CMPUT 466
Machine Learning: Day 3

Professor: Bailey Kacsmar
kacsmar@ualberta.ca

Winter 2024

Many of these slides are derived from Alona Fyshe, Alex Thomo. Thanks!

From Last Class:
Probability Density Functions (PDFs)
Maximum likelihood estimation and MAP

MAP and MLE, Basically, Probabilistic Modeling

Given: observations, find a model or function that “shows agreement” and:

MAP and MLE, Basically, Probabilistic Modeling

Given: observations, find a model or function that “shows agreement” and:

- The ability to generalize well
- The ability to incorporate prior knowledge and assumptions
- Scalability

Finding the "best" model?

- Consider learning parameters of a distribution
- Given a set of observations, and some knowledge, the goal is …

Finding the “best” model?

- Consider learning parameters of a distribution
- Given a set of observations, and some knowledge, the goal is …

Essentially…parameter estimation

Finding the "best" model?

- Consider learning parameters of a distribution
- Given a set of observations, and some knowledge, the goal is …

Essentially…parameter estimation

MLE

MAP

Resume: Maximum Likelihood Estimation (MLE)

Rich vs Poor

What is the probability of a person being rich, given you know nothing else about that person?

3:2

What is the probability of a person being rich, given you know nothing else about that person?

There exists a true distribution (from the hypothesis space)

Let’s say 3/5?

We assume that the wealth of the people in our dataset *D* is independently distributed

θ = Probability of being rich = P(rich)

? = Probability of being poor = P(poor)

D = { r, p, r, r, p} α_r = # rich α_p= # poor

$$
\begin{aligned}
P(D) =& P(r \text{ and } p \text{ and } r \text{ and } r \text{ and } p) \\
=& P(rich) * P(poor) * P(rich) * \\
& P(rich) * P(poor) \\
=& \theta * (1-\theta) * \theta * \theta * (1-\theta) \\
=& (1-\theta)^{\alpha_p} * \theta^{\alpha_r}
\end{aligned}
$$

$$\underset{\theta}{\operatorname{argmax}}\ P(D) = (1-\theta)^{\alpha_F} * \theta^{\alpha_H}$$

That’s Maximum Likelihood Estimation (MLE)

It’s not always the best solution…

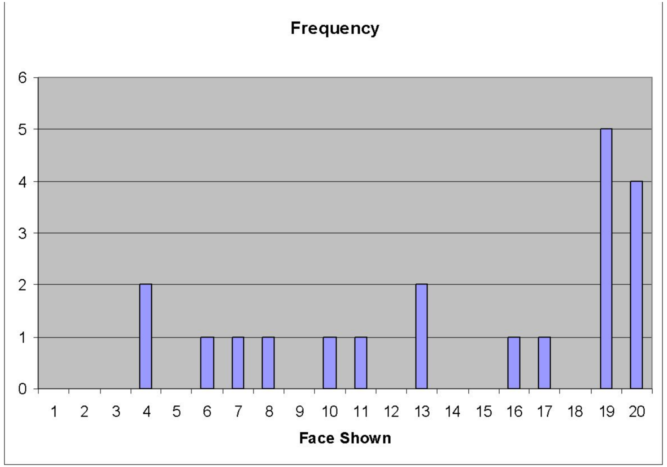
That’s Maximum Likelihood Estimation (MLE)

It’s not always the best solution…

Because: The assumption that the function is constant is problematic.

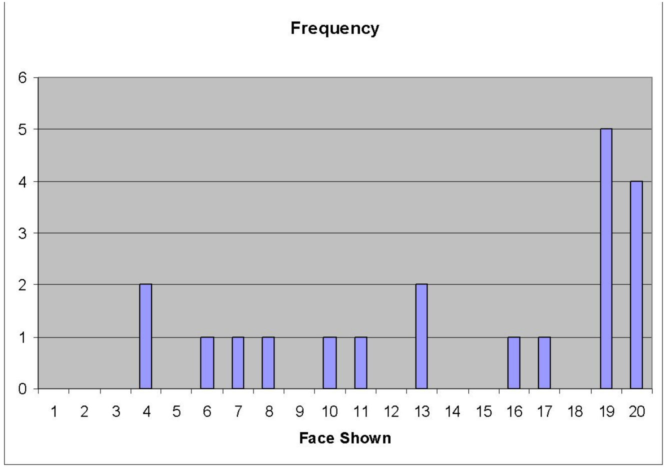
Consider: Issues with MLE estimate

I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



Issues with MLE estimate

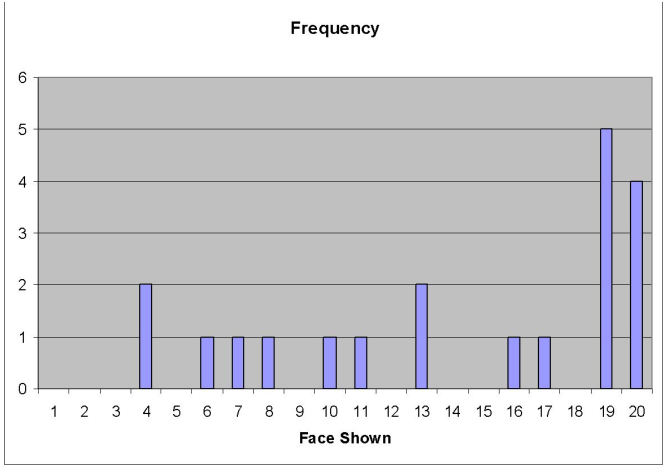
I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



1. Collect some data (20 rolls)

Issues with MLE estimate

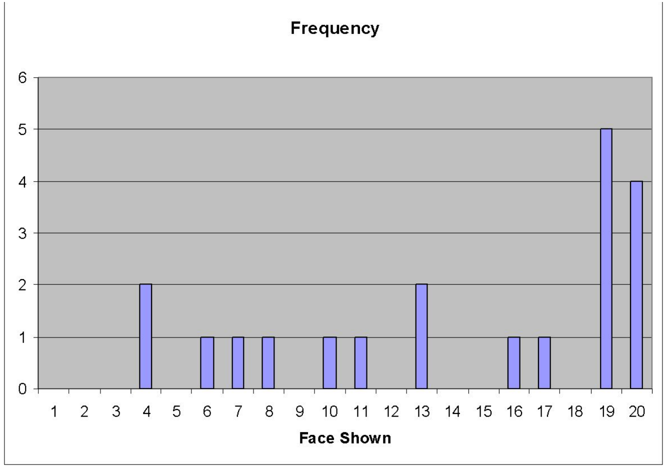
I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



1. Collect some data (20 rolls)
2. Estimate P(i)=CountOf(rolls of i)/CountOf(any roll)

Issues with MLE estimate

I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



P(1)=0

P(2)=0

P(3)=0

P(4)=0.1

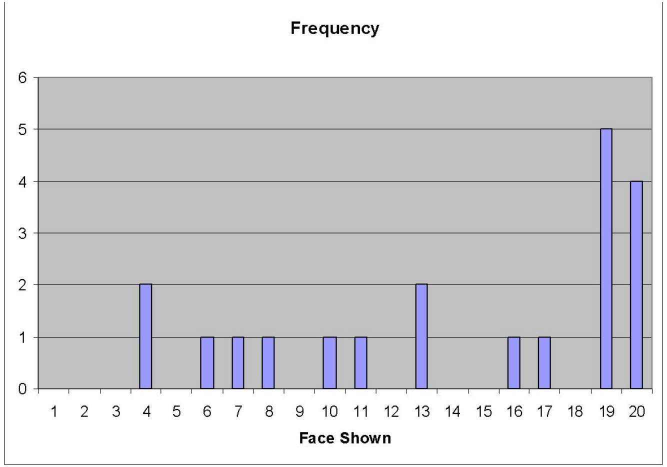
…

P(19)=0.25

P(20)=0.2

Issues with MLE estimate

I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



P(1)=0

P(2)=0

P(3)=0

P(4)=0.1

…

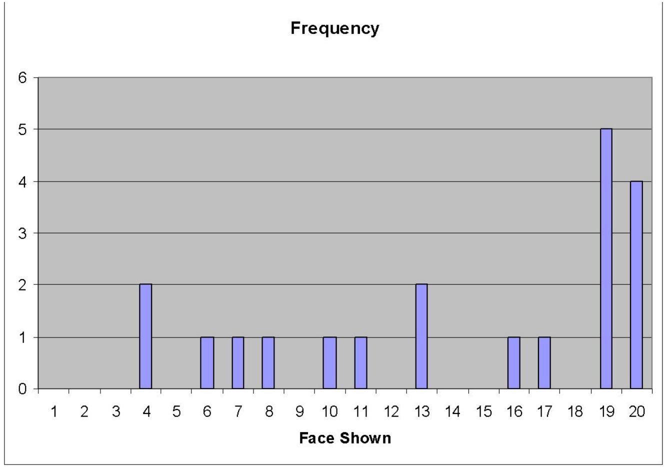
P(19)=0.25

P(20)=0.2

But: Do I really think it's *impossible* to roll a 1,2 or 3?

A better solution?

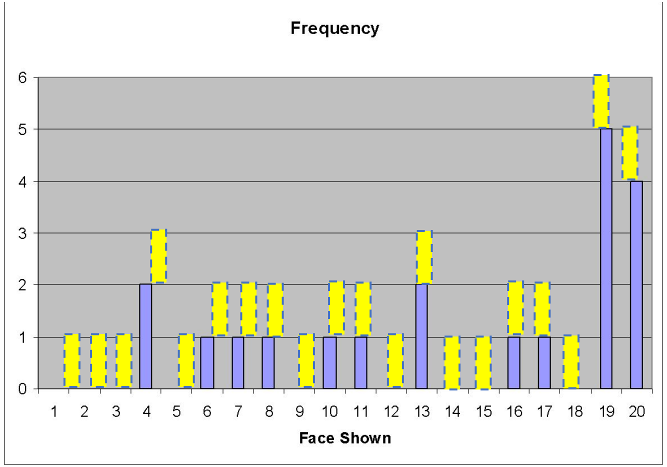
I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



1. Collect some data (20 rolls)
2. Estimate P(i)

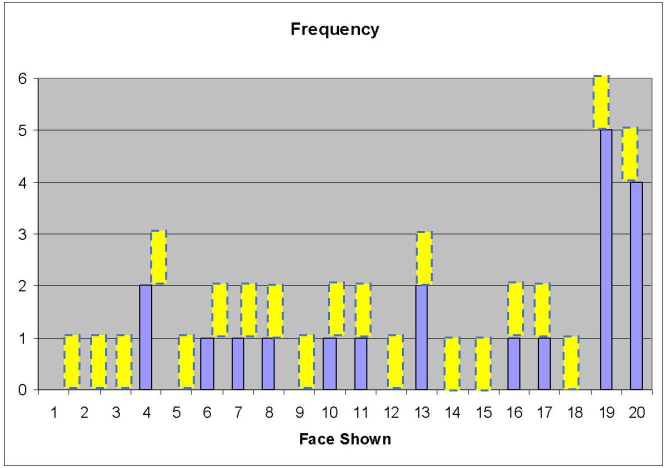
A better solution

I bought a loaded 20-faced die (d20) on EBay…but it didn't come with any specs. How can I find out how it behaves?



0. *Imagine* some data (20 rolls, each i shows up 1x)
1. Collect some data (20 rolls)
2. Estimate P(i)

A better solution?



$$\hat{P}(i) = \frac{CountOf(i)+1}{CountOf(ANY)+CountOf(IMAGINED)}$$

P(1)=1/40

P(2)=1/40

P(3)=1/40

P(4)=(2+1)/40

…

P(19)=(5+1)/40

P(20)=(4+1)/40=1/8

0.2 *vs.* 0.125 – really different! Maybe I should "imagine" less data?

What if we know that poor people are much more common than rich people?

We have a belief about θ

- P(θ|D) = P(D|θ)*P(θ)/P(D)

P(θ|D) is calculating the posterior distribution

P(D|θ) is the likelihood function

P(θ) is the prior distribution

P(f) is the marginal distribution of the data

Now we can incorporate our belief about θ

We have a belief about θ

- $P(\theta|D) = P(D|\theta) * P(\theta) / P(D)$

$$\propto P(D|\theta) * P(\theta)$$

Now we can incorporate our belief about θ

This is a MAP (Maximum A Posteriori) Estimate

We have a belief about θ

- $P(\theta|D) = P(D|\theta) * P(\theta) / P(D)$

$$\propto P(D|\theta) * P(\theta)$$

can incorporate our belief about θ

This is a MAP (Maximum A Posteriori) Estimate

Key idea: find the most probable model(aka function) for the observed data

Conjugate Prior

- Our likelihood so far has been based on a Bernoulli distribution.
- **Beta is a conjugate prior to Bernoulli**
  - This means their pdfs (probability density functions) play nice together

  - **P(D|θ)*P(θ)** will be easy to deal with
  - Called the posterior likelihood

Estimating Parameters

- Maximum Likelihood Estimate (MLE): choose θ that maximizes probability of observed data

$$\widehat{\theta} = \arg\max_{\theta} P(\mathcal{D} \mid \theta)$$

- Maximum a Posteriori (MAP) estimate: choose θ that is most probable given **prior probability and the data**

$$\widehat{\theta} = \arg\max_{\theta} P(\theta \mid \mathcal{D})$$

$$= \arg\max_{\theta} = \frac{P(\mathcal{D} \mid \theta) P(\theta)}{P(\mathcal{D})}$$

A tutorial:
http://www.cs.ubc.ca/~murphyk/Teaching/CS340-Fall06/reading/bernoulli.pdf

Today: Supervised Learning, Classification, Decision Trees

Recall (again)

(Adapted from Leslie Kaelbling's example in the MIT courseware)

- Imagine I'm trying predict whether my neighbor is going to drive into work, so I can ask for a ride.

- Whether she drives into work seems to depend on the following attributes of the day:
  - **temperature**
  - **expected precipitation**
  - **day of the week**
  - **what she's wearing**

Memory

- Now, we find ourselves on a snowy “-5” degree Monday, and the neighbor is wearing casual clothes.

Temp	Precip	Day	Clothes	
25	None	Sat	Casual	**Walk**
-5	Snow	Mon	Casual	**Drive**
15	Snow	Mon	Casual	**Walk**
-5	**Snow**	**Mon**	**Casual**	

Averaging

- One strategy would be to predict the majority outcome.

Temp	Precip	Day	Clothes	
25	None	Sat	Casual	**Walk**
25	None	Sat	Casual	**Walk**
25	None	Sat	Casual	**Drive**
25	None	Sat	Casual	**Drive**
25	None	Sat	Casual	**Walk**
25	None	Sat	Casual	**Walk**
25	None	Sat	Casual	**Walk**
25	None	Sat	Casual	**Walk**

Generalization

- Dealing with previously unseen cases
- Will she walk or drive?

Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	Rain	Mon	Casual	**?**

We might plausibly make any of the following arguments:

- She's going to walk because it's raining today and the only other time it rained, she walked.
- She's going to drive because she has always driven on Mondays…

Today:

Today: A different way to not ask our neighbour whether she’s driving to work

Decision Trees

- Predict by **splitting on attribute values**

Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	**Rain**	**Mon**	**Casual**	**?**

– **She's going to walk because it's raining today and the only other time it rained, she walked.**

Decision Trees

- Predict by **splitting on attribute values**

Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	**Rain**	**Mon**	**Casual**	**?**

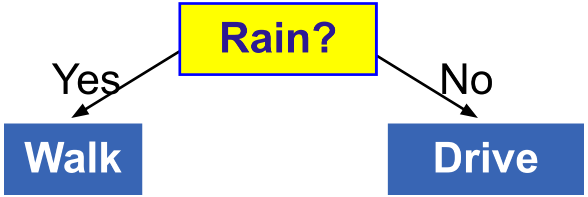
–**She's going to walk because it's raining today and the only other time it rained, she walked.**

Decision Trees

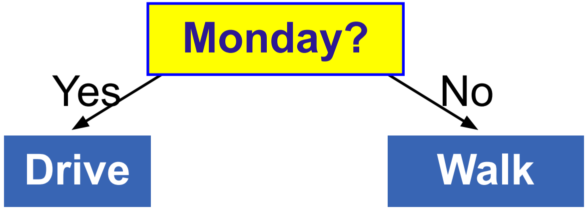
- Predict by **splitting on attribute values**

Temp	Precip	Day	Clothes	
22	None	Fri	Casual	**Walk**
3	None	Sun	Casual	**Walk**
10	Rain	Wed	Casual	**Walk**
30	None	Mon	Casual	**Drive**
20	None	Sat	Formal	**Drive**
25	None	Sat	Casual	**Drive**
-5	Snow	Mon	Casual	**Drive**
27	None	Tue	Casual	**Drive**
24	**Rain**	**Mon**	**Casual**	**?**

– **She's going to walk because it's raining today and the only other time it rained, she walked.**



– **She's going to drive because she has always driven on Mondays…**

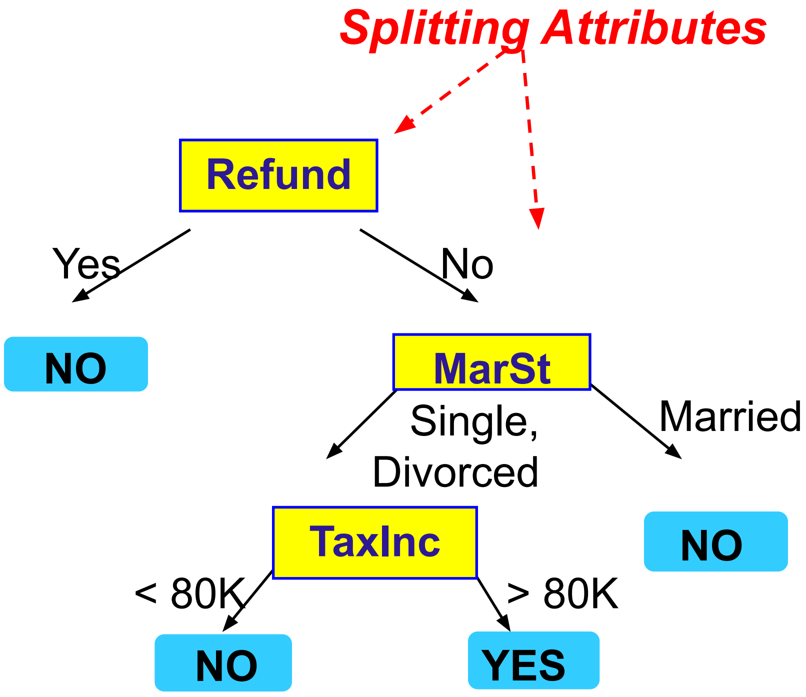


Example of a (Good?) Decision Tree

categorical | categorical | continuous | class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

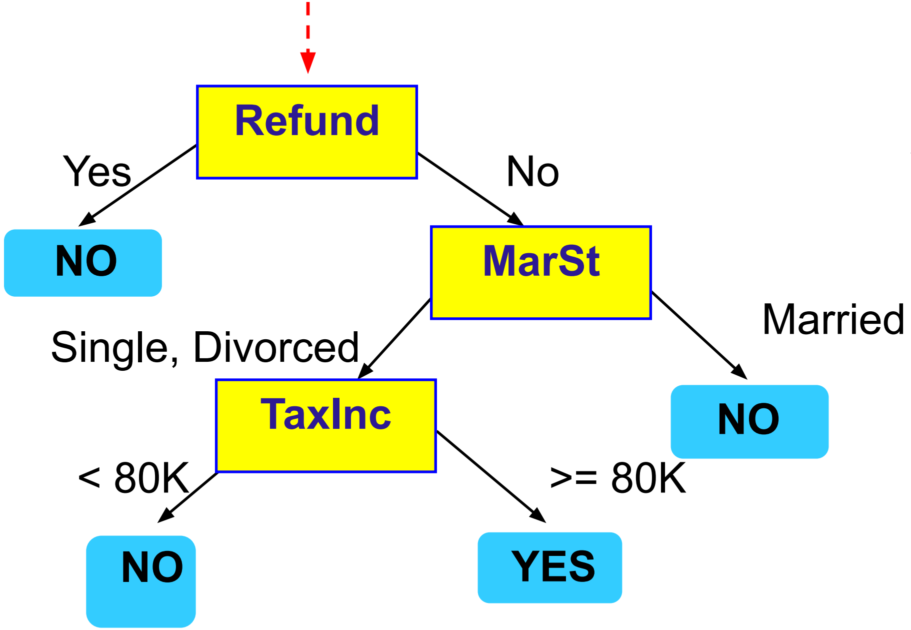
Training Data



Model: Decision Tree

Apply Model to Test Data

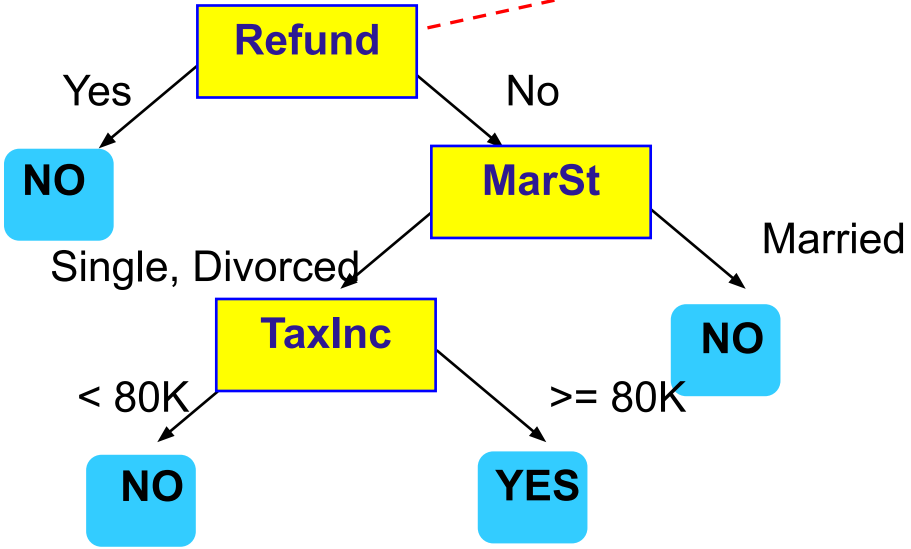
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

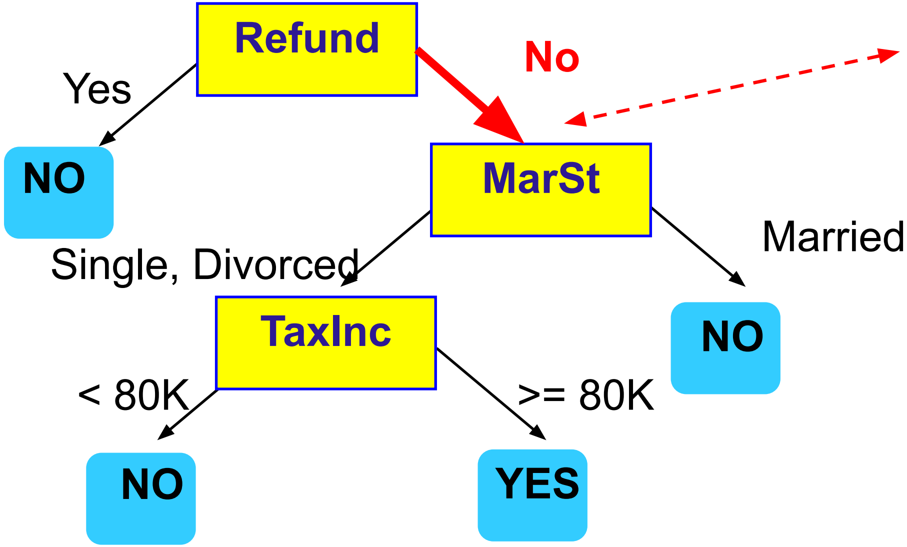
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

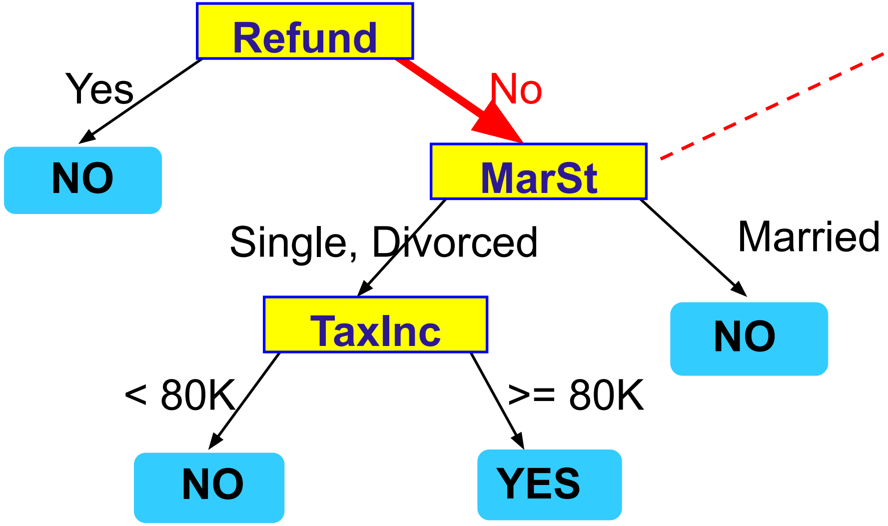
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

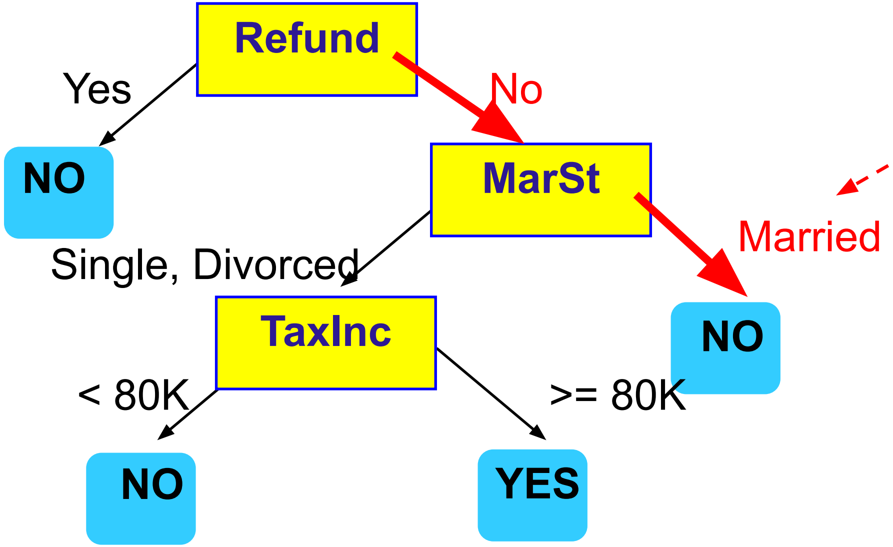
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

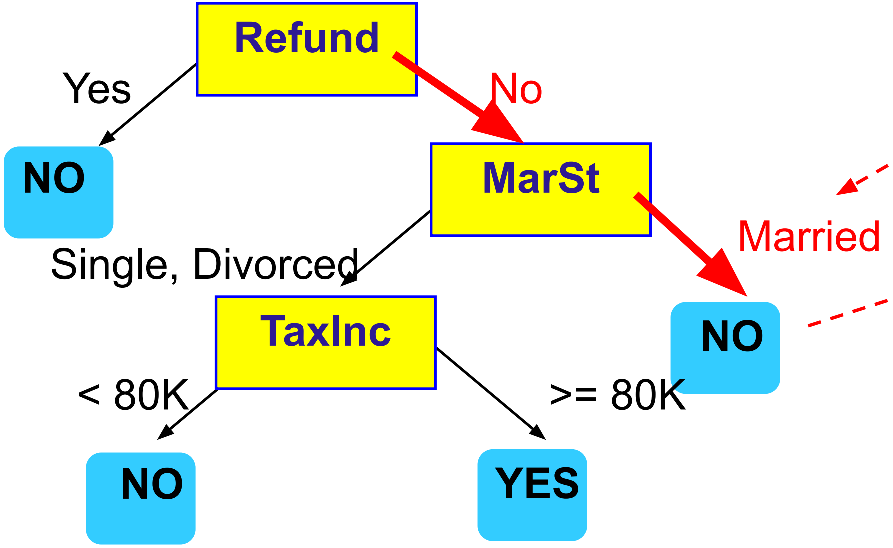
Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to “No”

Constructing decision trees (ID3)

- Top down in a recursive **divide-and-conquer** fashion
  - **First**:

Constructing decision trees (ID3)

- Top down in a recursive **divide-and-conquer** fashion
  - **First**: an attribute is selected for the root node and a branch is created for each possible attribute value
  - **Then**:

Constructing decision trees (ID3)

- Top down in a recursive **divide-and-conquer** fashion
  - **First**: an attribute is selected for the root node and a branch is created for each possible attribute value
  - **Then**: the instances are split into subsets (one for each branch extending from the node)
  - **Finally**:

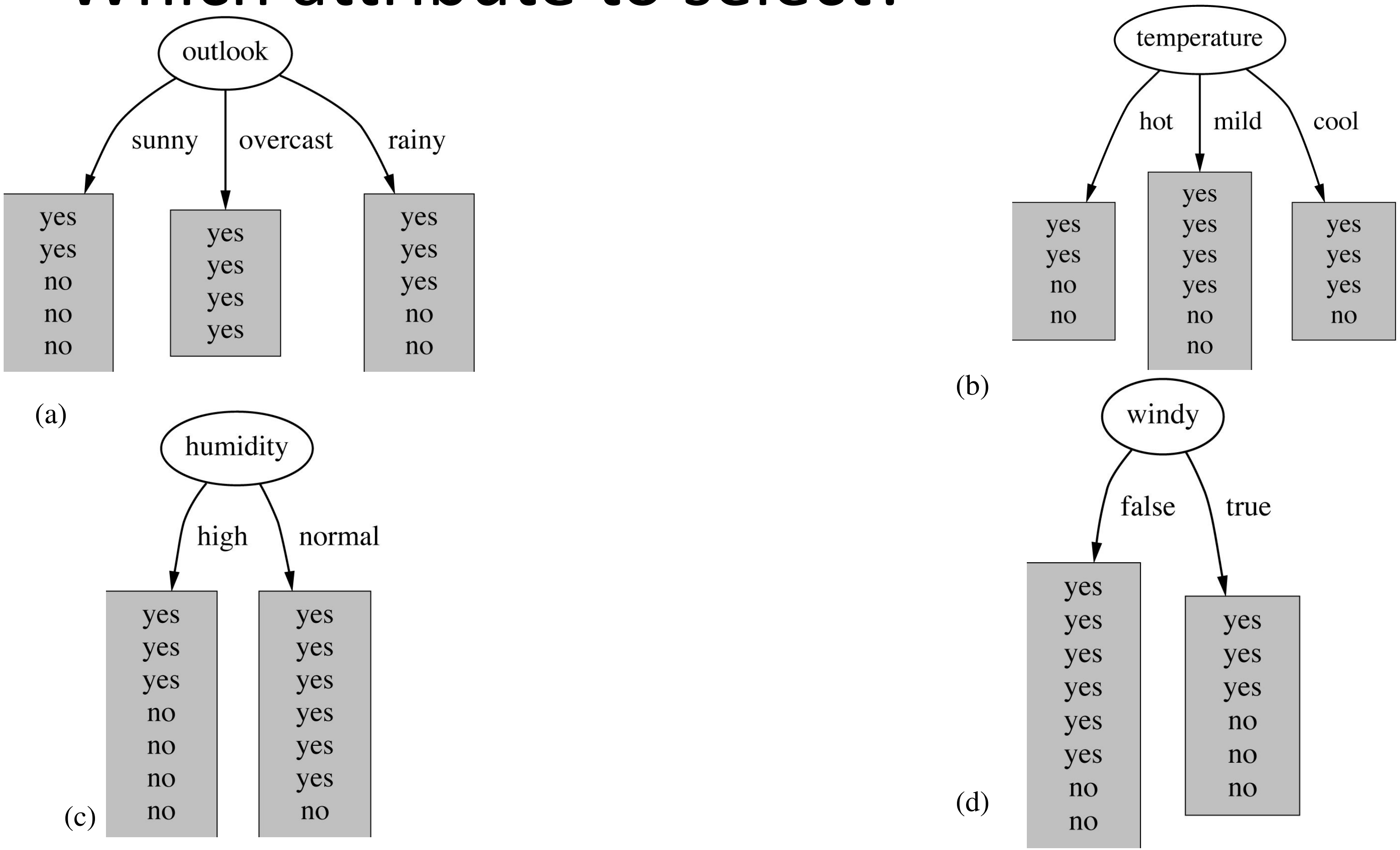
Constructing decision trees (ID3)

- Top down in a recursive **divide-and-conquer** fashion
  - **First**: an attribute is selected for the root node and a branch is created for each possible attribute value
  - **Then**: the instances are split into subsets (one for each branch extending from the node)
  - **Finally**: the same procedure is repeated recursively for each branch, using only instances that reach the branch
- Process stops if all instances have the same class

New Example: Playing soccer

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

Which attribute to select?



WE NEED…A criterion for attribute selection

- Which is the best attribute?

WE NEED…A criterion for attribute selection

- Which is the best attribute?

- The one which will result in the smallest tree
  - **Heuristic**: **choose the attribute that produces the "purest" nodes**

WE NEED…A criterion for attribute selection

- Which is the best attribute?

- The one which will result in the smallest tree
  - **Heuristic: choose the attribute that produces the "purest" nodes**

- Popular impurity criterion: **entropy** of nodes
  - **Lower the entropy, purer the node.**

Entropy

- H(X) = E(I(X))   **Expected** value of the **information** in X

Entropy

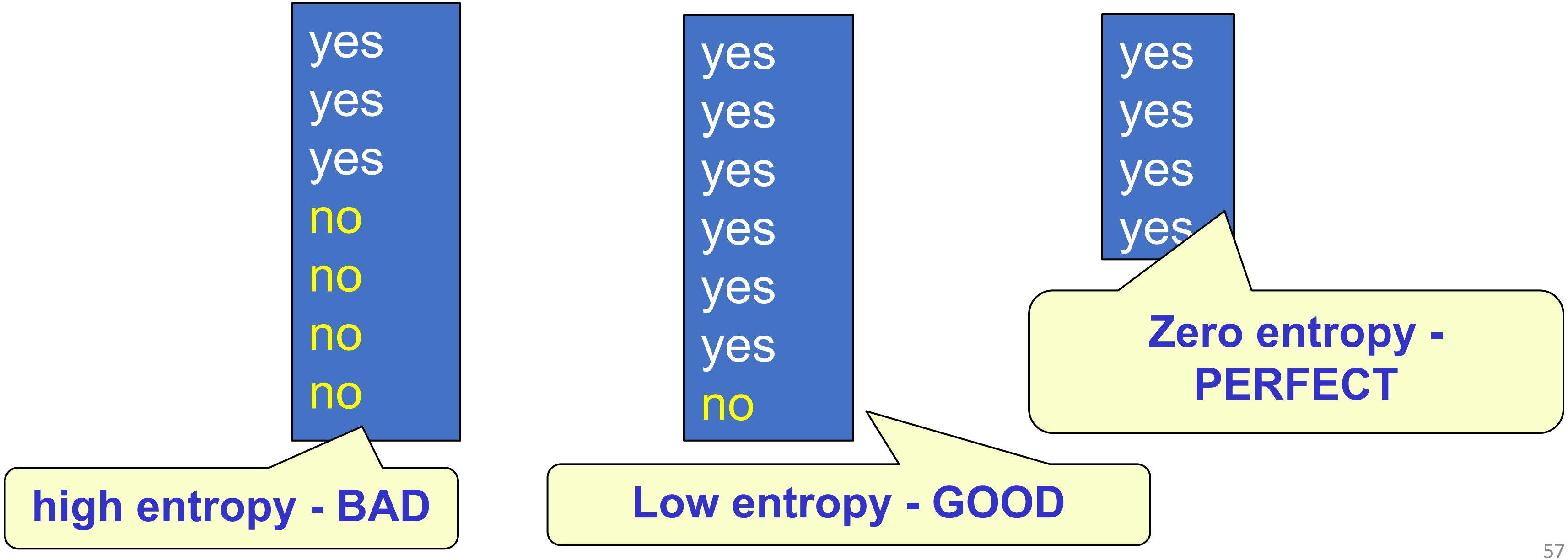
- H(X) = E(I(X)) **Expected** value of the **information** in X
- Expected value: $\mathrm{E(f(X))} = \sum_i P(x_i) * f(x_i)$
- Information: $I(x_i) = -\log_2 P(x_i)$
- Entropy: $H(X) = E(I(X)) = \sum_i P(x_i) I(x_i) = -\sum_i P(x_i) \log_2 P(x_i)$

- **Strategy: choose attribute that results in lowest entropy of the children nodes.**

Why low entropy?

Measuring Purity with Entropy

- Entropy is a measure of **disorder.** Aka **amount of information**.
  - The **higher** the entropy, the **messier** the bag
  - The **lower** the entropy, the **purer** the bag



$$E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d)$$ where:

d=total # of rows, **a** = # yes, **b** = # no

yes	yes	yes
yes	yes	yes
yes	yes	yes
no	yes	yes
no	yes	
no	yes	
no	no	

$$E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d)$$ where:

d=total # of rows, a = # yes, b = # no

yes	yes	yes
yes	yes	yes
yes	yes	yes
no	yes	yes
no	yes	
no	yes	
no	no	

$E(3/7,4/7) =$

$-(3/7)*\log_2(3/7)-(4/7)*\log_2(4/7) =$ **.985**

$E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d)$ where:

d=total # of rows, a = # yes, b = # no

yes	yes	yes
yes	yes	yes
yes	yes	yes
no	yes	yes
no	yes	
no	yes	
no	no	

$E(3/7,4/7) =$
$-(3/7)*\log_2(3/7)-(4/7)*\log_2(4/7) =$ **.985**

$E(6/7,1/7) =$
$-(6/7)*\log_2(6/7)-(1/7)*\log_2(1/7) =$ **.5917**

$E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d)$ where:

d=total # of rows, a = # yes, b = # no

yes
yes
yes
no
no
no
no

$E(3/7,4/7) =$
$-(3/7)*\log_2(3/7)-(4/7)*\log_2(4/7) =$ **.985**

yes
yes
yes
yes
yes
yes
no

$E(6/7,1/7) =$
$-(6/7)*\log_2(6/7)-(1/7)*\log_2(1/7) =$ **.5917**

yes
yes
yes
yes

$E(4/4,0/4) =$
$-(4/4)*\log_2(4/4)-(0/4)*\log_2(0/4) =$ **0**

$E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d)$ where:

d=total # of rows, a = # yes, b = # no

(0/4)*$\log_2$(0/4) = **0*$\log_2$(0)** is **indeterminate**. We consider it to be 0.

yes
yes
yes
no
no
no
no

$E(3/7,4/7) = -(3/7)*\log_2(3/7)-(4/7)*\log_2(4/7) =$ **.985**

yes
yes
yes
yes
yes
yes
no

$E(6/7,1/7) = -(6/7)*\log_2(6/7)-(1/7)*\log_2(1/7) =$ **.5917**

yes
yes
yes
yes

$E(4/4,0/4) = -(4/4)*\log_2(4/4)-(0/4)*\log_2(0/4) =$ **0**

Entropy Chart

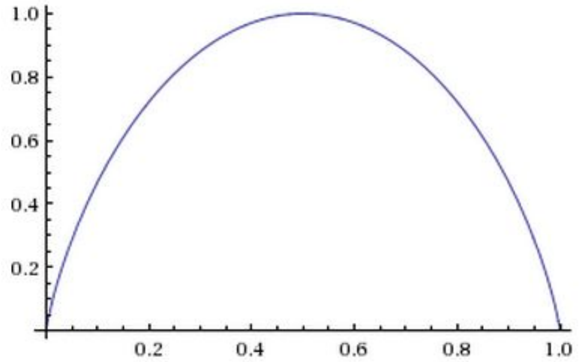
- In the entropy formula: a/d + b/d = 1
- Denote

  a/d with x

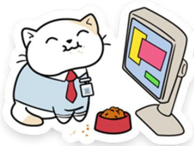
  b/d with 1-x.

Entropy Chart

- In the entropy formula: a/d + b/d = 1
- Denote
  a/d with x
  b/d with 1-x.
- $E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d) =$
- $\mathbf{-x*\log_2(x) - (1-x)*\log_2(1-x)}$



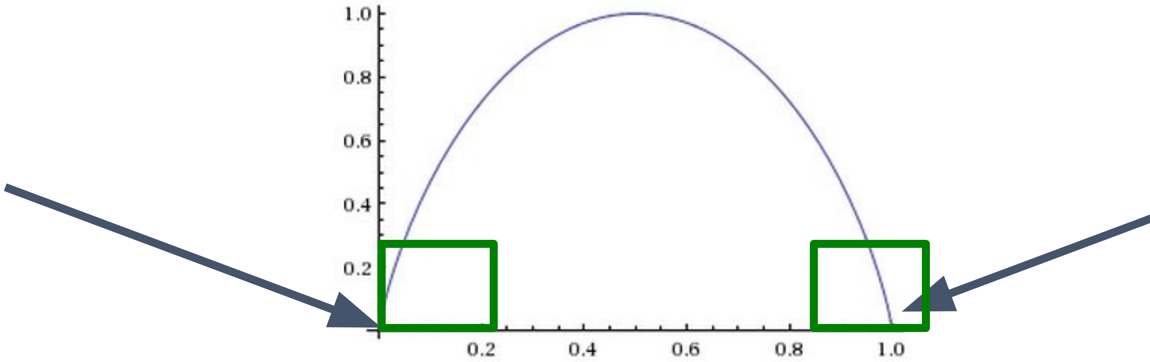
Entropy Chart



- In the entropy formula: a/d + b/d = 1
- Denote

  a/d with x

  b/d with 1-x.
- $E(a/d, b/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d) =$
- $\mathbf{-x*\log_2(x) - (1-x)*\log_2(1-x)}$

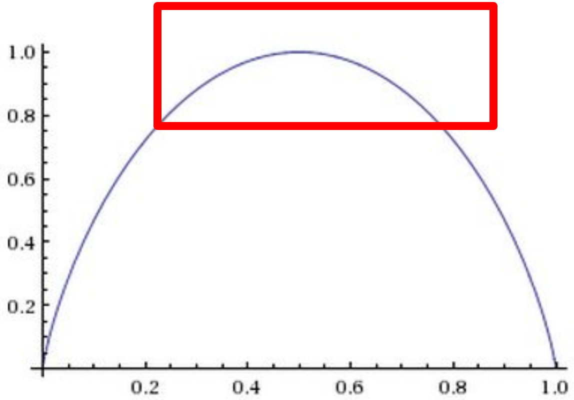


Entropy Chart

- In the entropy formula: a/d + b/d = 1
- Denote

  a/d with x

  b/d with 1-x.
- E(a/d, b/d) = -(a/d)*$\log_2$(a/d) - (b/d)*$\log_2$(b/d) =
- **-x*$\log_2$(x) - (1-x)*$\log_2$(1-x)**



Question to think about:
Can entropy be larger than 1?

Entropy for more than two class values

For three class values:

$E(a/d, b/d, c/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d) - (c/d)*\log_2(c/d)$

$a/d + b/d + c/d = 1$

Entropy for more than two class values

For three class values:

$E(a/d, b/d, c/d) = -(a/d)*\log_2(a/d) - (b/d)*\log_2(b/d) - (c/d)*\log_2(c/d)$

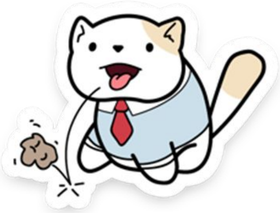
$a/d + b/d + c/d = 1$

For more class values:

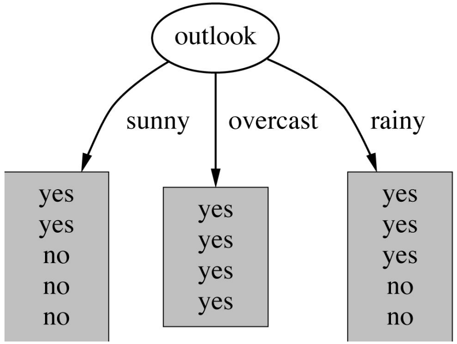
$E(a_1/d,\ldots, a_n/d) = -(a_1/d)*\log_2(a_1/d) - \ldots - (a_n/d)*\log_2(a_n/d)$

$a_1/d + \ldots + a_n/d = 1$

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



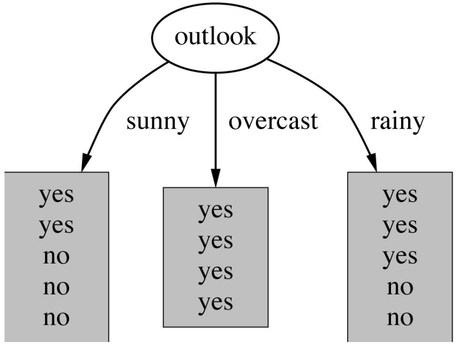
Attribute “Outlook”



Attribute “Outlook”

outlook=sunny

entropy(2/5,3/5) = -2/5*log2(2/5) -3/5*log2(3/5) = .971



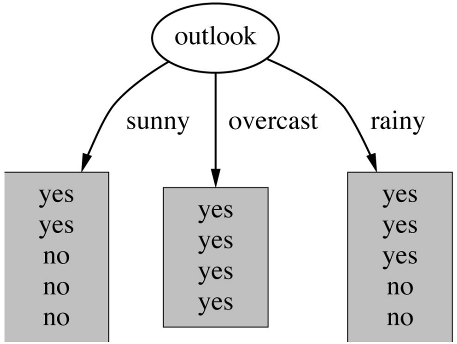
Attribute “Outlook”

outlook=sunny

entropy(2/5,3/5) = -2/5*log2(2/5) -3/5*log2(3/5) = .971

outlook=overcast

entropy(4/4,0/4) = -1*log2(1) -0*log2(0) = 0



Attribute “Outlook”

outlook=sunny

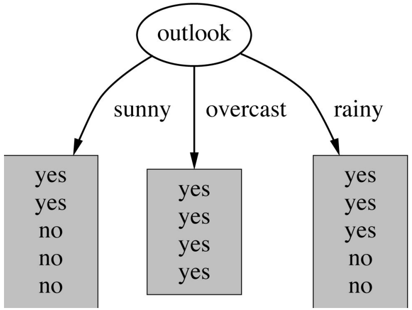
entropy(2/5,3/5) = -2/5*log2(2/5) -3/5*log2(3/5) = .971

outlook=overcast

entropy(4/4,0/4) = -1*log2(1) -0*log2(0) = 0

outlook=rainy

entropy(3/5,2/5) = -3/5*log2(3/5)-2/5*log2(2/5) = .971



Attribute “Outlook”

outlook=sunny

entropy(2/5,3/5) = -2/5*log2(2/5) -3/5*log2(3/5) = .971

outlook=overcast

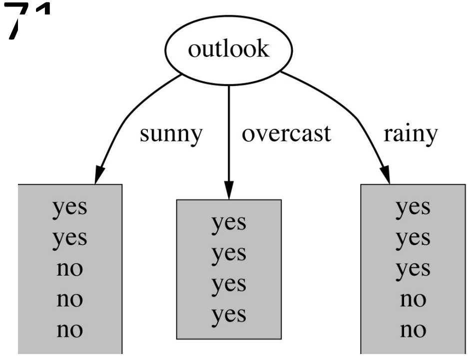
entropy(4/4,0/4) = -1*log2(1) -0*log2(0) = 0

outlook=rainy

entropy(3/5,2/5) = -3/5*log2(3/5)-2/5*log2(2/5) = .971

Expected info:

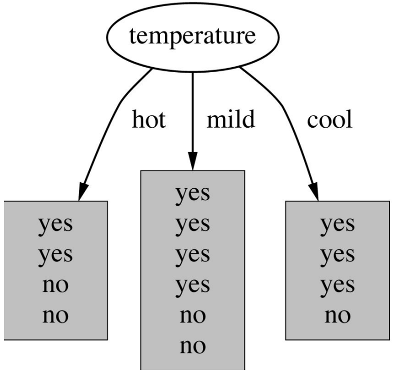
AE = .971*(5/14) + 0*(4/14) + .971*(5/14) = **.693**



Attribute “Temperature”

temperature=hot

entropy(2/4,2/4) = -2/4*log2(2/4) -2/4*log2(2/4) = 1



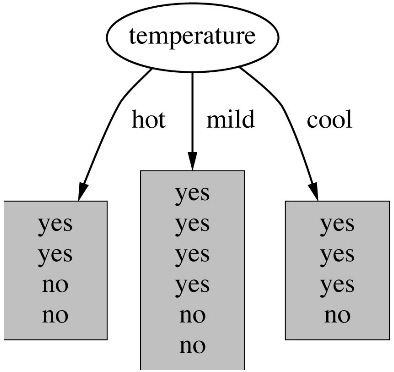
Attribute "Temperature"

temperature=hot

entropy(2/4,2/4) = -2/4*log2(2/4) -2/4*log2(2/4) = 1

temperature=mild

entropy(4/6,2/6) = -4/6*log2(4/6) -2/6*log2(2/6) = .918



Attribute “Temperature”

temperature=hot

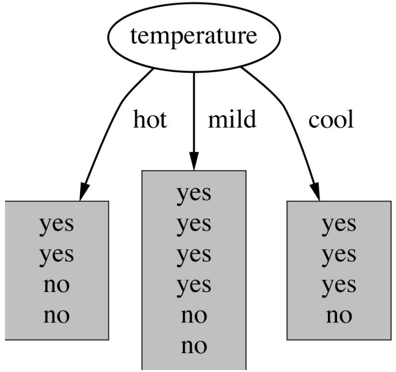
entropy(2/4,2/4) = -2/4*log2(2/4) -2/4*log2(2/4) = 1

temperature=mild

entropy(4/6,2/6) = -4/6*log2(4/6) -2/6*log2(2/6) = .918

temperature=cool

entropy(3/4,1/4) = -3/4*log2(3/4)-1/4*log2(1/4) = .811



Attribute “Temperature”

temperature=hot

entropy(2/4,2/4) = -2/4*log2(2/4) -2/4*log2(2/4) = 1

temperature=mild

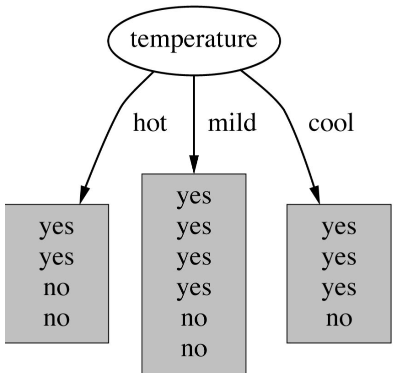
entropy(4/6,2/6) = -4/6*log2(4/6) -2/6*log2(2/6) = .918

temperature=cool

entropy(3/4,1/4) = -3/4*log2(3/4)-1/4*log2(1/4) = .811

Expected info:

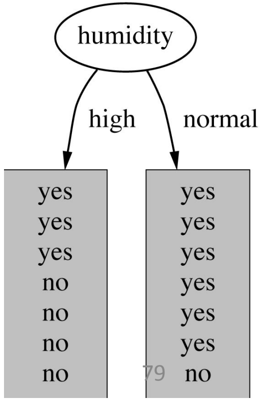
AE = 1*(4/14) + .918*(6/14) + .811*(4/14) = **0.911**



Attribute “Humidity”

humidity=high

entropy(3/7,4/7) = -3/7*log2(3/7) -4/7*log2(4/7) = .985



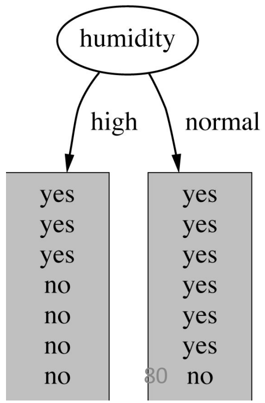
Attribute “Humidity”

humidity=high

entropy(3/7,4/7) = -3/7*log2(3/7) -4/7*log2(4/7) = .985

humidity=normal

entropy(6/7,1/7) = -6/7*log2(6/7) -1/7*log2(1/7) = .592



Attribute “Humidity”

humidity=high

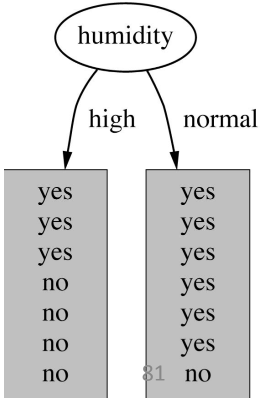
entropy(3/7,4/7) = -3/7*log2(3/7) -4/7*log2(4/7) = .985

humidity=normal

entropy(6/7,1/7) = -6/7*log2(6/7) -1/7*log2(1/7) = .592

Expected info:

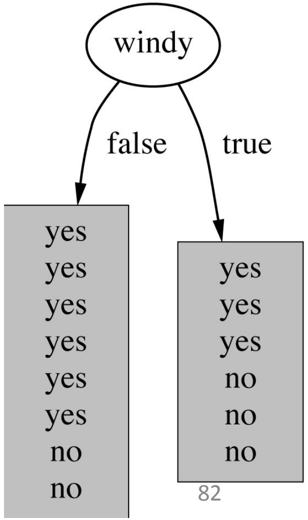
AE = .985*(7/14) + .592*(7/14) = **.789**



Attribute "Windy"

windy=false

entropy(6/8,2/8) = -6/8*log2(6/8) -2/8*log2(2/8) = .811



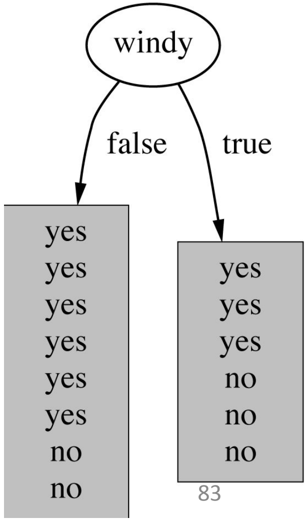
Attribute “Windy”

windy=false

entropy(6/8,2/8) = -6/8*log2(6/8) -2/8*log2(2/8) = .811

windy=true

entropy(3/6,3/6) = -3/6*log2(3/6) -3/6*log2(3/6) = 1



Attribute “Windy”

windy=false

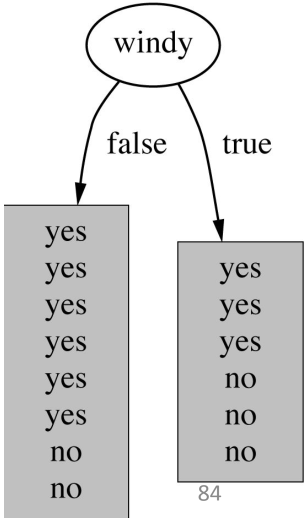
entropy(6/8,2/8) = -6/8*log2(6/8) -2/8*log2(2/8) = .811

windy=true

entropy(3/6,3/6) = -3/6*log2(3/6) -3/6*log2(3/6) = 1

Expected info:

AE = .811*(8/14) + 1*(6/14) = **.892**

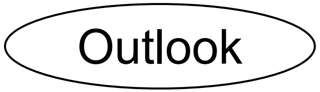


And the winner is...

And the winner is...

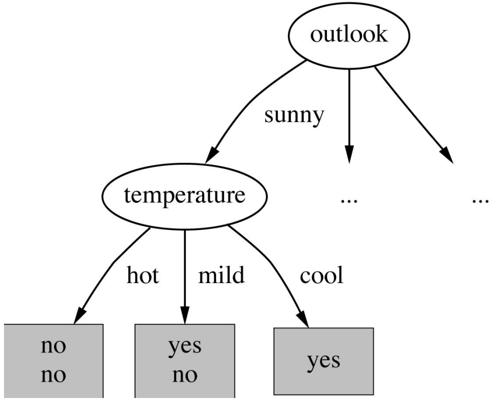
"Outlook"

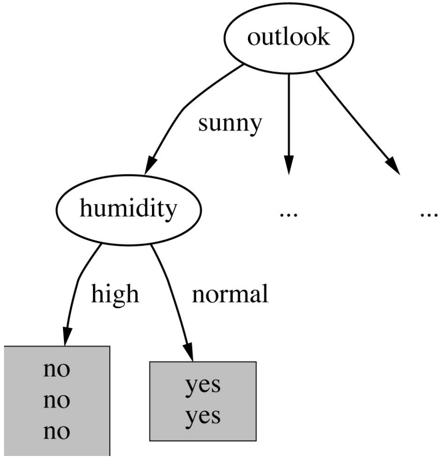
...So, the root will be "Outlook"



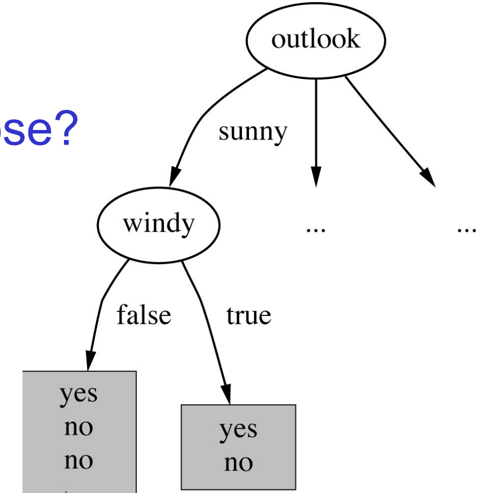
Continuing to split (for Outlook="Sunny")

Outlook	Temp	Humidity	Windy	Play
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Sunny	Mild	Normal	True	Yes





Which one to choose?



Continuing to split (for Outlook="Sunny")

temperature=hot: entropy(2/2,0/2) = 0

temperature=mild: entropy(1/2,1/2) = 1

temperature=cool: entropy(1/1,0/1) = 0, **AE = 0*(2/5) + 1*(2/5) + 0*(1/5) = .4**

humidity=high: entropy(3/3,0/3) = 0

humidity=normal: info(2/2,0/2) = 0, **AE = 0**

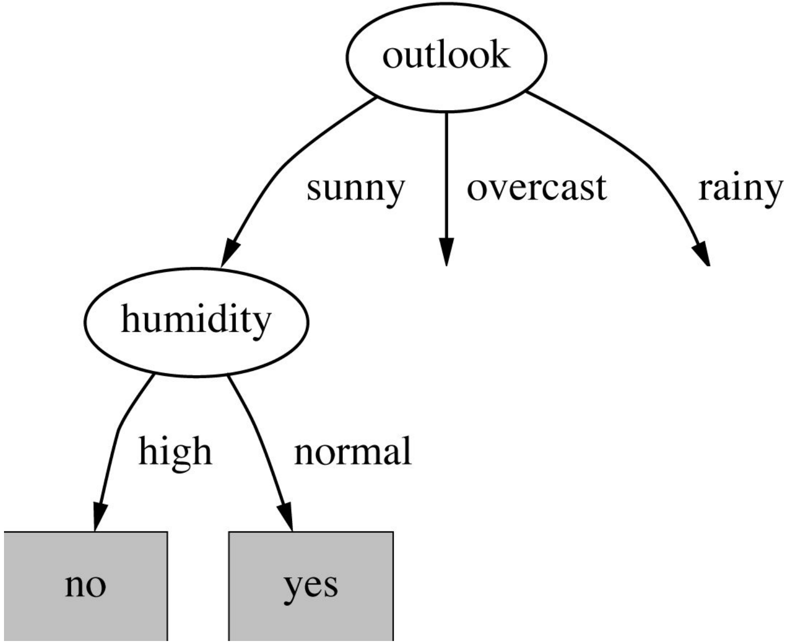
windy=false: entropy(1/3,2/3) = -1/3*log2(1/3) -2/3*log2(2/3) = .918

windy=true: entropy(1/2,1/2) = 1

AE = .918*(3/5) + 1*(2/5) = .951

Winner is "humidity"

Tree so far



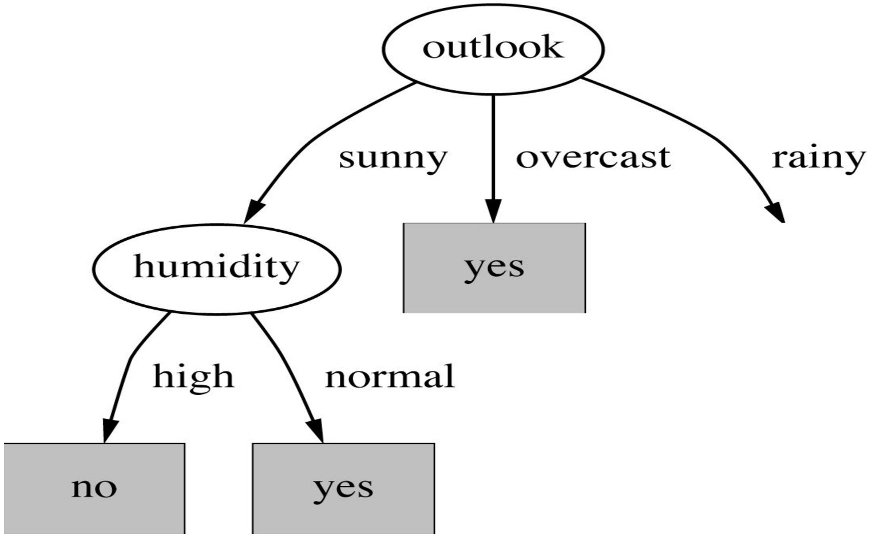
Continuing to split (for Outlook="Overcast")

Outlook	Temp	Humidity	Windy	Play
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

Continuing to split (for Outlook="Overcast")

Outlook	Temp	Humidity	Windy	Play
Overcast	Hot	High	False	Yes
Overcast	Cool	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes

- Nothing to split here, "play" is always "yes".



Tree so far

Continuing to split (for Outlook="Rainy")

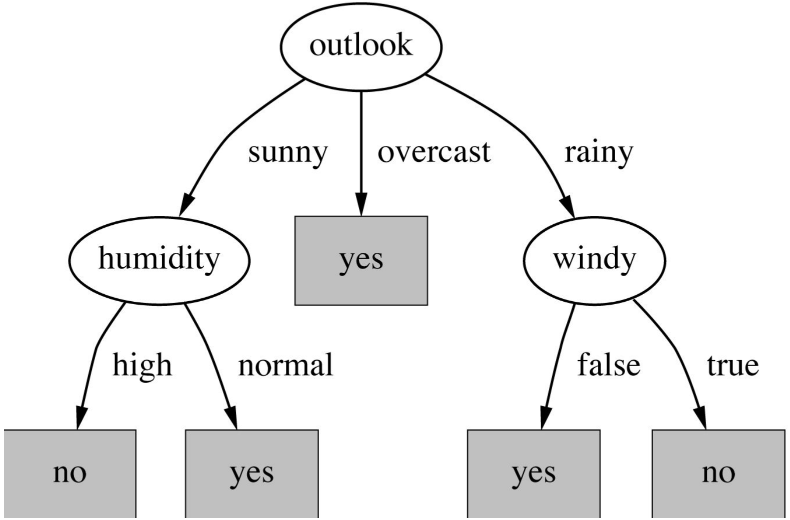
Outlook	Temp	Humidity	Windy	Play
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

Continuing to split (for Outlook="Rainy")

Outlook	Temp	Humidity	Windy	Play
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Rainy	Mild	Normal	False	Yes
Rainy	Mild	High	True	No

- We can easily see that "Windy" is the one to choose. (Why?)

The final decision tree



- **Note**: not all leaves need to be pure; sometimes identical instances have different classes

⇒ Splitting stops when data can't be split any further

Algorithms

- Algorithm described so far is called

  "ID3" - **Iterative Dichotomiser**

developed by **Ross Quinlan** at University of Sydney Australia

Algorithms

- Algorithm described so far is called

  "ID3" - **Iterative Dichotomiser**

developed by **Ross Quinlan** at University of Sydney Australia

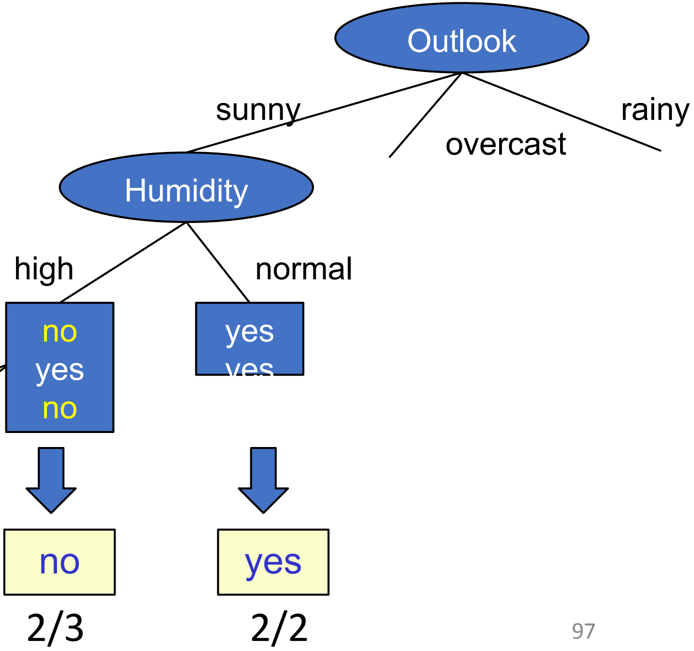
- Led to development of C4.5 (and its commercial version, C5.0, J48 in Java) which deals with
  - noisy data
  - missing values
  - numeric attributes
  - pruning the tree

Noisy data

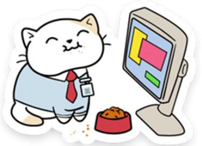
- Not all leaves need to be pure; sometimes identical tuples have different class values
  - Splitting stops when data can't be split any further

ID	Outlook	Temp	Humidity	Windy	Play
1	sunny	hot	high	false	no
2	sunny	hot	high	false	yes
8	sunny	mild	high	false	no
9	sunny	cool	normal	false	yes
11	sunny	mild	normal	true	yes

No chance to split and achieve perfect purity.
All attributes (except ID and Play) have the same values for tuple 1 and 2.



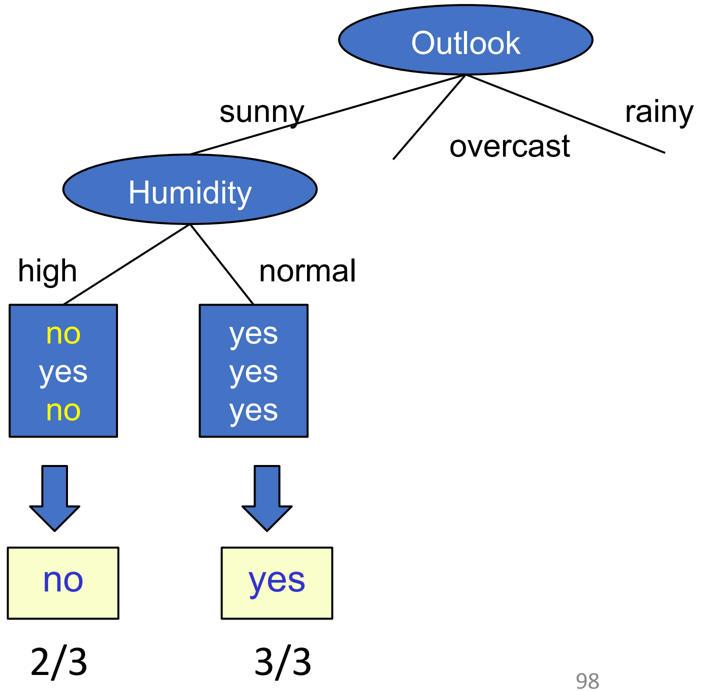
Missing data



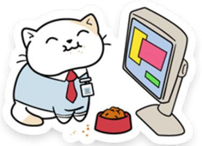
- Sometimes, some attributes of some tuples have missing values

ID	Outlook	Temp	Humidity	Windy	Play
1	sunny	hot	high	false	no
2	sunny	hot	?	false	yes
8	sunny	mild	high	false	no
9	sunny	cool	normal	false	yes
11	sunny	mild	normal	true	yes

Tuple 2 is sent both branches of Humidity.
This is because we don’t know its Humidity value.

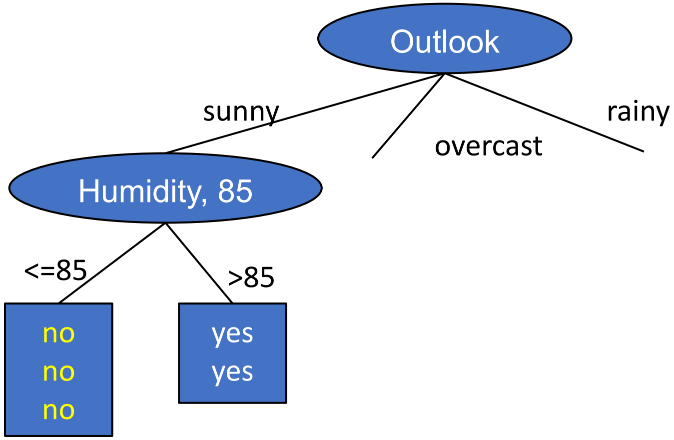


Continuous-valued attributes



- Some attributes can be numeric (continuous).
- No problem, we can have binary splits (≥v, <v), still use Entropy

ID	Outlook	Temp	Humidity	Windy	Play
1	sunny	85	85	false	no
2	sunny	80	90	true	no
3	overcast	83	86	false	yes
4	rainy	70	96	false	yes
5	rainy	68	80	false	yes
6	rainy	65	70	true	no
7	overcast	64	65	true	yes
8	sunny	72	95	false	no
9	sunny	69	70	false	yes
10	rainy	75	80	false	yes
11	sunny	75	70	true	yes
12	overcast	72	90	true	yes
13	overcast	81	75	false	yes
14	rainy	71	91	true	no



ID	Outlook	Temp	Humidity	Windy	Play
1	sunny	69	70	false	no
2	sunny	75	70	true	no
8	sunny	85	85	false	no
9	sunny	80	90	false	yes
11	sunny	72	95	true	yes

Pruning the tree

- Not always a good idea to grow the tree exhaustively
  - Saying goes:
    - “tree will **over fit** the training data”
    - “tree will **not abstract well** to classify new data”

Pruning the tree

- Not always a good idea to grow the tree exhaustively
  - Saying goes:
    - “tree will **over fit** the training data”
    - “tree will **not abstract well** to classify new data”

- Solutions
  - **Pre-pruning**
    - **Stop when error on new data doesn’t go down much**
  - **Post-pruning**
    - Chi-squared test for generalizability. See: http://www.saedsayad.com/decision_tree_overfitting.htm

Decision Trees

- Pros:
  - Easy to visualize/interpret
  - Efficient to use
  - Handles discrete and continuous values
-

Decision Trees

- Pros:
  - Easy to visualize/interpret
  - Efficient to use
  - Handles discrete and continuous values
- Cons:
  - Can create overly-complex trees (pruning helps)
  - Unstable (small changes in data can give very different trees)
  - Finding optimal tree exhaustively is a combinatorial problem (and is thus very expensive to compute and verify)

Applications

- CNNs (convolutional neural nets) are often hard to interpret
  - I.e. it's unclear why the model makes a particular classification decision
- Proposed that decision trees can help us to interpret CNNs
  - https://openaccess.thecvf.com/content_CVPR_2019/papers/Zhang_Interpreting_CNNs_via_Decision_Trees_CVPR_2019_paper.pdf

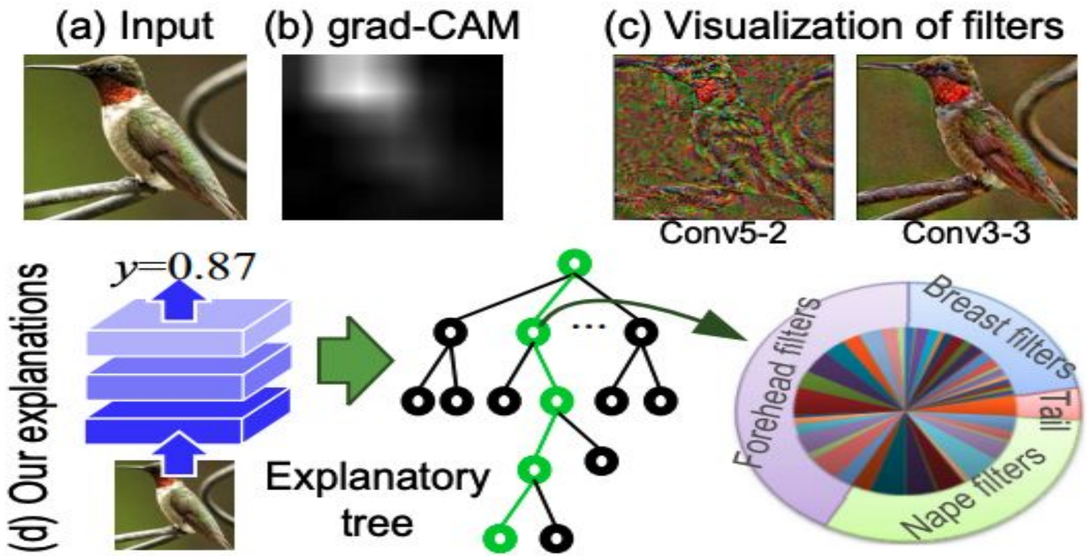
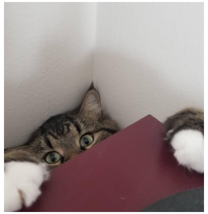


Figure 1. Different types of explanations for CNNs. We compare (d) our task of quantitatively and semantically explaining CNN predictions with previous studies of interpreting CNNs, such as (b) the grad-CAM [26] and (c) CNN visualization [23]. Given an input image (a), we infer a parse tree (green lines) within the decision tree to project neural activations onto clear concepts of object parts. Our method quantitatively explains which filters/parts (in the small/big round) are used for the prediction and how much they contribute to the prediction. We visualize numerical contributions from randomly selected 10% filters for clarity.

Forestssssssssssssssssssss…



Are cool.