Data X

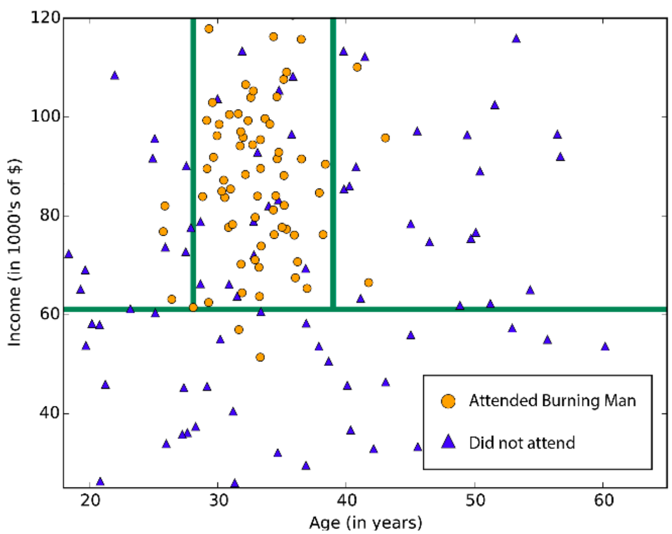
About Me:

Trees, Entropy, Forests, Boosting

Data, Signals, and Systems

Ikhlaq Sidhu
Chief Scientist & Founding Director,
Sutardja Center for Entrepreneurship & Technology
IEOR Emerging Area Professor Award, UC Berkeley

Decision Tree Illustration



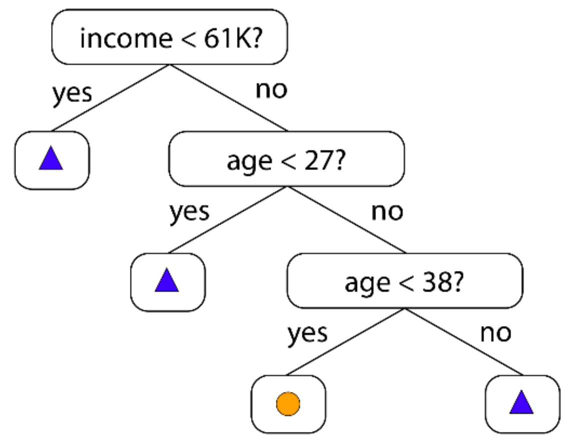
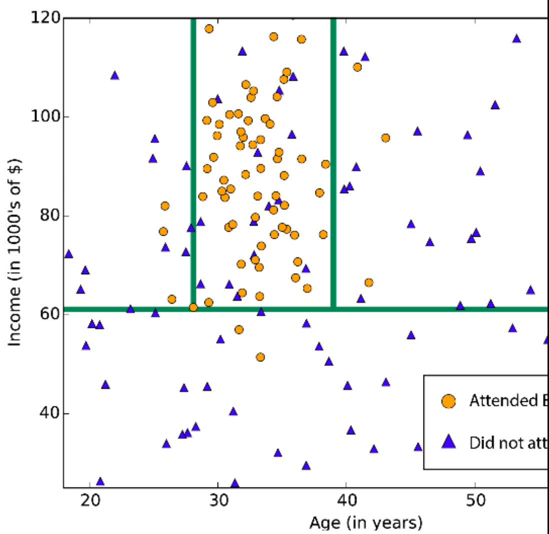


Illustration Source: https://docs.microsoft.com/en-us/azure/machine-learning/machine-learning-algorithm-choice

Decision Tree Illustration

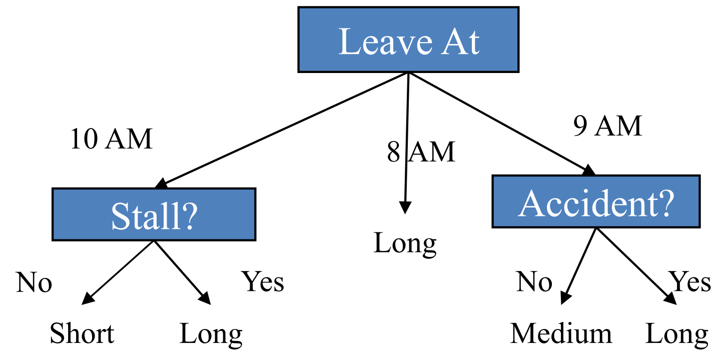



```
from sklearn import tree

decision_tree = DecisionTreeClassifier()
decision_tree.fit(X_train, Y_train)
Y_pred = decision_tree.predict(X_test)

# Error
acc_decision_tree =
round(decision_tree.score(X_train, Y_train) *
100, 2)
acc_decision_tree

# or compare Y_pred with Y_test
```

Illustration Source: https://docs.micr ... ce

How Are Trees Created?

	Attributes				Target
	Hour	Weather	Accident	Stall	Commute
D1	8 AM	Sunny	No	No	Long
D2	8 AM	Cloudy	No	Yes	Long
D3	10 AM	Sunny	No	No	Short
D4	9 AM	Rainy	Yes	No	Long
D5	9 AM	Sunny	Yes	Yes	Long
D6	10 AM	Sunny	No	No	Short
D7	10 AM	Cloudy	No	No	Short
D8	9 AM	Rainy	No	No	Medium
D9	9 AM	Sunny	Yes	No	Long
D10	10 AM	Cloudy	Yes	Yes	Long
D11	10 AM	Rainy	No	No	Short
D12	8 AM	Cloudy	Yes	No	Long
D13	9 AM	Sunny	No	No	Medium

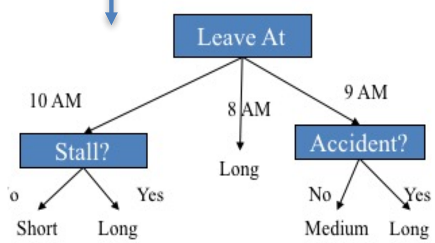


The choice of which feature/attribute to split depends on "entropy"

Example from Jeff Story, source Internet

How Are Trees Created?

	Attributes				Target
	Hour	Weather	Accident	Stall	Commute
D1	8 AM	Sunny	No	No	Long
D2	8 AM	Cloudy	No	Yes	Long
D3	10 AM	Sunny	No	No	Short
D4	9 AM	Rainy	Yes	No	Long
D5	9 AM	Sunny	Yes	Yes	Long
D6	10 AM	Sunny	No	No	Short
D7	10 AM	Cloudy	No	No	Short
D8	9 AM	Rainy	No	No	Medium
D9	9 AM	Sunny	Yes	No	Long
D10	10 AM	Cloudy	Yes	Yes	Long
D11	10 AM	Rainy	No	No	Short
D12	8 AM	Cloudy	Yes	No	Long
D13	9 AM	Sunny	No	No	Medium



Prediction can be coded

All features might not even be used (Occam's Razor)

```
if hour == 8am
    commute time = long
else if hour == 9am
    if accident == yes
      commute time = long
    else
      commute time = medium
else if hour == 10am
    if stall == yes
      commute time = long
    else
      commute time = short
```

Example from Jeff Story, source Internet

A Table

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
1	1	0	0

P(Y=1) = .5

P(X1=1) = .5

P(X2=1) = .5

P(X3=1) = .5

Y = Result output

If you want an efficient tree, which Variable (X1, X2, or X3) will you use as the first branch of the tree.

A Table

Y	X1	X2	X3
0	0	0	1
0	0	0	1
1	0	1	0
1	0	1	0
0	1	0	1
0	1	1	1
1	1	1	0
1	1	0	0
Sorted X1 and then Y			

Y=1 w/p=.5

X1

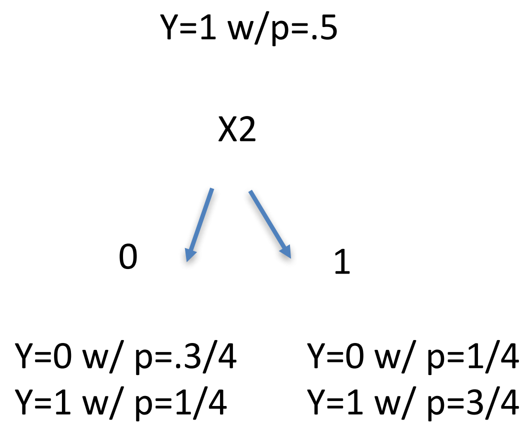
0   1

Y=0 w/ p=.5
Y=1 w/p=.5

Y=0 w/ p=.5
Y=1 w/ p=.5

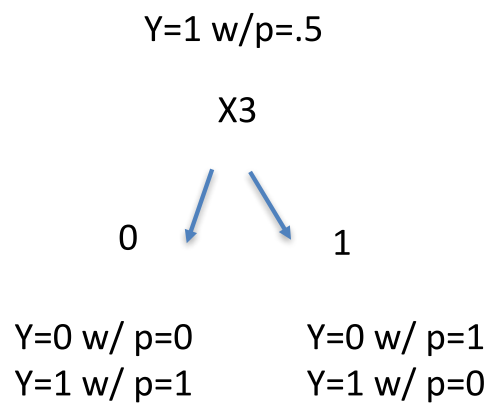
A Table

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
1	1	0	0
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
sorted X2 and then by Y			



A Table

Y	X1	X2	X3
1	0	1	0
1	1	1	0
1	0	1	0
1	1	0	0
0	0	0	1
0	1	0	1
0	0	0	1
0	1	1	1
Sorted X3 and then by Y			



How much information is in a ‘data value’ from column, eg ‘Y’?

P(Y=1) = .5

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
1	1	0	0

How much information is in an observed value
In the Y column?

$$Y = \begin{cases} 1 \text{ w } p = 1/2 \\ 0 \text{ with } p = 1/2 \end{cases} \longrightarrow \text{Suppose outcome} = 1$$

$$Y = \begin{cases} 1 \text{ w } p = 1 \\ 0 \text{ w } p = 0 \end{cases} \longrightarrow \text{Suppose outcome} = 1$$

We can use “Entropy” as a Measure of Information

$$H(x) = \sum_x p(x) \log\left(\frac{1}{p(x)}\right)$$

A Measure of a distribution P(X)

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
1	1	0	0

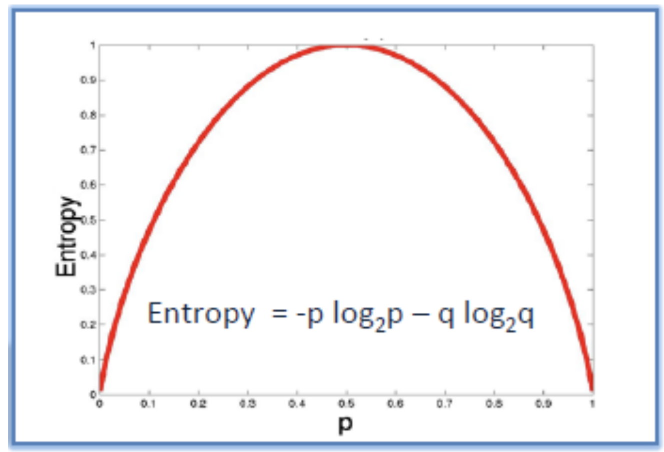
Examples:

X = 1 with p = .5
H(X) = .5 $\log_2$ (1/.5) + .5 $\log_2$ (1/.5) = 1 bits

X = 1 with p = 1/4
H(X) = 1/4 $\log_2$ (4) + 3/4 $\log_2$ (4/3) = 0.811 bits

X = 1 with p = 1/10
H(X) = .1 $\log_2$ (10) + 0.9 $\log_2$ (10/9/3) = .46 bits

X=1 w p = 0
H(X) = 0 $\log_2$ ($1/0^+$) + 1 $\log_2$ (1) = 0 bits



DataX

How Do We Construct a Tree. Example ID3 Algorithm

- Find H(Y) = Entropy of Outcome
- Then try each possible feature. Variable: X1, X2, and X3
- Calculate the Conditional Entropy of Y given a choice of either X1, X2 or X3 = H(Y|X)
- Information Gained =
- I(Y;X) = H(Y) – H(Y|X)
- Choose the variable X1, X2, X3 that leads to the most information gained.

Y=1 w/p=.5

H(Y) — Entropy of Outcome

X1 = 0 | X1 = 1

Which Variable to Split? X1, X2 or X3?

	X1 = 0	X1 = 1
Distribution	P(Y|X1=0)	P(Y|X1=1)
Conditional Entropy	H(Y|X1=0)	H(Y|X1=1)

How Do We Construct a Tree. Example ID3 Algorithm

Y	X1	X2	X3
0	0	0	1
0	0	0	1
1	0	1	0
1	0	1	0
0	1	0	1
0	1	1	1
1	1	1	0
1	1	0	0
Sorted X1 and then Y			

X1

0 1

Y=0 w/ p=.5
Y=1 w/p=.5

Y=0 w/ p=.5
Y=1 w/ p=.5

H(Y)

X1 = 0
W p = 1/2

X1 = 1

Entropy of Outcome
H(Y) = 1

Which Variable to Split?
Lets Try X1

P(Y=1|X1=0)
=0.5

P(Y=1|X1=1)
=0.5

H(Y|X1=0)
= 1

H(Y|X1=1)
= 1

H(Y|X) = 1 x .5 + 1 x .5 = 1.0

Info Gained = H(Y) – H(Y|X) = 1 – 1 = 0

How Do We Construct a Tree. Example ID3 Algorithm

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
1	1	0	0
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
sorted X2 and then by Y			

X2

0   1

Y=0 w/ p=.3/4
Y=1 w/ p=1/4

Y=0 w/ p=1/4
Y=1 w/ p=3/4

H(Y)

X2 = 0
W p = 1/2

X2 = 1

Entropy of Outcome
H(Y) = 1

Which Variable to Split?
Lets Try X1

P(Y=1|X2=0)
= 1/4

P(Y=1|X2=1)
= 3/4

H(Y|X1=0)
= ¼log4+ 3/4 log4/3
=.81

H(Y|X1=1)
= 3/4log4/3+ 1/4log4
= 0.81

H(Y|X) = 0.81 x 0.5 + 0.81 x 0.5 = 0.81
Info Gained = H(Y) – H(Y|X) = 1 – 0.81 = 0.19

How Do We Construct a Tree. Example ID3 Algorithm

Y	X1	X2	X3
0	0	0	1
0	1	0	1
0	0	0	1
0	1	1	1
1	0	1	0
1	1	1	0
1	0	1	0
1	1	0	0

Y=1 w/p=.5

X3

0 | 1

Y=1 w/ p=1 | Y=1 w/ p=0

H(Y)

X3 = 0
W p = 1/2

X3 = 1

Entropy of Outcome
H(Y) = 1

Which Variable to Split?
Lets Try X1

P(Y=1|X3=0)
= 1

P(Y=1|X3=1)
= 0

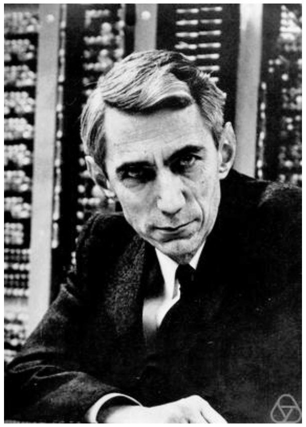
H(Y|X1=0)
= 1 log(1/1)+
0 log(1/ 0)
=0

H(Y|X1=1)
= 1 log(1/1)+ 0 log 1/0
=0

H(Y|X) = 0 x 0.5 + 0 x 0.5 = 0
Info Gained = H(Y) – H(Y|X) = 1 – 0 = 1

Lets Understand Entropy

Claude Shannon,
"the father of
information theory"



Entropy is a measure of Information in a source
(or transmitted, lost or received)

Example:
- 4 Symbols: A, B, C, D
- P(A) = 6/12,   P(B) = 3/12,   P(C)= 2/12,   P(D) = 1/12

Possibly transmitted in this sequence: AABA ABBC CAAD … → “?”

Q: How many bits of information is in A, B, C, D and the “next missing random symbol”

Lets First Understand Entropy

Entropy is a measure of Information in a source (or transmitted, lost or received)

Information:
- Increases with less likelihood
- Additive for combinations
  Options A x B -> H(A) + H(B)
- An event with p = 0 has no info

Example:
- 4 Symbols: A, B, C, D
- P(A) = 6/12,   P(B) = 3/12,   P(C)= 2/12,   P(D) = 1/12

Possibly transmitted in this sequence: AABA ABBC CAAD … → "?"

Q: How many bits of information is in A, B, C, D and the "next missing random symbol"

Answer relates to Entropy

- Info Content: h(p) = p $\log_2(1/p)$ for a symbol with probability p
- h(A) = $\log_2(2)$ = 1 bit, h(B) = $\log_2(4)$ = 2 bits, h(C)= $\log_2(6)$, h(D) = $\log_2(12)$

- H(S) = ½ $\log_2(2)$ + 1/4 $\log_2(4)$ + 1/6 $\log_2(6)$ + 1/12 $\log_2(12)$ = 1.72 bits

$$H(x) = \sum_x p(x) \log\left(\frac{1}{p(x)}\right)$$

Entropy Related to the Information Contained in the Symbol (or Table)

Entropy is a measure of Information in a source (or transmitted, lost or received)

Example:

- P(A) = 6/12
- P(B) = 3/12
- P(C)= 2/12
- P(D) = 1/12
- Possibly transmitted in this sequence: AABA ABBC CAAD … → "?"

Source Coding Theorem:
The H(S) is the smallest codeword length that is theoretically possible for signal 'S'

Theoretically smallest code is 1.72 bits per symbol

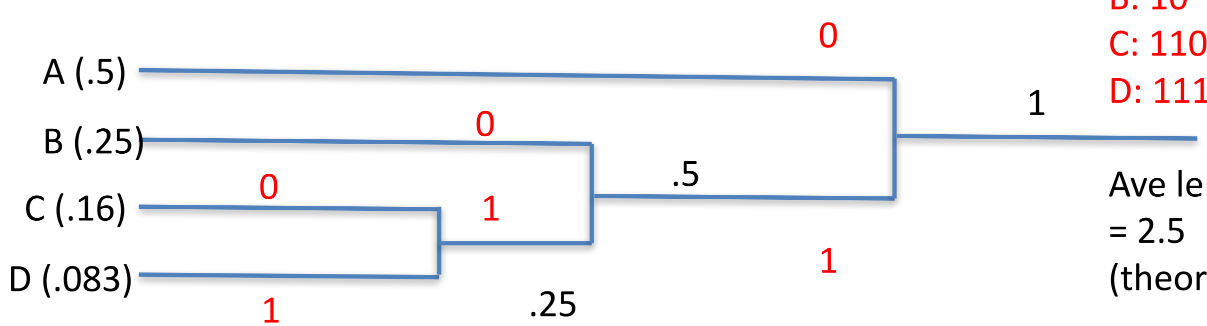
Sample Coding Schemes

Example:

- 4 Symbols: A, B, C, D
- Suppose each is “encoded” in 4 bits, ie A = ‘0001’
- P(A) = 6/12,   P(B) = 3/12,   P(C)= 2/12,   P(D) = 1/12

- We could code in 2 bits (00, 01, 10, 11)
- Most famous prefix free coding is called a Huffman Code:

A prefix free code:
A: 0
B: 10
C: 110
D: 111



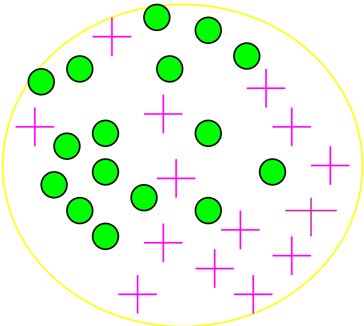
Ave length: 0.5(1) + 0.25(2) + 0.16(3)+.083(3)
= 2.5
(theoretically best would be 1.72)

DataX

Examples with Tables, Trees and Entropy

Entropy Example

- Entropy = $\sum_i -p_i \log_2 p_i$



p_i is the probability of class i

Compute it as the proportion of class i in the set.

16/30 are green circles; 14/30 are pink crosses

$\log_2(16/30) =$ -.9; $\log_2(14/30) =$ -1.1

Entropy = -(16/30)(-.9) –(14/30)(-1.1) = .99

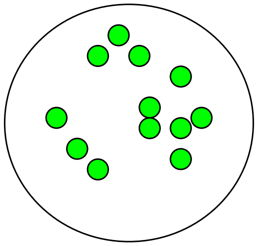
Source Internet

2-Class Cases:

- What is the entropy of a group in which all examples belong to the same class?
  - entropy = $-1 \log_2 1 = 0$

not a good training set for learning

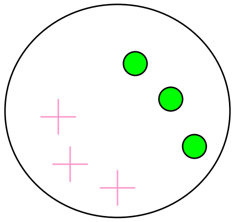
Minimum impurity



- What is the entropy of a group with 50% in either class?
  - entropy = $-0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

good training set for learning

Maximum impurity



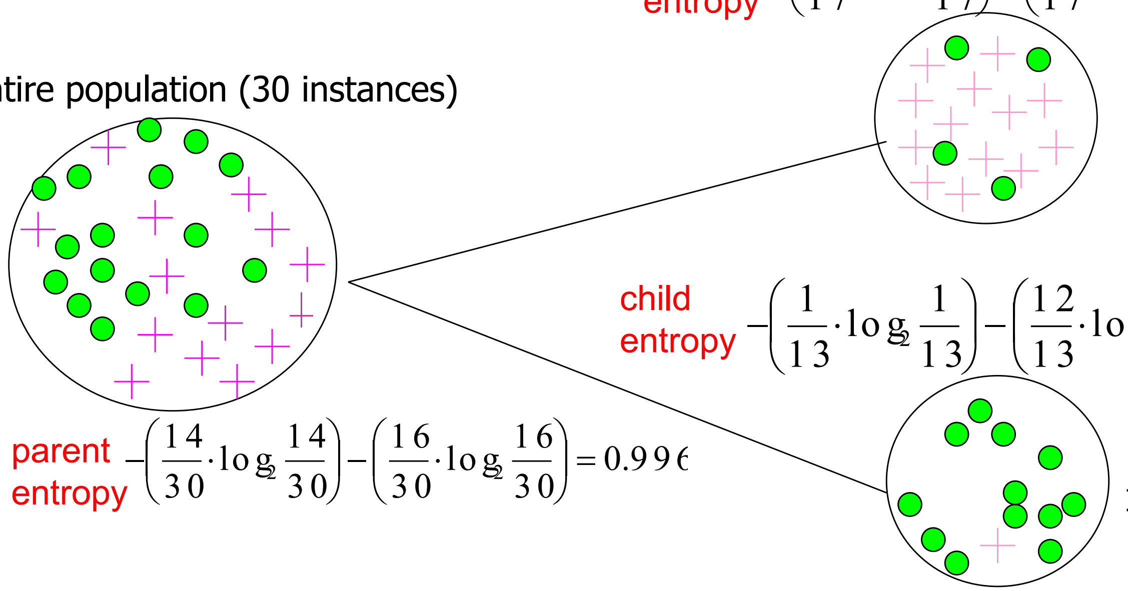
Source Internet

Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$

Entire population (30 instances)



child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

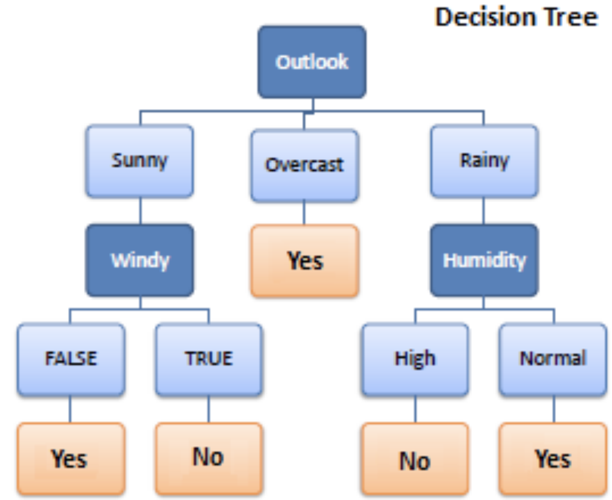
(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38 **for this split**

Golf Example

Predictors | Target

Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No

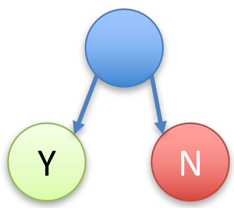


Start with table of experiences

- Distribution: P(golf) = 9 yes, 5 no14 observations
- We calculate entropy of distribution: playing golf
- Need to calculate entropy of each next possible decision (Outlook, Temp, Humidity, and Windy)
- Information gained is E(Golf) – E(Subset, Golf),
- We choose the next branch to be the decision with the highest information gained for the next branch

Golf Example

Top Level
Play Golf
Distribution

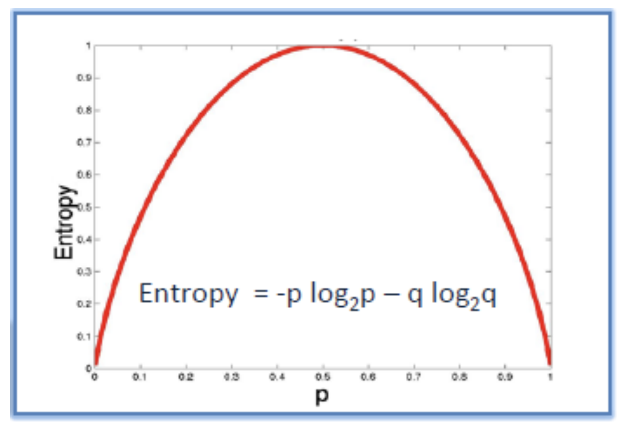


9/14 5/14

$$E(G) = 9/14 \log_2 1/(9/14) + 5/14 \log_2 1/(5/14) = 0.94$$

- Distribution: P(golf) = 9 yes, 5 no14 observations
- We calculate entropy of distribution: playing golf
- Need to calculate entropy of each next possible decision (Outlook, Temp, Humidity, and Windy)
- Information gained is E(Golf) – E(Subset, Golf),
- We choose the next branch to be the decision with the highest information gained for the next branch

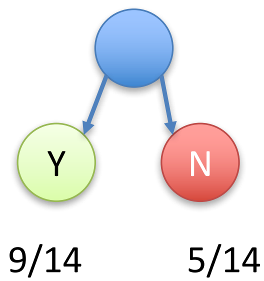
Predictors				Target
Outlook	Temp	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No



$$\text{Entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$$

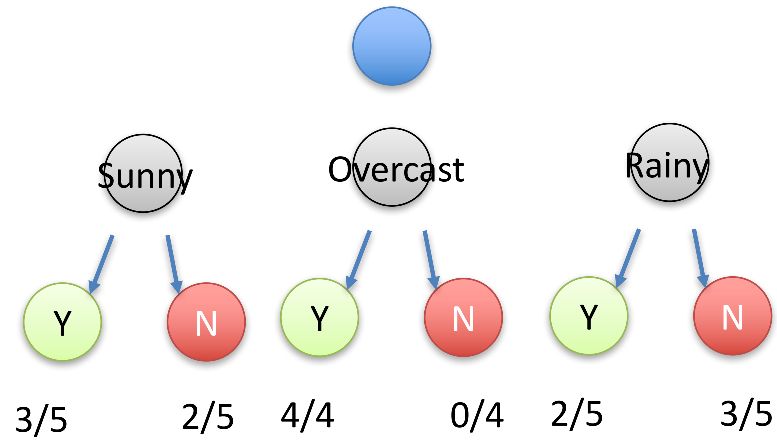
DataX

Top Level
Play Golf
Distribution



$$H(G) = 9/14 \log_2 1/(9/14) + 5/14 \log_2 1/(5/14) = 0.94$$

Next Level: Play Golf with Outlook



H(G,OL) =
P(Sunny) H(G,OL) + P(OV)H(G,OV) + P(Rainy) H(G,Rainy)

= 5/14* 0.971+4/14*0 + 5/14*.971 = 0.693

Info Gained = H(G) - H(G,OL) = 0.94 – 0.693 = 0.247

Do the same with Temp (.29), Windy(.048), and Humidity(.152):
Choose next node to be the one with the most info gained

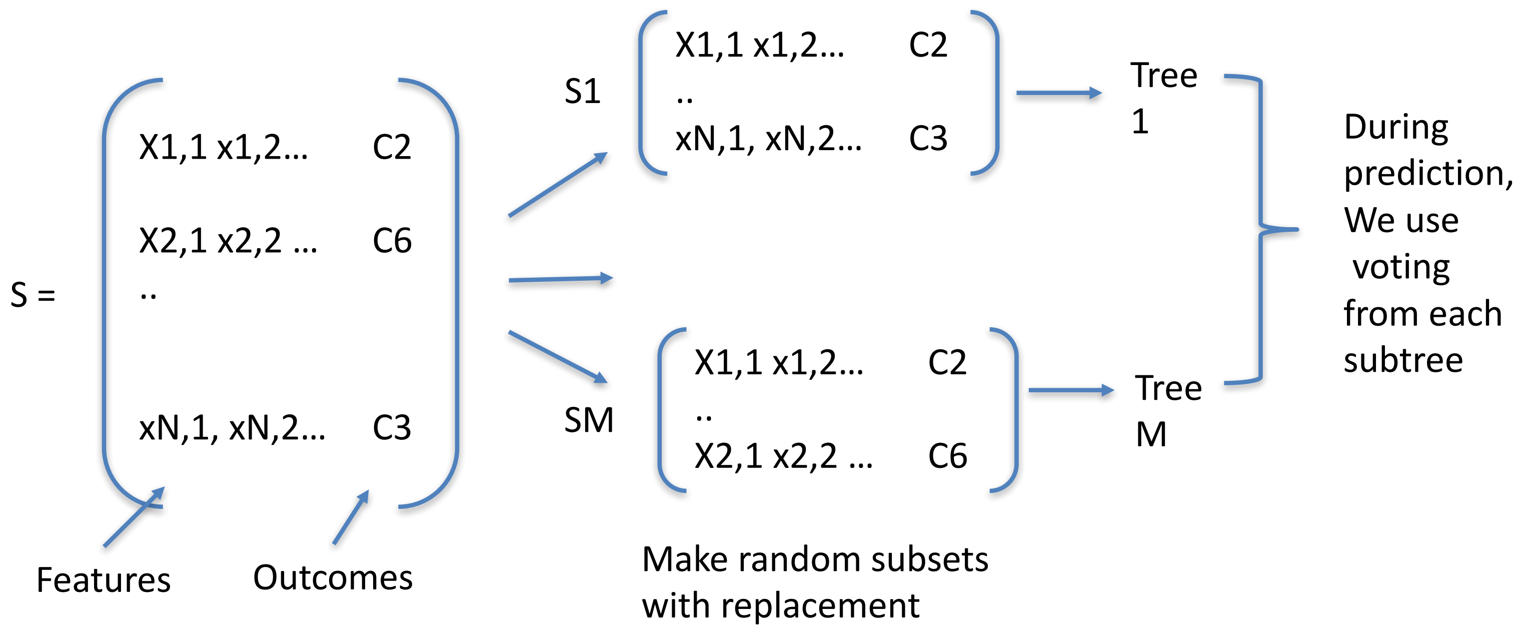
Random Forrest

Trees Can be Extended with Bagging

Explain bagging and Random Forrest

```
from sklearn.ensemble import RandomForestClassifier

random_forest =
RandomForestClassifier(n_estimators=1000)
random_forest.fit(X_train, Y_train)
Y_pred = random_forest.predict(X_test)
random_forest.score(X_train, Y_train)

# Error
acc_random_forest = round(random_forest.score(X_train,
Y_train) * 100, 2)
acc_random_forest

# or compare Y_pred with Y_test
```

Random Forest – A type of bagging/ensemble approach



Advantages: One of most accurate
Efficient prediction over large data

Disadvantages: Overfit and Training time

Our experiment with the Titanic Data Set

	Model	Score
	Random Forest	86.76
	Decision Tree	86.76
	KNN	84.74
	Support Vector Machines	83.84
	Logistic Regression	80.36
	Linear SVC	79.01
	Perceptron	78.00
	Naive Bayes	72.28
	Stochastic Gradient Decent	72.28

More Accuracy
Generally more training time
More risk of overfitting

Less Accuracy
Generally less computation

Boosting

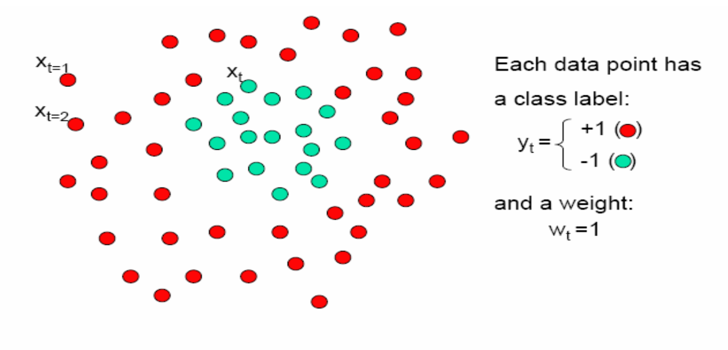
Boosting as in AdaBoost

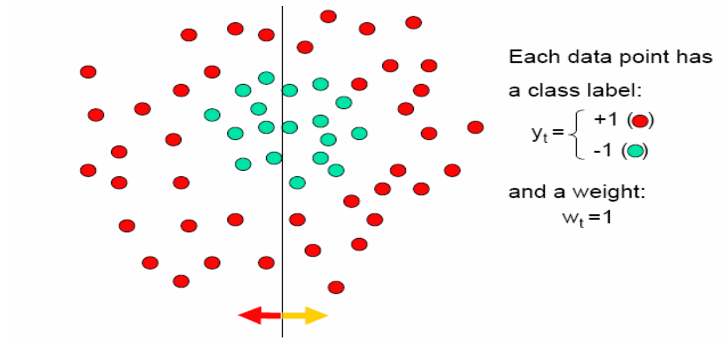
- Motivation - a procedure that combines the outputs of many “weak” classifiers to produce a powerful “committee”
- A machine learning algorithm
- Perform supervised learning
- Increments improvement of learned function
- Forces the weak learner to generate new hypotheses that make less mistakes on “harder” parts.

- Freund & Schapire (1995) – **AdaBoost**
  - strong practical advantages over previous boosting algorithms

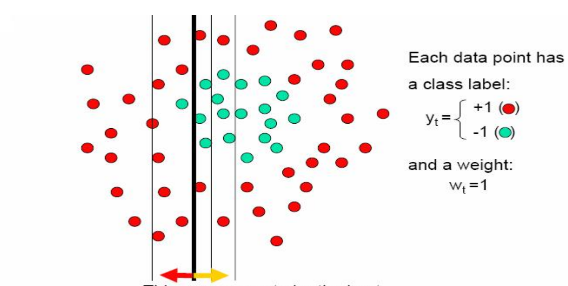
Benk Erika

Kelemen Zsolt



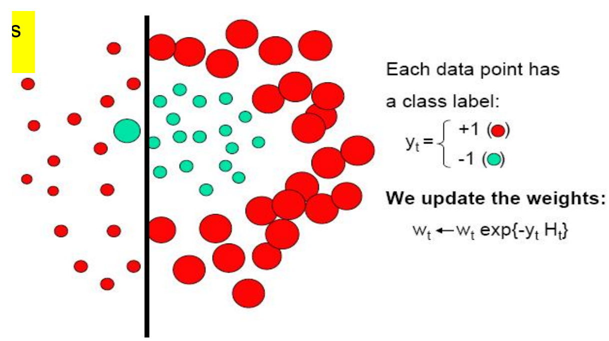


h => p(error) = 0.5 it is at chance

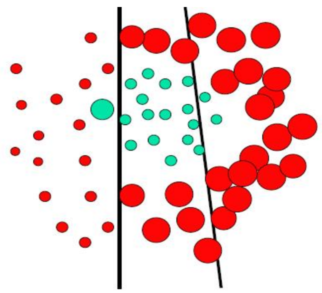


This one seems to be the best

This is a '**weak classifier**': It performs slightly better than chance.



DataX



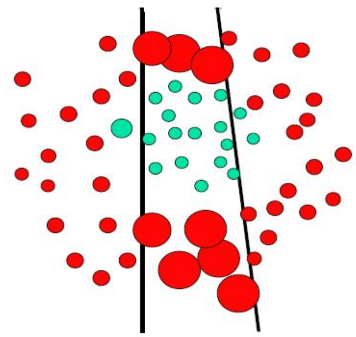
Each data point has
a class label:

$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{green}) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

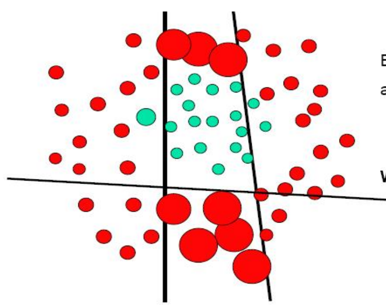


Each data point has
a class label:

$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{green}) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

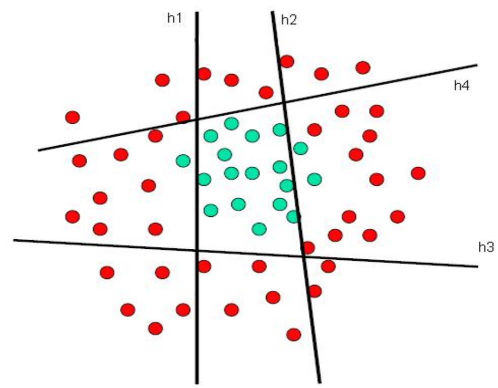


Each data point has
a class label:

$$y_t = \begin{cases} +1 \ (\text{red}) \\ -1 \ (\text{green}) \end{cases}$$

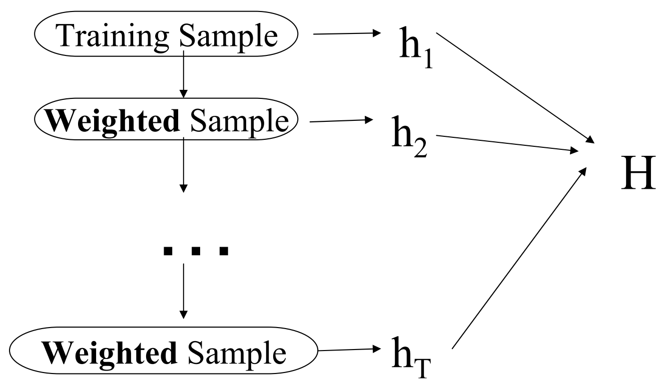
We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$



he strong (non- linear) classifier is built as the comb

General Concept of Boosting



- Train a set of weak hypotheses: $h_1, \ldots, h_T$.
- The combined hypothesis H is a **weighted** majority vote of the T weak hypotheses.
  - ➔ Each hypothesis h_t has a weight α_t.

$$H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$$

Benk Erika
Kelemen Zsolt

Boosting as in AdaBoost

Boosting

- Binary classification problem
- Training data:

$$(x_1, y_1), \ldots., (x_m, y_m), \textit{where} \quad x_i \in X, y_i \in Y = \{-1,1\}$$

- $D_t(i)$: the weight of x_i at round t. $D_1(i)=1/m$.
- A learner L that finds a weak hypothesis h_t: X ➔ Y given the training set and D_t
- The error of a weak hypothesis h_t:

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i] = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$$

Benk Erika
Kelemen Zsolt

AdaBoost Algorithm

- **Given** $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \epsilon X$, $y_i \epsilon \{-1, +1\}$
- **Initialise** weights $D_1(i) = 1/m$
- **Iterate** $t = 1, \ldots, T$:
  - Train weak learner using distribution ***Dt***
  - Get weak classifier: $h_t : X \rightarrow \mathrm{R}$
  - Choose $\alpha_t \varepsilon \mathrm{R}$
  - Update: $D_{t+1}(i) = \dfrac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
    - where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution), and α_t:

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) > 0$$

- **Output** – the final classifier

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Benk Erika
Kelemen Zsolt

AdaBoost Algorithm

Discrete AdaBoost - Algorithm

- **Given** $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \epsilon X$, $y_i \epsilon \{-1, +1\}$
- **Initialise** weights $D_1(i) = 1/m$
- **Iterate** $t=1,\ldots,T$:
  - Find $h_t = \arg\min_{h_j} \epsilon_j$ where $\epsilon_j = \sum_{i=1}^{m} D_t(i)[\![h_t(x_i) \neq y_i]\!]$
  - Set

$$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$$

  - Update: $D_{t+1}(i) = \dfrac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$

- **Output** – the final classifier

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

Benk Erika
Kelemen Zsolt

AdaBoost – Pros and Contras

- Pros:
  - Very simple to implement
  - Fairly good generalization
  - The prior error need not be known ahead of time
- Contras:
  - Suboptimal solution
  - Can over fit in presence of noise

End of Section