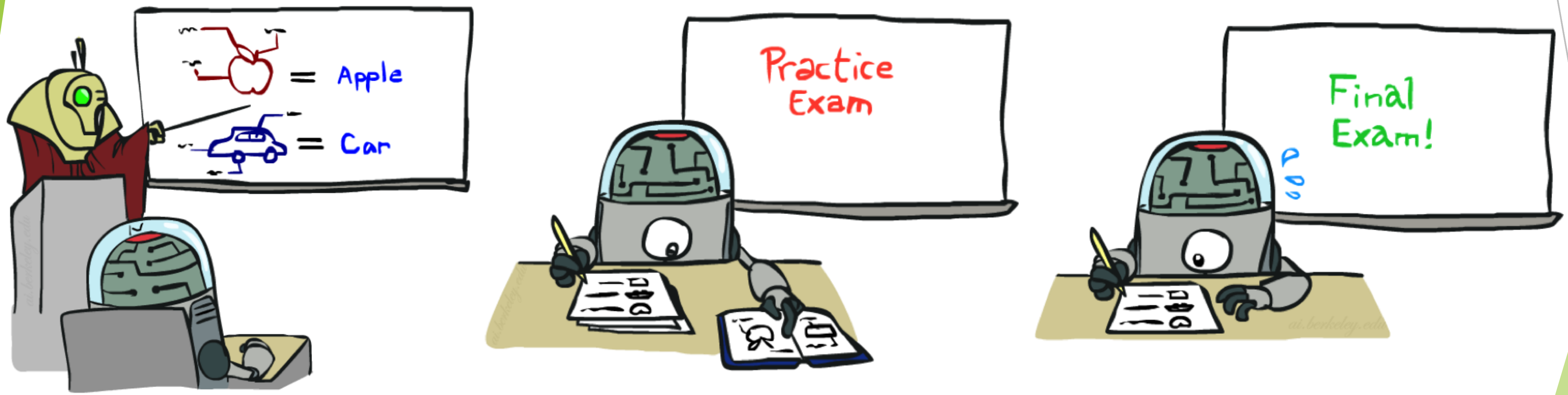
Machine Learning
Naïve Bayes



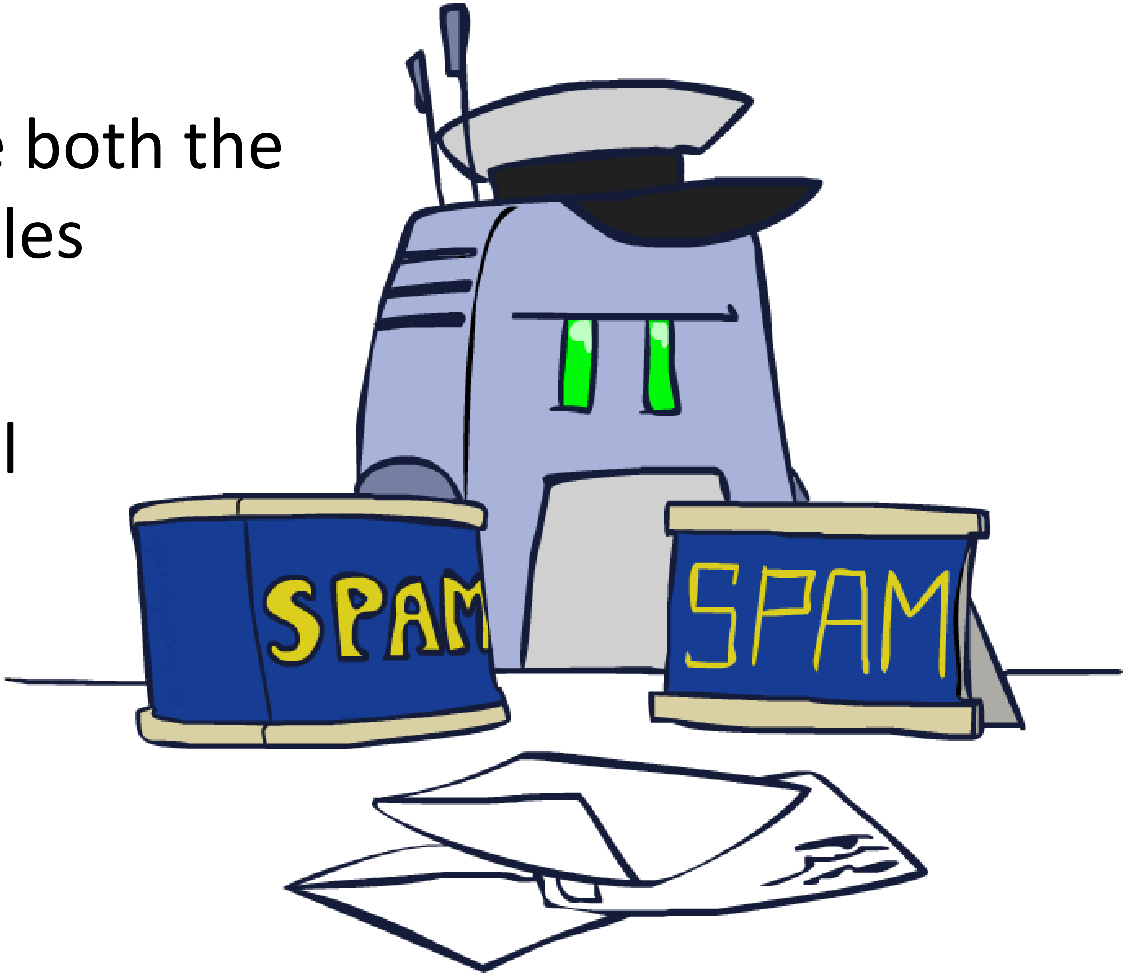
These slides are based on the slides created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley - http://ai.berkeley.edu. The artwork is by Ketrina Yim.

AIMA Chapters?

- Selected Sections from chapters 18 and 20.

Model-Based Classification

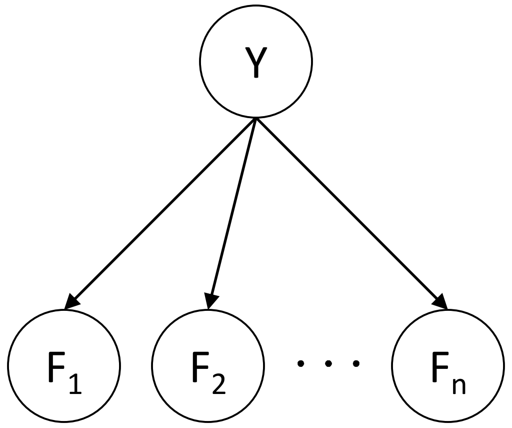
- Model-based approach
  - Build a model (e.g. Bayes' net) where both the label and features are random variables
  - Instantiate any observed features
  - Query for the distribution of the label conditioned on the features



General Naïve Bayes

- A general Naive Bayes model:

$$P(\mathsf{Y}, \mathsf{F}_1 \dots \mathsf{F}_n) = P(\mathsf{Y}) \prod_i P(\mathsf{F}_i|\mathsf{Y})$$



- We only have to specify how each feature depends on the class
- Total number of parameters is *linear* in n
- Model is very simplistic, but often works well anyway

General Naïve Bayes

What do we need in order to use Naïve Bayes?

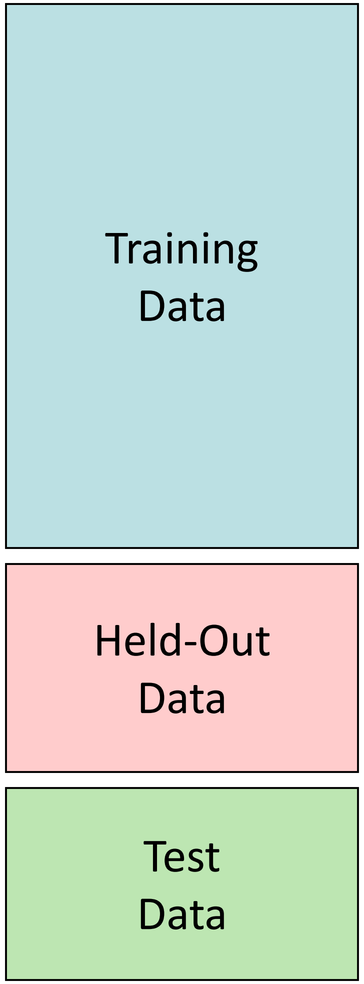
- Inference method (we saw this in the last lecture)

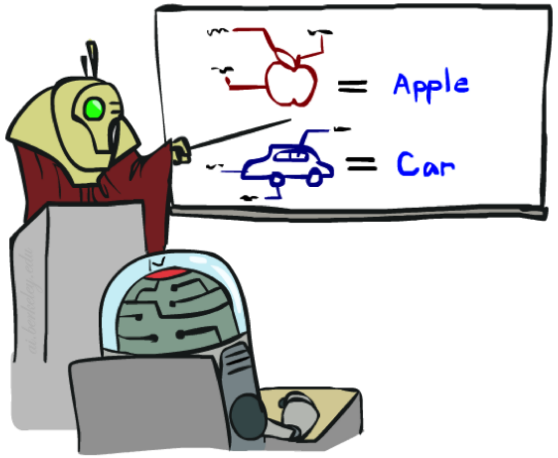
- Estimates of local conditional probability tables
  - P(Y), the prior over labels (P(spam), P(ham))
  - $P(F_i|Y)$ for each feature (evidence variable – P(free|spam)).
  - These probabilities are collectively called the *parameters* of the model: $\boldsymbol{\theta}$
  - Theses probabilities come from training data counts: let's see how to get them next.

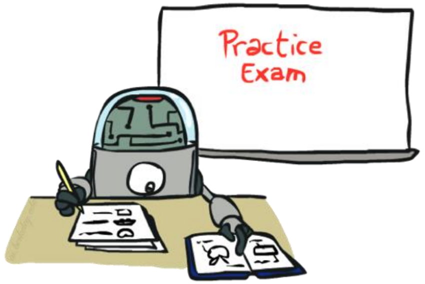
Training and Testing

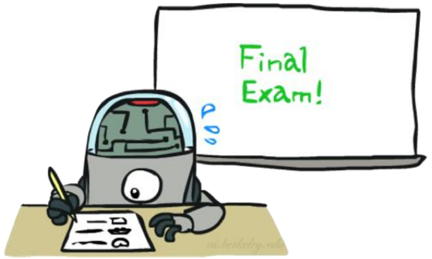
Data

- Data: labeled instances, e.g. emails marked spam/ham
- We divide our data into 3 sets:
  - Training set (60%)
  - Held out set / validation (20%)
  - Test set (20%)
- In Naïve Bayes we decide on the features
  - We learn their probabilities from the training set





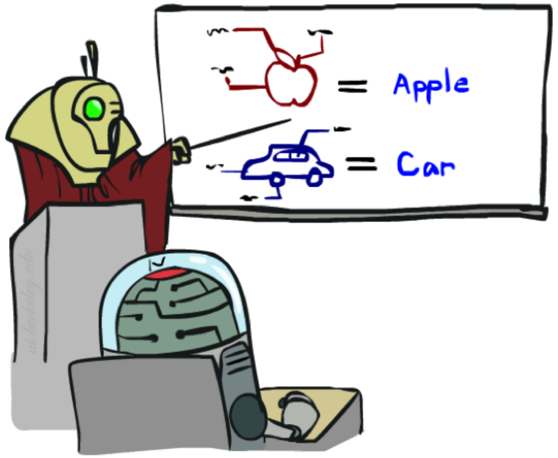




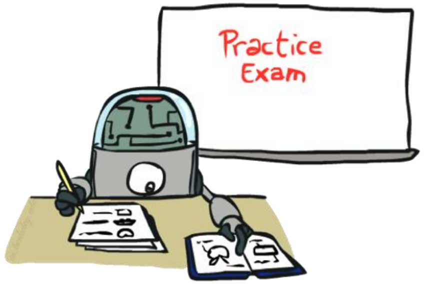
Training and Testing

- Experimentation cycle
  - Learn parameters (model probabilities) from training set
  - Tune hyperparameters on held-out set
  - Compute accuracy on test set
  - Very important: never "peek" at the test set!

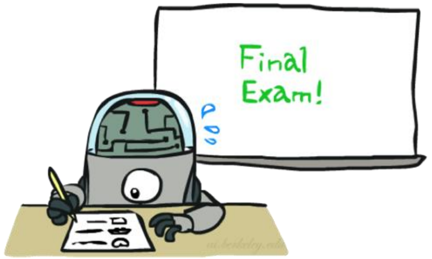
Training Data



Held-Out Data



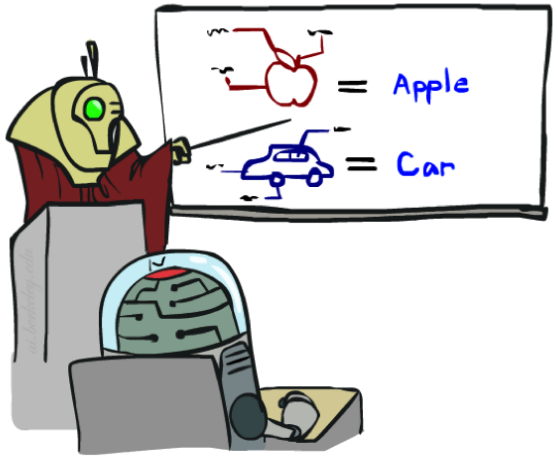
Test Data



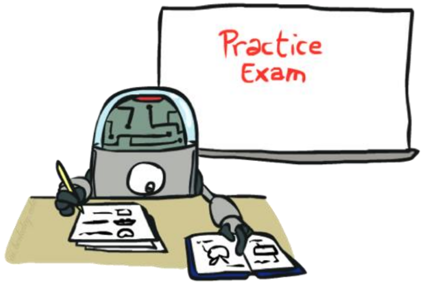
Training and Testing

- Evaluation
  - Accuracy: fraction of instances predicted correctly
  - Risk and utility: some ‘mistakes’ are worse than others (classifying an important email as spam)

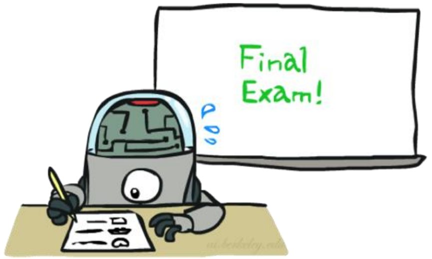
Training Data



Held-Out Data



Test Data



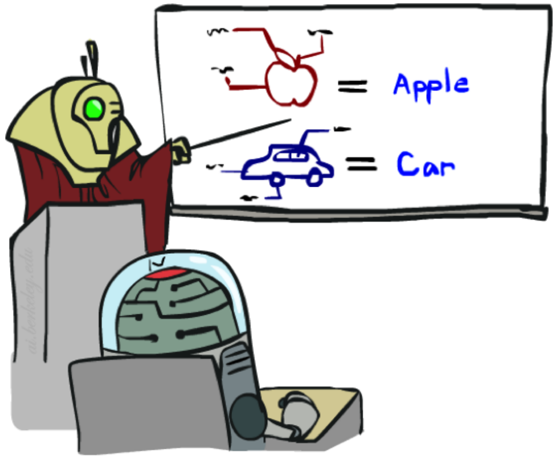
Training and Testing

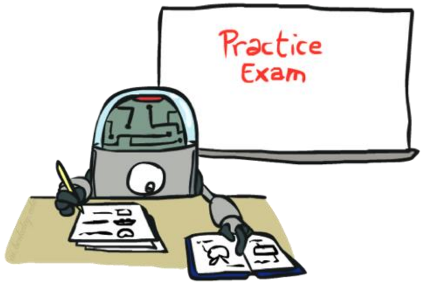
- Overfitting and generalization
  - We want a classifier which does well on test data
  - Overfitting: fitting the training data very closely, but not generalizing well
  - More on overfitting later

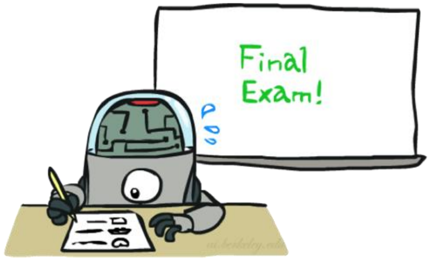
Training Data

Held-Out Data

Test Data





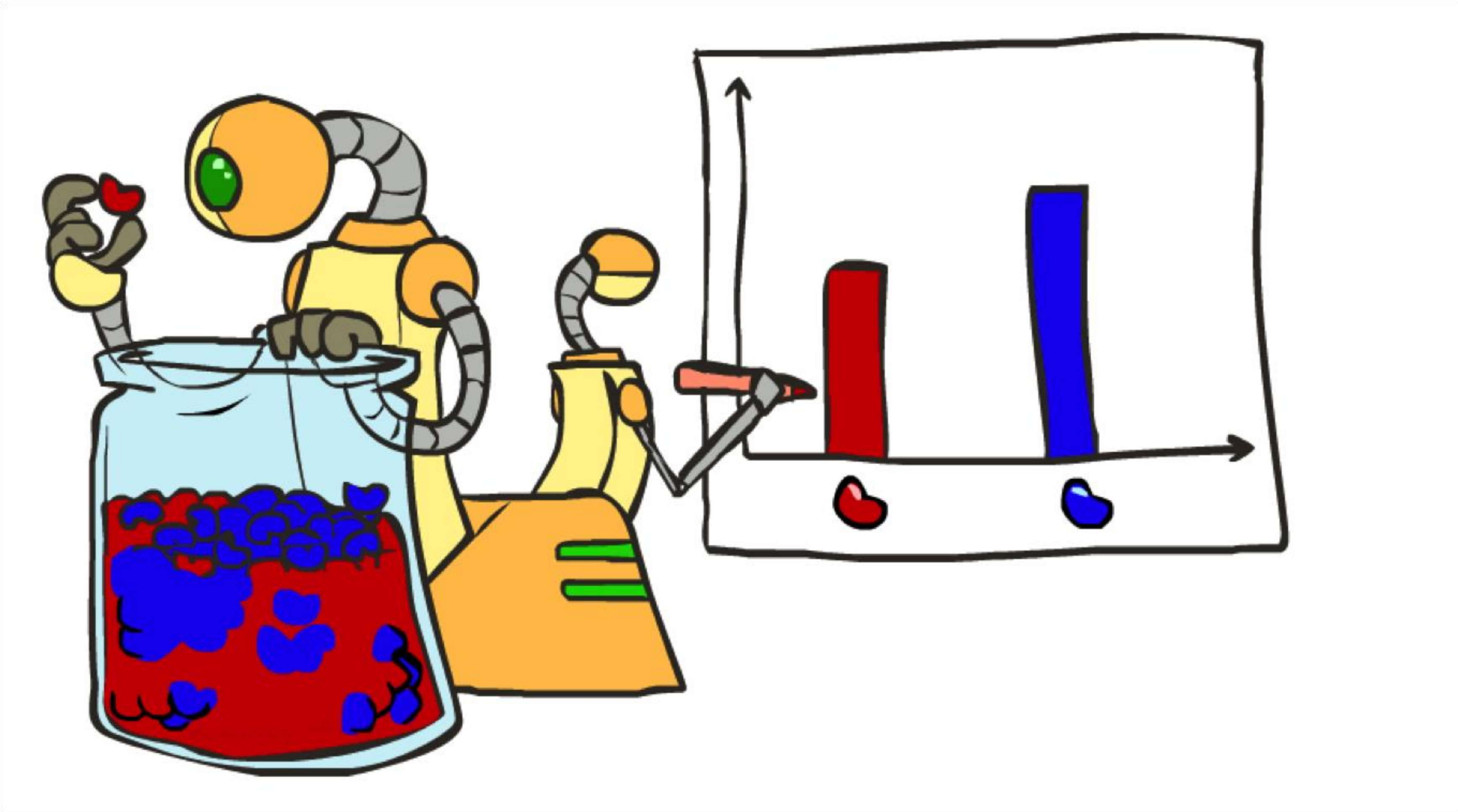


Supervised vs Unsupervised

Naïve Bayes is an example of a supervised learning algorithm.

This means that we give the algorithm a data set in which the "right answers" are given.

Parameter Estimation



Parameter Estimation

- Estimate the distribution of a random variable (candy flavor)
- Empirically*:* using the training data (learning!)

For each flavor, look at the empirical rate of that flavor:

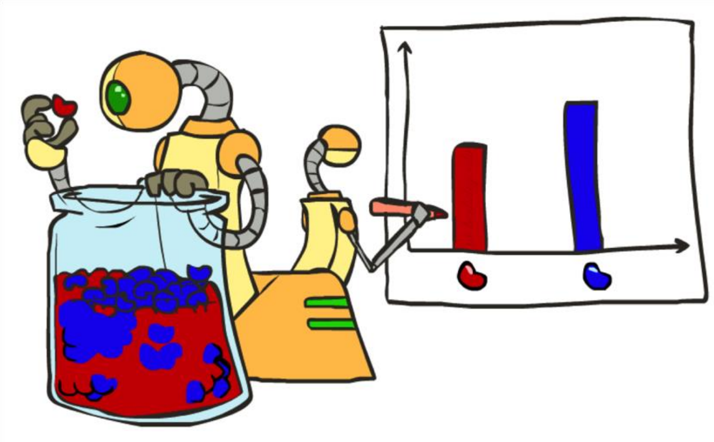
P(red) = $\frac{count(red)}{total\ samples}$

P(orange) = $\frac{count(orange)}{total\ samples}$

P(yellow) = $\frac{count(yellow)}{total\ samples}$

P(pink) = $\frac{count(pink)}{total\ samples}$

Parameter Estimation



- Estimate the distribution of a random variable (feature)
- Empirically: using the training data (learning!)
  - For each feature, look at the empirical rate of that feature:

$$P_{\text{ML}}(x) = \frac{\text{count}(x)}{\text{total samples}}$$

r r b

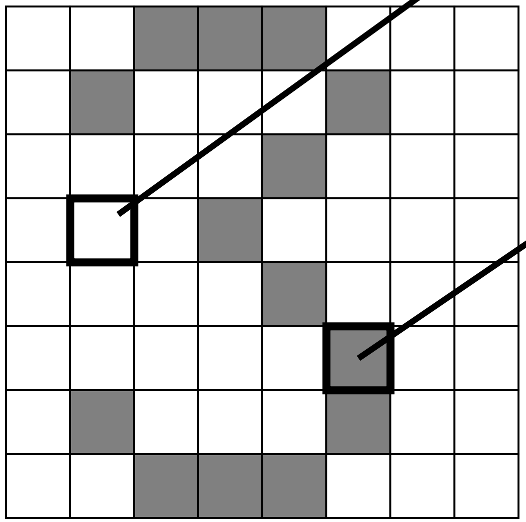
$$P_{\text{ML}}(\text{r}) = 2/3$$

  - This is the maximum likelihood estimate.
  - distribution estimate = observed distribution

Example: Digit Recognition

$P(Y)$

1	0.14
2	0.08
3	0.10
4	0.06
5	0.07
6	0.09
7	0.09
8	0.11
9	0.13
0	0.13



$P(F_{3,1} = on|Y)$

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

$P(F_{5,5} = on|Y)$

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

Example: Spam Filtering

- Bag-of-words model:
  - Simplified representation commonly used in Natural Language Processing
  - Message is modeled as an unordered collection (bag) of words
  - Grammar is also ignored

Example: Spam Filtering

- 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
...	

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

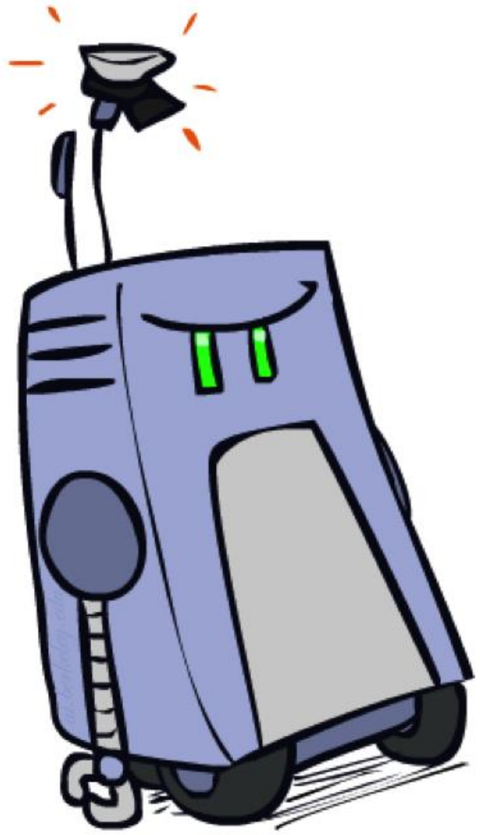
the :	0.0210
to :	0.0133
of :	0.0119
2002:	0.0110
with:	0.0108
from:	0.0107
and :	0.0105
a :	0.0100
...	

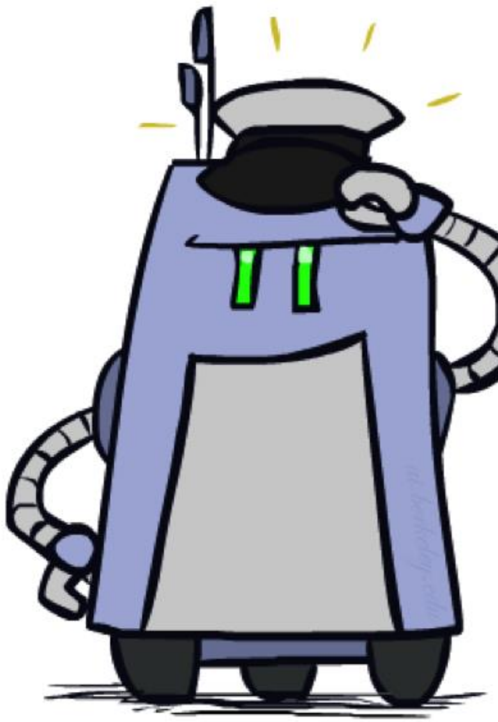
Example: Spam Filtering

Word	P(w|spam)	P(w|ham)	Tot Spam	Tot Ham
(prior)	0.33333	0.66666	0.33	0.67
Gary	0.00002	0.00021	0.05	0.95
would	0.00069	0.00084	0.04	0.96
you	0.00881	0.00304	0.10	0.90
like	0.00086	0.00083	0.11	0.89
to	0.01517	0.01339	0.12	0.88
lose	0.00008	0.00002	0.35	0.65
weight	0.00016	0.00002	0.81	0.19
while	0.00027	0.00027	0.81	0.19
you	0.00881	0.00304	0.93	0.07
sleep	0.00006	0.00001	0.99	0.01

P(spam | w) = 99%

Generalization and Overfitting





Generalization and Overfitting

P(orange) = 0

P(pink) = 0

P(red) = 0.7

P(yellow) = 0.3

Example: Overfitting

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

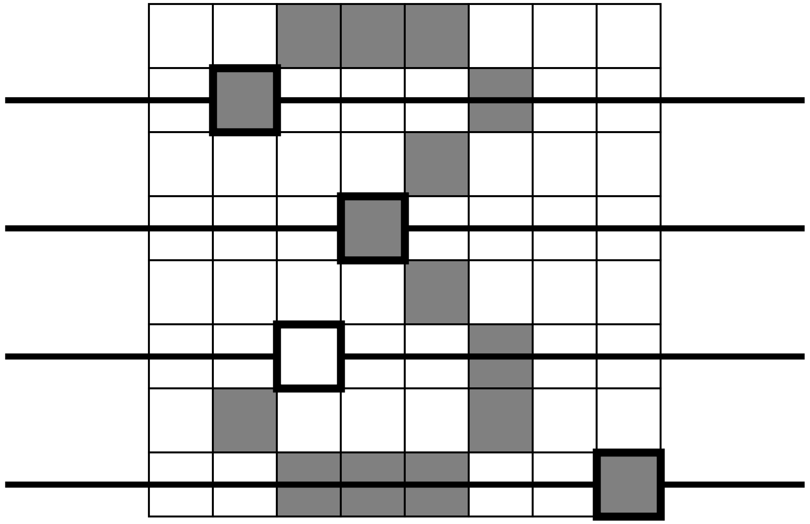
$P(C = 2) = 0.1$

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

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

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

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



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

$P(C = 3) = 0.1$

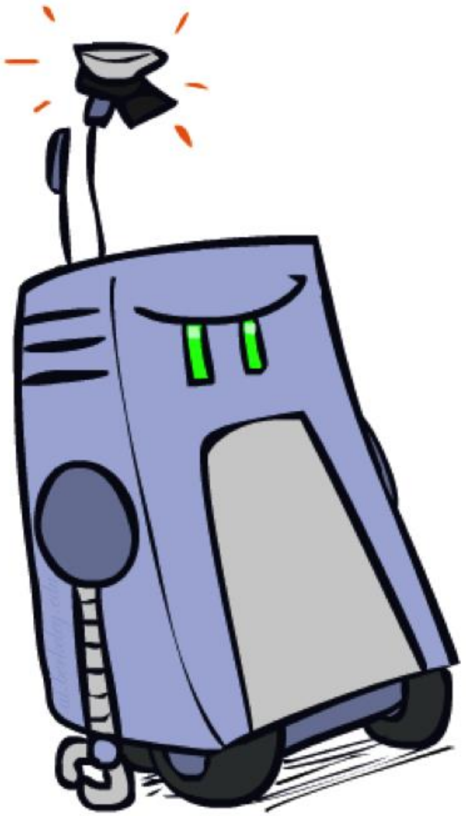
$P(\text{on}|C = 3) = 0.8$

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

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

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

2 wins!!



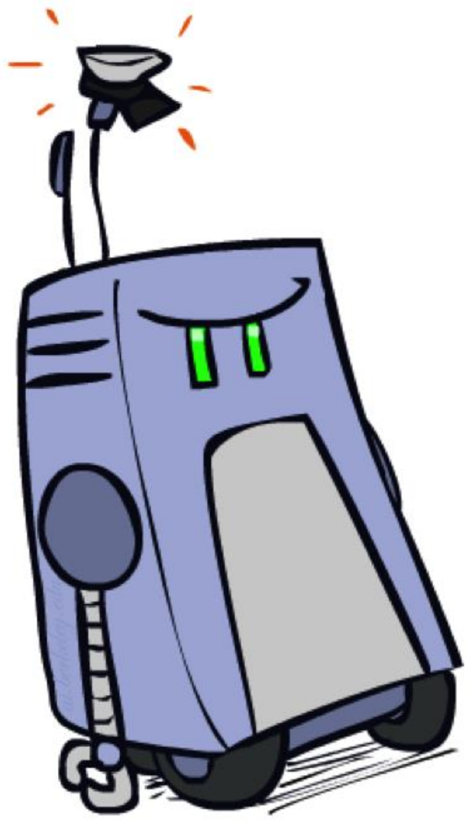
Example: Overfitting

- Posteriors determined by *relative* probabilities (odds ratios):

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

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

south-west : inf	screens : inf
nation : inf	minute : inf
morally : inf	guaranteed : inf
nicely : inf	\$205.00 : inf
extent : inf	delivery : inf
seriously : inf	signature : inf
...	...



What went wrong here?

Generalization and Overfitting

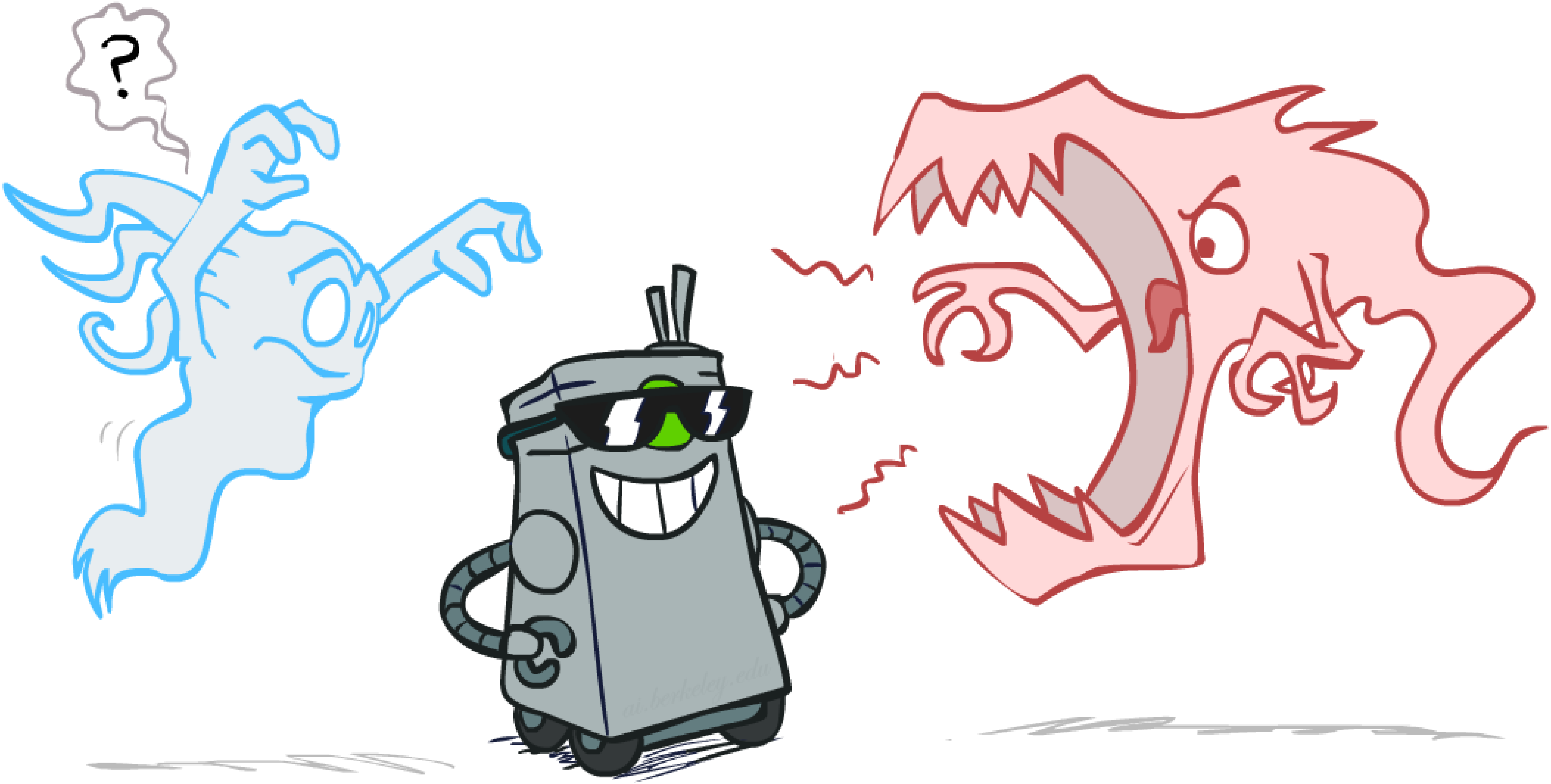
- Relative frequency parameters will overfit the training data!
  - Just because we never saw a 3 with pixel (15,15) on during training doesn't mean we won't see it at test time
  - Unlikely that every occurrence of "minute" is 100% spam
  - Unlikely that every occurrence of "seriously" is 100% ham
  - What about all the words that don't occur in the training set at all?
  - In general, we can't go around giving unseen events zero probability

Generalization and Overfitting

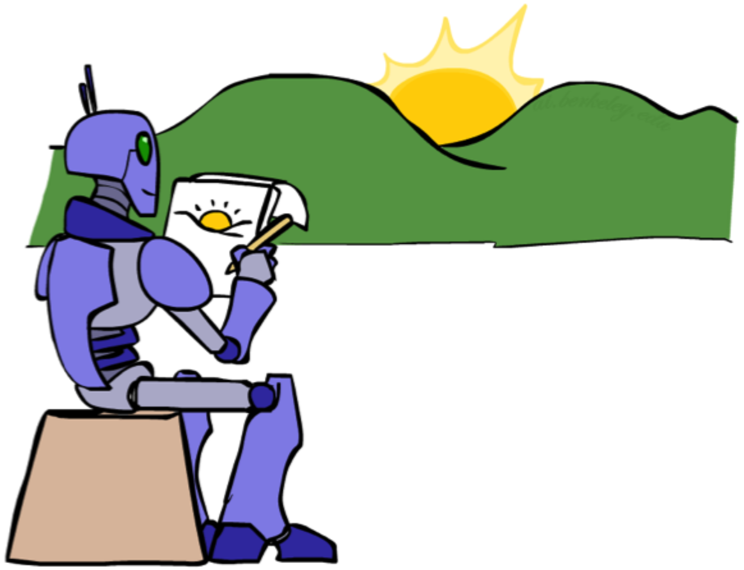
- As an extreme case, imagine using the entire email as the only feature
  - Would get the training data perfect (deterministic labeling)
  - Wouldn't *generalize* at all
  - Just making the bag-of-words assumption gives us some generalization, but isn't enough

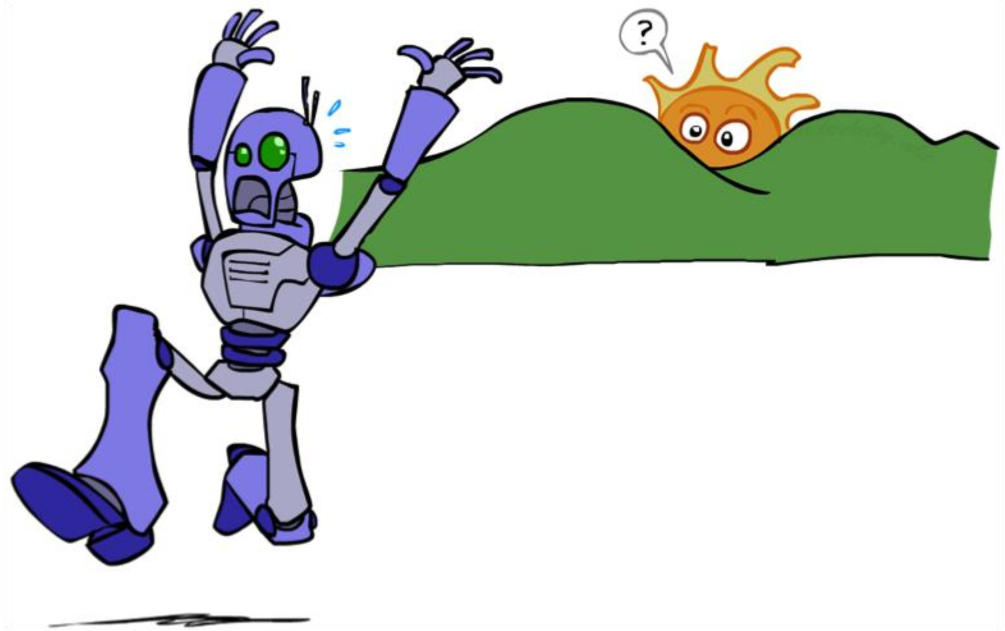
- To generalize better: we need to smooth or regularize the estimates

Smoothing



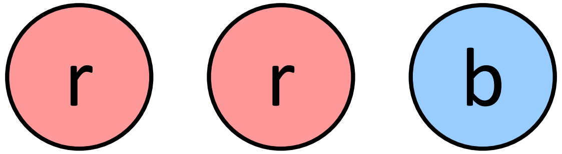
Unseen Events





Laplace Smoothing

- Laplace's estimate:
  - Pretend we saw every outcome once more than we actually did



$$P_{ML}(X) = \left\langle \frac{2}{3}, \frac{1}{3} \right\rangle$$

$$P_{LAP}(X) = \left\langle \frac{3}{5}, \frac{2}{5} \right\rangle$$

$$P_{LAP}(x) = \frac{c(x) + 1}{\sum_x [c(x) + 1]}$$

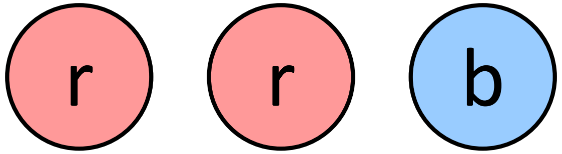
$$= \frac{c(x) + 1}{N + |X|}$$

Laplace Smoothing

- Laplace's estimate (extended):
  - Pretend we 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



$$P_{LAP,0}(X) = \left\langle \frac{2}{3}, \frac{1}{3} \right\rangle$$

$$P_{LAP,1}(X) = \left\langle \frac{3}{5}, \frac{2}{5} \right\rangle$$

$$P_{LAP,100}(X) = \left\langle \frac{102}{203}, \frac{101}{203} \right\rangle$$

Laplace Smoothing

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

Count(yellow) = 3

Count(red) = 6

Count(orange) = 1

Count(pink) = 0

N = 10

$P_{LAP,0}$(yellow) = 0.3

$P_{LAP,0}$(red) = 0.6

$P_{LAP,0}$(orange) = 0.1

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

Laplace Smoothing

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

Count(yellow) = 3

Count(red) = 6

Count(orange) = 1

Count(pink) = 0

N = 10

$P_{LAP,1}(\text{yellow}) = \frac{3+1}{10+4} = 0.29$

$P_{LAP,1}(\text{red}) = \frac{6+1}{10+4} = 0.5$

$P_{LAP,1}(\text{orange}) = \frac{1+1}{10+4} = 0.14$

$P_{LAP,1}(\text{pink}) = \frac{0+1}{10+4} = 0.07$

Laplace Smoothing

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

Count(red) = 6

Count(yellow) = 3

Count(orange) = 1

Count(pink) = 0

N = 10

$P_{LAP,2}(\text{yellow}) = \frac{3+2}{10+8} = 0.28$

$P_{LAP,2}(\text{red}) = \frac{6+2}{10+8} = 0.44$

$P_{LAP,2}(\text{orange}) = \frac{1+2}{10+8} = 0.17$

$P_{LAP,2}(\text{pink}) = \frac{0+2}{10+8} = 0.11$

Real NB: Smoothing

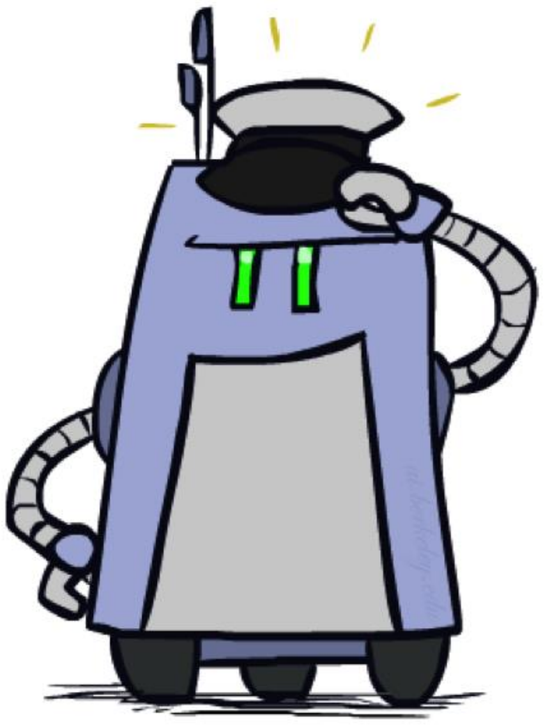
- For real classification problems, smoothing is critical
- New odds ratios:

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

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

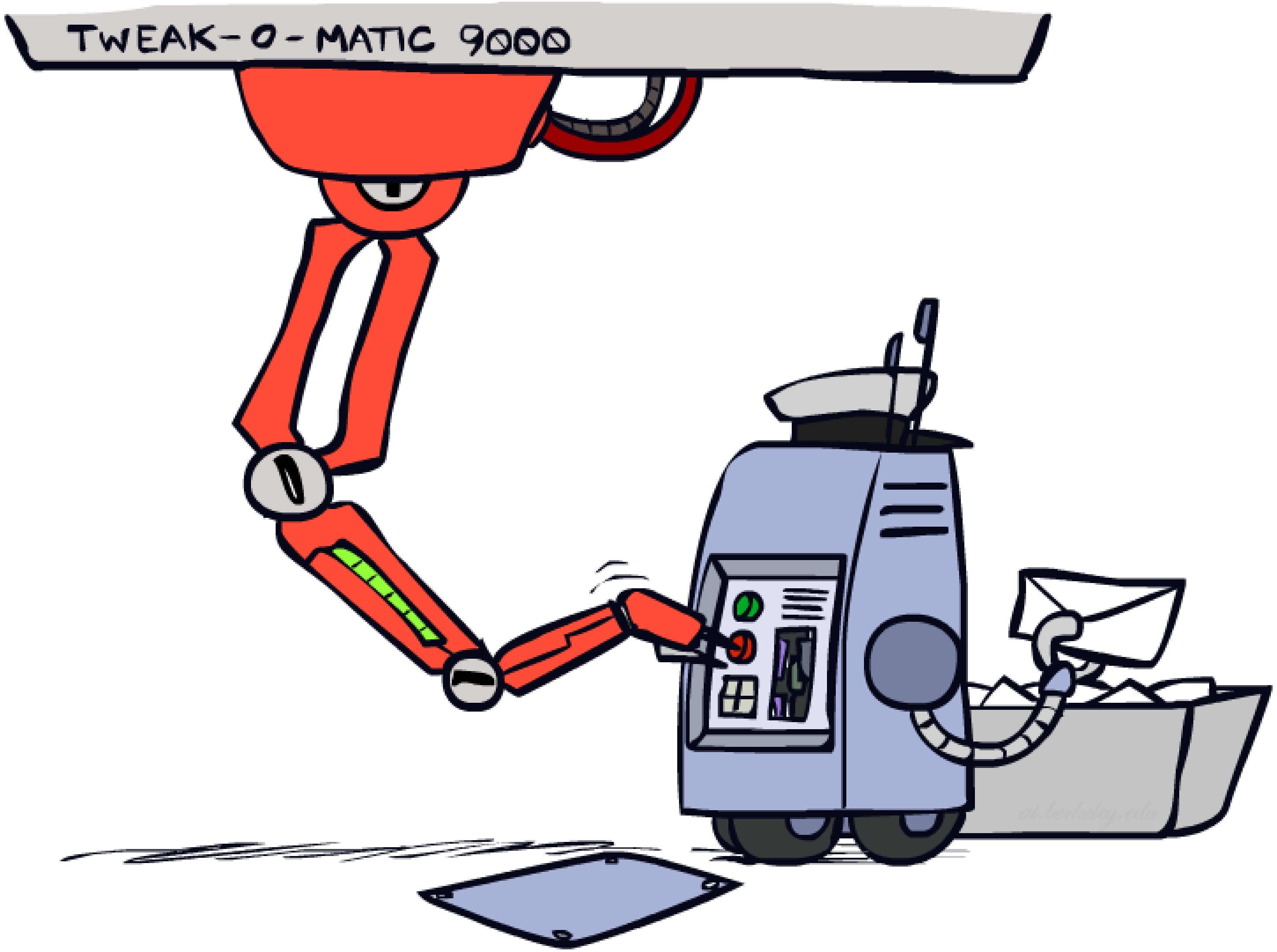
$$\frac{P(W|\text{spam})}{P(W|\text{ham})}$$

verdana : 28.8
Credit : 28.4
ORDER : 27.2
<FONT> : 26.9
money : 26.5
...



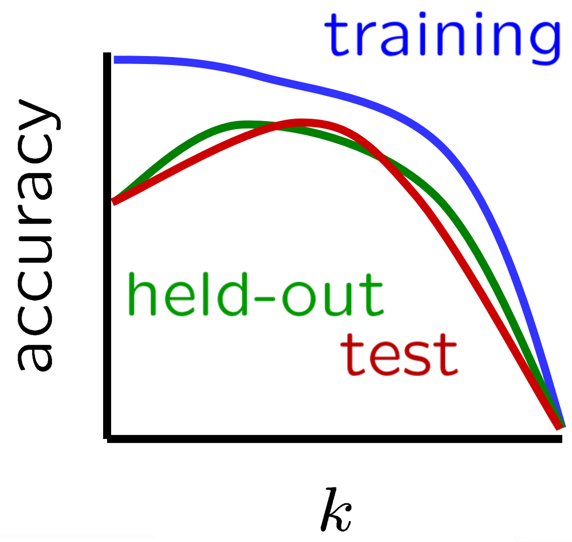
Do these make more sense?

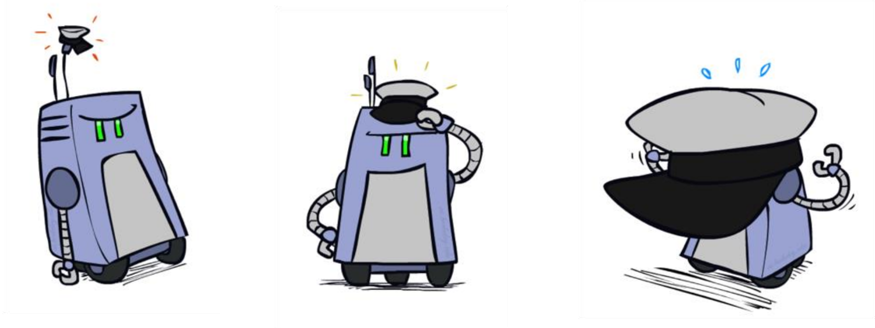
Tuning



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)
- What should we learn where?
  - Learn parameters from training data
  - Tune hyperparameters on different data
  - For each value of the hyperparameters (k), test on the held-out data
  - Choose the best value (best accuracy)
  - Do a final test on the test data





Summary

- Bayes rule lets us do diagnostic queries with causal probabilities
- The naïve Bayes assumption takes all features to be independent given the class label
- We can build classifiers out of a naïve Bayes model using training data
- Smoothing is important in real systems