INM431: Machine Learning

Decision trees

Pranava Madhyastha (pranava.madhyastha@city.ac.uk)

Based on materials by Artur S. d'Avila Garcez + Bishop's book + Murphy's book

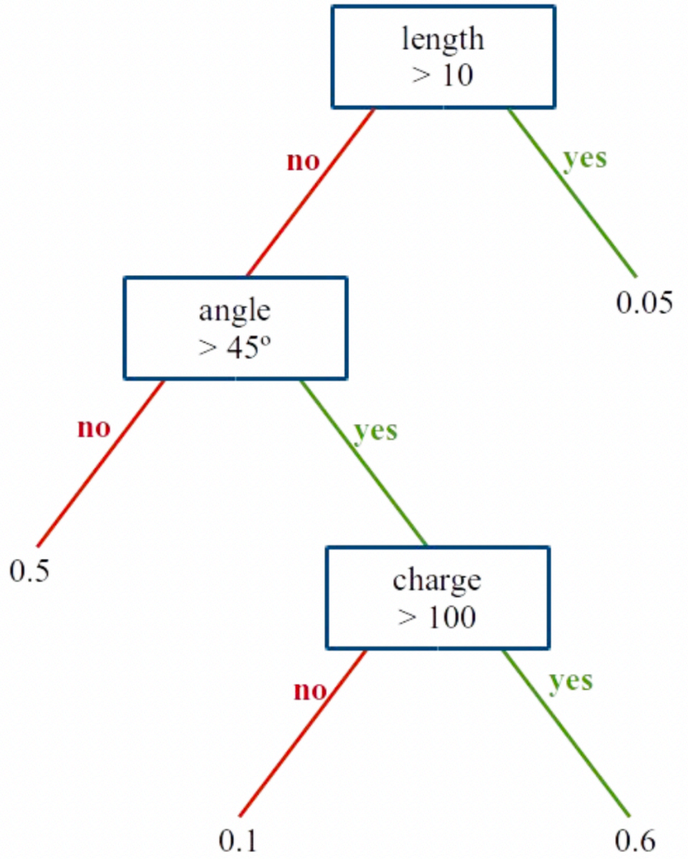
Decision Trees $f : X \rightarrow Y$

Each internal node tests an attribute x_i

One branch for each possible attribute value $x_i = v$

Each leaf assigns a class y

To classify input x: traverse the tree from root to leaf, output the labeled y



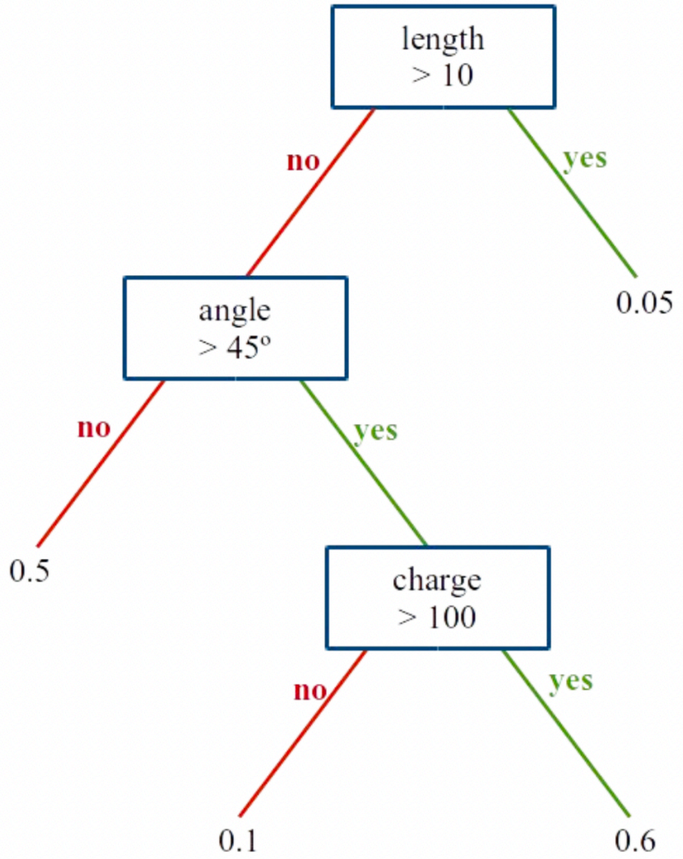
A popular algorithm of choice

One of the very popular algorithms

Naive decision trees are human interpretable

Boosting & bagging make them very powerful

One of the best performing algorithm for a variety of classification problems



Setup

Input: set of properties describing object/situation

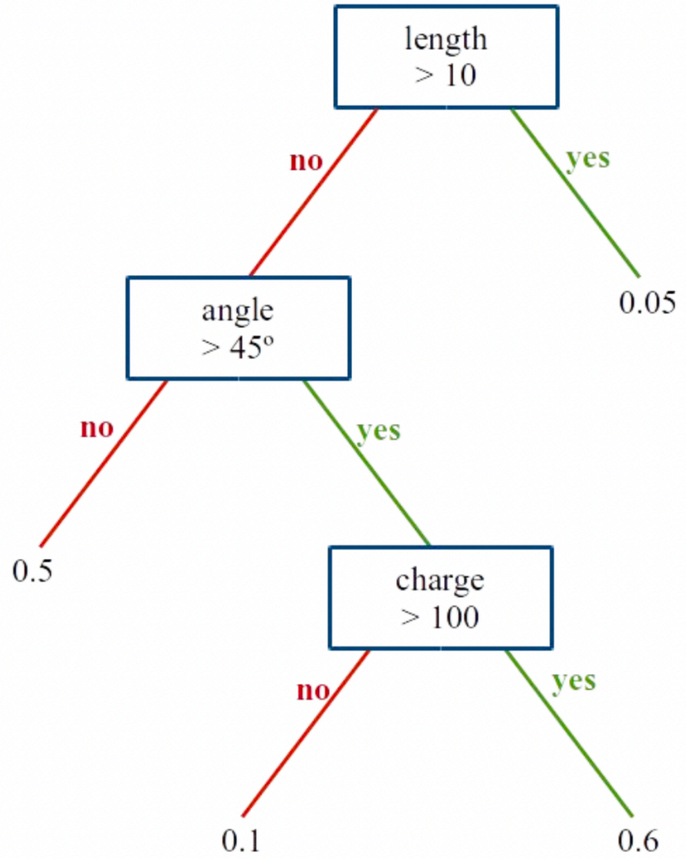
Output: $\in S$, where S is a finite set of classes

Non-terminal nodes: test on values of property

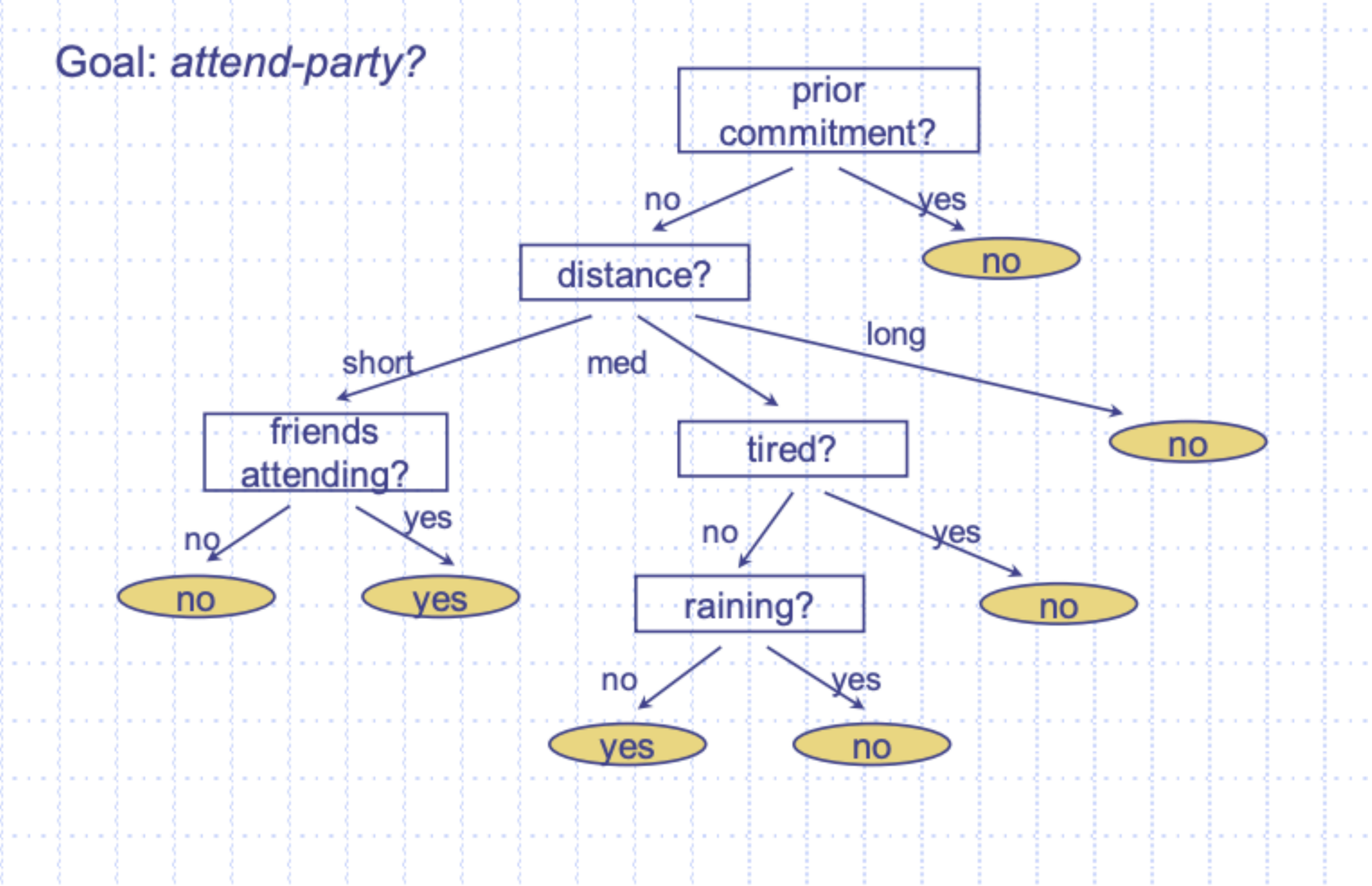
Terminal nodes: Boolean value of the goal class

Branches: values of test

Properties (length, angle, charge): attributes/features



Example


Goal: attend-party?
prior commitment?
no
yes
no
distance?
short
med
long
friends attending?
tired?
no
no
yes
no
yes
no
yes
raining?
no
yes
no
yes
no

Basic principles

Choice of model is function of input variables

–Different models become responsible for making predictions in different regions of input space

–A sequence of binary selections corresponding to traversal of a tree structure

If features are continuous, internal nodes can test the value of a feature against a threshold

Representational power

Decision trees can represent any function of the input attributes!

Could require exponentially many nodes

Some equivalence to propositional logic

Decision trees implicitly define conjunctions of disjunctions

Learning simplest decision tree is NP-hard

Greedy heuristics:

–Start from empty decision tree

–Split on next best attribute (feature)

Learning algorithm

1) Start with a set of examples (training set), set of attributes $\{A\}$, default value for goal.

2) If the set of examples == empty,
    then add a leaf with the default value for the goal and terminate,
    otherwise:
3) If all examples have the same classification,
        then add a leaf with that classification and terminate,
        otherwise:
4) If the set of attributes SA is empty,
        then return the default value for the goal and terminate,
        otherwise:
5) Choose an attribute A to split on.
6) Add a corresponding test to the tree.
7) Create new branches for each value of the attribute.
8) Assign each example to the appropriate branch.
9) Recurse from step 1) on each branch, with set of attributes $\{A\}$and default value the majority value for the current set of examples.

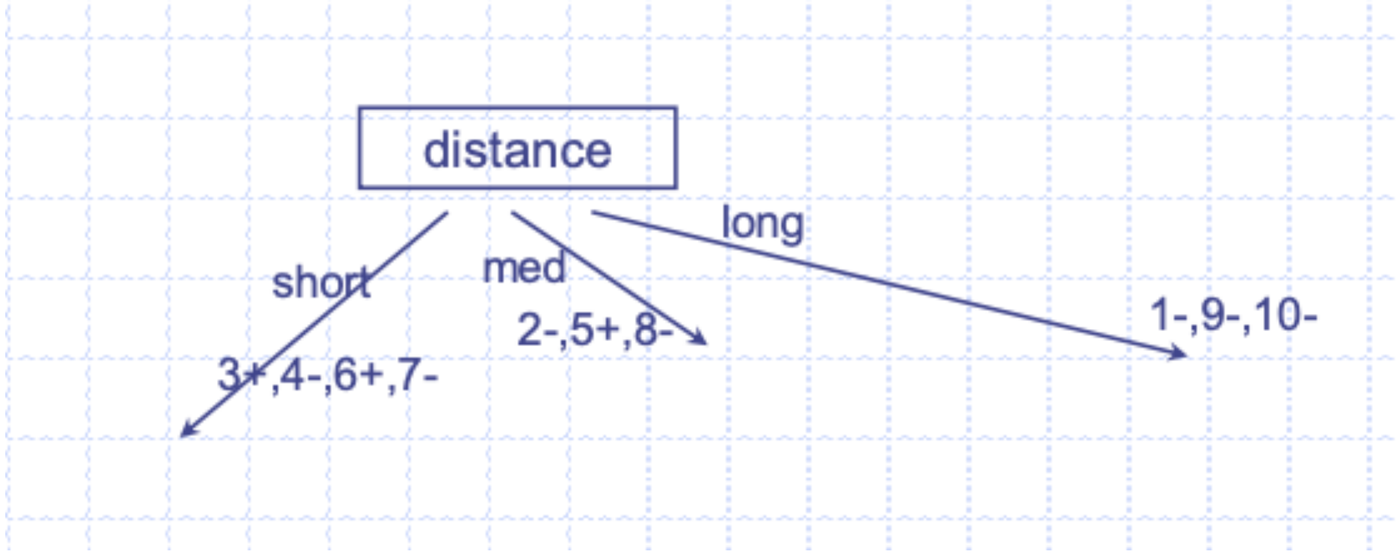
Example: training set

Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Use step 5, split on a random attribute

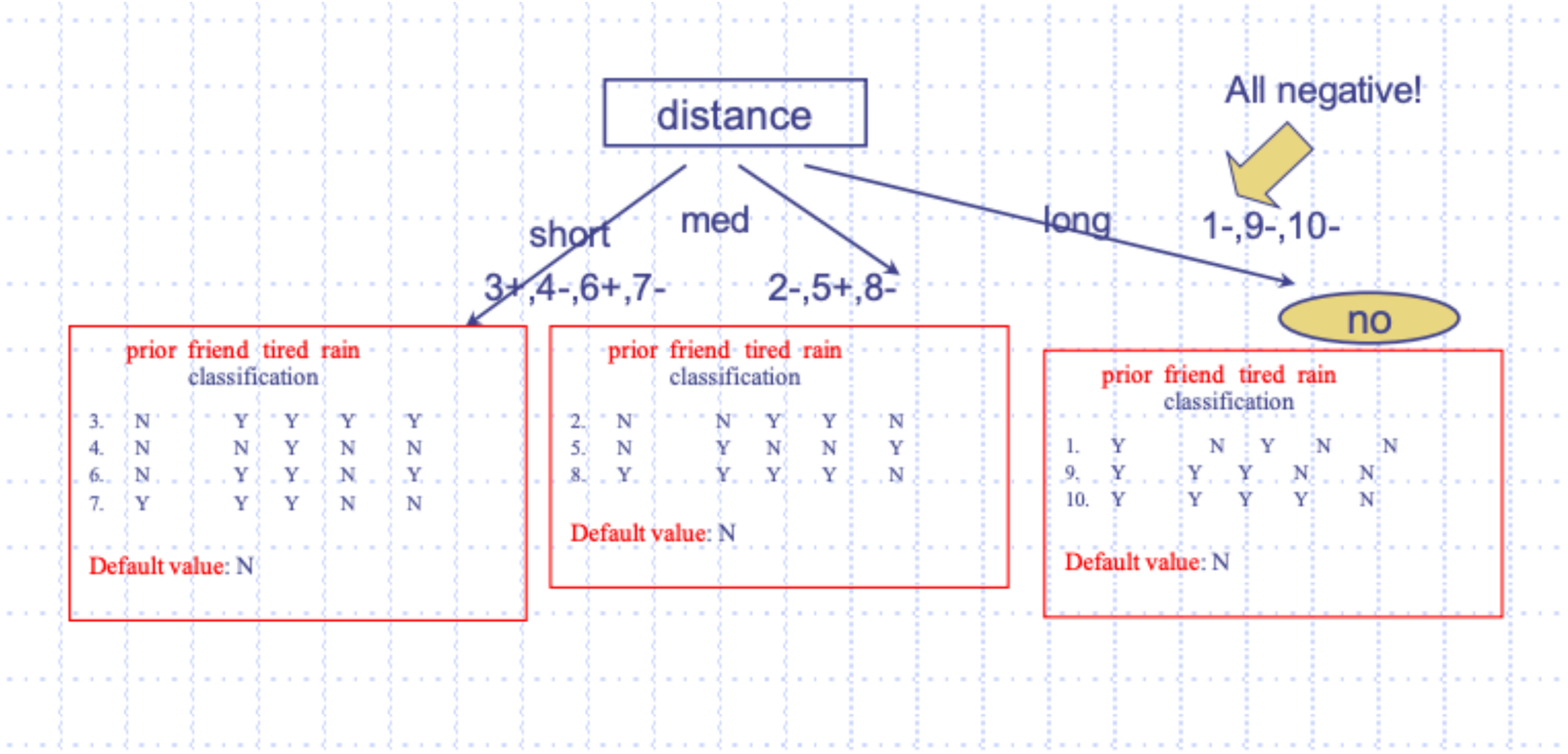


Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Followed by

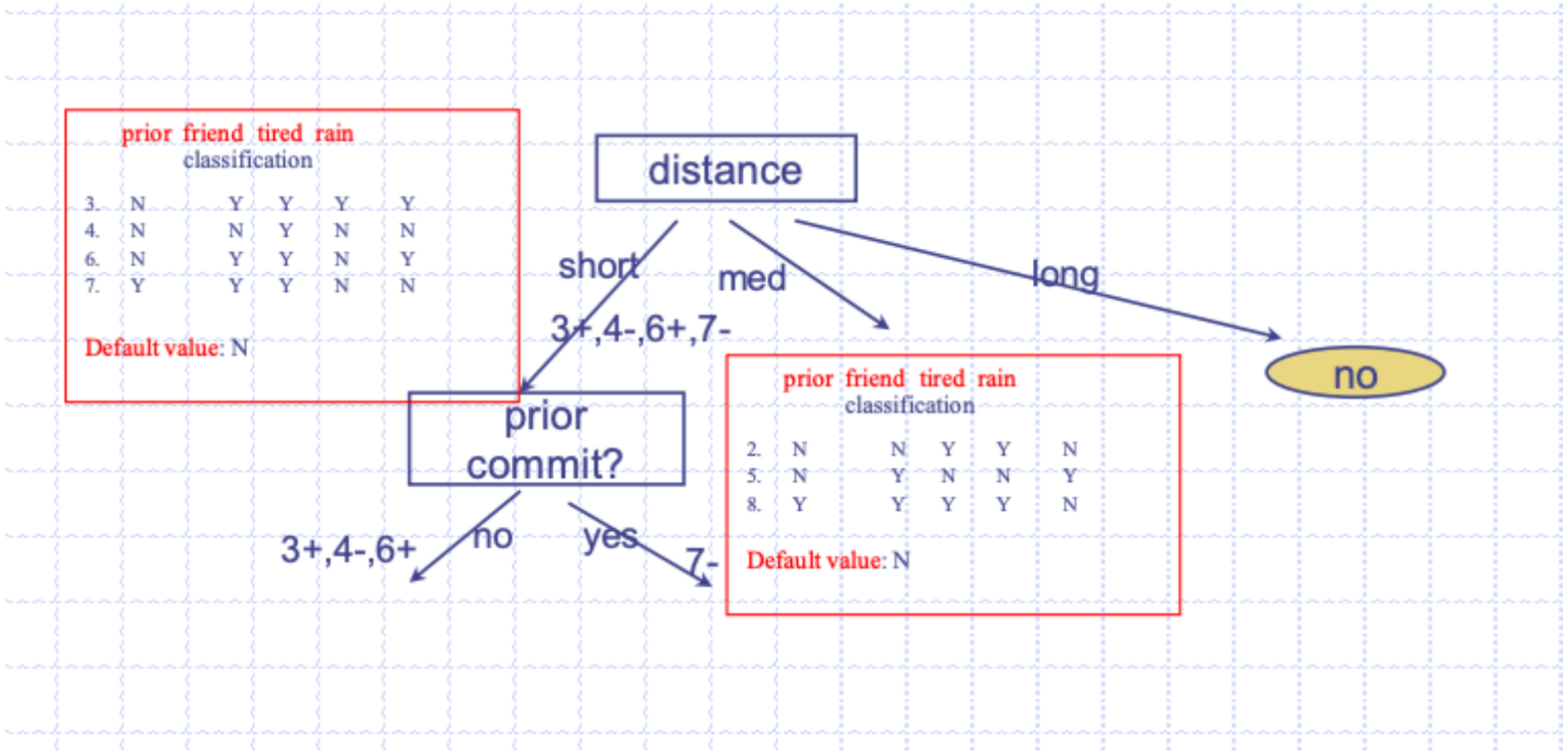


Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Followed by

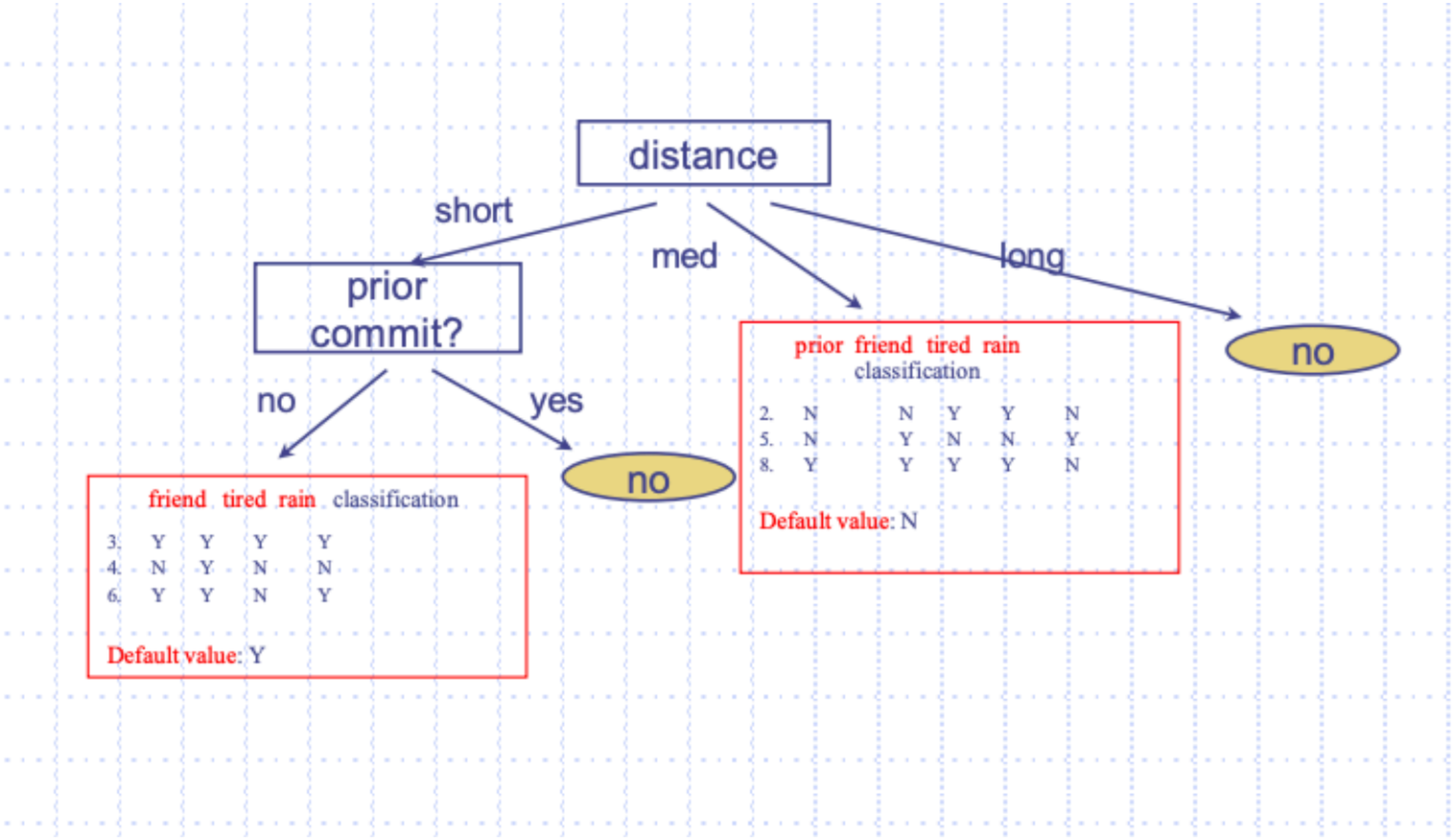


Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Followed by

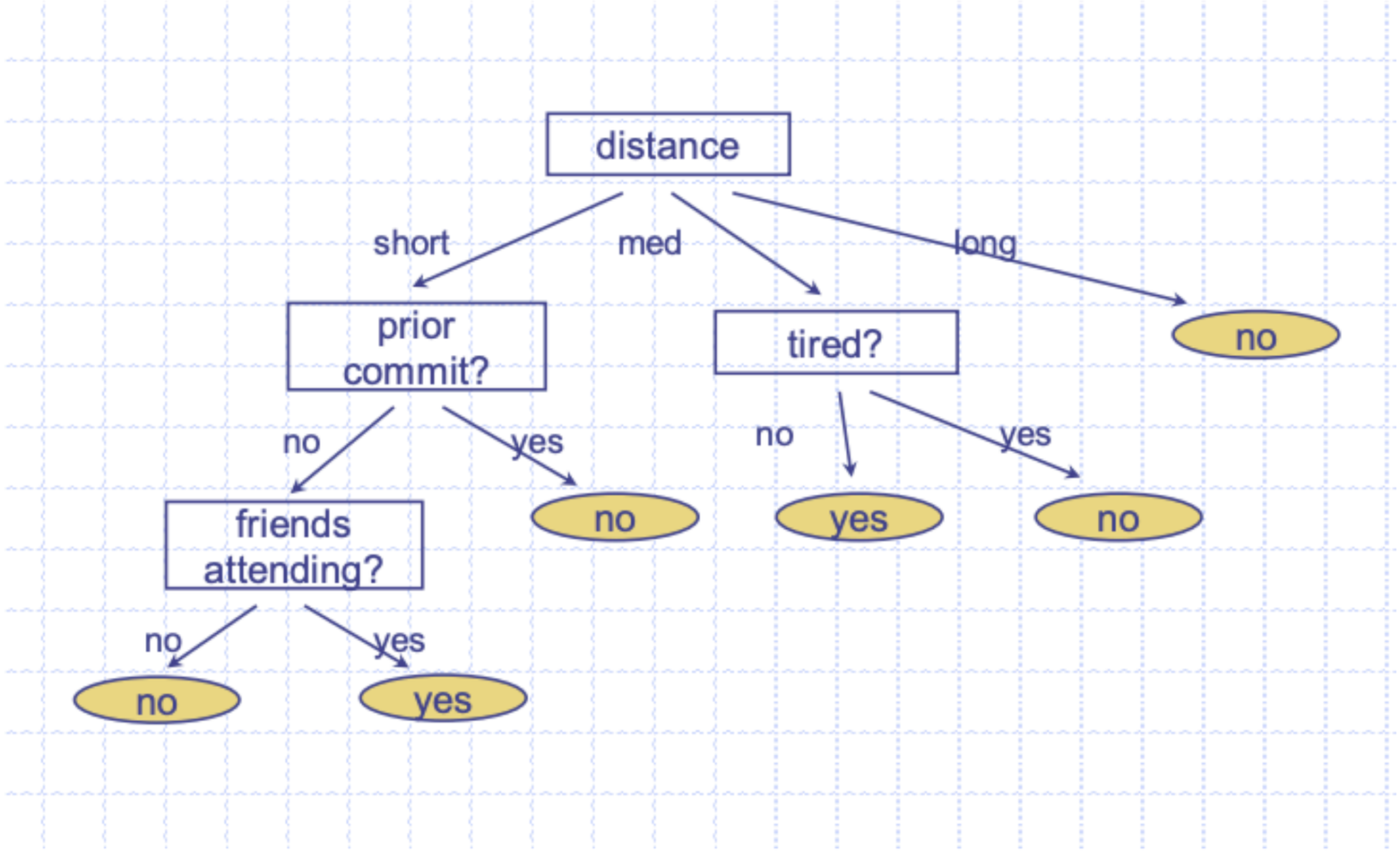


Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Finally



Set of attributes SA

	prior	dist	friend	tired	rain	classification
1.	Y	L	N	Y	N	N
2.	N	M	N	Y	Y	N
3.	N	S	Y	Y	Y	Y
4.	N	S	N	Y	N	N
5.	N	M	Y	N	N	Y
6.	N	S	Y	Y	N	Y
7.	Y	S	Y	Y	N	N
8.	Y	M	Y	Y	Y	N
9.	Y	L	Y	Y	N	N
10.	Y	L	Y	Y	Y	N

Default value: Y

Choosing the “best” attribute

Intuition:

– The aim is to minimise the depth of the final tree

– Choose attribute that provides as exact as possible a classification:

perfect attribute: all examples are either positive or negative

useless attribute: the proportion of positive and negative examples in the new set is roughly the same as in the old set

Use information theory for defining perfect/useful/useless by computing the information gain from choosing attributes

The good and the bad

A binary-valued attribute is good if it helps us discriminate the instances

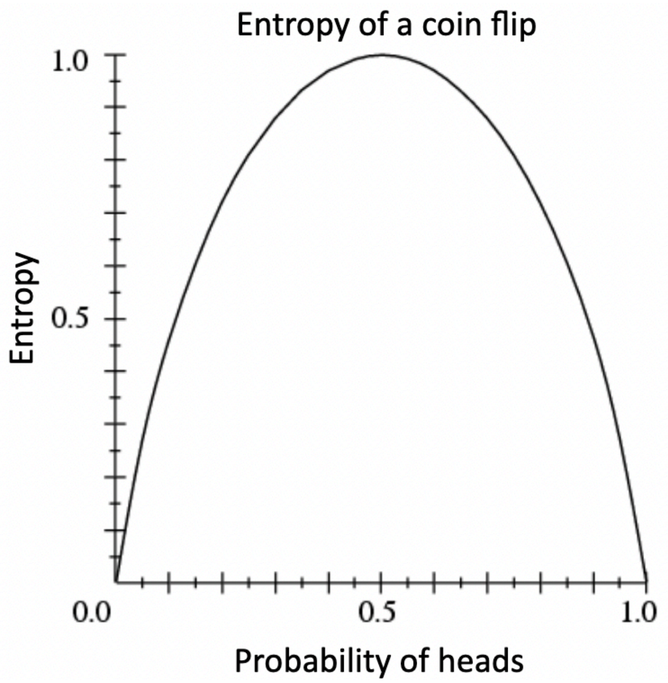
The attributes are poor if it provides no discrimination i.e., attribute is not useful for the decision

Entropy

Entropy $\mathscr{H}(X)$ of a random variable X

$$\mathscr{H}(X) = -\sum_{i=1}^{k} P(X = x_i)\log_2 P(X = x_i)$$

Information Theory interpretation: $\mathscr{H}(X)$ is the expected number of bits needed to encode a randomly drawn value of X (under most efficient code)



Interpretation of entropy

High indicates:

- X is sampled from a uniform distribution
- Prediction is difficult

Low indicates:

- X varies, there are peaks and valleys
- Values are more predictable

Toy example: at least one heads

Setting: 5 trials, 2 coins, 2 types of events

$$\mathscr{H}(X) = -\sum_{i=1}^{k} P(X = x_i)\log_2 P(X = x_i)$$

T_1	T_2	X
T	T	T
T	F	T
F	T	T
T	T	T
F	F	F

$$P(X = \text{True}) = \frac{4}{5};\ P(X = \text{False}) = \frac{1}{5}$$

$$\mathscr{H}(X) = -\frac{4}{5}\log_2(\frac{4}{5}) - \frac{1}{5}\log_2(\frac{1}{5}) = ?$$

Conditional entropy

Conditional Entropy $\mathcal{H}(Y|X)$: Entropy of a random variable Y conditioned on a random variable X

$$\mathcal{H}(Y|X) = -\sum_{i=1}^{k} P(X = x_i) \sum_{j=1}^{v} P(Y = y_j | X = x_i) \log_2 P(Y = y_j | x = x_i)$$

Here:

$P(X_1 = \text{True}) = \frac{4}{6}$; $P(X_1 = \text{False}) = \frac{2}{6}$

$P(Y = \text{True} | X_1 = \text{True}) = ?$; $P(Y = \text{False} | X_1 = \text{True}) = ?$

$P(Y = \text{True} | X_1 = \text{False}) = ?$; $P(Y = \text{False} | X_1 = \text{False}) = ?$

$$\mathcal{H}(Y|X) = -\frac{4}{6}(1 \log_2 1 + 0 \log_2 0) - \frac{2}{6}(\frac{1}{2} \log_2 \frac{1}{2} + \frac{1}{2} \log_2 \frac{1}{2}) = ?$$

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

Information gain

This is defined as the decrease in entropy after ‘splitting’

$$IG(X) = \mathscr{H}(Y) - \mathscr{H}(Y \mid X)$$

If $IG(\text{attribute}) > 0$, prefer that attribute

Going back to our examples: solve for IG.

Choosing the “best” attribute

Intuition:

– The aim is to minimise the depth of the final tree

– Choose attribute that provides as exact as possible a classification:

perfect attribute: all examples are either positive or negative

useless attribute: the proportion of positive and negative examples in the new set is roughly the same as in the old set

Use information theory for defining perfect/useful/useless by computing the information gain from choosing attributes

Some issues

Training set error is always zero (with no label noise)

Decision trees will overfit!

Strategise for simpler trees

Simple = fixed depth, small number of samples per leaf

Further limitations

Multiple trees can exist for the same function

Some problems may only work with exponentially large trees

While it is interpretable, it may be too difficult to work with

Boosting (GBM) and bagging (random forests)

Exercise

Weather	Parents Visiting	Money	Decision
Sunny	Yes	Rich	Cinema
Sunny	No	Rich	Tennis
Windy	Yes	Rich	Cinema
Rainy	Yes	Poor	Cinema
Rainy	No	Rich	Stay in
Rainy	Yes	Poor	Cinema
Windy	No	Poor	Cinema
Windy	No	Rich	Shopping
Windy	Yes	Rich	Cinema
Sunny	No	Rich	Tennis