Grado en ingeniería informática
Artificial Intelligence 2020/2021

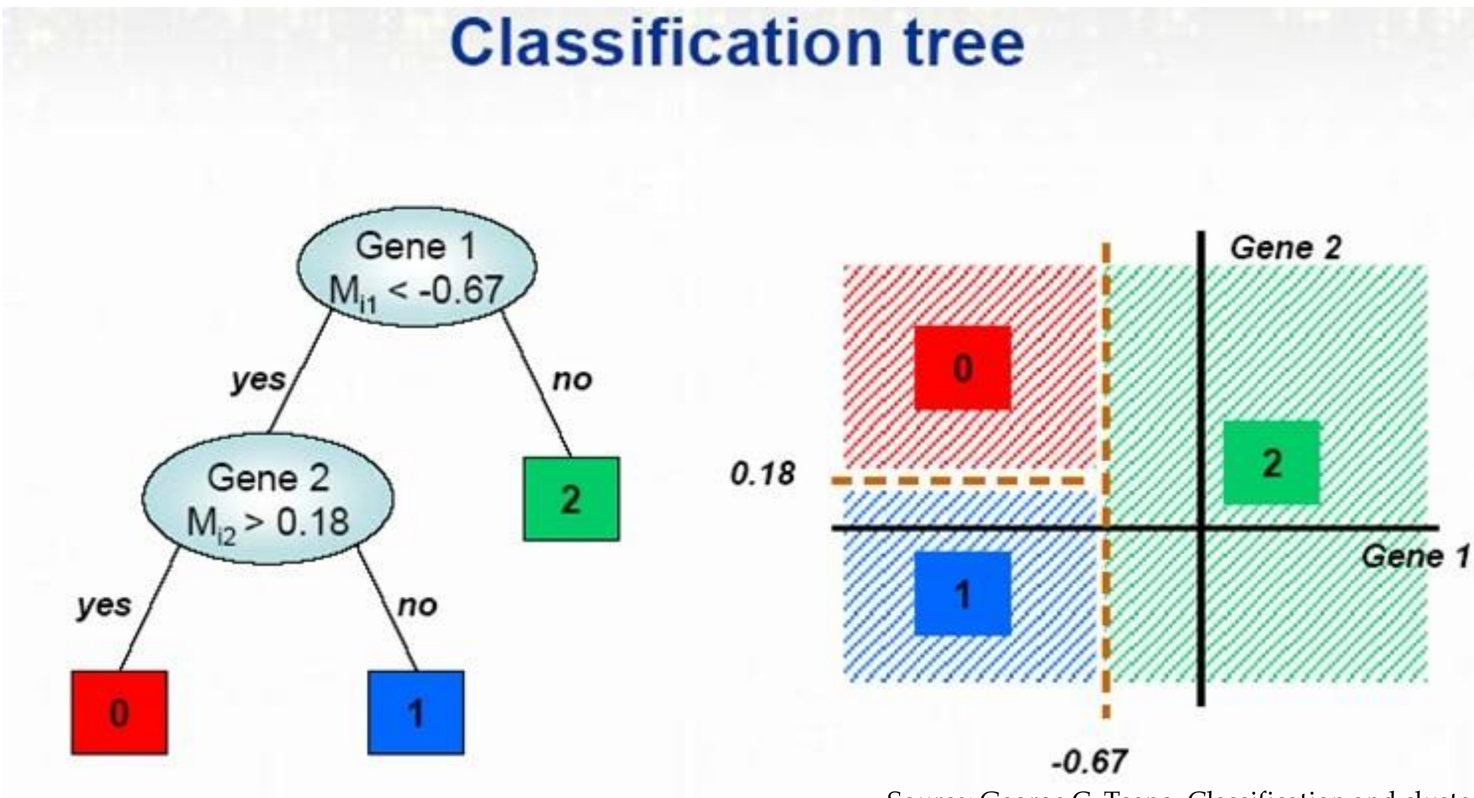
Decision Trees

Lara Quijano Sánchez

UAM
Universidad Autónoma
de Madrid

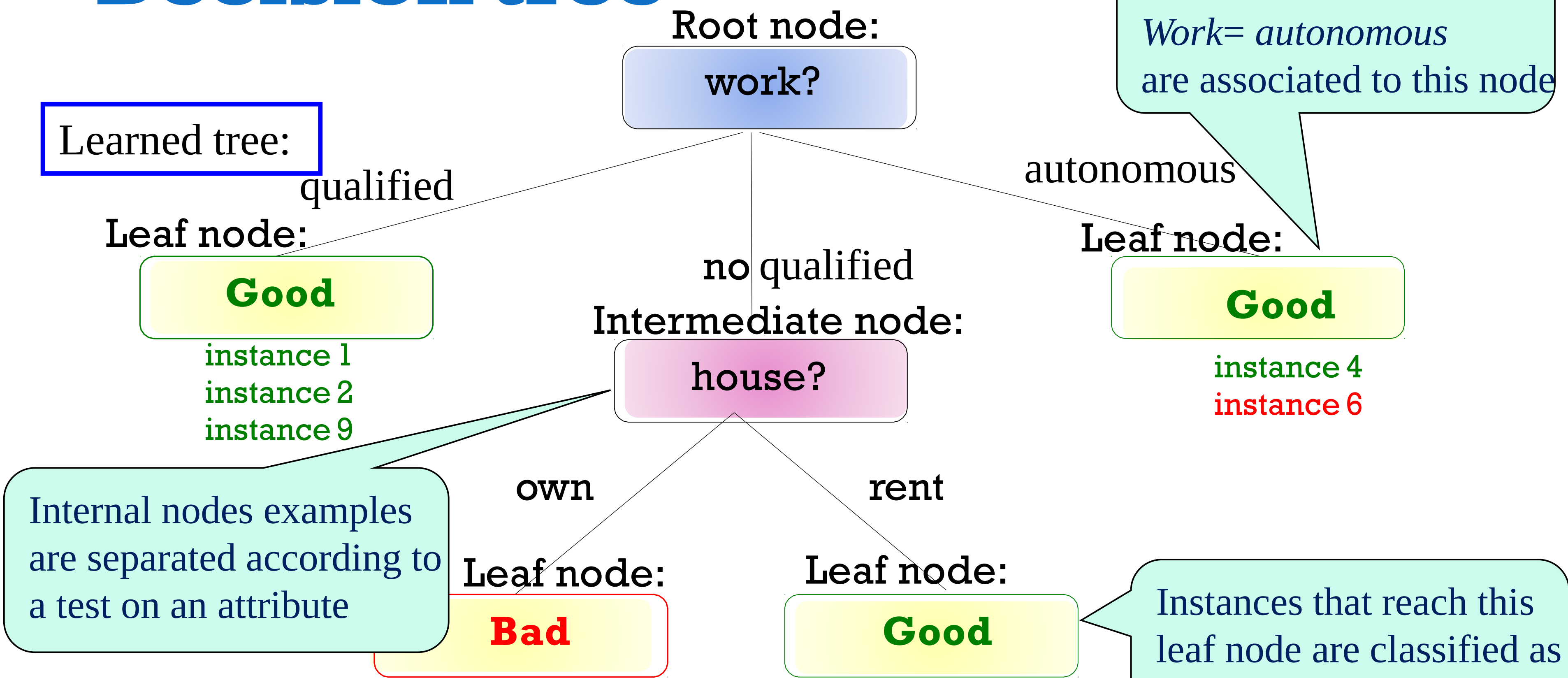
Classification and Regression Tree (CART)

- Partition the sample space into rectangles and then predict a simple model in each of them
- binary trees discriminate the space in two subsamples (nodes) from a previous one



Source: George C. Tseng. Classification and clustering problems in microarray analysis and some recent advances. 2004

Decision tree



Training data

id	age	marital status	savings	Education level	Work	house	amount	class
1	35	single	7,000	highschool	qualified	own	50K	good
2	23	married	2,000	vocational training	qualified	rent	70K	good
3	30	married	1,000	highschool	No-qualified	own	60K	bad
4	26	single	15,000	Bachelor's	autonom.	own	120K	good
5	50	divorced	3,500	Bachelor's	No-qualified	rent	40K	good
6	43	single	NA	highschool	autonom.	NA	30K	bad
7	31	divorced	28,000	Master	No-qualified	own	90K	bad
8	33	married	NA	Bachelor's	No-qualified	rent	30K	good
9	40	single	11,000	Master	qualified	own	100K	good

Interpretability: Decision Rules

Example
IF house= `own` **AND** age<**=** `28`
THEN class = **good**

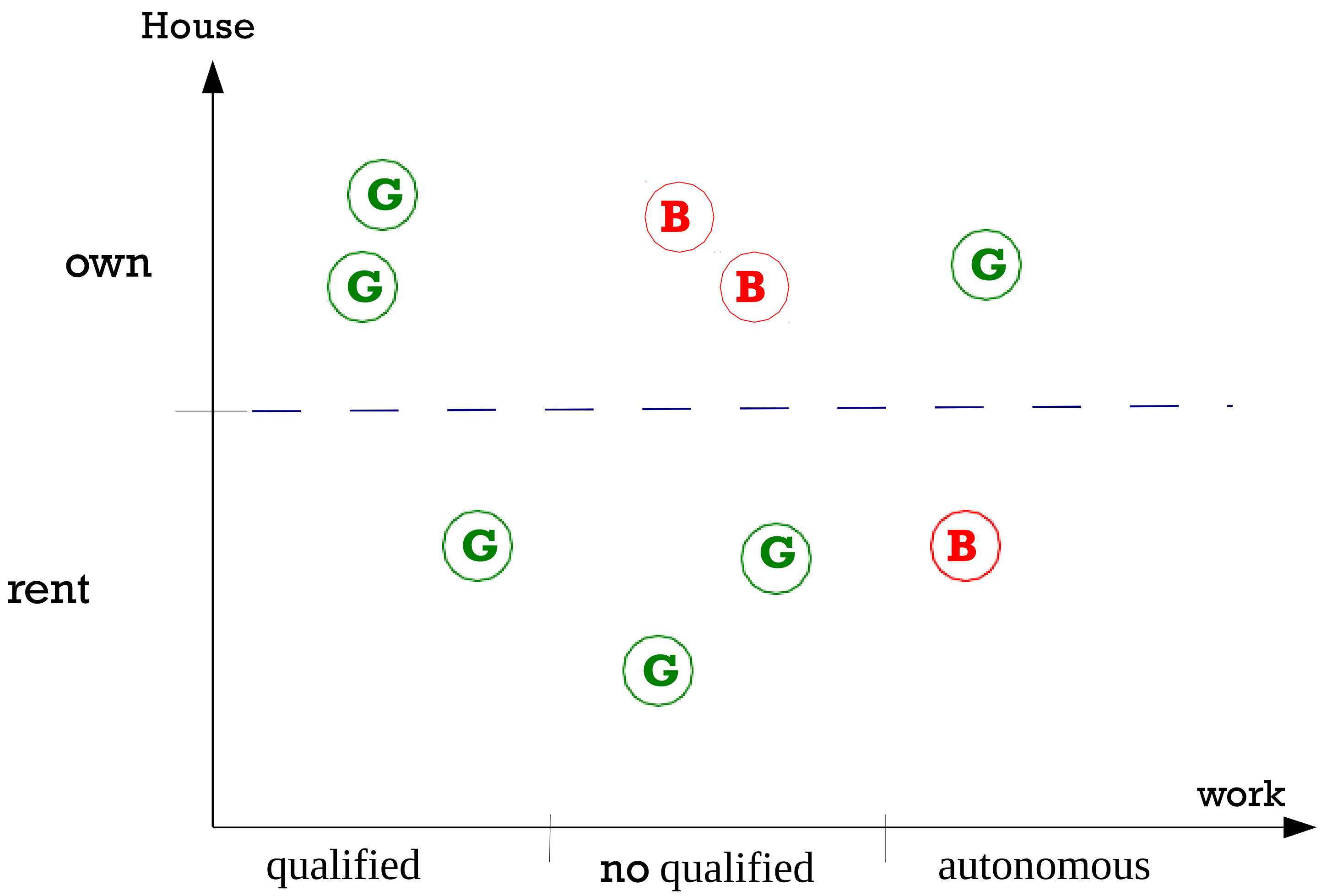
IF work= `no qualified` **AND** amount **=>** `50K`
THEN class = **bad**

- Inductive construction based on a separation or coverage criterion
- Complete separation of instances is not required (overlapping rules)
- All decision trees can be translated into decision rules

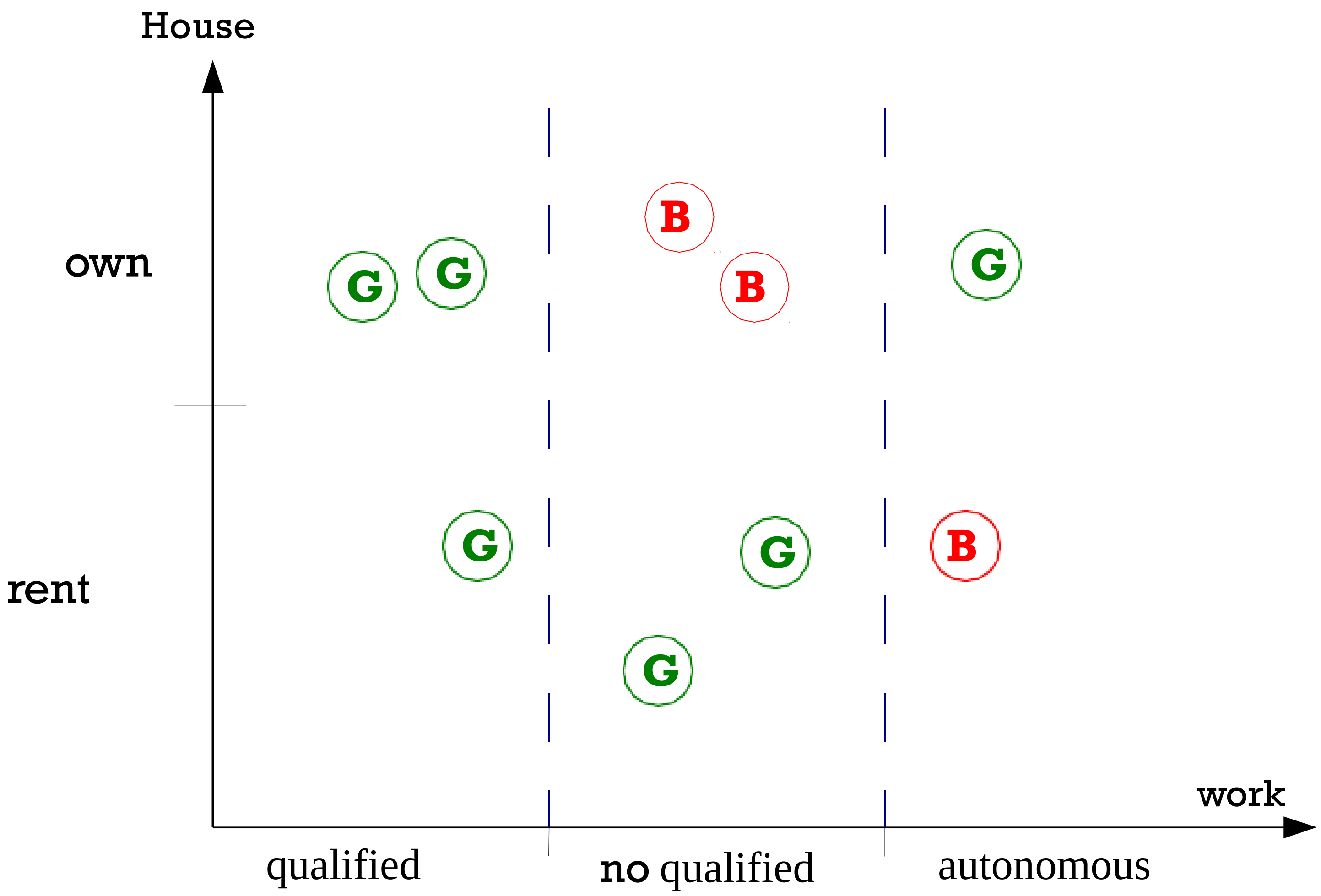
Concepts

- Diagram representing successive conditions on attributes to classify an instance
- Node types
  - Internal nodes:
    - Conditional questions about attributes
    - Each answer follows and arrow
    - Complete separation of the examples between the possible answers
  - Leaf nodes
    - Class → prediction
    - Prediction confidence
    - Training examples that fulfilled the conditions up to the leaf node
- Modelling objectives
  - Build the simplest tree that best separates the instances by class
  - The final model must generalize to classify future instances well

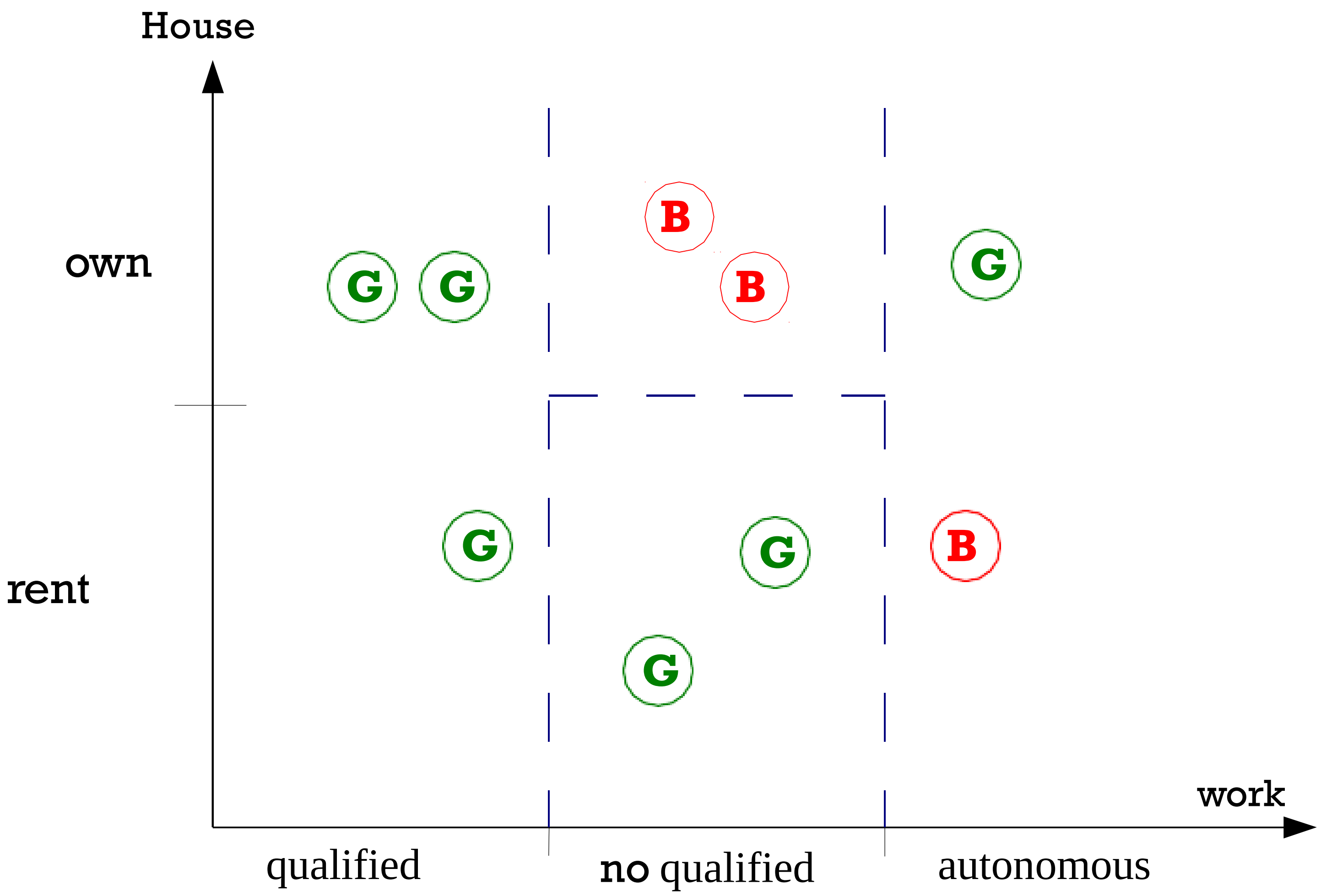
Decision tree strategy (inefficient)



Decision tree strategy



Decision tree strategy



Decision tree generation

- Inductive construction: An attribute with all its possible answers is added in each step
- The attribute that **best separates** (orders) the examples according to the classes is selected
- Separation criteria
  - Entropy (ID3)
  - Gini impurity (CART)
  - Information Gain (ID3, C4.5)
  - Information Gain ratio
  - Precision
- The process stops when adding a new attribute does not improve the separation criteria

Attribute relevance

- Decision tree construction performs implicit attribute selection

CART separation criteria: Gini impurity

- It reaches its minimum (zero) when all cases in the node fall into a single target category
- Given *N* observations/rows
- Given feature A that can take *M* different values = A_1, A_2,…A_M
- Used in binary (trees with 2 branches per node) trees = CART
  - Simplification of Gini = 2*probability(class1) *probability(class2)
- **Gini Impurity** of A

$$Gini(A) = 1 - \sum_{i=1}^{M} P(A_i)^2 = 1 - \sum_{i=1}^{M} \left(\frac{Freq(A_i)}{N}\right)^2$$

Other (hiper)parameters

- Minimum IG
- Maximum tree depth
- Minimum examples per leaf
- Pruning tree

That is

- A decision tree is a hierarchical questionnaire that splits the data according to a sequence of tests on their attributes.
- Each instance/row, when processed by the tree, follows a unique path from the root node to the corresponding leaf according to the results of the tests on the attributes performed at each of the intermediate internal nodes.
- The class associated to a leaf node corresponds to the majority label of the training instances assigned to that node.
- Order/choice of internal nodes are determined by maximizing a quantity (e.g. the information IG) that favours a clearer separation of the classes in the children of such nodes.

ID3: Learning algorithm

function DECISION-TREE-LEARNING(*examples, attributes, default*) **returns** a decision tree
  inputs: *examples*, set of examples
    attributes, set of attributes
    default, default value for the goal predicate

  if *examples* is empty **then return** *default*
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** MAJORITY-VALUE(*examples*)
  else
    best ← CHOOSE-ATTRIBUTE(*attributes, examples*)
    tree ← a new decision tree with root test *best*
    for each value v_i of *best* **do**
      $examples_i$ ← {elements of *examples* with $best = v_i$}
      subtree ← DECISION-TREE-LEARNING($examples_i$, *attributes* − *best*,
        MAJORITY-VALUE(*examples*))
      add a branch to *tree* with label v_i and subtree *subtree*
    end
    return *tree*

Simplified version of Quinlan's ID3 (1986)

The **best attribute** is the one that provides the largest amount of **information** on the class label.

Recursion

ID3

- The tree is built from top to bottom, working in levels
- In each iteration of the algorithm it is intended:
  - Obtain the attribute based on which to branch the problem node
  - Select the one that best discriminates between the set of examples
    - Heuristic to get small trees (in depth)
    - The most discriminating attribute will be the one that leads to a state with less entropy or less disorder (more information)
- Entropy (Shannon, 1948) measures the lack of homogeneity of a set of examples with respect to their class
  - It is a standard measure of disorder (0 is total homogeneity)
- Information IG is the difference between
  - the entropy of the original set and that of the subsets obtained

ID3

- For each attribute the decrease in entropy (information IG) caused by its use is calculated
  - Information IG = Entropy $\text{decrease}_A(X) = E(X) - E_A(X)$
  - Information IG = Entropy $\text{decrease}_B(X) = E(X) - E_B(X)$
  - Information IG = Entropy $\text{decrease}_C(X) = E(X) - E_C(X)$ ...

- In each node, the attribute that causes the greatest decrease in entropy is selected

- This measure tends to favour the choice of:
  - attributes with many possible values
  - which results in a worse generalization

Computing entropy on a feature

- Given N observations/rows
- Given feature A that can take M different values = $A_1, A_2, \ldots A_M$
- **Entropy of A** =

$$H(A) = -\sum_{i=1}^{M} P(A_i) \log_2 P(A_i) = -\sum_{i=1}^{M} \frac{Freq(Ai)}{N} \log_2 \frac{Freq(A_i)}{N}$$

Example: Computing entropy on a feature

- Dataset: play tennis outside for previous 14 days

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	85	85	Weak	No
2	Sunny	80	90	Strong	No
3	Overcast	83	78	Weak	Yes
4	Rain	70	96	Weak	Yes
5	Rain	68	80	Weak	Yes
6	Rain	65	70	Strong	No
7	Overcast	64	65	Strong	Yes
8	Sunny	72	95	Weak	No
9	Sunny	69	70	Weak	Yes
10	Rain	75	80	Weak	Yes
11	Sunny	75	70	Strong	Yes
12	Overcast	72	90	Strong	Yes
13	Overcast	81	75	Weak	Yes
14	Rain	71	80	Strong	No

- Global entropy:
  - Prediction class= decision.
  - There are 14 examples; 9 instances refer to yes decision, and 5 instances refer to no decision.
- Entropy(Decision) =

$-$ p(Yes) * $\log_2$p(Yes) $-$ p(No) * $\log_2$p(No) =

$-$ (9/14) *$\log_2$(9/14) $-$ (5/14) *$\log_2$(5/14) = 0.940

Computing conditional entropy on a feature

- Given N observations/rows
- Given the class to predict C that can take L different values = C_1, C_2,…C_L
- Given feature A that can take M different values = A_1, A_2,…A_M
- **Conditional Entropy of C|A** ->

$$H(C|A) = \sum_{i=1}^{M} P(A_i) \times H(C|A_i) = \sum_{i=1}^{M} \frac{Freq(A_i)}{N} \times H(C|A_i).$$

- **Entropy of C conditioned on A = A_i** ->
  - Given that A_i appears in the dataset in K observations/rows

$$H(C|A_i) = -\sum_{j=1}^{L} P(C_j|A_i) \log_2 P(C_j|A_i) = -\sum_{j=1}^{L} \frac{Freq(C_j)}{K} \log_2 \frac{Freq(C_j)}{K}$$

Example: Computing conditional entropy on feature

- Wind **Categorical Attribute**. Two possible values: weak and strong
  - P(weak) = 8/14 =0.571
  - P(strong) = 6/14 =0.428
- There are 8 weak wind instances. 2 of them are concluded as no, 6 of them are concluded as yes.
  - Entropy(Decision|Wind=Weak) = – p(No) $\log_2$p(No) – p(Yes) $\log_2$p(Yes) = – (2/8)$\log_2$(2/8) – (6/8)$\log_2$(6/8) = 0.811
- There are 6 strong wind instances. 3 of them are concluded as no, 3 of them are concluded as yes
  - Entropy(Decision|Wind=Strong) = – (3/6)$\log_2$(3/6) – (3/6)$\log_2$(3/6) = 1
- Entropy(Decision|Wind) =

0.571*0.811 + 0.428*1= 0.891

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	85	85	Weak	No
2	Sunny	80	90	Strong	No
3	Overcast	83	78	Weak	Yes
4	Rain	70	96	Weak	Yes
5	Rain	68	80	Weak	Yes
6	Rain	65	70	Strong	No
7	Overcast	64	65	Strong	Yes
8	Sunny	72	95	Weak	No
9	Sunny	69	70	Weak	Yes
10	Rain	75	80	Weak	Yes
11	Sunny	75	70	Strong	Yes
12	Overcast	72	90	Strong	Yes
13	Overcast	81	75	Weak	Yes
14	Rain	71	80	Strong	No

Example: Computing conditional entropy on feature

- Outlook **Categorical Attribute.** Three possible values: weak and strong
  - P(sunny) = 5/14 =0.3571 P(rain) = 5/14 =0.3571
  - P(overcast) = 4/14 =0.2857
- There are 5 sunny instances. 3 of them are concluded as no, 2 of them are concluded as yes.
  - Entropy(Decision|Outlook=sunny) = – p(No)$\log_2$p(No) – p(Yes)$\log_2$p(Yes) = – (3/5)$\log_2$(3/5) – (2/5)$\log_2$(2/5) = 0.441 + 0.528 = 0.970
- There are 4 overcast instances all of them are concluded as yes
  - Entropy(Decision|Outlook=overcast)= – (0/4)$\log_2$(0/4) – (4/4)$\log_2$(4/4) = 0
- There are 5 rain instances. 2 of them are concluded as no, 3 of them are concluded as yes
  - Entropy(Decision|Outlook=rain)= – (2/5)$\log_2$(2/5) – (3/5)$\log_2$(3/5) = 0.970
- Entropy(Decision|Outlook) = 0.357*0.97 + 0.285*0+ 0.357*0.97= 0.692

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	85	85	Weak	No
2	Sunny	80	90	Strong	No
3	Overcast	83	78	Weak	Yes
4	Rain	70	96	Weak	Yes
5	Rain	68	80	Weak	Yes
6	Rain	65	70	Strong	No
7	Overcast	64	65	Strong	Yes
8	Sunny	72	95	Weak	No
9	Sunny	69	70	Weak	Yes
10	Rain	75	80	Weak	Yes
11	Sunny	75	70	Strong	Yes
12	Overcast	72	90	Strong	Yes
13	Overcast	81	75	Weak	Yes
14	Rain	71	80	Strong	No

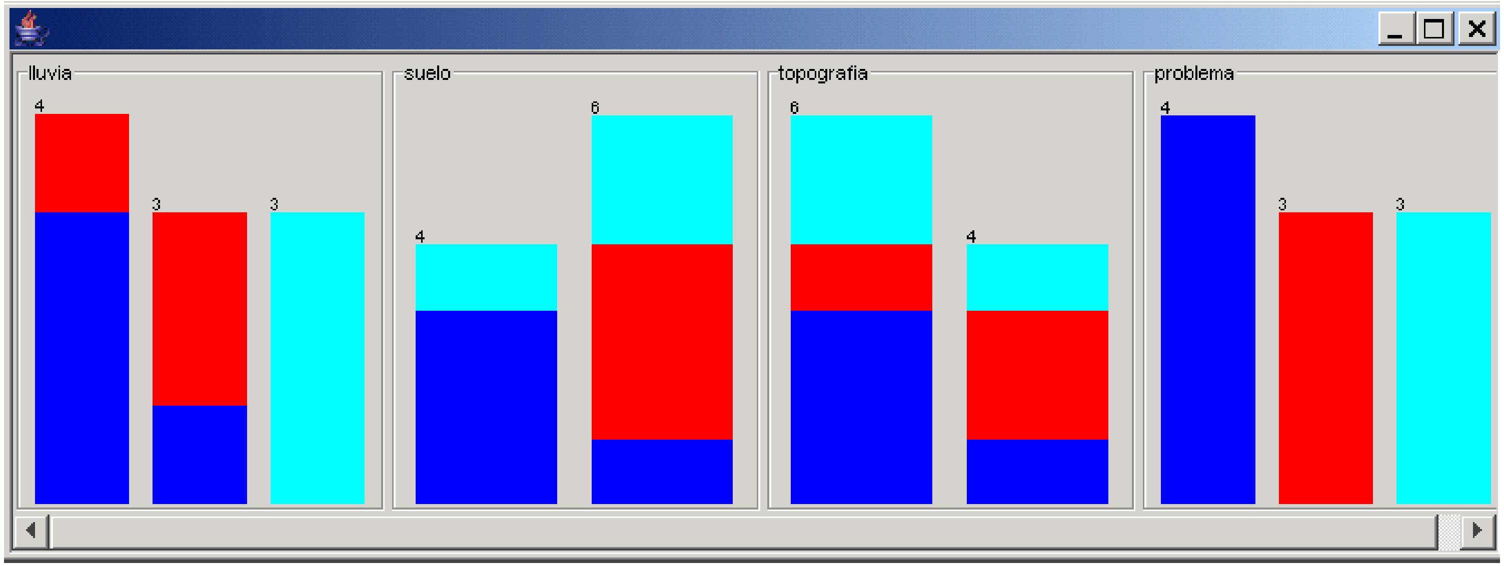
Information Gain (IG)

- ID3 Selects as best attribute the one that maximizes the IG of the class given by that attribute.
- $IG(C \mid A) = H(C) - H(C|A)$
- In the example:
  - IG(Decision|Wind) = H(Decision) – H(Decision|Wind) = 0.940 – 0.891 = 0.049
  - IG(Decision|Outlook) = H(Decision) – H(Decision|Outlook) = 0.940 – 0.692 = 0.246

Steps in ID3 algorithm

- Calculate the Information IG of each feature.
- Considering that all rows don't belong to the same class, split the dataset S into subsets using the feature for which the Information IG is maximum.
- Make a decision tree node using the feature with the maximum Information IG.
- If all rows belong to the same class, make the current node as a leaf node with the class as its label.
- Repeat for the remaining features until we run out of all features, or the decision tree has all leaf nodes.

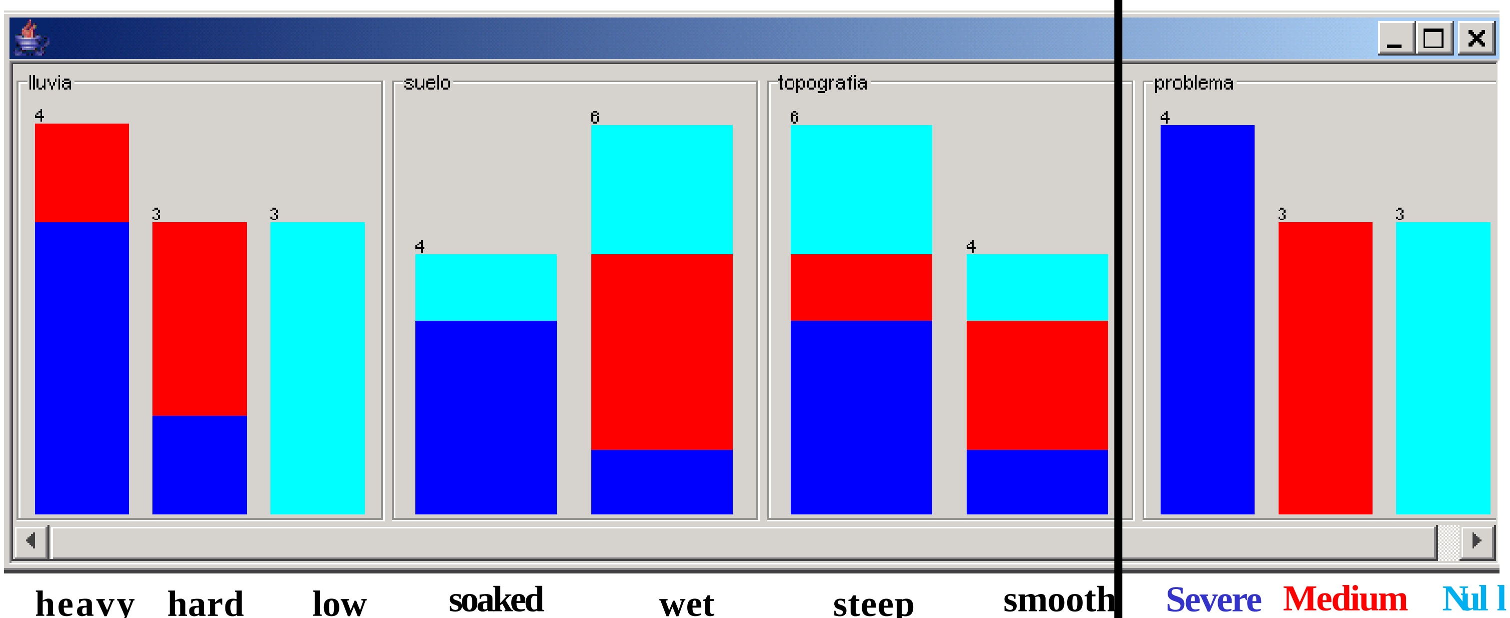
ID3: example



heavy hard low soaked wet steep smooth Severe Medium Nul l

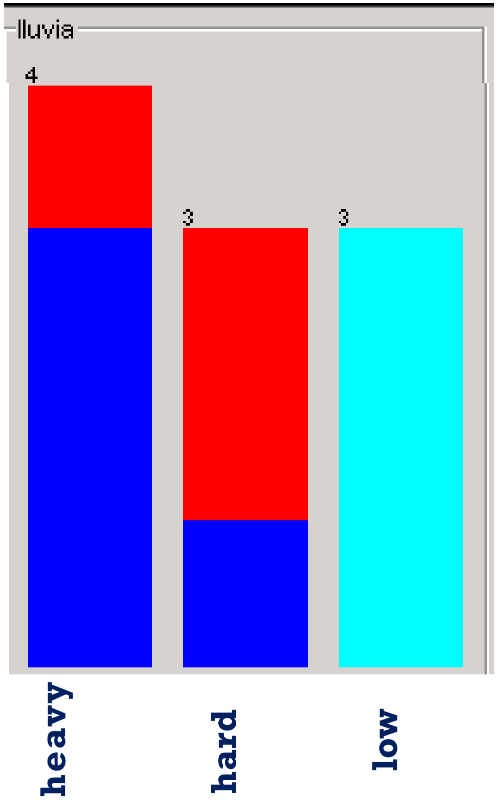
ID3 example: Problem definition

- Initial entropy at the root of the tree: (of the global problem)
  - P(severe) = 0,4 P(medium) = 0,3 P(null) = 0,3
  - $H(root) = -0{,}4 \log_2 0{,}4 - 0{,}3 \log_2 0{,}3 - 0{,}3 \log_2 0{,}3 = 1{,}571$



ID3 example: Entropy in rain

- Final entropy classifying according to rain(A):
  - A_1: heavy rain, $P(A_1)= 4/10$
  - A_2: hard rain, $P(A_2)= 3/10$
  - A_3: low rain, $P(A_3)= 3/10$



- $H(A_1)= -0{,}75 \log_2 0{,}75 - 0{,}25 \log_2 0{,}25 = 0{,}811$
- $H(A_2)= -1/3 \log_2 1/3 - 2/3 \log_2 2/3 = 0{,}918$

$H(A_3)= 0$

entropy

$H(C/A)= 0{,}4*0{,}811 + 0{,}3*0{,}918= 0{,}6$

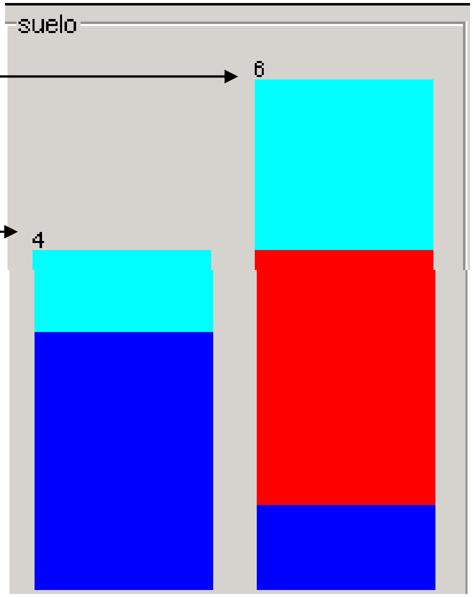
"heavy" probability

$IG(A)=$ Entropy decrease$_A$(root) $= 1{,}571 - 0{,}60 = 0{,}971$

ID3 example: Entropy in soil

- Final entropy classifying according to (soil):

  $H(C/B) = 0{,}4 * 0{,}811 + 0{,}6 * 1{,}459 = 1{,}20$



$-0{,}75 \log_2 0{,}75 - 0{,}25 \log_2 0{,}25$

$-1/6 \log_2 1/6 - 3/6 \log_2 3/6 - 2/6 \log_2 2/6$

IG(B)= Entropy decrease$_B$(root) = 1,571 – 1,20 = 0,371

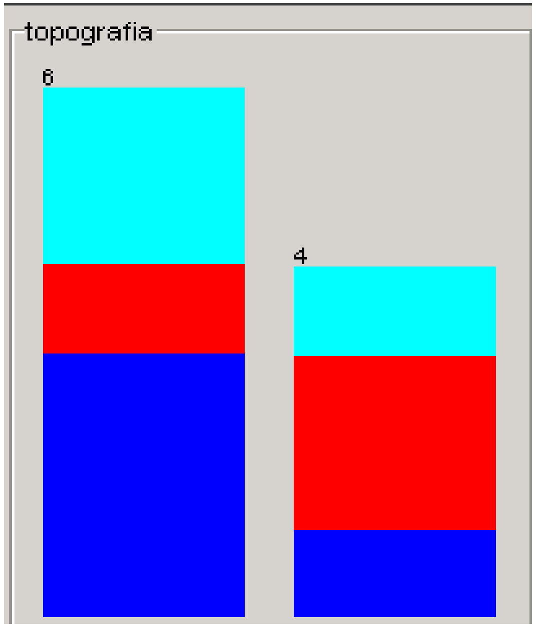
ID3 example: Entropy in topography

- Final entropy classifying according to topography(C):

  - $-1/4 \log_2 1/4 - 2/4 \log_2 2/4 - 1/4 \log_2 1/4$

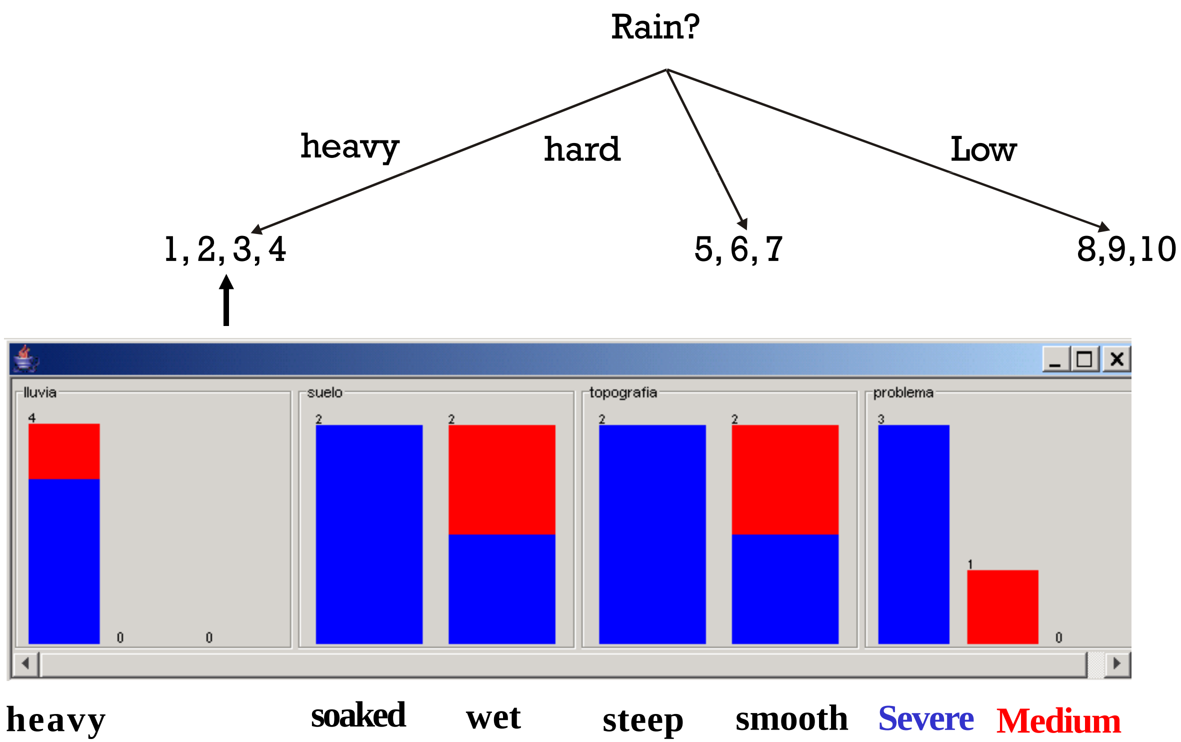
- H(C/C) 0,6*1,459 + 0,4*1,50= 1,475

  $-1/6 \log_2 1/6 - 3/6 \log_2 3/6 - 2/6 \log_2 2/6$



- IG(C)= Entropy decrease$_C$(root) = 1,571 – 1,475 = 0,096
- The greatest decrease in entropy is achieved with attribute A and therefore this is the one selected for the first level of the tree

ID3 example: root



- In the next iteration, the algorithm is applied on each of the three new nodes, considering in each one the subset of examples obtained and having eliminated the rain attribute from the set of attributes
- Problem attributes with more possible values tend to always give more information gain and be picked first
  - Causing a bias for this type of variables

When does one stop splitting a node?

- The training examples assigned to that node belong to the same class.
  - The leaf node assigns that class label
- Node has no examples associated to it.
  - The leaf node assigns the default class label
- No more attributes left for splitting the data.
  - The leaf node assigns the majority class label in that node
- Prepruning (limit the tree size to avoid overfitting)
  - The number of training examples associated to the node is below a threshold.
  - The Information Gain is below a threshold. (Eg. Threshold= I_g of a random split)

  [The leaf node assigns the majority class label in that node]

Pruning to avoid overfitting in Decision Trees

- Bias towards smaller (less complex) trees.
  - Prepruning
  - Postpruning: Grow tree to a large size and then prune subtrees that do not provide significant IGs in predictive accuracy.
    - Consider an internal node.
      - Replace it by a leaf node denoting the most frequent class label
    - If turning that node into a leaf does not lead to a significant decrease in the predictive accuracy of the pruned decision tree, then eliminate the subtree which has that node as its root.
    - For this process, accuracy can be estimated on a separate validation set (reduced error pruning), or by Cross Validation (e.g. as in CART)
    - Continue pruning until significant deterioration of accuracy

Postpruning is generally preferred. This is a common strategy in machine learning: consider first a potentially complex model and then penalize complexity

Limitations

- How do we compute the entropy for continuous variables?
  - Eg Temp, Humidity….

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	85	85	Weak	No
2	Sunny	80	90	Strong	No
3	Overcast	83	78	Weak	Yes
4	Rain	70	96	Weak	Yes
5	Rain	68	80	Weak	Yes
6	Rain	65	70	Strong	No
7	Overcast	64	65	Strong	Yes
8	Sunny	72	95	Weak	No
9	Sunny	69	70	Weak	Yes
10	Rain	75	80	Weak	Yes
11	Sunny	75	70	Strong	Yes
12	Overcast	72	90	Strong	Yes
13	Overcast	81	75	Weak	Yes
14	Rain	71	80	Strong	No

Improvement: C4.5 Decision tree

- Evolution of ID3 by Quinlan (1992)
- Improvements
  - Handling both continuous and discrete attributes
    - Convert continuous values to discrete ones
      - Perform binary split based on a threshold value
      - Threshold is a value of the attribute which offers maximum IG for that attribute
  - Handling training data with missing attribute values
    - Missing attribute values are simply not used in IG and entropy calculations
  - Handling attributes with differing costs
  - Pruning trees after creation
    - Goes back through the tree once it's been created and attempts to remove branches that do not help by replacing them with leaf nodes
  - Normalization of Information Gain for multivalued attributes
    - To avoid a bias in favour of features with a lot of different values C4.5 uses information gain ratio instead of information gain

Normalized Information Gain: Information Gain ratio

- Information Gain Ratio: $\mathrm{IGR}(C|A) = \frac{IG(C \mid A)}{IV(A)}$
  - Information gain: $IG(C \mid A) = H(C) - H(C|A)$
  - Intrinsic value:

$$IV(A) = H(A)$$
$$= -\sum_{i=1}^{M} P(A_i) \log_2 P(A_i) = -\sum_{i=1}^{M} \frac{Freq(A_i)}{N} \log_2 \frac{Freq(A_i)}{N}$$

Example: Information Gain Ratio

- In the example:
  - IG(Decision|Wind) = H(Decision) – H(Decision|Wind) = 0.940 – 0.891 = 0.049
  - IG(Decision|Outlook) = H(Decision) – H(Decision|Outlook) = 0.940 – 0.692 = 0.246
- There are 8 decisions for weak wind, and 6 decisions for strong wind
  - IV(wind) =-(8/14)$\log_2$(8/14) – (6/14)$\log_2$(6/14) = 0.461 + 0.524 = 0.985
- There are 5 instances for sunny, 4 instances for overcast and 5 instances for rain
  - IV(outlook) = -(5/14)$\log_2$(5/14) -(4/14)$\log_2$(4/14) -(5/14)$\log_2$(5/14) = 1.577
- Information IG ration:
  - IGR(Decision|Wind)= IG(Decision|Wind)/ IV(wind) = 0.049 / 0.985 = 0.049
  - IGR(Decision|Outlook)= IG(Decision|Outlook)/ IV(outlook) = 0.246/1.577 = 0.155

Day	Outlook	Temp.	Humidity	Wind	Decision
1	Sunny	85	85	Weak	No
2	Sunny	80	90	Strong	No
3	Overcast	83	78	Weak	Yes
4	Rain	70	96	Weak	Yes
5	Rain	68	80	Weak	Yes
6	Rain	65	70	Strong	No
7	Overcast	64	65	Strong	Yes
8	Sunny	72	95	Weak	No
9	Sunny	69	70	Weak	Yes
10	Rain	75	80	Weak	Yes
11	Sunny	75	70	Strong	Yes
12	Overcast	72	90	Strong	Yes
13	Overcast	81	75	Weak	Yes
14	Rain	71	80	Strong	No

Example: Computing conditional entropy on continuous feature

- Humidity **Continuous Attribute**
  - Convert it to nominal by choosing a value that binarizes the series. That value/threshold is the one that achieves maximum IG
  - Lets sort humidity values – to +
  - Iterate on all humidity values and separate the dataset into two parts:
    - instances less than or equal to current value
    - and instances greater than the current value
  - Calculate the information IG and IG ratio for every step
  - The value that maximizes the IG ratio will be the threshold

Day	**Humidity**	**Decision**
7	65	Yes
6	70	No
9	70	Yes
11	70	Yes
13	75	Yes
3	78	Yes
5	80	Yes
10	80	Yes
14	80	No
1	85	No
2	90	No
12	90	Yes
8	95	No
4	96	Yes

Example: Computing conditional entropy on continuous feature

- Check 65 as a threshold for humidity
  - H(Decision|Humidity<=65) = – p(No)$\log_2$p(No) – p(Yes)$\log_2$p(Yes) = -(0/1).$\log_2$(0/1) – (1/1).$\log_2$(1/1) = 0
  - H(Decision|Humidity>65) = -(5/13)$\log_2$(5/13) – (8/13)$\log_2$(8/13) =0.530 + 0.431 = 0.961
  - IG(Decision, Humidity<> 65) = 0.940 – (1/14)*0 – (13/14)*0.961 = 0.048
  - IV(Humidity<> 65) = -(1/14)$\log_2$(1/14) -(13/14)$\log_2$(13/14) = 0.371
  - IGR(Decision, Humidity<> 65) = 0.048/0.371= 0.126

Day	**Humidity**	**Decision**
7	65	Yes
6	70	No
9	70	Yes
11	70	Yes
13	75	Yes
3	78	Yes
5	80	Yes
10	80	Yes
14	80	No
1	85	No
2	90	No
12	90	Yes
8	95	No
4	96	Yes

Example: Computing conditional entropy on continuous feature

- Check 70 as a threshold for humidity
  - $H(\text{Decision}|\text{Humidity}<=70) = -(1/4)\log_2(1/4) - (3/4)\log_2(3/4) = 0.811$
  - $H(\text{Decision}|\text{Humidity}>70) = -(4/10)\log_2(4/10) - (6/10)\log_2(6/10) = 0.970$
  - $IG(\text{Decision}, \text{Humidity}<> 70) = 0.940 - (4/14)*(0.811) - (10/14)*(0.970) = 0.940 - 0.231 - 0.692 = 0.014$
  - $IV(\text{Humidity}<> 70) = -(4/14)\log_2(4/14) -(10/14)\log_2(10/14) = 0.863$
  - $IGR(\text{Decision}, \text{Humidity}<> 70) = 0.016$

Day	Humidity	Decision
7	65	Yes
6	70	No
9	70	Yes
11	70	Yes
13	75	Yes
3	78	Yes
5	80	Yes
10	80	Yes
14	80	No
1	85	No
2	90	No
12	90	Yes
8	95	No
4	96	Yes

Example: Computing conditional entropy on continuous feature

- List of all IG and IGR for all possible threshold values
  - IG(Decision, Humidity<> 65) = 0.048 , IGR(Decision, Humidity<> 65) = 0.126
  - IG(Decision, Humidity<> 70) = 0.014 , IGR(Decision, Humidity<> 70) = 0.016
  - IG(Decision, Humidity <> 78) =0.090, IGR(Decision, Humidity <> 78) =0.090
  - **IG(Decision, Humidity <> 80) = 0.101, IGR(Decision, Humidity <> 80) = 0.107**
  - IG(Decision, Humidity <> 85) = 0.024, IGR(Decision, Humidity <> 85) = 0.027
  - IG(Decision, Humidity <> 90) = 0.010, IGR(Decision, Humidity <> 90) = 0.016
  - IG(Decision, Humidity <> 95) = 0.048, IGR(Decision, Humidity <> 95) = 0.128
  - Here, I ignore the value 96 as threshold because humidity cannot be greater than this value

Day	Humidity	Decision
7	65	Yes
6	70	No
9	70	Yes
11	70	Yes
13	75	Yes
3	78	Yes
5	80	Yes
10	80	Yes
14	80	No
1	85	No
2	90	No
12	90	Yes
8	95	No
4	96	Yes

Finishing the example tree

- Information Gain maximizes when threshold is equal to 80 for humidity. This means that we need to compare other nominal attributes and comparison of humidity to 80 to create a branch in our tree.
- Temperature feature is continuous as well. When applying binary split to temperature for all possible split points, the following decision rule maximizes for both gain and gain ratio.
  - IG(Decision, Temperature <> 83) = 0.113, IGR(Decision, Temperature<> 83) = 0.305
- Summary of calculated IG and IGR

Attribute	IG	IGR
Wind	0.049	0.049
Outlook	0.246	0.155
Humidity <> 80	0.101	0.107
Temperature <> 83	0.113	0.305

- If we will use IG metric (ID3) , then outlook will be the root node because it has the highest IG value. On the other hand, if we use IGR metric (C4.5), then temperature will be the root node because it has the highest IGR value.

Finishing the example tree

- Imagine we use ID3 technique and use IG
  - Root attribute = Outlook.
    - Nominal with 3 possible values =>Repeat the process 3 times
  - **Outlook = Sunny**
    - Split humidity for greater than 80 => all instances YES, and less than or equal to 80 => all instances NO

Day	Outlook	Temp.	Hum. > 80	Wind	Decision
1	Sunny	85	Yes	Weak	No
2	Sunny	80	Yes	Strong	No
8	Sunny	72	Yes	Weak	No
9	Sunny	69	No	Weak	Yes
11	Sunny	75	No	Strong	Yes

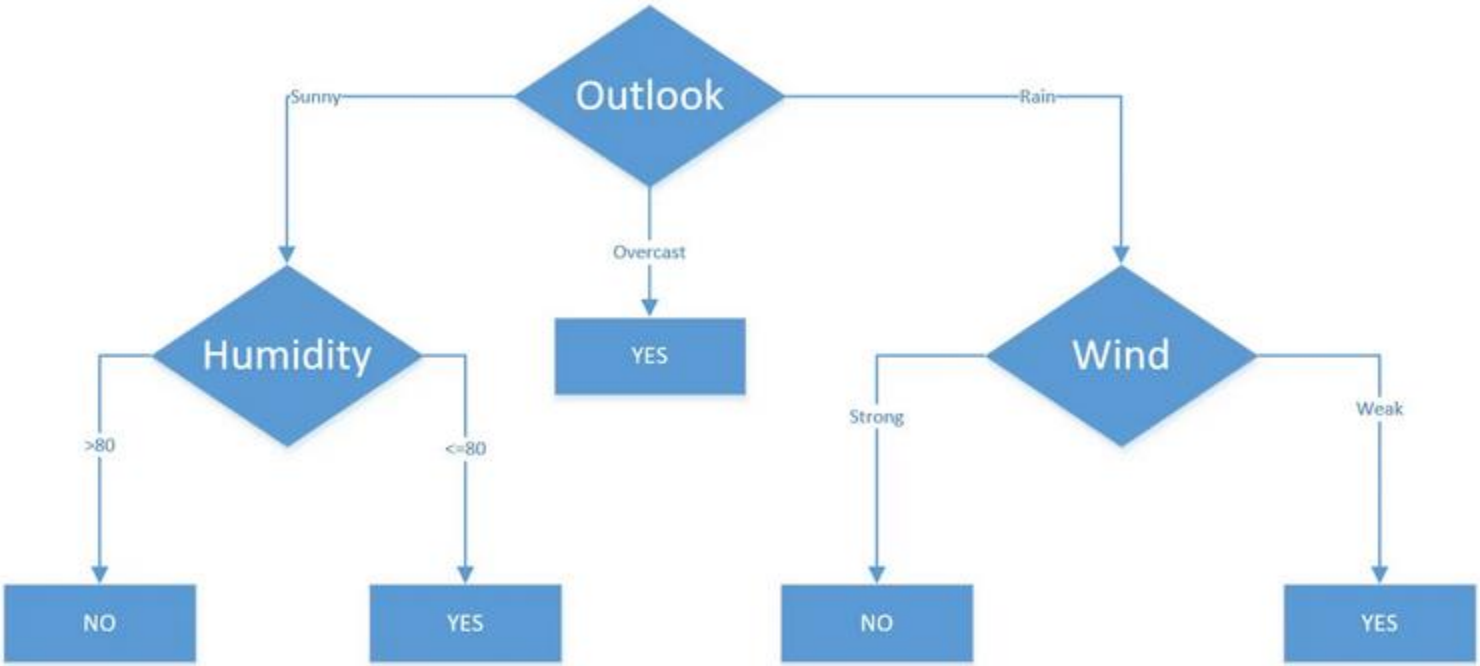
  - **Outlook = Overcast**
    - All instances yes => leaf node

Day	Outlook	Temp.	Hum. > 80	Wind	Decision
3	Overcast	83	No	Weak	Yes
7	Overcast	64	No	Strong	Yes
12	Overcast	72	Yes	Strong	Yes
13	Overcast	81	No	Weak	Yes

  - **Outlook = Rain**
    - Look at wind, if weak=> all instances YES if strong =>all instances NO

Day	Outlook	Temp.	Hum. > 80	Wind	Decision
4	Rain	70	Yes	Weak	Yes
5	Rain	68	No	Weak	Yes
6	Rain	65	No	Strong	No
10	Rain	75	No	Weak	Yes
14	Rain	71	No	Strong	No

Final tree


Outlook
Sunny
Overcast
Rain
Humidity
YES
Wind
>80
<=80
Strong
Weak
NO
YES
NO
YES

Decision trees: pros & cons

- **Advantages**
  - Simple implementation.
  - Little data preparation
  - Variable selection
  - Interpretable results.
  - Fast training & prediction
  - Non linear relationships do not affect performance
- **Drawbacks**
  - Not very accurate predictions.
  - Low variance -> rigid models
    - Performance severely affected by resampling data
  - However, they can be used as base learners for an ensemble.
    - Random forests

https://scikit-learn.org/stable/modules/tree.html