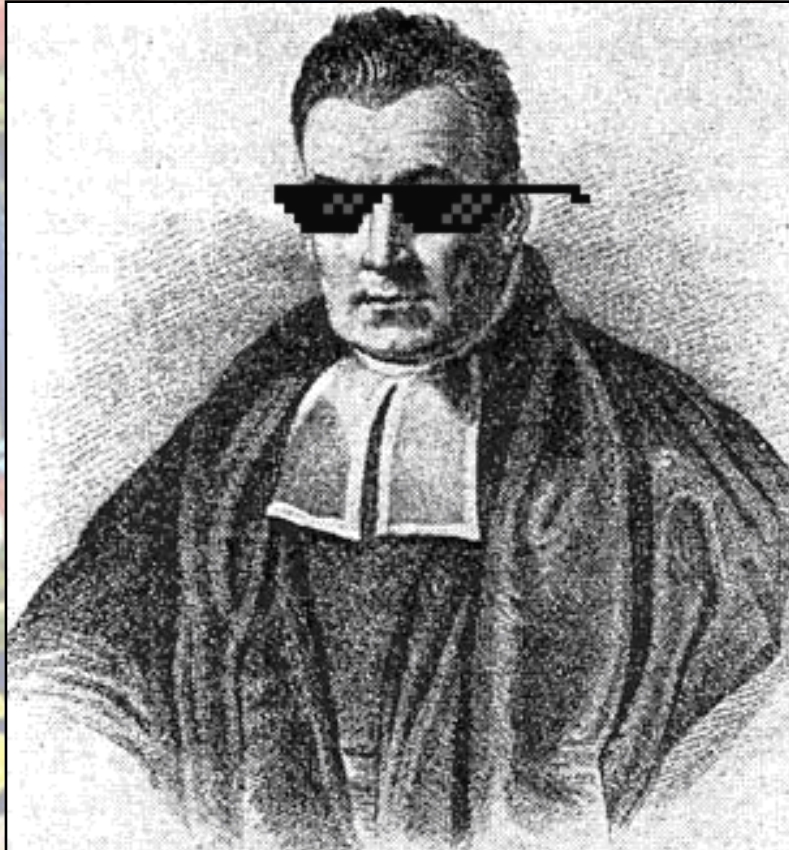# Naïve Bayes Classifiers

Jonathan Lee and Varun Mahadevan

Independence

Recap:

Definition: Two events X and Y are independent if

$$P(XY) = P(X)P(Y),$$

and if $P(Y) > 0$, then $P(X|Y) = P(X)$

Conditional Independence

Conditional Independence tells us that:

Two events A and B are conditionally independent given C if

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

and if P(B) > 0 and P(C) > 0, then $P(A|BC) = P(A|C)$

Example:

Randomly choose a day of the week

A = { It is not a Monday }

B = { It is a Saturday }

C = { It is the weekend }

A and B are dependent events

P(A) = 6/7, P(B) = 1/7, P(AB) = 1/7

Now condition both A and B on C:

P(A|C) = 1, P(B|C) = ½, P(AB|C) = ½

P(AB|C) = P(A|C)P(B|C) => A|C and B|C are independent

Programming Project: Spam Filter

Due: Check the Calendar

- Implement a Naive Bayes classifier for classifying emails as either spam or ham.

- You may use C, Java, Python, or R; **ask if you have a different preference.** We've provided starter code in Java, Python and R.

- Read Jonathan's notes on the website, start early, and ask for help if you get stuck!

Spam vs. Ham

- (at least in the past) the bane of any email user's existence
- You know it when you see it!
- Easy for humans to identify, but not necessarily easy for computers
- Less of a problem for consumers now, because spam filters have gotten really good...

❌ 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 labelled *spam* or *ham*
  - Someone has to label these by hand!
  - Usually 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 446 for more!)
- Test your model on emails whose label isn't known to the model, and see how well it does
  - Usually called the *test data*

Naïve Bayes in the real world

- One of the oldest, simplest models for *classification*

- Still, very powerful and used *all the time* 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!

How do we represent an email?

- There's a lot of different things about emails that might give a computer a hint about whether or not it's spam
  - Possible *features*: words in body, subject line, sender, message header, time sent…

- For this assignment, we choose to represent an email just as the set of **distinct** words $(x_1, x_2, …, x_n)$ 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!

Naïve Bayes in theory

The concepts behind Naïve Bayes are nothing new to you -- we'll be using what we've learned in the past few weeks.

Specifically

- Bayes Theorem

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

- Law of Total Probability

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

- Conditional Probability

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

- Chain Rule

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

- Conditional Independence

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

- Take the set of distinct words $(x_1, x_2, \ldots, x_n)$ to represent the text in an email.

- We are trying to compute
$$P(Spam|x_1, x_2, \ldots, x_n) = ???$$

- By applying Bayes Theorem, we can reverse the conditioning. It's easier to find the probability of a word appearing in a spam email than the reverse.

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

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

- Let's take a look at the numerator and apply the rule for Conditional Probability

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

- And now let's use the Chain Rule to decompose this

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

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

$$= \cdots$$

$$= P(x_1 | x_2, \ldots, x_n, Spam) \ldots P(x_{n-1} | x_n, Spam)P(x_n | Spam)P(Spam)$$

But this is still hard to compute.

- 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. **This isn't true irl!**

- So how does this help?

$$P(x_1, x_2, \ldots, x_n, Spam)$$
$$= P(x_1|x_2, \ldots, x_n, Spam) \ldots P(x_{n-1}|x_n, Spam) P(x_n|Spam) P(Spam)$$
$$\approx P(x_1|Spam) P(x_2|Spam) \ldots P(x_{n-1}|Spam) P(x_n|Spam) P(Spam)$$

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

- So we know that

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

- 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

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

How spammy is a word?

- Have a nice formula for email spam probability, using conditional probabilities of words given ham/spam

- $P(Spam)$ and $P(Ham)$ are just the proportion of total emails that are spam and ham

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

- Would be easy to just count up how many spam emails have this word in them, so

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

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

- Suppose the word *Pokemon* only ever showed up in ham emails in the training data, never in spam

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

- Since the overall spam probability is the product of a bunch of individual probabilities, if any of those is 0, the whole thing 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*

$$P(w|Spam) = \frac{[number\ of\ spam\ emails\ containing\ w] + 1}{[total\ number\ of\ spam\ emails] + 2}$$

- Then, $P(Pokemon|Spam) > 0$

- No one word can "poison" the overall probability too much

- General technique to avoid assuming that unseen events will never happen