



INSTITUTO TECNOLÓGICO BELTRÁN  
Centro de Tecnología e Innovación

## Algoritmos de Machine Learning

### “Algoritmo FOIL”



Alumno = Nicolas Mesquiatti

Materia = Aprendizaje Automático

## Algoritmo de FOIL

### 3. Algoritmo de FOIL

Ejercicio 1: Identificar empleados en formación Este ejercicio permite practicar cómo inducir reglas lógicas simples a partir de diferencias entre ejemplos positivos y negativos, tal como lo hace el algoritmo FOIL en su forma más básica.

Se tiene un conjunto de datos con personas que trabajan en una empresa. Algunas están en formación (aprendices) y otras no.

Cada persona tiene atributos como:

Edad, departamento, nivel\_educativo, en\_formacion (booleano: True o False)

#### **Dataset de ejemplo:**

```
datos = [{"edad": 22, "departamento": "IT", "nivel_educativo":  
"terciario", "en_formacion": True}, {"edad": 24, "departamento": "IT",  
"nivel_educativo": "universitario", "en_formacion": True}, {"edad": 21,  
"departamento": "RRHH", "nivel_educativo": "terciario",  
"en_formacion": True}, {"edad": 35, "departamento": "IT",  
"nivel_educativo": "universitario", "en_formacion": False}, {"edad": 40,  
"departamento": "Finanzas", "nivel_educativo": "maestría",  
"en_formacion": False}, {"edad": 29, "departamento": "RRHH",  
"nivel_educativo": "universitario", "en_formacion": False}, {"edad": 23,  
"departamento": "IT", "nivel_educativo": "terciario", "en_formacion":  
True}, {"edad": 38, "departamento": "Finanzas", "nivel_educativo":  
"universitario", "en_formacion": False}]
```

#### **Objetivo del ejercicio:**

1. Separar los ejemplos positivos (en\_formacion == True) y negativos (en\_formacion == False).

2. Para cada atributo, identificar los valores que aparecen en los positivos, pero no en los negativos.

3. Desarrollar un programa en Python que muestre la Regla inducida para identificar a personas en formación:

Regla inducida para identificar a personas en formación: Respuestas a las preguntas guía:

¿Qué valores de departamento aparecen solo en los positivos?

- Rta : Ninguno, Ya que IT también aparece en algún caso en falso

¿Qué niveles educativos son comunes en los positivos, pero no en los negativos?

- Rta : El nivel educativo común en los positivos y no en los negativos es el nivel terciario

¿Qué edades aparecen solo en los positivos?

- Rta : 21,22,23 y 24

Ejercicio 2:

a) Desarrollar un programa en Python que obtenga el cálculo del FOIL Gain para la condición `nivel_educativo == 'terciario'`, con la siguiente salida:

Condición: `nivel_educativo == 'terciario'`

$P(\text{positivos antes}) = 4$

$N(\text{negativos antes}) = 4$

$p(\text{positivos después}) = 3$

$n(\text{negativos después}) = 0$

$p / (p + n) = 1.000$

$P / (P + N) = 0.500$

$\log_2(p / (p + n)) = 0.000$

$\log_2(P / (P + N)) = -1.000$

FOIL Gain = 3.000



```
43 # Mostrar la regla
44 print("Regla inducida para identificar a alguien en formación:")
45 for atributo, valores in regla_inducida.items():
46     print(f"- {atributo} debe ser uno de: {valores}")
47
48 # Valores antes de aplicar la condición
49 P = sum(1 for d in datos if d["en_formacion"])
50 N = sum(1 for d in datos if not d["en_formacion"])
51 # Aplicar condición: departamento == "IT"
52 filtrados = [d for d in datos if d["nivel_educativo"] == "terciario"]
53 p = sum(1 for d in filtrados if d["en_formacion"])
54 n = sum(1 for d in filtrados if not d["en_formacion"])
55
56 # Cálculo FOIL Gain
57 def log2_safe(x):
58     return math.log2(x) if x > 0 else float('-inf')
59 foil_gain = p * (log2_safe(p / (p + n)) - log2_safe(P / (P + N)))
60 # Mostrar resultados
61 print(f"P = {P}, N = {N}")
62 print(f"p = {p}, n = {n}")
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS D:\Tecnicas De Procesamiento del habla> & C:\Users\nikit\AppData\Local\Programs\Python\
del habla/Materia2-Machine learning/FOIL.py"
Regla inducida para identificar a alguien en formación:
- edad debe ser uno de: [24, 21, 22, 23]
- nivel_educativo debe ser uno de: ['terciario']
P = 4, N = 4
p = 3, n = 0
p / (p + n) = 1.000
P / (P + N) = 0.500
log2(p / (p + n)) = 0.000
log2(P / (P + N)) = -1.000
```

Me dio el mismo resultado al evaluar la condición nivel\_educativo == 'terciario'

b) Realizar el calculo de FOIL Gain manual para comprobar los cálculos.

Condición: nivel\_educativo == 'terciario'

Valores utilizados:

Término Valor

P (positivos antes) 4

N (negativos antes) 4

p (positivos después) 3

n (negativos después) 0

1. Fracción después de aplicar la condición:

$$p / (p + n) = 3 / (3+0) = 3/3 = 1$$

2. Fracción antes de aplicar la condición:

$$P / (P + N) = 4 / (4+4) = 4/8 = 0,5$$

3. Logaritmos:

$$\text{Log2: } p / (p + n) = 3 / (3+0) = 3/3 = 1 \rightarrow \log_2(1) = 0,000$$

$$\text{Log2: } P / (P + N) = 4 / (4+4) = 4/8 = 0,5 \rightarrow \log_2(0,5) = -1,000$$

4. FOIL Gain:

$$p \times (\text{Log2: } p / (p + n) - \text{Log2: } P / (P + N))$$

$$3 \times (0,000 - (-1,000))$$

$$3 \times 1,000 = \mathbf{3.000}$$

5. Interpretación:

Una regla con 3 positivos y 0 negativos, significa una regla bastante precisa y la ganancia de información es bastante significativa (3,000)

c) Desarrollar un programa en Python que obtenga el cálculo del FOIL Gain para la condición edad  $\leq 23$ , con la siguiente salida:

$$P \text{ (positivos antes)} = 4$$

$$N \text{ (negativos antes)} = 4$$

$$p \text{ (positivos después)} = 3$$

$$n \text{ (negativos después)} = 0$$

$$p / (p + n) = 1.000$$

$$P / (P + N) = 0.500$$

$$\log_2(p / (p + n)) = 0.000$$

$$\log_2(P / (P + N)) = -1.000$$

$$\text{FOIL Gain} = 3.000$$

```
FOIL.py X
Materia2-Machine learning > FOIL.py > ...

92 #Ejercicio 3, C)
93 # Valores antes de aplicar la condición
94 P = sum(1 for d in datos if d["en_formacion"])
95 N = sum(1 for d in datos if not d["en_formacion"])
96 # Aplicar condición: edad == "23"
97 filtrados = [d for d in datos if d["edad"] <= 23]
98 p = sum(1 for d in filtrados if d["en_formacion"])
99 n = sum(1 for d in filtrados if not d["en_formacion"])
100
101 # Cálculo FOIL Gain
102 def log2_safe(x):
103     return math.log2(x) if x > 0 else float('-inf')
104 foil_gain = p * (log2_safe(p / (p + n)) - log2_safe(P / (P + N)))
105 # Mostrar resultados
106 print(f"P = {P}, N = {N}")
107 print(f"p = {p}, n = {n}")
108 print(f"p / (p + n) = {p / (p + n):.3f}")
109 print(f"P / (P + N) = {P / (P + N):.3f}")
110 print(f"log2(p / (p + n)) = {log2_safe(p / (p + n)):.3f}")
111 print(f"log2(P / (P + N)) = {log2_safe(P / (P + N)):.3f}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

log2(p / (p + n)) = -0.415
log2(P / (P + N)) = -1.000
FOIL Gain = 1.755
P = 4, N = 4
p = 3, n = 0
p / (p + n) = 1.000
P / (P + N) = 0.500
log2(p / (p + n)) = 0.000
log2(P / (P + N)) = -1.000
FOIL Gain = 3.000
PS D:\Tecnicas De Procesamiento del habla>
```

d) Realizar el cálculo de FOIL Gain manual para comprobar los cálculos.  
Condición: edad  $\leq$  23 Valores utilizados: Término

P (positivos antes) Valor 4

N (negativos antes) 4

p (positivos después) 3

n (negativos después) 0

1. Fracción después de aplicar la condición:

$$p / (p + n) = 3 / (3+0) = 3/3 = 1$$

2. Fracción antes de aplicar la condición:

$$P / (P + N) = 4 / (4+4) = 4/8 = 0,5$$

3. Logaritmos:

$$\text{Log2: } p / (p + n) = 3 / (3+0) = 3/3 = 1 \rightarrow \log_2(1) = 0,000$$

$$\text{Log2: } P / (P + N) = 4 / (4+4) = 4/8 = 0,5 \rightarrow \log_2(0,5) = -1,000$$

4. FOIL Gain:

$$p \times (\text{Log2: } p / (p + n) - \text{Log2: } P / (P + N))$$

$$3 \times (0,000 - (-1,000))$$

$$3 \times 1,000 = \mathbf{3.000}$$

5. Interpretación

Como con la regla anterior también hay 3 positivos y 0 negativos, es eficaz y precisa, con buena ganancia de información.

## Condición extra

Departamento = "IT"

```
FOIL.py X
Materia2-Machine learning > FOIL.py > ...
70 #Condición 2: departamento == "IT"
71 # Valores antes de aplicar la condición
72 P = sum(1 for d in datos if d["en_formacion"])
73 N = sum(1 for d in datos if not d["en_formacion"])
74 # Aplicar condición: departamento == "IT"
75 filtrados = [d for d in datos if d["departamento"] == "IT"]
76 p = sum(1 for d in filtrados if d["en_formacion"])
77 n = sum(1 for d in filtrados if not d["en_formacion"])
78
79 # Cálculo FOIL Gain
80 def log2_safe(x):
81     return math.log2(x) if x > 0 else float('-inf')
82 foil_gain = p * (log2_safe(p / (p + n)) - log2_safe(P / (P + N)))
83 # Mostrar resultados
84 print(f"P = {P}, N = {N}")
85 print(f"p = {p}, n = {n}")
86 print(f"p / (p + n) = {p / (p + n):.3f}")
87 print(f"P / (P + N) = {P / (P + N):.3f}")
88 print(f"log2(p / (p + n)) = {log2_safe(p / (p + n)):.3f}")
89 print(f"log2(P / (P + N)) = {log2_safe(P / (P + N)):.3f}")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

FOIL Gain = 3.000
P = 4, N = 4
p = 3, n = 1
p / (p + n) = 0.750
P / (P + N) = 0.500
log2(p / (p + n)) = -0.415
log2(P / (P + N)) = -1.000
FOIL Gain = 1.755
P = 4, N = 4
p = 3, n = 0
p / (p + n) = 1.000
```

## Conclusión

Al aplicar esta regla me da una ganancia distinta a los dos anteriores

Ya que ahora tengo 3 p y 1 n después de aplicar la condición Gracias a eso el FOIL gain es de 1,755, Como la mitad entre las otras dos condiciones, esto quiere decir que es mucho mejor aplicar las otras dos condiciones antes que esta ya que ambas eliminan los negativos y eso genera una mayor ganancia de información