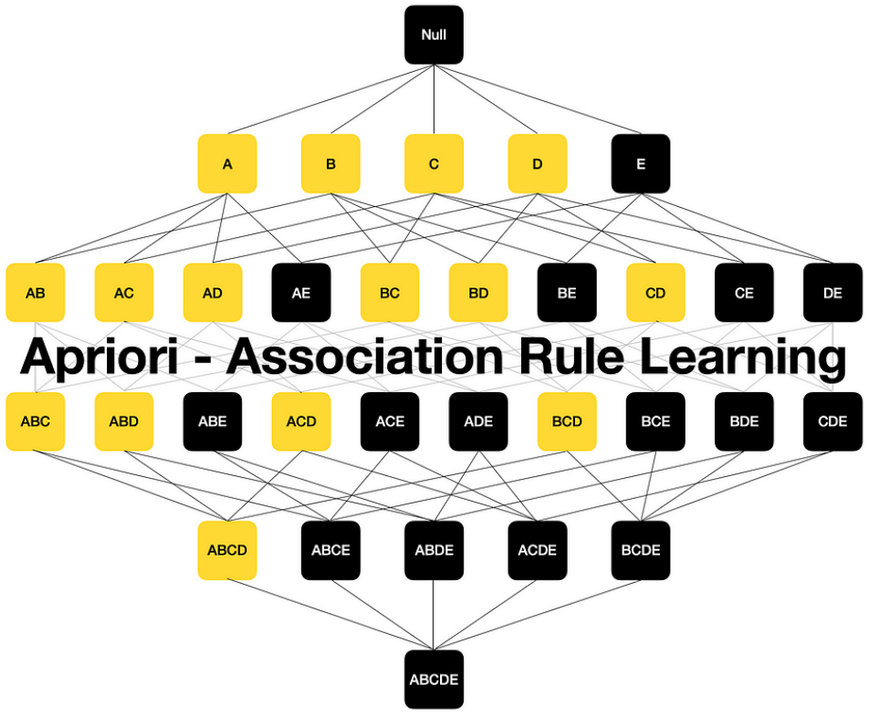
UNIVERSIDAD DE CUENCA
desde 1867

Reglas de Asociación

Andres Auquilla

2024

Content



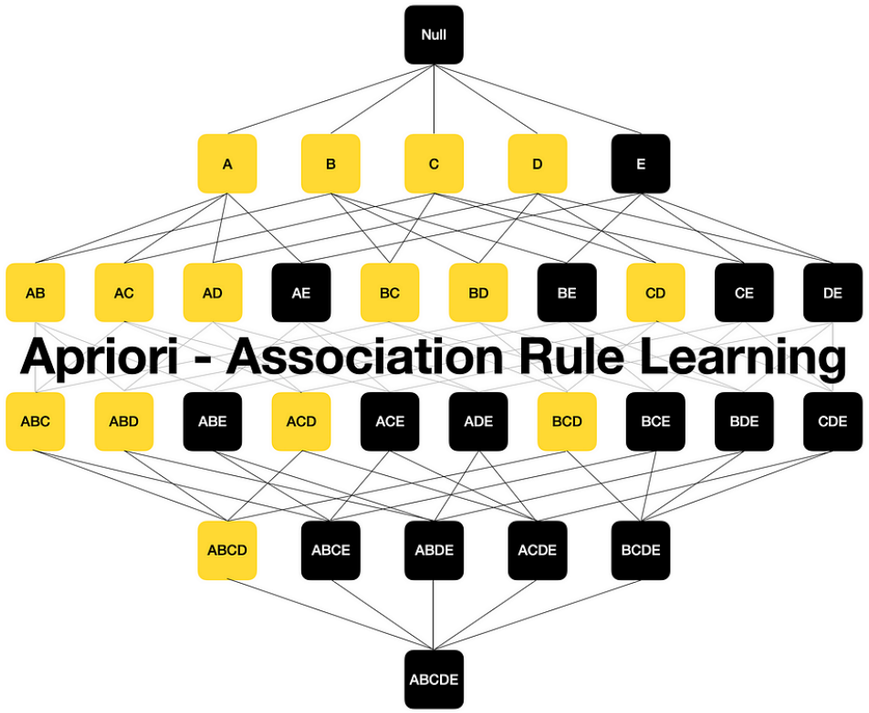
Inducción de conjuntos de reglas

Generalización a través de reglas

APRIORI

Reglas de asociación

Content



Inducción de conjuntos de reglas

Generalización a través de reglas

APRIORI

Reglas de asociación

Otra forma de representar teorías
es usando Reglas de Decisión

Anteriormente revisado: Arboles de decisión

Puede representar conceptos conjuntivos

Otra forma popular es reglas if-then

IF <condición> THEN “pertenece al concepto”

¿Como se pueden aprender esas reglas?

Usando árboles, algoritmos genéticos, algoritmos especiales

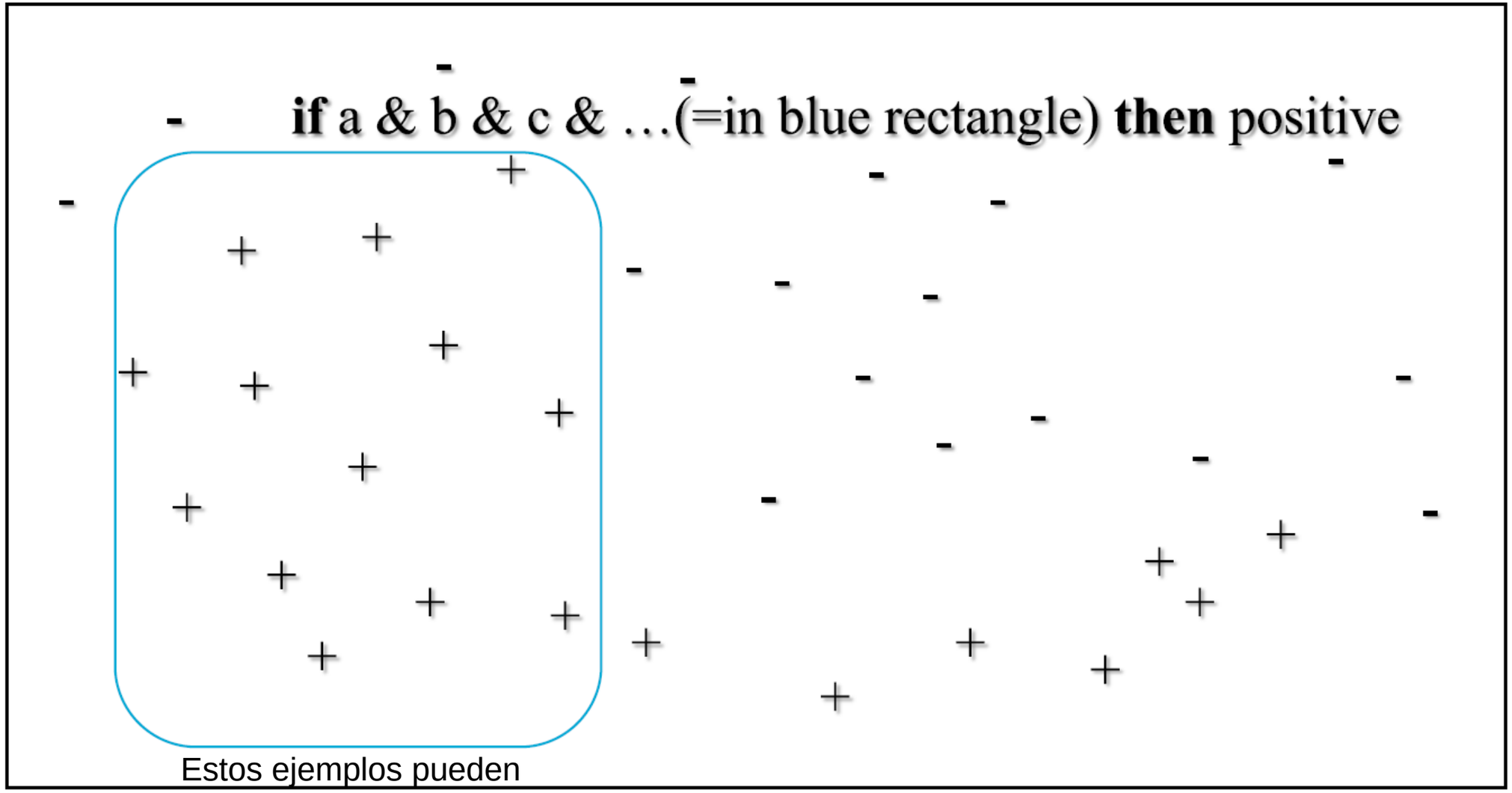
Un método popular para generar reglas es el Sequential Covering

También denominado “separate and conquer”

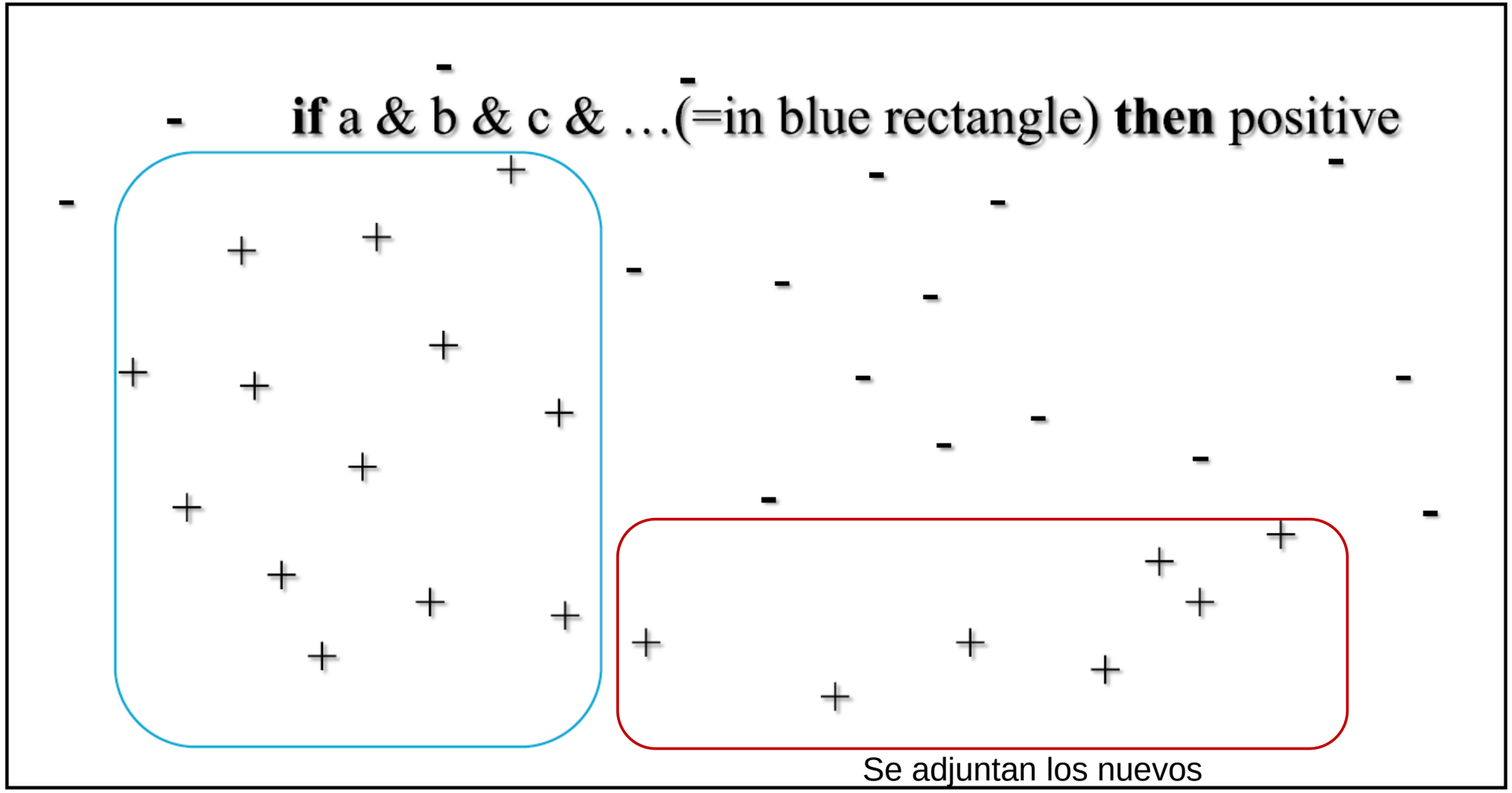
- Analogía con árboles -> “divide and conquer”

Principio general: aprender un conjunto de reglas, una a la vez

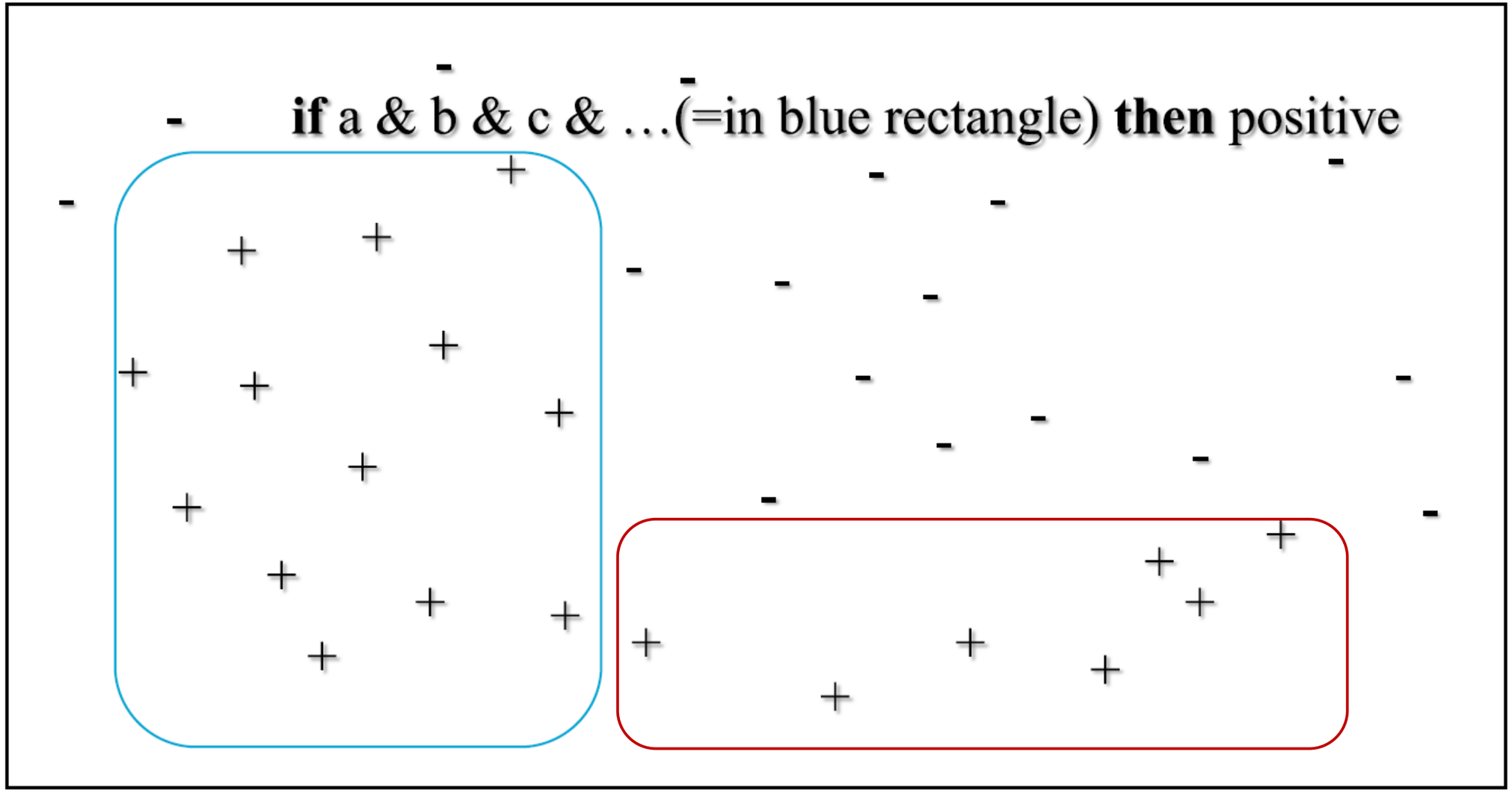
- Aprender una regla con alta precisión
- Puede hacer predicciones sobre algunos ejemplos
- Una vez que los ejemplos fueron cubiertos, centrarse en los otros
- Repetir el proceso hasta que se hayan cubierto todos los ejemplos



Estos ejemplos pueden ser olvidados ahora



Se adjuntan los nuevos ejemplos



Todos los ejemplos fueron incluidos: Terminar

Algoritmo general para aprender conjuntos de reglas

```
function LearnRuleSet(Target, Attrs, Examples, Threshold):
    LearnedRules := ∅
    Rule := LearnOneRule(Target, Attrs, Examples)
    while performance(Rule,Examples) > Threshold, do
        LearnedRules := LearnedRules ∪ {Rule}
        Examples := Examples \ {examples classified correctly by Rule}
        Rule := LearnOneRule(Target, Attrs, Examples)
    sort LearnedRules according to performance
    return LearnedRules
```

Aprender una regla individualmente

Se utiliza greedy search

Top-down

- Comenzar con la regla más general (máxima cobertura, pero baja presición)
- Agregue literales uno a uno
- Gradualmente maximice la precisión sin sacrificar cobertura

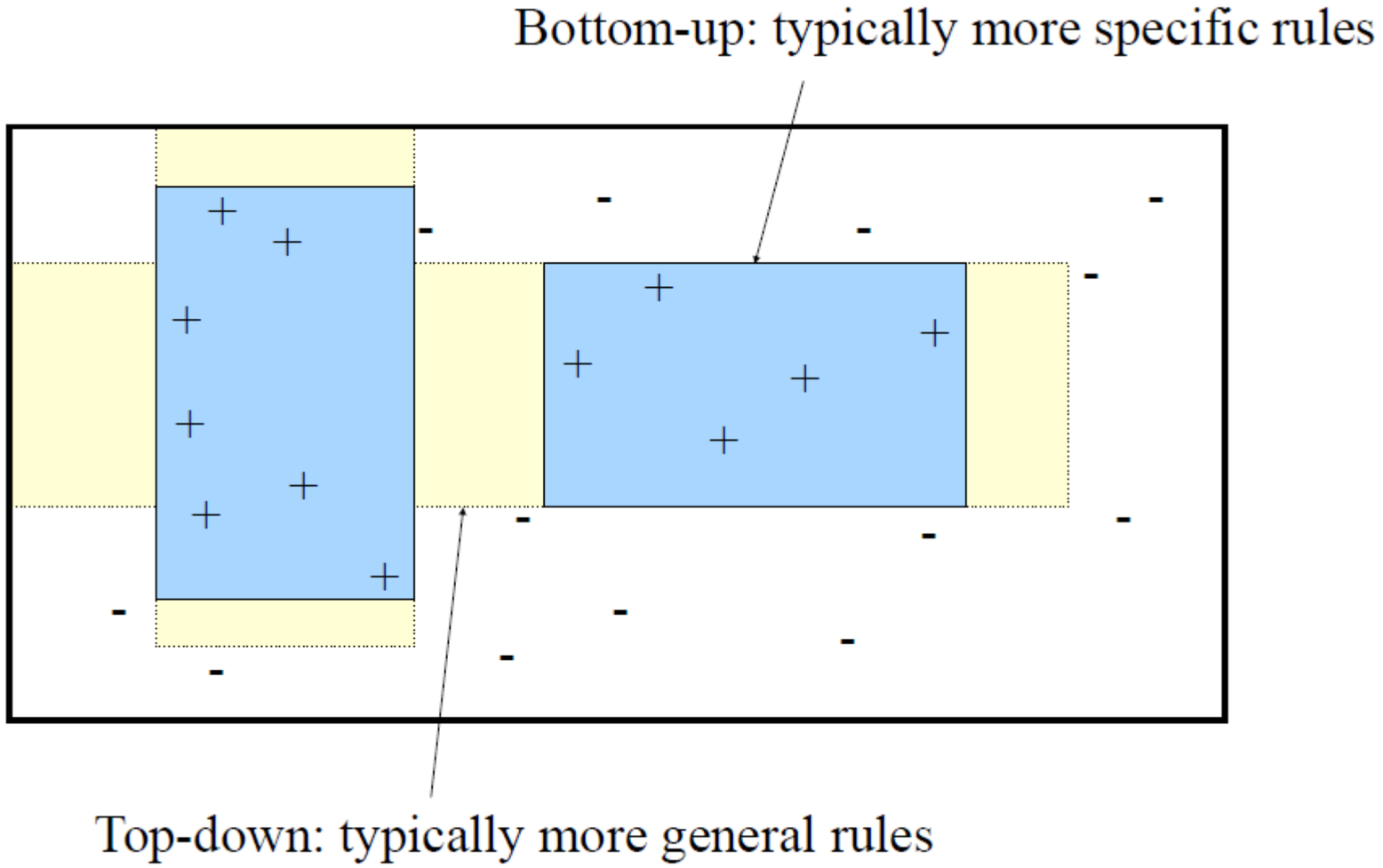
Botton-up

- Comenzar con una regla muy específica (cobertura mínima pero máxima presición)
- Remover literales uno a uno
- Gradualmente maximizar la cobertura sin sacrificar la presición

```
function LearnOneRule(Target, Attrs, Examples):
    NewRule := "IF true THEN pos"
    NewRuleNeg := Neg
    while NewRuleNeg not empty, do
        // add a new literal to the rule
            Candidates := generate candidate literals
            BestLit := argmax_{L∈Candidates} performance(Specialise(NewRule,L))
            NewRule := Specialise(NewRule, BestLit)
            NewRuleNeg := {x∈Neg | x covered by NewRule}
    return NewRule

function Specialise(Rule, Lit):
    let Rule = "IF conditions THEN pos"
    return "IF conditions and Lit THEN pos"
```

Bottom-up vs. Top-down



La determinación de la calidad de una regla se la realiza mediante heurísticas

Como se determina si una regla es “buena”

- Alta precisión y covertura

Funciones de evaluación

- Precisión: $p/(p+n)$
- Entropía: más simetría entre p y n

Post-pruning

- Misma idea que en árboles de desición

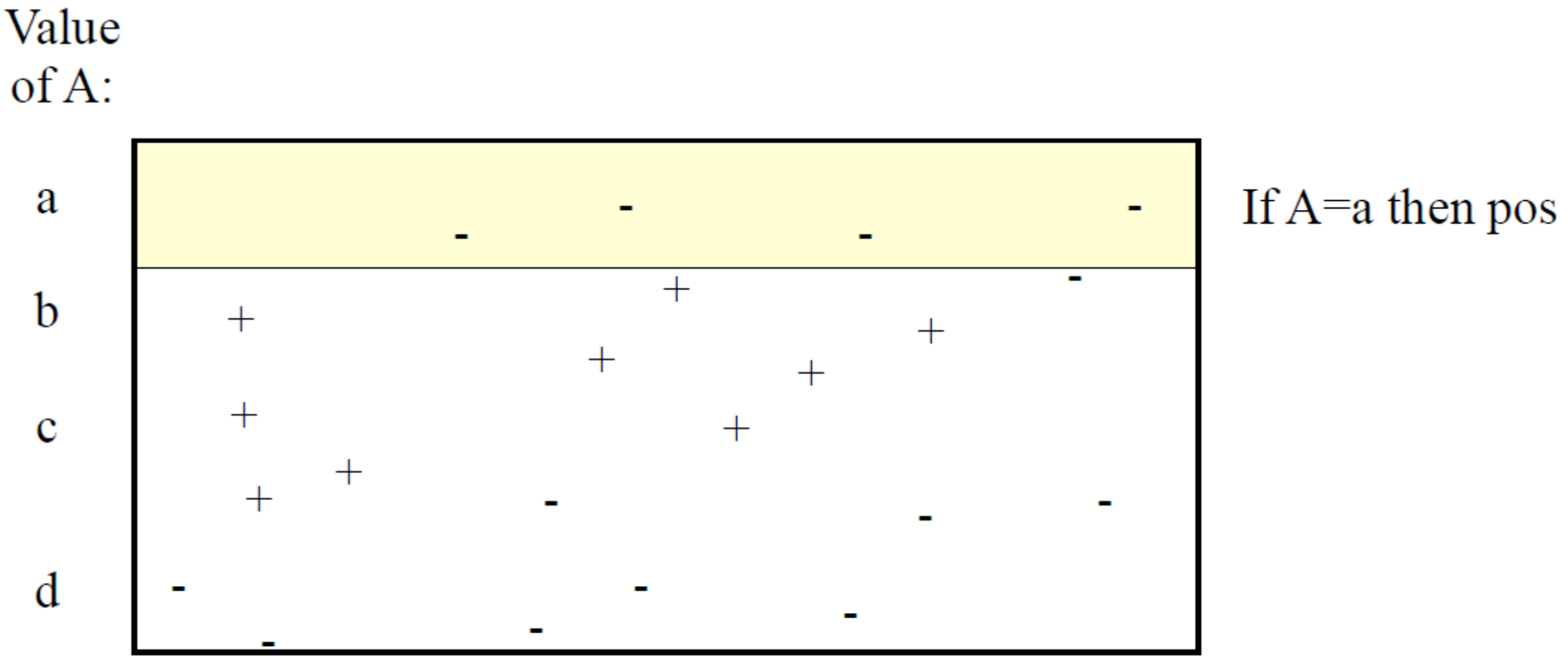
Algoritmo AQ (Michalski et al.): top-down

Para una clase C (hasta que se cubran todos los ejemplos)

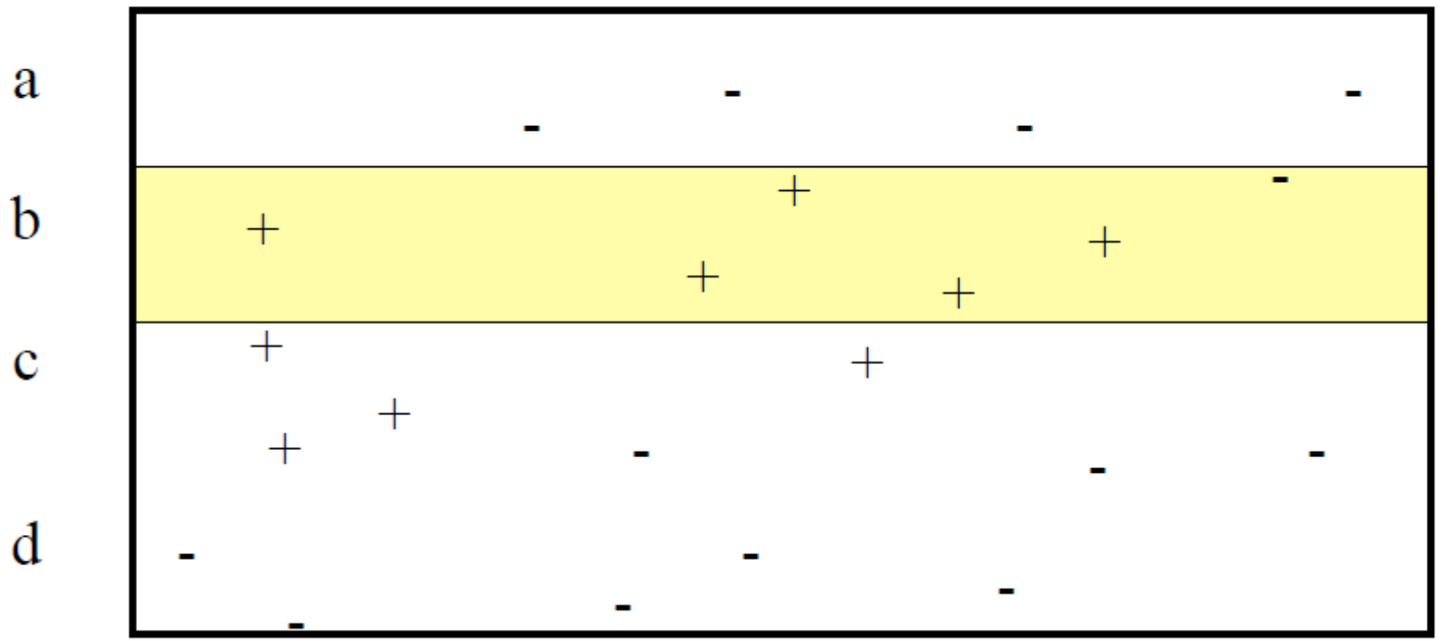
- Seleccionar un ejemplo e
- Considerar $H_e = \{reglas\ que\ cubren\ el\ ejemplo\}$
- Buscar top-down en H_e para encontrar la mejor regla

Es una búsqueda eficiente: H_e es mucho más pequeña q H

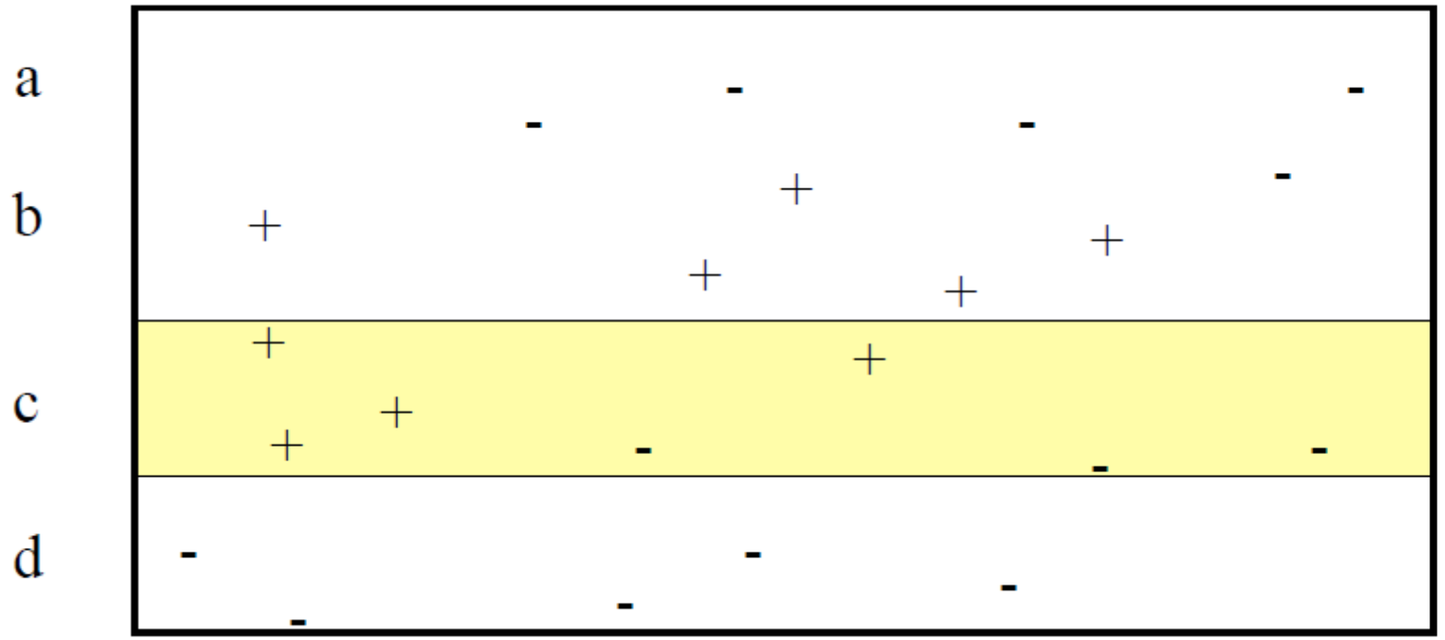
No es robusto con respecto al ruido



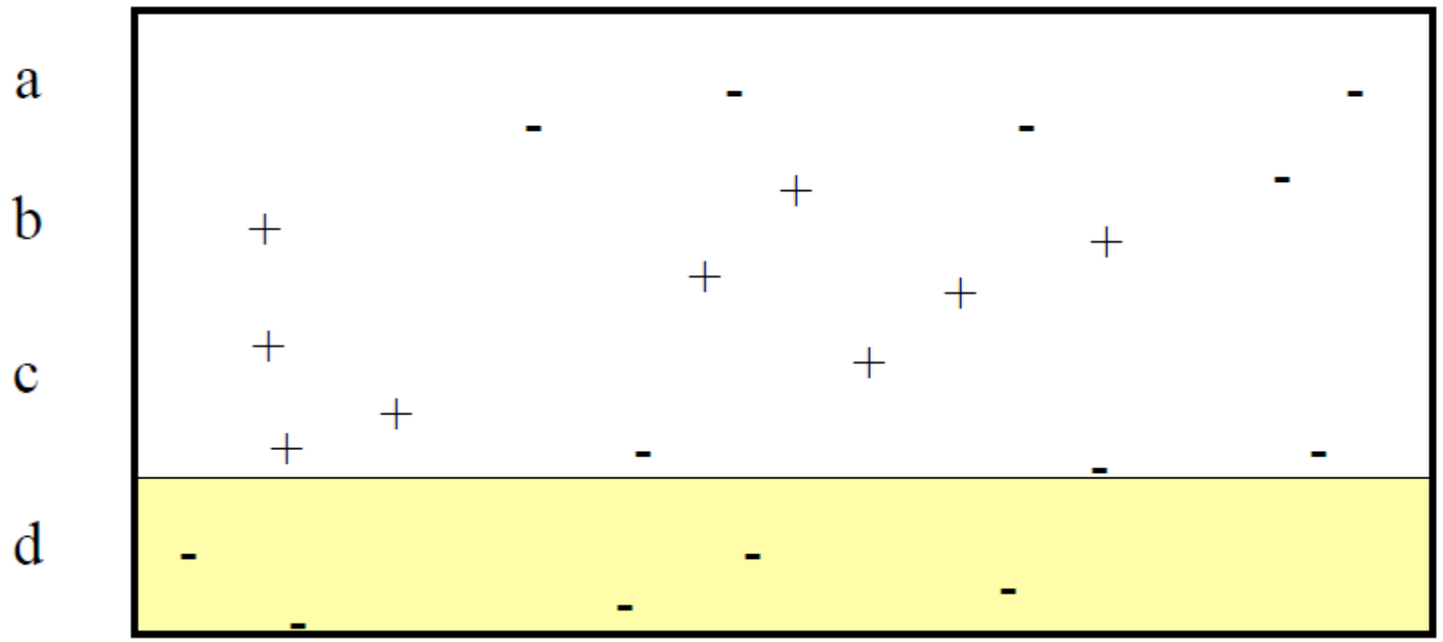
Looking for a good rule in the format “IF A=... THEN pos”



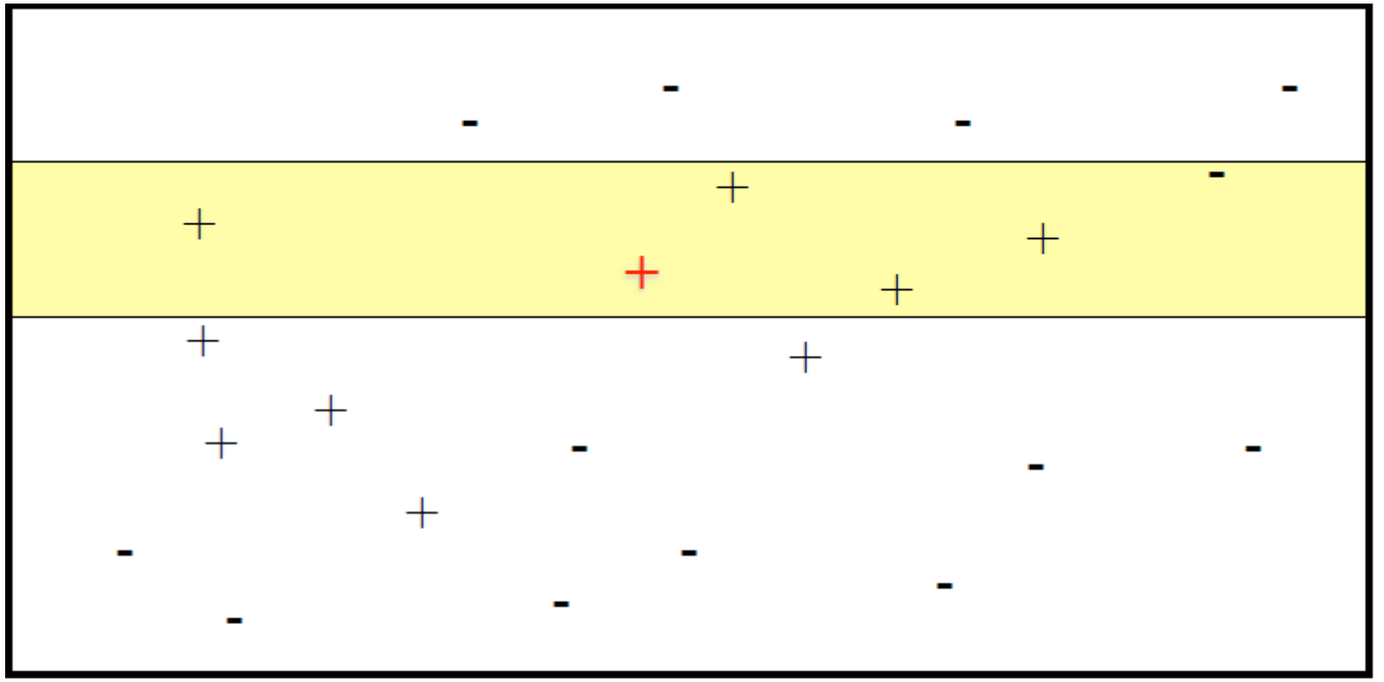
If A=b then pos



If A=c then pos



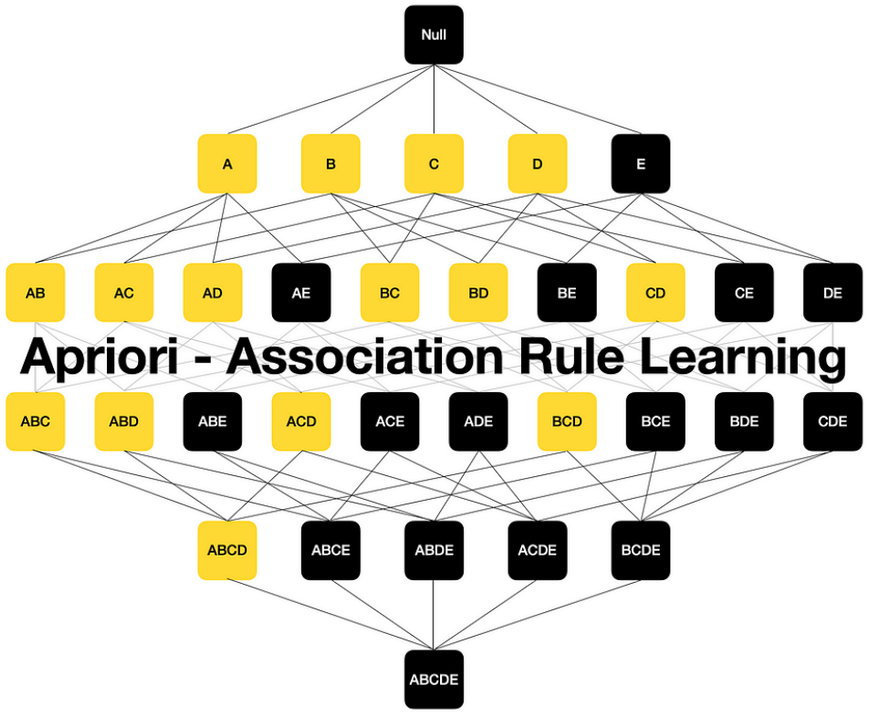
If A=d then pos



If A=b then pos

Try only rules that cover the seed "+" which has A=b.
Hence, A=b is a reasonable test, A=a is not.
We do not try all 4 alternatives in this case! Just one.

Content



Inducción de conjuntos de reglas

Generalización a través de reglas

APRIORI

Reglas de asociación

Las reglas de asociación describen relaciones entre conjuntos de atributos boléanos

Muy utilizadas en “basket analysis”, e.j. que productos se compran juntos

Client	cheese	bread	butter	wine	jam	ham
1	yes	yes	yes	yes	no	yes
2	yes	no	yes	no	no	no
3	no	yes	yes	no	no	yes
...	...	...	...	...	...	...

```
IF bread & butter THEN cheese
  confidence: 50%
  support: 5%
```

Las reglas de asociación buscan correlaciones entre elementos transaccionales de una base de datos

Diversas aplicaciones

- Soporte para la toma de decisiones
- Diagnóstico en sistemas industriales
- Análisis de información de ventas
- Segmentación de clientes por sus patrones de compra

Las reglas de asociación se parecen a las reglas de clasificación

Se obtienen mediante un procedimiento de covering

Pares atributo - valor

Enumeración exhaustiva de pares

Se irán descartando aquellas que no tengan importancia

Se basan en características específicas de calidad

Cobertura y precisión

Las reglas de asociación se caracterizan por *support* y *confidence*

Support: % de clientes que compraron $a_1, \dots, a_{n+m}$

- Probabilidad de que una transacción contenga $a_1, \dots, a_{n+m}$
- $P(a_1 \cup a_2 \cup \dots \cup a_{n+m})$

Confidence: % de compradores de $a_1, \dots, a_n$ que también compraron $a_{n+1}, \dots, a_{n+m}$

- $Confidence = \frac{P(a_1 \cup a_2 \cup \dots \cup a_{n+m})}{P(a_1 \cup a_2 \cup \dots \cup a_n)}$

Ejemplo

Transacción	Elementos Comprados
1	A,B,C
2	A,C
3	A,D
4	B,E,F

A, C: Cobertura 50%, Precisión 66.6%

C, A: Cobertura 50%, Precisión 100%

Un ejemplo típico de reglas de asociación es el problema de la canasta de mercado

Encontrar asociaciones entre productos

Las tiendas pueden optimizar sus distribuciones

Buscar reglas con un mínimo de confianza y precisión

Sólo encontrar asociaciones importantes

Algunos ítems aparecerán en más de una regla

Hay otros ítems que no aparecerán en ninguna

El algoritmo para buscar reglas de asociación debe ser eficiente

Algoritmo APRIORI (Agrawal et al., 1993)

- Parámetros: min support, min confidence
- Encuentra todas las asociaciones basadas en los parámetros
- Disponible en Weka

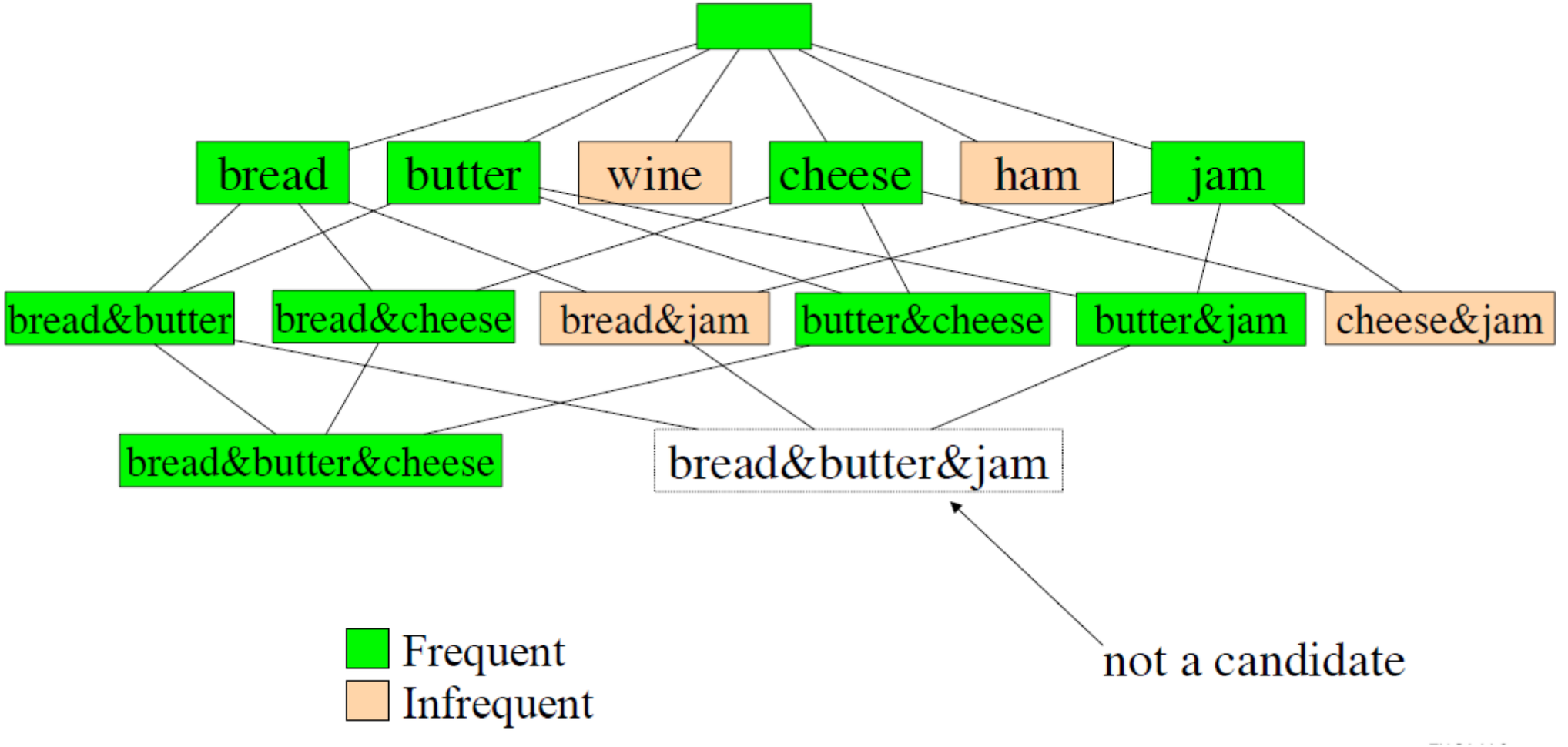
Observaciones importantes

- Sea $freq(S)$ = número de ejemplos conteniendo S
- Considerar IF $a_1 \dots a_n$ THEN $a_{n+1} \dots a_{n+m}$
- $Support = \ freq(\{a_1, \dots, a_{n+m}\})/freq(\{\})$
- $Confidence = \ freq(\{a_1, \dots, a_{n+m}\})/freq(\{a_1 \dots a_n\})$
- Las asociaciones con suficiente *confidence* y *support* pueden ser derivadas de la lista de “conjuntos frecuentes”
- S es frecuente iff $freq(S) > \min\_support * freq(\{\})$

Items frecuentes se encuentran utilizando breadth first

Paso 1: encontrar todos los conjuntos frecuentes

- Si $a_1 \dots a_j$ no es frecuente, tampoco lo es $a_1 \dots a_{j+1}$
- Encontrar todos los ítems frecuentes de cardinalidad 1
- Encontrar todos los ítems frecuentes de cardinalidad 2
- $\{a_1, a_2\}$ es frecuente solo si $\{a_1\}$ y $\{a_2\}$ lo son
- Encontrar todos los ítems frecuentes de cardinalidad 3
- …


bread
butter
wine
cheese
ham
jam
bread&butter
bread&cheese
bread&jam
butter&cheese
butter&jam
cheese&jam
bread&butter&cheese
bread&butter&jam
Frequent
Infrequent
not a candidate

Encontrando conjuntos frecuentes

```
min_freq := min_support * freq(∅);
d := 0;
Q_0 = {∅ };  /* Q_i = candidates for level i */
F := ∅ ;      /* F = frequent sets */
while Q_d ≠ ∅ do
        for all S in Q_d: find freq(S);  /* data access */
        delete those S in Q_d with freq(S)<min_freq;
        F := F ∪ Q_d;
        compute Q_{d+1};
        d := d+1
return F
```

Computación offline

compute Q_{d+1} from Q_d and F:

$Q_{d+1} := \varnothing$;

for each S in Q_d:

for each item x not in S:

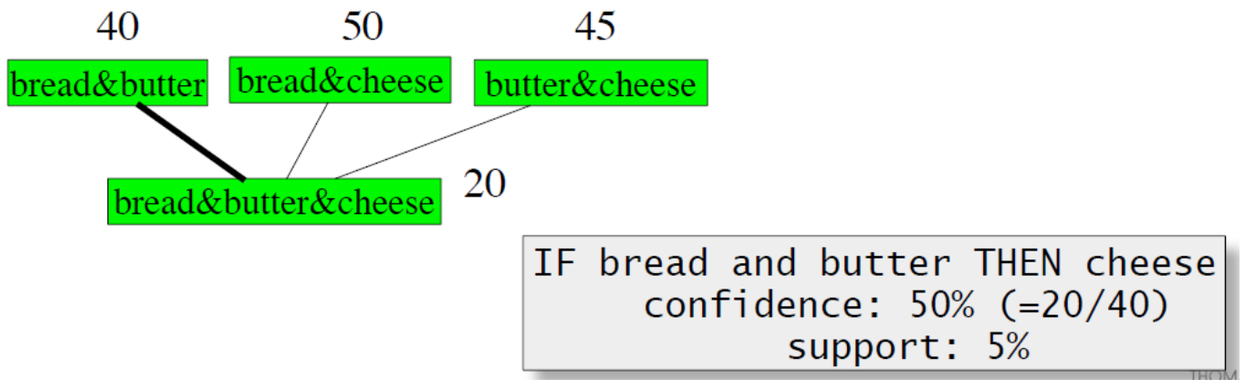
$S' := S \cup \{x\}$;

if each subset of S' obtained by removing 1 element of S' is in F

then add S' to Q_{d+1};

Paso 1: inferir reglas de asociación con conjuntos frecuentes

- Si $S \cup \{a\}$ in F and $freq(S \cup \{a\}) > \min\_confidence$
- Retornar la regla IF S THEN a



Las reglas encontradas deben ser post-procesadas para tener un orden

A veces se encuentran muchas reglas

¿Cómo procesarlas?

Ordenar bajo algún criterio

Support, confidence, significancia estadística

Otros métodos para post-procesado

Lenguaje de búsqueda

UNIVERSIDAD DE CUENCA
desde 1867

Reglas de Asociación

Andres Auquilla

2024