Aprendizaje Bayesiano

Eduardo Morales, Hugo Jair Escalante

INAOE

Contenido

Probabilidad

Existen diferentes interpretaciones de probabilidad, las más comunes son:

- Clásica: $P(A) = N(A)/N$
- Frecuencia relativa: $P(A) = \lim_{N\to\infty} N(A)/N$
- Subjetiva: *P(A) = "creencia en A"* (factor de apuesta)

Definición: Dado un experimento E y el espacio de muestreo S respectivo, a cada evento A le asociamos un número real $P(A)$, el cual es la *probabilidad* de A y satisface las siguientes propiedades:

Propiedades

1. $0 \leq P(A) \leq 1$
2. $P(S) = 1$
3. $P(A \cup B) = P(A) + P(B)$ si A y B son mutuamente exclusivos

Teorema 1: $P(\emptyset) = 0$
Teorema 2: $P(\overline{A}) = 1 - P(A)$
Teorema 3: $P(A \cup B) = P(A) + P(B) - P(A \cap B)$

Probabilidad Condicional

- Si A y B son dos eventos en S, la probabilidad de que ocurra A dado que ocurrió el evento B es la *probabilidad condicional* de A dado B, y se denota $P(A \mid B)$.
- La probabilidad condicional por definición es:

$$P(A \mid B) = P(A \cap B)/P(B)$$

  dado $P(B) > 0$
- Ejemplo: Para un dado, si sé que cayó impar, cuál es la probabilidad de 3?

Teorema de Bayes

- $P(A \mid B) = P(A \cap B)/P(B)$ y similarmente $P(B \mid A) = P(B \cap A)/P(A)$
- De donde: $P(B \mid A) = P(A \mid B)P(B)/P(A)$ y similarmente: $P(A \mid B) = P(B \mid A)P(A)/P(B)$
  Esta expresión se conoce como el *Teorema de Bayes*
- En su forma más general es:

$$P(B_j \mid A_i) = \frac{P(B_j)P(A_i \mid B_j)}{\sum_j P(A_i \mid B_j)P(B_j)}$$

El denominador se le conoce como el teorema de la probabilidad total.

Partición y Eventos Independientes

- Si $B_1, B_2, \ldots, B_k$ representan una *partición* (exclusivos, exhaustivos y mayores a cero) de S y A es un evento respecto a S, entonces la probabilidad de A la podemos escribir como:

$$P(A) = \sum_j P(A \mid B_j) P(B_j)$$

- Dos eventos, A y B, son *independientes* si la ocurrencia de uno no tiene que ver con la ocurrencia de otro.
  A es independiente de B si y sólo si:
  $P(A \cap B) = P(A)P(B)$
  Por lo que: $P(A \mid B) = P(A)$ y $P(B \mid A) = P(B)$

Independientes es diferente a mutuamente exclusivos.

Independencia condicional

Un evento A es condicionalmente independiente de otro B dado un tercer evento C, si el conocer C hace que A y B sean independientes. Esto es: $P(A \mid B, C) = P(A \mid C)$
Ejemplo:

- A - *regar el jardín*
- B - *predicción del clima*
- C - *lluvia*

De la definicíon de probabilidad condicional, podemos obtener una expresión para evaluar la *probabilidad conjunta* de N eventos:

$$P(A_1, A_2, \ldots, A_n) = \begin{array}{l} P(A_1 \mid A_2, \ldots, A_n) \\ P(A_2 \mid A_3, \ldots, A_n) \cdots P(A_n) \end{array}$$

Variables Aleatorias

- Si a cada posible evento A le asignamos un valor numérico real, $X(A)$, obtenemos una *variable aleatoria*
- A cada valor de la variable le corresponde una probabilidad, $P(X = k)$.
- Las variables aleatorias pueden ser de dos tipos: discretas y continuas. Nosotros nos enfocaremos a variables discretas.
- Ejemplos (var. discretas): lanzar una moneda, lanzar un dado, número de fallas antes de darle al blanco, etc.

Función acumulativa de probabilidad

Para una variable aleatoria X, se define la *función acumulativa de probabilidad* como la probabilidad de que la variable aleatoria sea menor a un valor x: $F(x) = P\{X \leq x\}$ que corresponde a la sumatoria de la función de probabilidad de $-\infty$ a x: $F(x) = \sum_{-\infty}^{x} p(X)$

Propiedades:

1. $0 \leq F(x) \leq 1$
2. $F(x1) \leq F(x2)$ si $x1 \leq x2$ (función siempre creciente)
3. $F(-\infty) = 0$
4. $F(+\infty) = 1$

Estadísticas de una variable aleatoria

Valores característicos de una variable aleatoria:

- Moda: valor de probabilidad máxima
- Media: valor medio (divide el área en 2 partes iguales)

Momentos

- Promedio (valor esperado o primer momento): $E\{X\} = M_1(X) = \sum x_i P(x_i)$
- Valor promedio-cuadrado (segundo momento): $M_2(X) = \sum x_i^2 P(x_i)$
- Momento *N*: $M_n(X) = \sum x_i^n P(x_i)$

Momentos "centrales"

- Varianza: $\sigma^2(X) = \sum (x_i - E\{X\})^2 P(x_i)$
- Desviación estandar: $\sigma(x) = \sqrt{\sigma^2(x)}$

Variables Aleatorias de 2-Dimensiones

- Dado un experimento E con espacio de muestreo S. Si X y Y son dos funciones que le asignan números reales a cada resultado posible, entonces (X, Y) es una *variable aleatoria bidimensional*
- Dadas dos variables aleatorias (discretas), X, Y, deben satisfacer lo siguiente:
  1. $P(x_i, y_j) \geq 0$
  2. $\sum_i \sum_j P(x_i, y_j) = 1$
- Ejemplos: número de artículos terminados en dos líneas de producción, número de pacientes con cancer y número de fumadores, etc.

Probabilidad marginal y condicional

- Probabilidad marginal es la probabilidad particular de una de las variables dada una variable aleatoria bidimensional, y se define como:
  $P(X) = \sum_j P(x_i, y_j)$
- Dada la probabilidad conjunta y marginal, la probabilidad condicional se define como:
  $P(X \mid Y) = P(X, Y)/P(Y)$

Dependencia

- Dos variables aleatorias son *independientes* si su probabilidad conjunta es igual al producto de las marginales, esto es:
  $P(x_i, y_j) = P(x_i)P(y_j), \forall(i,j)$
- El *coeficiente de correlación* (ρ) denota el grado de linearidad entre dos variables aleatorias y se define como:
  $\rho_{xy} = E\{[X - E\{X\}][Y - E\{Y\}]\}/\sigma_x\sigma_y$
- La correlación está dentro del intervalo: $\rho \in [-1, 1]$, donde un valor de 0 indica no-correlacionadas, y un valor de -1 ó 1 indica una relación lineal.
- Independencia $\rightarrow$ no-correlación (pero no viceversa)

Distribución Binomial

Una distribución binomial da la probabilidad de observar r eventos (e.g., soles) de n muestras independientes con dos posibles resultados (e.g., tirar monedas).

$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{(n-r)}$$

- El valor esperado es: $E\{x\} = np$
- La varianza es: $Var(x) = np(1-p)$
- La desviación estandar es: $\sigma_x = \sqrt{np(1-p)}$

Si n es grande, se aproxima a una distribución Normal

Distribución Normal o Gaussiana

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

- El valor esperado es: $E\{x\} = \mu$
- La varianza es: $Var(x) = \sigma^2$
- La desviación estandar es: $\sigma_x = \sigma$

El Teorema Central del Límite dice que la suma de un número grande de variables aleatorias independientes identicamente distribuidas siguen una distribución Normal.

Aprendizaje Bayesiano (BL)

Algunas características:

- Cada nuevo ejemplo puede aumentar o disminuir la estimación de una hipótesis (flexibilidad - incrementalidad)
- Conocimiento *a priori* se puede combinar con datos para determinar la probabilidad de las hipótesis
- Da resultados con probabilidades asociadas
- Puede clasificar combinando las predicciones de varias hipótesis
- Sirve de estandar de comparación de otros algoritmos

Aprendizaje Bayesiano (BL)

Es importante por:

- ser práctico
- provee un enfoque de comprensión (y diseño) de otros algoritmos

Problemas:

- Se requieren conocer muchas probabilidades
- Es computacionalmente caro (depende linealmente del número de hipótesis)

Aprendizaje Bayesiano (BL)

- Lo que normalmente se quiere saber en aprendizaje es cuál es la mejor hipótesis (más probable) dados los datos
- Si $P(D)$ = probabilidad *a priori* de los datos (i.e., cuales datos son más probables que otros), $P(D \mid h)$ = probabilidad de los datos dada una hipótesis, lo que queremos estimar es: $P(h \mid D)$, la probabilidad posterior de h dados los datos.
- Esto lo podemos estimar con Bayes.

$$P(h \mid D) = \frac{P(D \mid h)P(h)}{P(D)}$$

La Hipótesis más Probable

Para estimar la hipótesis más probable o MAP (*maximum a posteriori hypothesis*):

$$\begin{aligned} h_{MAP} &= argmax_{h \in H} \left(P(h \mid D) \right) \\ &= argmax_{h \in H} \left(\frac{P(D|h)P(h)}{P(D)} \right) \\ &\approx argmax_{h \in H} \left(P(D \mid h)P(h) \right) \end{aligned}$$

Ya que $P(D)$ es una constante independiente de h.

Si suponemos que las hipótesis son igualmente probables, nos queda la hipótesis de máxima verosimilitud o ML (*maximum likelihood*):

$$h_{ML} = argmax_{h \in H} \left(P(D \mid h) \right)$$

Ejemplo

- Se tienen dos hipótesis, el paciente tiene cancer o no
- Sabemos que sólo el 0.008% de la población tiene ese tipo de cancer
- La prueba sobre cancer no es infalible, nos da resultados positivos correctos en el 98% de los casos y resultados negativos correctos en el 97% de los casos

$P(cancer) = 0.008$ y $P(\neg cancer) = 0.992$
$P(\oplus|cancer) = 0.98$ y $P(\ominus|cancer) = 0.02$
$P(\oplus|\neg cancer) = 0.03$ y $P(\ominus|\neg cancer) = 0.97$

Ejemplo

- Si a un paciente le dieron un resultado positivo en la prueba:
  $P(cancer|\oplus) = P(cancer)P(\oplus|cancer) =$
  $0.008 * 0.98 = 0.0078$
  $P(\neg cancer|\oplus) = P(\neg cancer)P(\oplus|\neg cancer) =$
  $0.992 * 0.03 = 0.0298$
- Que al normalizar, nos da:
  $P(cancer|\oplus) = 0.21$
  $P(\neg cancer|\oplus) = 0.69$
- Por lo que sigue siendo más probable que no tenga cancer

Diferentes vistas de BL

Al aprendizaje bayesiano lo podemos relacionar con diferentes aspectos de aprendizaje:

- Espacio de Versiones
- Clases contínuas con ruido
- Predecir probabilidades de clases
- Principio de Longitud de Descripción Mínima
- Clasificador bayesiano óptimo
- ...

BL y Espacio de Versiones

- Una forma (impráctica) de un algoritmo Bayesiano es calcular todas las posibles hipótesis $P(h \mid D) = \frac{P(D|h)P(h)}{P(D)}$ y quedarse con la de mayor probabilidad
- Además necesitamos especificar los valores para $P(h)$ y para $P(D \mid h)$
- Si suponemos que no hay ruido y que todas las hipótesis son igualmente probables (i.e., $P(h) = \frac{1}{|H|} \forall h \in H$), $P(D \mid h) = 1$ sii D es consistente con h
- Esto es:

$$P(h \mid D) = \frac{1}{\mid VS_{H,D} \mid}$$

donde, $VS_{H,D}$ es el subconjunto de hipótesis de H que es consistente con D (su espacio de versiones).

BL y Espacio de Versiones

- Por lo mismo, toda hipótesis consistente es una hipótesis MAP
- Cualquier sistema de aprendizaje que nos de hipótesis consistentes (suponiendo que no hay ruido y que todas las hipótesis son igualmente probables) nos está dando hipótesis MAP.
- Un sistema de aprendizaje lo podemos caracterizar suponiendo que las hipótesis más generales (o específicas) son más probables que las otras
- En general, podemos caracterizar varios algoritmos de aprendizaje con un enfoque Bayesiano, al caracterizar sus distribuciones de probabilidad $P(h)$ y $P(D \mid h)$

BL, Variables Continuas y Ruido

- Los métodos más usados para buscar funciones con variables continuas a partir de datos con ruido, son regresiones líneales, ajustes de polinomios y redes neuronales.
- La idea es aprender funciones $h : X \rightarrow \mathcal{R}$ lo más cercanas a f, en donde los datos están descritos por: $d_i = f(x_i) + e_i$, donde $f(x_i)$ es la función sin ruido y e_i es una variable aleatoria representando el error
- De nuevo lo que queremos es encontrar la hipótesis más probable:

$$h_{ML} = argmax_{h \in H} \left(p(D \mid h) \right)$$

BL, Variables Continuas y Ruido

- Suponiendo que los datos son independientes entre sí dado h, la probabilidad se puede expresar como el producto de varias $p(d_i \mid h)$ para cada dato:

$$h_{ML} = argmax_{h \in H} \left(\prod_{i=1}^{m} p(d_i \mid h) \right)$$

- Si suponemos el ruido con una distribución Gaussiana con media cero y varianza σ^2, cada d_i debe de seguir la misma distribución centrada alrededor de $f(x_i)$.

$$h_{ML} = argmax_{h \in H} \left(\prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i-\mu)^2} \right)$$

$$h_{ML} = argmax_{h \in H} \left(\prod_{i=1}^{m} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(d_i-h(x_i))^2} \right)$$

BL, Variables Continuas y Ruido

- Podemos maximizar tomando su logaritmo (dado que es una función monotónica creciente):

$$h_{ML} = argmax_{h \in H} \left(\sum_{i=1}^{m} ln(\frac{1}{\sqrt{2\pi\sigma^2}}) - \frac{1}{2\sigma^2}(d_i - h(x_i))^2 \right)$$

- Eliminando el primer término (que no depende de h):

$$h_{ML} = argmax_{h \in H} \left(\sum_{i=1}^{m} -\frac{1}{2\sigma^2}(d_i - h(x_i))^2 \right)$$

BL, Variables Continuas y Ruido

- Que es igual a minimizar lo mismo con el signo contrario. Al cambiar signo y eliminar constantes que no dependen de h nos queda:

$$h_{ML} = argmin_{h \in H} \left(\sum_{i=1}^{m} (d_i - h(x_i))^2 \right)$$

- Lo que nos dice que la hipótesis de máxima verosimilitud es la que minimiza la suma de los errores al cuadrado entre los datos observados (d_i) y los datos predichos ($h(x_i)$), siempre y cuando el error siga una distribución Normal con media cero.
- Supone que el error está únicamente en la meta y no en los atributos

Aprendiendo a Predecir Probabilidades

- Otra aplicación común es querer aprender una función de probabilidad sobre un función que tiene dos posibles resultados (e.g., 0 o 1)
- Si los datos (D) son: $D = \{(x_1, d_1), \ldots, (x_m, d_m)\}$, donde d_i es el valor observado (0 o 1) de $f(x_i)$, y suponiendo que los datos son independientes:

$$P(D \mid h) = \prod_{i=1}^{m} P(x_i, d_i \mid h)$$

- Si x_i es independiente de h

$$P(D \mid h) = \prod_{i=1}^{m} P(d_i \mid h, x_i) P(x_i)$$

Aprendiendo a Predecir Probabilidades

- Como h es la probabilidad de la función meta $P(d_i = 1 \mid h, x_i) = h(x_i)$, y en general:

$$P(d_i \mid h, x_i) = \begin{cases} h(x_i) & \text{si } d_i = 1 \\ 1 - h(x_i) & \text{si } d_i = 0 \end{cases}$$

- Esto mismo lo podemos expresar como:

$$P(d_i \mid h, x_i) = h(x_i)^{d_i}(1 - h(x_i))^{1-d_i}$$

- Por lo que:

$$P(D \mid h) = \prod_{i=1}^{m} h(x_i)^{d_i}(1 - h(x_i))^{1-d_i} P(x_i)$$

Aprendiendo a Predecir Probabilidades

- La máxima verosimilitud es entonces:

$$h_{ML} = argmax_{h \in H} \left(\prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} P(x_i) \right)$$

- Al eliminar el último término (que no depende de h), tenemos:

$$h_{ML} = argmax_{h \in H} \left(\prod_{i=1}^{m} h(x_i)^{d_i} (1 - h(x_i))^{1-d_i} \right)$$

- Lo cual es una generalización de la distribución Binomial (e.g., describe la probabilidad de que al tirar m monedas se tenga $(d_1, \ldots, d_m)$ resultados suponiendo que cada moneda tiene probabilidad $h(x_i)$ de salir sol)

Aprendiendo a Predecir Probabilidades

- En la descripción de la distribución binomial se supone que todas las monedas tienen la misma probabilidad de que salga sol.
- Trabajando (de nuevo) con el logaritmo:

$$h_{ML} = argmax_{h \in H} \left(\sum_{i=1}^{m} d_i ln(h(x_i)) + (1 - d_i) ln(1 - h(x_i)) \right)$$

- Lo cual, por su parecido con la medida de entropía, también se llama a su negativo, entropía cruzada (*cross entropy*).

BL y el Principio de Longitud de Descripción Mínima

- Como el proceso inductivo no es seguro se necesita alguna medida de calidad
- Normalmente se hace con base en evaluaciones con los ejemplos de entrenamiento y prueba
- Una alternativa es encontrar la hipótesis más probable dados los datos
- El MDL está motivado al interpretar la definición de h_{MAP} con base en conceptos de teoría de información.

BL y MDL

$$
\begin{aligned}
h_{MAP} &= argmax_{h \in H} \left(P(D \mid h) P(h) \right) \\
&= argmax_{h \in H} \left(log_2(P(D \mid h)) + log_2(P(h)) \right) \\
&= argmin_{h \in H} \left(-log_2(P(D \mid h)) - log_2(P(h)) \right)
\end{aligned}
$$

Lo cual puede pensarse como el problema de diseñar el mensaje de transmisión de información más compacto para transmitir la hipótesis y los datos dada la hipótesis

BL y MDL

- MDL recomienda seleccionar la hipótesis que minimiza la suma de estas dos descripciones:

$$h_{MDL} = argmin_{h \in H} \left(L(h) + L(D \mid h) \right)$$

- Si queremos aplicarlo a un árbol de decisión, tenemos que buscar una codificación para los árboles de decisión y una para los ejemplos mal clasificados junto con su clasificación
- Permite establecer un balance entre complejidad de la hipótesis ($L(h)$) y número de errores o calidad de la hipótesis ($L(D \mid h)$)

Clasificador Bayesiano Óptimo

- En lugar de la hipótesis más probable, podemos preguntar, cuál es la clasificación más probable
- Se puede obtener combinando las clasificaciones de todas las hipótesis aplicables pesadas por su probabilidad.

$$P(v_j \mid D) = \sum_{h_i \in H} P(v_j \mid D, h_i) P(h_i \mid D) = \sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D)$$

Donde v_j es el valor de la clasificación y la clasificación óptima será:

$$argmax_{v_j \in V} \left(\sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D) \right)$$

Ejemplo

- Tenemos 2 clases y 3 hipótesis (h_1, h_2, h_3), cuyas probabilidades dados los datos son $(0.4, 0.3, 0.3)$
- Un nuevo ejemplo x se clasifica positivo por h_1 y negativo por h_2 y h_3
- Su clasificación por la hipótesis MAP sería positivo, pero considerando todas las hipótesis sería negativo.

Ejemplo

$$P(h_1|D) = 0.4, P(\ominus|h_1) = 0, P(\oplus|h_1) = 1$$
$$P(h_2|D) = 0.3, P(\ominus|h_2) = 1, P(\oplus|h_2) = 0$$
$$P(h_3|D) = 0.3, P(\ominus|h_3) = 1, P(\oplus|h_3) = 0$$

$$\sum_{h_i \in H} P(\oplus \mid h_i) P(h_i \mid D) = 0.4$$

$$\sum_{h_i \in H} P(\ominus \mid h_i) P(h_i \mid D) = 0.6$$

$$argmax_{v_j \in \{\oplus, \ominus\}} \left(\sum_{h_i \in H} P(v_j \mid h_i) P(h_i \mid D) \right) = \ominus$$

Clasificador Bayesiano Óptimo

- Aplicar el clasificador Bayesiano óptimo puede ser muy costoso (muchas hipótesis)
- Una posibilidad es seleccionar una hipótesis (h) aleatoriamente de acuerdo con la distribución de probabilidad de las probabilidades posteriores de H, y usar h para predecir
- Se puede mostrar que el error esperado es a lo más el doble del error esperado del clasificador Bayesiano óptimo.

Naïve Bayes

- Se usa para clasificar una instancia descrita por un conjunto de atributos (a_i's) en un conjunto finito de clases (V)
- Clasifica de acuerdo con el valor más probable dados los valores de sus atributos:

$$v_{MAP} = argmax_{v_j \in V} \left(P(v_j \mid a_1, \ldots, a_n) \right)$$

- Usando Bayes:

$$\begin{aligned} v_{MAP} &= argmax_{v_j \in V} \left(\frac{P(a_1,\ldots,a_n|v_j)P(v_j)}{P(a_1,\ldots,a_n)} \right) \\ &= argmax_{v_j \in V} \left(P(a_1, \ldots, a_n \mid v_j)P(v_j) \right) \end{aligned}$$

Naïve Bayes

- $P(v_j)$ se puede estimar con la frecuencia de las clases, pero para $P(a_1, \ldots, a_n \mid v_j)$ tenemos pocos datos
- El clasificador NB supone que los valores de los atributos son condicionalmente independientes entre sí dado el valor de la clase:
  $P(a_1, \ldots, a_n \mid v_j) = \prod_i P(a_i \mid v_j)$
- Por lo que:

$$v_{NB} = argmax_{v_j \in V} \left(P(v_j) \prod_i P(a_i \mid v_j) \right)$$

  Los valores $P(a_i \mid v_j)$ se estiman con la frecuencia de los datos observados.

Nota: no se hace búsqueda de hipótesis, simplemente se cuentan frecuencias de ocurrencias.

Ejemplo

Tomando la tabla de jugar golf, si queremos clasificar el siguiente ejemplo con un *naïve Bayes*:
Ambiente=soleado, Temperatura=baja, Humedad=alta, Viento=si

$$v_{NB} = \begin{aligned} & argmax_{v_j \in \{P,N\}} P(v_j) \left(P(Ambiente = soleado \mid v_j) \right. \\ & P(Temperature = baja \mid v_j) P(Humedad = alta \mid v_j) \\ & \left. P(Viento = si \mid v_j) \right) \end{aligned}$$

Aprendizaje Bayesiano

- $P(Clase = P) = 9/14$ y $P(Clase = N) = 5/14$
- $P(Viento = si \mid P) = 3/9 = 0.33$ y $P(Viento = si \mid N) = 3/5 = 0.60$
- ...
- $P(P)P(soleado \mid P)P(baja \mid P)P(alta \mid P)P(si \mid P) = 0.0053$
- $P(N)P(soleado \mid N)P(baja \mid N)P(alta \mid N)P(si \mid N) = 0.0206$
- Normalizando el último nos da: $\frac{0.0206}{0.0206+0.0053} = 0.795$.

Estimación de Probabilidades

- Como ya vimos, podemos estimar probabilidades por frecuencia simple: $\frac{n_c}{n}$ (malo con pocos ejemplos)
- Podemos usar un estimador m (*m-estimate*):

$$\frac{n_c + m * p}{n + m}$$

donde p es una estimación *a priori* de lo que queremos estimar y m es una constante llamada "tamaño de muestra equivalente" (*equivalent sample size*).

- Una valor típico para p es suponer una distribución uniforme, por lo que: $p = \frac{1}{k}$ cuando existen k valores
- m se usa como estimador de ruido

Ejemplo: Clasificar Textos

- Los ejemplos son textos asociados con una clase (e.g., me interesa vs. no me interesa ó política, deportes, espectáculos, sociales, etc.)
- Suponemos que las palabras son independientes entre sí y de su posición en el texto
- *Vocabulario* = todas las palabras distintivas (eliminando palabras muy comunes y poco distintivas como artículos, puntuaciones, etc.)

Ejemplo: Clasificar Textos

- $doc(clase)$ = subconjunto de textos de esa clase
- $P(clase) = \frac{|doc(clase)|}{Ejemplos}$
- *Texto* = concatenación de todos los textos en $doc(clase)$, n = número de palabras distintas en *Texto*
- Para cada palabra (w) en *Vocabulario*: n_k = número de veces que aparece la palabra w en *Texto*
- Se calcula la probabilidad considerando el estimador m, $\frac{n_c+mp}{n+m}$ con probabilidad uniforme en las clases (Laplace) y $m = |Vocabulario|$

$$P(w|clase) = \frac{n_k + 1}{n + |Vocabulario|}$$

Ejemplo: Clasificar Textos

Para clasificar un nuevo documento (considerando sólo las palabras en el nuevo documento que tenemos en *Vocabulario*):

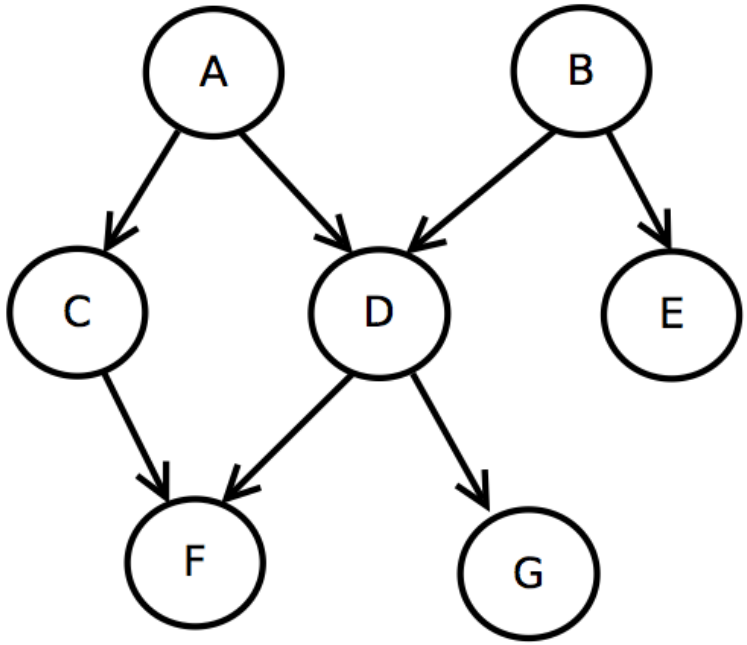
$$v_{NB} = argmax_{v_j \in V} \left(P(v_j) \prod_i P(a_i \mid v_j) \right)$$

Redes Bayesianas

- Las redes bayesianas o probabilísticas son una representación gráfica de dependencias para razonamiento probabilístico
- Es un gráfo acíclico dirigido (DAG) en el cual cada nodo representa una variable aleatoria y cada arco una dependencia probabilística, en la cual se especifica la probabilidad condicional de cada variable dados sus padres
- La variable a la que apunta el arco es dependiente de la que está en el origen de éste

Redes Bayesianas

- Una Red Bayesiana representa la distribución de la probabilidad conjunta de las variables representadas en la red. Por ejemplo:
$P(A, B, C, D, E, F, G) =$
$P(G|D)P(F|C, D)P(E|B)P(D|A, B)P(C|A)P(B)P(A)$



Redes Bayesianas

- La topología o estructura de la red nos da información sobre las dependencias probabilísticas entre las variables
- La red también representa las independencias condicionales de una variable (o conjunto de variables) dada(s) otra(s) variable(s)
- $\{E\}$ *es cond. indep. de* $\{A,C,D,F,G\}$ *dado* $\{B\}$
  Esto es: $P(E|A, C, D, F, G, B) = P(E|B)$
  Esto se representa gráficamente por el nodo B separando al nodo E del resto de las variables.

Redes Bayesianas

- En general, el conjunto de variables A es independiente del conjunto B dado C si al remover C hace que A y B se desconecten
- Es decir, NO existe una trayectoria entre A y B en que las siguientes condiciones sean verdaderas:
  1. Todos los nodos con flechas convergentes están o tiene descendientes en C.
  2. Todos los demás nodos están fuera de C.
- Esto se conoce como *Separación–D*

Redes Bayesianas

- En una BN todas las relaciones de independencia condicional representadas en el grafo corresponden a relaciones de independencia de la distribución de probabilidad
- Dichas independencias simplifican la representación del conocimiento (menos parámetros) y el razonamiento (propagación de las probabilidades)

Propagación de Probabilidades

- El razonamiento probabilístico o propagación de probabilidades consiste en propagar la evidencia a través de la red para conocer la probabilidad *a posteriori* de las variables
- La propagación consiste en darle valores a ciertas variables (evidencia), y obtener la probabilidad posterior de las demás variables dadas las variables conocidas (instanciadas)
- Los algoritmos de propagación dependen de la estructura de la red:
  1. árboles
  2. Poliárboles
  3. Redes multiconectadas

Aprendizaje de Redes Bayesianas

Las redes bayesianas son una alternativa para algoritmos de aprendizaje, la cual tiene varias ventajas:

- Permiten aprender sobre relaciones de dependencia y causalidad.
- Permiten combinar conocimiento con datos.
- Evitan el sobre-ajuste de los datos.
- Pueden manejar bases de datos incompletas.

Aprendizaje de Redes Bayesianas

El obtener una red bayesiana a partir de datos es un proceso de aprendizaje el cual se divide, naturalmente, en dos aspectos:

1. **Aprendizaje paramétrico**: Dada una estructura, obtener las probabilidades *a priori* y condicionales requeridas.
2. **Aprendizaje estructural**: Obtener la estructura de la red Bayesiana, es decir, las relaciones de dependencia e independencia entre las variables involucradas.

Aprendizaje Paramétrico

- Consiste en encontrar los parámetros asociados a una estructura dada de una red bayesiana
- Ésto es, las probabilidades *a priori* de los nodos raíz y las probabilidades condicionales de las demás variables, dados sus padres
- Para que se actualicen las probabilidades con cada caso observado, éstas se pueden representar como razones enteras y actualizarse con cada observación

Aprendizaje Paramétrico en Árboles

Probabilidades previas:

$$\begin{aligned} P(A_i) &= (a_i + 1)/(s + 1) \quad i = k \\ P(A_i) &= a_i/(s + 1) \quad\quad\quad\quad i \neq k \end{aligned}$$

Probabilidades condicionales:

$$\begin{aligned} P(B_j \mid A_i) &= (b_j + 1)/(a_i + 1) \quad i = k \text{ y } j = l \\ P(B_j \mid A_i) &= b_j/(a_i + 1) \quad\quad\quad i = k \text{ y } j \neq l \\ P(B_j \mid A_i) &= b_j/a_i \quad\quad\quad\quad\quad i \neq k \end{aligned}$$

Donde s corresponde al número de casos totales, i, j los índices de las variables, k, l los índices de las variables observadas.

Variables no observadas

- En algunos casos, existen variables que son importantes para el modelo pero para las cuales no se tienen datos (nodos no observables o *escondidos*)
- Si algunos nodos son parcialmente observables, se pueden estimar de acuerdo a los observables con el siguiente algoritmo:
  1. Instanciar todas las variables observables.
  2. Propagar su efecto y obtener las probabilidades posteriores de las no observables
  3. Para las variables no observables, *suponer* el valor con mayor probabilidad como observado
  4. Actualizar las probabilidades previas y condicionales de acuerdo a las formulas anteriores.
  5. Repetir 1 a 4 para cada observación

Variables no observadas

- Existen otras formas más sofisticadas en las que las probabilidades se actualizan dando incrementos no sólo al valor mayor, sino a todos en proporción de las probabilidades posteriores
- El aprendizaje se basa en el gradiente, lo cual es análogo al aprendizaje del peso en capas ocultas de redes neuronales.
- En este caso, maximizar $P(D \mid h)$ siguiendo el gradiente del $ln(P(D \mid h))$ con respecto a los parámetros que definen las tablas de probabilidad condicional
- Estos algoritmos suponen que se tienen algunos datos, a partir de los cuales es posible estimar una probabilidad (aunque por tener pocos datos se tenga que ajustar)
- Cuando no se tiene ningún valor para un dato, se puede usar EM el cual se describe en la clase de *Clustering*

Aprendizaje Estructural

- Las técnicas de aprendizaje estructural dependen del tipo de estructura de red: Árboles, poliárboles y redes multicomectadas
- Una alternativa es combinar conocimiento subjetivo del experto con aprendizaje. Para ello se parte de la estructura dada por el experto, la cual se valida y mejora utilizando datos estadísticos

Aprendizaje Estructural

- En el caso de **Naïve Bayes** sólo tenemos que aprender los parámetros
- Una forma de mejorar la estructura de un NB es añadiendo arcos entre los nodos o atributos que tengan cierta dependencia
- Existen dos estructuras básicas:
  1. TAN: Clasificador bayesiano simple aumentado con un árbol.
  2. BAN: Clasificador bayesiano simple aumentado con una red.

Aprendizaje Estructural

Otra forma es realizando operaciones locales hasta que no mejore la predicción:

1. Eliminar un atributo,
2. Unir dos atributos en una nueva variable combinada,
3. Introducir un nuevo atributo que haga que dos atributos dependientes sean independientes (nodo oculto).

Se pueden ir probando cada una de las opciones anteriores midiendo la dependencia de los atributos dada la clase:

$$I(X_i, X_j \mid C) = \sum_{X_i, X_j} P(X_i, X_j \mid C) log(\frac{P(X_i, X_j \mid C)}{P(X_i \mid C) P(X_j \mid C)})$$

Aprendizaje Estructiural

Algoritmo de Mejora Estructural:

1. Obtener la información mutua condicional (IMC) entre cada par de atributos.
2. Seleccionar el par de atributos de IMC mayor.
3. Probar las 3 operaciones básicas (i) eliminación, (ii) unión, (iii) inserción.
4. Evaluar las 3 estructuras alternativas y la original, y quedarse con la "mejor" opción.
5. Repetir 2–4 hasta que ya no mejore el clasificador.

Para evaluar las estructuras resultantes se pueden usar datos de prueba o una medida basada en MDL.

Algoritmo de Chow y Liu

- Aprendizaje estructural de árboles basado en el algoritmo desarrollado por Chow y Liu (68) para aproximar una distribución de probabilidad por un producto de probabilidades de segundo orden
- La probabilidad conjunta de n variables se puede representar (aproximar) como:

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i) P(X_i \mid X_{j(i)}))$$

donde $X_{j(i)}$ es la causa o padre de X_i.

Algoritmo de Chow y Liu

- Se plantea el problema como uno de optimización y lo que se desea es obtener la estructura en forma de árbol que más se aproxime a la distribución "real"
- Se utiliza una medida de la diferencia de información entre la distribución real (P) y la aproximada (P^*):

$$I(P, P^*) = \sum_x P(\mathbf{X}) log(\frac{P(\mathbf{X})}{P^*(\mathbf{X})})$$

donde el objetivo es minimizar I

Algoritmo de Chow y Liu

- Se puede definir dicha diferencia en función de la información mutua entre pares de variables, que se define como:

$$I(X_i, X_j) = \sum_x P(X_i, X_j) log(\frac{P(X_i, X_j)}{P(X_i)P(X_j)})$$

- Se puede demostrar (Chow 68) que la diferencia de información es una función del negativo de la suma de las informaciones mutuas (pesos) de todos los pares de variables que consituyen el árbol
- Por lo que encontrar el árbol más próximo equivale a encontrar el árbol con mayor peso

Algortimo de Chow y Liu

1. Calcular la información mutua entre todos los pares de variables ($n(n-1)/2$).
2. Ordenar las informaciones mutuas de mayor a menor
3. Seleccionar la rama de mayor valor como árbol inicial
4. Agregar la siguiente rama mientras no forme un ciclo, si es así, desechar
5. Repetir (4) hasta que se cubran todas las variables ($n-1$ ramas)

El algoritmo NO provee la direccionalidad de los arcos, por lo que ésta se puede asignar en forma arbitraria o utilizando semántica externa (experto)

Ejemplo Bayesiano

Para los ejemplos de jugar golf:

No.	Var 1	Var 2	Info. mutua
1	temp.	ambiente	.2856
2	juega	ambiente	.0743
3	juega	humedad	.0456
4	juega	viento	.0074
5	humedad	ambiente	.0060
6	viento	temp.	.0052
7	viento	ambiente	.0017
8	juega	temp.	.0003
9	humedad	temp.	0
10	viento	humedad	0

Aprendizaje de Poliárboles

- Rebane y Pearl [89] extendieron el algoritmo de Chow y Liu para poliárboles
- Parten del esqueleto obtenido con Chow y Liu y determinan las dirección de los arcos utilizando pruebas de dependencia entre tripletas de variables
- Dadas 3 variables, existen 3 casos posibles:
  1. Arcos divergentes: $X \leftarrow Y \rightarrow Z$.
  2. Arcos secuenciales: $X \rightarrow Y \rightarrow Z$.
  3. Arcos convergentes: $X \rightarrow Y \leftarrow Z$.
- Los primeros dos casos son indistinguibles, pero el tercero es diferente, ya que las dos variables “padre” son marginalmente independientes

Algoritmo para Poliárboles

1. Obtener el esqueleto utilizando Chow y Liu
2. Encontrar tripletas de nodos que sean convergentes (tercer caso) -nodo multipadre-.
3. A partir de un nodo multipadre determinar las direcciones de los arcos utilizando la prueba de tripletas hasta donde sea posible (base causal).
4. Repetir 2-3 hasta que ya no se puedan descubrir más direcciones
5. Si quedan arcos sin direccionar utilizar semántica externa para obtener su dirección

Sólo para poliárboles, no garantiza obtener todas las direcciones y requiere de un umbral

Aprendizaje de Redes Generales

- Existen dos clases de métodos para el aprendizaje genérico de redes bayesianas, que incluyen redes multiconectadas:
  1. Métodos basados en medidas de ajuste y búsqueda.
  2. Métodos basados en pruebas de independencia.

Aprendizaje de Redes Basados en Búsqueda

- En general se generan diferentes estructuras y se evalúan respecto a los datos utilizando alguna medida de ajuste
- Estos métodos tienen dos aspectos principales:
  1. Una medida para evaluar qué tan *buena* es cada estructura respecto a los datos (e.g., BIC, MDL, etc.)
  2. Un método de búsqueda que genere diferentes estructuras hasta encontrar la *óptima*, de acuerdo a la medida seleccionada

Medida Bayesiana (BIC)

- El criterio de información bayesiano (BIC) estima la probabilidad de la estrutura dado los datos la cual se trata de maximizar
- Busca maximizar la probabilidad de la estructura dados los datos, esto es:

$$P(Es \mid D)$$

  Donde Es es la estructura y D son los datos
- La podemos escribir en términos relativos al comparar dos estructuras, i y j como:

$$P(Es_i \mid D)/P(Es_j \mid D) = P(Es_i, D)/P(Es_j, D)$$

- Considerando variables discretas y que los datos son independientes, las estructuras se pueden comparar en función del número de ocurrencias (frecuencia) de los datos predichos por cada estructura

Medida basada en MDL

- Estima la longitud (tamaño en bits) requerida para representar la probabilidad conjunta con cierta estructura, la cual se compone de dos partes:
  1. Representación de la estructura,
  2. Representación del error de la estructura respecto a los datos
- Hace un compromiso entre exactitud y complejidad
- La exactitud se estima midiendo la información mutua entre los atributos y la clase; y la complejidad contando el número de parámetros

Medida basada en MDL

- Se utiliza una constante, α, en [0, 1] para balancear exactitud contra complejidad
- La medida de calidad está dada por:

$$MC = \alpha(W/Wmax) + (1 - \alpha)(1 - L/Lmax)$$

Donde W y $Wmax$ representa la exactitud y máxima exactitud, y L y $Lmax$ representan la complejidad y máxima complejidad del modelo

- Para determinar estos máximos normalmente se considera un máximo en cuanto al número de padres máximo permitido por nodo

Medida basada en MDL: Complejidad

- La complejidad está dada por el número de parámetros requeridos para representar el modelo
- Se puede calcular como:

$$L = S_i[k_i log_2 n + d(S_i - 1)F_i]$$

Donde, n es el número de nodos, k es el número de padres por nodo, S_i es el número de valores promedio por variable, F_i el número de valores promedio de los padres, y d el número de bits por parámetro

Medida basada en MDL: Exactitiud

- La exactitud se puede estimar con base en el "peso" de cada nodo
- El peso de cada nodo se estima con la información mutua con sus padres:

$$w(xi, Fxi) = \sum_{xi} P(xi, Fxi) log[P(xi, Fxi)/P(xi)P(Fxi)]$$

- El peso (exactitud) total está dado por la suma de los pesos de cada nodo:

$$W = \sum_{i} w(xi, Fxi)$$

Búsqueda usando MDL

- Se puede hacer un *hill-climbing* iniciando con una estructura simple, por ejemplo un árbol construido con Chow-Liu (o compleja – altamente conectada), agregando (o eliminando) ligas que mejoren la medida MDL hasta alcanzar un mínimo local
- Algoritmo - búsqueda de la mejor estructura:
  1. Generar estructura incial - árbol (o multiconectada)
  2. Calcular medida de calidad de la estructura inicial
  3. Agregar (eliminar) / invertir un arco en la estructura actual
  4. Calcular medida de calidad de la nueva estructura
  5. Si se mejora la calidad conservar el cambio, si no dejar la estructura anterior
  6. Repetir 3 a 5 hasta que ya no haya mejoras.
- También se pueden combinar los dos enfoques

Métodos basados en Pruebas de Independiencia

- Usa medidas de dependencia local entre subconjuntos de variables
- El caso más sencillo es el del algoritmo de Chow y Liu (información mutua entre pares de variables)
- En general, se hacen pruebas de dependencia entre subconjuntos de variables, normalmente dos o tres variables
- La desventaja es que pueden generarse muchos arcos "innecesarios", por lo que se incorporan formas para luego eliminar arcos
- Hay diferentes variantes de este enfoque que consideran diferentes medidas de dependencia y diferentes estrategias para eliminar arcos innecesarios

Algoritmo de PC

- PC obtiene el esqueleto (grafo no dirigido) y después las orientaciones de los arcos
- Para el esqueleto empieza con un grafo no dirigido completamente conectado y determina la independencia condicional de cada par de variables dado un subconjunto de otras variables $I(X, Y \mid \mathbf{S})$
- Se puede obtener con una medida de entropia condicional cruzada y si el valor es menor a un umbral se elimina el arco.
- La dirección se obtiene buscando estructuras de la forma $X - Z - Y$ sin arco en $X - Y$. Si X, Y no son independientes dado Z, orienta los arcos creando una estructura "V": $X \rightarrow Z \leftarrow Y$.
- Al terminar trata de otientar el resto basado en pruebas de independencia y evitando ciclos

Algoritmo de PC

Require: Set of variables **X**, Independence test I

Initialize a complete undirected graph G'
i=0
repeat
  for $X \in \mathbf{X}$ **do**
    for $Y \in ADJ(X)$ **do**
      for $S \subseteq ADJ(X) - \{Y\}, \mid S \mid = i$ **do**
        if $I(X, Y \mid S)$ **then**
          Remove the edge $X - Y$ from G'
        end if
      end for
    end for
  end for
  i=i + 1
until $\mid ADJ(X) \mid \leq i, \forall X$
Orient edges in G'

Algoritmo K2

Require: Conjunto n de variables ordenadas y número máximo de padres por nodo (u)

1: **for** $i := 1$ hasta n **do**
2: $\pi_i = \emptyset$
3: $P_{old} := f(i, \pi_i)$ {Ver fórmula}
4: *Bandera* := *true*
5: **while** *Bandera* and $|\pi_i| < u$ **do**
6: sea $z \in Pred(x_i) - \pi_i$ que maximice $f(i, \pi_i \cup \{z\})$
7: $P_{new} := f(i, \pi_i \cup \{z\})$
8: **if** $P_{new} > P_{old}$ **then**
9: $P_{old} := P_{new}$; $\pi_i := \pi_i \cup \{z\}$
10: **else**
11: *Bandera* := *false*
12: **end if**
13: **end while**
14: escribe: 'Los padres de: ' x_i 'son:' π_i
15: **end for**

Algoritmo K2

$$f(i,\pi_i) = \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} \alpha_{ijk}!$$

donde:

- π_i = padres de x_i
- $q_i = |\phi_i|$, donde ϕ_i = posibles instanciaciones de los padres de x_i
- $r_i = |V_i|$, donde V_i = posibles valores de x_i
- α_{ijk} = número de ejemplos en que el nodo x_i tiene el valor k y sus padres tienen el valor j
- $N_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$ = ejemplos donde los padres de x_i toman valores j

Otras Estrategias

- El encontrar la estructura óptima es difícil, ya que el espacio de búsqueda es muy grande (e.g., más de 10^{40} estructuras para 10 variables)
- Una alternativa es combinar conocimiento de expertos con datos
- Otra alternativa es obtener una estructura inicial a partir de datos y luego utilizar conocimiento del experto
- Se puede:
  1. *Eliminar un nodo*
  2. *Combinar nodos*
  3. *Crear un nodo*
- Finalmente se puede hacer *transfer learning* (utilizar lo aprendido en un dominio parecido para facilitar el aprendizaje en el dominio actual)