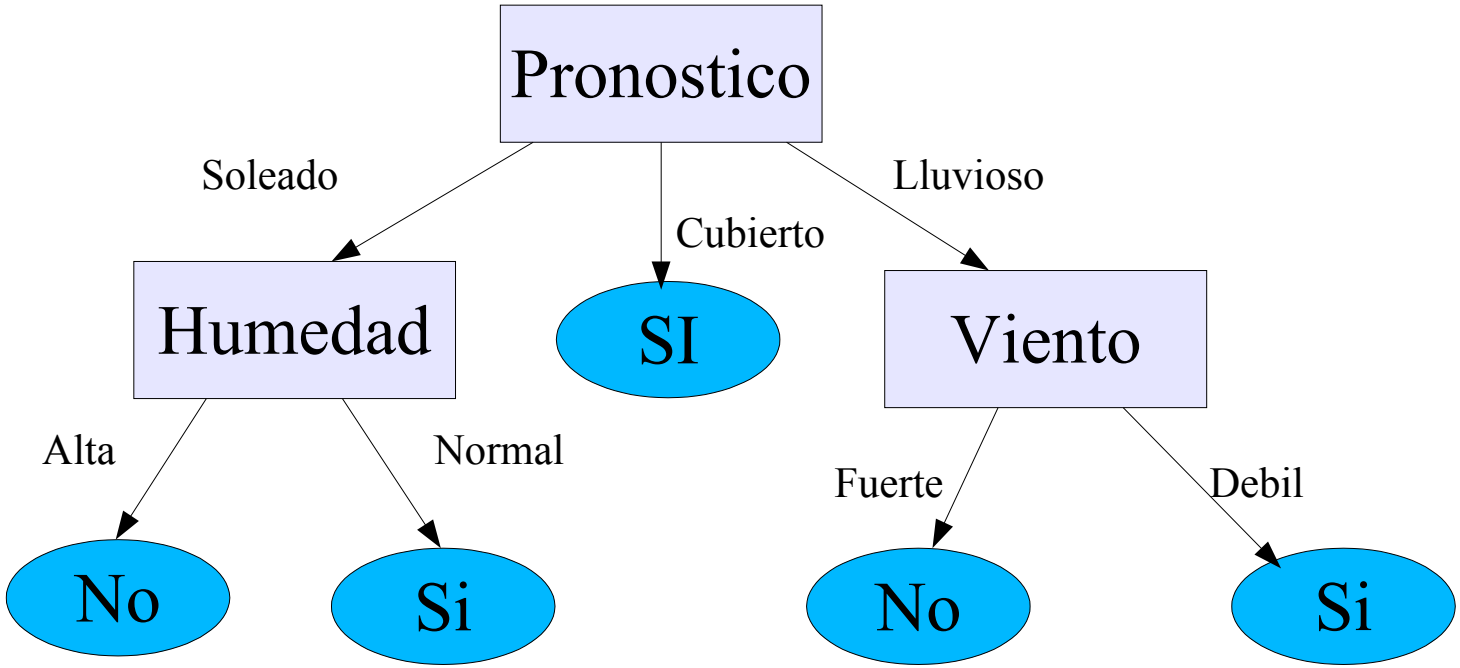
Aprendizaje automático mediante árboles de decisión

Aprendizaje por inducción

- Los árboles de decisión son uno de los métodos de aprendizaje inductivo más usado.
  - **Hipótesis de aprendizaje inductivo**: *cualquier hipótesis encontrada que clasifique un número suficientemente grande de ejemplos de entrenamiento clasificará otros ejemplos no observados.*
  - **Razonamiento deductivo**: partiendo de unas premisas se llega necesariamente a una conclusión. No aporta información nueva.
  - **Razonamiento abductivo**: partiendo del conocimiento de unos efectos (síntomas) se llega a la causa (enfermedad)
- Se trata de aproximar una función desconocida a patir de ejemplos positivos y negativos de esa función. Esos ejemplos serán en realidad pares <x,f(x)>, donde x es el valor de entrada y f(x) el valor de la función aplicada a x.
- Dado un conjunto de ejemplos de f, la inducción consiste en obtener una función h que aproxime f. A esta función h se la denomina hipótesis

Arboles de Decisión

- **Pueden ser leídas como conjunto de reglas (en el caso de abajo tres)**
- **En un árbol de decisión cada nodo del árbol es un atributo (campo) de los ejemplos, y cada rama representa un posible valor de ese atributo**



Arboles de Decisión

- **Problemas de clasificación apropiados**
  - Los ejemplos están representados por pares <atributo,valor>. Por ejemplo <Temperatura,alta> A ser posible los valores deben ser disjuntos <frío, caliente, templado>. Aún así se pueden manejar valores numéricos <Temperatura, [1º, 20º]
  - La función de clasificación tiene valores de salida discretos, y a ser posible booleanos.
  - Los patrones de ejemplo pueden contener errores, tanto por ruido presente en los ejemplos (errores en los valores) como errores en la clasificación de los ejemplos del conjunto de entrenamiento (recordemos que es un aprendizaje supervisado)
  - Los ejemplos pueden contener atributos sin valor

Algoritmo ID3

- Propuesto por Quinlan en 1986
- Empieza por responder a la cuastión: ¿qué atributo debe ser la raíz del árbol?
- Para ello se evalúa cada atributo usando un test estadístico para determinar cuán bien clasifica esos ejemplos (en realidad se determina el más representativo o el que mejor describe ese conjunto)

Algoritmo ID3

- Las medidas que se usan son:
  - **Entropía**: introducida por Shannon en su teoría de la información

$$S(E) = info(E) = -\sum_{(j=1)}^{(k)} p_j \log_2 p_j$$

siendo p_j la proporción de ejemplos de clase C_j en el conjunto E

  - **Efectividad** de un atributo para subdividir un conjunto de ejemplos en n subconjuntos (uno por cada posible valor de X): es el valor esperado de la entropía tras efectuar la partición,y se cacula como una suma ponderada de cada subconjunto E_i

$$S(E, X) = info(E, X) = \sum_{(i=1)}^{(n)} \frac{|E_i|}{|E|} * info(E_i)$$

Algoritmo ID3

- **Ganancia de información**: propiedead estadística que mide cómo clasifica ese atributo a los ejemplos.

$$ganancia(E, X) = info(E) - info_{atrib}(E, X)$$

  - Elijo como nodo del árbol aquél que tenga mayor ganancia de información.
  - Después expando sus ramas y sigo igual, pero considerando la nuava partición formada por el subconjunto de ejemplos que tienen ese valor para el atributo elegido.

Algoritmo ID3

- Interpretación de los parámetros
  - Entropia: nº de bits necesarios para codificar un suceso. Cuanto más bits más información menos probable es un suceso. Es decir, al aparecer más cuando aparece aporta menos información al conjunto que cuando aparece un suceso más raro.
  - Ganancia: Información del conjunto menos la que aporta el atributo X. Cuanto mayor sea menor es la cantidad de información que aporta X, es decir, es un suceso muy probable lo que implica que sea un buen candidato como atributo representativo del conjunto.

Ejemplo del algoritmo ID3

Pronóstico	Temperatura	Humedad	Viento	¿Adecuado?
Soleado	Alta	Alta	Flojo	No
Soleado	Alta	Alta	Fuerte	No
Nublado	Alta	Alta	Flojo	Si
Lluvia	Moderada	Alta	Flojo	Si
Lluvia	Baja	Normal	Flojo	Si
Lluvia	Baja	Normal	Fuerte	No
Nublado	Baja	Normal	Fuerte	Si
Soleado	Moderada	Alta	Flojo	No
Soleado	Baja	Normal	Flojo	Si
Lluvia	Moderada	Normal	Flojo	Si
Soleado	Moderada	Normal	Fuerte	Si
Nublado	Moderada	Alta	Fuerte	Si
Nublado	Alta	Normal	Flojo	Si
Soleado	Moderada	Alta	Fuerte	No

Mejoras del ID3

- **Mejoras de ID3** (1)
  - Manejo de atributos de gran número de valores.
  - Manejo de atributos con valores continuos.
  - Manejo de valores desconocidos en algunos de los Atributos.
  - El coste de conocer el valor de un atributo no es cte.:
    - Ejemplo: Presión arterial v.s. Biopsia.
  - Cuándo se debe parar de subdividir el árbol:
    - Sobreajuste v.s. Poda.
  - Los ejemplos vienen dados incrementalmente:
    - Solución: Algoritmos incrementales ID4 e ID5.
      - (1) La solución a muchas de estas cuestiones la incorpora el algoritmo C4.5 que constituye una extensión del algoritmo ID3.

Algoritmo C4.5

- Modificación propuesta por Quinlan para mejorar el algoritmo ID3
- Se basa en introducir una medida alternativa, el **ratio de ganancia,** definido por:

$$ratio(E, X) = \frac{ganancia(E, X)}{info_{part}(E, X)}$$

Donde

$$info_{part}(E, X) = -\sum_{(i=1)}^{n} \frac{|E_i|}{E} * \log_2 \frac{|E_i|}{E}$$

- **El sistema C4.5 selecciona en cada nodo el atributo con mayor ratio de ganancia de información**

Manejo de atributos con valores continuos

- Se ordenan los valores del atributo y se especifica la clase a la que pertenecen:
  - Temperatura 40 48 60 72 80 90
  - Adecuado No No Si Si Si No
- El objetivo es definir atributos discretos dividiendo el atributo continuo en intervalos, y dando valores booleanos.
- La división del intervalo se realiza basándose en un punto umbral, dividiéndolo así en 2 intervalos.

Manejo de atributos con valores continuos

- Otras posibilidades:
  - Dividir en tantos subintervalos como puntos de corte tengamos.
    - T<54, 54<T<85, T>85

Manejo de atributos con valores continuos

- Candidatos a valor umbral: valores para los cuales la clase cambia:
  - (48+60)/2=54 -> T>54
  - (80+90)/2=85 -> T>85
- Computo la ganancia respecto a cada valor y escojo el de mayor ganancia:
  - G(T>54)=E([3+,3-])-4/6E([3+,1-])-2/6E([0+,2-])
  - G(T>85)=E([3+,3-])-1/6E([0+,1-])-5/6E([3+,2-])
- Como el valor mayor es 54, los posibles valores del atributo pasan a ser: T>54={Verdadero, Falso}

Atributos con valores desconocidos

- Supongamos que <x,c(x)> es uno de los ejemplos de entrenamiento en S y que A(x) es desconocido
- Dos estrategias:
  - Asignar a A(x) el valor más común entre todos los ejemplos de entrenamiento pertenecientes al nodo n (...y se clasifiquen según c(x)).
  - Asignar una probabilidad a cada uno de los posibles valores del atributo A basada en la frecuencia observada en los ejemplos pertenecientes al nodo n. Finalmente, distribuir de acuerdo a dicha probabilidad

Manejando atributos con costes diferentes

- En ámbitos como la medicina, elegir un atributo u otro puede depender del coste asociado (escáner, biopsia, análisis de sangre,...)
- Podemos preferir entonces atributos con menores costes asociados (bias a favor de atributos con coste bajo).
- Atribuimos pues a cada atributo un coste coste(A), y calcularemos la ganancia como G=Ganancia/coste.
- También se pueden usar otras medidas como [$Ganancia^2$/coste] o $\{2^{Ganancia} -1 / [Coste(A)+1]^w\}$, donde $w \in [0,1]$ es la importancia relativa del Coste respecto a la ganancia de información.

Sobreajuste

- Sobreentrenamiento o sobreajuste (overfitting): A medida que añadimos niveles al AD, las hipótesis se refinan tanto que describen muy bien los ejemplos utilizados en el aprendizaje, pero el error de clasificación puede aumentar al evaluar ejemplos.
- Se dice que una hipótesis h se sobreajusta al conjunto de entrenamiento si existe alguna otra hipótesis h' tal que el error de h es menor que el de h' sobre el conjunto de entrenamiento, pero es mayo sobre la distribución completa de ejemplos del problema (entrenamiento + test)
  - Es decir, clasifica muy bien los datos de entrenamiento pero luego no sabe generalizar al conjunto de test. Es debido a que aprende hasta el ruido del conjunto de entrenamiento, adaptándose a las regularidades del conjunto de enrenamiento).

Tratamiento del sobreajuste

- Se intenta evitar con técnicas de **poda:**
  - Técnicas de pre-poda: tratan de detener el crecimiento del árbol antes de que éste llegue a adaptarse perfectamente al conjunto de entrenamiento.
  - Técnicas de post-poda: permiten que el árbol se sobreajuste a los datos y luego se efectúa sobre él una poda.
- De alguna forma estas técnicas tratan de compensar la falta de backtracking del proceso de inducción.
- En la práctica se usa más la técnica de postpoda, porque es difícil estimar con precisión cuando se debe detener el crecimiento del árbol.
- También se puede usar validación cruzada (o sea, técnicas de entrenar - testear)

Técnicas de pre-poda.

- Solución: Test χ^2 : No se utiliza un atributo para refinar el árbol si la partición a la que da lugar ofrece aproximadamente la misma proporción de ejemplos positivos/negativos que la de su nodo padre.
- Se basa en medir la desviación respecto de la esperada de un muestreo aleatorio:
  - Problemas:
    - No se comporta como χ^2 para pocos ejemplos.
    - Es muy conservador y puede parar antes de lo conveniente.

Reduced-error pruning

- Técnica propuesta por Quinlan en 1987
- Considera cada nodo del árbol como candidato para la poda, es decir, para eliminarlo del árbol.
- Podar un nodo significa eliminar la rama que parte de ese nodo convirtiéndolo en un nodo hoja (y por lo tanto más general). Esto se hará siempre y cuando el árbol resultante no sea peor que el original sobre el conjunto de test.
- Esto se repetirá hasta que tengamos un árbol que empeore el rendimiento.

Técnica de pot-poda de reglas

- Usada en el algoritmo C4.5
- Se realizan los siguientes pasos:
  - Construir el árbol de decisión a partir del conjunto de entrenamiento.
  - Convertir dicho árbol en un conjunto equivalente de reglas.
  - Podar (generalizar) cada regla eliminando cualquier predicado del antecedente que conlleve un aumento estimado del rendimiento sobre el conjunto de test o incluso sobre el conjunto de entrenamiento.
  - Ordenar las reglas eliminadas por su rendimiento estimado.

Técnica de pot-poda de reglas

- Conjunto de reglas
  - Mejora la legibilidad del árbol.
    - Rule 1: (31, lift 42.7)thyroid surgery = f TSH > 6 TT4 <= 37 -> class primary [0.970]
    - Rule 2: (63/6, lift 39.3) TSH > 6 FTI <= 65 -> class primary [0.892]
    - Rule 3: (270/116, lift 10.3) TSH > 6 -> class compensated [0.570]
    - Rule 4: (2225/2, lift 1.1) TSH <= 6 -> class negative [0.999]
    - Rule 5: (296, lift 1.1) on thyroxine = t FTI > 65 -> class negative [0.997]
    - Rule 6: (240, lift 1.1) TT4 > 153 -> class negative [0.996]
    - Rule 7: (29, lift 1.1) thyroid surgery = t FTI > 65 -> class negative [0.968]
    - Default class: negative

Técnica de pot-poda de reglas

- Cada regla consiste en:
  - Un número de regla que la identifica.
  - Una estimación de la exactitud de la regla basada en métodos estadísticos:
    - (n, lift x) o (n/m, lift x) . Al igual que una hoja, n n es el número de ejemplos de entrenamiento clasificados por la reglay m, si aparece, muestra cuántos de ellos no pertenecen a la clase predicha por la regla. La exactitud de clasificación se estima por el ratio de Laplace (n-m+1)/(n+2). Lift x es el resultado de dividir la exactitud de clasificación estimada por la frecuencia relativa de la clase en el conjunto de entrenamiento.
  - Una o más condiciones que deben cumplirse para que la regla sea aplicable.
  - Una clase predicha por la regla.
  - Un valor entre 0 y 1 que indica la confianza en la predicción hecha.

Aplicaciones

- El aprendizaje mediante AD constituye uno de los métodos inductivos más empleados en aplicaciones reales.
- Ejemplos:
  - GASOIL(1986): Diseño de sistemas de separación gas-petróleo en plataformas petrolíferas marinas de BP. Más de 2.500 reglas, 100 días/persona (10 años/persona). Ahorró a BP millones de dólares.
  - BMT (1990): Configuración de equipo de protección de incendios en edificios. Más de 30.000 reglas.
  - Aprendiendo a volar (1992): En lugar de construir un modelo de la dinámica del sistema (muy complejo), se aprendió un mapeo entre el estado actual y la decisión de control correcta para volar un Cessna en un simulador de vuelo. Resultados: aprendió a volar e incluso mejoraba algunas decisiones de sus “maestros”.