DMML, 22 Jan 2019

Frequent itemsets with multiple min support

item $i \longrightarrow$ mm-sup(i)

Set $\{i_1, i_2, \ldots, i_k\}$ — min support is

$$\min_{j \in \{1,2,\ldots,k\}} (\text{min-sup}(i_j))$$

Destroys a-priori property wrt smallest min-sup

$\{a, b, c\}$ → set has min-sup 0.01

0.01 0.1 0.2 → $\{b, c\}$ has min-sup 0.1

Adopt level-by-level algorithm

$C_1 \rightarrow F_1 \rightarrow C_2 \rightarrow F_2 \rightarrow \cdots \rightarrow F_{k-1} \rightarrow C_k \rightarrow F_k$

$F_1 = \{ i_j \mid sup(i_j) \geq min\text{-}sup(i_j) \}$

$\downarrow$

$C_2 \neq F_1 \times F_1 \longrightarrow \{ (i_j, i_k) \mid sup(i_j) \geq min\text{-}sup(i_j), sup(i_k) \geq min\text{-}sup(i_k) \}$

C_2 should have

(i_j, i_k), $sup(i_j) \geq min\text{-}sup(i_j)$

$sup(i_k) \geq min\text{-}sup(i_j)$

ass. $sup(i_j) \leq sup(i_k)$

Convention

Enumerate any $X \subseteq I$ in ascending order of min-sup

$$\{i_1, i_2, \ldots, i_k\}$$

$$\text{min-sup}(i_1) \leq \text{min-sup}(i_2) \leq \ldots \leq \text{min-sup}(i_k)$$

Correct C_2

$= \{(i_j, i_k) \mid \sup(i_j), \sup(i_k) \geq \text{min-sup}(i_j)\}$

↳ count & get F_2 as usual

$F_{k-1} \rightarrow C_k$

Merge $\{i_1, i_2, \ldots, i_{k-2}, i_{k-1}\}$ $\in F_{k-1}$

$\{i_1, i_2, \ldots, i_{k-2}, i'_{k-1}\}$

↓

$\{i_1, i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\} \in C_k$

⇓ Prune using a-priori

$\{i_2, \ldots, i_{k-2}, i_{k-1}, i'_{k-1}\}$ Check all k-1 subsets

has a higher min-sup $\in F_{k-1}$ [uniform case]

- not consider this k-1 subset when pruning

Generating rules

$$X = \{i_1, i_2, \ldots, i_k\} \in F_k$$

$$X \setminus \alpha \longrightarrow \alpha$$

$$\frac{\sup(X)}{\sup(X \setminus \alpha)} \geq \text{min-conf}$$

↳ What if $\alpha = \{i_j\}$?

Modify level-by-level to keep track of this

If $\{i_1, i_2, \ldots, i_k\} \in C_k$

Count occurrences of $\{i_1, i_2, \ldots, i_k\}$

and $\{i_2, \ldots, i_k\}$

need not qualify for F_{k-1},
but available for rule gen.

Other variation - Sequential rules

Buy product today
accessory later

Special Case

Last attribute is a category — topic
↑ doc

"Class Association Rules"

$$X \rightarrow \{c\}$$

General problem ↑ supervised learning

Build a model to predict a category given some attributes

Bank loans

Outcome

ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	**No**
2	young	false	false	good	**No**
3	young	true	false	good	**Yes**
4	young	true	true	fair	**Yes**
5	young	false	false	fair	**No**
6	middle	false	false	fair	**No**
7	middle	false	false	good	**No**
8	middle	true	true	good	**Yes**
9	middle	false	true	excellent	**Yes**
10	middle	false	true	excellent	**Yes**
11	old	false	true	excellent	**Yes**
12	old	false	true	good	**Yes**
13	old	true	false	good	**Yes**
14	old	true	false	excellent	**Yes**
15	old	false	false	fair	**No**

Training data — historical data with class labels

Model

Attributes of unknown item → Predict class

"labelled data"

Supervised learning

Implicit assumption: Training data is statistically representative of future data

How to evaluate the model?

By defn, we don't know correct answers for unseen data

What does it mean "to predict"?

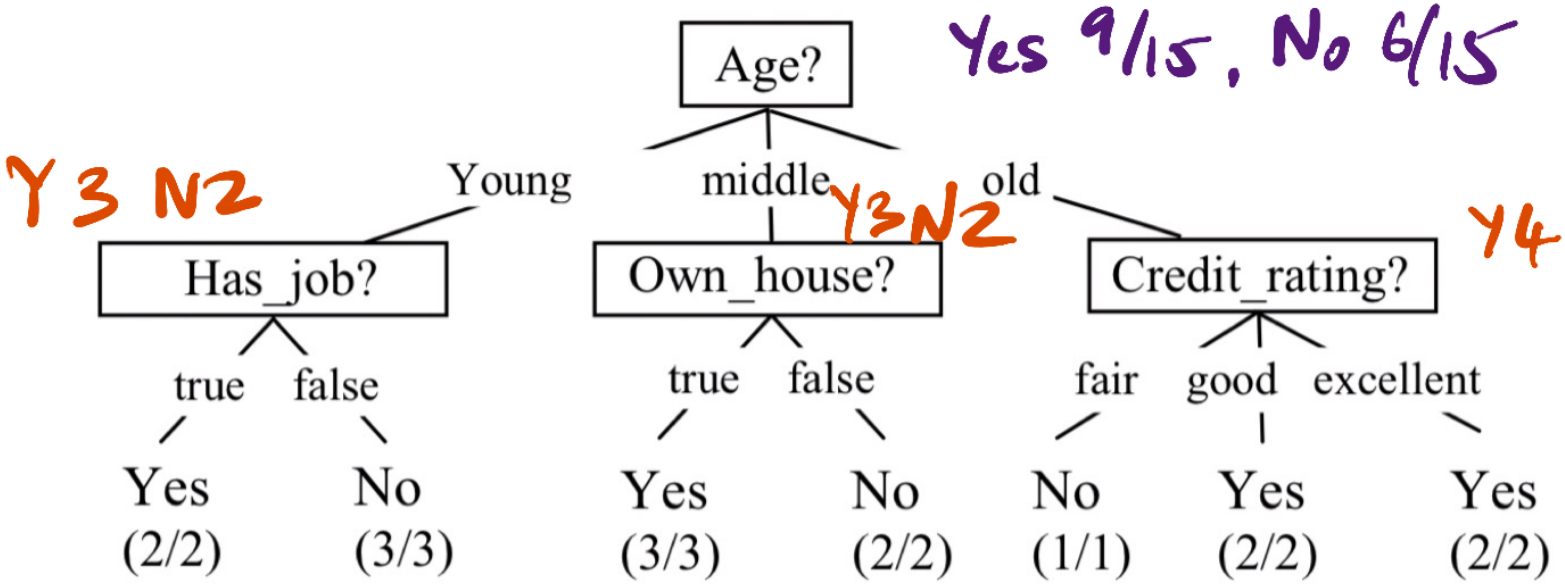
Data is not always "complete"

Attributes not captured

Decision Tree

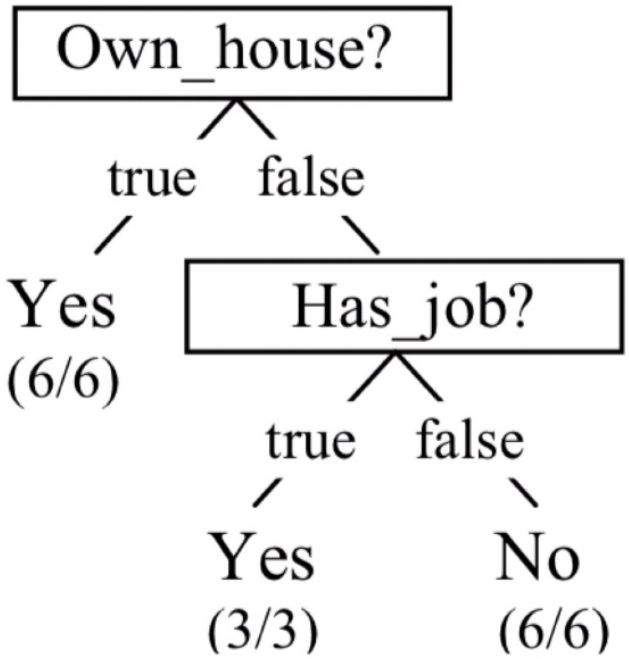
ID	Age	Has_job	Own_house	Credit_rating	Class
1	young	false	false	fair	No
2	young	false	false	good	No
3	young	true	false	good	Yes
4	young	true	true	fair	Yes
5	young	false	false	fair	No
6	middle	false	false	fair	No
7	middle	false	false	good	No
8	middle	true	true	good	Yes
9	middle	false	true	excellent	Yes
10	middle	false	true	excellent	Yes
11	old	false	true	excellent	Yes
12	old	false	true	good	Yes
13	old	true	false	good	Yes
14	old	true	false	excellent	Yes
15	old	false	false	fair	No

Yes 9/15, No 6/15



1. How to decide what question to ask next?

Answer is not unique!

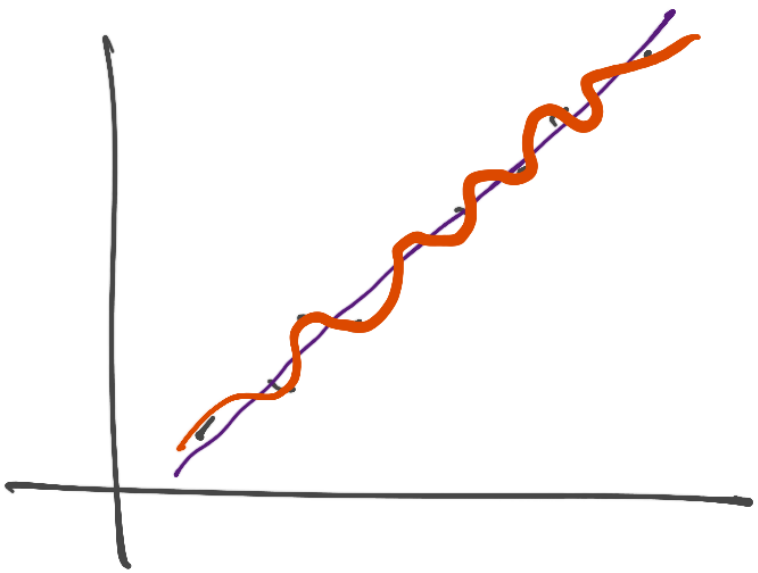


"Smaller" tree

2. Which tree are we looking for?

Small trees are better

- Simpler explanation
- Shorter trees generalize the training data better



By any defn of "smallest", finding the smallest tree is computationally hard

NP-complete

"Greedy" strategy - heuristic to get as good a tree as we can

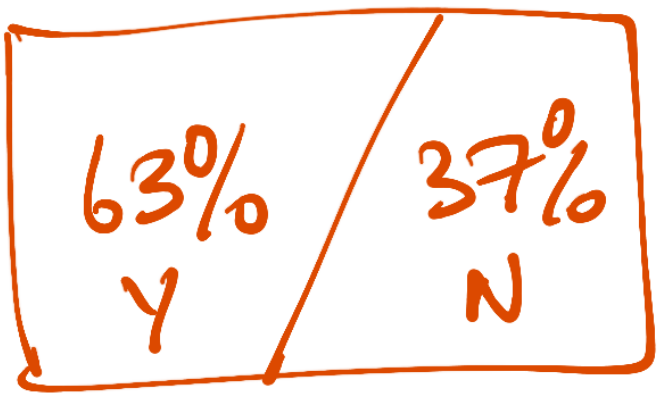
Goal → Narrow down candidates to all Yes or No - Assume binary classification

Go from "impure" mix to "pure" set

Purity as a measure to make a locally optimal choice — which question to ask

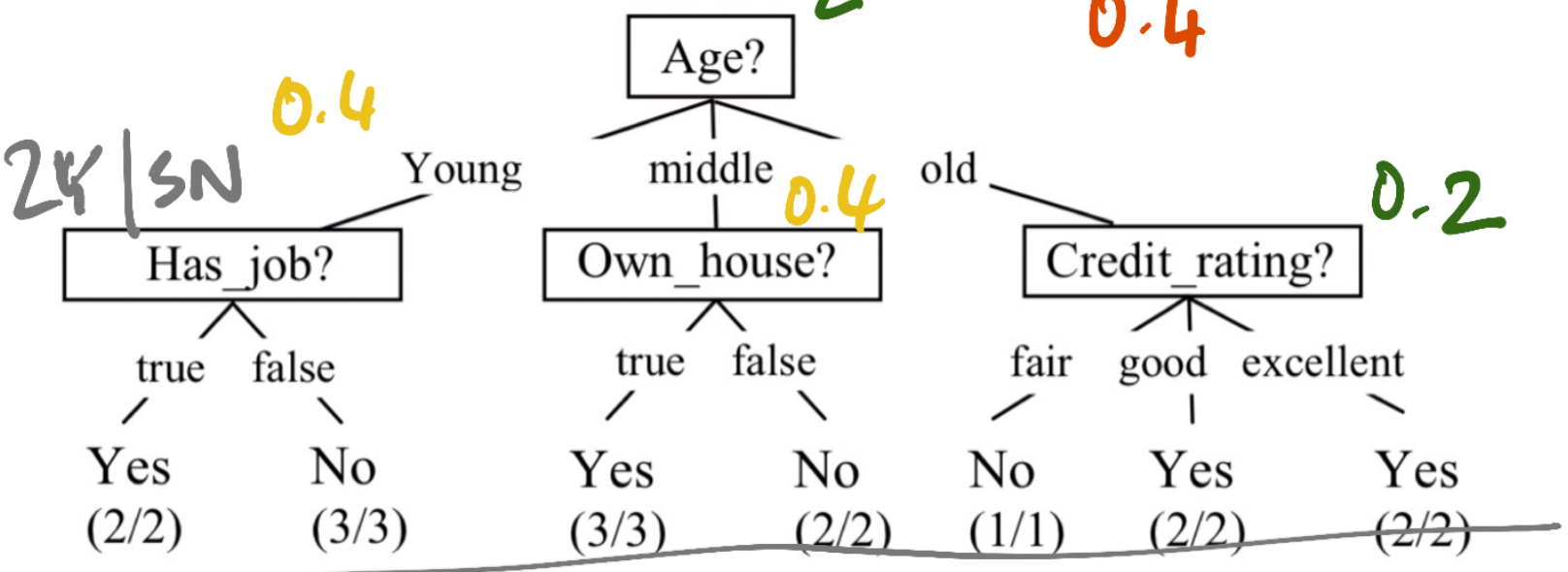
Suppose we must ask a question

End up with



→ Predict majority value = Y

→ Error rate is 37%

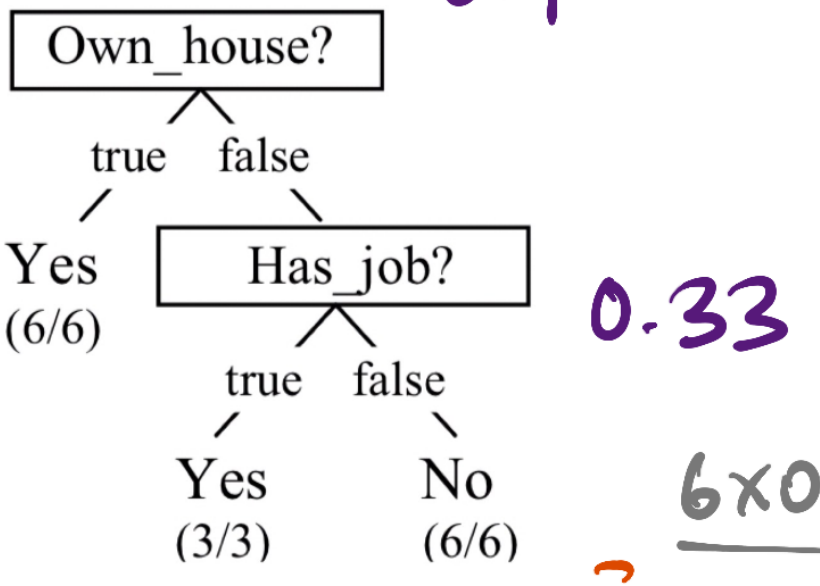


Weighted avg

$= 5 \cdot (0.4) + 5(0.4) + 5(0.2) = 0.33$

$\frac{}{15}$

0



$= \frac{6 \times 0 + 9 \times 0.33}{15}$

Algorithm

Among the attributes still to be explored, choose the one that improves purity by largest amount

Stopping criterion

- Impurity = 0
- Run out of questions — use majority value as prediction

A "better" notion of impurity

Based on empirical observations