Introducción al Aprendizaje automático

Dra Ana Georgina Flesia

Optativa Ciencias de la Computación
FaMAF-UNC
Oficina 370
georgina.flesia@unc.edu.ar

2020

Agenda

- Recordamos FIND-S
- Espacio de versiones
- Algoritmo de eliminación de candidatos

Aprendizaje de conceptos

Aprendizaje por inferencia inductiva

Se busca inferir la definición de un concepto a partir de instancias rotuladas como pertenecientes y no pertenecientes al concepto.

Para ello se aproxima una función booleana que representa al concepto a partir del conjunto de instancias de entrenamiento

Notación y terminología

Población

Conjunto X de todas las instancias posibles dadas las variables medidas

Objetivo

Un concepto es representado por $c : X \to \{1, 0\}$

- $c(x) = 1$, x instancia positiva, satisface el concepto.
- $c(x) = 0$ es una instancia negativa.

Muestra

$D \subset X$ muestra de entrenamiento etiquetada.

Ejemplo de aprendizaje de concepto por inferencia inductiva:

Concepto

Días en los cuales se puede practicar un deporte acuático

Objetivo

Dado un día nuevo, con ciertas características, queremos dar una respuesta de SI si se puede practicar deporte o NO si no se puede.

Muestra

Tenemos un conjunto de ejemplos de días SI y días NO, y los valores que tomaron las características en esos días.

Ejemplo de aprendizaje de concepto por inferencia inductiva:

- Las seis variables que se estudian por día son:
  - `Cielo`: sol, nublado, lluvia
  - `Temperatura del aire`: templada, fría
  - `Humedad Ambiente`: normal, alta
  - `Viento`: débil, fuerte
  - `Temperatura del agua`: templada, fría
  - `Pronóstico`: igual, cambio
- Instancias de entrenamiento:

Cielo	Aire	Humedad	Viento	Agua	Pronóstico	Deporte	c
sol	templada	normal	fuerte	templada	igual	si	1
sol	templada	alta	fuerte	templada	igual	si	1
lluvia	frío	alta	fuerte	templada	cambio	no	0
sol	templada	alta	fuerte	fría	cambio	si	1

Aprendizaje de lo general a lo especifico

Aprendizaje de conceptos puede ser visto como la tarea de buscar dentro de un gran espacio de hipótesis implícitamente definidas por la representación del concepto

El objetivo de la búsqueda es encontrar la hipótesis que mejor ajusta a las instancias de entrenamiento.

Aprendizaje de lo general a lo especifico

Espacio de Hipótesis H

Es un conjunto de funciones $h : X \rightarrow \{0, 1\}$ que representan restricciones a las variables categóricas medidas.

Restricciones posibles

$\emptyset$, ?, valor

- $\emptyset$ indica que ningún valor de la variable es aceptable
- ? indica que cualquier valor de la variable es aceptable.
- `valor` es uno de los valores posibles de la variable.

Objetivo del aprendizaje

Se busca una hipótesis h tal que para cualquier instancia etiquetada $< x, c(x) > \in D$ se tenga $h(x) = c(x)$.

Aprendizaje de conceptos

Si una instancia x satisface todos las restricciones de la hipótesis h, entonces h clasifica x como positivo $h(x) = 1$.

- Hipótesis mas específica: $h = \{< \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset >\}$ dice que ninguna es positiva.
- Hipótesis mas general: $h = \{<?, ?, ?, ?, ?, ? >\}$ dice que todas las instancias son positivas.
- $h_2 = \{< Soleado, ?, ?, ?, ?, ? >\}$ es más general que $h_1 = \{< Soleado, ?, ?, Fuerte, ?, ? >\}$, porque impone menos restricciones en una instancia que h_1.

Orden de generalidad

- Dadas $h_1, h_2 \in H$, h_1 es mas general que h_2, ($h_1 \geq_g h_2$) sí y solo sí cualquier instancia que satisface h_2 satisface h_1.

$$(\forall x \in X)\,[(h_2(x) = 1) \rightarrow (h_1(x) = 1)]$$

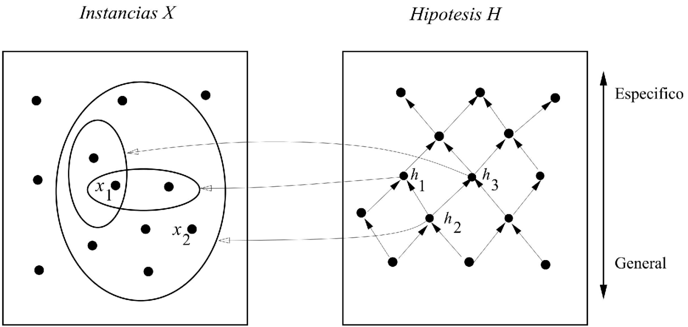
- Dadas $h_1, h_2 \in H$, h_2 es más específica que h_1, ($h_2 \leq_g h_1$) sí y solo toda instancia que satisface h_2 satisface h_1.

Consistencia

Una hipótesis h es consistente con D si y solo si h coincide con c en la muestra D

$$\text{Consistente } (h, D) \equiv (\forall (x, c(x)) \in D) \;\; h(x) = c(x)$$

Ejemplo:



x_1 = <*Soleado, Templada, Alta, Fuerte, Fria, Igual*>
x_2 = <*Soleado, Templada, Alta, Suave, Templada, Igual*>

h_1 = <*Soleado, ?, ?, Fuerte, ?, ?*>
h_2 = <*Soleado, ?, ?, ?, ?, ?*>
h_3 = <*Soleado, ?, ?, ?, Fria, ?*>

Algoritmo Find S

Pseudocodigo:

1. Inicialmente, h es la hipótesis mas específica de H.
2. Por cada ejemplo positivo x del conjunto de entrenamiento:
   - Si $h(x) = 1$, no hacer nada.
   - En otro caso, reemplazar h por la menor generalización h' de h, tal que $h'(x) = 1$
3. Devolver h

Find S

- Empieza por $h_0 = \{< \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset >\}$
- Toma el primer x positivo y reemplaza el anterior por esta instancia

$$h_1 = \{< sol, templada, normal, fuerte, templada, igual >\}$$

- Toma el segundo positivo $x = ($ sol, templada, alta, fuerte, templada, igual $)$ y generaliza h_1 por

$$h_2 = \{< sol, templada, ?, fuerte, templada, igual >\}$$

- Toma el tercer positivo $x = ($ sol, templada, alta, fuerte, fría, cambio $)$ y generaliza

$$h_3 = \{< sol, templada, ?, fuerte, ?, ? >\}$$

el cual es el resultado, la aproximación del concepto c.

Comentarios sobre Find S

- Find-S encuentra una hipótesis de máxima especificidad que es consistente con todas las instancias positivas de la muestra.
- Las instancias negativas se ignoran.
- ¿ Resulta consistente con los ejemplos negativos?
  - Sí, si el concepto objetivo esta en H (expresividad del espacio de hipótesis) y los ejemplos de entrenamiento son correctos (ausencia de ruido)
  - Ejemplo problemático
    Positivos: $< Sol, Templ, Fuerte >, < Lluvia, Frio, Fuerte >$
    Negativos: $< Luvia, Templ, Fuerte >$
    La hipótesis resultante es

    $$h = \{<?, ?, Fuerte>\}$$

    la cual clasifica al negativo como positivo, por lo cual h no es consistente con la muestra D.
- Otros algoritmos buscan hipótesis consistentes con D.

Espacio de versiones

- Espacio de versiones: $VS_{H,D} \equiv \{h \in H | h \text{ es consistente con } D\}$
- Ejemplo

Cielo	Aire	Humedad	Viento	Agua	Pronóstico	Deporte
sol	templada	normal	fuerte	templada	igual	si
sol	templada	alta	fuerte	templada	igual	si
lluvia	frío	alta	fuerte	templada	cambio	no
sol	templada	alta	fuerte	fría	cambio	si

Espacio de versiones del ejemplo, esto es todas las hipótesis consistentes con D son

$< Soleado, Templada, ?, Fuerte, ?, ? >< Soleado, ?, ?, Fuerte, ?, ? >$
$< Soleado, Templada, ?, ?, ?, ? ><?, Templada, ?, Fuerte, ?, ? >$
$< Soleado, ?, ?, ?, ?, ? ><?, Templada, ?, ?, ?, ? >$

- Pueden ordenarse en capas, de la mas general a la mas específica.

Espacio de versiones

- El espacio de versiones es una representación jerárquica del conocimiento que permite seguir la información útil presente en las instancias sin recordarlas a todas.
- Permite manejar muchos modelos en el mismo espacio.
- Representa todas las hipótesis consistentes con la muestra.
- El algoritmo List-Them-Eliminate encuentra el espacio de versiones consistentes con D, pero no es eficiente porque enumerar espacios es poco realista aún en los espacios mas triviales.
- El algoritmo Candidate Elimination representa al espacio de versiones con los conjuntos de las hipótesis de máxima generalidad y máxima especificidad sin necesidad de listar. Permite definir la aproximación a c uniendo la información de todas estas hipótesis.

Definiciones

Máxima generalidad

$h \in H$ es una hipótesis de máxima generalidad de H si no existe $h' \in H$ tal que $h' \geq_g h$

Máxima especificidad

$h \in H$ es una hipótesis de máxima especificidad de H si no existe $h' \in H$ tal que $h \geq_g h'$

Cota general G respecto de D

elementos de máxima generalidad del espacio de versiones.

Cota especifica S respecto de D

elementos de máxima especificidad del espacio de versiones (Find-S).

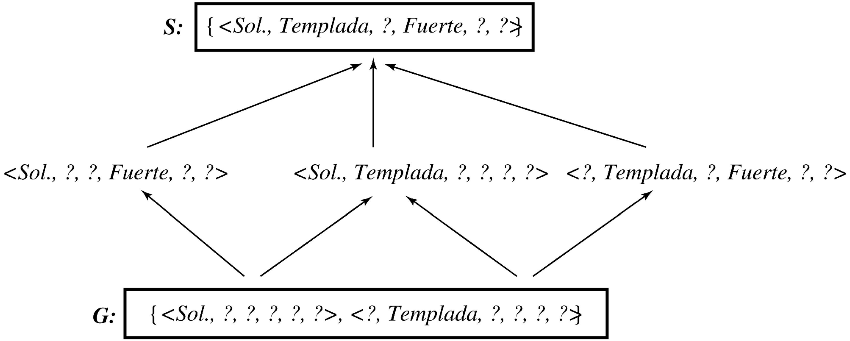
Una representación compacta del espacio de versiones $VS_{H,D}$

Teorema de representación del espacio de versiones $VS_{H,D}$

El espacio de versiones $VS_{H,D}$ es el conjunto de todas las hipótesis que están entre la cota general G y la cota específica S correspondientes a la muestra "D.

$$VS_{H,D} = \{h \in H | (\exists s \in S)(\exists g \in G)\,(g \geq_g h \geq_g s)\}$$

Ejemplo:


S:
{<Sol., Templada, ?, Fuerte, ?, ?>}
<Sol., ?, ?, Fuerte, ?, ?>
<Sol., Templada, ?, ?, ?, ?>
<?, Templada, ?, Fuerte, ?, ?>
G:
{<Sol., ?, ?, ?, ?, ?>, <?, Templada, ?, ?, ?, ?>}

Algoritmo de eliminación de candidatos

Sean G y S los conjunto elementos de máxima generalidad y máxima especificidad de H, respectivamente. Para cada instancia x del conjunto de entrenamiento D:

- Si x es una instancia positiva, entonces:
  - Eliminar de G cualquier hipótesis inconsistente con x.
  - Para cada hipótesis s de S inconsistente con x:
    - Eliminar s de S.
    - Incluir en S todas las generalizaciones minimales h de s, tales que h es consistente con x y existe una hipótesis en G mas general que h.
    - Eliminar de S aquellas hipótesis tales que exista en S otra hipótesis mas general.

Algoritmo de eliminación de candidatos

- Si x es un ejemplo negativo, entonces:
  - Eliminar de S cualquier hipótesis inconsistente con x.
  - Para cada hipótesis g de G inconsistente con x:
    - Eliminar g de G.
    - Incluir en G todas las especializaciones minimales h de g, tales que h es consistente con x y existe una hipótesis en S mas especifica que h.
    - Eliminar de G aquellas hipótesis tales que exista en G otra hipótesis mas especifica.

Algoritmo de eliminación de candidatos

- Paso 0: $S0 = \{< \emptyset, \emptyset, \emptyset, \emptyset, \emptyset >\}$, $G0 = \{<?, ?, ?, ?, ?, ? >\}$
- Paso 1:
  - Ejemplo positivo: $< Sol, Templ, Normal, Fuerte, Templ, Igual >$
  - Nada que eliminar de $G0$
  - Generalización minimal de $S0$:
    $\{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
    Esta generalización es mas especifica que la hipótesis de $G0$
  - Luego:

$S1 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$,
$G1 = \{<?, ?, ?, ?, ?, ? >\}$

Algoritmo de eliminación de candidatos

$S1 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$,
$G1 = \{<?, ?, ?, ?, ?, ? >\}$

- Paso 2:
  - Ejemplo positivo: $< Sol, Templ, Alta, Fuerte, Templ, Igual >$
  - Nada que eliminar de $G1$
  - Generalizacion minimal de $S1$:
    $\{< Sol, Templ, ?, Fuerte, Templ, Igual >\}$
    Esta generalización es mas específica que la hipótesis de $G1$
  - Luego:

  $S2 = \{< Sol, Templ, ?, Fuerte, Templ, Igual >\}$
  $G2 = \{<?, ?, ?, ?, ?, ? >\}$

Algoritmo de eliminación de candidatos

$S2 = \{< Sol, Templ, ?, Fuerte, Templ, Igual >\}$
$G2 = \{<?, ?, ?, ?, ?, ? >\}$

- Paso 3:
  - Ejemplo negativo: $< Lluvia, Fria, Alta, Fuerte, Templ, Cambio >$
  - Nada que eliminar de $S2$
  - Especializaciones minimales de $G2$ que son mas generales que la hipótesis de $S2$ y satisfacen x : $< Sol, ?, ?, ?, ?, ? >; <?, Templ, ?, ?, ?, ? >; <?, ?, ?, ?, ?, Igual >$.
  - Luego:

$S3 = \{< Sol, Templ, ?, Fuerte, Templ, Igual >\}$
$G3 = \{< Sol, ?, ?, ?, ?, ? >; <?, Temp, ?, ?, ?, ? >; <?, ?, ?, ?, ?, Igual >\}$

Algoritmo de eliminación de candidatos

$S3 = \{< Sol, Templ, ?, Fuerte, Templ, Igual >\}$
$G3 = \{< Sol, ?, ?, ?, ?, ? >; <?, Temp, ?, ?, ?, ? >; <?, ?, ?, ?, ?, Igual >\}$

- Paso 4:
  - Ejemplo positivo: $< Sol, Templ, Alta, Fuerte, Fria, Cambio >$
  - Eliminamos de $G3$ la hipótesis $<?, ?, ?, ?, ?, Igual >$
  - Generalización minimal de $S3 : \{< Sol, Templ, ?, Fuerte, ?, ? >\}$ Esta generalización es mas específica que hipótesis de $G3$.
  - Luego:

$S4 = \{< Sol, Templ, ?, Fuerte, ?, ? >\}$
$G4 = \{< Sol, ?, ?, ?, ?, ? >; <?, Templ, ?, ?, ?, ? >\}$

Clasificación de nuevas instancias

- Usamos S y G obtenidos por eliminación de candidatos para clasificar nuevas instancias:
- Si es consistente con todo S, positivo
  - Si no es consistente con ninguno de G, negativo
  - En otro caso, voto mayoritario o simplemente no se clasifica
- Ejemplos:

  $< Sol, Templ, Normal, Fuerte, Fria, Cambio > \rightarrow Si$

  $< Lluvia, Fria, Normal, Debil, Templ, Igual > \rightarrow No$

  $< Sol, Fria, Normal, Fuerte, Templ, Igual > \rightarrow No$ (por mayoría)

  $< Sol, Templ, Normal, Debil, Templ, Igual > \rightarrow Si$ (por mayoría)

Propiedades

- Sean S y G obtenidos por eliminación de candidatos.
  - Si S y G son no vacíos, resultan ser respectivamente la cota especifica y cota general del espacio de versiones (respecto del conjunto de entrenamiento).
  - Si $S = G = \{h\}$, entonces h es la única hipótesis de H consistente con todos los ejemplos.
  - Si $S = G = \emptyset$, no existe $h \in H$ consistente con los ejemplos
- Convergencia hacia el concepto objetivo, siempre que:
  - Conjunto de entrenamiento suficientemente grande
  - Ejemplos sin errores (ausencia de ruido)
  - El concepto objetivo esta en H

Qué ocurrre si D contiene errores?

Cielo	Aire	Humedad	Viento	Agua	Pronóstico	Deporte	c
sol	templada	normal	fuerte	templada	igual	si	1
sol	templada	alta	fuerte	templada	igual	no	0
lluvia	frío	alta	fuerte	templada	cambio	no	0
sol	templada	alta	fuerte	fría	cambio	si	1

Algoritmo de eliminación de candidatos

- Paso 0: $S0 = \{< \emptyset, \emptyset, \emptyset, \emptyset, \emptyset >\}$, $G0 = \{<?, ?, ?, ?, ?, ? >\}$
- Paso 1:
  - Ejemplo positivo: $< Sol, Templ, Normal, Fuerte, Templ, Igual >$
  - Nada que eliminar de $G0$
  - Generalización minimal de $S0$
    $\{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
    Esta generalización es mas especifica que la hipótesis de $G0$
  - Luego:

$S1 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
$G1 = \{<?, ?, ?, ?, ?, ? >\}$

Algoritmo de eliminación de candidatos

$S1 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
$G1 = \{<?, ?, ?, ?, ?, ? >\}$

- Paso 2:
  - Ejemplo negativo: $< Sol, Templ, Alta, Fuerte, Templ, Igual >$
  - Nada que eliminar de $S1$ pues da negativa la instancia
  - Especializaciones minimales de $G1$ que son mas generales que la hipótesis de $S1$: $\{<?, ?, Normal, ?, ?, ? >\}$
  - Luego:

  $S2 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
  $G2 = \{<?, ?, ?, Normal, ?, ? >\}$

Algoritmo de eliminación de candidatos

$S2 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
$G2 = \{<?, ?, ?, Normal, ?, ? >\}$

- Paso 3:
  - Ejemplo negativo: $< Lluvia, Fria, Alta, Fuerte, Templ, Cambio >$
  - Nada que eliminar de $S2$
  - Especializaciones minimales de $G2$ que son mas generales que la hipótesis de $S2$: $\{< Sol, ?, Normal, ?, ?, ? >; <?, Templ, Normal, ?, ?, ? >; < ?, ?, Normal, ?, ?, Igual >\}$
  - Luego:

$S3 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
$G3 = \{< Sol, ?, Normal, ?, ?, ? >; <?, Templ, Normal, ?, ?, ? >; < ?, ?, Normal, ?, ?, Igual >\}$

Algoritmo de eliminación de candidatos

$S3 = \{< Sol, Templ, Normal, Fuerte, Templ, Igual >\}$
$G3 = \{< Sol, ?, Normal, ?, ?, ? >; < ?, Templ, Normal, ?, ?, ? >; < ?, ?, Normal, ?, ?, Igual >\}$

- Paso 4:
  - Ejemplo positivo: $< Sol, Templ, Alta, Fuerte, Fria, Cambio >$
  - Generalización minimal de $S3 :< Sol, Templ, ?, Fuerte, Fria, Cambio >$
  - $G3$ no es consistente

$S4 = \emptyset$ $G4 = \emptyset$

Qué ocurrre si hay un concepto presente pero no lo halla?

Cielo	Aire	Humedad	Viento	Agua	Pronóstico	Deporte
sol	templada	normal	fuerte	templada	igual	si
nublado	templada	normal	fuerte	templada	igual	si
lluvia	templada	normal	fuerte	templada	igual	no

El concepto objetivo es aprendido solo cuando S y G convergen a la misma hipótesis y esta es única.