Probabilistic and Bayesian Analytics

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: http://www.cs.cmu.edu/~awm/tutorials . Comments and corrections gratefully received.

Andrew W. Moore

Professor

School of Computer Science

Carnegie Mellon University

www.cs.cmu.edu/~awm

awm@cs.cmu.edu

412-268-7599

Probability

- The world is a very uncertain place
- 30 years of Artificial Intelligence and Database research danced around this fact
- And then a few AI researchers decided to use some ideas from the eighteenth century

What we're going to do

- We will review the fundamentals of probability.
- It's really going to be worth it
- In this lecture, you'll see an example of probabilistic analytics in action: Bayes Classifiers

Discrete Random Variables

- A is a Boolean-valued random variable if A denotes an event, and there is some degree of uncertainty as to whether A occurs.
- Examples
- A = The US president in 2023 will be male
- A = You wake up tomorrow with a headache
- A = You have Ebola

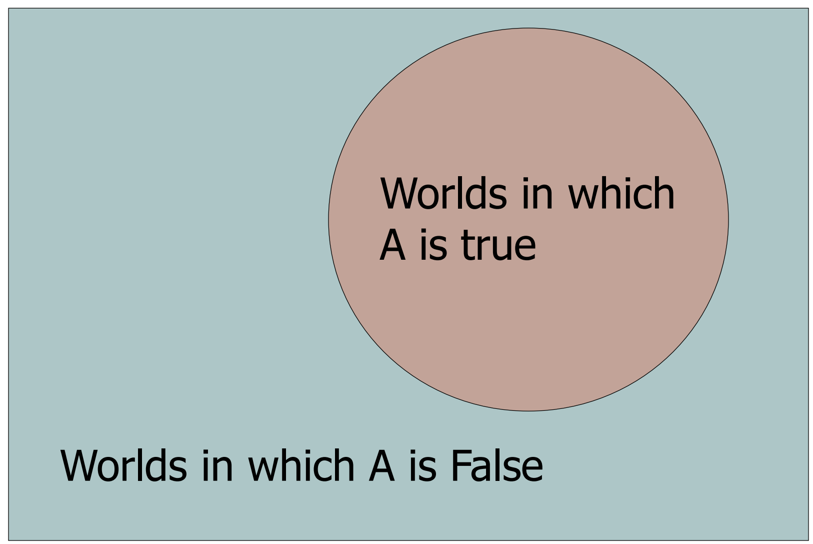
Probabilities

- We write P(A) as “the fraction of possible worlds in which A is true”
- We could at this point spend 2 hours on the philosophy of this.
- But we won’t.

Visualizing A

Event space of all possible worlds

Its area is 1



P(A) = Area of reddish oval


The
Axioms
Of
Probabi
lity

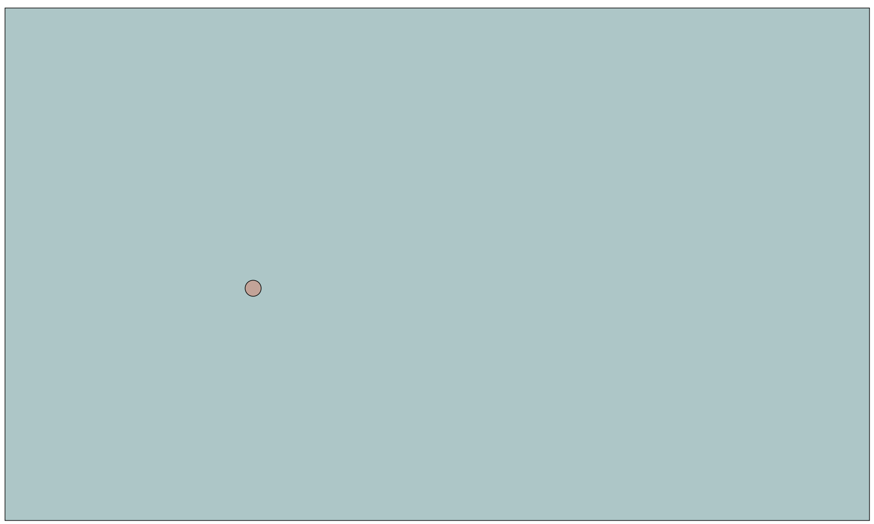
The Axioms of Probability

- 0 <= P(A) <= 1
- P(True) = 1
- P(False) = 0
- P(A or B) = P(A) + P(B) - P(A and B)

Where do these axioms come from? Were they "discovered"?
Answers coming up later.

Interpreting the axioms

- 0 <= P(A) <= 1
- P(True) = 1
- P(False) = 0
- P(A or B) = P(A) + P(B) - P(A and B)

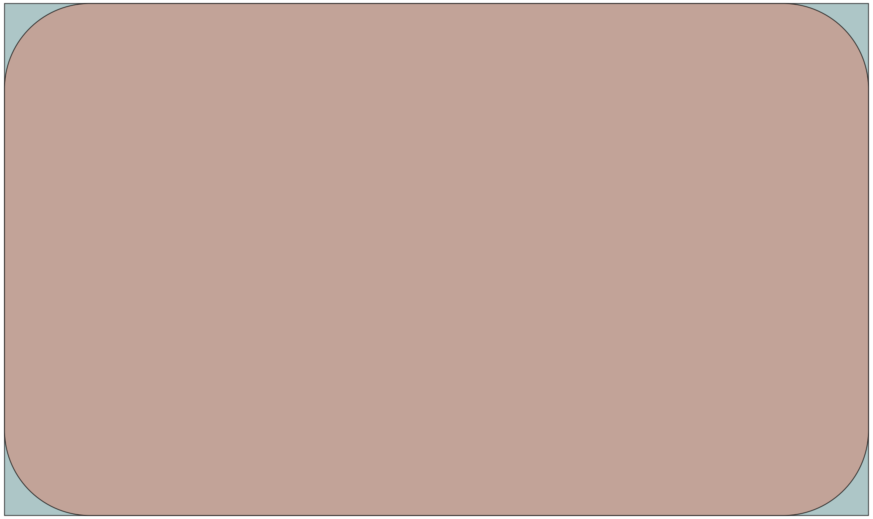


The area of A can't get any smaller than 0

And a zero area would mean no world could ever have A true

Interpreting the axioms

- 0 <= P(A) <= 1
- P(True) = 1
- P(False) = 0
- P(A or B) = P(A) + P(B) - P(A and B)

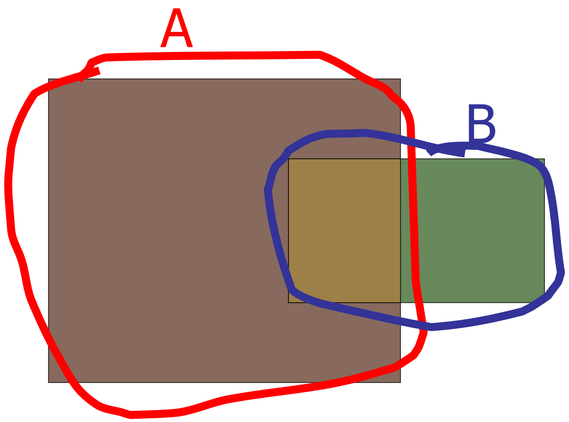


The area of A can't get any bigger than 1

And an area of 1 would mean all worlds will have A true

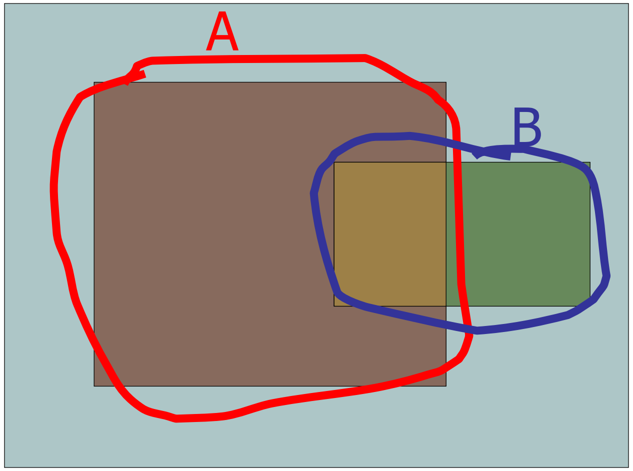
Interpreting the axioms

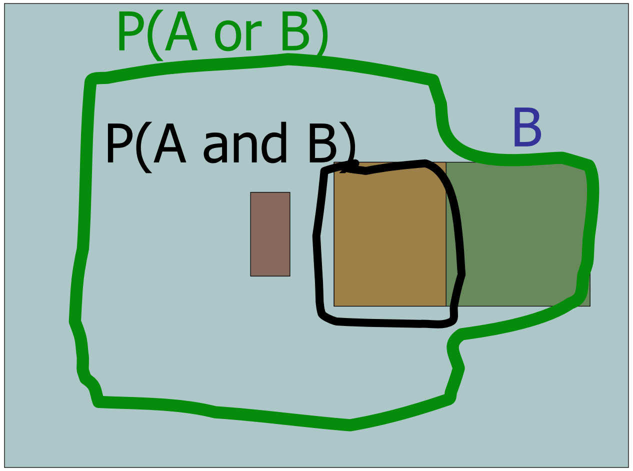
- 0 <= P(A) <= 1
- P(True) = 1
- P(False) = 0
- P(A or B) = P(A) + P(B) - P(A and B)



Interpreting the axioms

- 0 <= P(A) <= 1
- P(True) = 1
- P(False) = 0
- P(A or B) = P(A) + P(B) - P(A and B)





Simple addition and subtraction

These Axioms are Not to be Trifled With

- There have been attempts to do different methodologies for uncertainty
  - Fuzzy Logic
  - Three-valued logic
  - Dempster-Shafer
  - Non-monotonic reasoning

- But the axioms of probability are the only system with this property:

  If you gamble using them you can't be unfairly exploited by an opponent using some other system [di Finetti 1931]

Theorems from the Axioms

- $0 <= P(A) <= 1$, $P(True) = 1$, $P(False) = 0$
- $P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B)$

From these we can prove:

P(not A) = P(~A) = 1-P(A)

- How?

Side Note

- I am inflicting these proofs on you for two reasons:
  1. These kind of manipulations will need to be second nature to you if you use probabilistic analytics in depth
  2. Suffering is good for you

Another important theorem

- $0 <= P(A) <= 1$, $P(True) = 1$, $P(False) = 0$
- $P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B)$

From these we can prove:

$$P(A) = P(A \wedge B) + P(A \wedge \sim B)$$

- How?

Multivalued Random Variables

- Suppose A can take on more than 2 values
- A is a *random variable with arity k* if it can take on exactly one value out of $\{v_1, v_2, .. v_k\}$
- Thus…

$$P(A = v_i \wedge A = v_j) = 0 \text{ if } i \neq j$$
$$P(A = v_1 \vee A = v_2 \vee A = v_k) = 1$$

An easy fact about Multivalued Random Variables:

- Using the axioms of probability…

  0 <= P(A) <= 1, P(True) = 1, P(False) = 0

  P(A or B) = P(A) + P(B) - P(A and B)

- And assuming that A obeys…

$$P(A = v_i \wedge A = v_j) = 0 \text{ if } i \neq j$$

$$P(A = v_1 \vee A = v_2 \vee A = v_k) = 1$$

- It's easy to prove that

$$P(A = v_1 \vee A = v_2 \vee A = v_i) = \sum_{j=1}^{i} P(A = v_j)$$

An easy fact about Multivalued Random Variables:

- Using the axioms of probability…

  0 <= P(A) <= 1, P(True) = 1, P(False) = 0

  P(A or B) = P(A) + P(B) - P(A and B)

- And assuming that A obeys…

$$P(A = v_i \wedge A = v_j) = 0 \text{ if } i \neq j$$

$$P(A = v_1 \vee A = v_2 \vee A = v_k) = 1$$

- It's easy to prove that

$$P(A = v_1 \vee A = v_2 \vee A = v_i) = \sum_{j=1}^{i} P(A = v_j)$$

- And thus we can prove

$$\sum_{j=1}^{k} P(A = v_j) = 1$$

Another fact about Multivalued Random Variables:

- Using the axioms of probability…

  0 <= P(A) <= 1, P(True) = 1, P(False) = 0

  P(A or B) = P(A) + P(B) - P(A and B)

- And assuming that A obeys…

$$P(A=v_i \wedge A=v_j)=0 \text{ if } i \neq j$$

$$P(A=v_1 \vee A=v_2 \vee A=v_k)=1$$

- It's easy to prove that

$$P(B \wedge [A=v_1 \vee A=v_2 \vee A=v_i]) = \sum_{j=1}^{i} P(B \wedge A=v_j)$$

Another fact about Multivalued Random Variables:

- Using the axioms of probability…

  0 <= P(A) <= 1, P(True) = 1, P(False) = 0

  P(A or B) = P(A) + P(B) - P(A and B)

- And assuming that A obeys…

$$P(A = v_i \wedge A = v_j) = 0 \text{ if } i \neq j$$

$$P(A = v_1 \vee A = v_2 \vee A = v_k) = 1$$

- It's easy to prove that

$$P(B \wedge [A = v_1 \vee A = v_2 \vee A = v_i]) = \sum_{j=1}^{i} P(B \wedge A = v_j)$$

- And thus we can prove

$$P(B) = \sum_{j=1}^{k} P(B \wedge A = v_j)$$

Elementary Probability in Pictures

- $P(\sim A) + P(A) = 1$

Elementary Probability in Pictures

- P(B) = P(B ^ A) + P(B ^ ~A)

Elementary Probability in Pictures

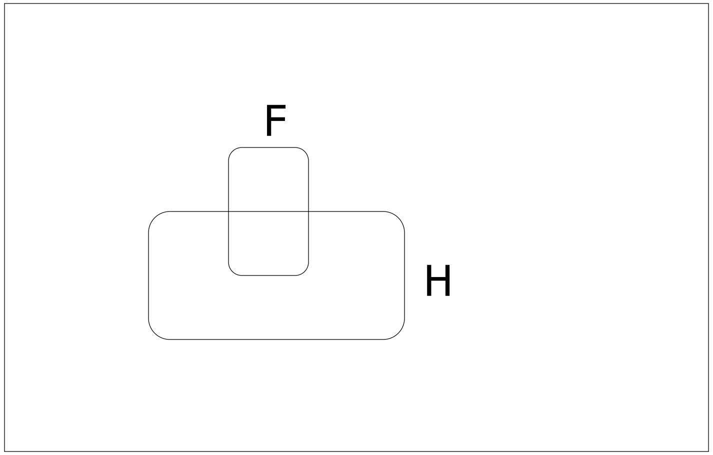
$$\sum_{j=1}^{k} P(A = v_j) = 1$$

Elementary Probability in Pictures

$$P(B) = \sum_{j=1}^{k} P(B \wedge A = v_j)$$

Conditional Probability

- P(A|B) = Fraction of worlds in which B is true that also have A true

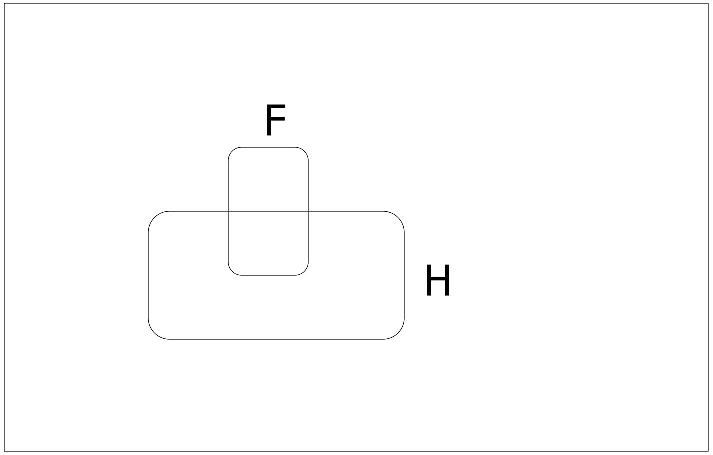


H = "Have a headache"
F = "Coming down with Flu"

P(H) = 1/10
P(F) = 1/40
P(H|F) = 1/2

"Headaches are rare and flu is rarer, but if you're coming down with 'flu there's a 50-50 chance you'll have a headache."

Conditional Probability



H = "Have a headache"
F = "Coming down with Flu"

P(H) = 1/10
P(F) = 1/40
P(H|F) = 1/2

P(H|F) = Fraction of flu-inflicted worlds in which you have a headache

$$= \frac{\text{\#worlds with flu and headache}}{\text{\#worlds with flu}}$$

$$= \frac{\text{Area of "H and F" region}}{\text{Area of "F" region}}$$

$$= \frac{P(H \wedge F)}{P(F)}$$

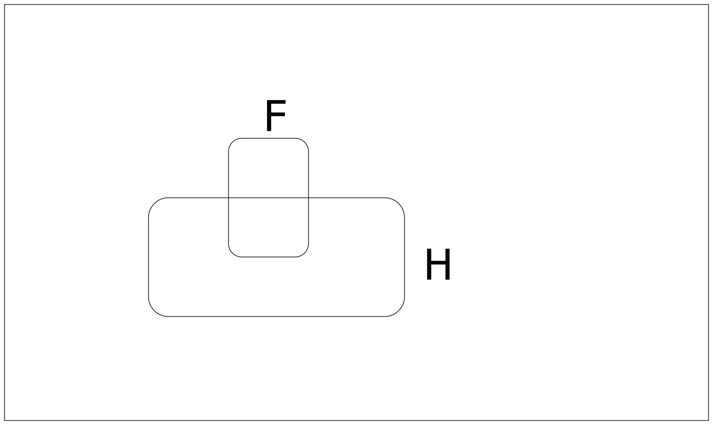
Definition of Conditional Probability

$$P(A|B) = \frac{P(A \wedge B)}{P(B)}$$

Corollary: The Chain Rule

$$P(A \wedge B) = P(A|B)\, P(B)$$

Probabilistic Inference



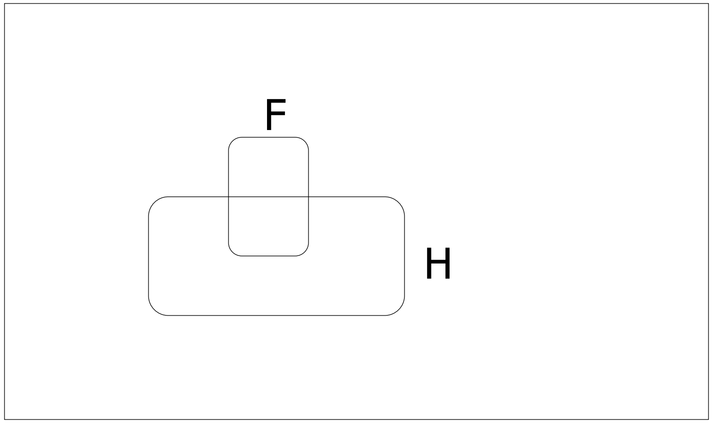
H = “Have a headache”
F = “Coming down with Flu”

P(H) = 1/10
P(F) = 1/40
P(H|F) = 1/2

One day you wake up with a headache. You think: “Drat! 50% of flus are associated with headaches so I must have a 50-50 chance of coming down with flu”

Is this reasoning good?

Probabilistic Inference



H = "Have a headache"
F = "Coming down with Flu"

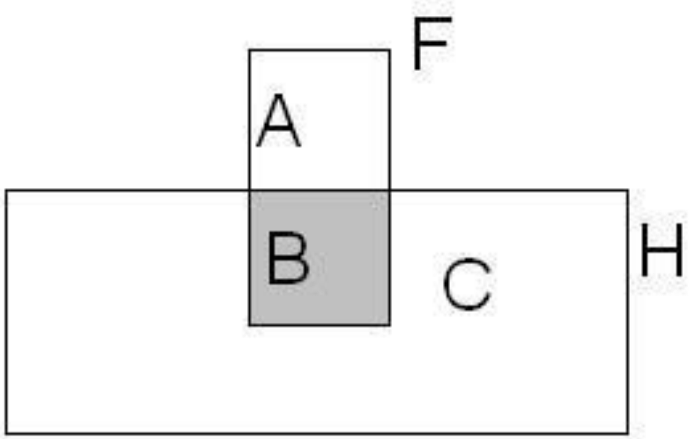
P(H) = 1/10
P(F) = 1/40
P(H|F) = 1/2

P(F ^ H) = …

P(F|H) = …

Another way to understand the intuition

Thanks to Jahanzeb Sherwani for contributing this explanation:



Let's say we have P(F), P(H), and P(H|F), like in the example in class.

Areawise, P(F) = A + B,   P(H) = B + C,

Also, $P(H|F) = \frac{B}{A + B}$

Thus, to get the opposite conditional probability, ie, P(F|H), we need to figure out $\frac{B}{B + C}$

Since we know B / (A+B), we can get B / (B+C) by multiplying by (A+B) and dividing by (B+C). But since we already calculated, A+B = P(F), and B+C = P(H), so we are actually multiplying by P(F) and dividing by P(H). Which is Bayes Rule:

$$P(F|H) = P(H|F) * \frac{P(F)}{P(H)}$$

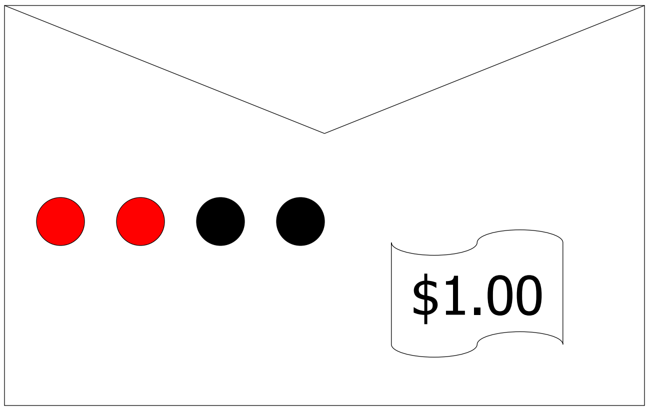
What we just did…

$$P(B|A) = \frac{P(A \wedge B)}{P(A)} = \frac{P(A|B)\,P(B)}{P(A)}$$

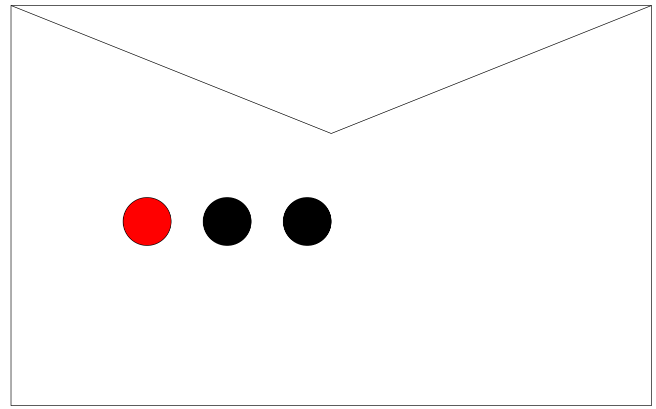
This is Bayes Rule

Bayes, Thomas (1763) An essay towards solving a problem in the doctrine of chances. *Philosophical Transactions of the Royal Society of London,* **53:370-418**

Using Bayes Rule to Gamble



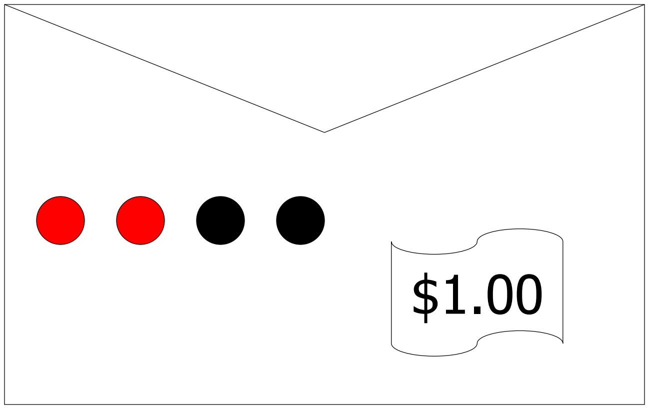
The “Win” envelope has a dollar and four beads in it

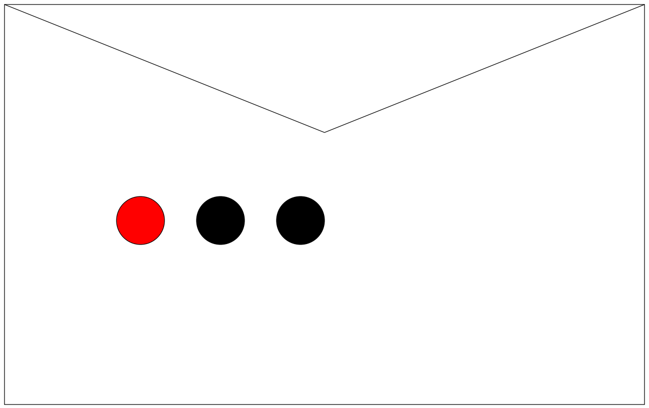


The “Lose” envelope has three beads and no money

Trivial question: someone draws an envelope at random and offers to sell it to you. How much should you pay?

Using Bayes Rule to Gamble





The "Win" envelope has a dollar and four beads in it

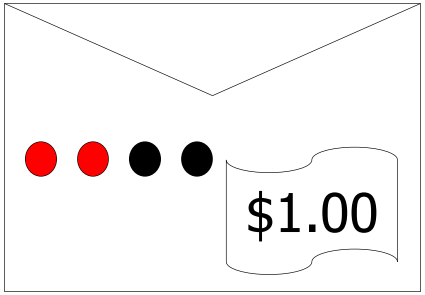
The "Lose" envelope has three beads and no money

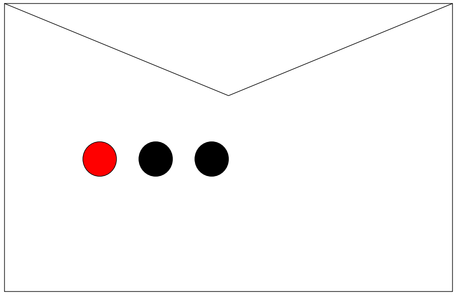
Interesting question: before deciding, you are allowed to see one bead drawn from the envelope.

Suppose it's black: How much should you pay?

Suppose it's red: How much should you pay?

Calculation…





More General Forms of Bayes Rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|\sim A)P(\sim A)}$$

$$P(A|B \wedge X) = \frac{P(B|A \wedge X)P(A \wedge X)}{P(B \wedge X)}$$

More General Forms of Bayes Rule

$$P(A = v_i | B) = \frac{P(B | A = v_i) P(A = v_i)}{\sum_{k=1}^{n_A} P(B | A = v_k) P(A = v_k)}$$

Useful Easy-to-prove facts

$$P(A \mid B) + P(\neg A \mid B) = 1$$

$$\sum_{k=1}^{n_A} P(A = v_k \mid B) = 1$$

The Joint Distribution

Example: Boolean variables A, B, C

Recipe for making a joint distribution of M variables:

The Joint Distribution

Example: Boolean variables A, B, C

Recipe for making a joint distribution of M variables:

1. Make a truth table listing all combinations of values of your variables (if there are M Boolean variables then the table will have 2^M rows).

A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

The Joint Distribution

Example: Boolean variables A, B, C

A	B	C	Prob
0	0	0	0.30
0	0	1	0.05
0	1	0	0.10
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.25
1	1	1	0.10

Recipe for making a joint distribution of M variables:

1. Make a truth table listing all combinations of values of your variables (if there are M Boolean variables then the table will have 2^M rows).
2. For each combination of values, say how probable it is.

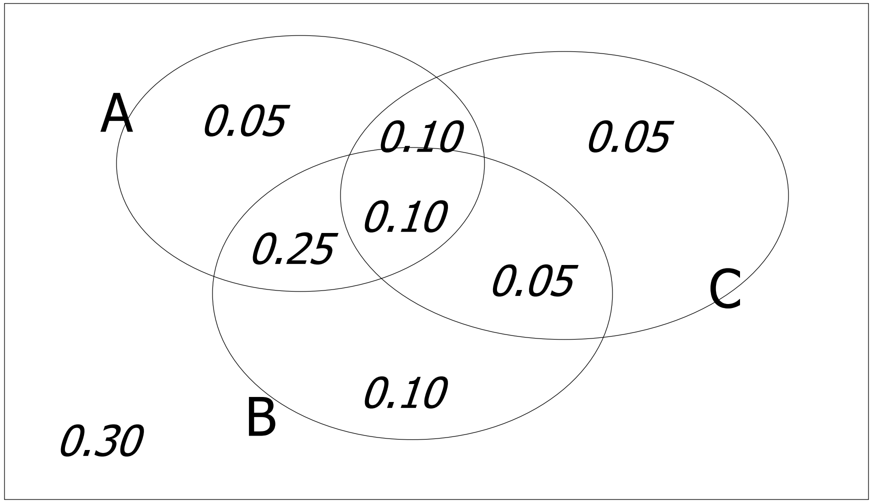
The Joint Distribution

Example: Boolean variables A, B, C

Recipe for making a joint distribution of M variables:

1. Make a truth table listing all combinations of values of your variables (if there are M Boolean variables then the table will have 2^M rows).
2. For each combination of values, say how probable it is.
3. If you subscribe to the axioms of probability, those numbers must sum to 1.

A	B	C	Prob
0	0	0	0.30
0	0	1	0.05
0	1	0	0.10
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	0.25
1	1	1	0.10



Using the Joint

gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

One you have the JD you can ask for the probability of any logical expression involving your attribute

$$P(E) = \sum_{\text{rows matching } E} P(\text{row})$$

Using the Joint

gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

P(Poor Male) = 0.4654

$$P(E) = \sum_{\text{rows matching } E} P(\text{row})$$

Using the Joint

gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

P(Poor) = 0.7604

$$P(E) = \sum_{\text{rows matching } E} P(\text{row})$$

Inference with the Joint

gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

$$P(E_1 \mid E_2) = \frac{P(E_1 \wedge E_2)}{P(E_2)} = \frac{\sum_{\text{rows matching } E_1 \text{ and } E_2} P(\text{row})}{\sum_{\text{rows matching } E_2} P(\text{row})}$$

Inference with the Joint

gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

$$P(E_1 \mid E_2) = \frac{P(E_1 \wedge E_2)}{P(E_2)} = \frac{\sum_{\text{rows matching } E_1 \text{ and } E_2} P(\text{row})}{\sum_{\text{rows matching } E_2} P(\text{row})}$$

P(Male | Poor) = 0.4654 / 0.7604 = 0.612

Inference is a big deal

- I’ve got this evidence. What’s the chance that this conclusion is true?
  - I’ve got a sore neck: how likely am I to have meningitis?
  - I see my lights are out and it’s 9pm. What’s the chance my spouse is already asleep?

Inference is a big deal

- I’ve got this evidence. What’s the chance that this conclusion is true?
  - I’ve got a sore neck: how likely am I to have meningitis?
  - I see my lights are out and it’s 9pm. What’s the chance my spouse is already asleep?

Inference is a big deal

- I’ve got this evidence. What’s the chance that this conclusion is true?
  - I’ve got a sore neck: how likely am I to have meningitis?
  - I see my lights are out and it’s 9pm. What’s the chance my spouse is already asleep?

- There’s a thriving set of industries growing based around Bayesian Inference. Highlights are: Medicine, Pharma, Help Desk Support, Engine Fault Diagnosis

Where do Joint Distributions come from?

- Idea One: Expert Humans
- Idea Two: Simpler probabilistic facts and some algebra

Example: Suppose you knew

P(A) = 0.7	P(C|A^B) = 0.1
	P(C|A^~B) = 0.8
P(B|A) = 0.2	P(C|~A^B) = 0.3
P(B|~A) = 0.1	P(C|~A^~B) = 0.1

Then you can automatically compute the JD using the chain rule

P(A=x ^ B=y ^ C=z) =
P(C=z|A=x^ B=y) P(B=y|A=x) P(A=x)

In another lecture: Bayes Nets, a systematic way to do this.

Where do Joint Distributions come from?

- Idea Three: Learn them from data!

Prepare to see one of the most impressive learning algorithms you'll come across in the entire course….

Learning a joint distribution

Build a JD table for your attributes in which the probabilities are unspecified

A	B	C	Prob
0	0	0	?
0	0	1	?
0	1	0	?
0	1	1	?
1	0	0	?
1	0	1	?
1	1	0	?
1	1	1	?

The fill in each row with

$$\hat{P}(\text{row}) = \frac{\text{records matching row}}{\text{total number of records}}$$

A	B	C	Prob
0	0	0	0.30
0	0	1	0.05
0	1	0	0.10
0	1	1	0.05
1	0	0	0.05
1	0	1	0.10
1	1	0	**0.25**
1	1	1	0.10

Fraction of all records in which A and B are True but C is False

Example of Learning a Joint

- This Joint was obtained by learning from three attributes in the UCI "Adult" Census Database [Kohavi 1995]

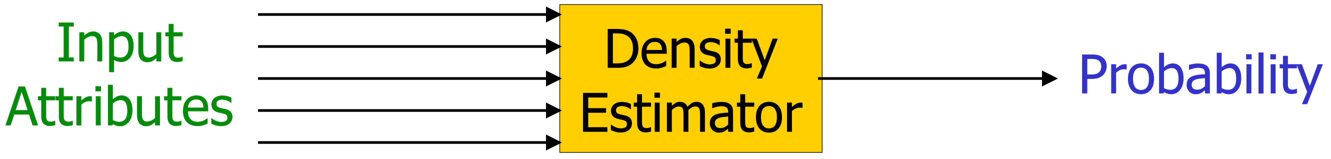
gender	hours_worked	wealth	
Female	v0:40.5-	poor	0.253122
		rich	0.0245895
	v1:40.5+	poor	0.0421768
		rich	0.0116293
Male	v0:40.5-	poor	0.331313
		rich	0.0971295
	v1:40.5+	poor	0.134106
		rich	0.105933

Where are we?

- We have recalled the fundamentals of probability
- We have become content with what JDs are and how to use them
- And we even know how to learn JDs from data.

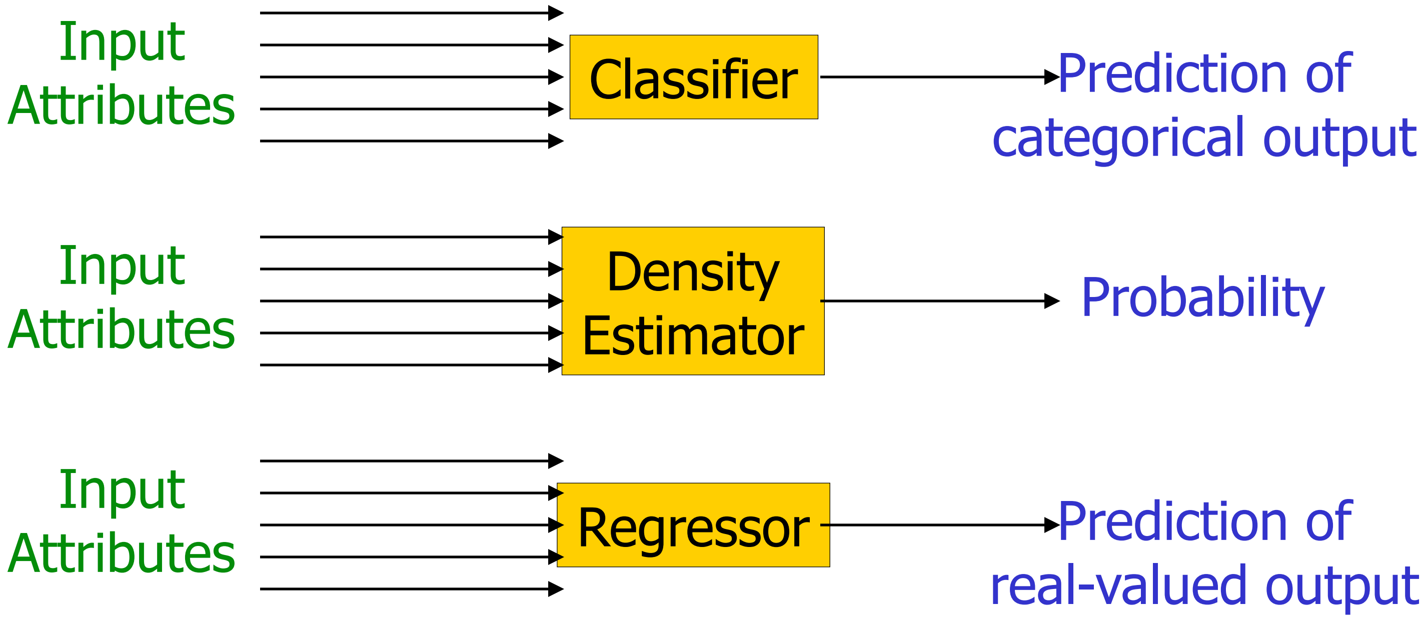
Density Estimation

- Our Joint Distribution learner is our first example of something called Density Estimation
- A Density Estimator learns a mapping from a set of attributes to a Probability



Density Estimation

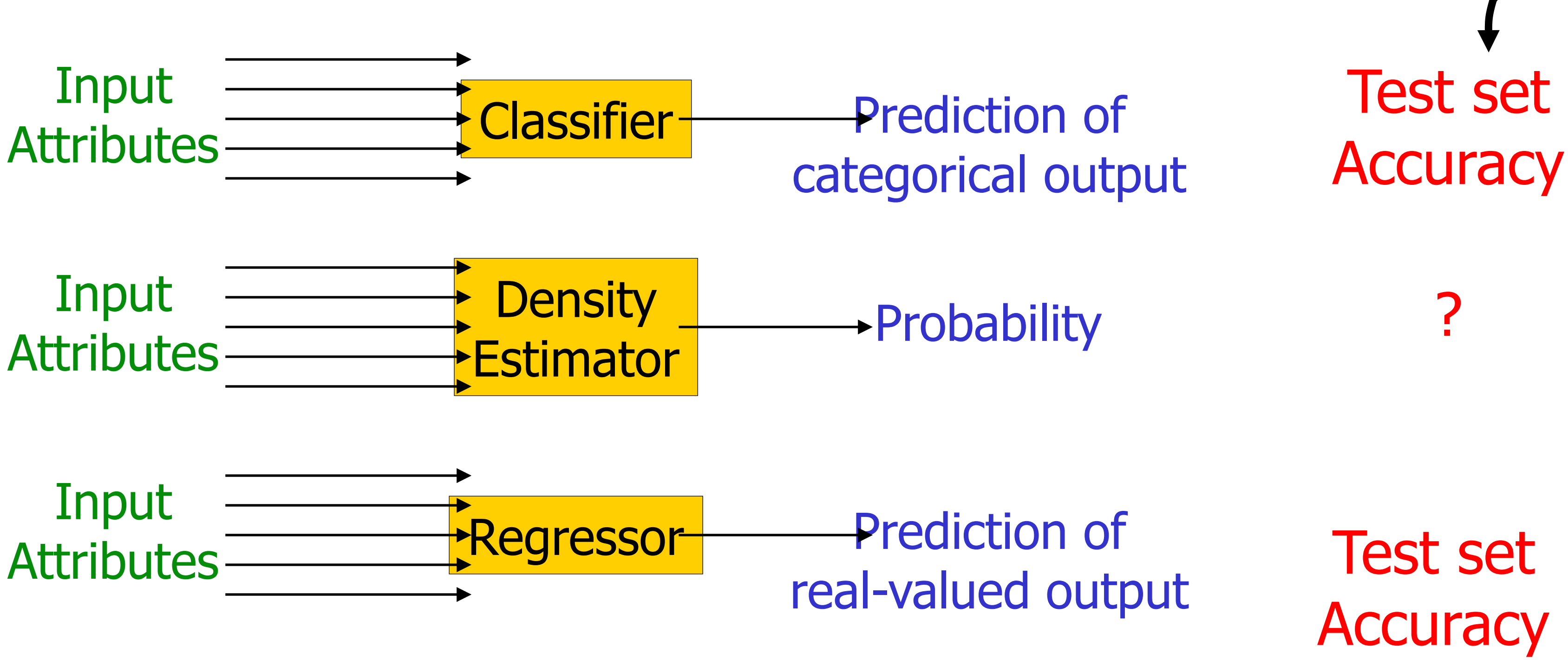
- Compare it against the two other major kinds of models:



Evaluating Density Estimation

Test-set criterion for estimating performance on future data*

** See the Decision Tree or Cross Validation lecture for more detail*



Evaluating a density estimator

- Given a record **x**, a density estimator *M* can tell you how likely the record is:

$$\hat{P}(\mathbf{x}|M)$$

- Given a dataset with *R* records, a density estimator can tell you how likely the dataset is:

(Under the assumption that all records were independently generated from the Density Estimator's JD)

$$\hat{P}(\text{dataset}\,|M) = \hat{P}(\mathbf{x}_1 \wedge \mathbf{x}_2 \ldots \wedge \mathbf{x}_R|M) = \prod_{k=1}^{R} \hat{P}(\mathbf{x}_k|M)$$

A small dataset: Miles Per Gallon

192
Training
Set
Records

mpg	modelyear	maker
good	75to78	asia
bad	70to74	america
bad	75to78	europe
bad	70to74	america
bad	70to74	america
bad	70to74	asia
bad	70to74	asia
bad	75to78	america
:	:	:
:	:	:
:	:	:
bad	70to74	america
good	79to83	america
bad	75to78	america
good	79to83	america
bad	75to78	america
good	79to83	america
good	79to83	america
bad	70to74	america
good	75to78	europe
bad	75to78	europe

From the UCI repository (thanks to Ross Quinlan)

A small dataset: Miles Per Gallon

192 Training Set Records

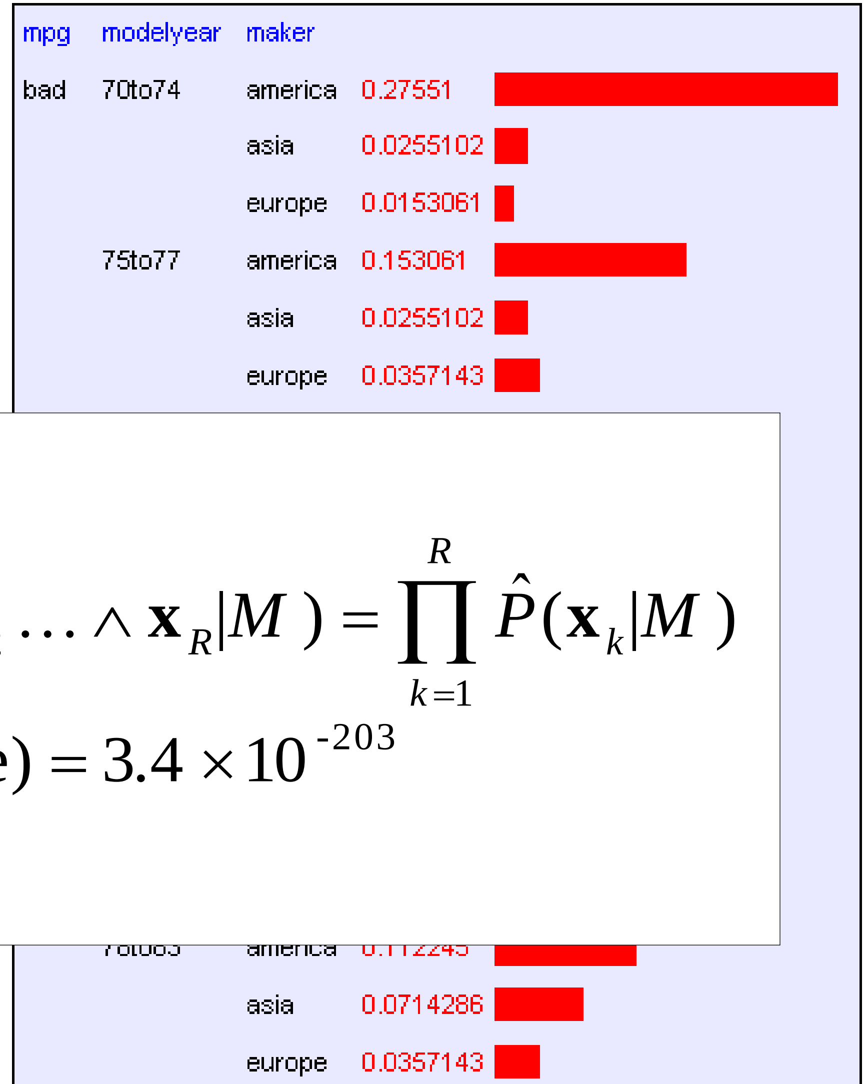
mpg	modelyear	maker
good	75to78	asia
bad	70to74	america
bad	75to78	europe
bad	70to74	america
bad	70to74	america
bad	70to74	asia
bad	70to74	asia
bad	75to78	america
:	:	:
:	:	:
:	:	:
bad	70to74	america
good	79to83	america
bad	75to78	america
good	79to83	america
bad	75to78	america
good	79to83	america
good	79to83	america
bad	70to74	america
good	75to78	europe
bad	75to78	europe

mpg	modelyear	maker	
bad	70to74	america	0.27551
		asia	0.0255102
		europe	0.0153061
	75to77	america	0.153061
		asia	0.0255102
		europe	0.0357143
	78to83	america	0.0561224
		asia	Never
		europe	Never
good	70to74	america	0.0102041
		asia	0.0306122
		europe	0.0459184
	75to77	america	0.0306122
		asia	0.0408163
		europe	0.0357143
	78to83	america	0.112245
		asia	0.0714286
		europe	0.0357143

A small dataset: Miles Per Gallon

192 Training Set

mpg	modelyear	maker
good	75to78	asia
bad	70to74	america
bad	75to78	europe
bad	70to74	america
bad	70to74	america
bad	70to74	asia
bad	70to74	asia



$$\hat{P}(\text{dataset} \mid M) = \hat{P}(\mathbf{x}_1 \wedge \mathbf{x}_2 \ldots \wedge \mathbf{x}_R \mid M) = \prod_{k=1}^{R} \hat{P}(\mathbf{x}_k \mid M)$$

$$= (\text{in this case}) = 3.4 \times 10^{-203}$$

Log Probabilities

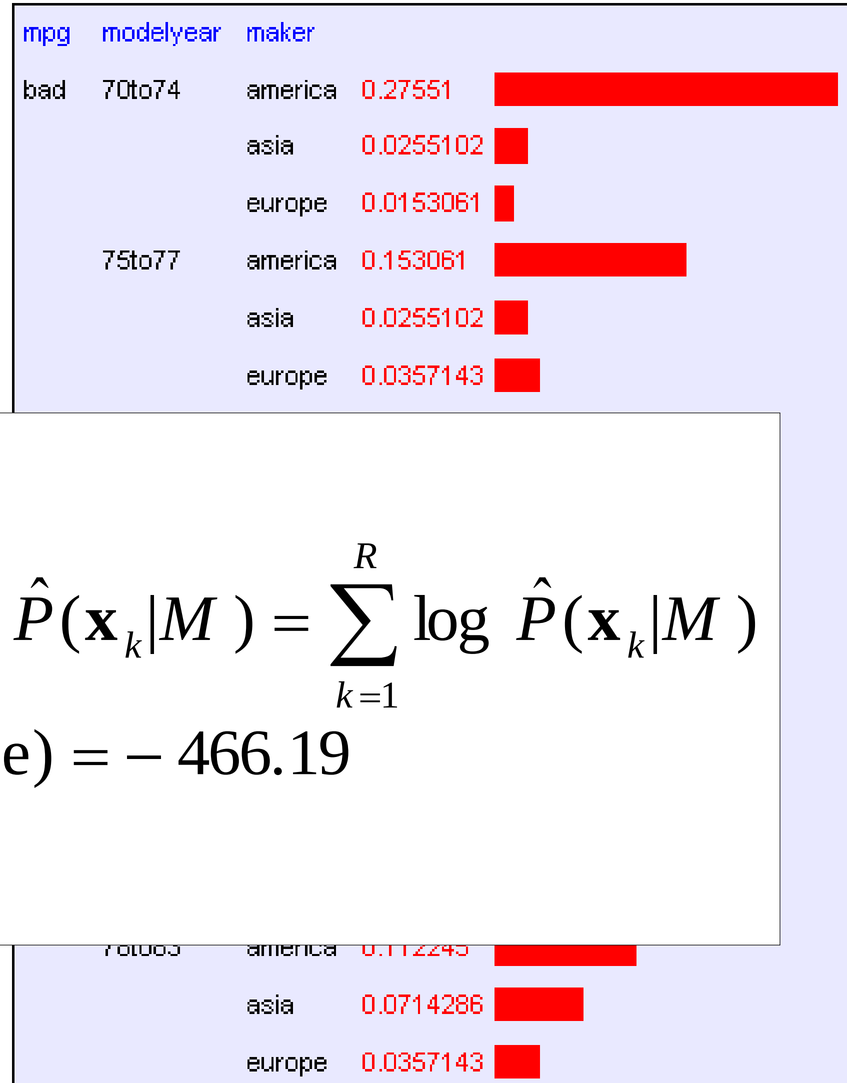
Since probabilities of datasets get so small we usually use log probabilities

$$\log \hat{P}(\text{dataset} \mid M) = \log \prod_{k=1}^{R} \hat{P}(\mathbf{x}_k \mid M) = \sum_{k=1}^{R} \log \hat{P}(\mathbf{x}_k \mid M)$$

A small dataset: Miles Per Gallon

192 Training Set

mpg	modelyear	maker
good	75to78	asia
bad	70to74	america
bad	75to78	europe
bad	70to74	america
bad	70to74	america
bad	70to74	asia
bad	70to74	asia



$$\log \hat{P}(\text{dataset} \mid M) = \log \prod_{k=1}^{R} \hat{P}(\mathbf{x}_k \mid M) = \sum_{k=1}^{R} \log \hat{P}(\mathbf{x}_k \mid M)$$

$$= (\text{in this case}) = -466.19$$

Summary: The Good News

- We have a way to learn a Density Estimator from data.
- Density estimators can do many good things…
  - Can sort the records by probability, and thus spot weird records (anomaly detection)
  - Can do inference: P(E1|E2)

    Automatic Doctor / Help Desk etc
  - Ingredient for Bayes Classifiers (see later)

Summary: The Bad News

- Density estimation by directly learning the joint is trivial, mindless and dangerous

Using a test set

	Set Size	Log likelihood
Training Set	196	-466.1905
Test Set	196	-614.6157

An independent test set with 196 cars has a worse log likelihood

(actually it's a billion quintillion quintillion quintillion quintillion times less likely)

….Density estimators can overfit. And the full joint density estimator is the overfittiest of them all!

Overfitting Density Estimators

If this ever happens, it means there are certain combinations that we learn are impossible

mpg	modelyear	maker	
bad	70to74	america	0.27551
		asia	0.0255102
		europe	0.0153061
	75to77	america	0.153061
		asia	0.0255102
		europe	0.0357143
	78to83	america	0.0561224
		asia	Never
		europe	Never
good	70to74	america	0.0102041

$$\log \hat{P}(\text{testset}\,|M) = \log \prod_{k=1}^{R} \hat{P}(\mathbf{x}_k|M) = \sum_{k=1}^{R} \log \hat{P}(\mathbf{x}_k|M)$$

$$= -\infty \text{ if for any } k \ \hat{P}(\mathbf{x}_k|M) = 0$$

Using a test set

	Set Size	Log likelihood
Training Set	196	-466.1905
Test Set	196	-614.6157

The only reason that our test set didn't score -infinity is that my code is hard-wired to always predict a probability of at least one in 10^{20}

We need Density Estimators that are less prone to overfitting

Naïve Density Estimation

The problem with the Joint Estimator is that it just mirrors the training data.

We need something which generalizes more usefully.

The naïve model generalizes strongly:

Assume that each attribute is distributed independently of any of the other attributes.

Independently Distributed Data

- Let *x[i]* denote the *i*'th field of record *x*.
- The independently distributed assumption says that for any $i, v, u_1\ u_2 \ldots\ u_{i-1}\ u_{i+1} \ldots\ u_M$

$$P(x[i]=v \mid x[1]=u_1, x[2]=u_2, \ldots x[i-1]=u_{i-1}, x[i+1]=u_{i+1}, \ldots x[M]=u_M)$$
$$= P(x[i]=v)$$

- Or in other words, *x[i]* is independent of *{x[1],x[2],..x[i-1], x[i+1],…x[M]}*
- This is often written as

$$x[i] \perp \{x[1], x[2], \ldots x[i-1], x[i+1], \ldots x[M]\}$$

A note about independence

- Assume A and B are Boolean Random Variables. Then

"A and B are independent"

if and only if

$$P(A|B) = P(A)$$

- "A and B are independent" is often notated as

$$A \perp B$$

Independence Theorems

- Assume $P(A|B) = P(A)$
- Then $P(A \wedge B) =$

$= P(A)\, P(B)$

- Assume $P(A|B) = P(A)$
- Then $P(B|A) =$

$= P(B)$

Independence Theorems

- Assume P(A|B) = P(A)
- Then P(~A|B) =

= P(~A)

- Assume P(A|B) = P(A)
- Then P(A|~B) =

= P(A)

Multivalued Independence

For multivalued Random Variables A and B,

$$A \perp B$$

if and only if

$$\forall u, v : P(A = u \mid B = v) = P(A = u)$$

from which you can then prove things like…

$$\forall u, v : P(A = u \wedge B = v) = P(A = u)P(B = v)$$

$$\forall u, v : P(B = v \mid A = v) = P(B = v)$$

Back to Naïve Density Estimation

- Let x[i] denote the i'th field of record x:
- Naïve DE assumes *x[i]* is independent of *{x[1],x[2],..x[i-1], x[i+1],…x[M]}*
- Example:
  - Suppose that each record is generated by randomly shaking a green dice and a red dice
    - Dataset 1: A = red value, B = green value
    - Dataset 2: A = red value, B = sum of values
    - Dataset 3: A = sum of values, B = difference of values
  - Which of these datasets violates the naïve assumption?

Using the Naïve Distribution

- Once you have a Naïve Distribution you can easily compute any row of the joint distribution.
- Suppose *A*, *B*, *C* and *D* are independently distributed. What is $P(A \wedge \sim B \wedge C \wedge \sim D)$?

Using the Naïve Distribution

- Once you have a Naïve Distribution you can easily compute any row of the joint distribution.
- Suppose A, B, C and D are independently distributed. What is P(A^~B^C^~D)?

= P(A|~B^C^~D) P(~B^C^~D)

= P(A) P(~B^C^~D)

= P(A) P(~B|C^~D) P(C^~D)

= P(A) P(~B) P(C^~D)

= P(A) P(~B) P(C|~D) P(~D)

= P(A) P(~B) P(C) P(~D)

Naïve Distribution General Case

- Suppose *x[1], x[2], … x[M]* are independently distributed.

$$P(x[1]=u_1, x[2]=u_2, \ldots x[M]=u_M) = \prod_{k=1}^{M} P(x[k]=u_k)$$

- So if we have a Naïve Distribution we can construct any row of the implied Joint Distribution on demand.
- So we can do any inference
- But how do we learn a Naïve Density Estimator?

Learning a Naïve Density Estimator

$$\hat{P}(x[i]=u) = \frac{\#\,\text{records in which } x[i]=u}{\text{total number of records}}$$

Another trivial learning algorithm!

Contrast

Joint DE	Naïve DE
Can model anything	Can model only very boring distributions
No problem to model "C is a noisy copy of A"	Outside Naïve's scope
Given 100 records and more than 6 Boolean attributes will screw up badly	Given 100 records and 10,000 multivalued attributes will be fine

Empirical Results: "Hopeless"

The "hopeless" dataset consists of 40,000 records and 21 Boolean attributes called a,b,c, … u. Each attribute in each record is generated 50-50 randomly as 0 or 1.

Name	Model	Parameters	LogLike		
Model1	joint	submodel=gauss gausstype=general	-272625	+/-	301.109
Model2	naive	submodel=gauss gausstype=general	-58225.6	+/-	0.554747

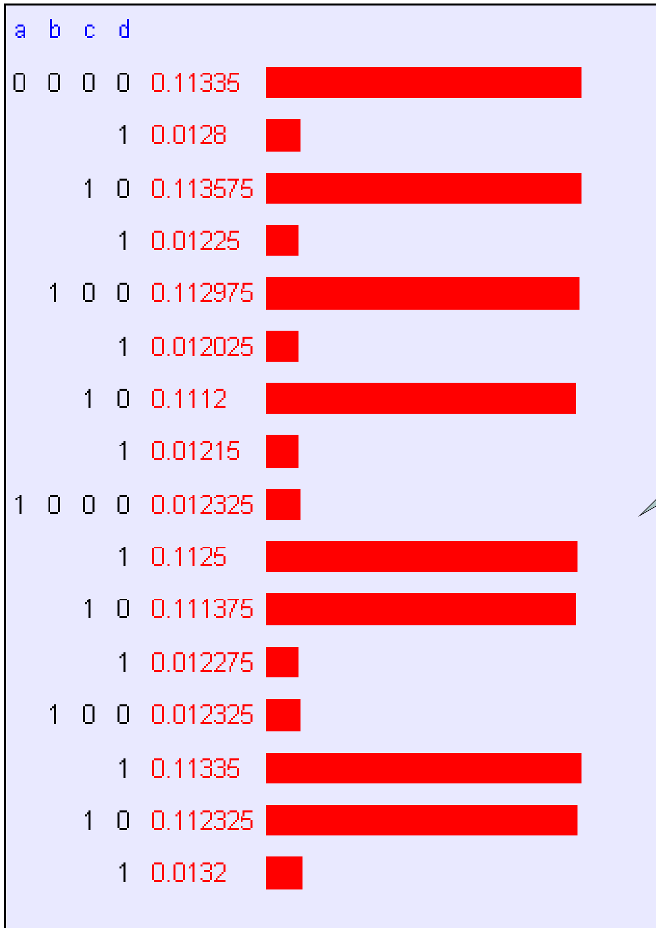
Average test set log probability during 10 folds of k-fold cross-validation*

Described in a future Andrew lecture

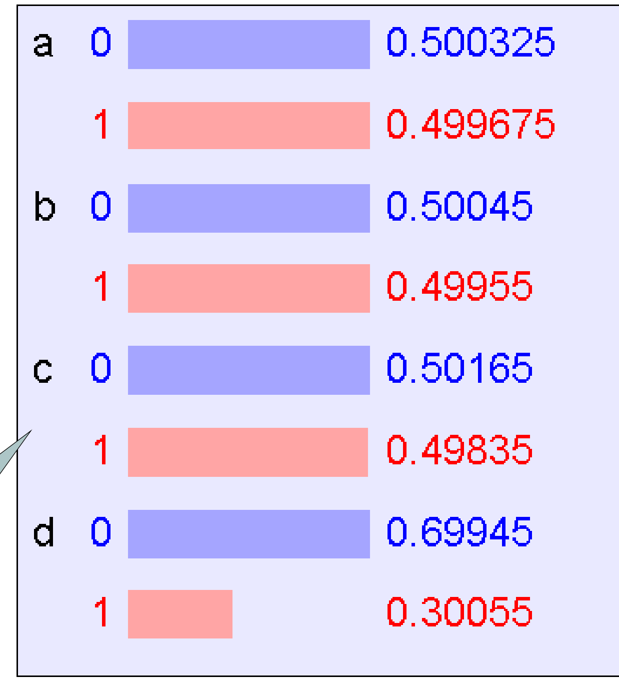
Despite the vast amount of data, "Joint" overfits hopelessly and does much worse

Empirical Results: "Logical"

The "logical" dataset consists of 40,000 records and 4 Boolean attributes called a,b,c,d where a,b,c are generated 50-50 randomly as 0 or 1. D = A^~C, except that in 10% of records it is flipped



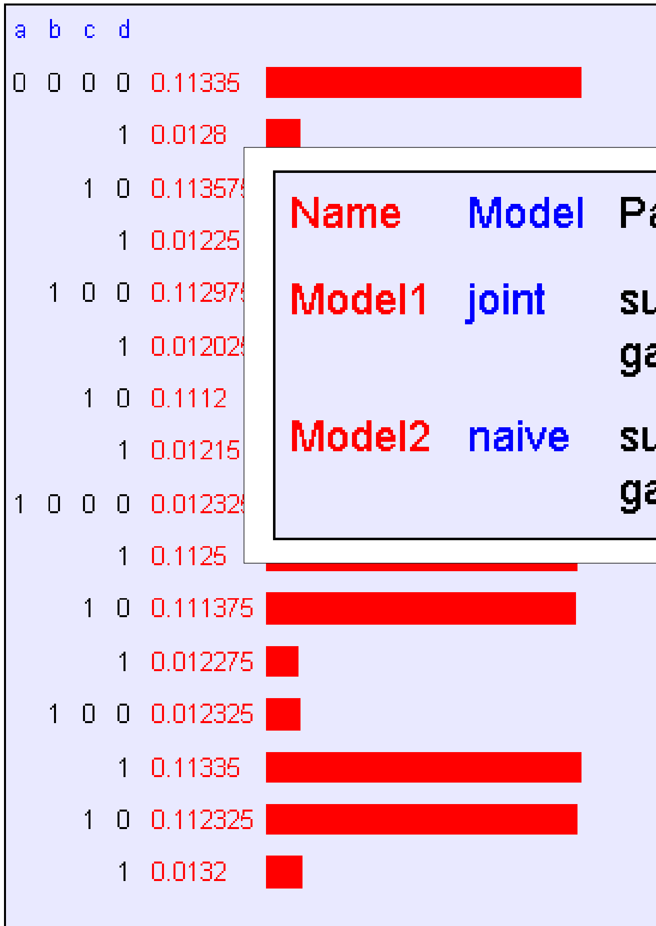
The DE learned by "Joint"



The DE learned by "Naive"

Empirical Results: "Logical"

The "logical" dataset consists of 40,000 records and 4 Boolean attributes called a,b,c,d where a,b,c are generated 50-50 randomly as 0 or 1. D = A^~C, except that in 10% of records it is flipped



Name	Model	Parameters	LogLike
Model1	joint	submodel=gauss gausstype=general	-9613.79 +/- 26.6781
Model2	naive	submodel=gauss gausstype=general	-10763.4 +/- 11.0538

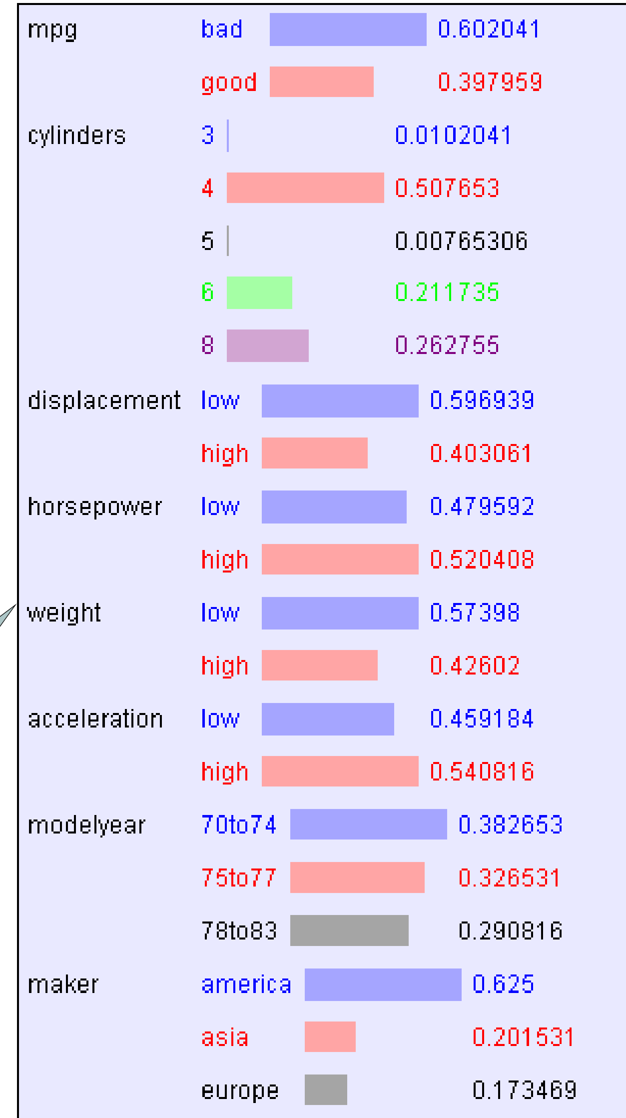
The DE learned by "Naive"

Empirical Results: "MPG"

The "MPG" dataset consists of 392 records and 8 attributes

mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker	
bad	3	low	low	low	low	70to74	america	Never
							asia	0.00255102
							europe	Never
						75to77	Never	
						78to83	Never	
					high	Never		
				high	Never			
			high	low	low	70to74	america	Never
							asia	0.00255102
							europe	Never
						75to77	america	Never
							asia	0.00255102
							europe	Never
						78to83	america	Never
							asia	0.00255102
							europe	Never
					high	Never		
				high	Never			
		high	Never					
	4	low	low	low	low	70to74	america	0.00255102
							asia	Never
							europe	0.00255102
						75to77	Never	
						78to83	Never	
					high	70to74	america	0.0204082
							asia	0.00255102

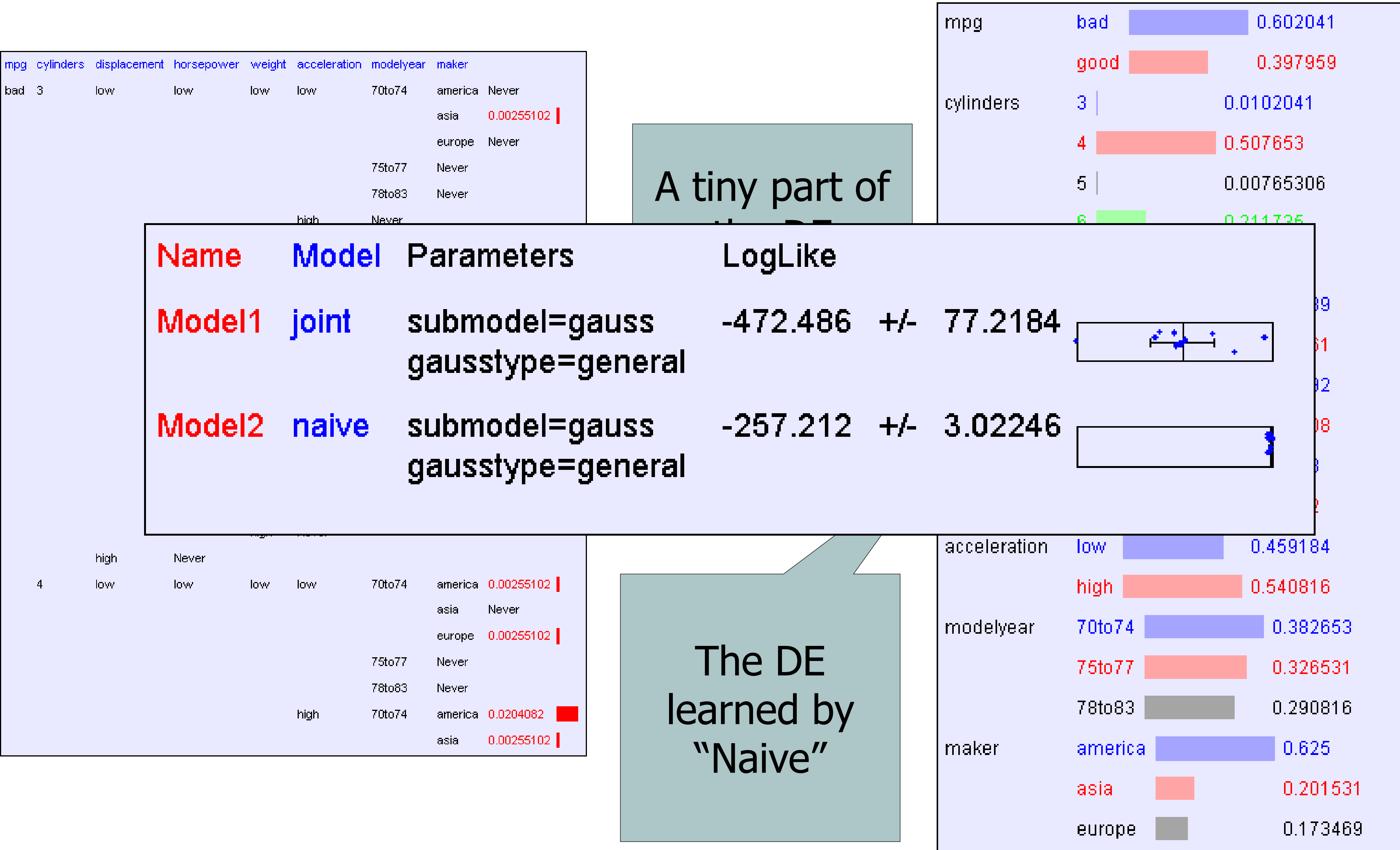
A tiny part of the DE learned by "Joint"



The DE learned by "Naive"

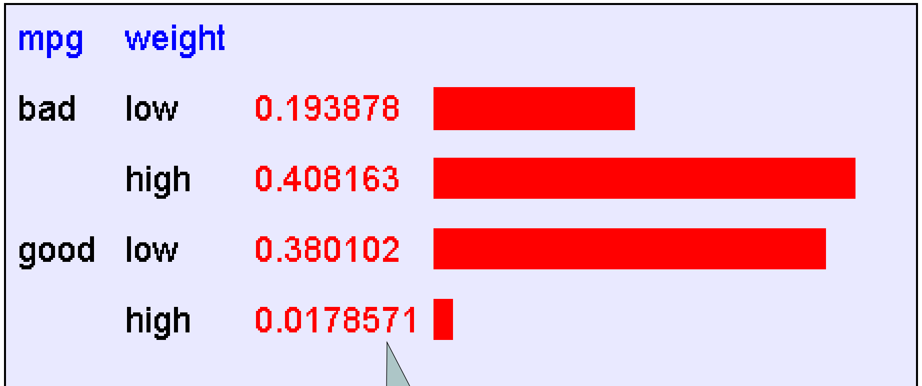
Empirical Results: “MPG”

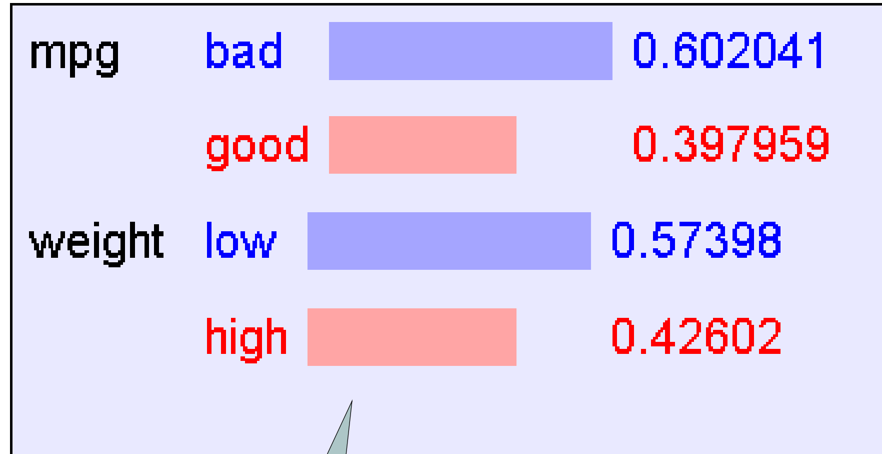
The “MPG” dataset consists of 392 records and 8 attributes



Empirical Results: "Weight vs. MPG"

Suppose we train only from the "Weight" and "MPG" attributes





The DE learned by "Joint"

The DE learned by "Naive"

Empirical Results: "Weight vs. MPG"

Suppose we train only from the "Weight" and "MPG" attributes

mpg	weight		
bad	low	0.193878	
	high	0.408163	
good	l		
	h		

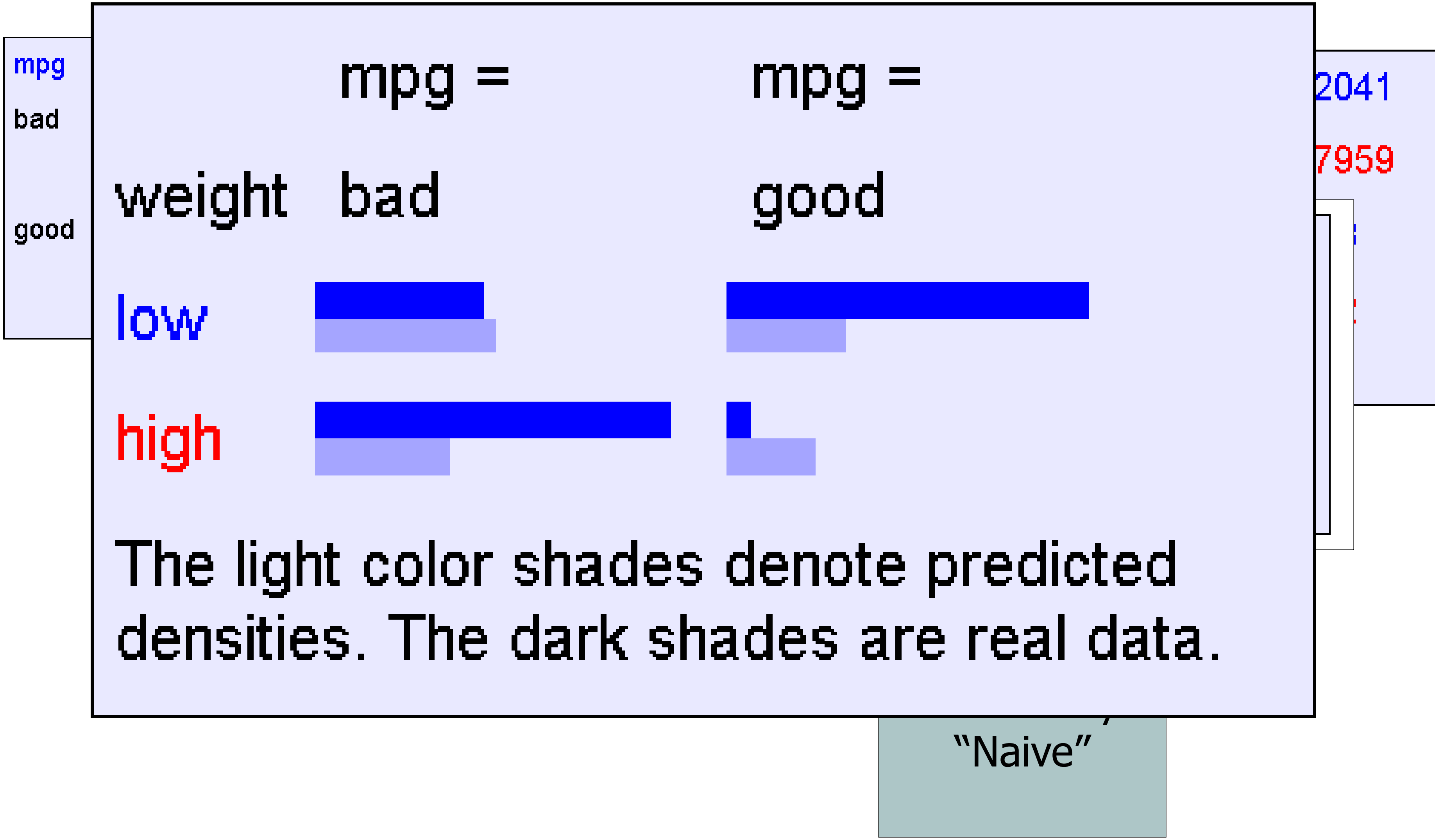
mpg		
bad		0.602041
good		0.397959

Name	Model	Parameters	LogLike
Model1	joint	submodel=gauss gausstype=general	-44.3562 +/- 2.27547
Model2	naive	submodel=gauss gausstype=general	-53.2231 +/- 0.610411

learned by "Joint"

The DE learned by "Naive"

“Weight vs. MPG”: The best that Naïve can do

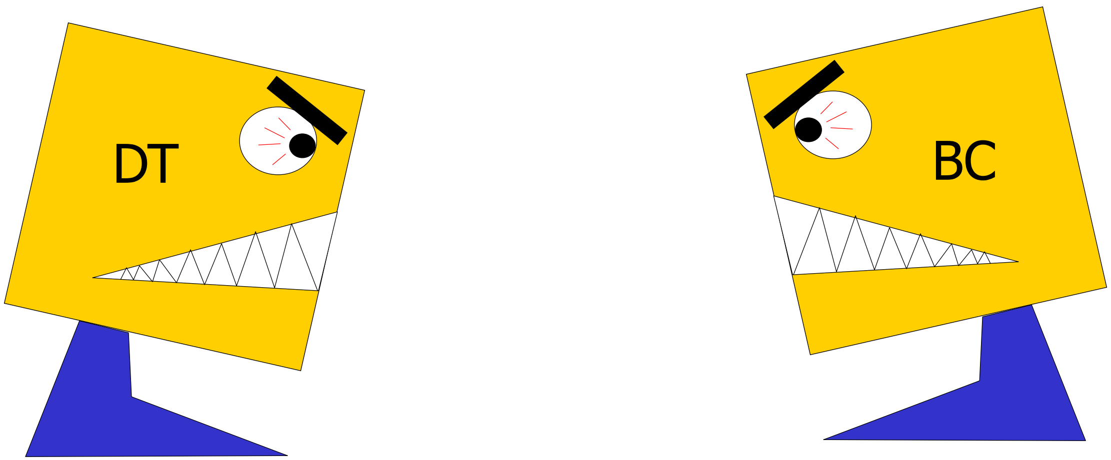


Reminder: The Good News

- We have two ways to learn a Density Estimator from data.
- *In other lectures we'll see vastly more impressive Density Estimators (Mixture Models, Bayesian Networks, Density Trees, Kernel Densities and many more)
- Density estimators can do many good things…
  - Anomaly detection
  - Can do inference: P(E1|E2) Automatic Doctor / Help Desk etc
  - Ingredient for Bayes Classifiers

Bayes Classifiers

- A formidable and sworn enemy of decision trees



How to build a Bayes Classifier

- Assume you want to predict output Y which has arity n_Y and values $v_1, v_2, \ldots v_{ny}$.
- Assume there are m input attributes called $X_1, X_2, \ldots X_m$
- Break dataset into n_Y smaller datasets called $DS_1, DS_2, \ldots DS_{ny}$.
- Define DS_i = Records in which $Y=v_i$
- For each DS_i , learn Density Estimator M_i to model the input distribution among the $Y=v_i$ records.

How to build a Bayes Classifier

- Assume you want to predict output Y which has arity n_Y and values $v_1, v_2, \ldots v_{ny}$.
- Assume there are m input attributes called $X_1, X_2, \ldots X_m$
- Break dataset into n_Y smaller datasets called $DS_1, DS_2, \ldots DS_{ny}$.
- Define DS_i = Records in which $Y=v_i$
- For each DS_i , learn Density Estimator M_i to model the input distribution among the $Y=v_i$ records.
- M_i estimates $P(X_1, X_2, \ldots X_m \mid Y=v_i)$

How to build a Bayes Classifier

- Assume you want to predict output Y which has arity n_Y and values $v_1, v_2, \ldots v_{ny}$.
- Assume there are m input attributes called $X_1, X_2, \ldots X_m$
- Break dataset into n_Y smaller datasets called $DS_1, DS_2, \ldots DS_{ny}$.
- Define DS_i = Records in which $Y=v_i$
- For each DS_i , learn Density Estimator M_i to model the input distribution among the $Y=v_i$ records.
- M_i estimates $P(X_1, X_2, \ldots X_m \mid Y=v_i)$

- Idea: When a new set of input values ($X_1 = u_1, X_2 = u_2, \ldots. X_m = u_m$) come along to be evaluated predict the value of Y that makes $P(X_1, X_2, \ldots X_m \mid Y=v_i)$ most likely

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)$$

Is this a good idea?

How to build a ~~Bayes~~ Classifier

- Assume you want to predict output Y which has arity n_Y and values $v_1, v_2, \ldots v_{ny}$.
- Assume there are m input attribu
- Break dataset into n_Y smaller dat
- Define DS_i = Records in which Y
- For each DS_i , learn Density Esti distribution among the $Y=v_i$ reco
- M_i estimates P($X_1, X_2, \ldots X_m$ / $Y=v_i$)

This is a Maximum Likelihood classifier.

It can get silly if some Ys are very unlikely

- Idea: When a new set of input values ($X_1 = u_1, X_2 = u_2, \ldots. X_m = u_m$) come along to be evaluated predict the value of Y that makes P($X_1, X_2, \ldots X_m$ / $Y=v_i$) most likely

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)$$

Is this a good idea?

How to build a Bayes Classifier

- Assume you want to predict output Y which has arity n_Y and values $v_1, v_2, \ldots v_{ny}$.
- Assume there are m input attributes calle
- Break dataset into n_Y smaller datasets call
- Define DS_i = Records in which $Y=v_i$
- For each DS_i , learn Density Estimator M_i
distribution among the $Y=v_i$ records.
- M_i estimates P($X_1, X_2, \ldots X_m$ / $Y=v_i$)

Much Better Idea

- Idea: When a new set of input values ($X_1 = u_1, X_2 = u_2, \ldots. X_m = u_m$) come along to be evaluated predict the value of Y that makes P($Y=v_i$ / $X_1, X_2, \ldots X_m$) most likely

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

Is this a good idea?

Terminology

- MLE (Maximum Likelihood Estimator):

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)$$

- MAP (Maximum A-Posteriori Estimator):

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

Getting what we need

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \; P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

Getting a posterior probability

$$P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

$$= \frac{P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)}{P(X_1 = u_1 \cdots X_m = u_m)}$$

$$= \frac{P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)}{\sum_{j=1}^{n_Y} P(X_1 = u_1 \cdots X_m = u_m \mid Y = v_j) P(Y = v_j)}$$

Bayes Classifiers in a nutshell

1. Learn the distribution over inputs for each value Y.
2. This gives P(X_1, X_2, … X_m / $Y=v_i$).
3. Estimate P($Y=v_i$). as fraction of records with $Y=v_i$.
4. For a new prediction:

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(Y = v \mid X_1 = u_1 \cdots X_m = u_m)$$

$$= \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)$$

Bayes Classifiers in a nutshell

1. Learn the distribution over inputs for each value Y.
2. This gives P(X_1, X_2, … X_m / $Y=v_i$).
3. Estimate P($Y=v_i$). as fraction of records w… $Y=v_i$.
4. For a new prediction:

$$Y^{\text{predict}} = \underset{v}{\text{argmax}}\ P(Y = v \mid X_1 =$$

$$= \underset{v}{\text{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m$$

We can use our favorite Density Estimator here.

Right now we have two options:

- •Joint Density Estimator
- •Naïve Density Estimator

Joint Density Bayes Classifier

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)P(Y = v)$$

In the case of the joint Bayes Classifier this degenerates to a very simple rule:

$Y^{predict}$ = the most common value of Y among records in which $X_1 = u_1, X_2 = u_2, \ldots. X_m = u_m$.

Note that if no records have the exact set of inputs $X_1 = u_1, X_2 = u_2, \ldots. X_m = u_m$, then P($X_1, X_2, \ldots$ X$_m$ / $Y=v_i$) = 0 for all values of Y.

In that case we just have to guess Y's value

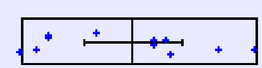
Joint BC Results: "Logical"

The "logical" dataset consists of 40,000 records and 4 Boolean attributes called a,b,c,d where a,b,c are generated 50-50 randomly as 0 or 1. D = A^~C, except that in 10% of records it is flipped

d = 0				d = 1			
(prior = 0.69945)				(prior = 0.30055)			
a	b	c		a	b	c	
0	0	0	0.162056	0	0	0	0.0425886
		1	0.162378			1	0.0407586
	1	0	0.16152		1	0	0.04001
		1	0.158982			1	0.0404259
1	0	0	0.017621	1	0	0	0.374314
		1	0.159232			1	0.0408418
	1	0	0.017621		1	0	0.377142
		1	0.16059			1	0.0439195

The Classifier learned by "Joint BC"

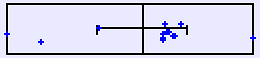
Name	Model	Parameters	FracRight		
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.90065	+/-	0.00301897



Joint BC Results: "All Irrelevant"

The "all irrelevant" dataset consists of 40,000 records and 15 Boolean attributes called a,b,c,d..o where a,b,c are generated 50-50 randomly as 0 or 1. v (output) = 1 with probability 0.75, 0 with prob 0.25

Name	Model	Parameters	FracRight		
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.70425	+/-	0.00583537



Naïve Bayes Classifier

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(X_1 = u_1 \cdots X_m = u_m \mid Y = v)P(Y = v)$$

In the case of the naive Bayes Classifier this can be simplified:

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}}\ P(Y = v)\prod_{j=1}^{n_Y} P(X_j = u_j \mid Y = v)$$

Naïve Bayes Classifier

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \; P(X_1 = u_1 \cdots X_m = u_m \mid Y = v) P(Y = v)$$

In the case of the naive Bayes Classifier this can be simplified:

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \; P(Y = v) \prod_{j=1}^{n_Y} P(X_j = u_j \mid Y = v)$$

Technical Hint:
If you have 10,000 input attributes that product will underflow in floating point math. You should use logs:

$$Y^{\text{predict}} = \underset{v}{\operatorname{argmax}} \left(\log P(Y = v) + \sum_{j=1}^{n_Y} \log P(X_j = u_j \mid Y = v) \right)$$

BC Results: "XOR"

The "XOR" dataset consists of 40,000 records and 2 Boolean inputs called a and b, generated 50-50 randomly as 0 or 1. c (output) = a XOR b

c = 0			c = 1		
(prior = 0.503175)			(prior = 0.496825)		
a	b		a	b	
0	0	0.50077	0	0	Never
	1	Never		1	0.499874
1	0	Never	1	0	0.500126
	1	0.49923		1	Never

The Classifier learned by "Joint BC"

c = 0			c = 1		
(prior = 0.503175)			(prior = 0.496825)		
a	0	0.50077	a	0	0.499874
	1	0.49923		1	0.500126
b	0	0.50077	b	0	0.500126
	1	0.49923		1	0.499874

The Classifier learned by "Naive BC"

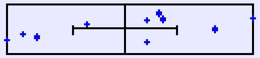
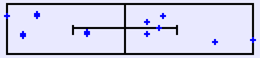
	del	Parameters	FracRight		
	yesclass	density=joint submodel=gauss gausstype=general	1	+/-	0
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.500125	+/-	0.00529626

Naive BC Results: "Logical"

The "logical" dataset consists of 40,000 records and 4 Boolean attributes called a,b,c,d where a,b,c are generated 50-50 randomly as 0 or 1. D = A^~C, except that in 10% of records it is flipped

d = 0			d = 1		
(prior = 0.69945)			(prior = 0.30055)		
a	0	0.644935	a	0	0.163783
	1	0.355065		1	0.836217
b	0	0.501287	b	0	0.498503
	1	0.498713		1	0.501497
c	0	0.358818	c	0	0.834054
	1	0.641182		1	0.165946

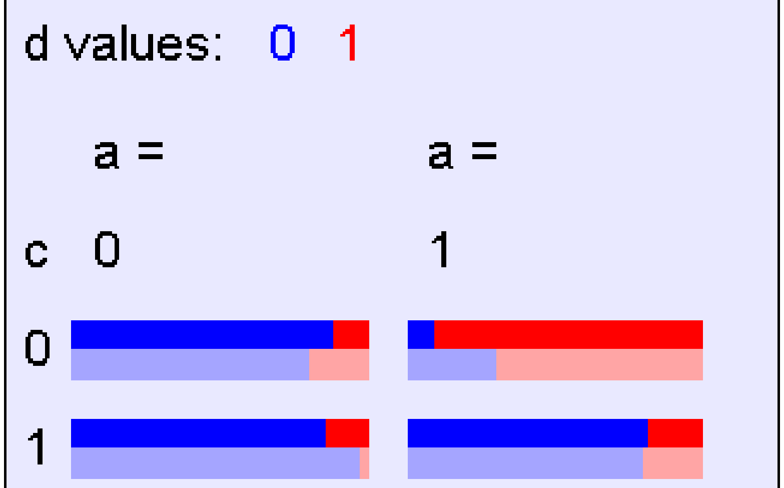
The Classifier learned by "Naive BC"

Name	Model	Parameters	FracRight			
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.90065	+/-	0.00301897	
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.90065	+/-	0.00301897	

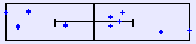
Naive BC Results: "Logical"

The "logical" dataset consists of 40,000 records and 4 Boolean attributes called a,b,c,d where a,b,c are generated 50-50 randomly as 0 or 1. D = A^~C, except that in 10% of records it is flipped

This result surprised Andrew until he had thought about it a little



The data shown in the figure is merely a subsample of the full dataset. The light color shades denote predicted classes. The dark shades are real data.

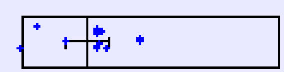
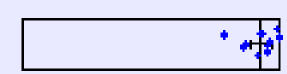
Name	Model	Parameters	FracRight			
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.90065	+/-	0.00301897	
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.90065	+/-	0.00301897	

Naïve BC Results: "All Irrelevant"

v = 0			v = 1		
(prior = 0.75065)			(prior = 0.24935)		
a	0	0.500067	a	0	0.501103
	1	0.499933		1	0.498897
b	0	0.5004	b	0	0.500602
	1	0.4996		1	0.499398
c	0	0.503031	c	0	0.497493
	1	0.496969		1	0.502507
d	0	0.501798	d	0	0.505013
	1	0.498202		1	0.494987
e	0	0.500466	e	0	0.500401
	1	0.499534		1	0.499599
f	0	0.498335	f	0	0.50401
	1	0.501665		1	0.49599
g	0	0.505595	g	0	0.509425

The "all irrelevant" dataset consists of 40,000 records and 15 Boolean attributes called a,b,c,d..o where a,b,c are generated 50-50 randomly as 0 or 1. v (output) = 1 with probability 0.75, 0 with prob 0.25

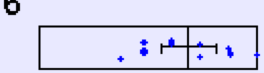
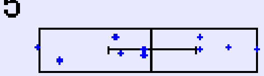
The Classifier learned by "Naive BC"

Name	Model	Parameters	FracRight			
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.70425	+/-	0.00583537	
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.75065	+/-	0.00281976	

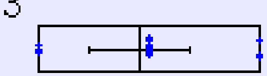
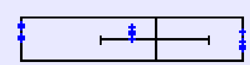
BC Results: "MPG": 392 records

Attribute	Value	mpg = bad (prior = 0.602041)	mpg = good (prior = 0.397959)
cylinders	3	0.0169492	0
	4	0.220339	0.942308
	5	0.00423729	0.0128205
	6	0.326271	0.0384615
	8	0.432203	0.00641026
displacement	low	0.338983	0.987179
	high	0.661017	0.0128205
horsepower	low	0.207627	0.891026
	high	0.792373	0.108974
weight	low	0.322034	0.955128
	high	0.677966	0.0448718
acceleration	low	0.550847	0.320513
	high	0.449153	0.679487
modelyear	70to74	0.504237	0.198718
	75to77	0.385593	0.237179
	78to83	0.110169	0.564103
maker	america	0.809322	0.346154
	asia	0.0805085	0.384615
	europe	0.110169	0.269231

The Classifier learned by "Naive BC"

Name	Model	Parameters	FracRight	
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.885256 +/- 0.0247796	
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.852372 +/- 0.0400495	

BC Results: "MPG": 40 records

Name	Model	Parameters	FracRight			
Model1	bayesclass	density=joint submodel=gauss gausstype=general	0.725	+/-	0.114333	
Model2	bayesclass	density=naive submodel=gauss gausstype=general	0.8	+/-	0.122227	

More Facts About Bayes Classifiers

- Many other density estimators can be slotted in*.
- Density estimation can be performed with real-valued inputs*
- Bayes Classifiers can be built with real-valued inputs*
- Rather Technical Complaint: Bayes Classifiers don't try to be maximally discriminative---they merely try to honestly model what's going on*
- Zero probabilities are painful for Joint and Naïve. A hack (justifiable with the magic words "Dirichlet Prior") can help*.
- Naïve Bayes is wonderfully cheap. And survives 10,000 attributes cheerfully!

*See future Andrew Lectures

What you should know

- Probability
  - Fundamentals of Probability and Bayes Rule
  - What's a Joint Distribution
  - How to do inference (i.e. $P(E1|E2)$) once you have a JD
- Density Estimation
  - What is DE and what is it good for
  - How to learn a Joint DE
  - How to learn a naïve DE

What you should know

- Bayes Classifiers
  - How to build one
  - How to predict with a BC
  - Contrast between naïve and joint BCs

Interesting Questions

- Suppose you were evaluating NaiveBC, JointBC, and Decision Trees
  - Invent a problem where only NaiveBC would do well
  - Invent a problem where only Dtree would do well
  - Invent a problem where only JointBC would do well
  - Invent a problem where only NaiveBC would do poorly
  - Invent a problem where only Dtree would do poorly
  - Invent a problem where only JointBC would do poorly