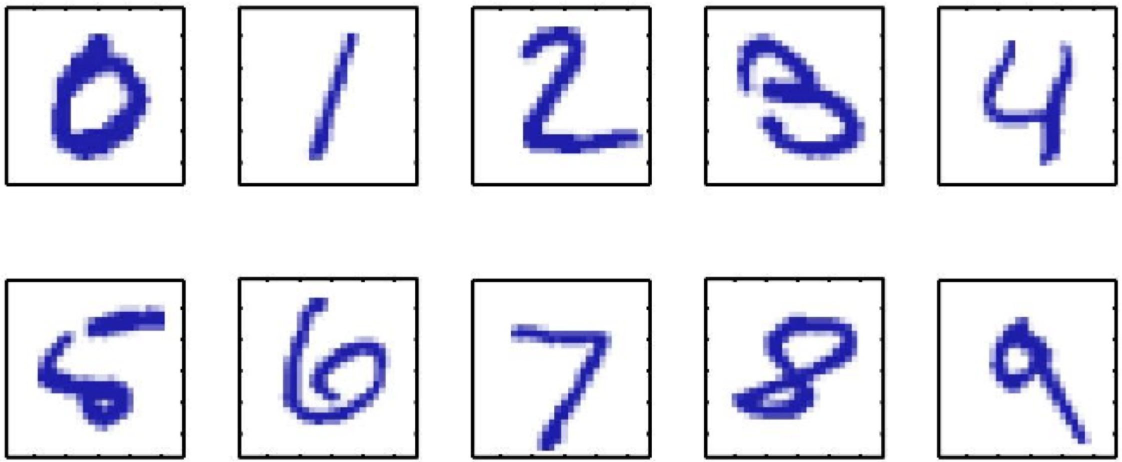
Machine Learning: algorithms that use "experience" to improve their performance

We use machine learning in situations where it is very challenging (or impossible) to define the rules by hand: e.g.

- face detection
- speech recognition
- stock prediction
- driving a car
- medical diagnosis
- figure out if a credit card purchase is fraudulent

Example 1: hand-written digit recognition



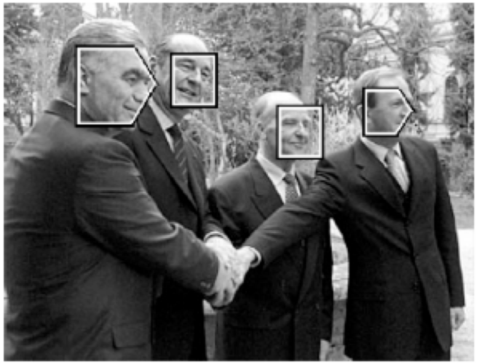
Images are 28 x 28 pixels

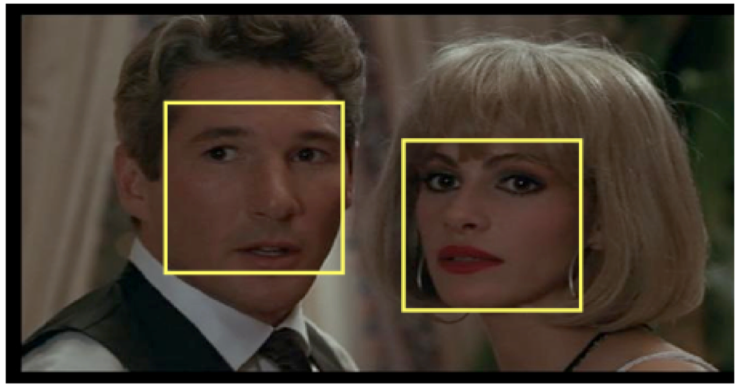
Represent input image as a vector $\mathbf{x} \in \mathbb{R}^{784}$

Learn a classifier $f(\mathbf{x})$ such that,

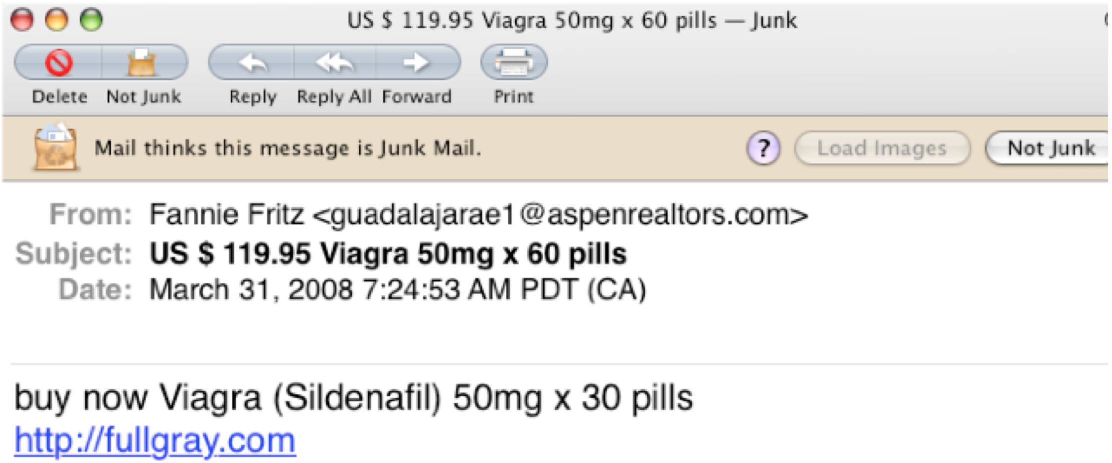
$$f : \mathbf{x} \rightarrow \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

Example 2: Face detection





Example 3: Spam detection



Example 4: Machine translation

What is the anticipated cost of collecting fees under the new proposal?

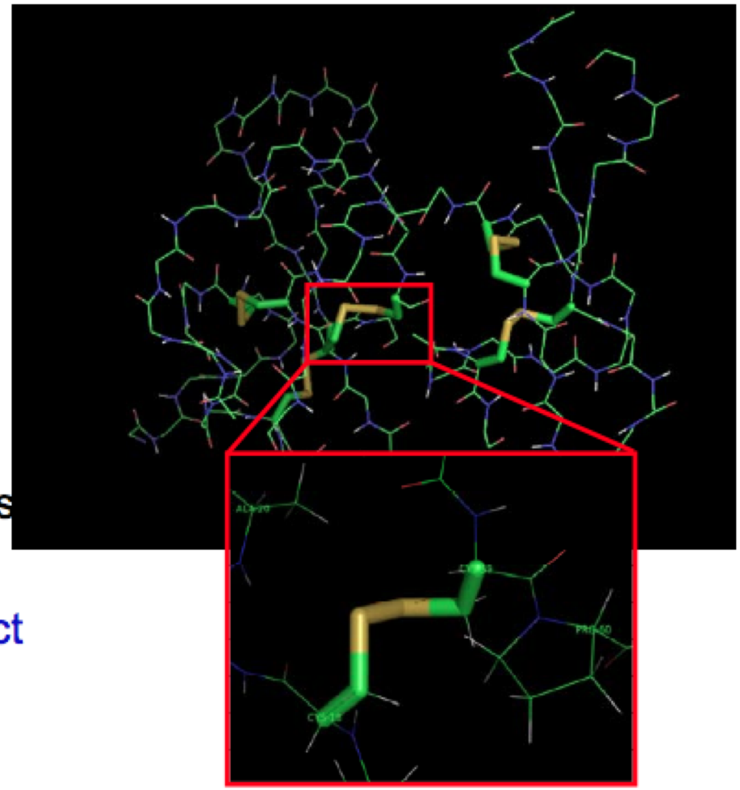
Example 5: Computational biology

$\mathbf{x}$

AVITGACERDLQCG
KGTCCAVSLWIKSV
RVCTPVGTSGEDCH
PASHKIPFSGQRMH
HTCPCAPNLACVQT
SPKKFKCLSK

Protein Structure and Disulfide Bridges

$\mathbf{y}$



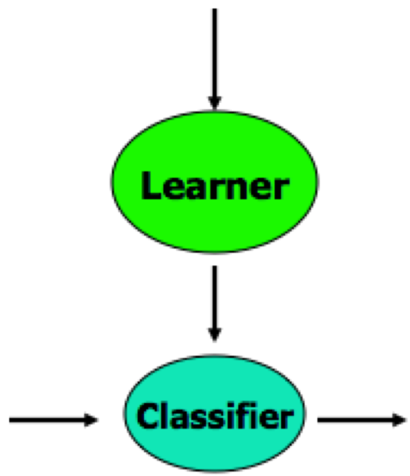
Regression task: given sequence predict 3D structure

Protein: 1IMT

- Given "labeled data"

Temp.	BP.	Sore Throat	...	Colour	diseaseX
35	95	Y	...	Pale	No
22	110	N	...	Clear	Yes
:	:			:	:
10	87	N	...	Pale	No

- Learn CLASSIFIER, that can predict label of *NEW* instance



Temp	BP	Sore-Throat	...	Color	diseaseX
32	90	N	...	Pale	?

diseaseX
No

Spam Detection Using
Naïve Bayes Classification

Jonathan Lee and Varun Mahadevan

Programming Project: Spam Filter

On homework 3, you'll be asked to implement a Naive Bayes classifier for classifying emails as either spam or ham (= nonspam).

Spam vs. Ham

In the past, the bane of any email user's existence

Less of a problem for consumers now, because spam filters have gotten really good

Easy for humans to identify spam, but not necessarily easy for computers

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, Iknow 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.

The spam classification problem

Input: collection of emails, already labeled *spam* or *ham*

Someone has to label these by hand

Called the ***training data***

Use this data to train a model that can predict whether an email is spam or ham

Many approaches: we'll use a Naïve Bayes classifier.

Test your model on emails whose label isn't provided, and see how well it does

Called the ***test data***

Naïve Bayes in the real world

One of the oldest, simplest methods for *classification*

Powerful and still used in the real world/industry

- Identifying credit card fraud
- Identifying fake Amazon reviews
- Identifying vandalism on Wikipedia
- **Still** used (with modifications) by Gmail to prevent spam
- Facial recognition
- Categorizing Google News articles
- Even used for medical diagnosis!

Naïve Bayes in theory

You will use what we've learned recently. Specifically:

Conditional Probability

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

Bayes' Theorem

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

Law of Total Probability

$$P(A) = \sum_n P(A|B_n)P(B_n)$$

Chain Rule

$$P(A_1, \dots, A_n) \\ = P(A_1)\, P(A_2|A_1) \dots P(A_n|A_{n-1} \dots A_1)$$

Conditional Independence of *A* and *B*, given *C*

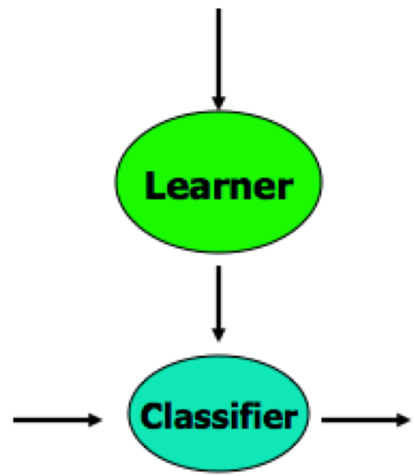
$$P(A \cap B|C) = P(A|C)P(B|C)$$
$$P(A|B \cap C) = P(A|C)$$

training examples

- Given "labeled data"

Temp.	BP.	Sore Throat	…	Colour	diseaseX
35	95	Y	…	Pale	No
22	110	N	…	Clear	Yes
:	:			:	:
10	87	N	…	Pale	No

- Learn CLASSIFIER, that can predict label of *NEW* instance



Temp	BP	Sore-Throat	…	Color	diseaseX
32	90	N	…	Pale	?

diseaseX
No

How do we represent an email?

- There are characteristics of emails that might give a computer a hint about whether it's spam
  - Possible *features*: words in body, subject line, sender, message header, time sent
- For this assignment, we choose to represent an email as the set $\{x_1, x_2, \dots, x_n\}$ of **distinct** words in the subject and body

How do we represent an email?

- There are characteristics of emails that might give a computer a hint about whether it's spam
  - Possible *features*: words in body, subject line, sender, message header, time sent
- For this assignment, we choose to represent an email as the set $\{x_1, x_2, \dots, x_n\}$ of **distinct** words in the subject and body

SUBJECT: Top Secret Business Venture

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…

{top, secret, business, venture, dear, sir, first, I, must, solicit, your, confidence, in, this, transaction, is, by, virture, of, its, nature, as, being, utterly, confidencial, and}

Notice that there are no duplicate words

words ind

1 0

Programming Project

Take the set $\{x_1, x_2, \ldots, x_n\}$ of distinct words to represent the email.

We are trying to compute

$$P(Spam|x_1, x_2, \ldots, x_n) = ???$$

Programming Project

Take the set $\{x_1, x_2, \ldots, x_n\}$ of distinct words to represent the email.

We are trying to compute

$$P(Spam|x_1, x_2, \ldots, x_n) = ???$$

Apply Bayes' Theorem. It's easier to find the probability of a word appearing in a spam email than the reverse.

$$P(Spam|x_1, x_2, \ldots, x_n) = \frac{P(x_1, x_2, \ldots x_n|Spam)P(Spam)}{P(x_1, x_2, \ldots x_n|Spam)P(Spam) + P(x_1, x_2, \ldots x_n|Ham)P(Ham)}$$

Apply the chain rule to the numerator:

$$P(x_1, x_2, \dots x_n | Spam)P(Spam) = P(x_1, x_2, \dots, x_n, Spam)$$

Apply the Chain Rule again to decompose this:

$$P(x_1, x_2, \dots, x_n, Spam)$$
$$= P(x_1 | x_2, \dots, x_n, Spam)P(x_2 | x_3, \dots, x_n, Spam) \dots P(x_n | Spam)P(Spam)$$

But this is still hard to compute.

How could you compute $\boxed{P(x_1 | x_2, \dots, x_n, Spam)}$?

We'll simplify the problem with an assumption (a big one!)

We will assume that the **words in the email are conditionally independent** of each other, given that we know whether or not the email is spam.

$$\Pr(\underbrace{\text{Nigeria prince} \mid \text{spam}}_{\text{viagra} \mid \text{spam}})$$

Definition: Two events A and B are *conditionally independent* given C if and only if

$$P(A \cap B|C) = P(A|C)P(B|C).$$

Equivalently, if P(B) > 0 and P(C) > 0, then

$$P(A|BC) = P(A|C).$$

Let's simplify the problem with an assumption.

We will assume that the **words in the email are conditionally independent** of each other, given that we know whether or not the email is spam.

This is why we call this *Naïve* Bayes: conditional independence isn't true.

So how does this help?

$$
\begin{aligned}
&P(x_1, x_2, \dots, x_n, Spam) \\
&= P(x_1|x_2, \dots, x_n, Spam)P(x_2|x_3, \dots, x_n, Spam) \dots P(x_n|Spam)P(Spam) \\
&\approx P(x_1|Spam)P(x_2|Spam) \dots P(x_n|Spam)P(Spam)
\end{aligned}
$$

$$P(x_1, x_2, \dots, x_n, Spam) \approx P(Spam) \prod_{i=1}^{n} P(x_i|Spam)$$

Using conditional independence

$$P(x_1, x_2, \ldots, x_n, Spam) \approx P(Spam)\prod_{i=1}^{n} P(x_i|Spam)$$

Similarly, $P(x_1, x_2, \ldots, x_n, Ham) \approx P(Ham)\prod_{i=1}^{n} P(x_i|Ham)$

Putting it all together

$$P(Spam|x_1, x_2, \ldots, x_n) \approx \boxed{\frac{P(Spam)\prod_{i=1}^{n} P(x_i|Spam)}{P(Spam)\prod_{i=1}^{n} P(x_i|Spam) + P(Ham)\prod_{i=1}^{n} P(x_i|Ham)}}$$

Given labelled training data, how do we compute these quantities?

$\boxed{P(Spam)}$ and $P(Ham)$?

$$\frac{\text{\# spam emails in training data}}{\text{\# emails in training data}}$$

What about $P(x_i|Spam)$, e.g., $P(viagra|Spam)$?

$$\frac{\text{\# spam emails w/ word } x_i}{\text{\# spam emails}}$$

$$P(x_1, x_2, \ldots, x_n, Spam) \approx P(Spam) \prod_{i=1}^{n} P(x_i|Spam)$$

Similarly, $P(x_1, x_2, \ldots, x_n, Ham) \approx P(Ham) \prod_{i=1}^{n} P(x_i|Ham)$

Putting it all together

$$P(Spam|x_1, x_2, \ldots, x_n) \approx \frac{P(Spam) \prod_{i=1}^{n} P(x_i|Spam)}{P(Spam) \prod_{i=1}^{n} P(x_i|Spam) + P(Ham) \prod_{i=1}^{n} P(x_i|Ham)}$$

$P(Spam)$ and $P(Ham)$ are just the fraction of training emails that are spam and ham

What about $P(x_i|Spam)$?

How spammy is a word?

What is $P(viagra|Spam)$ asking?

Would be easy to count how many spam emails contain this word:

$$P(w|Spam) = \frac{number\ of\ spam\ emails\ containing\ w}{total\ number\ of\ spam\ emails}$$

This seems reasonable, but there's a problem…

Suppose the word *Pokemon* only appears in ham in the training data, never in spam. Then we would estimate

$$P(Pokemon|Spam) = 0$$

Since the overall spam probability is the product of such individual probabilities, if any of those is 0, the whole product is 0

Any email with the word *Pokemon* would be assigned a spam probability of 0

What can we do?

SUBJECT: Get out of debt!

Cheap prescription pills! Earn fast cash using this one weird trick! Meet singles near you and get preapproved for a low interest credit card! Pokemon

→

definitely not spam, right?

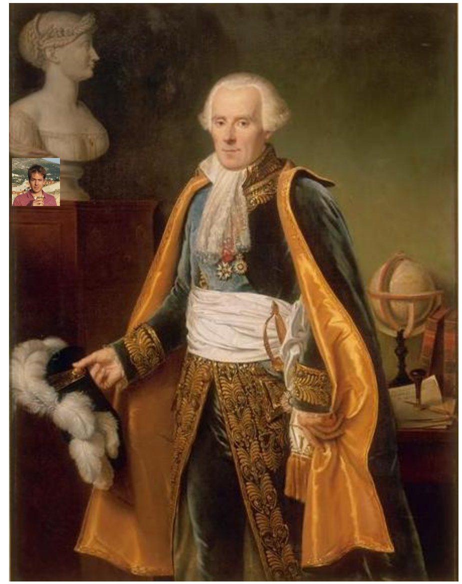
Laplace smoothing

w

- Crazy idea: what if we pretend we've seen every outcome once already?
- Pretend we've seen one more spam email *with* w, one more *without* w

$$\boxed{P(w|Spam)} = \frac{|spam\ emails\ containing\ w| + 1}{|spam\ emails| + 2}$$

- Then, $P(Pokemon|Spam) > 0$
- No one word will bias the overall probability too much
- General technique to avoid assuming that unseen events will never happen



Naïve Bayes Overview

For each word w in the spam training set, count how many spam emails contain w:

$$P(w|Spam) = \frac{|spam\ emails\ containing\ w| + 1}{|spam\ emails| + 2}$$

Compute $P(w|Ham)$ analogously

$$P(Spam) = \frac{|spam\ emails|}{|spam\ emails|+|ham\ emails|}\ ,\ P(Ham) = 1 - P(Spam)$$

For each test email with words $\{x_1, x_2, \ldots, x_n\}$,

$$P(Spam|x_1, x_2, \ldots, x_n) \approx \frac{P(Spam)\prod_{i=1}^{n} P(x_i|Spam)}{P(Spam)\prod_{i=1}^{n} P(x_i|Spam) + P(Ham)\prod_{i=1}^{n} P(x_i|Ham)}$$

Output “spam” iff $P(Spam|x_1, x_2, \ldots, x_n)$ > 1/2

Read the Notes!

Read Jonathan Lee's **Naïve Bayes Notes** on the course web for precise technical details, start early, and ask for help if you get stuck!

Describes how to avoid floating point underflow in formulas such as $\prod_{i=1}^{n} P(x_i|Spam)$