

Naïve Bayes Classifiers



Jonathan Lee and Varun Mahadevan

Programming Project: Spam Filter

- Implement a Naive Bayes classifier for classifying emails as either spam or ham (= nonspam).
- Read Jonathan Lee's notes on the course web, read the specification of your program, start early, and ask for help if you get stuck!

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 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.

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 “understands” what makes an email spam or ham
 - We’re using a Naïve Bayes classifier, but there are other approaches
 - This is a *Machine Learning* problem (take CSE 446 for more)
- 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!



WIKIPEDIA
The Free Encyclopedia



Naïve Bayes in theory

We will use what we've learned in the past few weeks. 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

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

- Conditional Independence of A and B , given C

$$\begin{aligned} P(A \cap B|C) &= P(A|C)P(B|C) \\ P(A|B \cap C) &= P(A|C) \end{aligned}$$

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 virtue of its nature as being utterly confidential and top secret...



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

Notice that there are no duplicate words

- Take the set $\{x_1, x_2, \dots, x_n\}$ of distinct words to represent the email.
- We are trying to compute

$$P(\text{Spam} | x_1, x_2, \dots, x_n) = ???$$

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

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

- 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:

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

But this is still hard to compute.

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

- 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?

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

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

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

- Putting it all together

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

- $P(Spam)$ 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 $P(\textit{viagra}|\textit{Spam})$ asking?
- Would be easy to count how many spam emails contain this word:

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

- This seems reasonable, but there's a problem...

- Suppose the word *Pokemon* only appears in ham in the training data, never in spam

$$P(\textit{Pokemon}|\textit{Spam}) = 0$$

- 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

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?

- What can we do?

Laplace smoothing

- 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

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

- Then, $P(\text{Pokemon}|Spam) > 0$
- 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 entire training set, count how many spam emails contain w :

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

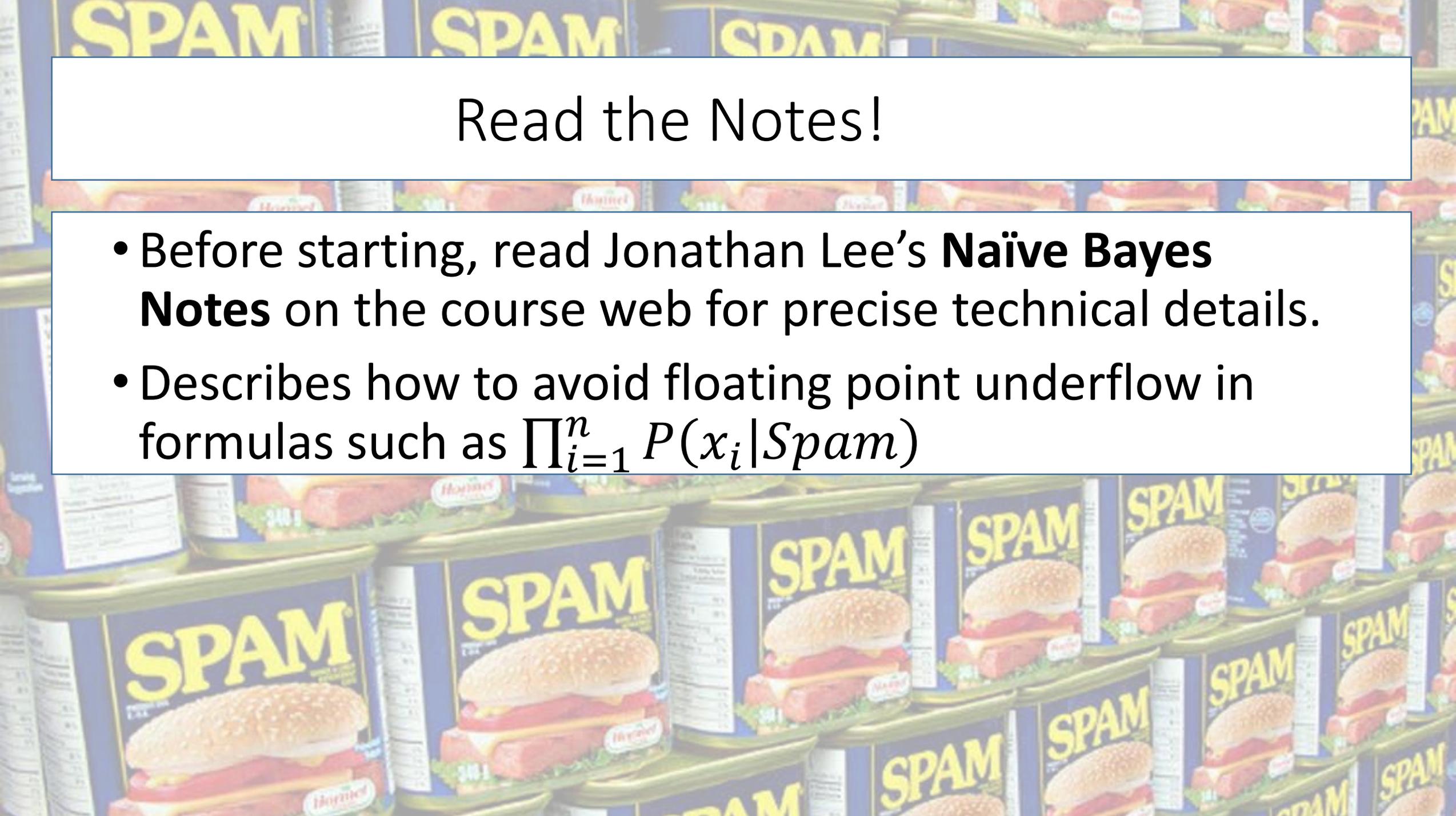
- Compute $P(w|Ham)$ analogously

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

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

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

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

The background of the slide is a repeating pattern of SPAM cans and sandwiches. The cans are blue with the word "SPAM" in large yellow letters. Next to each can is a sandwich made with a bun, lettuce, tomato, cheese, and a slice of SPAM. The cans and sandwiches are arranged in a grid-like pattern, creating a dense, textured background.

Read the Notes!

- Before starting, read Jonathan Lee's **Naïve Bayes Notes** on the course web for precise technical details.
- Describes how to avoid floating point underflow in formulas such as $\prod_{i=1}^n P(x_i | Spam)$