Naive Bayes Classifier

Alice Oh
alice.oh@kaist.edu

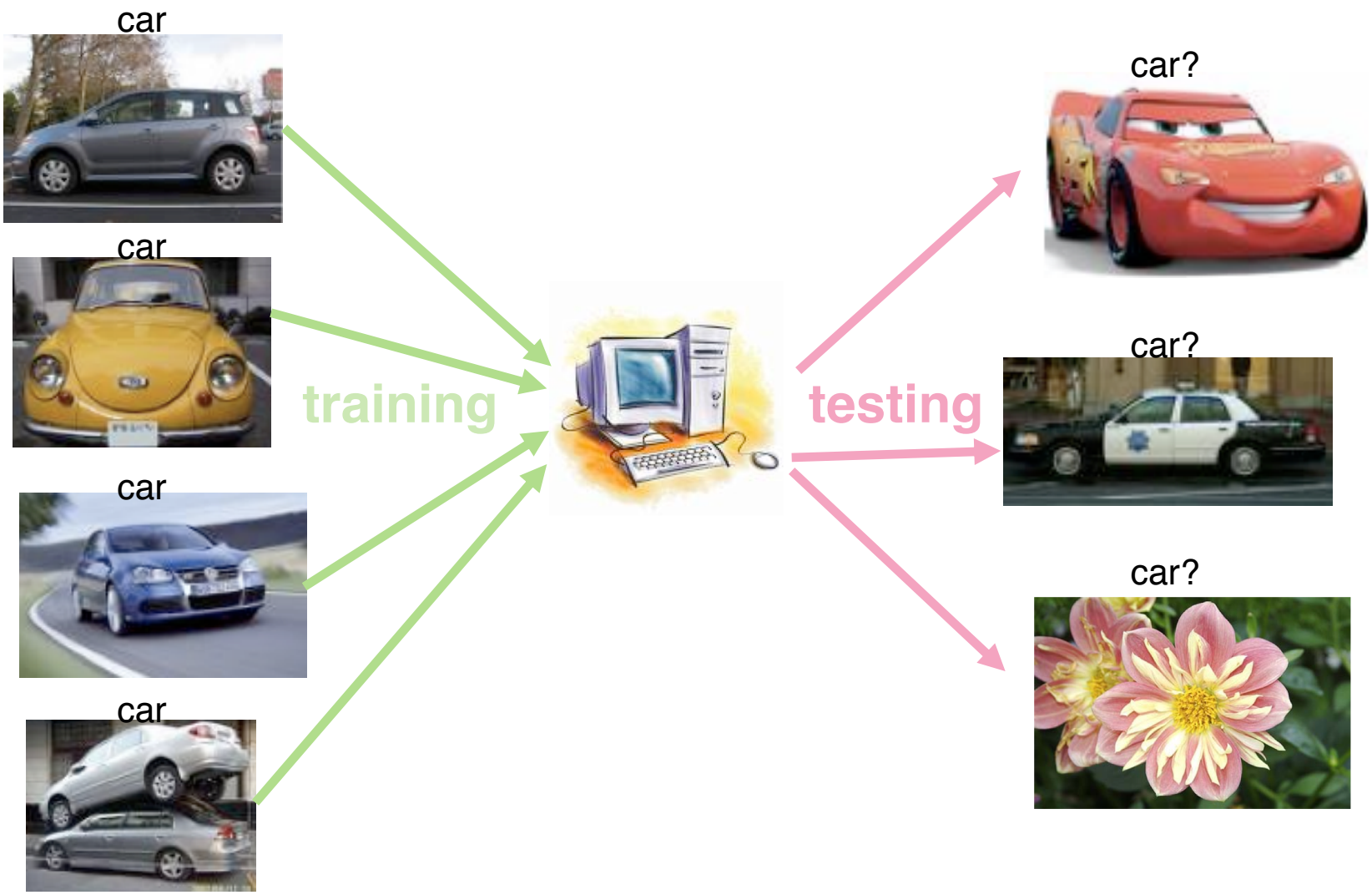
Machine Learning

u+i lab
users and information

A Problem for Machine Learning

u+i lab
users and information

Supervised Learning



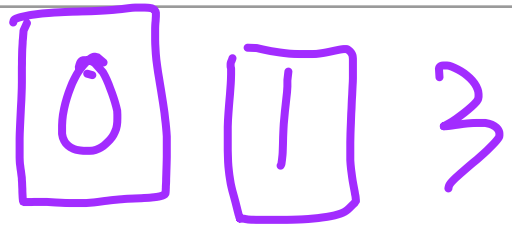
u+i lab
users and information

Features

- We use “features” to simplify the classification and clustering problems
- Features turn objects (images, documents, etc) into a set of numbers so that we can do computation over them
- What features can we use for the image classification? For spam email classification?

Problems

- Digit Recognition



- Email spam filtering

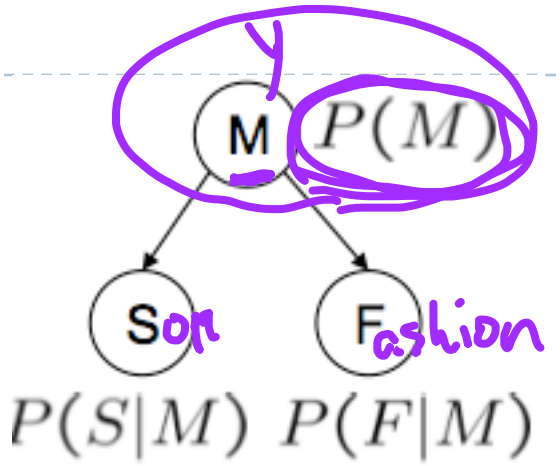
- How would you do this?

Classification

Is he married?

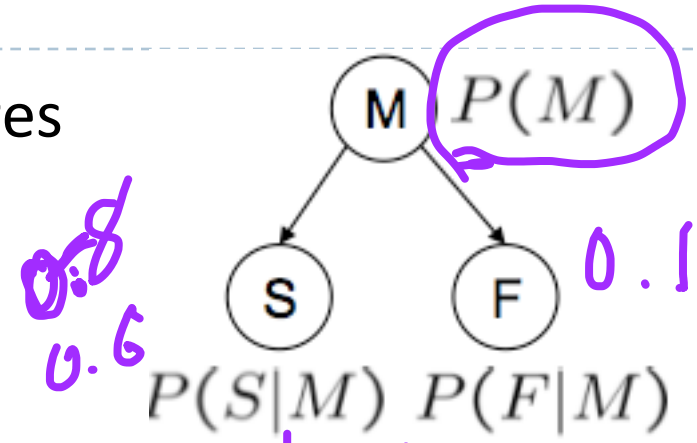
Simple Classification

- Simple example: two binary features



Simple Classification

- Simple example: two binary features



$P(m|s,f)$ ← direct estimate

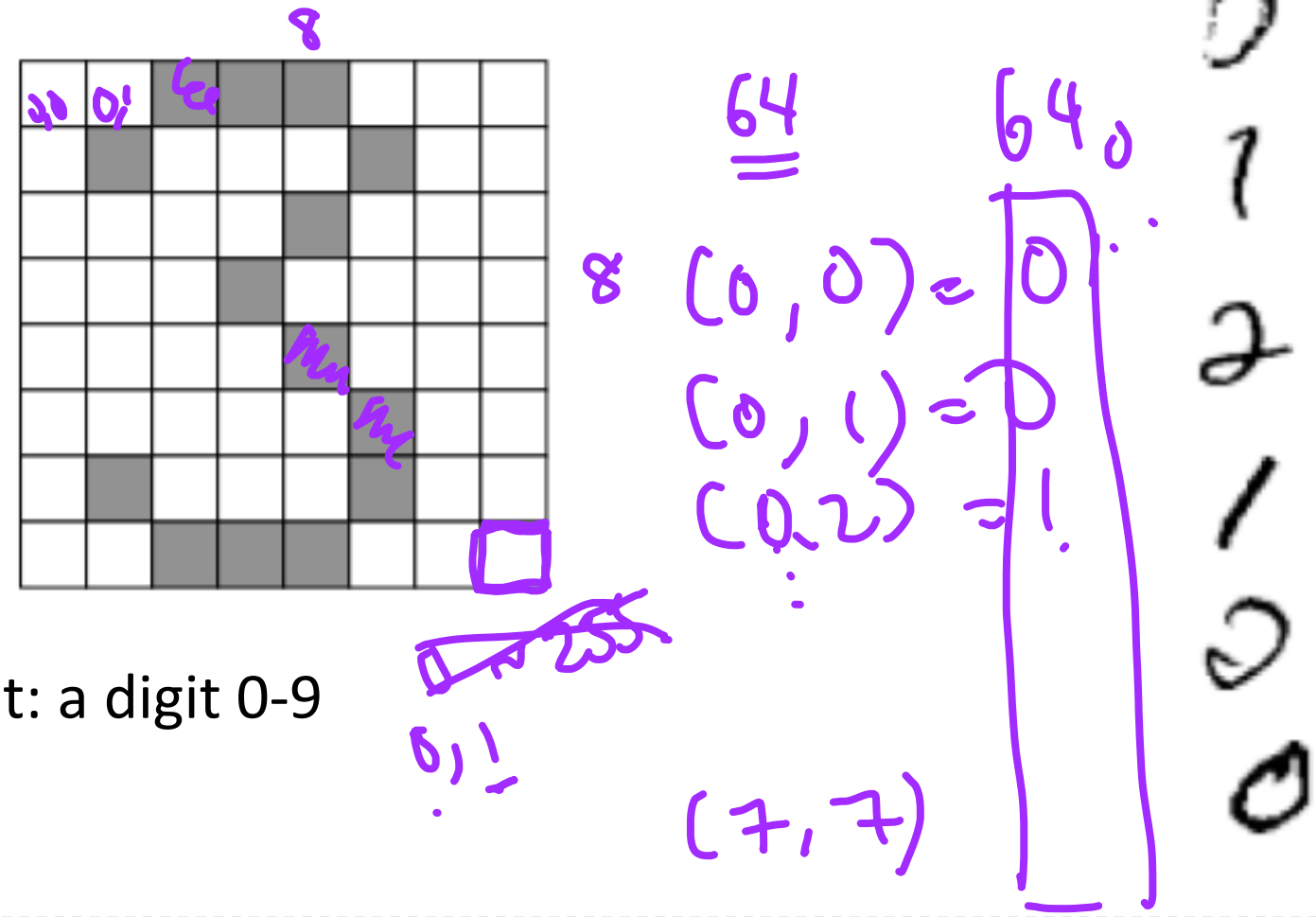
$$P(m|s,f) = \frac{P(s,f|m)P(m)}{P(s,f)}$$ ← Bayes estimate (no assumptions)

$$P(m|s,f) = \frac{P(s|m)P(f|m)P(m)}{P(s,f)}$$ ← Conditional independence

$$+ \begin{cases} P(m,s,f) = P(s|m)P(f|m)P(m) \\ P(\bar{m},s,f) = P(s|\bar{m})P(f|\bar{m})P(\bar{m}) \end{cases}$$

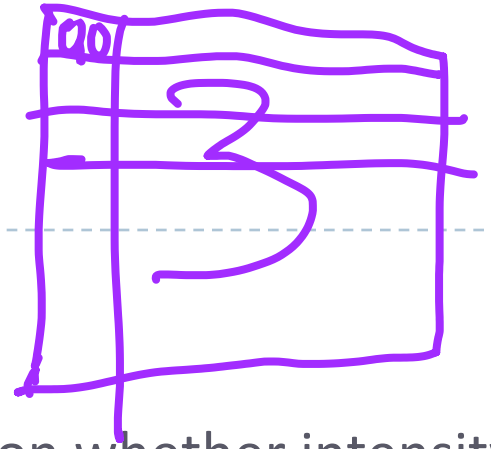
Digit Recognizer

- Input: pixel grids



- Output: a digit 0-9

Naïve Bayes for Digits



- Simple version:
  - One feature Fij for each grid position <i,j>
  - Possible feature values are on / off, based on whether intensity is more or less than 0.5 in underlying image
  - Each input maps to a feature vector, e.g.

$$\rightarrow \langle F_{0,0} = 0 \;\; F_{0,1} = 0 \;\; F_{0,2} = 1 \;\; F_{0,3} = 1 \;\; F_{0,4} = 0 \;\; \ldots F_{15,15} = 0 \rangle$$

  - Here: lots of features, each is binary
- Naïve Bayes model:

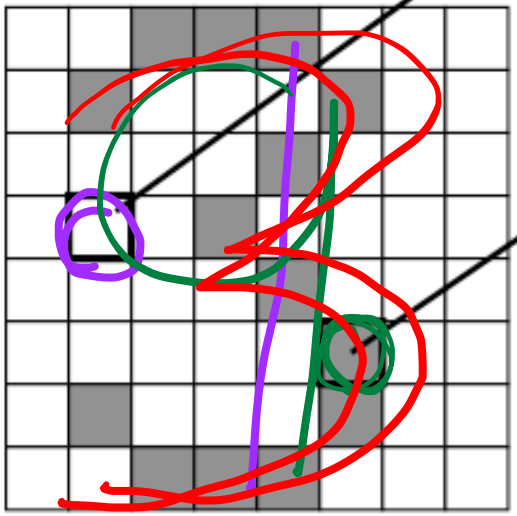
$$P(Y|F_{0,0} \ldots F_{15,15}) \propto P(Y) \prod_{i,j} P(F_{i,j}|Y)$$

- What do we need to learn?

Example: CPTs

$P(Y)$

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



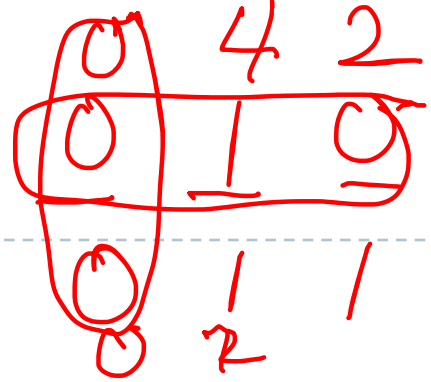
$P(F_{3,1} = on|Y)$

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: CPTs



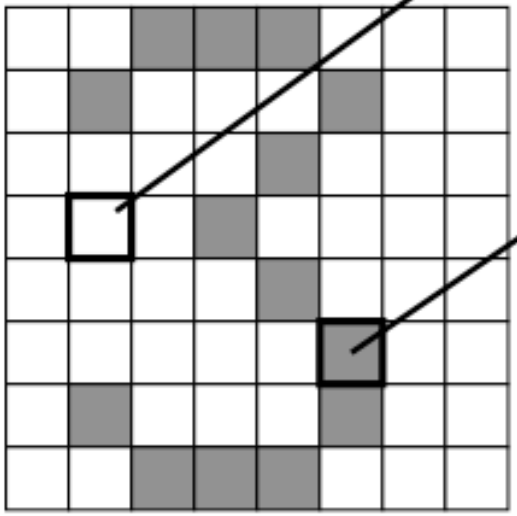
Prior

$P(Y)$

uniform distribution

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

0.8
0.1



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

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

When would this P(Y) table have non-uniform values?

Example: CPTs

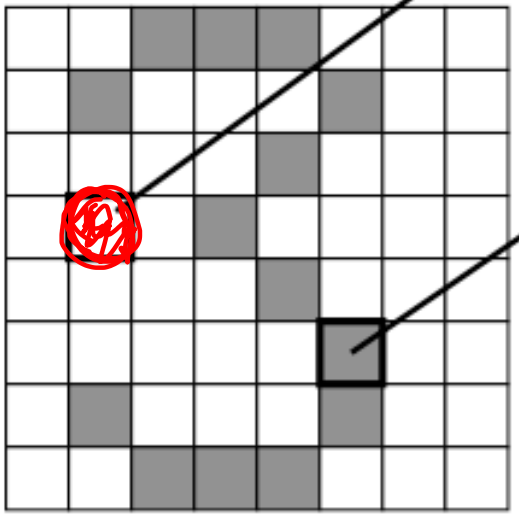
Empirical. From Data

15 0.015

100 x 10 = 1000

$P(Y)$

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



A: Training Data.

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

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

0.1

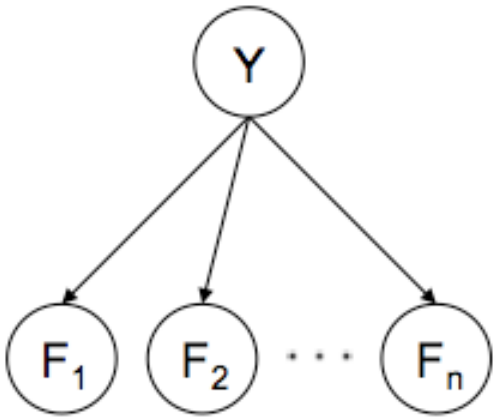
$P(F_{5,5} = on|Y)$

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

How would you get these P(F|Y) tables?

General Naïve Bayes

- A general *naive Bayes model:*



General Naïve Bayes

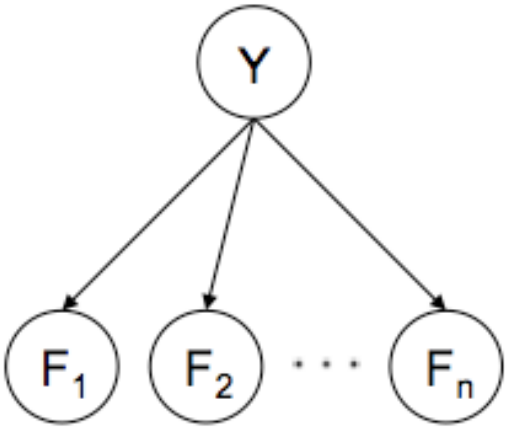
- A general *naive Bayes* *model:*

|Y| x |F|^n parameters

$$P(Y, F_1 \dots F_n) = P(Y) \prod_i P(F_i|Y)$$

|Y| parameters

n x |F| x |Y| parameters



- *We only specify how each feature depends on the class*
- *Total number of parameters is linear in n*

What is |Y| in digit recognition? |F|? n?

Inference for Naïve Bayes

- Goal: compute posterior over classes

Inference for Naïve Bayes

- Goal: compute posterior over classes

$$P(Y|F) \propto P(Y) \prod_i P(F_i|Y)$$

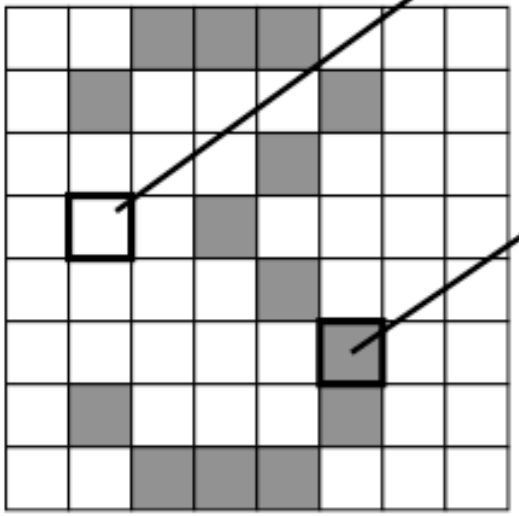
General Naïve Bayes

- What do we need in order to use naïve Bayes?
  - Inference
    - Start with a bunch of conditionals, P(Y) and the $P(F_i|Y)$ tables
    - Use standard inference to compute $P(Y|F_1 \ldots F_n)$
    - Nothing new here
  - Estimates of local conditional probability tables
    - P(Y), the prior over labels
    - $P(F_i|Y)$ for each feature (evidence variable)
    - These probabilities are collectively called the *parameters* of the model and denoted by *theta*
    - Up until now, we assumed these appeared by magic, but…
    - …they typically come from training data: we'll look at this now

Example: CPTs

$P(Y)$

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



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

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

Parameter Estimation

- Estimating distribution of random variables like X or X | Y
- *Empirically:* use training data
  - For each outcome x, look at the empirical rate of that value:

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

$P_{\text{ML}}(\text{r}) = 1/3$

  - This is the estimate that maximizes the likelihood of the data

$$L(x, \theta) = \prod_i P_\theta(x_i)$$

- *Elicitation*: ask a human!
  - Usually need domain experts, and sophisticated ways of eliciting probabilities (e.g. betting games)
  - Trouble calibrating

A Spam Filter

- Naïve Bayes spam filter
- Data:
  - Collection of emails, labeled spam or ham
  - Note: someone has to hand label all this data!
  - Split into training, held-out, test sets
- Classifiers
  - Learn on the training set
  - (Tune it on a held-out set)
  - Test it on new emails

Dear Sir. First, I must solicit your confidence in this transaction, this is by virture of its nature as being utterly confidencial 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 anddecided 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.

Naïve Bayes for Text

- Bag-of-Words Naïve Bayes:
  - Predict unknown class label (spam vs. ham)
  - Assume evidence features (e.g. the words) are independent
  - Warning: subtly different assumptions than before!
- Generative model

$$P(C, W_1 \dots W_n) = P(Y) \prod_i P(W_i|C)$$

Word at position i, not i^{th} word in the dictionary!

- Bag-of-words
  - Usually, each variable gets its own conditional probability distribution P(F|Y)
  - In a bag-of-words model
    - Each position is identically distributed
    - All positions share the same conditional probs P(W|C)
    - Why make this assumption?

Example: Spam Filtering

- Model: $P(C, W_1 \dots W_n) = P(C) \prod_i P(W_i|C)$

- What are the parameters?

$P(C)$

ham : 0.66
spam: 0.33

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

the : 0.0156
to : 0.0153
only : 0.0115
free : 0.0095
win : 0.0093
a : 0.0086
You : 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
...

Overfitting

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

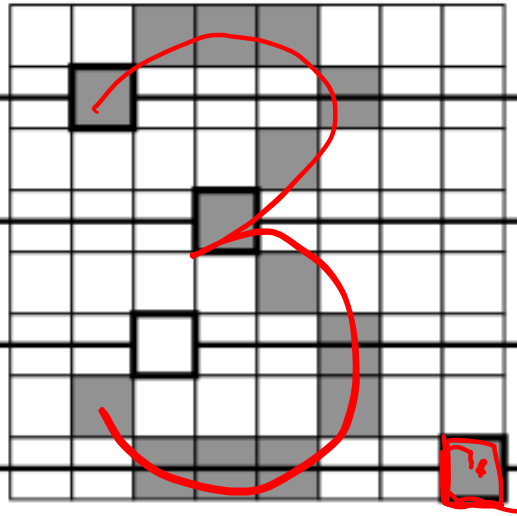
$P(C = 2) = 0.1$

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

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

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

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



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

$P(C = 3) = 0.1$

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

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

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

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

2 wins!!

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
- As an extreme case, imagine using the entire email as the only feature
  - Would get the training data perfect (if 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

Estimation: Smoothing

- Problems with maximum likelihood estimates:
  - If I flip a coin once, and it's heads, what's the estimate for P(heads)?
  - What if I flip 10 times with 8 heads?
  - What if I flip 10M times with 8M heads?
- Basic idea:
  - We have some prior expectation about parameters (here, the probability of heads)
  - Given little evidence, we should skew towards our prior
  - Given a lot of evidence, we should listen to the data

Laplace Smoothing

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

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

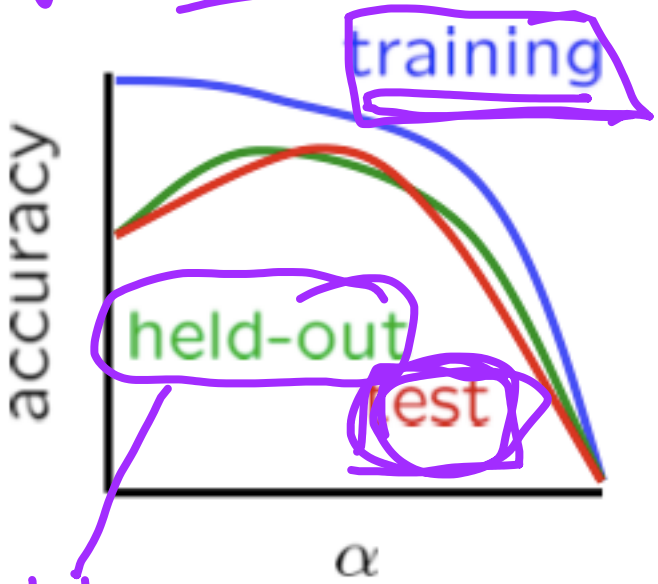
$$= \frac{c(x) + 1}{N + |X|}$$

$$P_{ML}(X) =$$

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

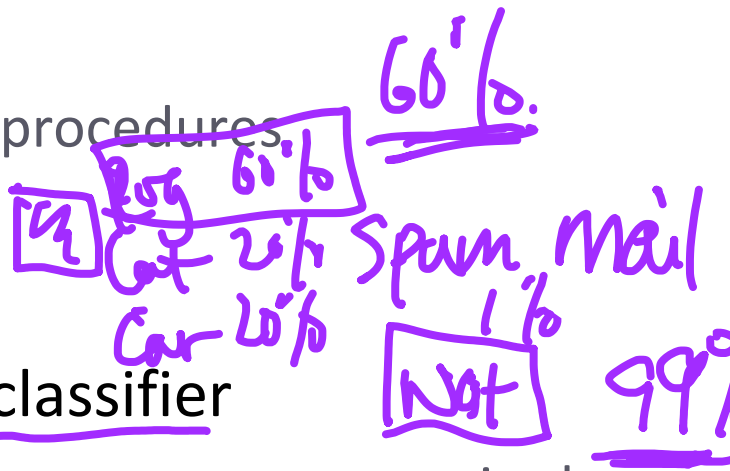
Tuning on Held-Out Data

- Now we've got two kinds of unknowns
  - Parameters: the probabilities P(Y|X), P(Y)
  - Hyperparameters, like the amount of smoothing to do: k, alpha
- Where to learn?
  - Learn parameters from training data
  - Must tune hyperparameters on different data
    - Why?
  - 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



Baselines

- First task: get a baseline
  - Baselines are very simple "straw man" procedures
  - Help determine how hard the task is
  - Help know what a "good" accuracy is
- Weak baseline: most frequent label classifier
  - Gives all test instances whatever label was most common in the training set
  - E.g. for spam filtering, might label everything as ham
  - Accuracy might be very high if the problem is skewed
- For real research, usually use previous work as a (strong) baseline



What to do about Errors

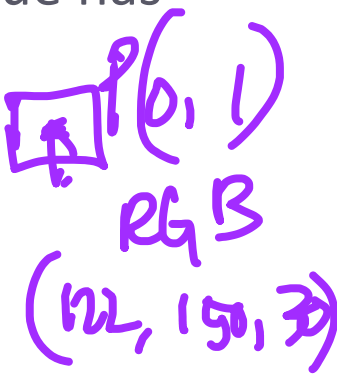
- Problem: there's still spam in your inbox
- Need more features–words aren't enough!
  - Have you emailed the sender before?
  - Have 1000 other people just gotten the same email?
  - Is the sending information consistent?
  - Is the email in ALL CAPS?
  - Do inline URLs point where they say they point?
  - Does the email address you by (your) name?
- Naïve Bayes models can incorporate a variety of features, but tend to do best in homogeneous cases (e.g. all features are word occurrences)

BOW

IND

Features

- A feature is a function that signals a property of the input
  - **Naïve Bayes:** features are random variables & each value has conditional probabilities given the label.
  - **Most classifiers:** features are real-valued functions
  - Common special cases:
    - **Indicator features** take values 0 and 1 or -1 and 1
    - **Count features** return non-negative integers
- Features are anything you can think of for which you can write code to evaluate on an input
  - Many are cheap, but some are expensive to compute
  - Can even be the output of another classifier or model
  - Domain knowledge goes here!



Summary

- Bayes rule lets us do diagnostic queries with conditional 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 estimates is important in real systems