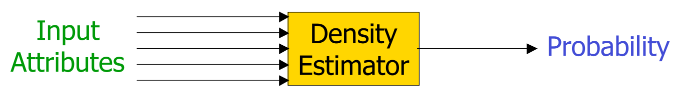
Density Estimation

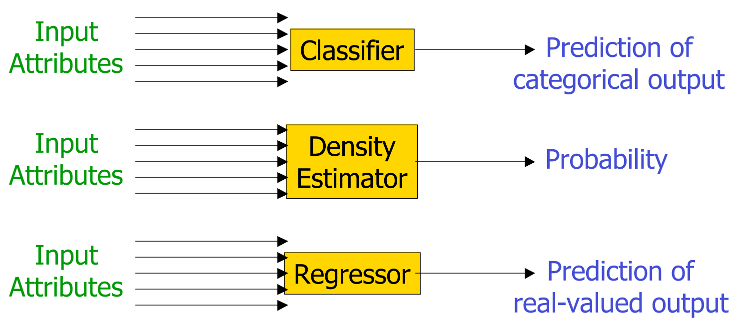
- Our Joint Distribution learner is our first example of something called Density Estimation
- A Density Estimator learns a mapping from a set of attributes to a Probability



- There are **many** different kinds of Estimators

Density Estimation

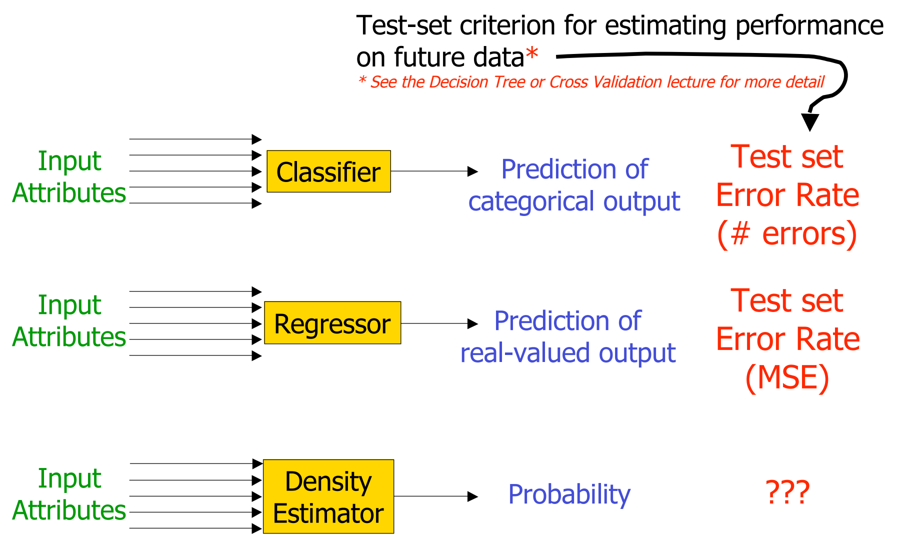
- Compare it against the two other major kinds of models:



Why are we doing this again?

- Density Estimators can do many magical things. They can answer:
  - P(chips|beer)
    - If you just want to know probabilities
  - P(canadian|features), P(icelander|features)
    - If you want to know which is more likely, so that you can classify
  - P(x_1 <= life_span <= x_2)
    - Like regression, but *better*.
- I.e., they can do all three of our first examples. This is pretty awesome.

Evaluating Density Estimation



Evaluating a density estimator

- Given one 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)

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

You can think of *M* as being similar to *w* from regression…

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

mpg	modelyear	maker	
bad	70to74	america	0.27551
		asia	0.0255102
		europe	0.0153061
	75to77	america	0.153061
		asia	0.0255102
		europe	0.0357143
		asia	0.0714286
		europe	0.0357143

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

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

Log Probabilities

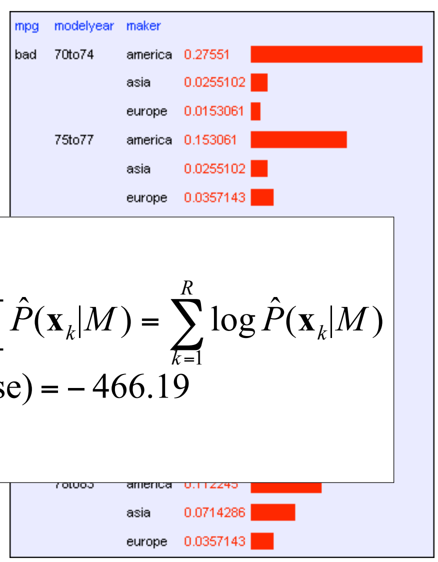
Since probabilities of datasets get so small we usually use log probabilities

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

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



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

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

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

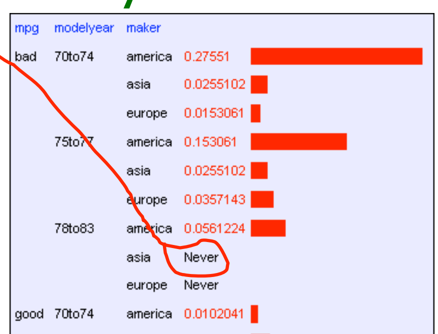
(actually it's a billion quintillion quintillion quintillion quintillion times less likely)

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

Remember the lookup table classifier with 2^{2^m} hypotheses?

Overfitting Density Estimators

If this ever happens during training, it means there are certain combinations that we learn are impossible. The probabilities will be set to 0.



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

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

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 $i, v, u_1\ u_2 \ldots\ u_{i-1}\ u_{i+1} \ldots\ u_M$

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

"Given everything *except* x[i]"

$$= P(x[i]=v)$$

- 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^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…

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

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

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?
  - Think: What do I know about A? What do I know about A if I know B? That is: What is P(A)? What is P(A|B)? What about P(A|B=5)?

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)*?

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.

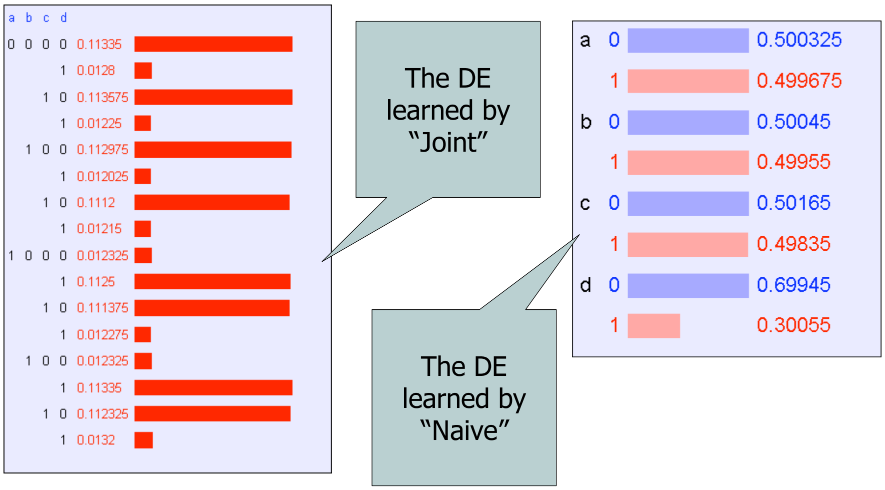
Name	Model	Parameters	LogLike		
Model1	joint	submodel=gauss gausstype=general	-272625	+/-	301.109
Model2	naive	submodel=gauss gausstype=general	-58225.6	+/-	0.554747

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

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

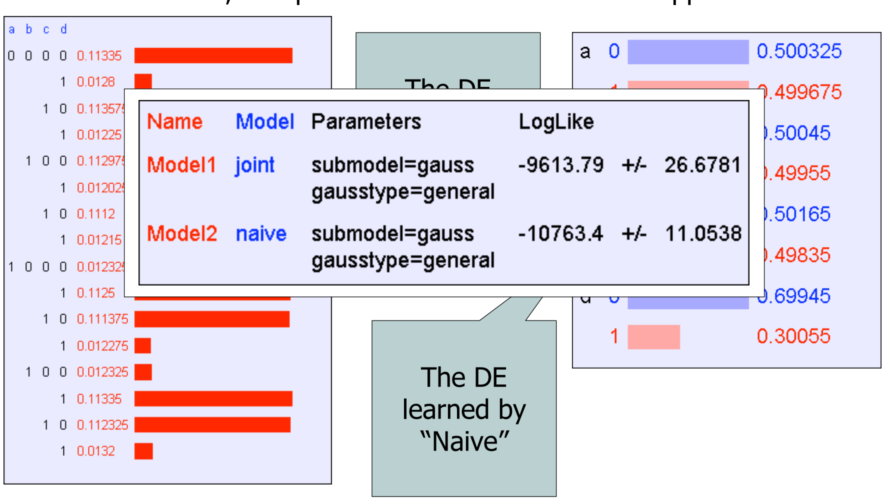
Empirical Results: "Logical"

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



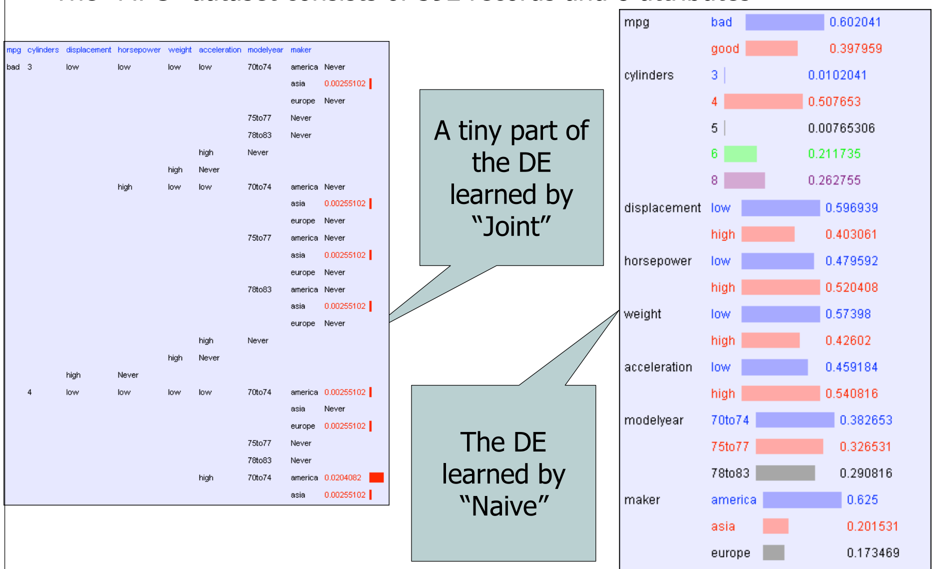
Empirical Results: "Logical"

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



Empirical Results: “MPG”

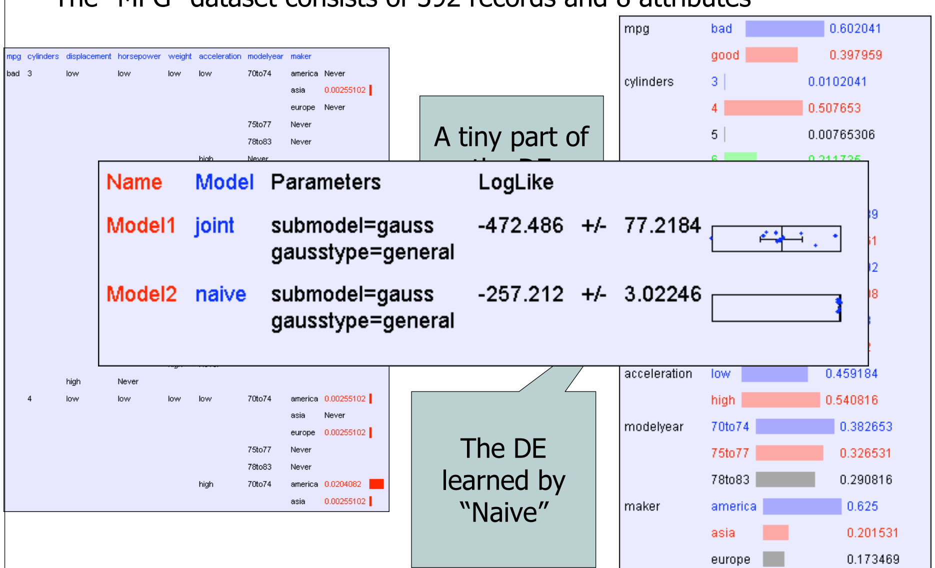
The “MPG” dataset consists of 392 records and 8 attributes



 Slide 85

Empirical Results: “MPG”

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



Name	Model	Parameters	LogLike
Model1	joint	submodel=gauss gausstype=general	-472.486 +/- 77.2184
Model2	naive	submodel=gauss gausstype=general	-257.212 +/- 3.02246

 Slide 86

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	low	0.380102
	high	0.0178571

The DE learned by "Joint"

mpg	bad	0.602041
	good	0.397959
weight	low	0.57398
	high	0.42602

The DE learned by "Naive"

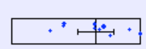
Empirical Results: "Weight vs. MPG"

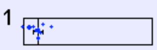
Suppose we train only from the "Weight" and "MPG" attributes

mpg	weight	
bad	low	0.193878
	high	0.408163
good		

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

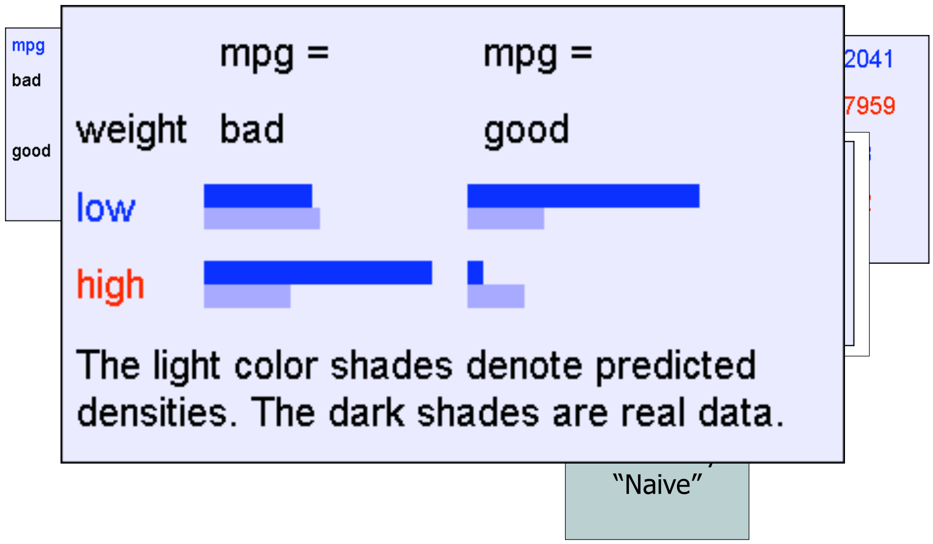




learned by "Joint"

The DE learned by "Naive"

"Weight vs. MPG": The best that Naïve can do

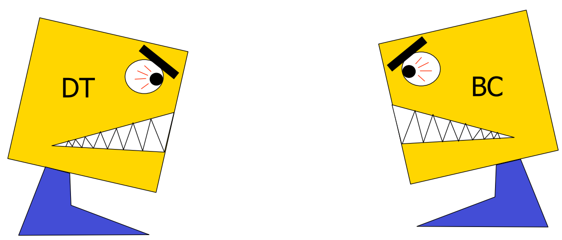


Reminder: The Good News

- 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, … v_{ny}.
- Assume there are m input attributes called X_1, X_2, … X_m
- Break dataset into n_Y smaller datasets called DS_1, DS_2, … 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 $P(X_1, X_2, \ldots X_m \mid Y=v_i)$

 Slide 93

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?

 Slide 94

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($X_1, X_2, \ldots X_m \mid Y=v_i$)

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($X_1, X_2, \ldots X_m \mid Y=v_i$) 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 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($X_1, X_2, \ldots X_m \mid Y=v_i$)

Much Better Idea

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

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

$$= \underset{v}{\operatorname{argmax}}\, P(X_1 = u_1 \cdots X_m = u_m \ldots$$

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 $X_1 = u_1$, $X_2 = u_2$, …. $X_m = u_m$.

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

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

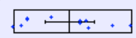
Joint BC Results: "Logical"

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 (prior = 0.69945) a b c				d = 1 (prior = 0.30055) 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

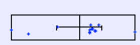


 Slide 103

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



 Slide 104

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

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 "Joint BC"

The Classifier learned by "Naive BC"

	del	Parameters	FracRight		
	esclass	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

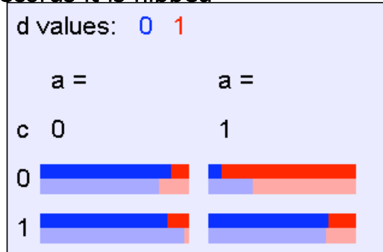
The Classifier learned by "Naive BC"

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

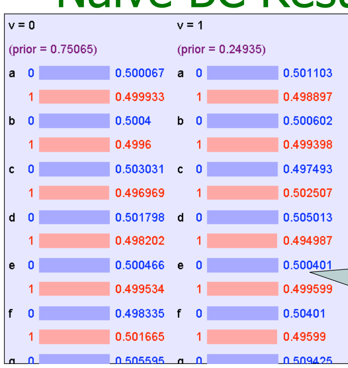
This result surprised Andrew until he had thought about it a little



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”

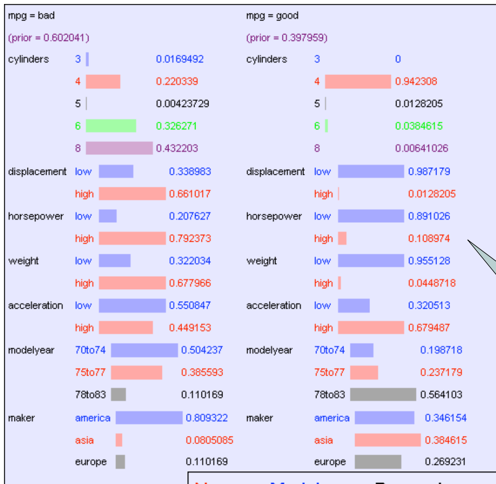


The “all irrelevant” dataset consists of 40,000 records and 15 Boolean attributes called a,b,c,d..o where a,b,c are generated 50-50 randomly as 0 or 1. v (output) = 1 with probability 0.75, 0 with prob 0.25

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



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