Decision Trees

Beatriz Sevilla Villanueva based on Tomas Aluja Slides

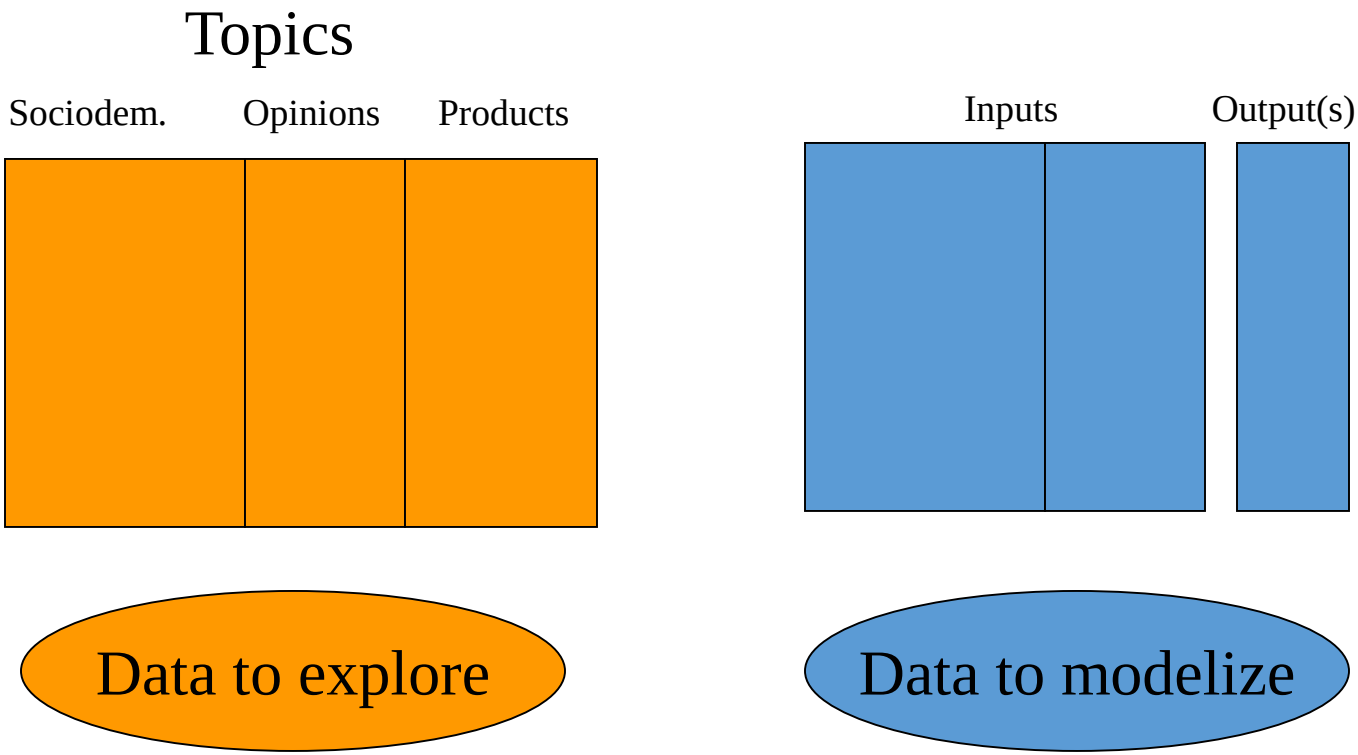
and on Mario Martin Slides

Decision Trees. Contents

- Data
- Decision Trees in Data Mining
  - Rules
- Decision Tree Learning
- Algorithm
  - ID3
    - Entropy
    - Information Gain
  - C4.5
    - Information Gain Ratio
  - CHIAD
  - CART

Two Data Matrix to Analyse

- Data in *Data Mining*:
  - massive, secondary, non-random, with errors, missings …



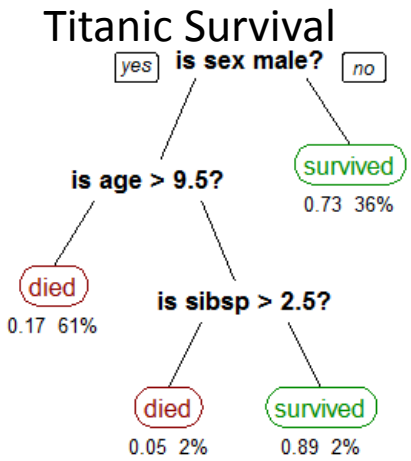
Decision trees in Data Mining

- The central task in DM is prediction
  - Classification if response is categorical
  - Regression if response is continuous)

- There are plenty of methods (models) to perform prediction:
  - Linear regression, logistic regression, discriminant analysis, Naive Bayes, Knn, Neural networks, SVM, …

- The decision trees are a **non parametric method** to perform predictions. Its popularity comes from the simplicity of the results (everyone can understand them) and their direct operationally

Decision tree in Data Mining



- Commonly used in data mining/ machine learning
- Goal: create a model that predicts the value of a target variable based on several input variables.
  - Example:
    - Each interior node: input variables;
    - Edges to children for each of the possible values of that input variable.
    - Leaf: a value of the target variable given the values of the input variables represented by the path from the root to the leaf.
- In data mining, decision trees can be described also as the combination of mathematical and computational techniques to aid the description, categorization and generalization of a given set of data.
- Data comes in records of the form:

$$(X, Y) = (X_1, X_2, \ldots, X_k, Y)$$

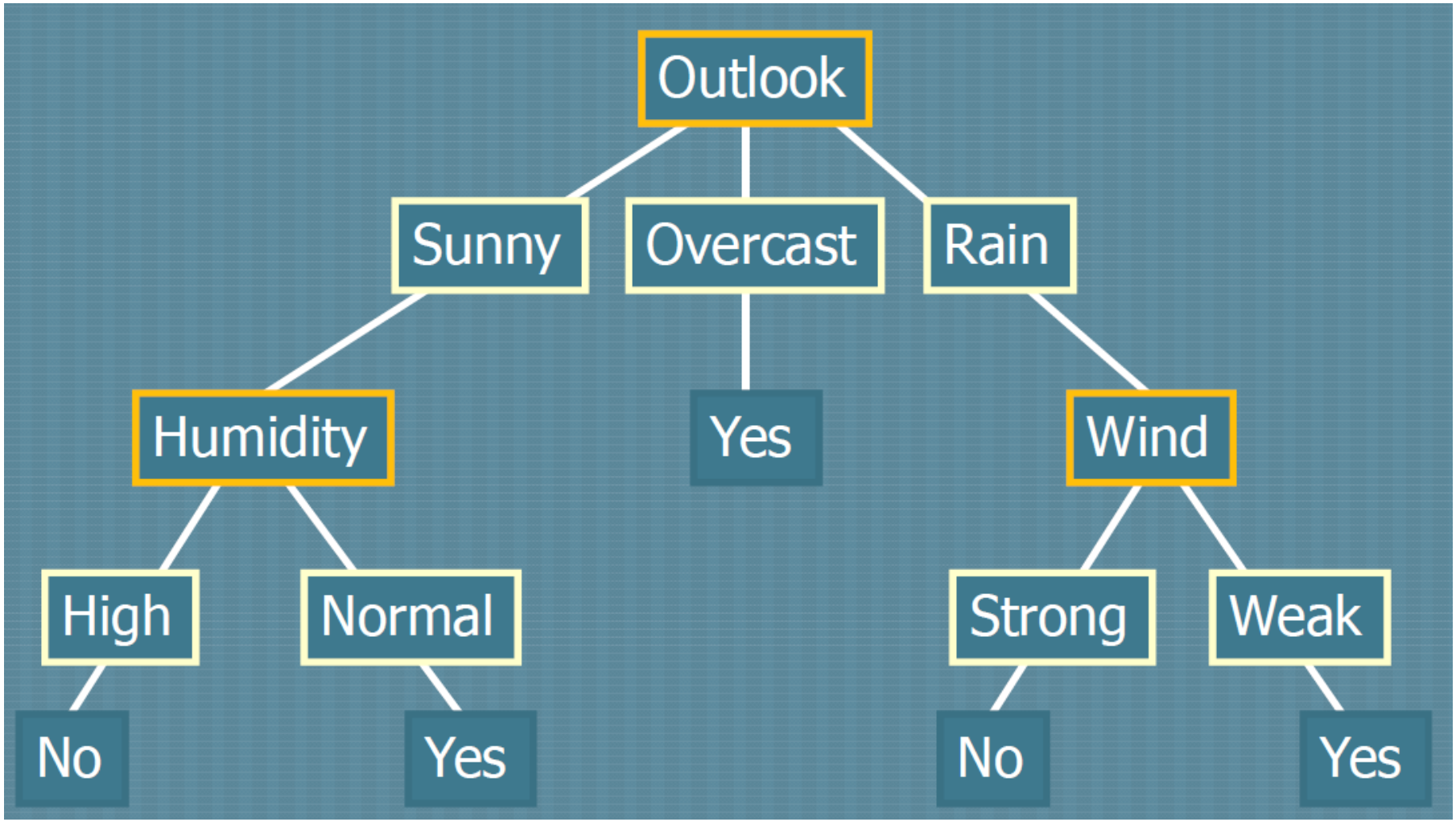
The dependent variable, Y, is the target variable that we are trying to understand, classify or generalize. The vector X is composed of the features, X_1, X_2, etc., that are used for that task.

Playing Tennis??

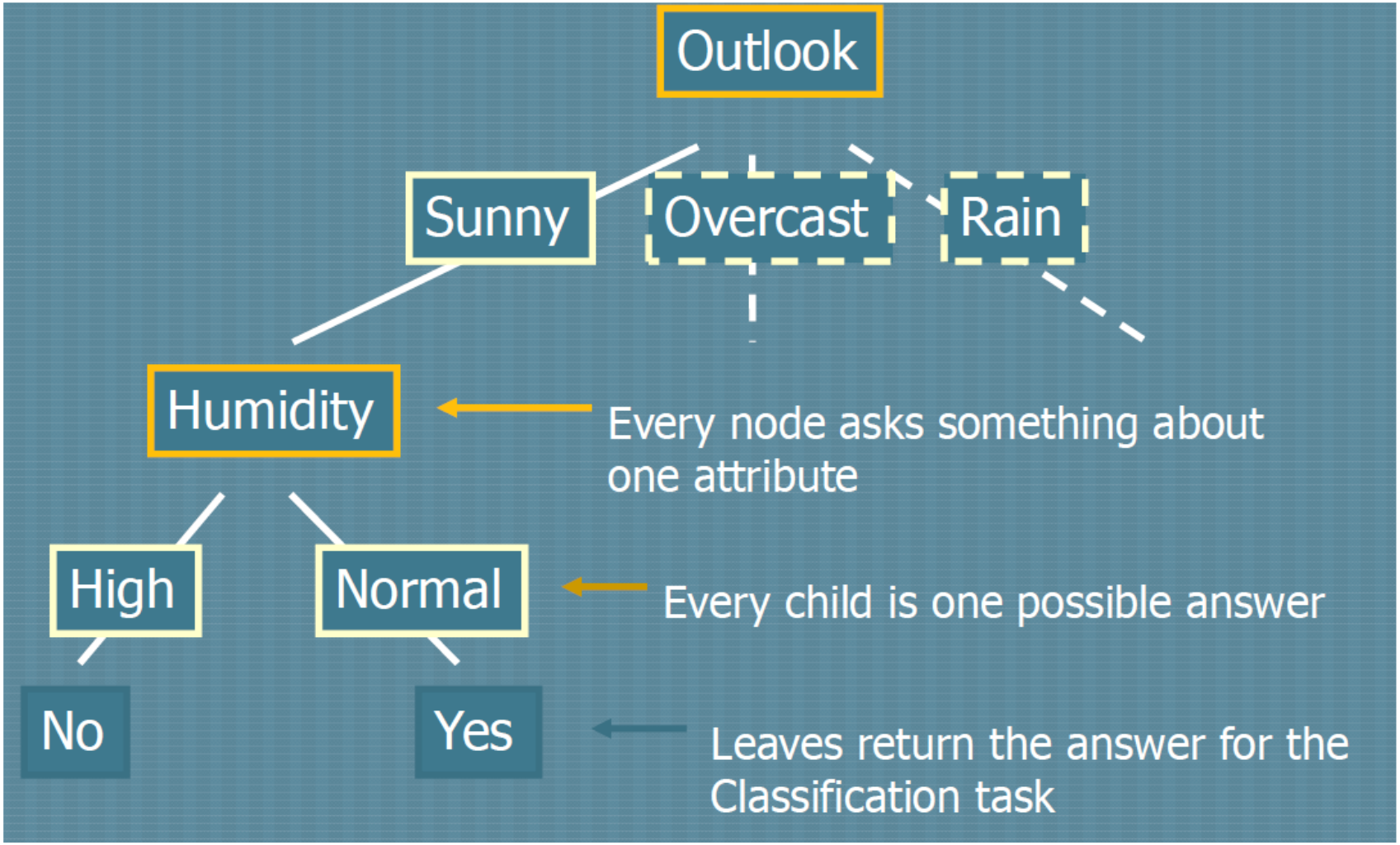
- Data:

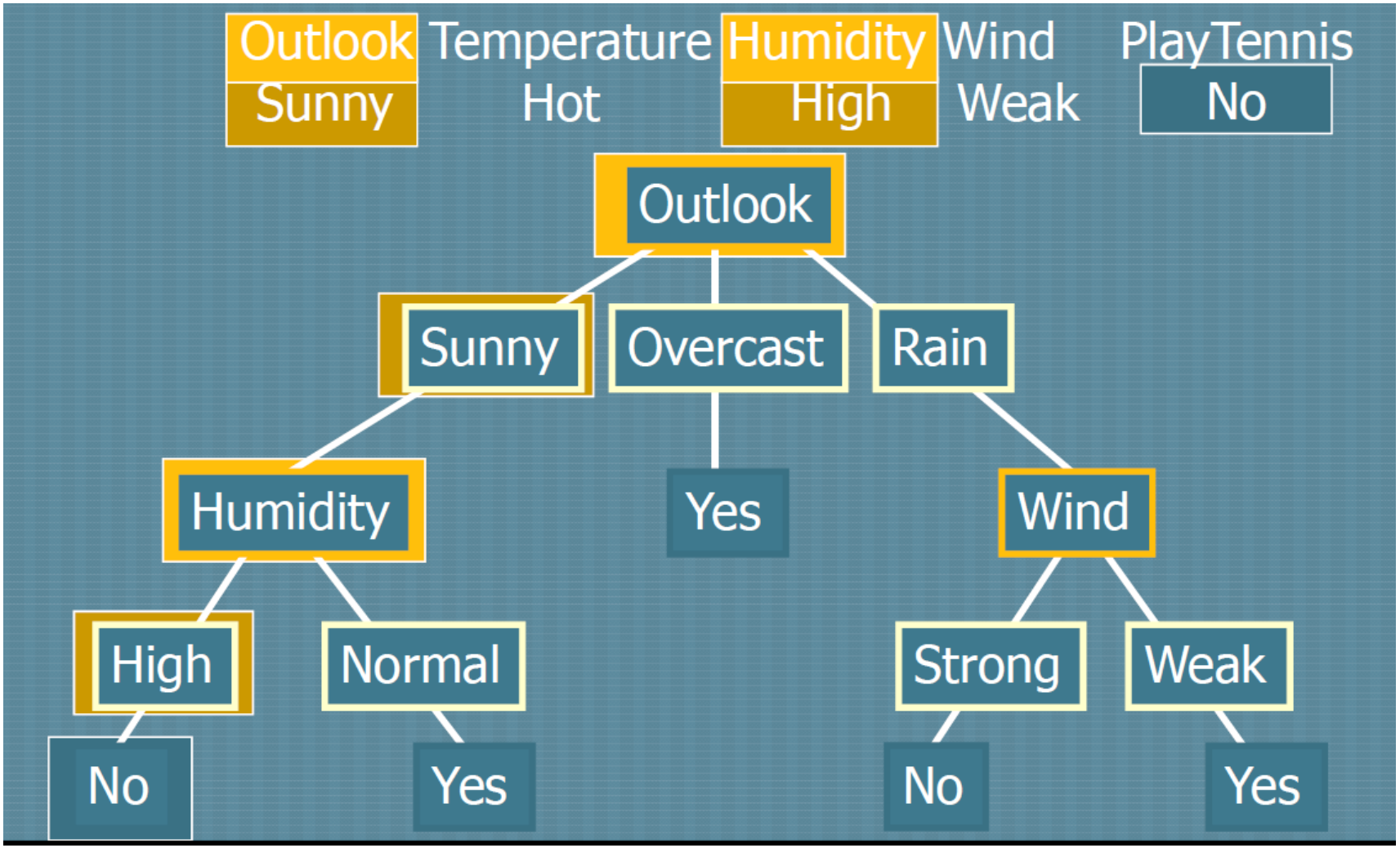
Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Weak	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cold	Normal	Weak	Yes
D10	Rain	Mild	Normal	Strong	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Playing Tennis??

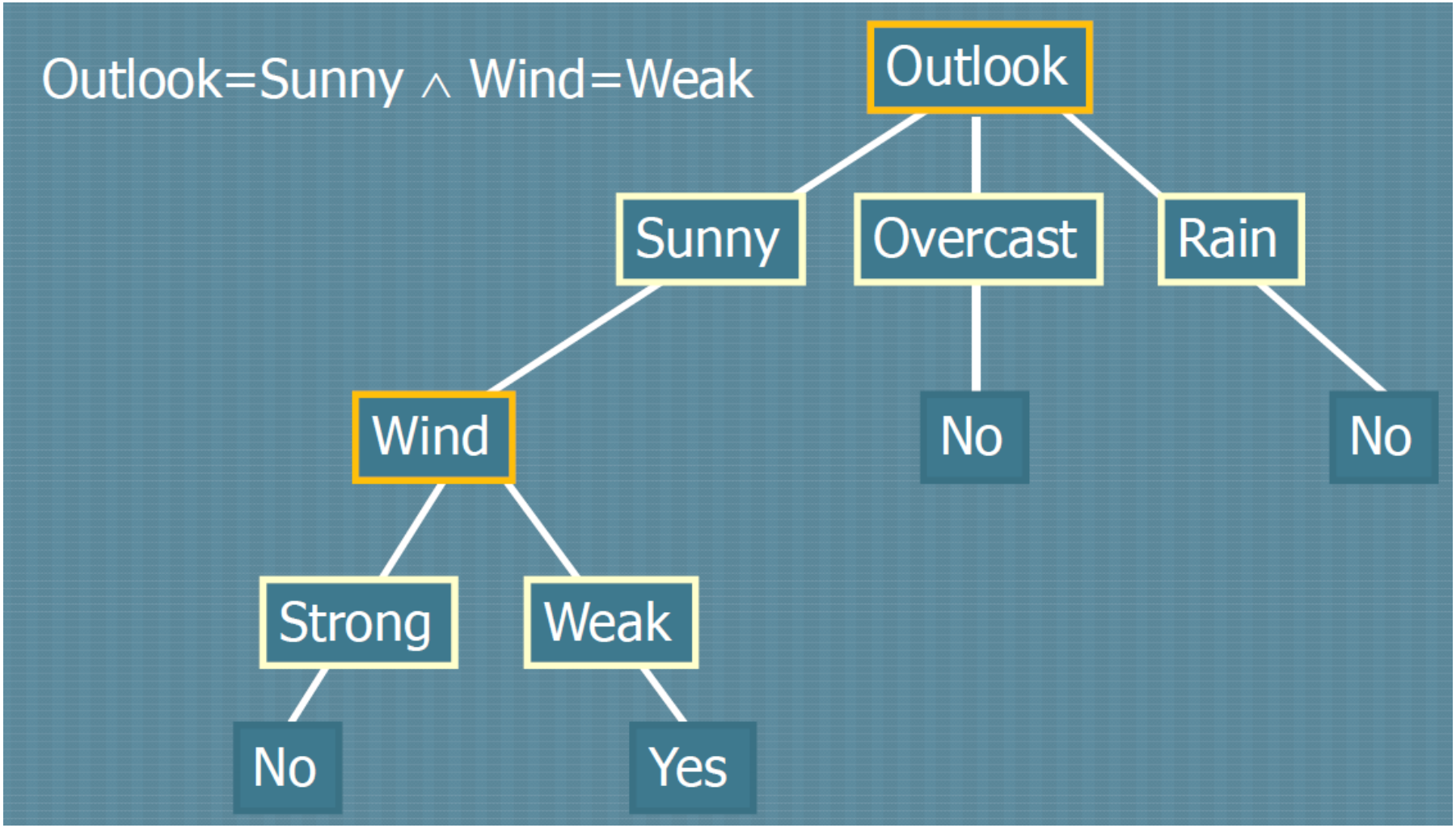


Playing Tenis??

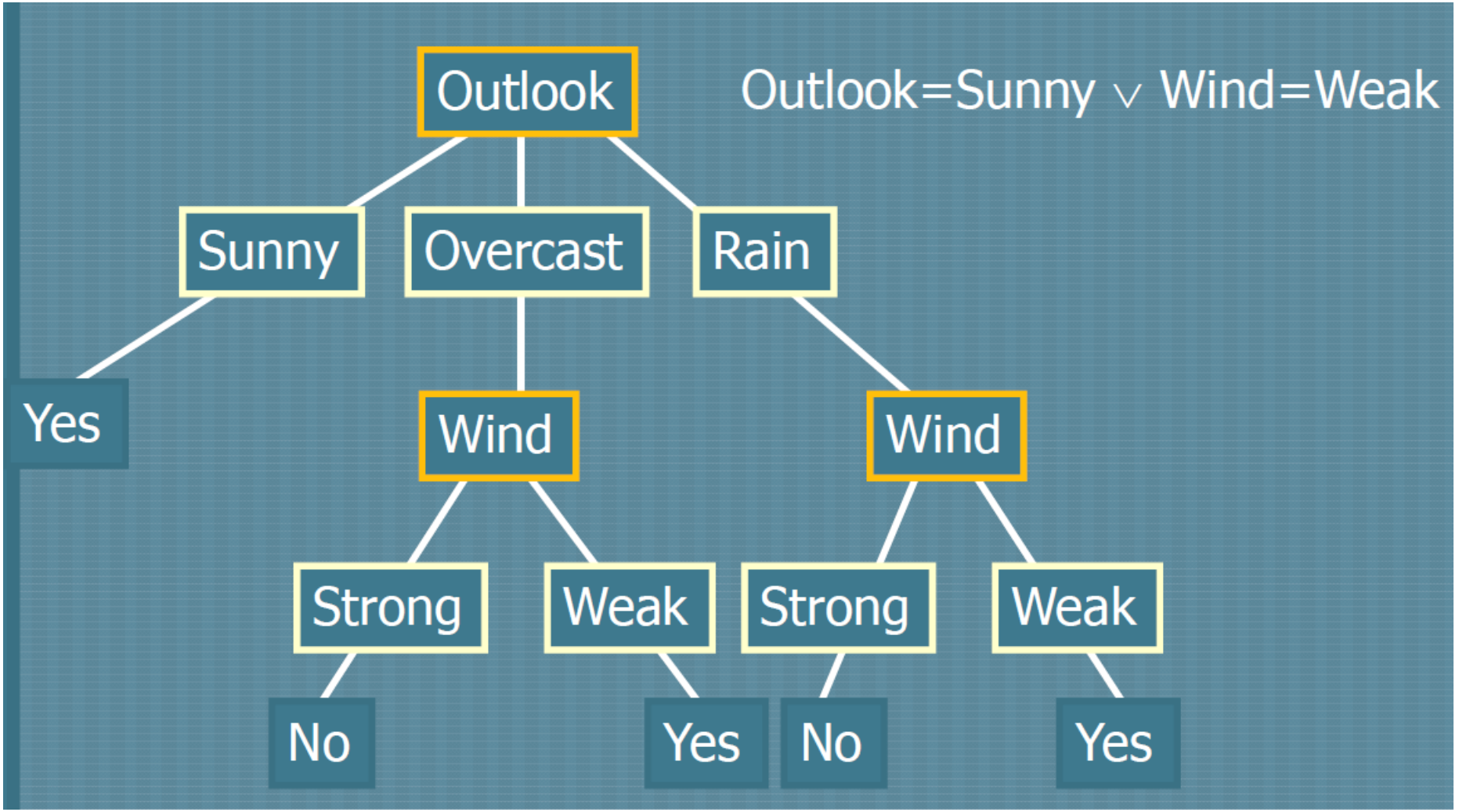



Outlook
Sunny
Temperature
Hot
Humidity
High
Wind
Weak
PlayTennis
No
Outlook
Sunny
Overcast
Rain
Humidity
Yes
Wind
High
Normal
Strong
Weak
No
Yes
No
Yes

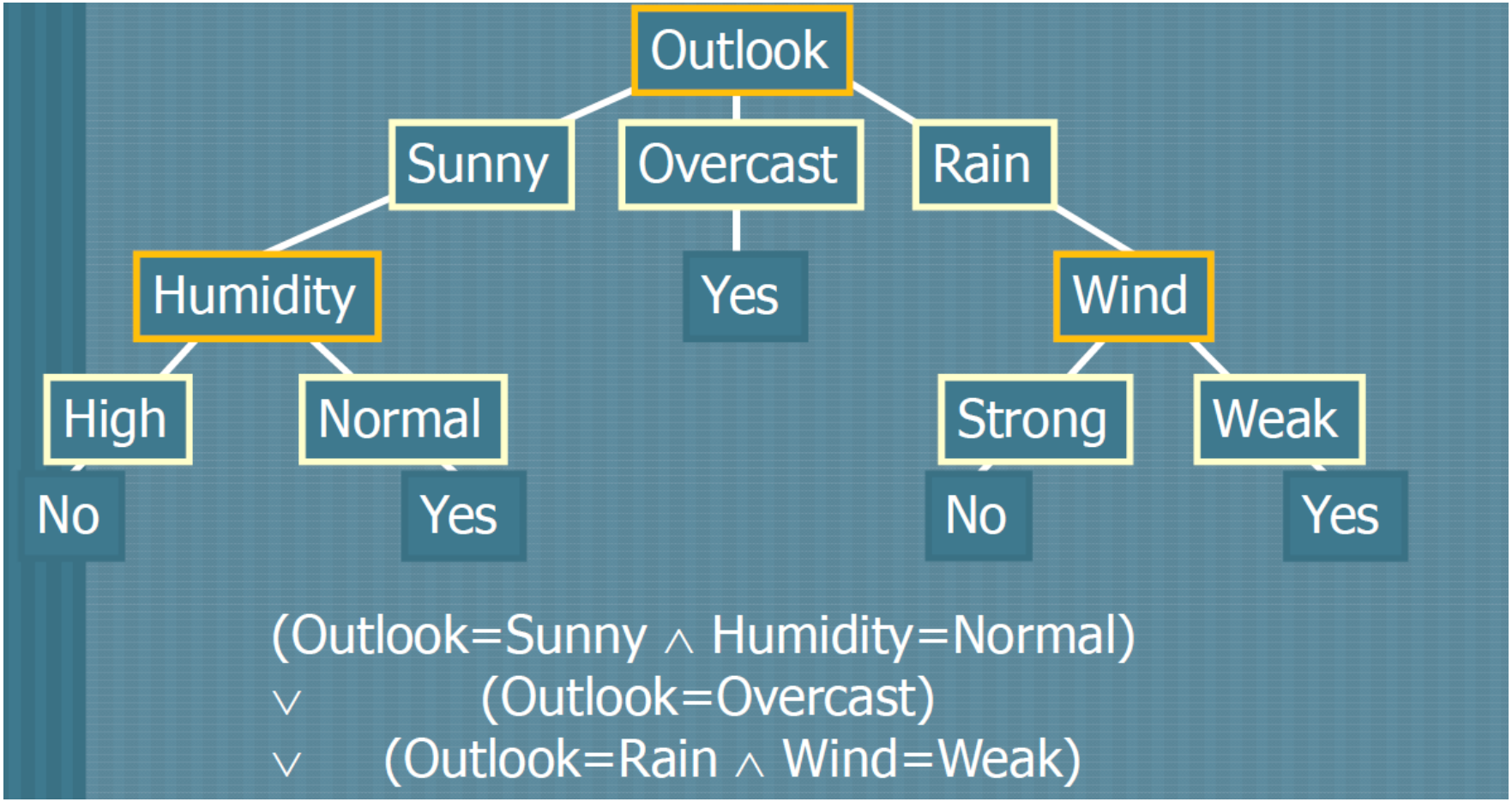
Decision Tree can represent conjuntions



Decision Tree can represent Disjunctions



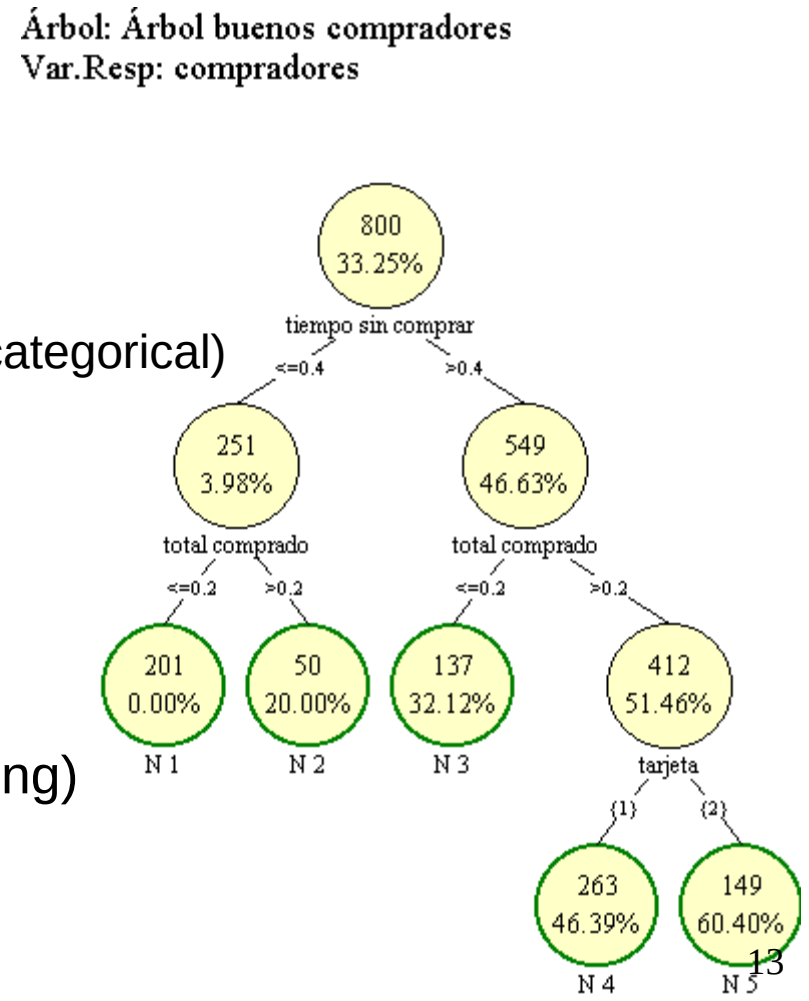
Complex Rules



Decision Tree. Purpose

To segment the given data in order to find homogeneous groups respect to the response variable.

Visual output



Trees may differ:

- Type of the response variable (numerical/categorical)
- Type of explanatory variables
  (binary, nominal, ordinal, numerical)
- Tree type: binary or multi-way

- Split criterion (information, gini, …)
- Stop criterion (pre-pruning, post-pruning)

Decision Trees in Data Mining

- Decision trees used in data mining are of two main types:
  - **Classification tree** analysis is when the predicted outcome is the class to which the data belongs (e.g. playtennis?, credit, type of car, etc.).
    - Response: categorical variable
  - **Regression tree** analysis is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient's length of stay in a hospital, etc.).
    - Response: numerical variable

Decision Tree Learning

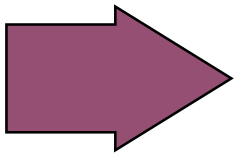
- From a training set, many possible trees
- Do not learn any tree but the simplest one! (simplicity for generalization)
- The simplest tree is the shortest one.
- One approach: Creating all the trees and keep the shortest one
- ... But this approach requires a too much computation time
- **Greedy Algorithm** builds short decision trees more efficiently
  - Not ensure to get the shortest one

Decision Tree Learning

- There are many specific decision-tree algorithms.
  - ID3 (Iterative Dichotomiser 3)
  - C4.5 (successor of ID3)
  - CART (Classification And Regression Tree)
  - CHAID (CHi-squared Automatic Interaction Detector). Performs multi-level splits when computing classification trees.
  - MARS: extends decision trees to handle numerical data better.
  - Conditional Inference Trees. Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid overfitting. This approach results in unbiased predictor selection and does not require pruning.
  - Etc.

Decision Tree Learning. Algorithm (top-down)

1. To set all individuals in the root node
2. Evaluate all possible splits and find the optimal partition in children nodes.
3. For every child node: Decide if we stop the process or we go back to step 2.



We need

- a split criterion
- a stop criterion (if prepruning)

How many splits can we make in a node

- Splits are defined by the number and the type of the explanatory variables and the type of the tree

	Multi way	Binary tree
Binary	1	1
Nominal	1	$2^{q-1}-1$
Ordinal	1	q-1
Numerical	n_t-1	n_t-1

★ Attention

★ Attention

ID3

- invented by Ross Quinlan (1986)
- Categorical or discrete variables
- Concept to be learnt can be described by logical rules
- Examples:
  - Medical diagnosis
  - Analysis of risk in credit
  - Classification of objects for manipulation

ID3. Algorithm Schema

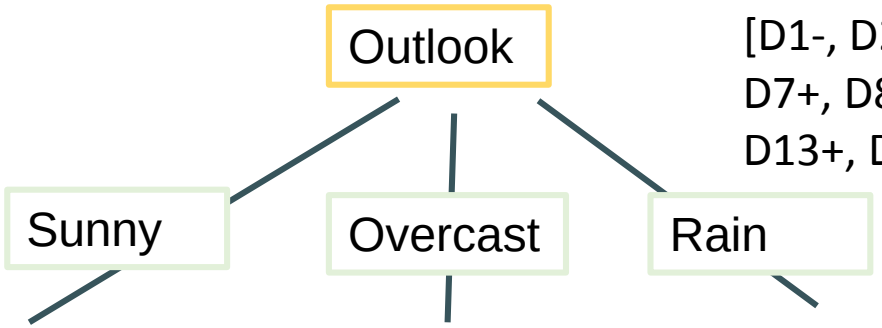
1. Set node to the training examples and put it as the root of the tree
2. A: the "best" test variables for the current node
3. For each value of variable A, create a new descendant node in the decision tree
4. Separate examples in node among descendants depending of the variable value for A
5. If there remain leafs nodes in the tree with mixed positive and negative examples, set this leaf to node and return to step 2.
6. In other case, label each leaf of the tree with the sign of the examples in the leaf, and finish

Playing Tennis??

- Data:

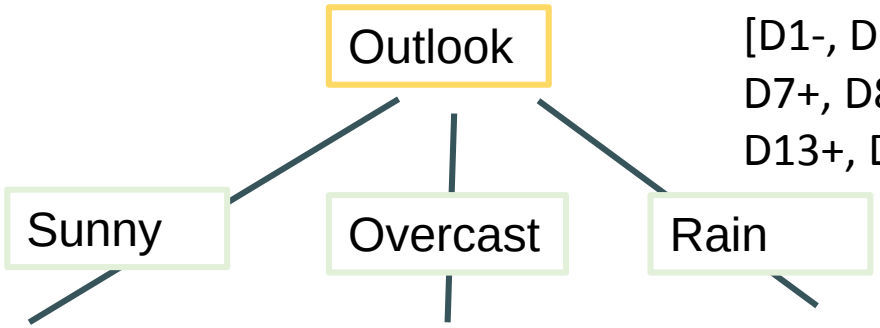
Day	*Outlook*	*Temp.*	*Humidity*	*Wind*	*PlayTennis*		
D1	Sunny	Hot	High	Weak	No	D1	-
D2	Sunny	Hot	High	Strong	No	D2	-
D3	Overcast	Hot	High	Weak	Yes	D3	+
D4	Rain	Mild	High	Weak	Yes	D4	+
D5	Rain	Cool	Normal	Weak	Yes	D5	+
D6	Rain	Cool	Normal	Strong	No	D6	-
D7	Overcast	Cool	Normal	Weak	Yes	D7	+
D8	Sunny	Mild	High	Weak	No	D8	-
D9	Sunny	Cold	Normal	Weak	Yes	D9	+
D10	Rain	Mild	Normal	Strong	Yes	D10	+
D11	Sunny	Mild	Normal	Strong	Yes	D11	+
D12	Overcast	Mild	High	Strong	Yes	D12	+
D13	Overcast	Hot	Normal	Weak	Yes	D13	+
D14	Rain	Mild	High	Strong	No	D14	-

ID3. Play Tennis



[D1-, D2-, D3+, D4+, D5+, D6-, D7+, D8-, D9+, D10+, D11+, D12+, D13+, D14-]

ID3. Play Tennis



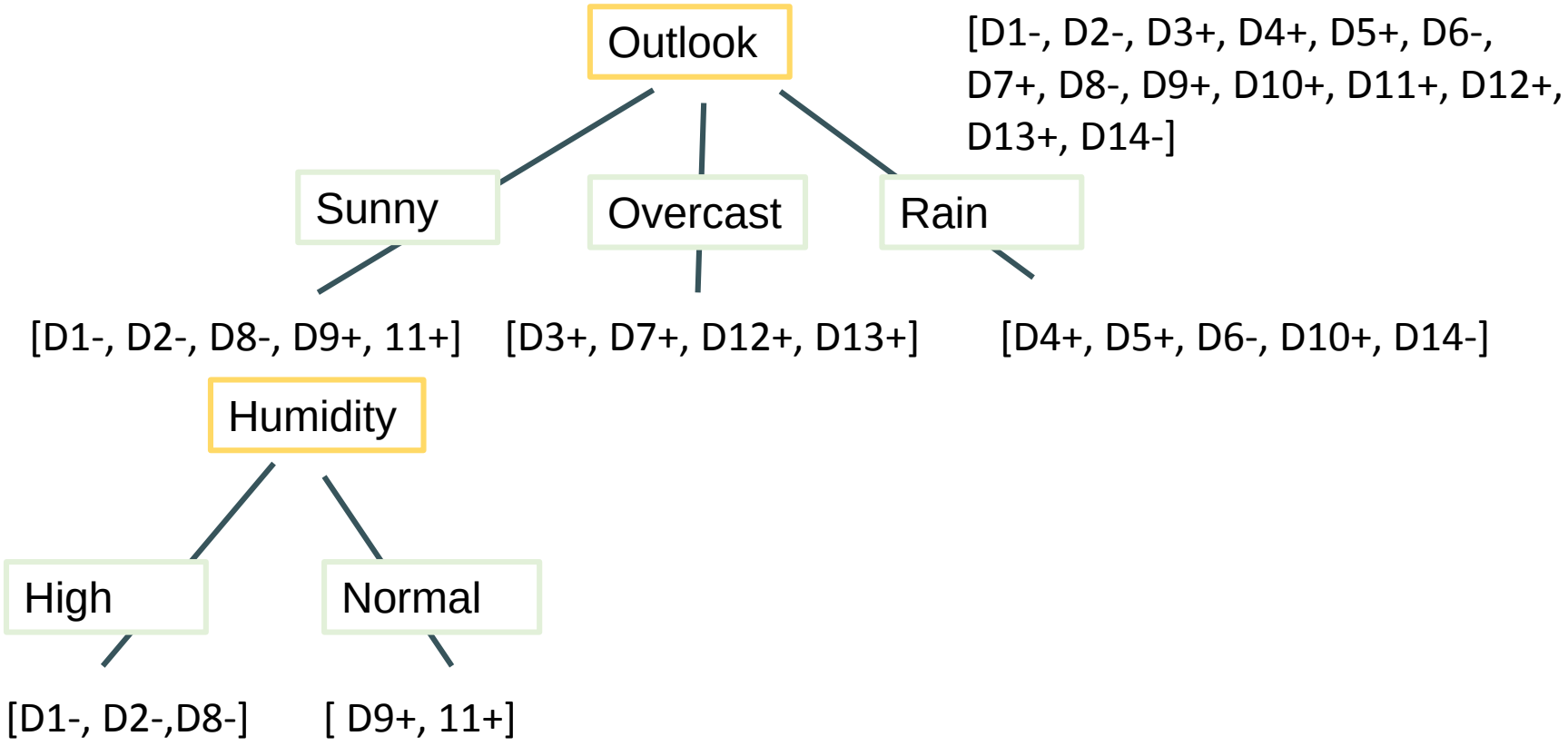
[D1-, D2-, D3+, D4+, D5+, D6-, D7+, D8-, D9+, D10+, D11+, D12+, D13+, D14-]

[D1-, D2-, D8+, D9+, 11+]

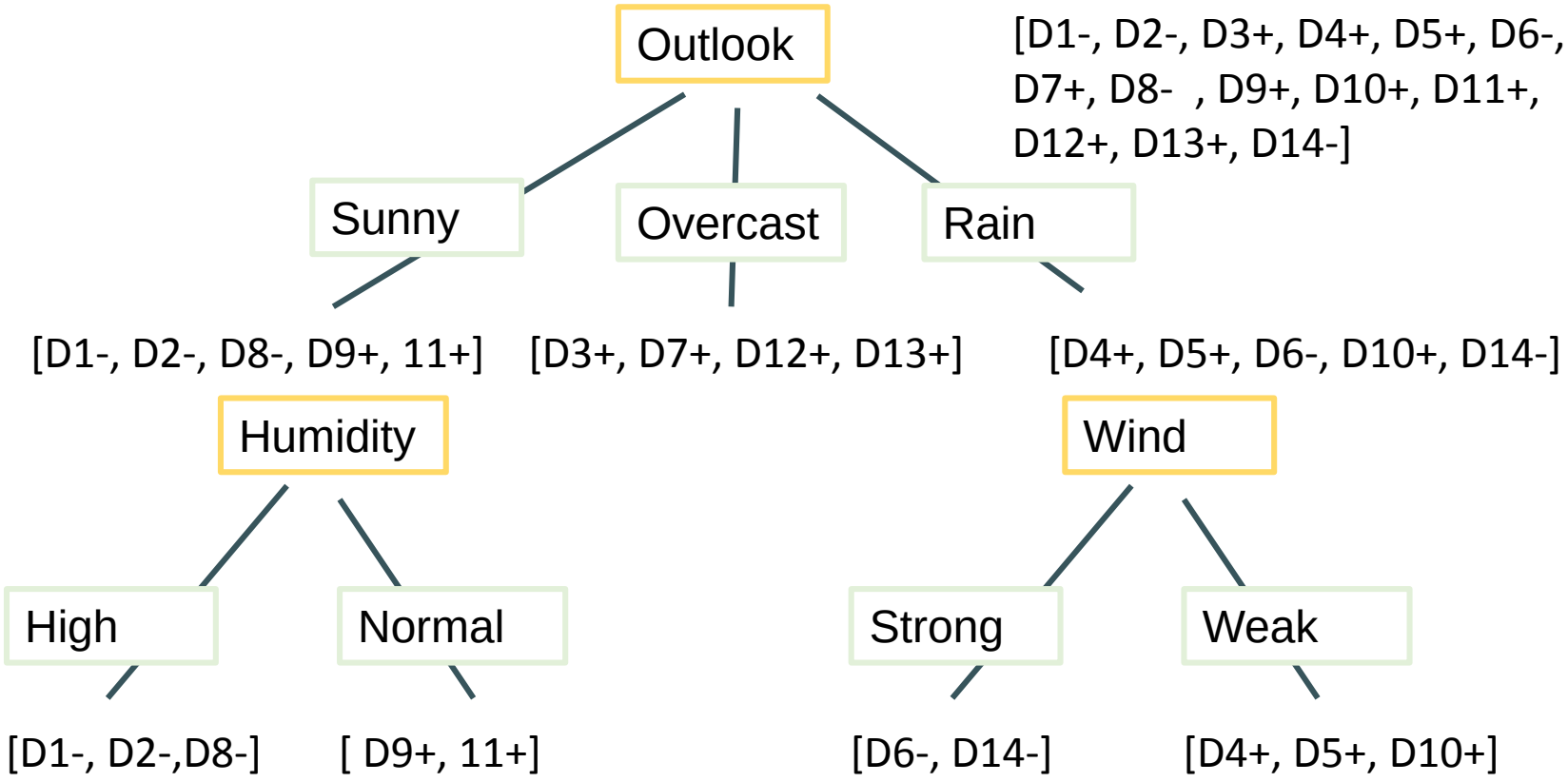
[D3+, D7+, D12+, D13+]

[D4+, D5+, D6-, D10+, D14-]

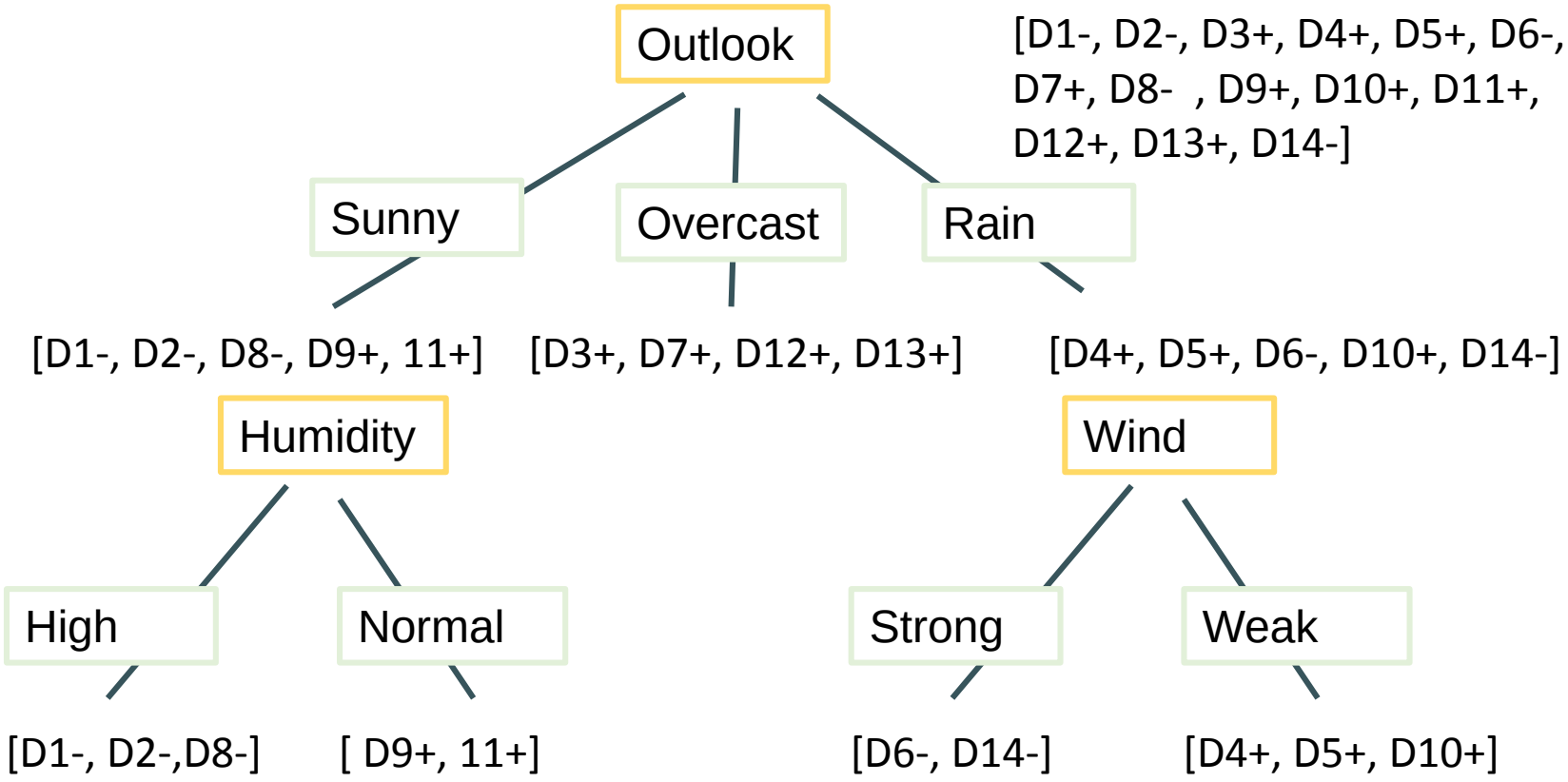
ID3. Play Tennis



ID3. Play Tennis

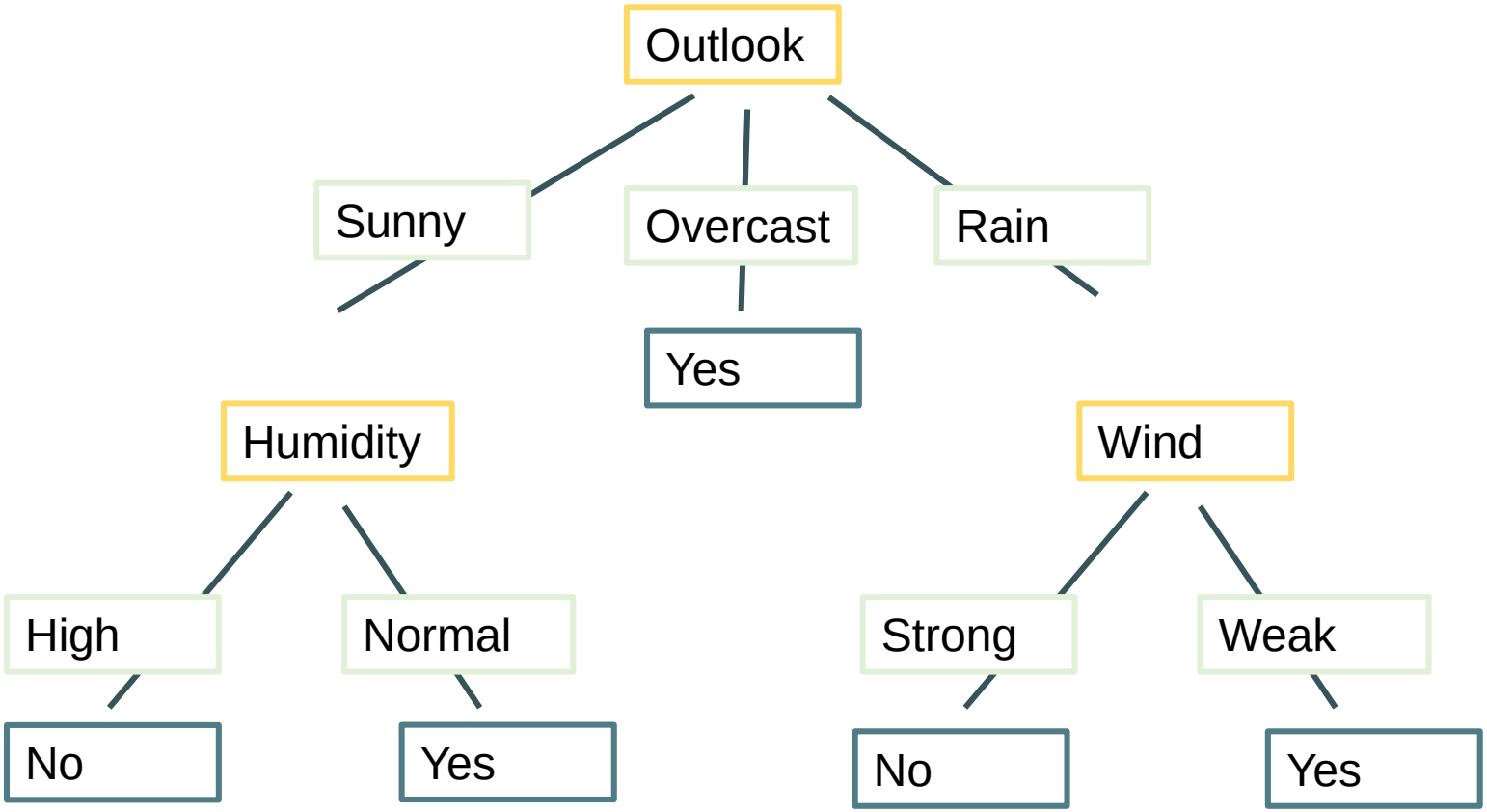


ID3. Play Tennis



How can we label these leafs??

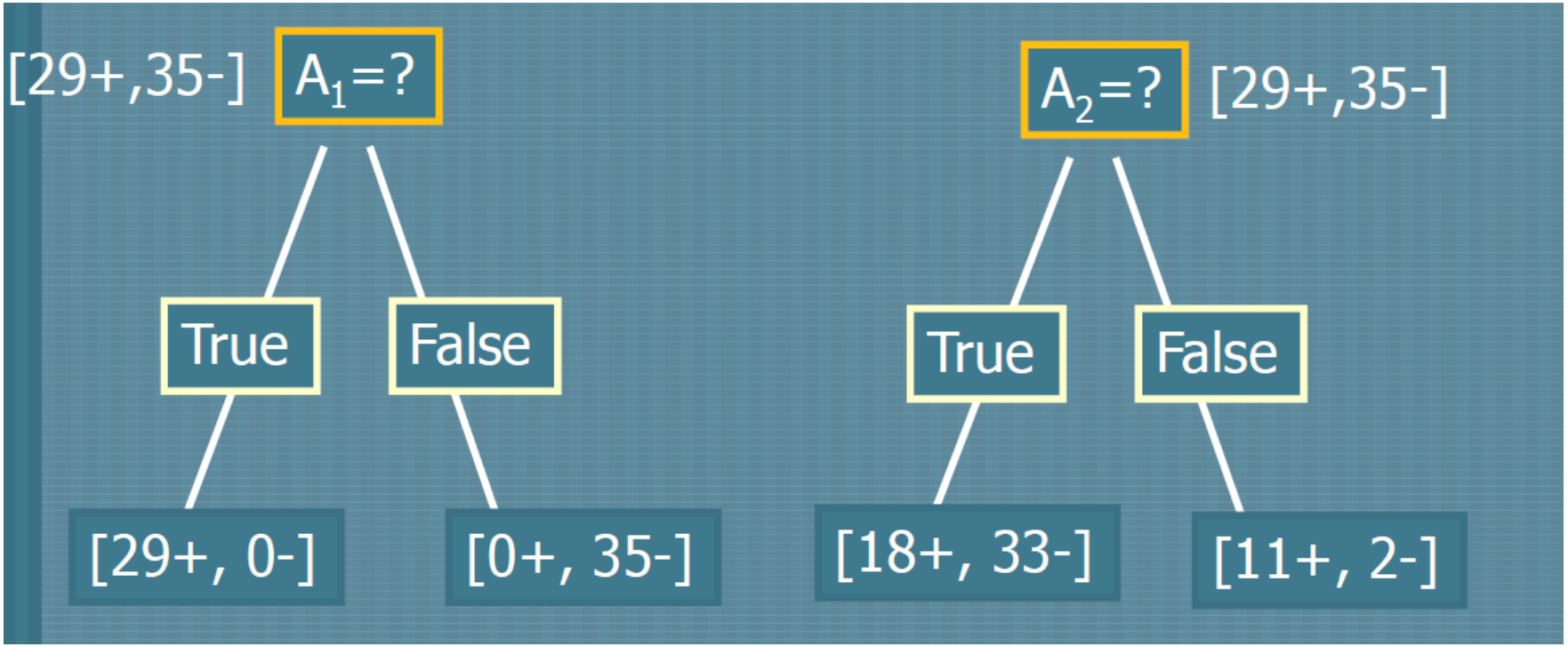
ID3. Play Tennis



How to choose the variables

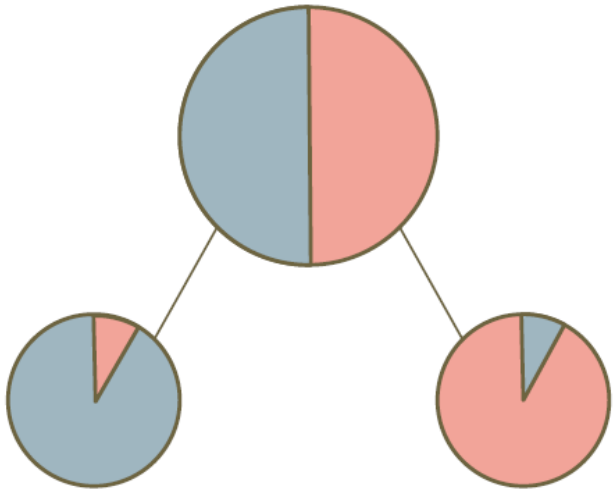
- Goal: shortest tree
  - Equivalent to separate as quick as possible the positive examples (+) to the negative ones (-)
- Split Criterion: Maximum Discriminative
  - We want to choose the variables that are the most discriminative ones.
    - Information Gain

Which variable is the best?

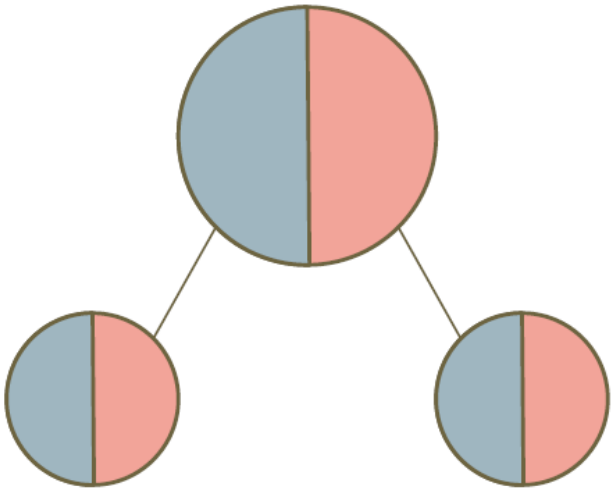


Which variable is the best?

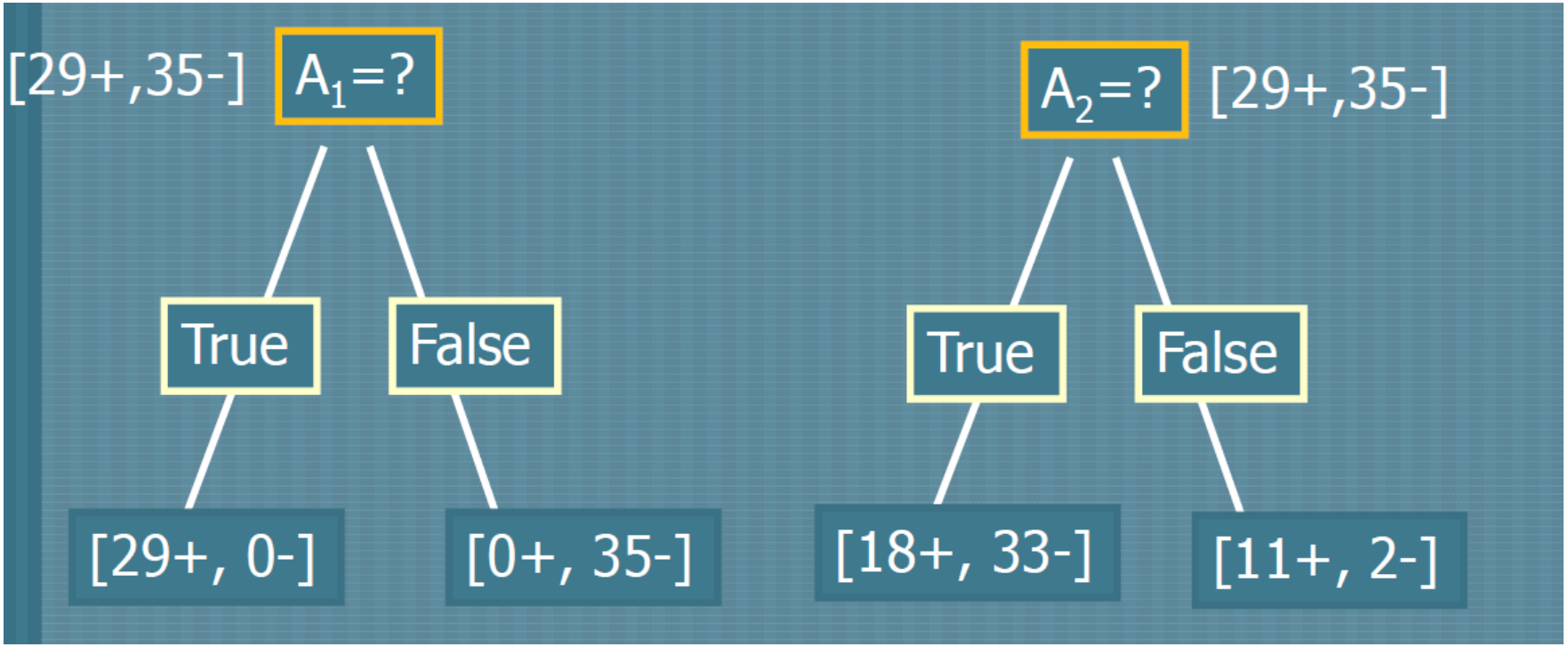
- Good split - Separates classes:



- Bad split - Classes still “impure”



Which variable is the best?



Not always so trivial!!

Information Theory. Entropy

- One answer gives us an information amount proportional to the unpredictability of the answer

$$E(X) = -\sum_{i}^{n} p_i \log(p_i)$$

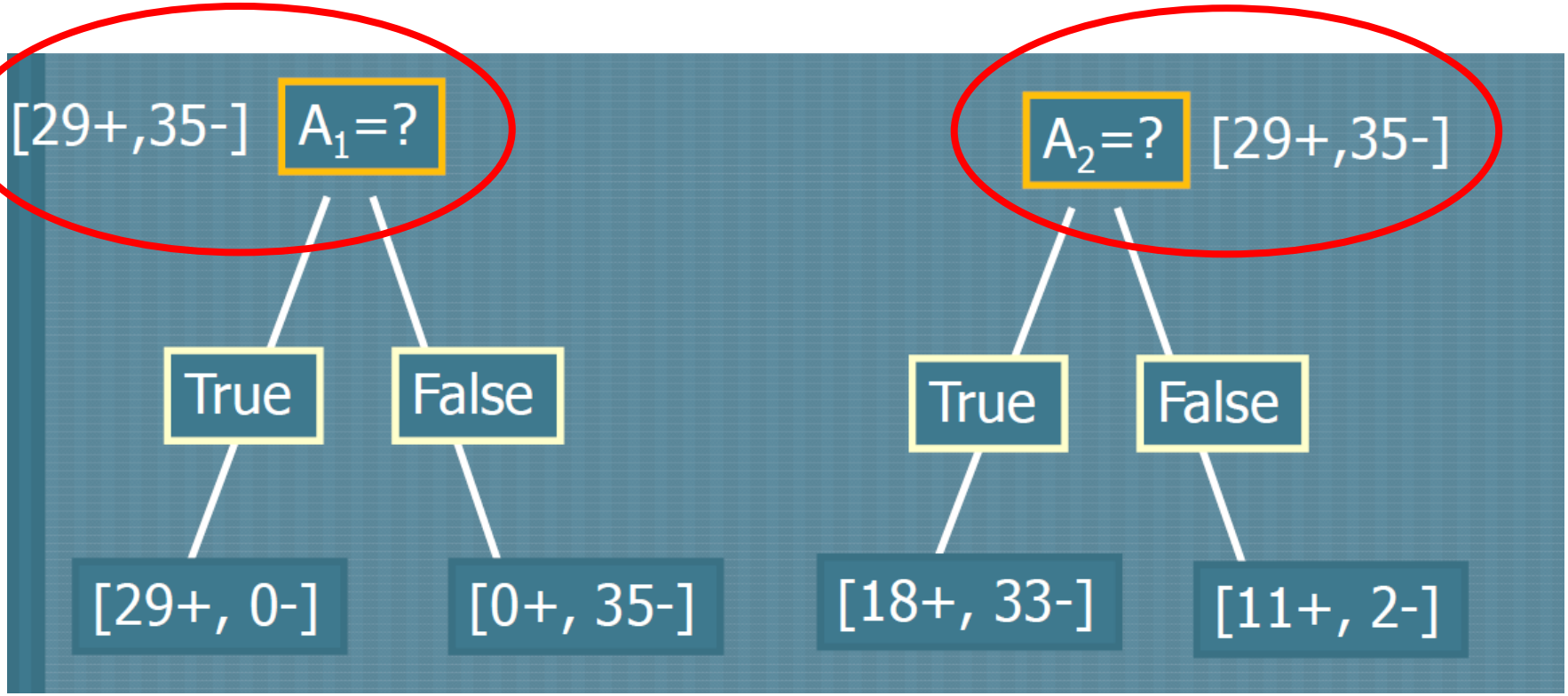
  - Where p_i is probability of answer *i*, (denotes the proportion of instances belonging to class i)
  - *n* is the number of possible answers

- E(X) – entropy of X - measures necessary bits to compress information about current distribution of + and – examples in X considering it a random variable. It is a measure of information content taken from the Information Theory field.
- Measure of disorder... That's why it is useful in ID3.

Properties of Entropy

- Maximized when elements are heterogeneous (impure, disordered):
  - $If\ p_{\mathrm{i}} = \frac{1}{N}, then\ Entropy = -N \cdot \frac{1}{i} \log_2 \frac{1}{i} = \log_2 N$
- Minimized when elements are homogenous (pure, ordered):
  - $If\ p_i = 1\ or\ p_i = 0, then\ Entropy = 0$

Which variable is the best?



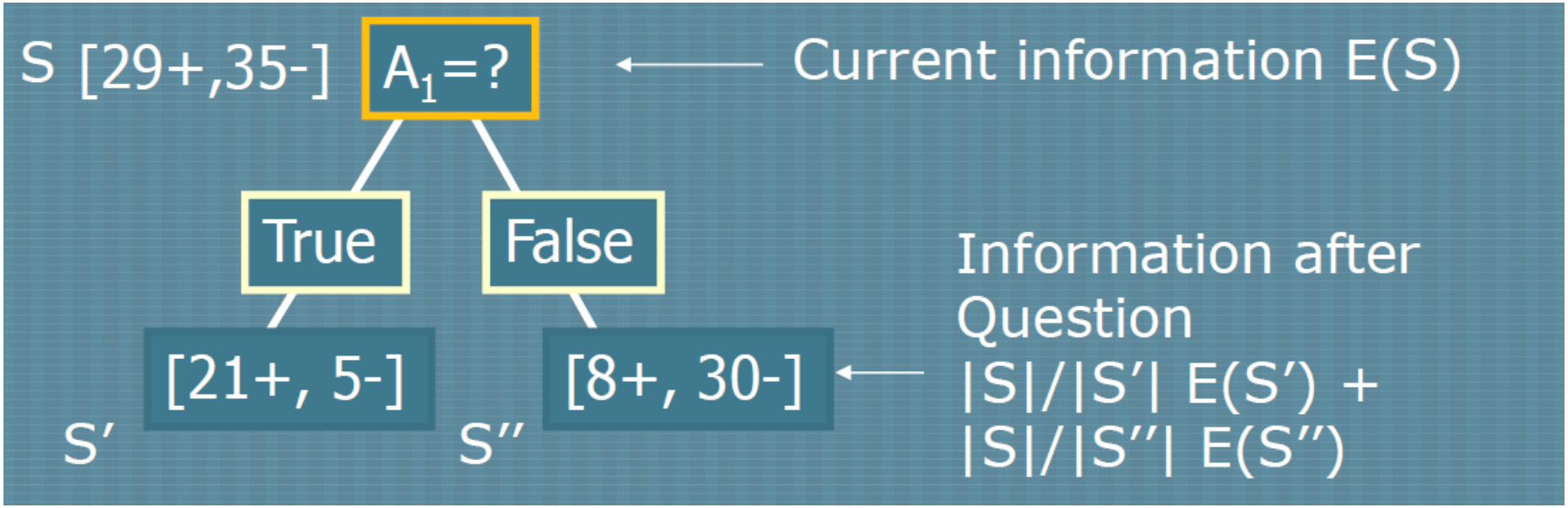
$$E(A_1) = -\frac{29}{64}\log\left(\frac{29}{64}\right) - \frac{35}{64}\log\left(\frac{35}{64}\right)$$

$$E(A_2) = -\frac{29}{64}\log\left(\frac{29}{64}\right) - \frac{35}{64}\log\left(\frac{35}{64}\right)$$

Entropy is the same!!

Information Gain

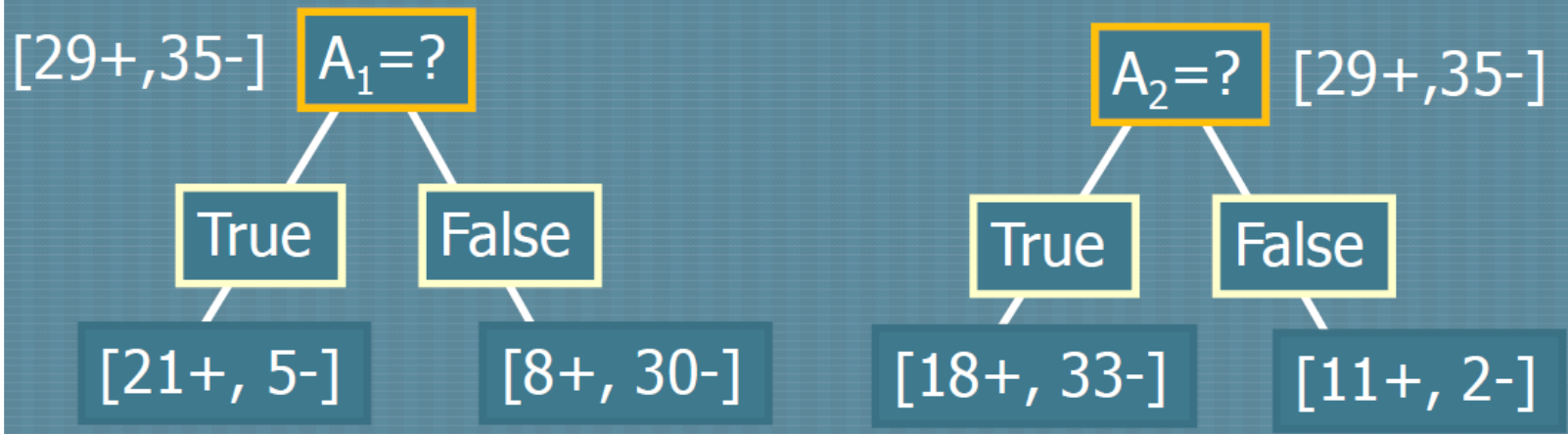
- Information Gain (IG) obtained from an answer is the difference of information before and after the answer



Information Gain

$$G(S,A)=E(S) - \sum_{v \in values(A)} |S_v|/|S| \; E(S_v)$$

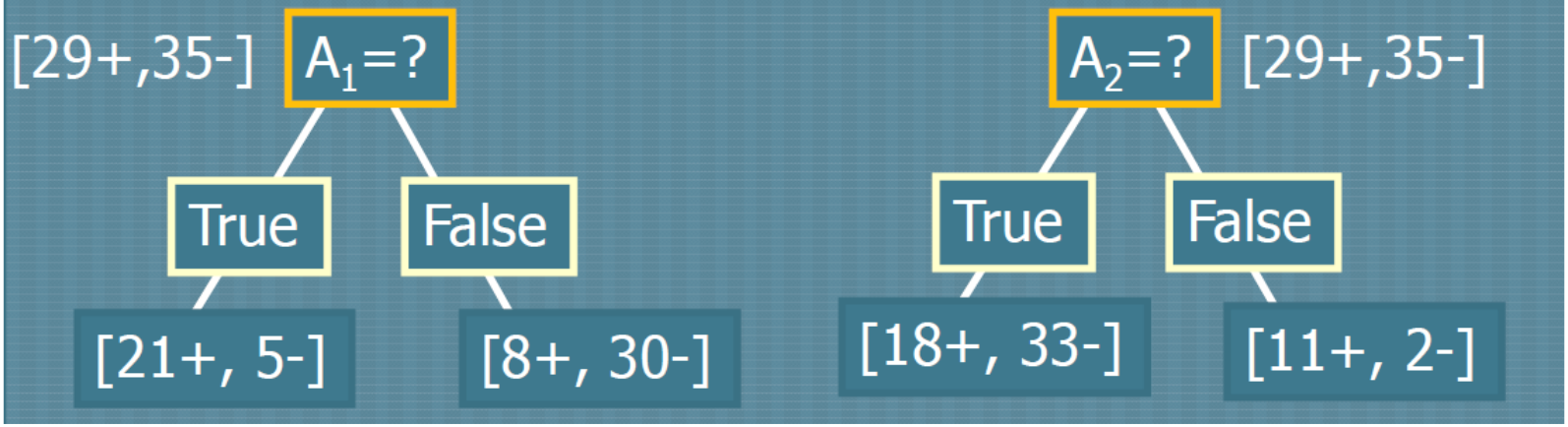
$$E([29+,35-]) = -29/64 \log_2 29/64 - 35/64 \log_2 35/64 = 0.99$$



Information Gain

E([21+,5-]) = 0.71
E([8+,30-]) = 0.74
$G(S,A_1)=E(S)$
-26/64*E([21+,5-])
-38/64*E([8+,30-])
=0.27

E([18+,33-]) = 0.94
E([8+,30-]) = 0.62
$G(S,A_2)=E(S)$
-51/64*E([18+,33-])
-13/64*E([11+,2-])
=0.12



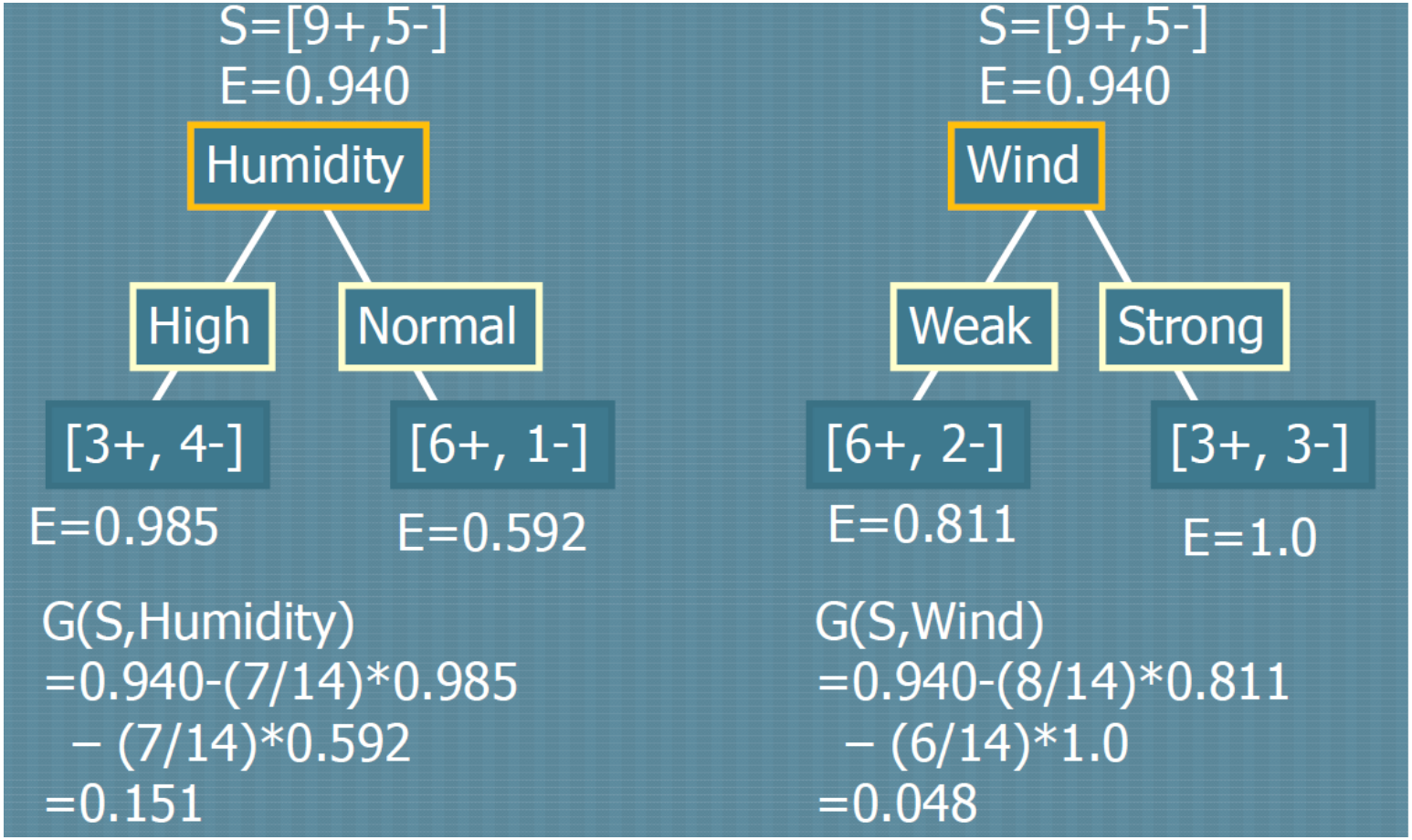
Which variable is the best??

- The one with highest information gain!

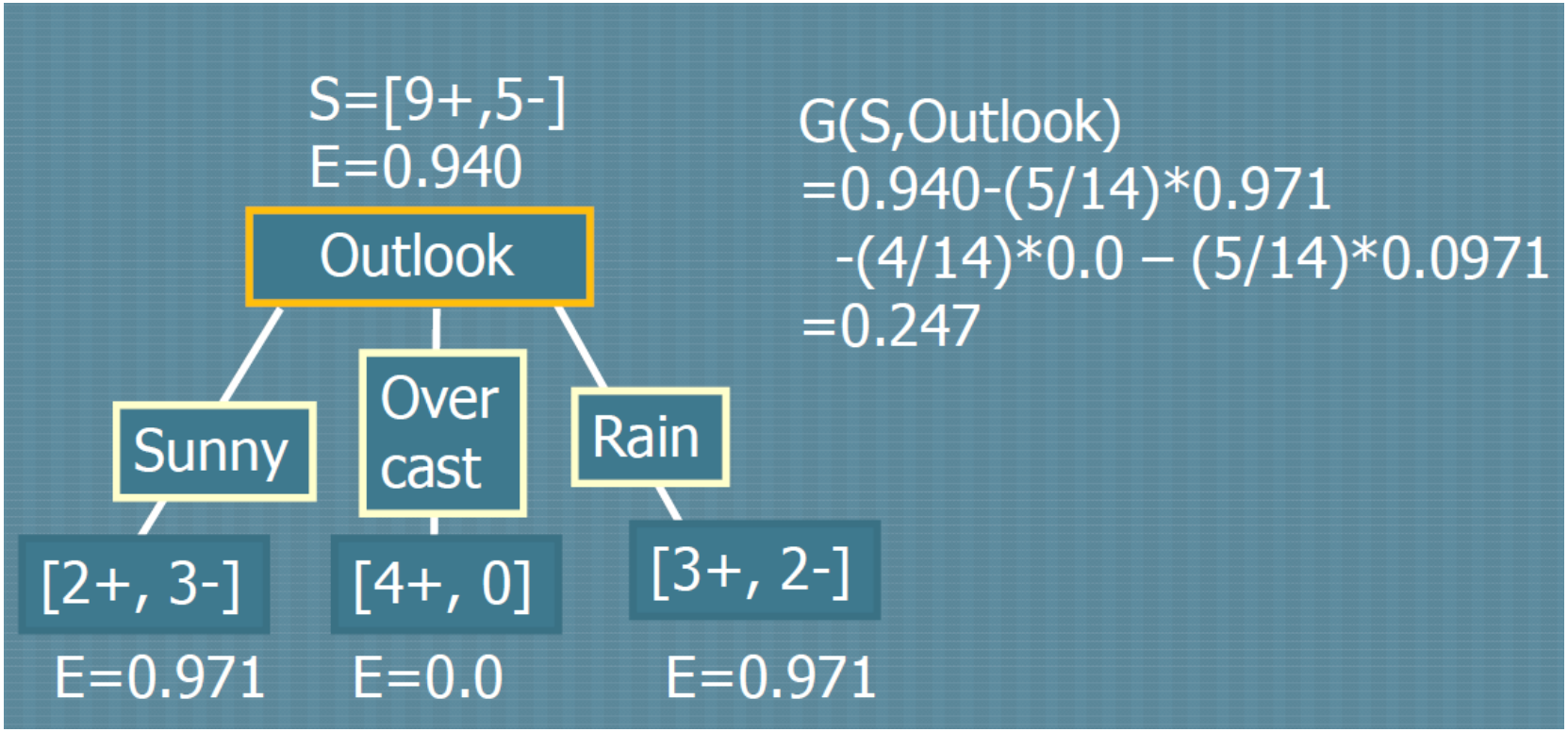
Information Gain

- Information gain (IG) measures how much “information” a variable gives us about the class.
  - Features that perfectly partition should give maximal information.
  - Unrelated features should give no information.
- It measures the reduction in **entropy**.
  - Entropy: **(im)purity** in an arbitrary collection of examples.

Information Gain



Information Gain

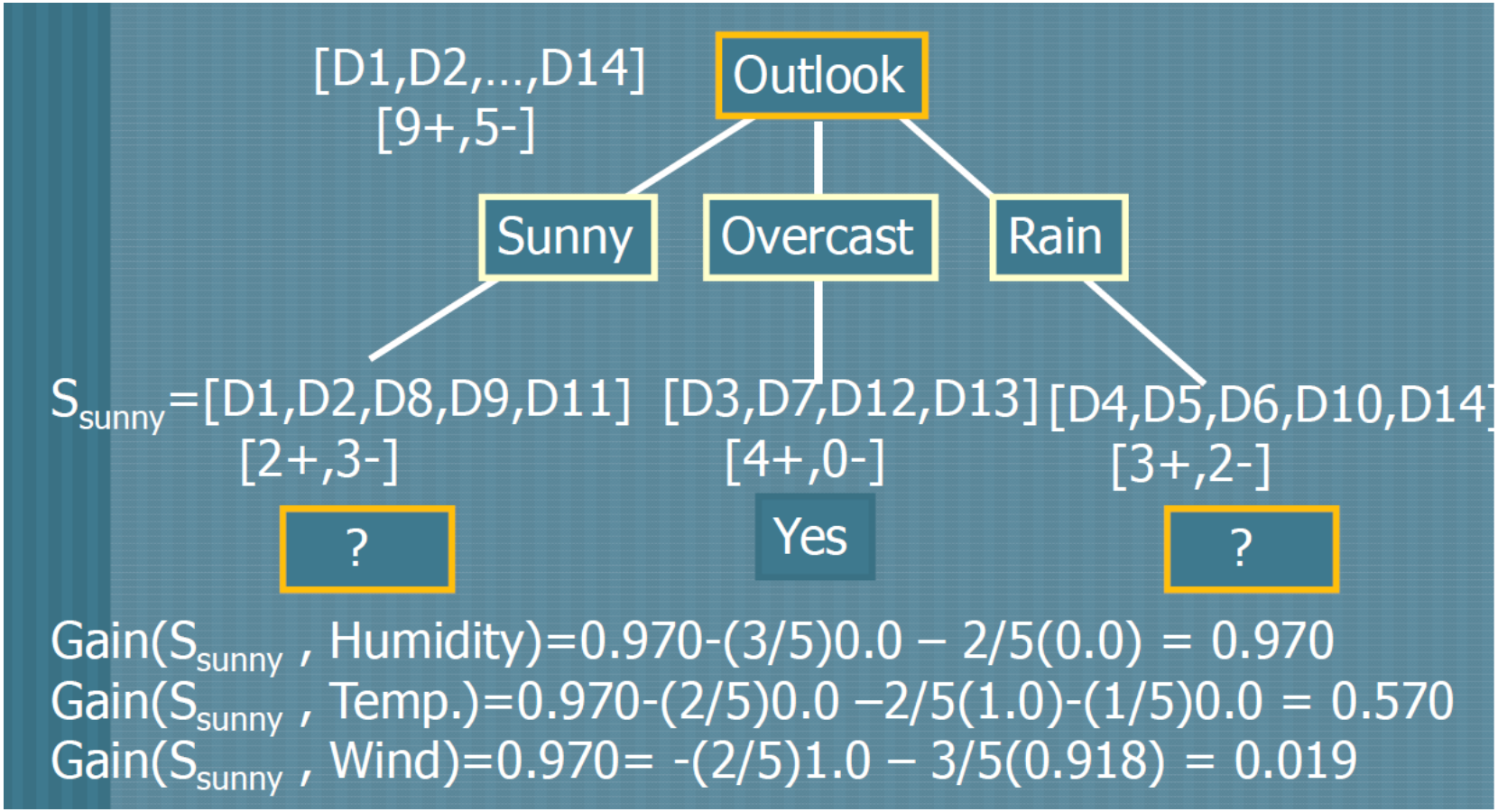


Information Gain

- Information gain for each feature:
  - Outlook=0.247
  - Temperature=0.029
  - Humidity=0.152
  - Windy=0.048

- Initial split is on outlook, because it is the feature with the highest information gain.

Information Gain



ID3 Algorithm

- For each node select the best variable (split criterion)
  - Use all variables that are not the current one or a variable in a parent node
  - This is done in all braches until all the examples in the node are the same class.
- Selecting the best variable:
  - Select the best variable using the entropy and information gain
  - best = attribute with highest IG

Possible Problems

- What if there are not examples for one value when expanding a node?
  - Do not generate branch.
  - When testing, go to the most popular branch
- When we cannot expand the node but still there are + and – examples mixed in the node
  - Label the node with the majority class.
- **Overfitting**

Overfitting Problem

- The previous algorithm can result in a tree with many levels and splits
  - Many of the leafs can contain a reduce number of examples
  - The tree is not general enough and almost there is a rule /path for each example

  -> **Overfitting**

How to avoid overfitting?

- Do not split examples when Information Gain above a threshold (pre-bounding)
- Prune some branches of the tree (post-pruning)
  - post-pruning in the tree
  - post-pruning of rules (C4.5)

C4.5

- **C4.5** developed by Ross Quinlan.
- C4.5 is an extension of Quinlan's earlier ID3 algorithm.
- Same dataset than ID3
  - Categorical Variables
- Split Criterion: Use the Normalized Information Gain (Information Gain Ratio) instead.
- Pruning the tree after creation

Information Gain Ratio

- Information Gain Ratio (IGR) is one solution to Undesired effect: Bias towards variables with large number of different values
  - **Tendency of overfitting**
- **IGR is the normalization of IG by the number of categories.**

$$IGR = \frac{IG}{IV}$$

$$IG = E(X) - \sum_{v \in X} \frac{|S_v|}{|S|} E(S_v)$$

$IV = -\sum_{v \in X} \frac{|S_v|}{|S|} \log_2 \left(\frac{|S_v|}{|S|} \right)$ (Entropy of the splitting variable)

Information Gain Ratio

- **Information gain ratio** is a ratio of information gain to the intrinsic information.
- Proposed by Ross Quinlan, to reduce a bias towards multi-valued attributes by taking into account the number and size of branches when choosing an attribute.
- The GAIN depends of the number of modalities of the splitting variable, thus the results depend largely of the actual coding of variables.
- Remedy. Relativize respect the intrinsic information (IV) of the splitting variable (which depends on its number of modalities):

$$IGR = \frac{IG}{IV}$$

$$IG = E(X) - \sum_{v \in X} \frac{|S_v|}{|S|} E(S_v)$$

$IV = -\sum_{v \in X} \frac{|S_v|}{|S|} \log_2 \left(\frac{|S_v|}{|S|} \right)$ (Entropy of the splitting variable)

C4.5

- **Split Criterion**: Allocation of each node to the response modality with maximum probability
- **Criterion stop**: It stops when the relative gain is below a certain threshold.
- **Post-Pruning** the branches if the probability of misclassification in descending nodes does not improve the probability of misclassification in the current node in a significant way.

Post pruning of the tree

- Split data in training and validation subsets

Training and Validation

- To obtain a reliable tree, we need to test it with independent data from the one used in the learning step.
  - Divide the total data at random in one part for learning (training) and the remaining for testing (validating)
    - Common percentage: 2/3 training and 1/3 validation

Post pruning of the tree

- Split data in training and validation subsets
- Repeat until no improvement in validation subset error:
  1. For each node compute the error on the validation set when removing the node (and so the subtree below the node)
  2. Do the pruning that decrease the most the error on the validation set

Probability of Misclassification

- We need to define how good (or bad) is a tree
  - Obvious criterion = *Probability of Misclassification (= cost of the tree)*
- How can we compute the Cost of the tree:
  - We assign every leave to the response class.
    - If there are examples of different class, we take the class with the maximum number of examples (mode)

Misclassification Rate

- *Confusion Matrix for Characterizing Classification Errors*
  - *Confusion Matrix* = visualization of predicted versus actual outcomes
    - Good if high values along diagonal, low values elsewhere

		Prediction	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

		Prediction				
		Class 1	Class 2	Class 3	...	Class n
Actual	Class 1	Accurate				
	Class 2		Accurate			
	Class 3			Accurate		
	...				Accurate	
	Class n					Accurate

$$ErrorRate = \frac{|Misclassification|}{|Total|} = \frac{FN + FP}{TP + TN + FN + FP}$$

There are other formulas…

Error rate

We apply the model in the test sample, and we classify every individual according a threshold (usually = 0.5) on the outcome.

	Predicted class YES	Predicted class NO	
Real class YES	TP	FN	P
Real class NO	FP	TN	N
	$predP$	$predN$	n

$$Error\ rate = \frac{FN + FP}{n}$$

$$Accuracy = 1 - Error\ rate$$

$$Precision = \frac{TP}{predP} \qquad Recall = Sensitivity = \frac{TP}{P} \qquad Specificity = \frac{TN}{N}$$

F-measure: $F = \frac{2PR}{P+R} = \frac{2}{\frac{1}{R}+\frac{1}{P}}$

It depends on the chosen threshold

with p=0.5

	Predicted class YES	Predicted class NO
Real class YES	*164*	*100*
Real class NO	*56*	*480*

Error rate= 0.195

Precision= 0.745

Recall= 0.621

F = 0.677

Post pruning of the tree

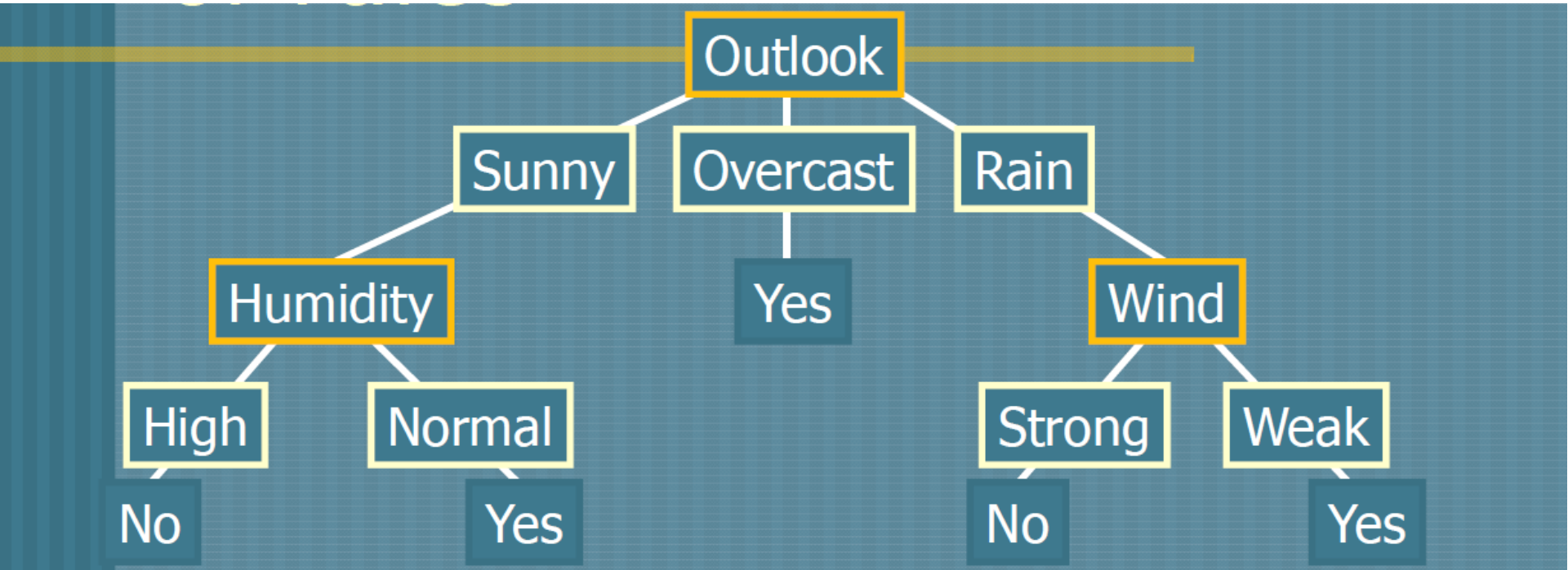
- Split data in training and validation subsets
- Repeat until no improvement in validation subset error:
  1. For each node compute the error on the validation set when removing the node (and so the subtree below the node)
  2. Do the pruning that decrease the most the error on the validation set

Post pruning of rules (C4.5)

- C4.5 Algorithm

1. Build the decision tree with ID3
2. Transform the tree into rules
3. Prune each rule independently by removing conditions that decrease the error on validation set
4. Sort the final set of rules to solve conflicts
5. Remove rules when it decreases error on validation set

Translating a tree into a set of rules



R_1: If (Outlook=Sunny) $\wedge$ (Humidity=High) Then PlayTennis=No
R_2: If (Outlook=Sunny) $\wedge$ (Humidity=Normal) Then PlayTennis=Yes
R_3: If (Outlook=Overcast) Then PlayTennis=Yes
R_4: If (Outlook=Rain) $\wedge$ (Wind=Strong) Then PlayTennis=No
R_5: If (Outlook=Rain) $\wedge$ (Wind=Weak) Then PlayTennis=Yes

Post pruning of rules (C4.5)

- C4.5 Algorithm

1. Build the decision tree with ID3
2. Transform the tree into rules
3. Prune each rule independently by removing conditions that decrease the error on validation set
4. Sort the final set of rules to solve conflicts
5. Remove rules when it decreases error on validation set

CHAID (Kass, 1980)

CHAID (**Chi-square automatic interaction detection**)

Split criterion: based in the Chi-square statistic computed crossing the response variable with the tentative partition defined by an explanatory variable.

The most significant is the Chi-square, more different is the distribution of the response in the parent node respect to the children nodes

response \ partition	n	n1	…	nk	…	nq
n	n	n1	…	nk	…	nq
n1	n1	n11	…	n1k	…	n1q
…	…	…	…	…	…	…
nj	nj	nj1	…	njk	…	njq
…	…	…	…	…	…	…
nm	nm	nm1	…	nmk	…	nmq

$$\chi^2 = \sum_{k=1}^{q} \sum_{j=1}^{m} \frac{\left(n_{jk} - n_k \frac{n_j}{n} \right)^2}{n_k \frac{n_j}{n}}$$

q: Number of child nodes (=2 if binary tree)

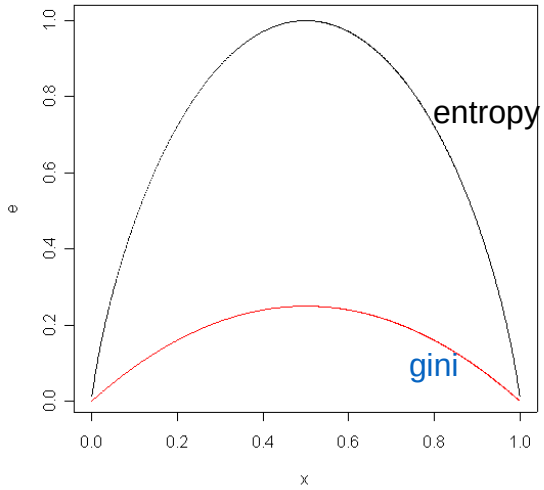
m: number of response classes

The (categorical) variable giving the most significative χ^2 defines the optimal split

CART

- Developed 1974-1984 by 4 statistics professors: Leo Breiman (Berkeley), Jerry Friedman (Stanford), Charles Stone (Berkeley), Richard Olshen (Stanford). Distributed by Salford Systems http://salford-systems.com/
- **Just perform Binary trees**
- Unifies the categorical and continuous (numerical) response under the same framework.
  - Classification tree
  - Regression tree
- Any kind of explanatory variable (numerical and categorical)
- Split criterion: Impurity of the node
- Post pruning (without stop criterion)
- Delivers honest estimates of the quality of a tree

Split Criterion: Impurity of a node



- For categorical responses:
  - Entropy (Information) $E(X) = -\sum_i^n p_i \log(p_i)$
  - Gini $Gini = 1 - \sum_{i=1}^n p_i^2$
- For continuous responses:
  - Variance $V(X) = \frac{\sum_i^n (x_i - \bar{x})^2}{n}$

Gini Index Properties

- Maximized when elements are heterogeneous (impure):
  - $If\ p_i = \frac{1}{i}, then\ Gini = 1 - \sum_{i=1}^{n} \left(\frac{1}{i}\right)^2 = 1 - \frac{1}{n}$
- Minimized when elements are homogenous (pure):
  - $If\ p_i = 1\ or\ p_i = 0, then\ Gini = 1 - 1 - 0 = 0$

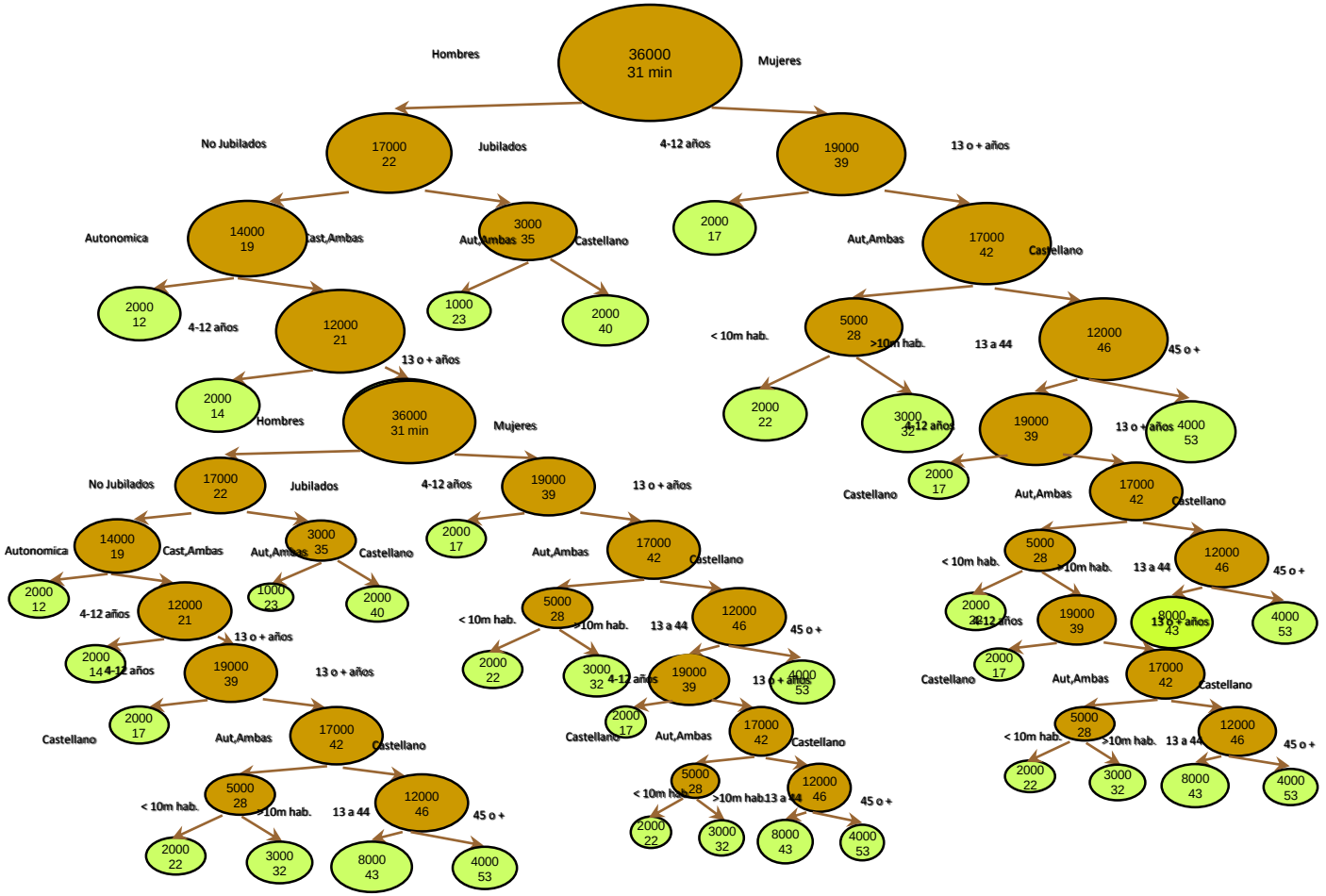
CART

- Greedy algorithm to minimize a cost function.
- (Not original version)Stop criterion:
  - Fix a minimum number of examples per node. If the node has less of this minimum, then the node is a leaf.
    - If minimum = 1 then probably too specific and overfitting.
- Post-pruning: Find nodes that are removable.
  - Simple: The cost of the tree improves when the node is removed.
  - Other: penalization for complexity

Stop criterion

There is no stop criterion. CART uses post-pruning, it builds a maximum tree and prunes the non interesting branches

Absolute maximum tree: Tree with all pure leaves



Misclassification Rate

- *Confusion Matrix for Characterizing Classification Errors*
  - *Confusion Matrix* = visualization of predicted versus actual outcomes
    - Good if high values along diagonal, low values elsewhere

		Prediction	
		Positive	Negative
Actual	Positive	TP	FN
	Negative	FP	TN

		Prediction				
		Class 1	Class 2	Class 3	...	Class n
Actual	Class 1	Accurate				
	Class 2		Accurate			
	Class 3			Accurate		
	...				Accurate	
	Class n					Accurate

$$ErrorRate = \frac{|Misclassification|}{|Total|} = \frac{FN + FP}{TP + TN + FN + FP}$$

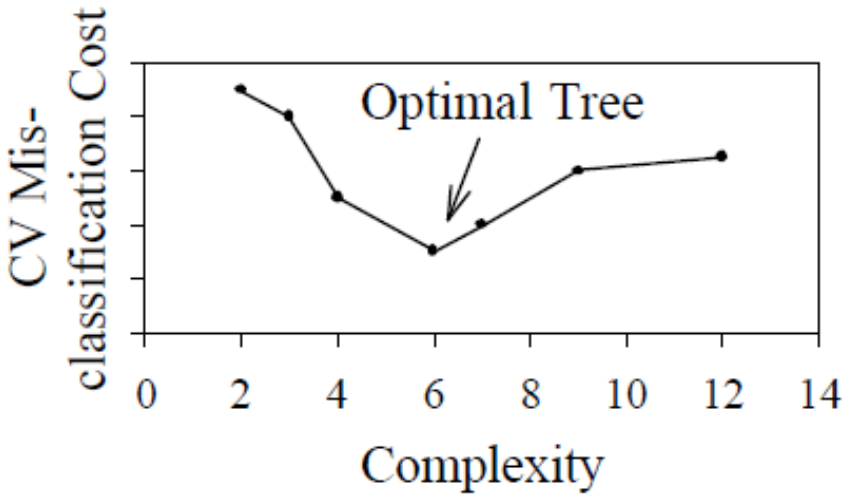
There are other formulas…

Penalization for complexity

Criterion to optimize: $Min\left(R(T) + \alpha \left| T \right| \right)$

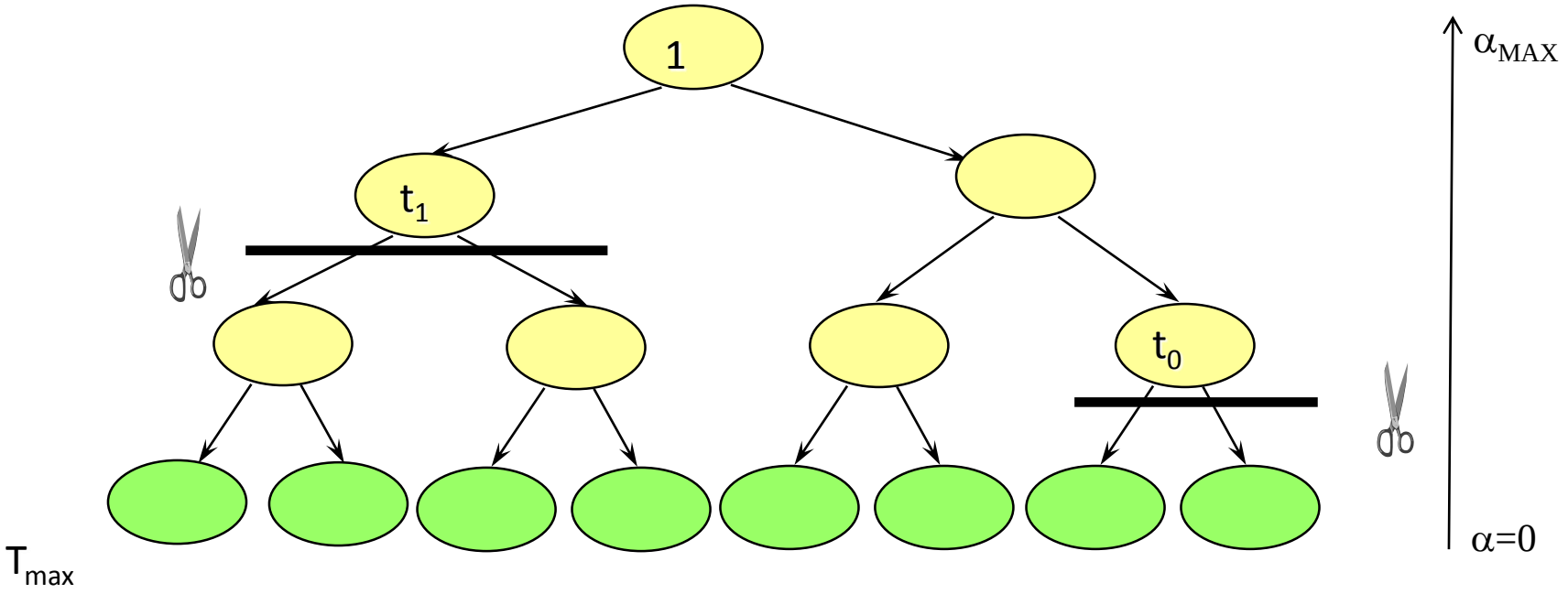
R(T) = Cost of the tree

Where α is the complexity parameter, expresses the penalization for building large trees



We can build the trees for increasing values α, obtaining a sequence of optimal trees (of increasing size) (CART does this pruning from the maximal tree in a more intelligent way)

$$T_{\max}, T_{\max} - T_{t_0}, T_{\max} - T_{t_0} - T_{t_1}, \mathrm{K}, 1$$



Each subtree is optimum (min R(T)), whithin the subtrees of complexity ITI

Decision Tree. Advantages

- **Simple to understand and interpret.**
  - Trees can also be displayed graphically in a way that is easy for non-experts to interpret
- **Able to handle both numerical and categorical data.**
  - Other techniques are usually specialised in analysing datasets that have only one type of variable.
    - Association rules can be used only with categorical variables
    - Neural networks can be used only with numerical variables or categorical converted to 0-1 values.
- **Requires little data preparation.**
  - No data normalization.
  - No need to create dummy variables.
- **Uses a white box model.**
  - If a given situation is observable in a model the explanation for the condition is easily explained by boolean logic.
  - By contrast, in a black box model, the explanation for the results is typically difficult to understand, for example with an artificial neural network.
- **Possible to validate a model using statistical tests.**
  - Possible to account for the reliability of the model.
- Non-statistical approach that makes no assumptions of the training data or prediction residuals; e.g., no distributional, independence, or constant variance assumptions
- **Performs well with large datasets.**
  - Large amounts of data can be analysed using standard computing resources in reasonable time.

Decision Tree. Limitations

- Trees can be very non-robust.
  - A small change in the training data can result in a large change in the tree and consequently the final predictions.
- Learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts.
  - Consequently, practical decision-tree learning algorithms are based on heuristics such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree.
- Decision-tree learners can create over-complex trees that do not generalize well from the training data. (This is known as overfitting)
  - Mechanisms such as pruning are necessary to avoid this problem
- For data including categorical variables with different numbers of levels, information gain in decision trees is biased in favour of attributes with more levels.

References

- Quinlan, J. Ross (1986). Induction of decision trees. *Machine learning*, *1*(1), 81-106.
- Quinlan, J. Ross (1996). Bagging, boosting, and C4. 5. In *AAAI/IAAI, Vol. 1* (pp. 725-730).
- Kass, G. V. (1980). An exploratory technique for investigating large quantities of categorical data. *Applied statistics*, 119-127.
- Breiman L., Friedman J. H., Olshen R. A., and Stone, C. J. (1984) Classification and Regression Trees. Wadsworth

CART - library rpart

To grow a tree, use
rpart(*formula*, **data=, method=,control=)** where

formula	is in the format *outcome ~ predictor1+predictor2+predictor3*+ect.
data=	specifies the data frame
method=	"**class**" for a classification tree "**anova**" for a regression tree
control=	optional parameters for controlling tree growth. For example, control=rpart.control(minsplit=30, cp=0.001) requires that the minimum number of observations in a node be 30 before attempting a split and that a split must decrease the overall lack of fit by a factor of 0.001 (cost complexity factor) before being attempted.

Random Forest

- Library: randomForest
- https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm
- Performs both regression and classification tasks.
- It also undertakes dimensional reduction methods, treats missing values, outlier values and other essential steps of data exploration.
- Improve predictive accuracy by generating a large number of bootstrapped trees (based on random samples of variables),
  - Classifying a case using each tree in this new "forest", and
  - Deciding a final predicted outcome by combining the results across all of the trees
    - an average in regression,
    - a majority vote in classification

Random Forest Breiman, 2001

Let *n* be the number of training cases, and the number of explanatory variables *M*.

Let *m* (*m*<<*M*) the number of variables to be used in a node (defaults: *m=sqrt(M)* or *m=M/3*).

Choose a bootstrap sample from the training data set. Use the rest of the cases as validation sample (out of bag OOB cases, on average 0.368).

Build a tree

In each node of the tree, randomly choose *m* variables to derive the best split.
The tree is fully grown (not pruned).
Use the OOB (out-of-bag) cases to compute the error rate.

The final error rate is the mean of the OOB errors rates.
For prediction a new sample is pushed down the tree. It is assigned the label of the training sample in the terminal node it ends up in. This procedure is iterated over all trees in the ensemble, and the average vote of all trees is reported as random forest prediction.

randomForest(formula, data=NULL, ..., subset, na.action=na.fail, ntree=50)