

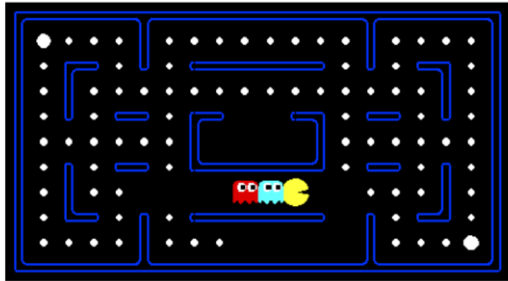
Machine Learning

Chen-Yu Wei

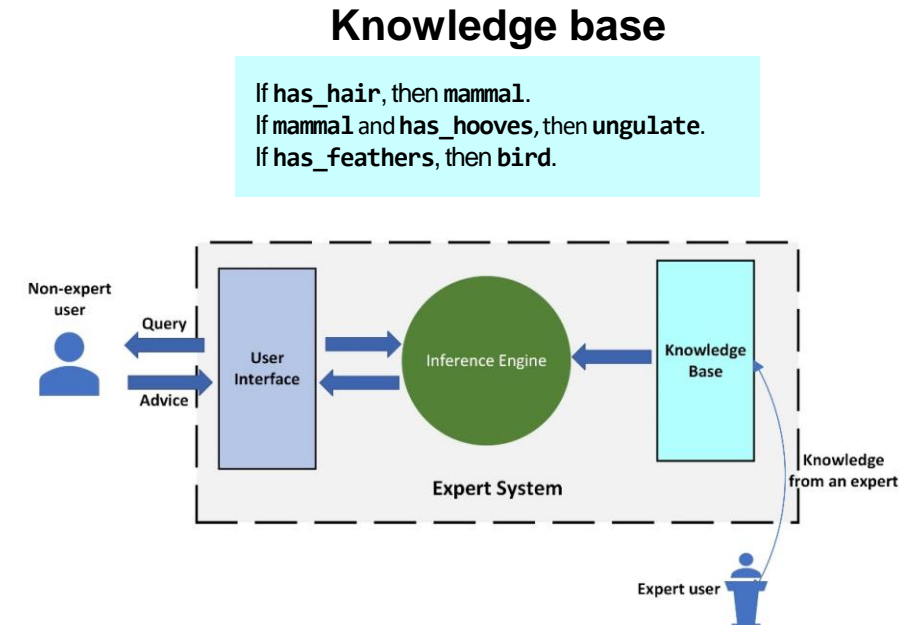
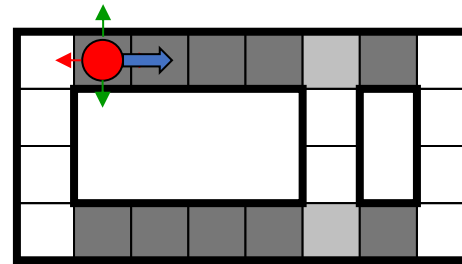
What we have studied so far...

How are these “rules” or “model of the world” obtained?

- Given the **rule of a game** (and/or **behavior model of the opponents**), find the optimal solution (search, search in multi-player games)
- Given the **relation among variables**, find a satisfied solution (constraint satisfaction)
- Given the **relation among variables**, find the probability of certain events, or the most probable events (Bayes nets, HMM)
- Given a **knowledge base**, infer some facts (logic)



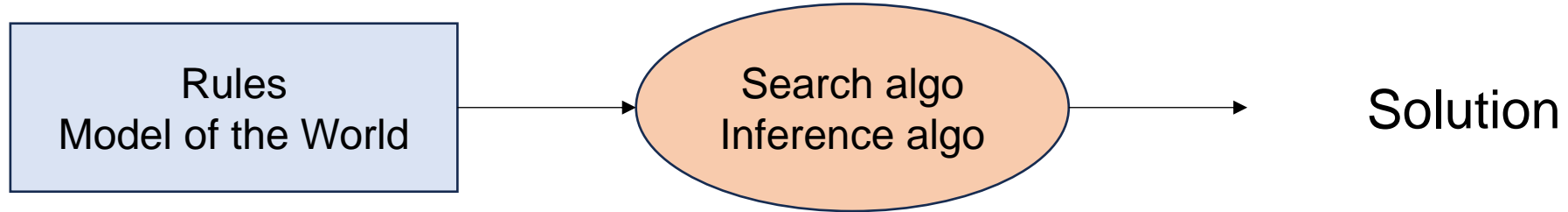
					8			4
	8	4		1	6			
			5			1		
1		3	8			9		
6		8				4		3
		2			9	5		1
		7			2			
			7	8		2	6	
2			3					



Rules or Model of the World

- In games designed by human, we simply have the ground-truth rules
 - Pacman
 - Chess, Go
 - Sudoku
- Some are rules set by human based on their observations/knowledge of the world
 - Knowledge base in expert systems
- Some have appeared magically so far
 - The behavior model of the ghosts in Pacman
 - Probability tables in Bayes nets, HMM

Machine Learning



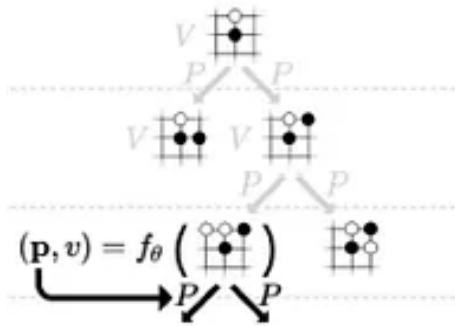
What we have taken for granted

We will discuss how to let machine **learn** these models through **data** collected from the world

Machine Learning

Machine Learning

In some cases, even when the world model is designed by human and known, we still want to perform machine learning



Evaluation function / Heuristic function

Depth-limited search

Guide the search in games with large branching factors



Low-level rules (known)



**Machine learning
from simulations**



High-level rules (learned)

Naïve Bayes

Learning Simple Bayesian Networks

Suppose we have a set of data:

$(Y, X) =$

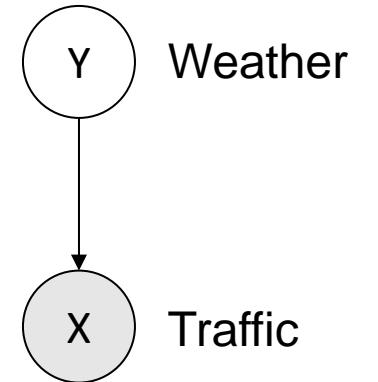
{ (sun, F) , (rain, F) , (rain, T) , (rain, T)
(sun, T) , (sun, F) , (sun, F) , (sun, T) }

How should we build the BN model?

Y	P(Y)
Sun	$5/8$
Rain	$3/8$

Y	X	P(X Y)
Sun	T	$2/5$
Sun	F	$3/5$
Rain	T	$2/3$
Rain	F	$1/3$

training



$P(Y)$

$P(X | Y)$

$P(Y | X)$

If now we observe X (traffic) = T, how to infer the Y (weather) distribution?

How did we obtain the parameters?

Why do we model $P(X = T \mid Y = \text{sun})$ as $\frac{\#(Y=\text{sun}, X=T)}{\#(Y=\text{sun})}$ in the dataset?

Maximum Likelihood Estimation (MLE)

can be used in training any BNs with finite domains

set of all possible models

$\subset \mathbb{R}^6$

Pick $\operatorname{argmax}_M \prod_{i=1}^n \underbrace{P_M(x_i, y_i)}_{\text{Likelihood}}$

Best explains the data (?)
--- has some drawbacks
(discussed later)

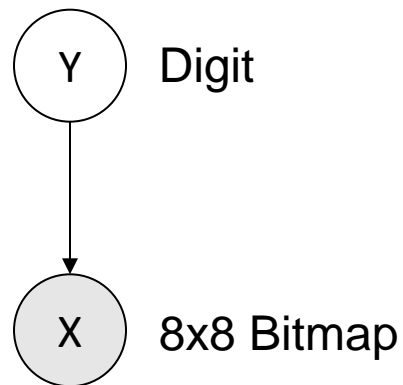
Approximate inference

We have the model, and thus the exact value of $P(Y|X)$ is available. But because the exact computation is expensive, we approximate it with samples **drawn from the model**.

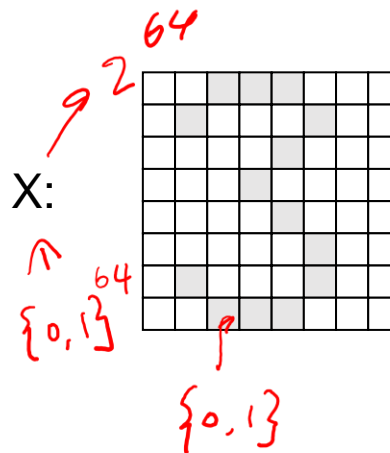
Model learning

We do not have the model, and try to build it from data **drawn from the nature**.

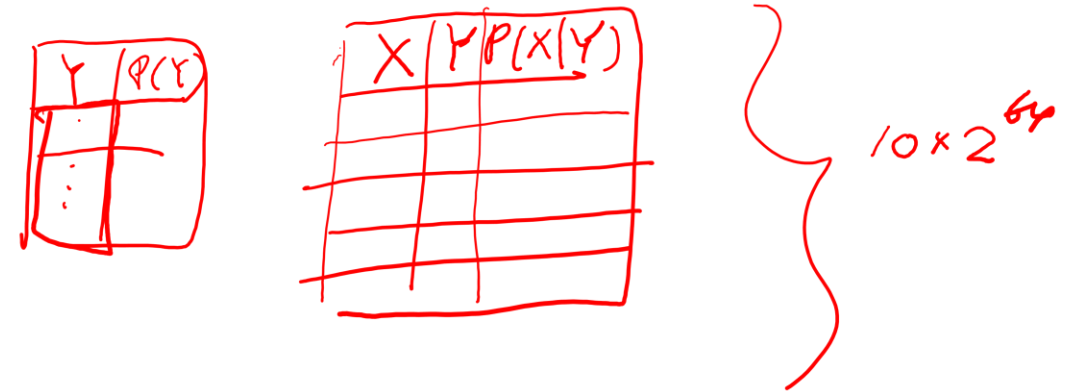
Dealing with High-Dimensional Observation



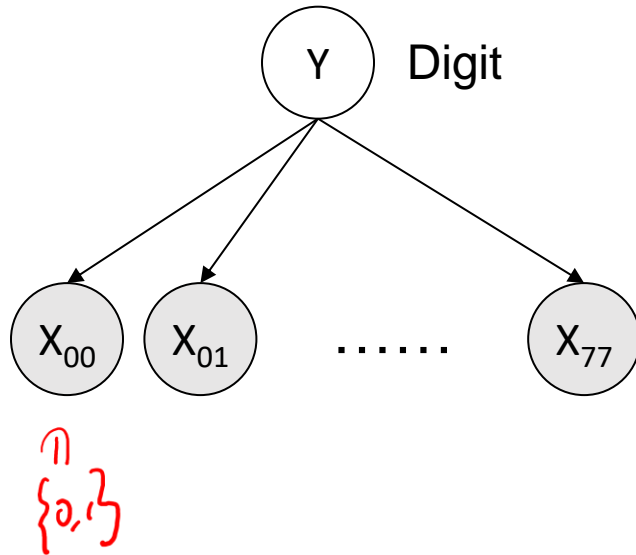
10 values
 $Y \in \{0, 1, 2, \dots, 9\}$



Number of parameters in this model?

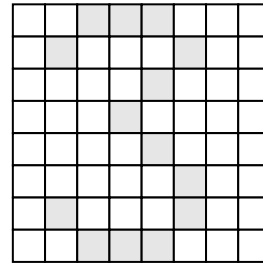


Dealing with High-Dimensional Observation



$Y \in \{0, 1, 2, \dots, 9\}$

X:



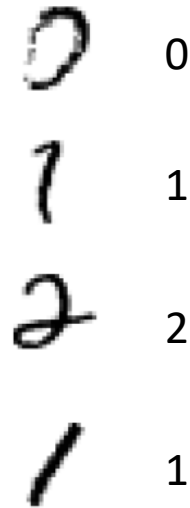
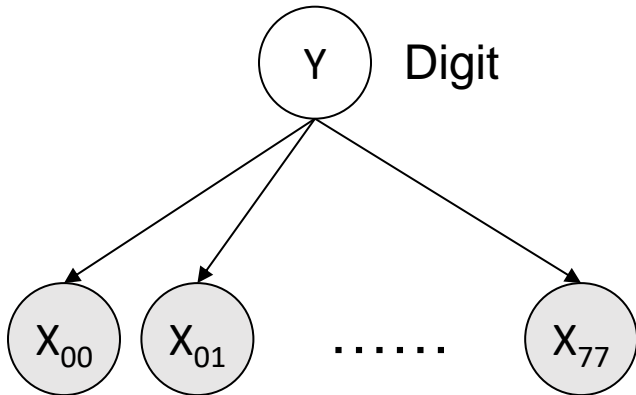
Number of parameters in this model?

$$10 \times 2 \times 64$$

Dealing with High-Dimensional Observation

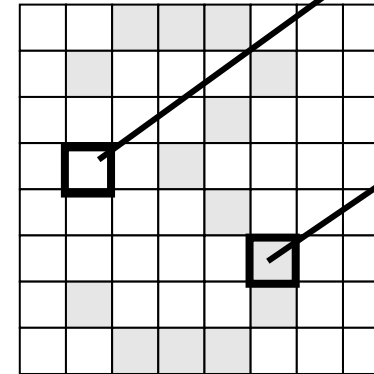
Training:

- 1) Get dataset
- 2) Match model with empirical frequency



$P(Y)$

1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.1
7	0.1
8	0.1
9	0.1
0	0.1



$P(X_{31}=\text{on} \mid Y)$

1	0.01
2	0.05
3	0.05
4	0.30
5	0.80
6	0.90
7	0.05
8	0.60
9	0.50
0	0.80

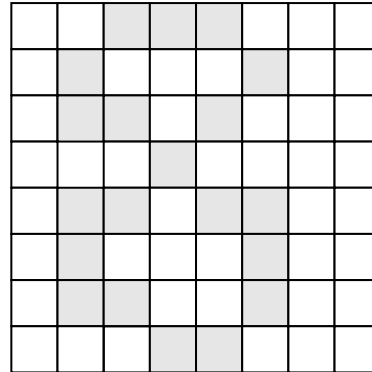
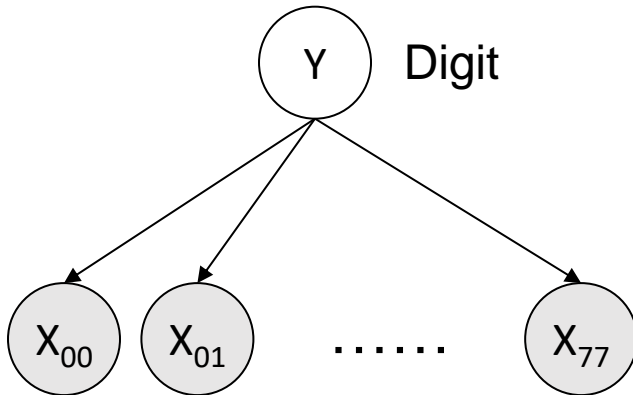
$P(X_{55}=\text{on} \mid Y)$

1	0.05
2	0.01
3	0.90
4	0.80
5	0.90
6	0.90
7	0.25
8	0.85
9	0.60
0	0.80

Dealing with High-Dimensional Observation

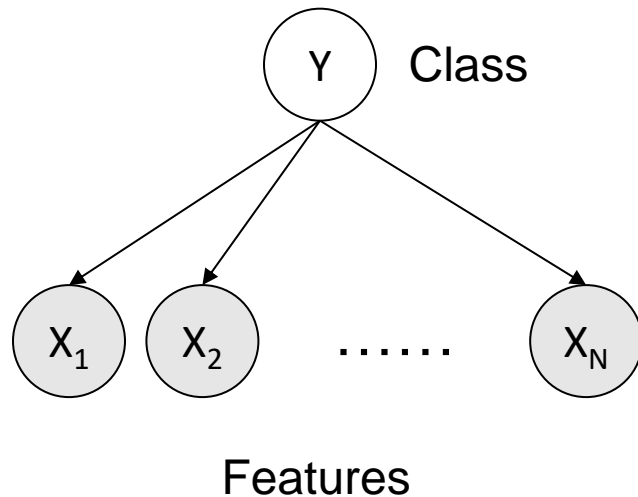
Inference:

After training, now given a bitmap, decide its likelihood to be each digit



$$\begin{aligned} P(Y \mid x_{00}, x_{01}, \dots, x_{77}) \\ &\propto p(Y, x_{00}, x_{01}, \dots, x_{77}) \\ &= \underline{p(Y) p(x_{00} \mid Y) p(x_{01} \mid Y) \dots p(x_{77} \mid Y)} \end{aligned}$$

General Naïve Bayes Model



Training:

- 1) Get dataset consisting of $(X, Y) = (X_1, \dots, X_N, Y)$ pairs
- 2) Train model $P(Y)$, $P(X_i | Y)$ with maximum likelihood estimation (= empirical frequency)

(more options discussed later)

Inference:

Given x ,

$$\text{Infer } P(Y | x) \propto P(Y) P(x_1 | Y) P(x_2 | Y) \dots P(x_N | Y)$$

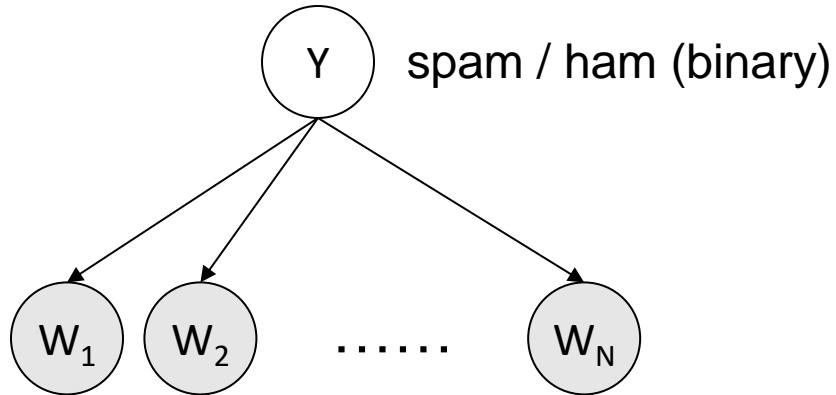
Finite domains for Y and X_i

Example: Spam Filtering

Training data:

Collection of emails, labeled spam or ham

Model (**bag-of-word**):



Special assumption (not in the digit example):

$P(W_i | Y)$ is identical for every i

→ This is why it is called bag-of-world (word ordering does not matter)



Dear Sir.

First, I must solicit your confidence in this transaction, this is by virtue of its nature as being utterly confidential and top secret. ...



TO BE REMOVED FROM FUTURE MAILINGS,
SIMPLY REPLY TO THIS MESSAGE AND PUT
"REMOVE" IN THE SUBJECT.

99 MILLION EMAIL ADDRESSES
FOR ONLY \$99



Ok, I know this is blatantly OT but I'm beginning to go insane. Had an old Dell Dimension XPS sitting in the corner and decided to put it to use, I know it was working pre being stuck in the corner, but when I plugged it in, hit the power nothing happened.

Example: Spam Filtering

- Model: $P(Y, W_1 \dots W_n) = P(Y) \prod_i P(W_i|Y)$
- What are the parameters?

$P(Y)$

ham : 0.66
spam: 0.33

$P(W|\text{spam})$

the : 0.0156
to : 0.0153
and : 0.0115
of : 0.0095
you : 0.0093
a : 0.0086
with: 0.0080
from: 0.0075
...

$P(W|\text{ham})$

the : 0.0210
to : 0.0133
of : 0.0119
2002: 0.0110
with: 0.0108
from: 0.0107
and : 0.0105
a : 0.0100
...

Spam Example

$$\log \left(P(Y) \prod_i P(w_i | Y) \right) = \log P(Y) + \sum_i \log(w_i | Y)$$

Word	P(w spam)	P(w ham)	Tot Spam	Tot Ham
(prior)	0.33333	0.66666	-1.1	-0.4
Gary	0.00002	0.00021	-11.8	-8.9
would	0.00069	0.00084	-19.1	-16.0
you	0.00881	0.00304	-23.8	-21.8
like	0.00086	0.00083	-30.9	-28.9
to	0.01517	0.01339	-35.1	-33.2
lose	0.00008	0.00002	-44.5	-44.0
weight	0.00016	0.00002	-53.3	-55.0
while	0.00027	0.00027	-61.5	-63.2
you	0.00881	0.00304	-66.2	-69.0
sleep	0.00006	0.00001	-76.0	-80.5

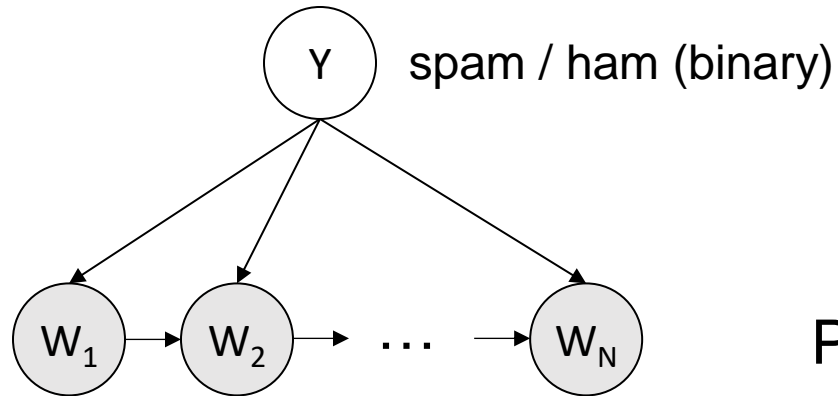
$P(\text{spam} | w) = 98.9$

Another Possible Model

May slightly improve accuracy

e.g., Earn money vs. Earn degree

with the price of larger model (usually
requires more data to train)



$$P(W_i \mid Y, W_{i-1})$$

Overfitting and Generalization

If using Maximum Likelihood Estimation...

For a new bitmap:

$$P(\text{features}, C = 2)$$

$$P(C = 2) = 0.1$$

$$P(\text{on}|C = 2) = 0.8$$

$$P(\text{on}|C = 2) = 0.1$$

$$P(\text{off}|C = 2) = 0.1$$

$$P(\text{on}|C = 2) = 0.01$$

$$P(\text{features}, C = 3)$$

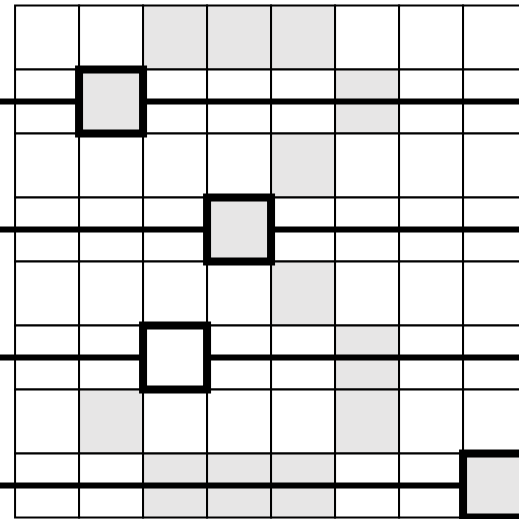
$$P(C = 3) = 0.1$$

$$P(\text{on}|C = 3) = 0.8$$

$$P(\text{on}|C = 3) = 0.9$$

$$P(\text{off}|C = 3) = 0.7$$

$$P(\text{on}|C = 3) = 0.0$$



2 wins!!

If using Maximum Likelihood Estimation...

Prediction determined by *relative* probabilities:

$$\frac{P(W|\text{ham})}{P(W|\text{spam})}$$

south-west	:	inf
nation	:	inf
morally	:	inf
nicely	:	inf
extent	:	inf
seriously	:	inf
...		

$$\frac{P(W|\text{spam})}{P(W|\text{ham})}$$

screens	:	inf
minute	:	inf
guaranteed	:	inf
\$205.00	:	inf
delivery	:	inf
signature	:	inf
...		

Overfitting

- Relative frequency parameters will **overfit** the training data!
 - Just because we never saw a 3 with pixel (15,15) on during training doesn't mean we won't see it at test time
 - Unlikely that every occurrence of "minute" is 100% spam
 - Unlikely that every occurrence of "seriously" is 100% ham
 - What about all the words that don't occur in the training set at all?
 - In general, we can't give unseen events zero probability
- For Naïve Bayes we use **smoothing** to address this issue
 - A special case of the general concept of "regularization"

Laplace Smoothing

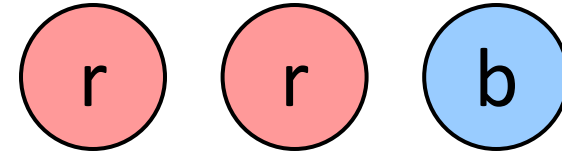
- Laplace's estimate:
 - Pretend you saw every outcome k extra times

$$P_{LAP,k}(x) = \frac{c(x) + k}{N + k|X|}$$

- What's Laplace with $k = 0$?
- k is the **strength** of the prior

- Laplace for conditionals:
 - Smooth each condition independently:

$$P_{LAP,k}(x|y) = \frac{c(x, y) + k}{c(y) + k|X|}$$



$$P_{LAP,0}(X) =$$

$$P_{LAP,1}(X) =$$

$$P_{LAP,100}(X) =$$

Real NB: Smoothing

- For real classification problems, smoothing is critical
- New odds ratios:

$$\frac{P(W|\text{ham})}{P(W|\text{spam})}$$

helvetica	:	11.4
seems	:	10.8
group	:	10.2
ago	:	8.4
areas	:	8.3
...		

$$\frac{P(W|\text{spam})}{P(W|\text{ham})}$$

verdana	:	28.8
Credit	:	28.4
ORDER	:	27.2
	:	26.9
money	:	26.5
...		

Tuning on Held-Out Data

- Now we've got two kinds of unknowns
 - Parameters: the probabilities $P(X|Y)$, $P(Y)$
 - Hyperparameters: e.g. the amount / type of smoothing to do, k , α
- What should we learn where?
 - Learn parameters from training data
 - Tune hyperparameters on different data
 - For each value of the hyperparameters, train and test on the held-out data
 - Choose the best value and do a final test on the test data

