Aprendizaje Bayesiano

Índice

- Introducción
- Teorema de Bayes
- Hipótesis MAP y ML
- Algoritmos MAP
- Principio MDL
- Clasificador óptimo

Introducción

- ¿Por qué estudiar métodos bayesianos?
  - Tienen gran aplicación práctica
  - Son competitivos con otros métodos conocidos [redes neuronales, árboles de decisión, etc.]
  - Permiten caracterizar otros métodos que no utilizan la probabilidad de forma explícita

Introducción

- Caracterísiticas de los métodos bayesianos:
  - Cada caso de entrenamiento cambia la probabilidad estimada de que una hipótesis sea correcta.
  - El conocimiento previo puede ser utilizado para determinar la probabilidad de una hipótesis.
  - Pueden dar predicciones probabilísticas.
  - Pueden clasificar nuevas instancias combinando probabilísticamente distintas hipótesis.
  - Se precisa conocer varias probabilidades
  - Los algoritmos tienen un costo alto

Teorema de Bayes

- Podemos caracterizar la ‘mejor’ hipótesis como la hipótesis más probable dados los datos.
- Esto es, buscamos obtener las hipótesis de H que maximizan P(h|D).
- Las hipótesis que cumplen esto son llamadas **M**aximum **A** **P**osteriori [hipótesis MAP]

Teorema de Bayes

- El teorema de Bayes nos permite obtener la probabilidad a posteriori de una hipótesis:

$$P(h|D) * P(D) = P(D|h) * P(h)$$

- Aplicándolo a nuestro problema:

$$h_{MAP} \equiv \text{argmax}_{h \in H} P(h|D)$$
$$\equiv \text{argmax}_{h \in H} P(D|h) * P(h) / P(D)$$
$$\equiv \text{argmax}_{h \in H} P(D|h) * P(h)$$

- Si las hipótesis son equiprobables:

$$h_{ML} \equiv \text{argmax}_{h \in H} P(D|h)$$

Teorema de Bayes

- Por ejemplo:

P(cancer) = 0.008
P(test ⊕ | cancer) = 0.98
P(test ⊕ | ¬cancer) = 0.03

P(¬cancer)=0.992
P(test ⊗ | cancer) =0.02
P(test ⊗ | ¬cancer) =0.97

- Si el test da positivo: ¿qué deberíamos diagnosticar?

P(test ⊕| cancer) * P(cancer) = 0.98 * 0.008 = 0.0078
P(test ⊕|¬cancer) * P(¬cancer) = 0.03 * 0.992 = 0.0298

- El diagnóstico más probable es que el paciente está sano.
- Esto lo puedo afirmar con una seguridad del 79%
  [0.0298/ (0.0298+0.0078)]

Algoritmos MAP

- ¿Cuál es la aplicación del teorema de Bayes al Aprendizaje Conceptual?
- Podemos buscar h_{MAP} en H.

- **Algoritmo Fuerza-Bruta** 
  - Para cada hipótesis h de H, calculamos:
    - $P(h|D) = P(D|h) * P(h) / P(D)$
  - Damos como salida $h_{MAP} \equiv \text{argmax}_{h \in H} P(h|D)$

- No es una buena opción cuando H es grande.

Algoritmos MAP

- Para aplicar el algoritmo debemos calcular las probabilidades P(D|h) y P(h)
- Estas probabilidades se pueden elegir de acuerdo a los conocimientos previos que tengamos sobre el espacio de búsqueda
- Por ejemplo, supongamos que:
  - El conjunto D no tiene ruido.
  - El concepto objetivo está en nuestro espacio H.
  - En principio, todas las hipótesis son equiprobables:

$$P(h) = |H|^{-1}$$

Algoritmos MAP

- ¿Cómo estimamos P(D|h)?

$$P(D|h) = \begin{cases} 1 & \text{si } d_i = h(x_i)\ \forall d_i \text{ en } D \\ 0 & \text{en caso contrario} \end{cases}$$

- En otras palabras: evaluamos en 1 si la hipótesis es consistente con D y 0 en caso contrario (no hay datos con ruido).

Algoritmos MAP

- ¿Cuál es la probabilidad a posteriori de las hipótesis?
- Si considero una hipótesis inconsistente con los datos, su probabilidad es nula

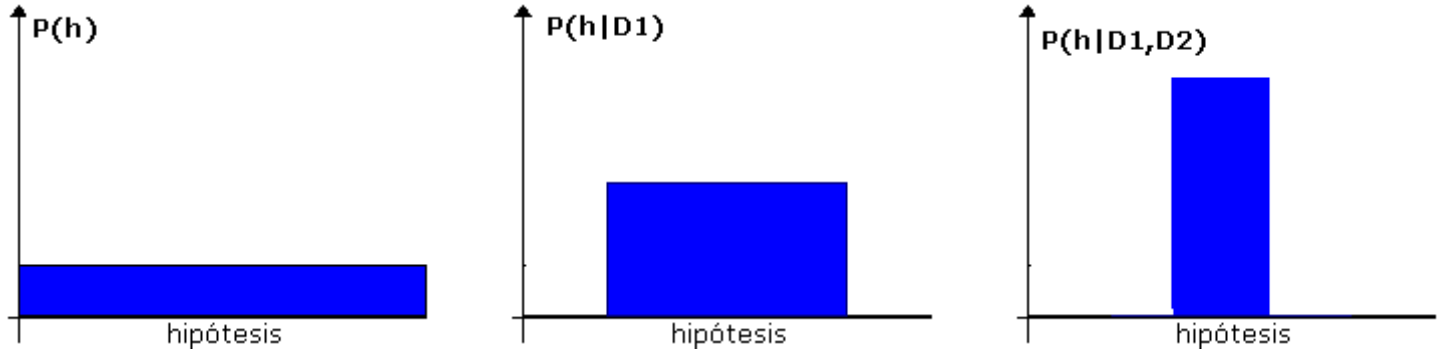
$$P(h|D) = 0 * P(h) / P(D) = 0$$

- En cambio, si es consistente:

$$P(h|D) = 1 * |H|^{-1} / (|VS_{H,D}|/|H|)$$
$$= |VS_{H,D}|^{-1}$$

Algoritmos MAP

- Por lo tanto, toda hipótesis consistente con el conjunto de entrenamiento es MAP.



Algoritmos MAP

- Todo algoritmo [consistente] da como resultado una hipótesis MAP, si asumimos una distribución a priori uniforme sobre H.
- **Find-S** y **Candidate-Elimination** no manejan ningún tipo de probabilidad y sin embargo son algoritmos MAP.
- ¿Existen otras condiciones bajo las cuales **Find-S** sea un algoritmo MAP?
- Sí. Cuando la distribución sobre H asigna más probabilidad a las hipótesis más específicas y no hay ruido en la entrada.

Algoritmos MAP

- En definitiva, podemos caracterizar los algoritmos aun cuando estos no utilizan explícitamente probabilidades.
- Para esto, basta encontrar P(h) y P(D|h) bajo las cuales los algoritmos dan hipótesis MAP.
- Esta es una alternativa al sesgo para caracterizar los supuestos bajo los cuales un algoritmo aprende.

Principio MDL

- Veamos la 'Navaja de Occam' desde un punto de vista bayesiano.
- Para esto, vamos a definir el Principio del 'Largo Mínimo de Descripción' [**MDL**]
- Volviendo a la def. de h_{MAP} :

$$h_{MAP} \equiv \text{argmax}_{h \in H} P(D|h) * P(h)$$
$$\equiv \text{argmin}_{h \in H} -\log P(D|h) - \log P(h)$$

- Esto se puede interpretar como que se debe preferir las hipótesis más cortas.

Principio MDL

- Para minimizar un código se debe elegir para el mensaje i-ésimo el largo $-(\log p_i)$ siendo p_i la probabilidad de aparición de este mensaje

- Entonces:
  - $L_{CH}(h) = -\log P(h)$: es el largo de la descripción de h dentro del codificación óptima de H (CH).
  - $L_{CD/h}(D|h) = -\log P(D|h)$: es el largo de la codificación del conjunto de entrenamiento si vale h, bajo la mejor codificación (CD|h).

- $h_{MAP} = \text{argminh } L_{CH}(h) + L_{CD/h}(D|h)$

Principio MDL

- Pero en la práctica la codificación ya está determinada [C1=codif.hip,C2=codif.Dat].

- Principio MDL:

$$h_{MAP} = \text{argmin}_h \, L_{C1}(h) + L_{C2}(D|h)$$

- Apliquémoslo a los árboles de decisión...

Principio MDL

- Para C_1 elegimos una codificación que asigne códigos más largos a árboles con más nodos.
- ¿Qué codificación elegimos para C_2 dada C_1?
- Supongamos que los ejemplos son conocidos por un emisor y su receptor, y lo único que se debe transmitir son $<f(x_1)....>$
- Si la hipótesis clasifica perfectamente los ejemplos, no hay nada para transmitir, pero con seguridad el árbol sea más grande.

Principio MDL

- En cambio, cuando algún ejemplo es clasificado erróneamente hay que transmitir un mensaje donde se identifica el error y su valor correcto.
- El principio MDL nos permite balancear la complejidad del árbol vs. los errores que se cometen
- Las aplicaciones de este principio, dieron resultados similares a los algoritmos 'clásicos'.
- ¿Podemos afirmar entonces que Occam tenía razón?

Principio MDL

- La respuesta es negativa, ya que la justificación bayesiana sólo es válida cuando estamos ante las codificaciones óptimas.

- Para determinar estas codificaciones precisamos saber

P(h) y P(D|h).

- Por lo general se busca una especificación que nos parezca mejor... y aplicamos el algoritmo

Clasificador bayesiano óptimo

- Sabemos determinar la(s) hipótesis más probable(s) dado un conjunto de entrenamiento.

- Pero, ¿es la hipótesis más probable las que nos da la clasificación más probable de una nueva instancia?

Clasificador bayesiano óptimo

- La respuesta es negativa.

- Por ejemplo:

$P(h_1|D) = 0.4$ $\quad$ $P(h_2|D)=0.3$

$P(h_3|D) = 0.3$

$h_1(x)= \oplus$ $\quad$ $h_2(x)= \otimes$ $\quad$ $h_3(x)= \otimes$

- Uno tiende a pensar que el valor más probables es $\otimes$ [con 60% de seguridad]

Clasificador bayesiano óptimo

- En general, el valor más probable se obtiene combinando los resultados de todas las hipótesis

$$P(v|D)=\Sigma_{h\in H}P(v|h)\ P(h|D)$$

- Clasificación bayesiana óptima:

$$\text{argmax}_{v\in V}\Sigma_{h\in H}P(v|h)\ P(h|D)$$

Clasificador bayesiano óptimo

- Por ejemplo:

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

$P(\oplus|D) = \Sigma_{h \in H} P(\oplus|h)\ P(h|D) = 0.4$
$P(\otimes|D) = \Sigma_{h \in H} P(\otimes|h)\ P(h|D) = 0.6$

$\text{argmax}_{v \in \{\oplus, \otimes\}} \Sigma_{h \in H} P(v|h)\ P(h|D) = \otimes$

Clasificador bayesiano óptimo

- Un Clasificador bayesiano óptimo es cualquier sistema que clasifique las instancias de esta manera.

- Estos sistemas maximizan la probabilidad de clasificar correctamente nuevas instancias dado el conjunto de entrenamiento y las probabilidades a priori de las hipótesis.

Clasificador bayesiano óptimo

- Un clasificador óptimo es, por ejemplo, tomar el resultado de la votación entre las hipótesis del espacio de versiones resultante del Candidate-Elimination

- Esto lleva a que la clasificación obtenida no tenga porque tener una hipótesis que la represente dentro del espacio H.

- La desventaja es que calcular las prioridad a posteriori con cada hipótesis puede ser muy costoso.

Algoritmo de Gibbs

- **Algoritmo de Gibbs:**
  - Elija una hipótesis h al azar según la distribución a posteriori que gobierna H
  - Use h para predecir el valor de la instancia

- Se puede probar que, si se toma h según la distribución a priori, el error del clasificador de Gibbs es a lo sumo el doble que el óptimo.

- Si supongo hipótesis equiprobables, y tomo una hipótesis del VS al azar, ¡el resultado tiene a lo sumo el doble de error que el clasificador óptimo!

Clasificador sencillo

- Consideremos instancias de la forma $<a_1...a_n>$, y una función objetivo f que toma valores sobre un conjunto finito V.

- Buscamos:

  $v = \text{argmax}_{vj \in V} P(v_j | a_1...a_n)$

  $= \text{argmax}_{vj \in V} P(a_1...a_n | v_j) * P(v_j) / P(a_1...a_n)$

  $= \text{argmax}_{vj \in V} P(a_1...a_n | v_j) * P(v_j)$

- ¿Cómo estimamos estos valores a partir del conjunto de entrenamiento?

Clasificador sencillo

- Podemos utilizar la frecuencia de aparición en el conjunto de entrenamiento para $P(v_j)$
- Calcular $P(a_1...a_n|v_j)$ no siempre es posible
- Suponiendo que los atributos son independientes entre sí.

$$v_{NB} = \text{argmax}_{vj \in V} \prod_i P(a_i|v_j) * P(v_j)$$

- Notar que no hay búsqueda en H.

Clasificador sencillo

- Por ejemplo:

#	Tiempo	Temp	Humedad	Viento	Juega
1	Soleado	Caluroso	Alta	Suave	No
2	Soleado	Caluroso	Alta	Fuerte	No
3	Nuboso	Caluroso	Alta	Suave	Sí
4	Lluvioso	Templado	Alta	Suave	Sí
5	Lluvioso	Frío	Normal	Suave	Sí
6	Lluvioso	Frío	Normal	Fuerte	No
7	Nuboso	Frío	Normal	Fuerte	Sí
8	Soleado	Templado	Alta	Suave	No
9	Soleado	Frío	Normal	Suave	Sí
10	Lluvioso	Templado	Normal	Suave	Sí
11	Soleado	Templado	Normal	Fuerte	Sí
12	Nuboso	Templado	Alta	Fuerte	Sí
13	Nuboso	Caluroso	Normal	Suave	Sí
14	Lluvioso	Templado	Alta	Fuerte	No

- Buscamos clasificar la instancia:

<soleado, frío, alta, fuerte>

Clasificador sencillo

- Utilizando el calsificador:

$$
\begin{aligned}
v_{NB} &= \text{argmax}_{vj \in V} \prod_i P(a_i|v_j) * P(v_j) = \\
&= \text{argmax}_{vj \in \{sí,no\}} P(\text{cielo=soleado}|v_j) \\
&\quad * P(\text{temp=frío}|v_j) \\
&\quad * P(\text{hum=alta}|v_j) \\
&\quad * P(\text{viento=fuerte }|v_j) * P(v_j)
\end{aligned}
$$

- Ahora, aproximamos las 10 probabilidades que precisamos utilizando la frecuencia de aparición en la tabla

Clasificador sencillo

- Haciendo cuentas:

  P(jugar=sí) = 9/14 = 0.64
  P(jugar=no) = 5/14 = 0.36
  P(Viento=fuerte|jugar=sí) = 3/9 = 0.33
  P(Viento=fuerte|jugar=no) = 3/5 = 0.60
  ...

  P(soleado|sí) * P(frío|sí) * P(fuerte|sí) * P(sí) =0.0053
  P(soleado|no) * P(frío|no) * P(fuerte|no) * P(no)=0.0206

- Con un 79.5% de seguridad puedo afirmar que la respuesta es negativa.

Clasificador sencillo

- No siempre es buena la aproximación utilizando la frecuencia e/n [e=número de veces que ocurre el evento, n=número de oportunidades]

- Cuando no hay ejemplos para algunos casos, la probabilidad asignada es cero [con lo cual "anula" todo el término del clasificador]

- **m-estimador**

$$[e + m*p] / [n + m]$$

p es la estimación a priori de la probabilidad buscada y **m** es el "tamaño equivalente de muestra".

- La fórmula puede verse como aumentar la muestra con m ejemplos, distribuidos según p (si equiprobables, p=1/k)

Resumiendo...

- Los métodos bayesianos me permiten obtener la hipótesis óptima [probabilísticamente]
- El clasificador óptimo me permite determinar el valor más probable de una nueva instancia a partir de las probabilidades de TODAS las hipótesis
- El clasificador 'sencillo' me ahorra el cálculo de todas las probabilidades a cambio de supuestos de independencia

Resumiendo...

- Puedo caracterizar algoritmos que no usan explícitamente probabilidades aplicando un 'razonamiento bayesiano'

- El principio MDL recomienda seleccionar la hipótesis que minimiza el largo de su descripción más los datos si ella se cumple.