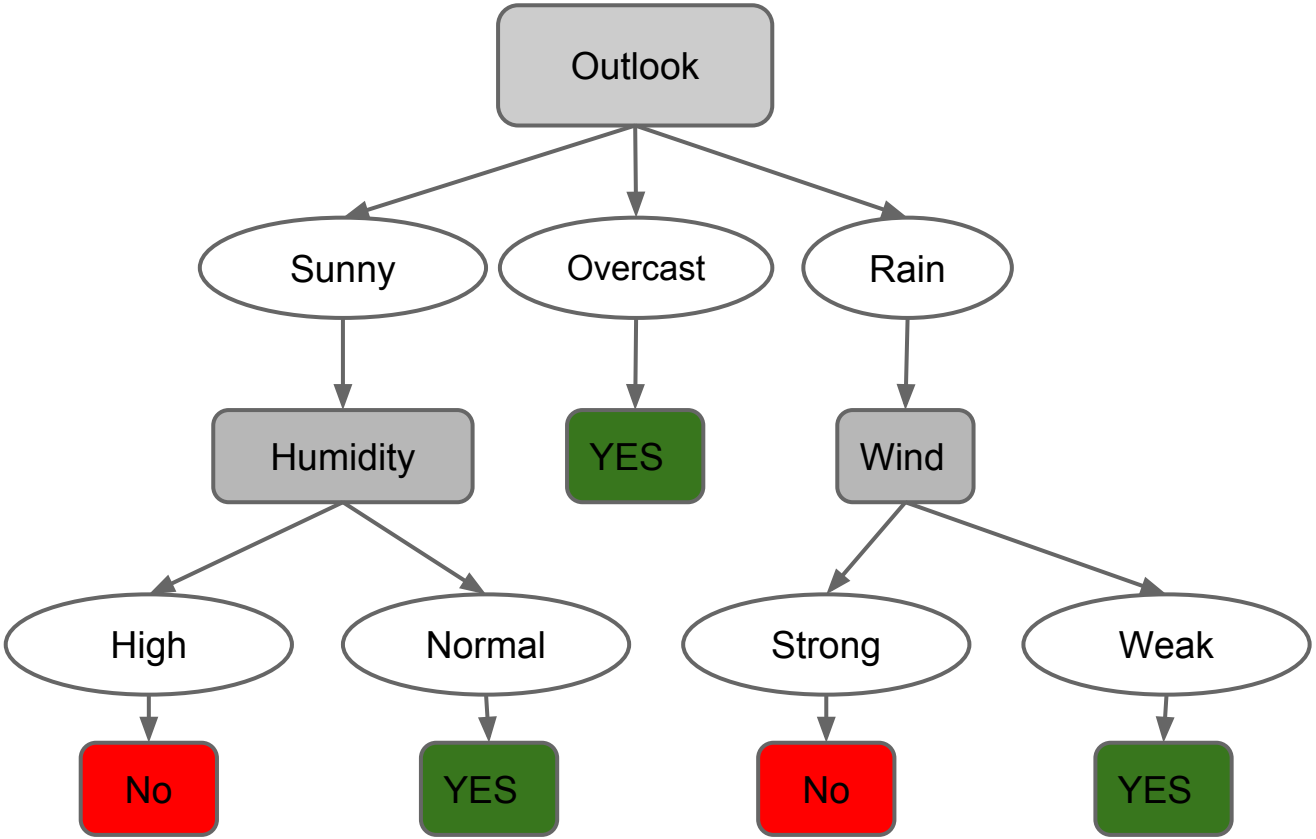
Decision Trees

(Supervised Machine Learning)



[Mitchell, *Machine Learning*]

Algorithms

- ID3 (Iterative Dichotomiser 3)
- C4.5
- CART (Classification and Regression Trees)
- Random Forest (many decision trees)

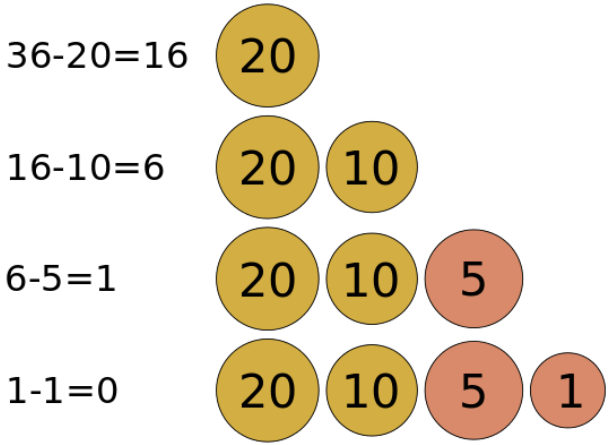
CART is used in sklearn decision trees

ID3

- Core in other decision tree algorithms
- Constructs the tree top down
- **Greedy** search algorithm
- Calculates **information gain**

Greedy Algorithm

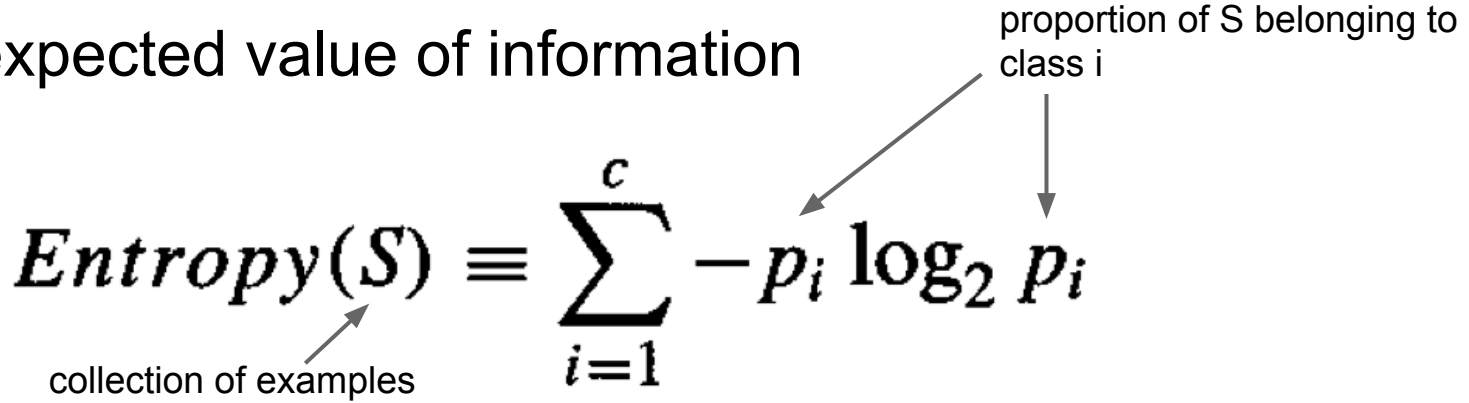
Finds local minimum at each iteration



Example: giving change of 36 cents using min number of 1,5,10,20 cent coins

Entropy (Information theory)

Quantifies expected value of information

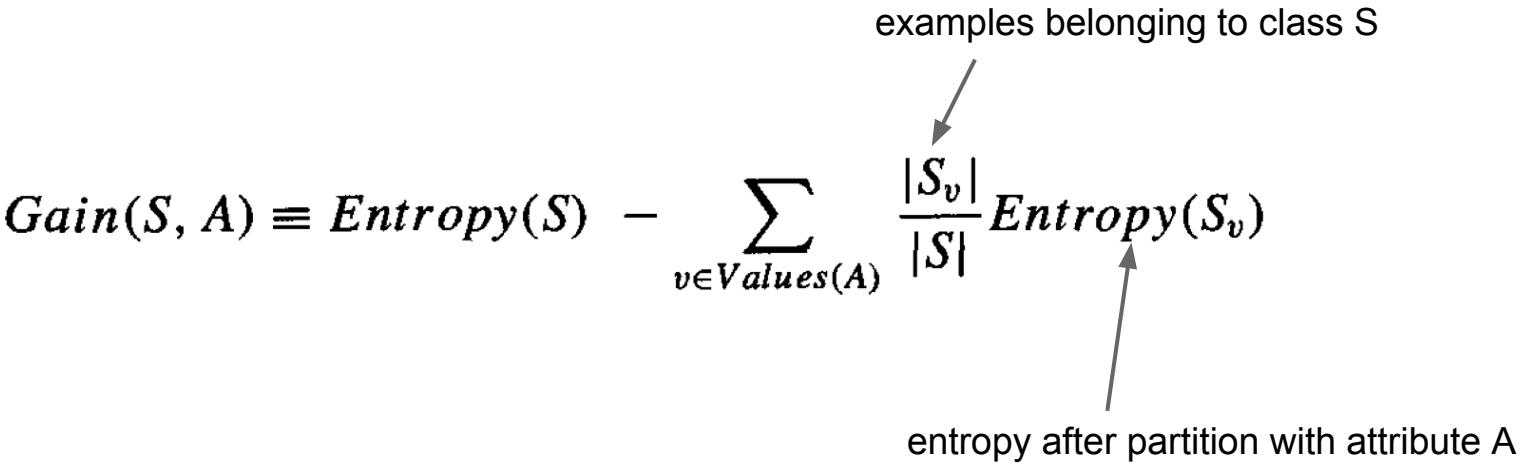


example:

coin toss: - (0.5)log(0.5) - (0.5)log(0.5) = 1 bit,

two coin tosses: 2 bits

Information gain



- Information gain indicates reduction of entropy due to attribute partitioning
- ID3 calculates information gain at each step

Example

From
[Mitchell, *Machine Learning*]

Outlook	Temperature	Humidity	Wind	Play Tennis?
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

Original Entropy

5 negative examples, 9 positive examples

Entropy = -(5/14)log(5/14) - (9/14)log(9/14) = 0.940

Entropy change due to wind attribute

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Wind = [Weak, Strong]

Wind = weak for 6 positive (+) and 2 negative (-)

Wind = strong for 3+ and 3-

Gain(S,Wind) = 0.940 - (8/14)Entropy(S_{weak})+(6/14)Entropy(S_{Strong}) =

= 0.940 - (8/14)[(2/8)log(2/8) - (6/8)log(6/8)] - (6/14)(1)

= 0.048

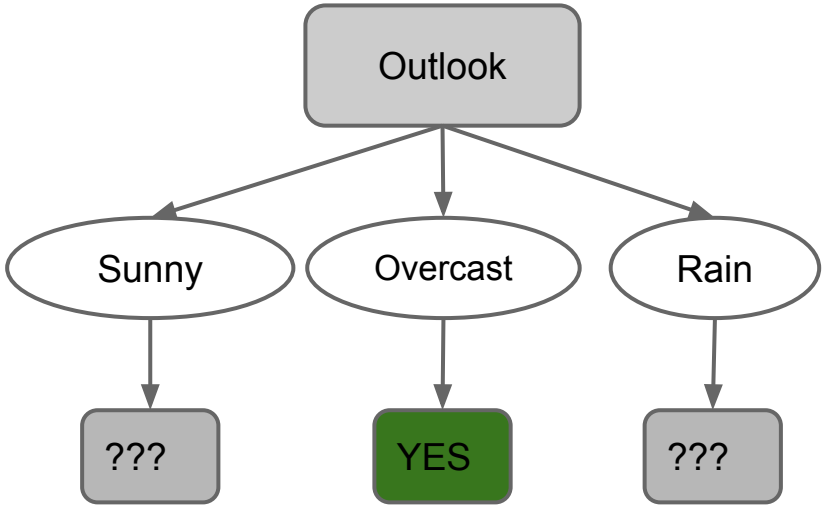
Top of the tree

Gain(S, Outlook) = 0.246

Gain(S, Humidity) = 0.151

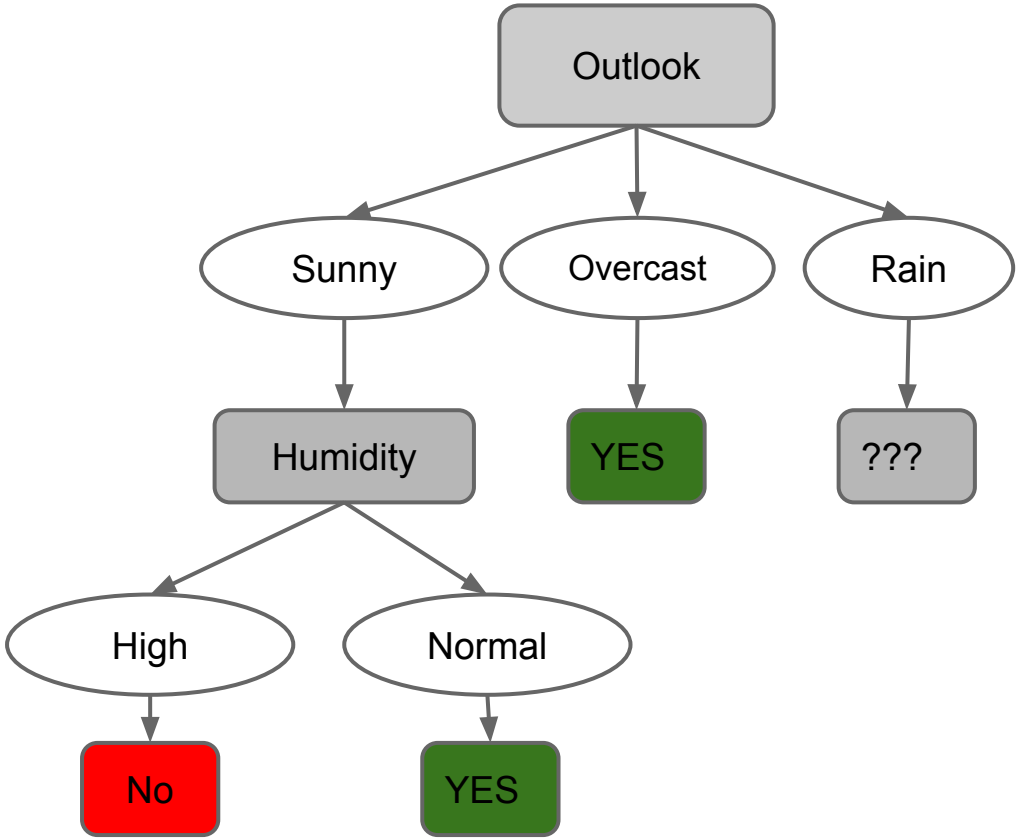
Gain(S, Wind) = 0.048

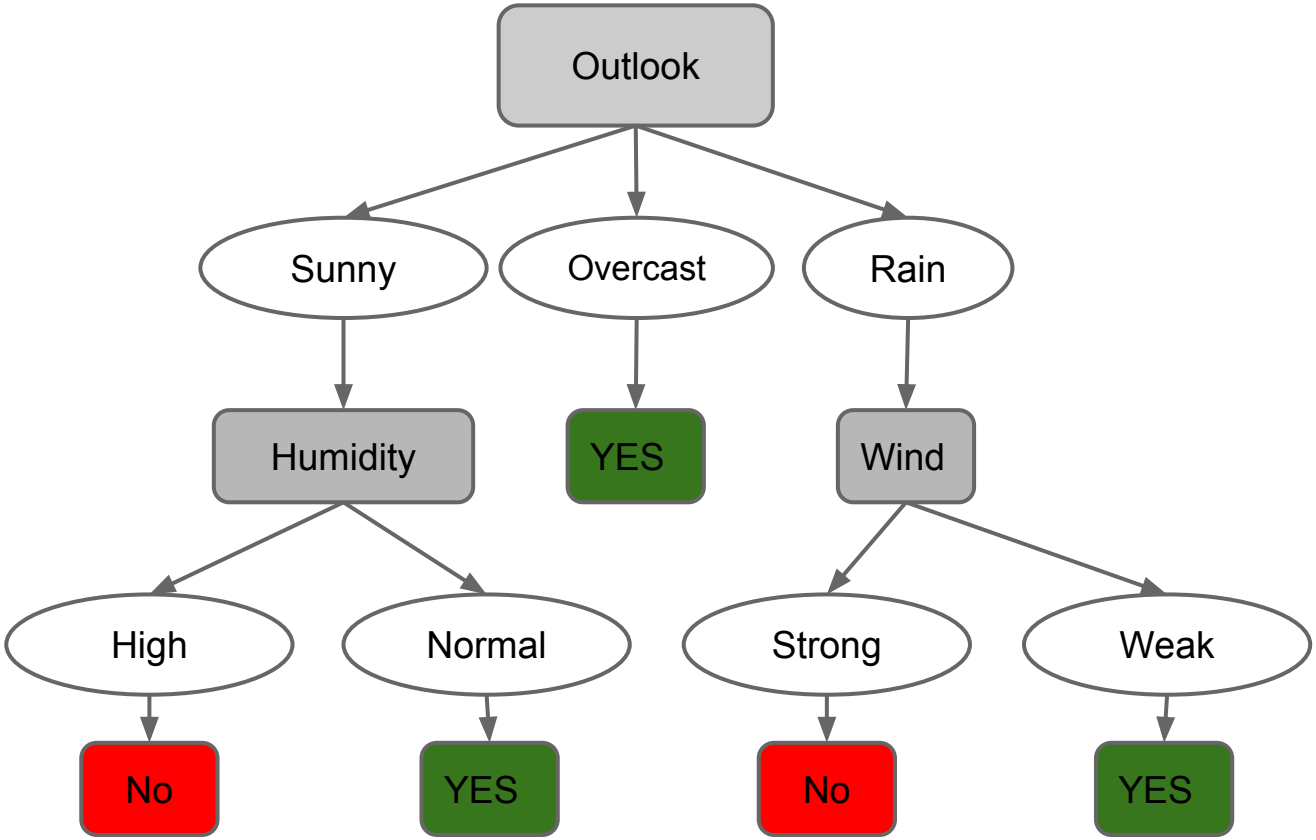
Gain(S, Temperature) = 0.029

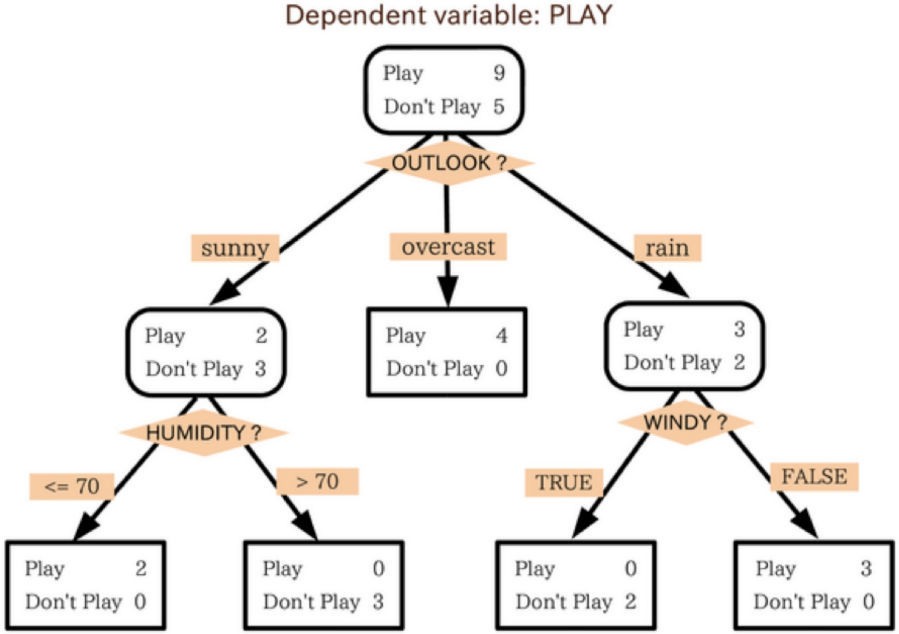


Branches

Gain(S_{sunny}, Humidity) = **0.970**
Gain(S_{sunny}, Temperature) = 0.570
Gain(S_{sunny}, Wind) = 0.019




Outlook
Sunny
Overcast
Rain
Humidity
YES
Wind
High
Normal
Strong
Weak
No
YES
No
YES


Dependent variable: PLAY
Play 9
Don't Play 5
OUTLOOK ?
sunny
overcast
rain
Play 2
Don't Play 3
Play 4
Don't Play 0
Play 3
Don't Play 2
HUMIDITY ?
WINDY ?
<= 70
> 70
TRUE
FALSE
Play 2
Don't Play 0
Play 0
Don't Play 3
Play 0
Don't Play 2
Play 3
Don't Play 0

Numerical data

- C4.5 can handle
- Information gain is used to produce a threshold

Decision Trees Summary

Trees is created by using greedy optimization of information gain

Advantages / Disadvantages

Advantages:

- Can be visualized
- Computation complexity is small
- Robust to noise
- Can take wide range of data

Disadvantages:

- Tend to overfit

Practical applications

To avoid overfitting

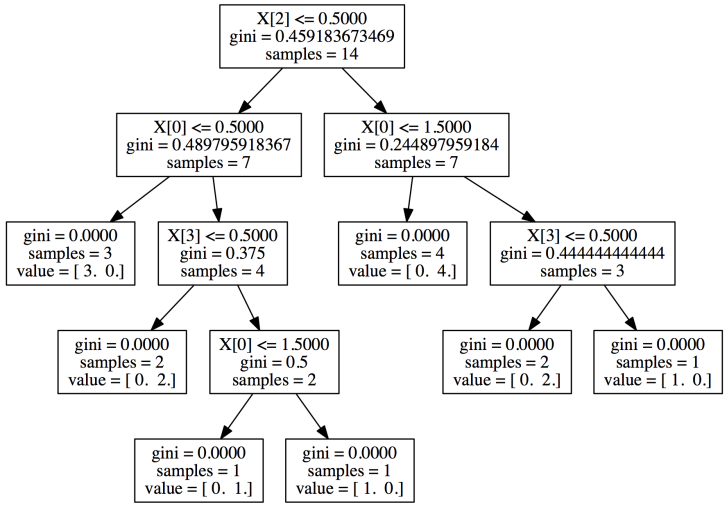
- Use separate training and evaluation datasets.
- Consider using only the key features.
- Limit number of branches

Same number of training data for each class

Decision trees in python

- DecisionTreeClassifier( ) and DecisionTreeRegression( )
- Parameter *max_depth* controls the size of the tree
- *Min_sample_leaf* controls number of samples at leave nodes
- Can visualize the tree:

```
from sklearn.externals.six import StringIO
with open("iris.dot", 'w') as f:
f = tree.export_graphviz(clf, out_file=f)
```



Random Forest

Ensemble Supervised Learning

Algorithm

- Ensemble: use multiple models for prediction
- Constructs a number of trees with random variations
- Incorporates bagging
- All trees predict by voting

Uses

- Good for predicting complex datasets
- Usually doesn’t overfit
- Computationally slow.