More about Naïve Bayes

Instructor: Wei Xu

Some slides adapted from Dan Jurfasky and Brendan O’connor

Market Research

(due 11:59pm, Friday, Feb 3rd)

- When was the company started?
- Who were the founders?
- What kind of organization is it? (publicly traded company, privately held company, non-profit organization, other)
- What is company's main business model?
- How does the company generate revenue?
- What is the finance situation of the company? (stock price, anual report, news)

Market Research

(due 11:59pm, Friday, Feb 3rd)

- Why is the company interested in speech or NLP technologies or both?
- What are specific areas or applications of speech/ NLP the company is interested in?
- What products of the company use speech or NLP technologies?
- What the main users of their speech or NLP technologies?
- Does the company hold any patent using speech or NLP technologies?
- Does the company publish any papers on speech or NLP technologies?

Market Research

(due 11:59pm, Friday, Feb 3rd)

- Is the company recently hiring in NLP? interns? PhD?
- What specific expertise within speech or NLP the company is looking to hire?
- How is the press coverage of the company?
- How many employees do the company have?
- An estimation of how many speech/NLP experts currently in the company?
- Any notable speech/NLP researcher or recent hires in the company?
- Which city is the company's speech/NLP research office located?

Text Classification

?

Test document

parser
language
label
translation
…

Machine Learning	NLP	Garbage Collection	Planning	GUI
learning	parser	garbage	planning	...
training	tag	collection	temporal	
algorithm	training	memory	reasoning	
shrinkage	translation	optimization	plan	
network...	language...	region...	language...	

Classification Methods: Supervised Machine Learning

- *Input:*
  - a document d
  - a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$
  - A training set of m hand-labeled documents $(d_1,c_1),\ldots,(d_m,c_m)$
- *Output:*
  - a learned classifier $y{:}d \rightarrow c$

Naïve Bayes Classifier

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

MAP is "maximum a posteriori"
= most likely class

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

Bayes Rule

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Dropping the denominator

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c)P(c)$$

Document *d* represented as features $x_1, \ldots, x_n$

Multinomial Naïve Bayes: Assumptions

$$P(x_1, x_2, \ldots, x_n \mid c)$$

- **Bag of Words assumption**: Assume position doesn't matter
- **Conditional Independence**: Assume the feature probabilities $P(x_i \mid c_j)$ are independent given the class c.

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

Multinomial Naïve Bayes: Assumptions

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c) P(c)$$

$$= \underset{c \in C}{\operatorname{argmax}} \; \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

estimate from data

Multinomial Naïve Bayes: Learning

- maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

use unique word w_i as features (in place of x_i)

Multinomial Naïve Bayes: Learning

- Calculate $P(c_j)$ terms
  - For each c_j in C do

$$docs_j \leftarrow \text{all docs with class} = c_j$$

$$P(c_j) \leftarrow \frac{|docs_j|}{|\text{total \# documents}|}$$

Multinomial Naïve Bayes: Learning

- maximum likelihood estimates
  - simply use the frequencies in the data

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\boxed{\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}}$$

use unique word w_i as features (in place of x_i)

Multinomial Naïve Bayes: Learning

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

fraction of times word w_i appears among all words in documents of topic c_j

- Create mega-document for topic *j* by concatenating all docs in this topic
  - Use frequency of *w* in mega-document

Zero Probabilities Problem

- What if we have seen no training documents with the word ***fantastic*** and classified in the topic **positive**?

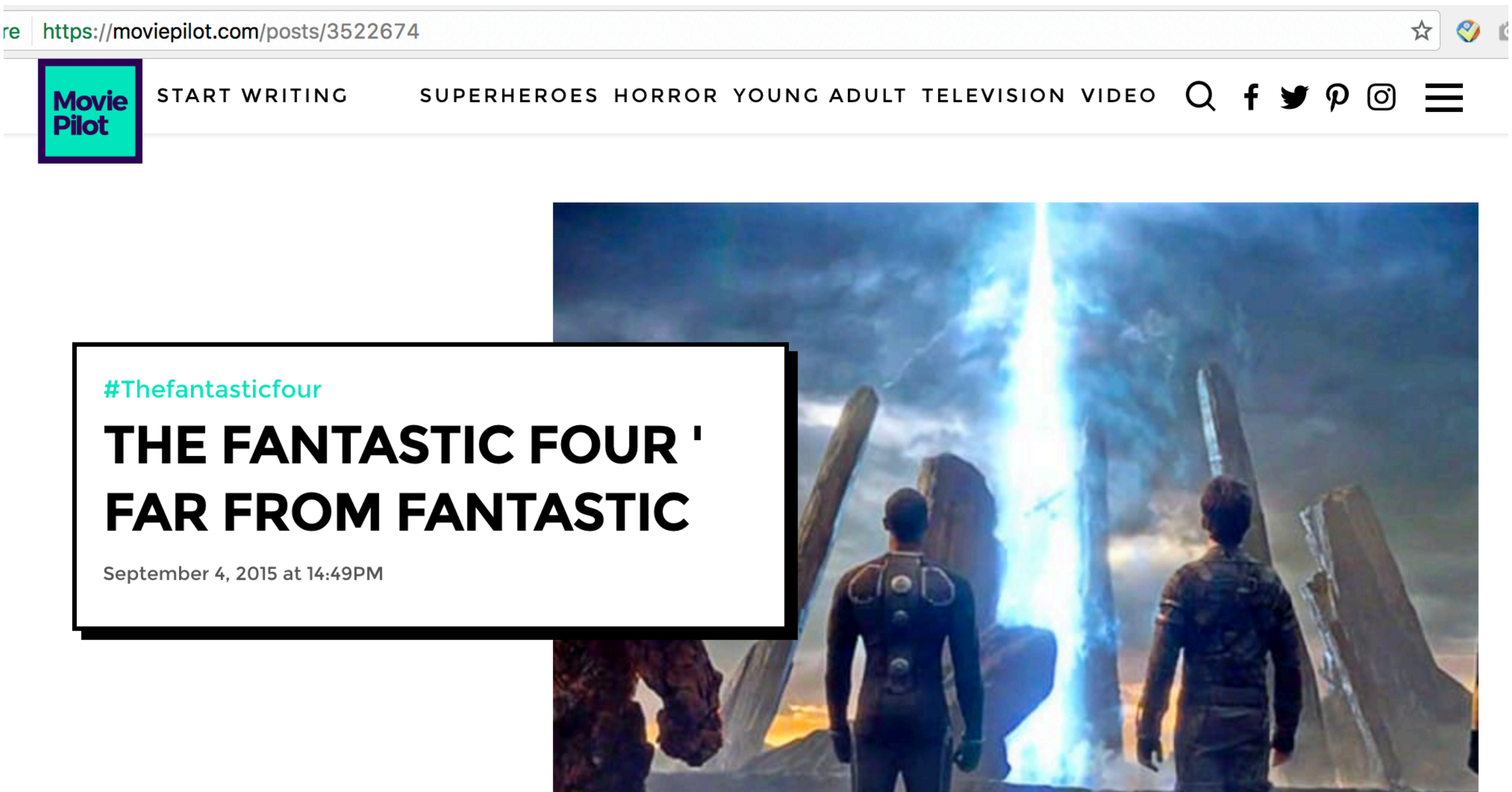
$$\hat{P}(\text{"fantastic"} \mid \text{positive}) = \frac{count(\text{"fantastic"}, \text{positive})}{\sum_{w \in V} count(w, \text{positive})} = 0$$

- Zero probabilities cannot be conditioned away, no matter the other evidence!

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} \ \hat{P}(c) \prod_i \hat{P}(x_i \mid c)$$

Problem with Maximum Likelihood

- What if we have seen no training documents with the word ***fantastic*** and classified in the topic **positive**?



Laplace (add-1) smoothing for Naïve Bayes

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c)}{\sum_{w \in V} \left(count(w, c) \right)}$$

$$= \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c) \right) + |V|}$$

smoothing to avoid zero probabilities

- For *unknown* words (which completely doesn't occur in training set), we can ignore them.

Multinomial Naïve Bayes: Learning

- From training corpus, extract *Vocabulary*
- Calculate $P(w_k \mid c_j)$ terms
  - $Text_j \leftarrow$ single doc containing all $docs_j$
  - For each word w_k in *Vocabulary*

    $n_k \leftarrow$ # of occurrences of w_k in $Text_j$

$$P(w_k \mid c_j) \leftarrow \frac{n_k + \alpha}{n + \alpha \, |Vocabulary|}$$

smoothing to avoid zero probabilities (often use α = 1)

Exercise

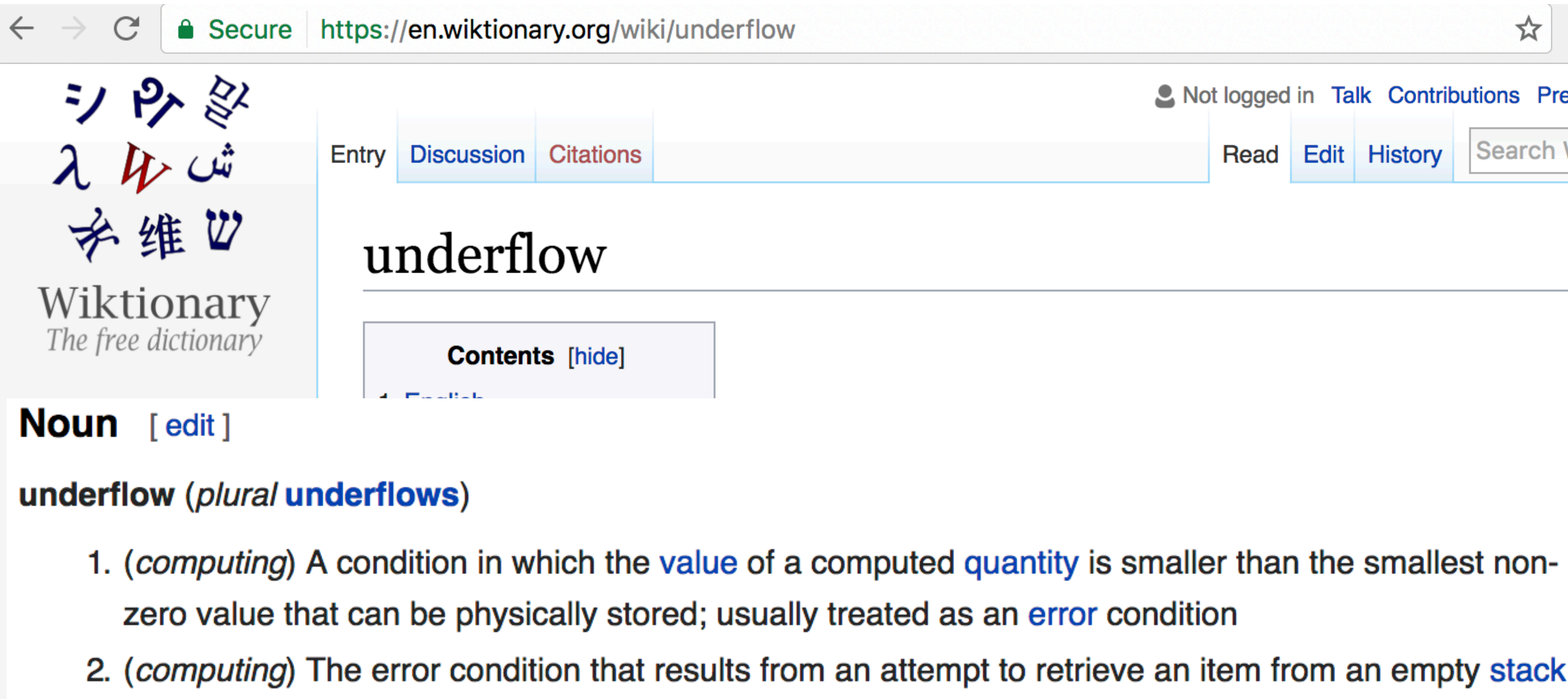
Naïve Bayes: Practical Issues

$$
\begin{aligned}
c_{MAP} &= \text{argmax}_c P(c|x_1, \ldots, x_n) \\
&= \text{argmax}_c P(x_1, \ldots, x_n|c)P(c) \\
&= \text{argmax}_c P(c) \prod_{i=1}^{n} P(x_i|c)
\end{aligned}
$$

- Multiplying together lots of probabilities
- Probabilities are numbers between 0 and 1

Q: What could go wrong here?

Underflow


Secure
https://en.wiktionary.org/wiki/underflow
Not logged in Talk Contributions
Entry Discussion Citations
Read Edit History
Wiktionary
The free dictionary
underflow
Contents [hide]
Noun [edit]
underflow (plural underflows)
1. (computing) A condition in which the value of a computed quantity is smaller than the smallest non-zero value that can be physically stored; usually treated as an error condition
2. (computing) The error condition that results from an attempt to retrieve an item from an empty stack

Floating Point Numbers

Exponent E significand F (also called ***mantissa***)

+/-	x x x x	y y y y y y y y y y y y

Sign bit

In **decimal** it means (+/-) **1**. yyyyyyyyyyyy × 10^{xxxx}

In **binary**, it means (+/-) **1**. yyyyyyyyyyyy × 2^{xxxx}

(The 1 is implied)

Floating Point Numbers

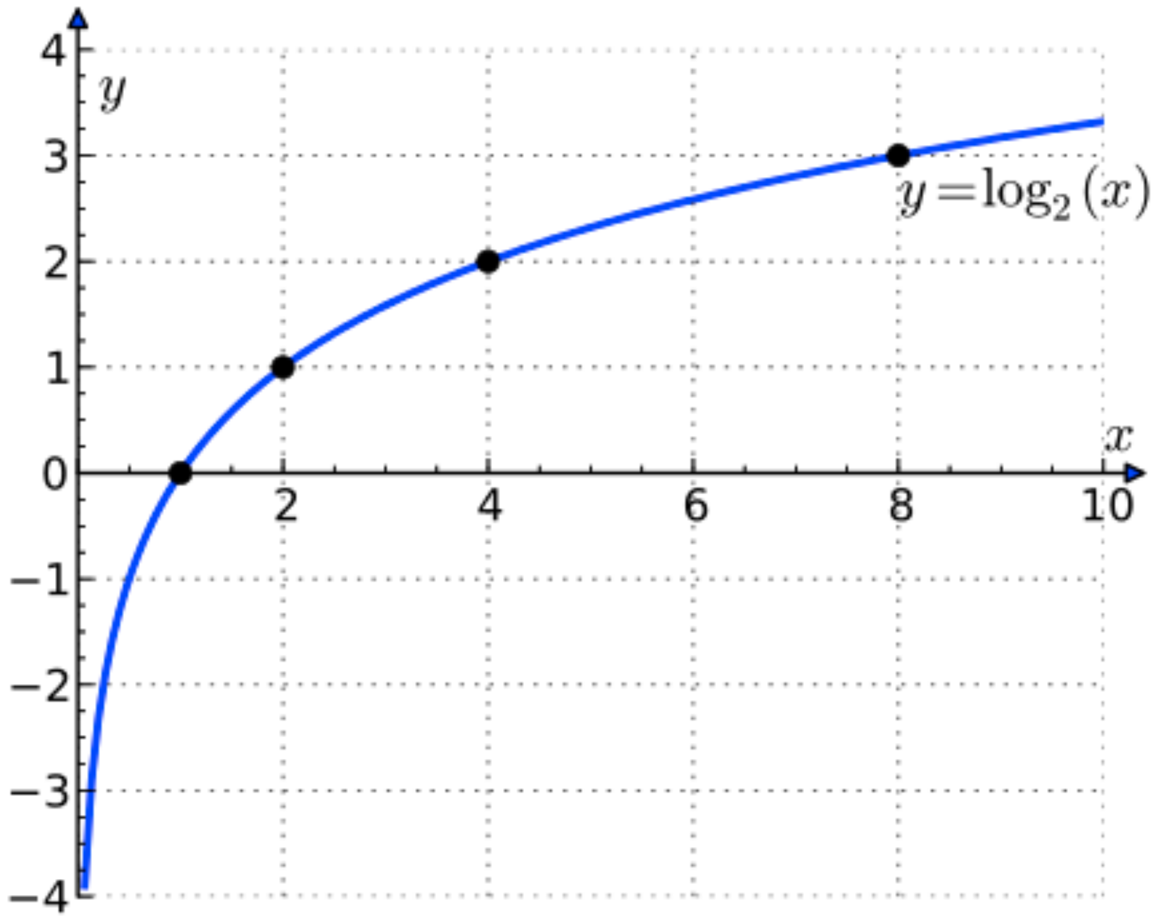
IEEE 754 double precision (64 bits)

S	exponent	significand
1	11 bits	52 bits

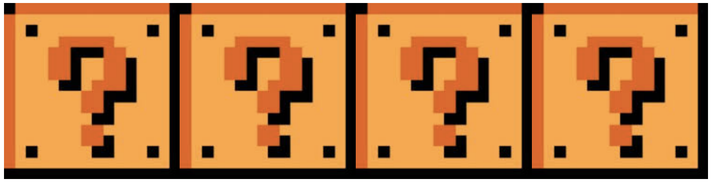
Largest = $1.111\ldots \times 2^{+1023}$

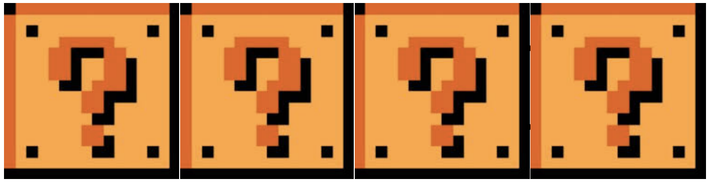
Smallest = $1.000\ldots \times 2^{-1024}$

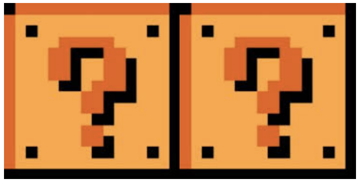
Working with Probabilities in Log Space



Log Identities (review)

$$\log(a \times b) =$$ 

$$\log(\frac{a}{b}) =$$ 

$$\log(a^n) =$$ 

Naïve Bayes with Log Probabilities

$$
\begin{aligned}
c_{MAP} &= \operatorname{argmax}_c P(c|x_1, \ldots, x_n) \\
&= \operatorname{argmax}_c P(c) \prod_{i=1}^{n} P(x_i|c) \\
&= \operatorname{argmax}_c \log \left(P(c) \prod_{i=1}^{n} P(x_i|c) \right) \\
&= \operatorname{argmax}_c \log P(c) + \sum_{i=1}^{n} \log P(x_i|c)
\end{aligned}
$$

because log is monotonic increasing

Naïve Bayes with Log Probabilities

$$c_{MAP} = \operatorname{argmax}_c \log P(c) + \sum_{i=1}^{n} \log P(x_i|c)$$

Q: Why don't we have to worry about floating point underflow anymore?

Working with Probabilities in Log Space

x	log(x)
0.00000000001	-11
0.00001	-5
0.0001	-4
0.001	-3
0.01	-2
0.1	-1

What if we want to calculate posterior log-probabilities?

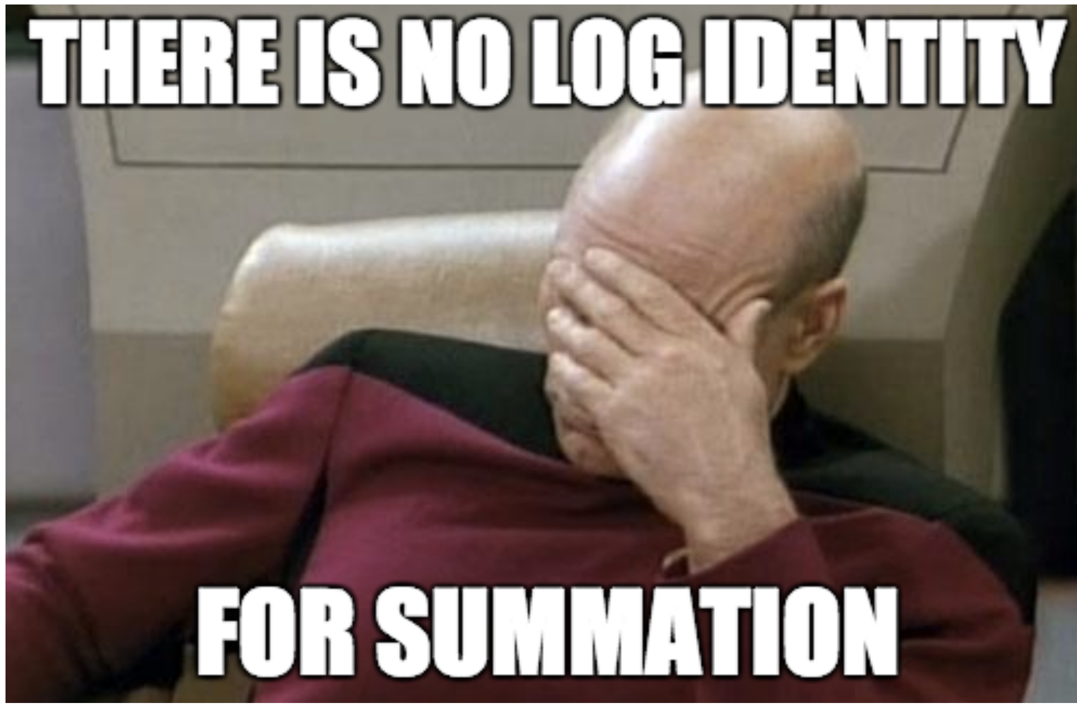
$$P(c|x_1,\ldots,x_n) = \frac{P(c)\prod_{i=1}^n P(x_i|c)}{\sum_{c'} P(c')\prod_{i=1}^n P(x_i|c')}$$

$$\log P(c|x_1,\ldots,x_n) = \log \frac{P(c)\prod_{i=1}^n P(x_i|c)}{\sum_{c'} P(c')\prod_{i=1}^n P(x_i|c')}$$

$$= \log P(c) + \sum_{i=1}^n \log P(x_i|c) - \boxed{\log\left[\sum_{c'} P(c')\prod_{i=1}^n P(x_i|c')\right]}$$

What if we want to calculate posterior log-probabilities?

$P(c|x_1, .$



$\frac{x_i|c)}{P(x_i|c')}$

$\log P(c|x_1$

$\frac{P(x_i|c)}{{}_1\, P(x_i|c')}$

$$= \log P(c) + \sum_{i=1}^{n} \log P(x_i|c) - \boxed{\log \left[\sum_{c'} P(c') \prod_{i=1}^{n} P(x_i|c') \right]}$$

Log Exp Sum Trick

- We have: a bunch of log probabilities: log(p1), log(p2), log(p3), … log(pn)
- We want: log(p1 + p2 + p3 + … pn)
- We could convert back from log space, sum then take the log.

Q: Is this a good idea?

Log Exp Sum Trick

- We have: a bunch of log probabilities:
  log(p1), log(p2), log(p3), … log(pn)

- We want: log(p1 + p2 + p3 + … pn)

- We could convert back from log space, sum then take the log.

If the probabilities are very small, this will result in floating point underflow

Log Exp Sum Trick

$$\log[\sum_i \exp(x_i)] = x_{max} + \log[\sum_i \exp(x_i - x_{max})]$$

Another issue: Smoothing

$$\hat{P}(w_i|c) = \frac{\text{count}(w, c) + 1}{\sum_{w' \in V} \text{count(w',c)} + |V|}$$

Another issue: Smoothing

Can think of alpha as a “pseudo-count”.
Imaginary number of times this word has been seen.

$$\hat{P}(w_i|c) = \frac{\text{count}(w,c) + \alpha}{\sum_{w' \in V} \text{count}(w',c) + \alpha|V|}$$

Another issue: Smoothing

Alpha doesn't necessarily need to be 1 (hyperparameter)

$$\hat{P}(w_i|c) = \frac{\text{count}(w,c) + \alpha}{\sum_{w' \in V} \text{count}(w',c) + \alpha|V|}$$

Hyperparameters are parameters that cannot be directly learned from the standard training process, and need to be predefined.

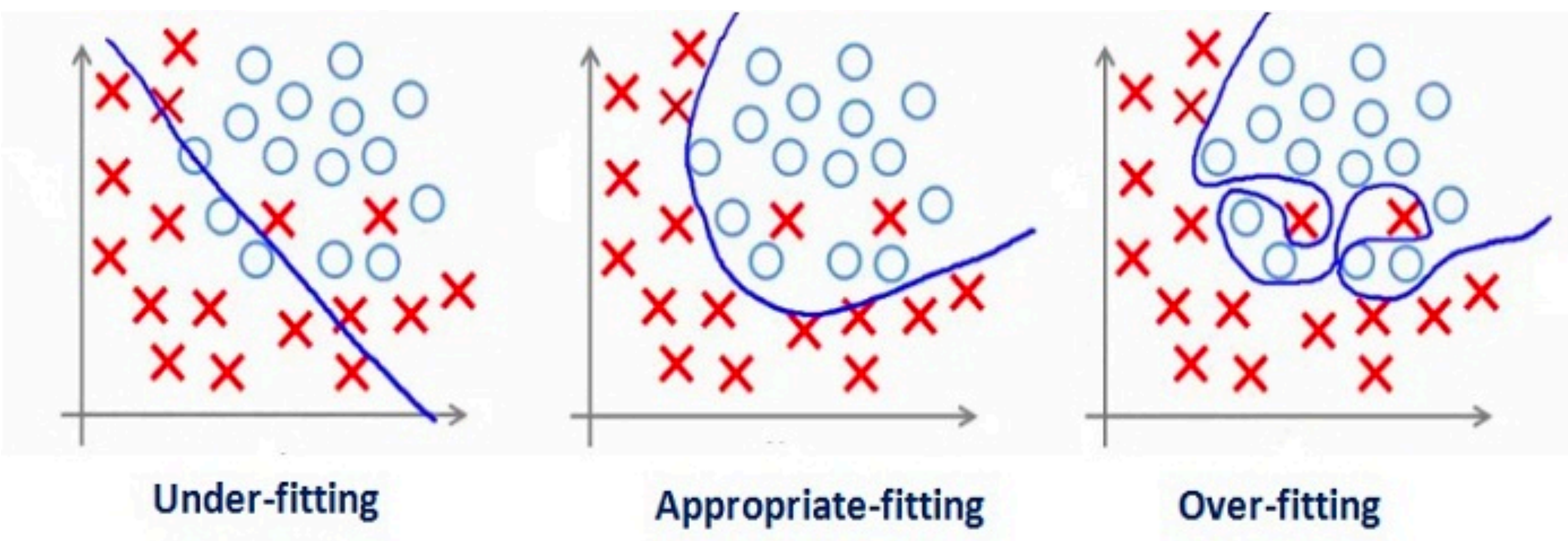
Another issue: Smoothing

$$\hat{P}(w_i|c) = \frac{\text{count}(w,c) + \alpha}{\sum_{w' \in V} \text{count}(w',c) + \alpha|V|}$$

- What if alpha = 0?
- What if alpha = 0.000001?
- What happens as alpha gets very large?

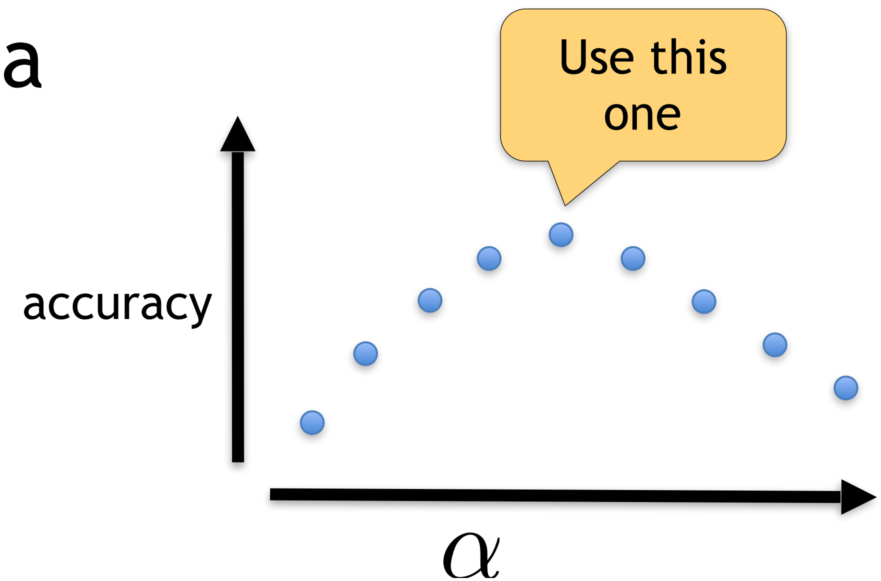
Overfitting

- Model parameters fits the training data well, but generalize poorly to test data
- How to check for overfitting?
  – Training vs. test accuracy
- Pseudo-count parameter combats overfitting



How to pick Alpha?

- Split train vs. test (dev)
- Try a bunch of different values
- Pick the value of alpha that performs best
- What values to try? Grid search
  - (10^-2,10^-1,…,10^2)



Data Splitting

- Train vs. Test

- Better:
  - Train (used for fitting model **parameters**)
  - Dev (used for tuning **hyperparameters**)
  - Test (reserve for final evaluation)

- Cross-validation

Feature Engineering

- What is your word / feature representation
  - Tokenization rules: splitting on whitespace?
  - Uppercase is the same as lowercase?
  - Numbers?
  - Punctuation?
  - Stemming?