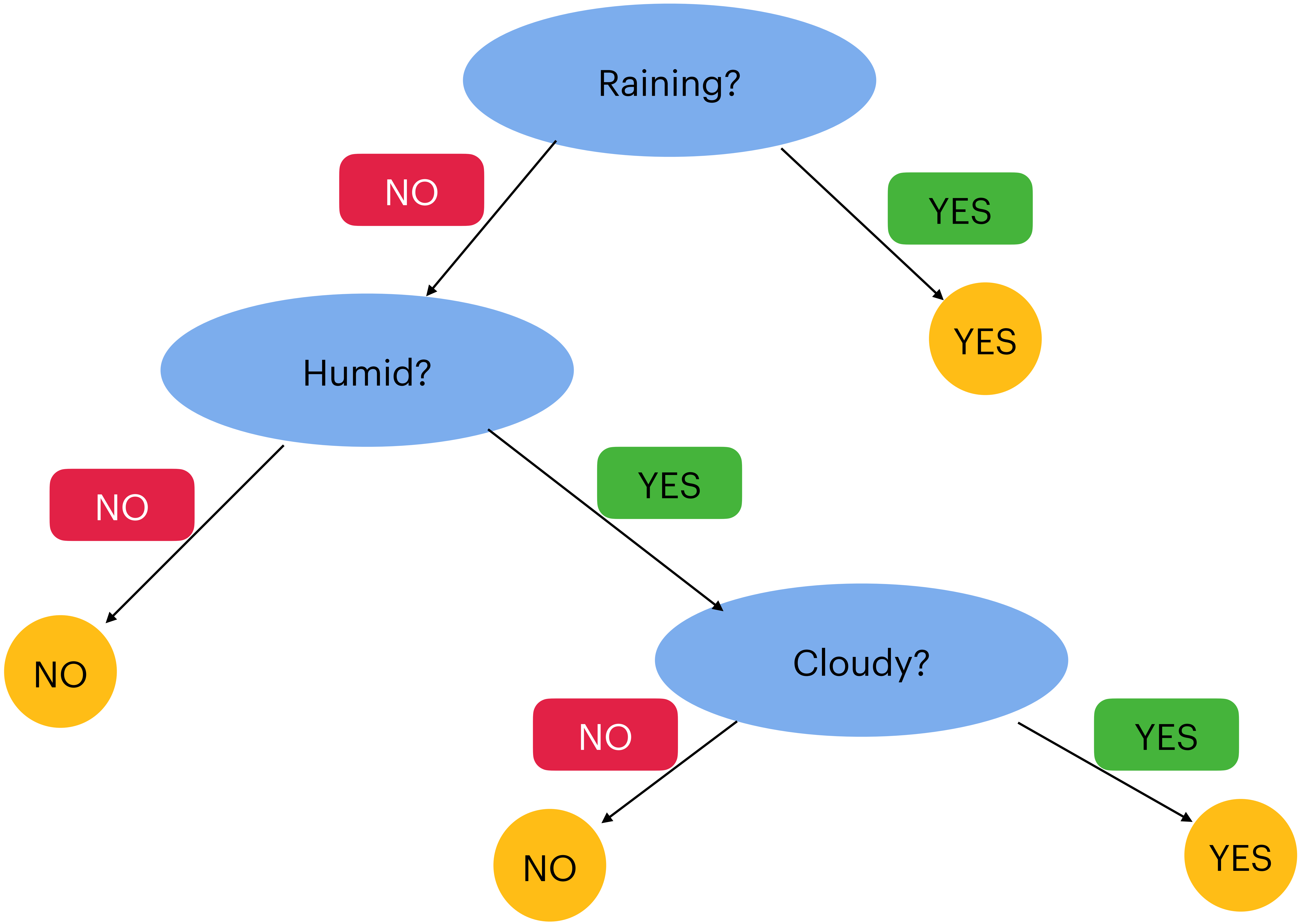
Decision Trees

Mar 18

Should I bring an Umbrella



From MacCormack, pg 90

Flavors

- ID3 & c4.5 & C5.0 -- J.R. Quinlan
- Classification and Regression Trees -- L Breiman, Freeman, Olshen, Stone
- etc

Why Decision Trees

- Simple to understand and interpret.
- Able to handle both numerical and categorical data.
- Requires little data preparation.
- Explainable
- Statistical validation (how reliable is it)
- Performs well with large datasets. (standard computing resources in reasonable time)
- Mirrors human decision making more closely than other approaches.
- Boostable!!
- In built feature selection. Additional irrelevant feature will be less used so that they can be removed on subsequent runs.

Building a Decision Tree

- Given a set of examples
    - Each example consists of a set of features
    - There is a special feature -- the decision variable (or class) -- that is the decision you are trying to make
        - Any of the features could be the decision variable
            - but usually there is some distinguished item
- So to build a decision tree
    - Examine set and find the "most informative" feature about the decision variable (that is not the decision variable itself)
        - Split the set according the most informative feature
        - With each subset return to the "Examine ..." step

Most Informative?????

It depends

- ID3 (Quinlan, 1986) C4.5 (Quinlan, 1993) both use "entropy"
  - Entropy is a measure of the is a measure of the amount of uncertainty in the (data) set
    - Originally suggested by Shannon (1948) as absolute mathematical limit on how data from the source can be losslessly compressed onto a perfectly noiseless channel

$$\mathrm{H}(S) = \sum_{x \in X} -p(x) \log_2 p(x)$$

-
- Where,
  - S – The current dataset for which entropy is being calculated
  - X – The set of classes (the values of the decision variable)
  - p(x) – The proportion of the number of elements in class to the number of elements in set
    - NOTE: When H(S)==0, the set is perfectly classified (i.e. all elements in are of the same class).
- Think of Entropy as a measure of how hard a problem is -- the bigger the number the harder the problem

A Sample data set

Play Ball?

ATTRIBUTE	POSSIBLE VALUES
outlook	sunny, overcast, rain
temperature	continuous
humidity	continuous
windy	true, false

Outlook	temperature	humidity	windy	decision
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

Calculating the information in the system

- I(P) = -(p1*log(p1) + p2*log(p2) + .. + pn*log(pn))
  - pn is the fraction of the items in the data set that have value n of the decision variable
  - The information required to "solve" the problem is the sum of the entropy of the "states"
- In the "play" example, we have 2 states for the decision variable -- "do" and "don't"
  - do 9 times
  - don't 5 times
  - so I(P) = -((9/14)*log(9/14) + (5/14)log(5/14))
  - = -(0.64*(-0.63) + 0.35*(-1.48))
  - = -(-0.409 + -0.530)
  - = 0.939

Information Gain

the reduction in the entropy of the system as a result of partitioning

```
Gain(X,T) = Info(X) - Info(X,T)


                                         |Ti|
Info(X,T) = Sum for i from 1 to n of  ---- * Info(Ti)
                                         |T|
```

- the information needed to identify the class of an element of T is the weighted average of the information needed to identify the class of an element of Ti, i.e. the weighted average of Info(Ti)
  - We can ask this of each
  - then pick the feature that makes the largest reduction to the info of the system

Calculating Info(X,T)

for the outlook feature

- 3 states:
  - sunny: 5 occurrences (3 do, 2 dont)
  - overcast: 4 occurrences (4 do, 0 dont)
  - rain: 5 occurrences (2 do, 3 dont)
- I(X,T) = 5/14*I(3/5, 2/5) + 4/14*I(4/4, 0/4) + 5/14*I(2/5, 3/5)
- = 2*(0.357*((3/5)*log(3/5) + (2/5)log(2/5))) + 0.285*(4/4*log(4/4)+0/4)
- = 2*(0.357*(0.6*-0.73 + 0.4*-1.32)) + 0.285*(0)
- =2*0.357*(0.528+0.442)
- =0.694

Information Gain

- Information Gain = I(X)-I(X,T)
- So for outlook
  - IG(Outlook) = I(X) - I(X, Outlook)
  - = 0.939 - 0.694
  - = 0.246

- IG(Windy) = 0.048

Handling continuous attributes

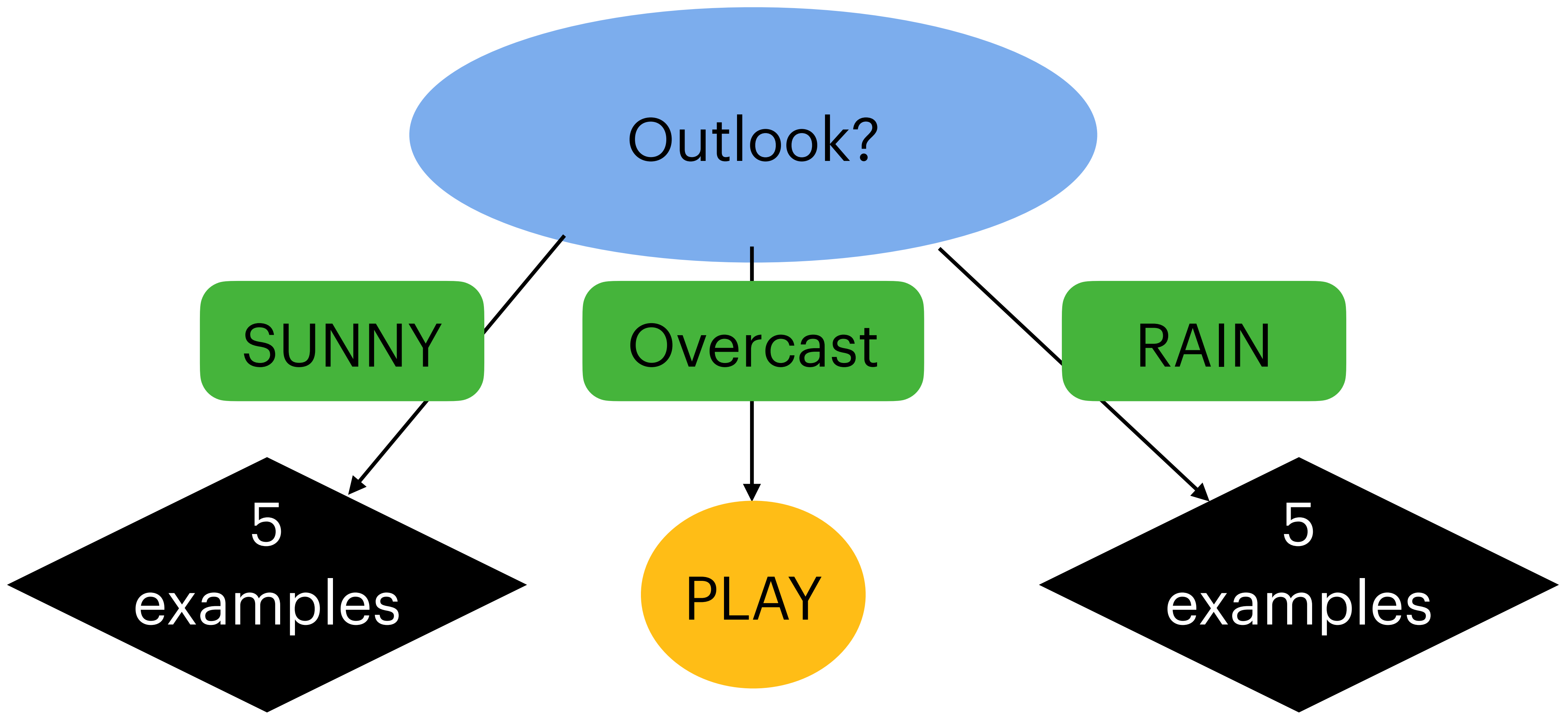
for instance, temperature

- Sort all values
- Create a T/F for each interval
- Compute IG for each interval

- For Temperature
  - 64, 65, 68, 69, 70, 71, 72, 75, 80, 81, 83, 85
  - so effectively make 11 boolean features for the top level decision

Having Identified the "best feature"

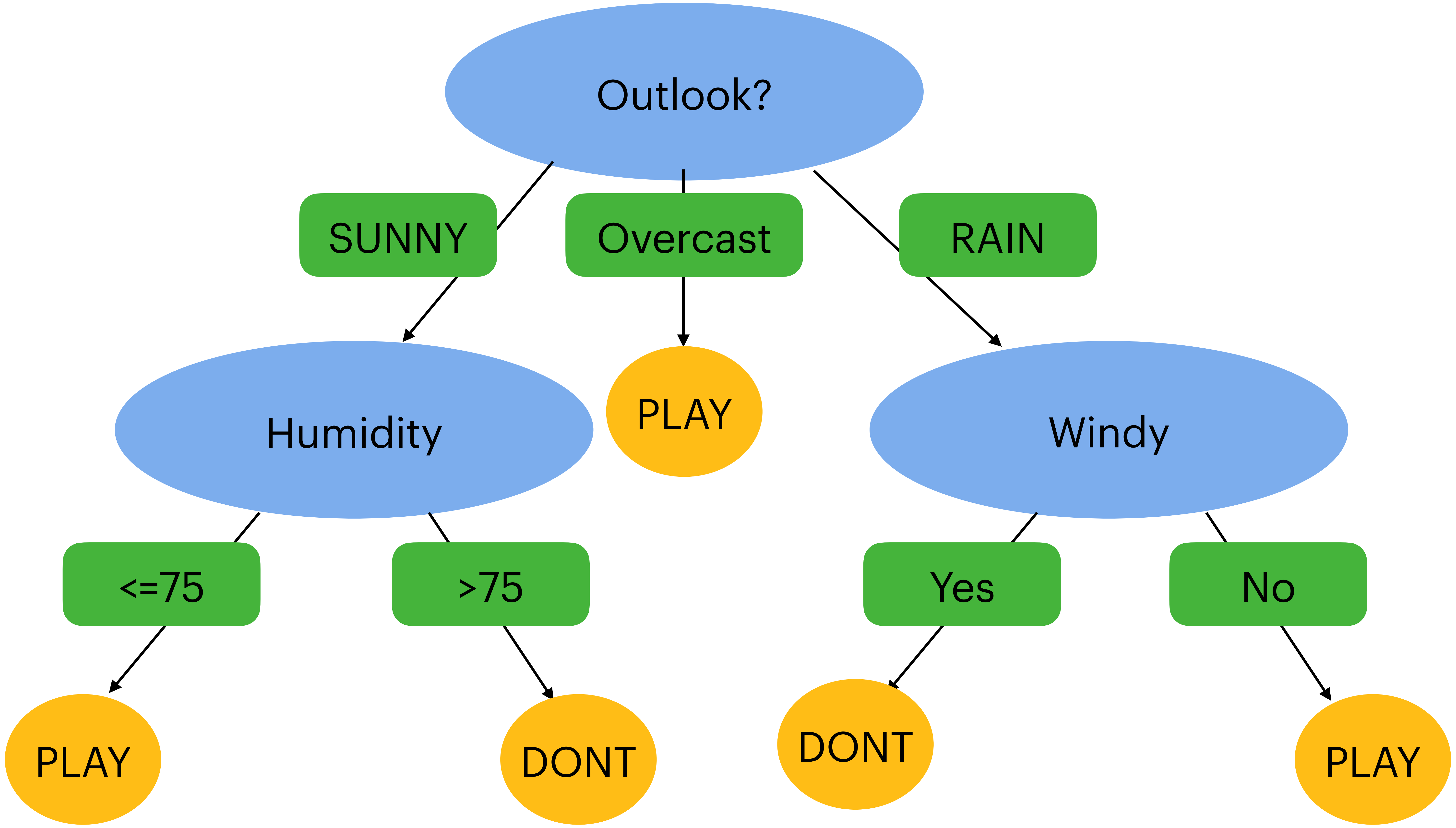
- Create a tree node and add it to the decision tree in the appropriate place
- Split the data
- Recur with each subset of the data



Outlook	temperature	humidity	windy	decision
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
sunny	75	70	true	Play

Outlook	temperature	humidity	windy	decision
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
rain	75	80	false	Play
rain	71	80	true	Don't Play

Final Decision Tree



What kind of algorithm is ID3?

Full Algorithm

```
function ID3 (R: the features,
              C: the decision feature,
              S: a training set) returns a decision tree;
begin
  If S is empty, return a single node with value Failure;
  If S consists of records all with the same value for
     the decision feature,
     return a single node with that value;
  If R is empty, then return a single node with as value
     the most frequent of the values of the decision feature
     that are found in records of S;
  Let D be the attribute with largest Gain(D,S)
     among attributes in R;
  Let {dj| j=1,2, .., m} be the values of attribute D;
  Let {Sj| j=1,2, .., m} be the subsets of S consisting
     respectively of records with value dj for attribute D;
  Return a tree with root labeled D and arcs labeled
     d1, d2, .., dm going respectively to the trees

       ID3(R-{D}, C, S1), ID3(R-{D}, C, S2), .., ID3(R-{D}, C, Sm);

end ID3;
```

Recursive base cases

Why?

There will be errors in training set!

You do not really have to remove D from R! Why?

Another Dataset

Decision Feature: Profit

ATTRIBUTE	POSSIBLE VALUES
age	old, midlife, new
competition	no, yes
type	software, hardware

AGE	COMPETITION	TYPE	PROFIT
old	yes	swr	down
old	no	swr	down
old	no	hwr	down
mid	yes	swr	down
mid	yes	hwr	down
mid	no	hwr	up
mid	no	swr	up
new	yes	swr	up
new	no	hwr	up
new	no	swr	up

Table 1

<table>
<tr><th></th><th>Age</th><th></th><th></th><th></th><th></th><th></th><th></th><th></th></tr>
<tr><td></td><td></td><td>/</td><td></td><td>|</td><td>\</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td>/</td><td></td><td>|</td><td>\</td><td></td><td></td><td></td></tr>
<tr><td></td><td>new/</td><td></td><td></td><td>|mid</td><td></td><td>\old</td><td></td><td></td></tr>
<tr><td></td><td>/</td><td></td><td></td><td>|</td><td></td><td>\</td><td></td><td></td></tr>
<tr><td></td><td>Up</td><td>Co</td><td>m</td><td>petit</td><td>io</td><td>n Down</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>/</td><td></td><td>\</td></tr>
<tr><td></td><td></td><td>/</td><td></td><td></td><td>\</td><td></td><td></td><td></td></tr>
<tr><td></td><td>no/</td><td></td><td></td><td></td><td></td><td>\yes</td><td></td><td></td></tr>
<tr><td></td><td>/</td><td></td><td></td><td></td><td></td><td>\</td><td></td><td></td></tr>
<tr><td></td><td>Up</td><td></td><td></td><td></td><td></td><td>Down</td><td></td><td></td></tr>
</table>

References

ID3: https://hunch.net/~coms-4771/quinlan.pdf

CART: https://books.google.com/books?hl=en&lr=&id=b3ujBQAAQBAJ&oi=fnd&pg=PP1&ots=sS2mWKCrF6&sig=-xOa9DAqbQkjxSodNrCAobWC3fw#v=onepage&q&f=false

Worksheet example (but there is at least one computational error): https://medium.com/machine-learning-researcher/decision-tree-algorithm-in-machine-learning-248fb7de819e