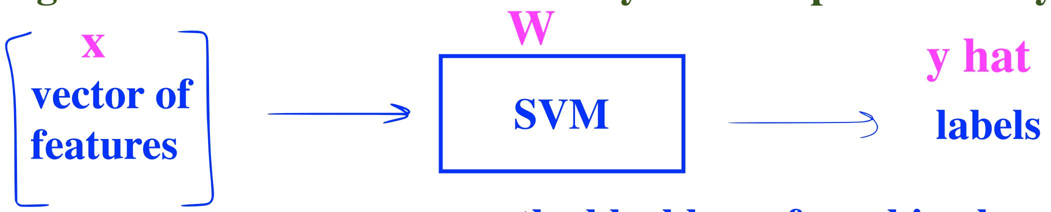
e.g. predict the author of a document
grammatical structures that they use or topics that they discuss



y hat = Wx

feature engineering
HOW TO GENERATE?

the blackbox of machine learning
we cloud replace this with a random forest or whatever you want, and none of those sort of explicitly come with intuition about language.
they’re just going to do their best to learn a function that maps this to that

Word/document features, Semantic vectors

Applications

Machine learning approaches to:

- document classification
- token classification
- sequence-to-sequence
- ...

Word → vector

- one-hot encoding
- semantic

One-hot encoding

1. Create a vector of zeros with length $|V|$
2. When representing word i, set $w_i = 1$

$$[0, 0, ..., 0, 1, 0, ..., 0, 0]$$

we want them to be informative and short

Vector semantics

- "semantic vector"
- "feature vector"
- "embedding"

The distance between any two points (words) in the "semantic space" reflect relationships between those words.

Desiderata

Vectors should be "close" if words are:

- (near-)synonymous/*similar*
    - e.g. "coffee" and "tea"
- *related*?
    - e.g. "coffee" and "drink"

Also consider:

- connotation/sentiment

How?

distributionalist intuition: defining a word by counting what other words occur in its environment

define a word as sort of the sum of all of the contexts in which it appears

the string itself doesn’t matter (we can change the spelling from “ss” to “sss” and the new string acquires the meaning of the old words) , the way that it’s used is what matters (just like the way we learn english — not by someone pointing to it in a dictionary or telling you. Learnt by hearing it and seeing it over and over in context and inferring the meaning

Term-document matrix

maybe some correlation
between ‘heart’ and ‘failure’

each column a word, each row a document

	heart	fire	she	TOTAL
Taylor Swift	101	29	238	52690
Billy Joel	41	27	323	30734
The Beatles	89	6	532	47707

assertion about documents: the distribution of terms
that Beatles use maybe is more similar to Taylor Swift
than the BillyJoe

Term-term matrix

count number of documents in which words co-occur

OR

count number of occurrences within *X* words (occurrences of word A *in the context of* word B)

synonyms: they would have the same or similar relationship to another words or a least the same proportion, one of which is just more common than the other

	lonely	blue	sky
lonely	58	3	0
blue	3	69	4
sky	0	4	16

Distance metrics

- Euclidean

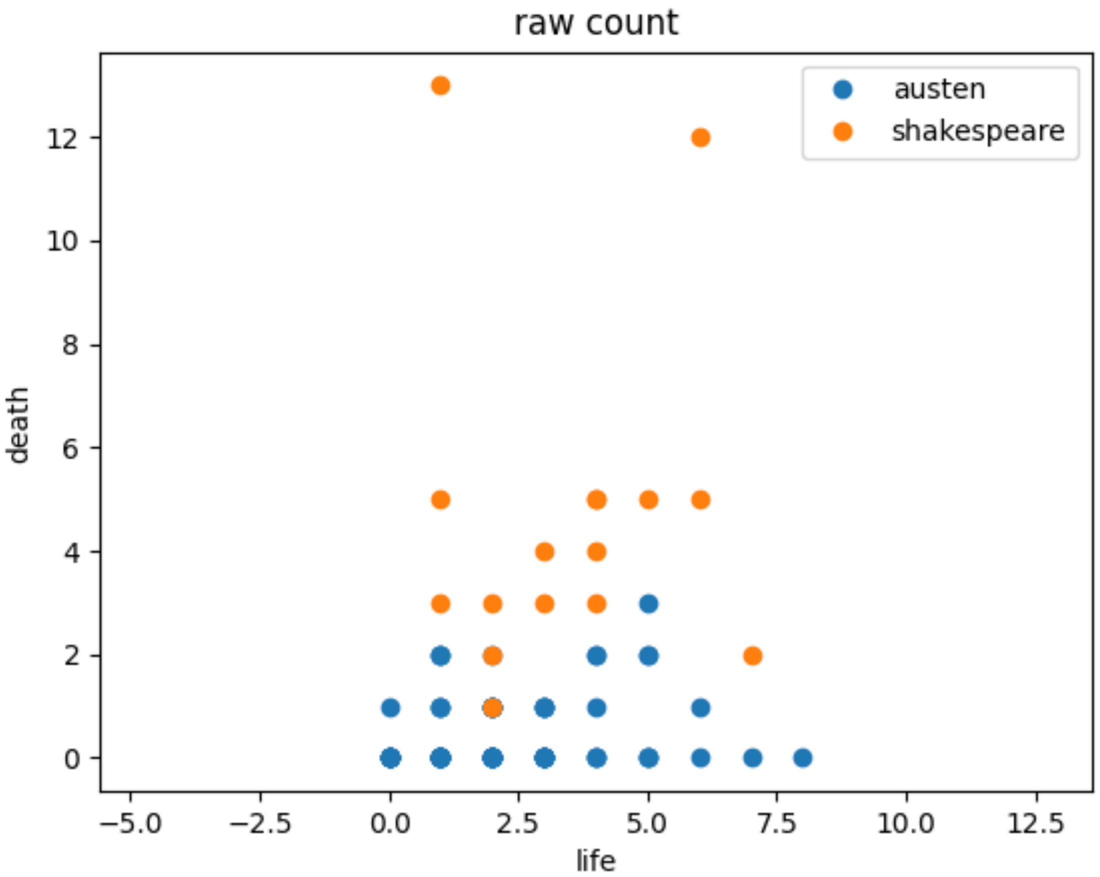
fall down: if two words are not equal popularity like ‘walk’ ‘trod ‘walk is used quite a bit more

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$$

- cosine "similarity" - *angle* between vectors
  - normalizes document length

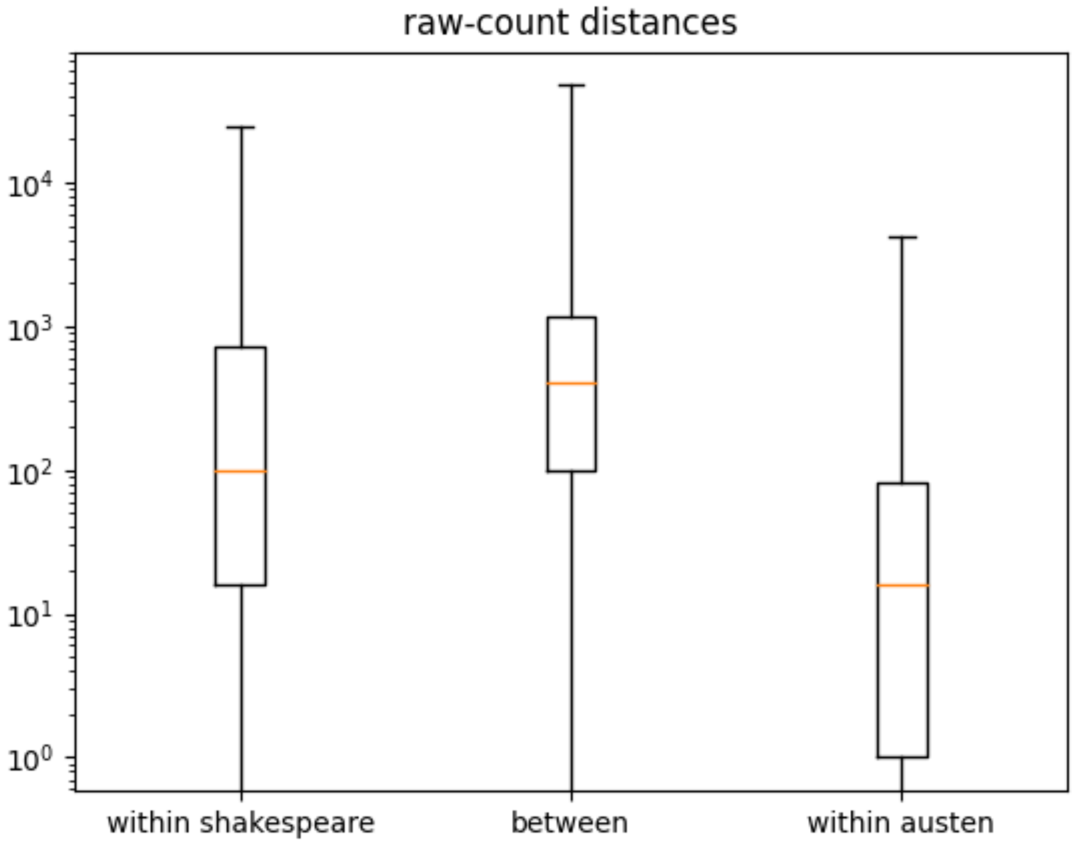
$$c(\mathbf{x}, \mathbf{y}) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

Experiment: raw counts



while they both talk about life the same amount,
Shakespeare talks about death a lot more

Experiment: raw-counts distances



distance on average are greater between classes than within classes

Weightings

- TF-IDF
- (P)PMI

TF-IDF

term frequency-inverse document frequency

- w - # word i in document
- n - # total words in document
- N - # documents total
- d - # documents with word i
- $\text{tf} = \log_{10}\left(\frac{w}{n} + 1\right)$
- $\text{idf} = \log_{10}\left(\frac{N}{d}\right)$
- $\text{tf-idf} = tf \times idf$

the document frequency for ‘the’ is approximately 1(nearly all document contains ‘the)

— the document frequency says something about globally how common is this word

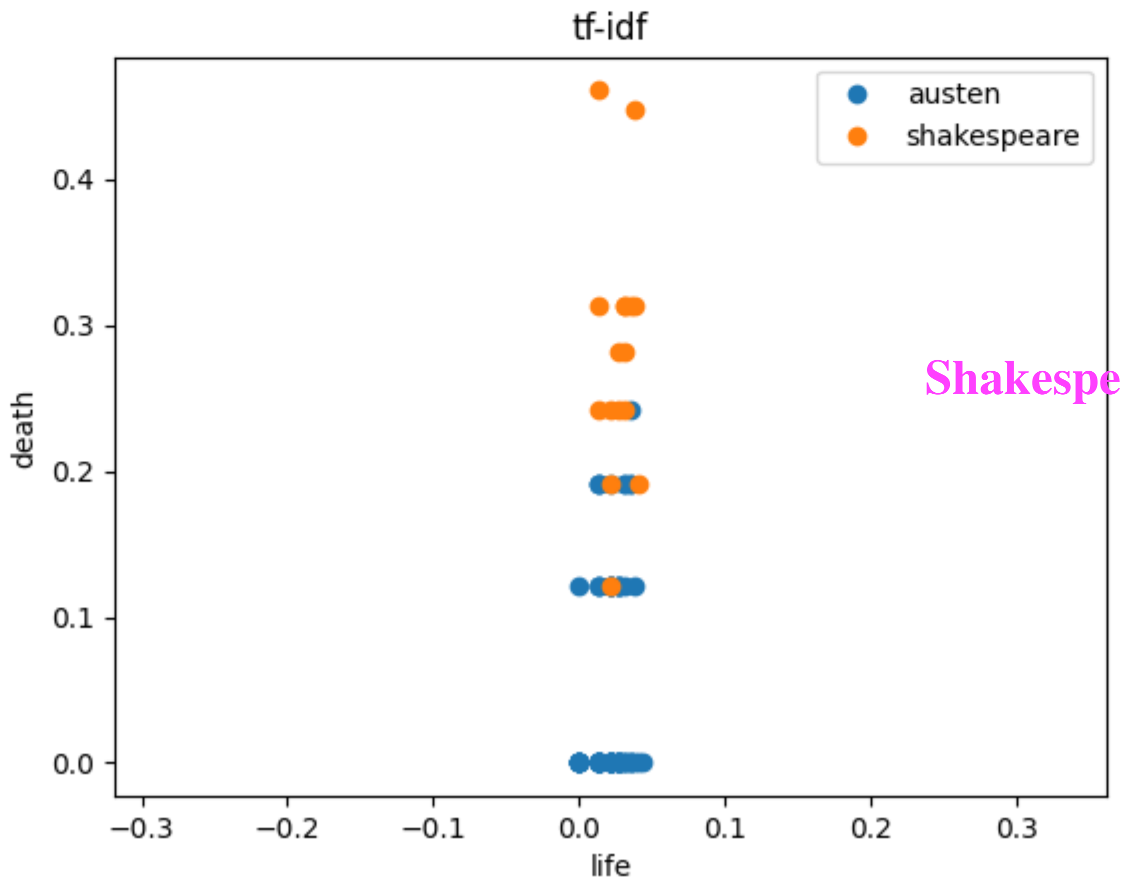
down weighing anything that is globally common

low for common word (because idf is low)

the inverse document frequency is going to be low when the term frequency is high

Experiment: tf-idf

apply this idea of weighting, which basically gives more weight to rarer or more interesting words, in this case, death is the more interesting word and it gets more weighting and essentially that dimension that stretched out relative to the other



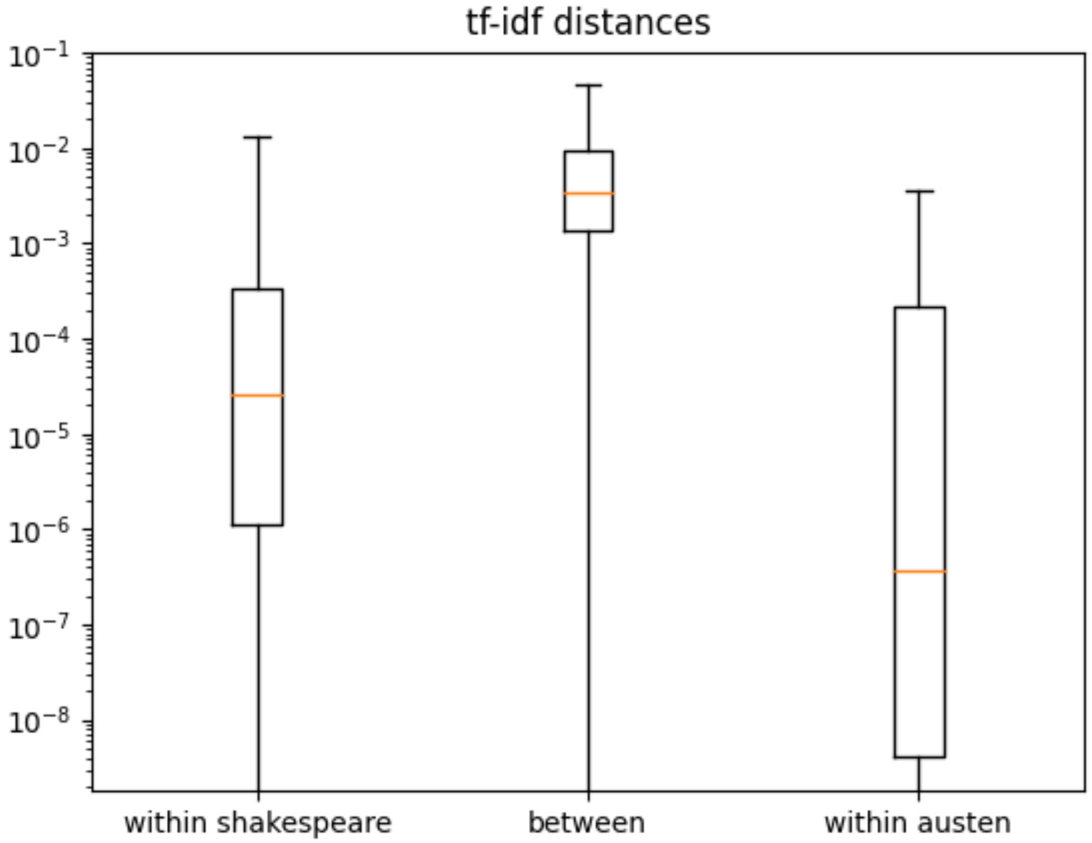
Shakespeare used death more than average, so tf-idf is higher

life is more common so x-axis get more compressed than y-axis

Experiment: tf-idf distances

it makes this sort of between class variants greater

which down the line should make the document classification problem easier



Pointwise mutual information (PMI)

How often do these co-occur, compared to how often they would co-occur by chance?

$p(w, c)$ - probability of word w and context c **the probability of this word occurring in this context**

"context" - document OR neighborhood of (another) word **p(w) is high — common word**

$$\mathrm{PMI} = \log_2\left(\frac{p(w,c)}{p(w)p(c)}\right)$$

What does it mean for this to be negative?

$$\mathrm{PPMI} = \max\left(\log_2\left(\frac{p(w,c)}{p(w)p(c)}\right), 0\right)$$

when PMI is high, we would say w is relevant to c

如果先假设c为另一个word 那么当c和w都是常见词如‘a’ ‘the’的时候 p(w,c)大但p(w)和p(c)也大

如果是‘dinosaur’ ‘fossil’ 那么p(w,c)大，但p(w)和p(c)不大 —they co-occur more often than we would expect by chance

(by chance mean when using unigram model to select word, it’s unlikely to choose them to near each other by chance

PMI log里的 = 1: w is a rare word, c is a common word
PMI log里的 <1 : those two words occur together less likely than we would expect by chance