# Web Mining and Recommender Systems

## Text Mining

# Learning Goals

- Introduce the topic of text mining
- Describe some of the difficulties of dealing with textual data

# Administrivia

- Midterm will be handed out **after class** next Monday Nov 9 (6:30pm PST) – due 24hr later
  - We'll do prep on Monday beforehand

- I'll release a pdf of the midterm along with a code stub. You will submit a pdf to gradescope

# Prediction tasks involving text

What kind of quantities can we model, and what kind of prediction tasks can we solve using **text?**

# Prediction tasks involving text

Does this article have a positive or negative sentiment about the subject being discussed?

## What can stop US Postal Service trucks? The inexorable march of time

The ageing fleet of delivery vehicles is long past due an overhaul. Among the common-sense upgrades employees want: air conditioning and more workspace

Neither snow nor rain nor heat nor gloom of night stays these trucks – but time, it turns out, will. Photograph: Bill Sikes/AP

For the better part of the last 30 years, the flatulent buzz of the US Postal Service's boxy delivery vans - audible as they lighted from mailbox to mailbox - has been a familiar sound to most Americans. Neither snow nor rain nor heat nor gloom of night stays the USPS's mail trucks from the swift completion of their appointed

# Prediction tasks involving text

## What is the category/subject/topic of this article?

### *Apple Is Forming an Auto Team*

By BRIAN X. CHEN and MIKE ISAAC FEB. 19, 2015

Email

Share

Tweet

Save

More

SAN FRANCISCO — While Apple has been preparing to release its first wearable computers, the company has also been busy assembling a team to work on an automobile.

The company has collected about 200 people over the last few years — both from inside Apple and potential competitors like Tesla — to develop technologies for an electric car, according to two people with knowledge of the company's plans, who asked not to be named because the plans were private.

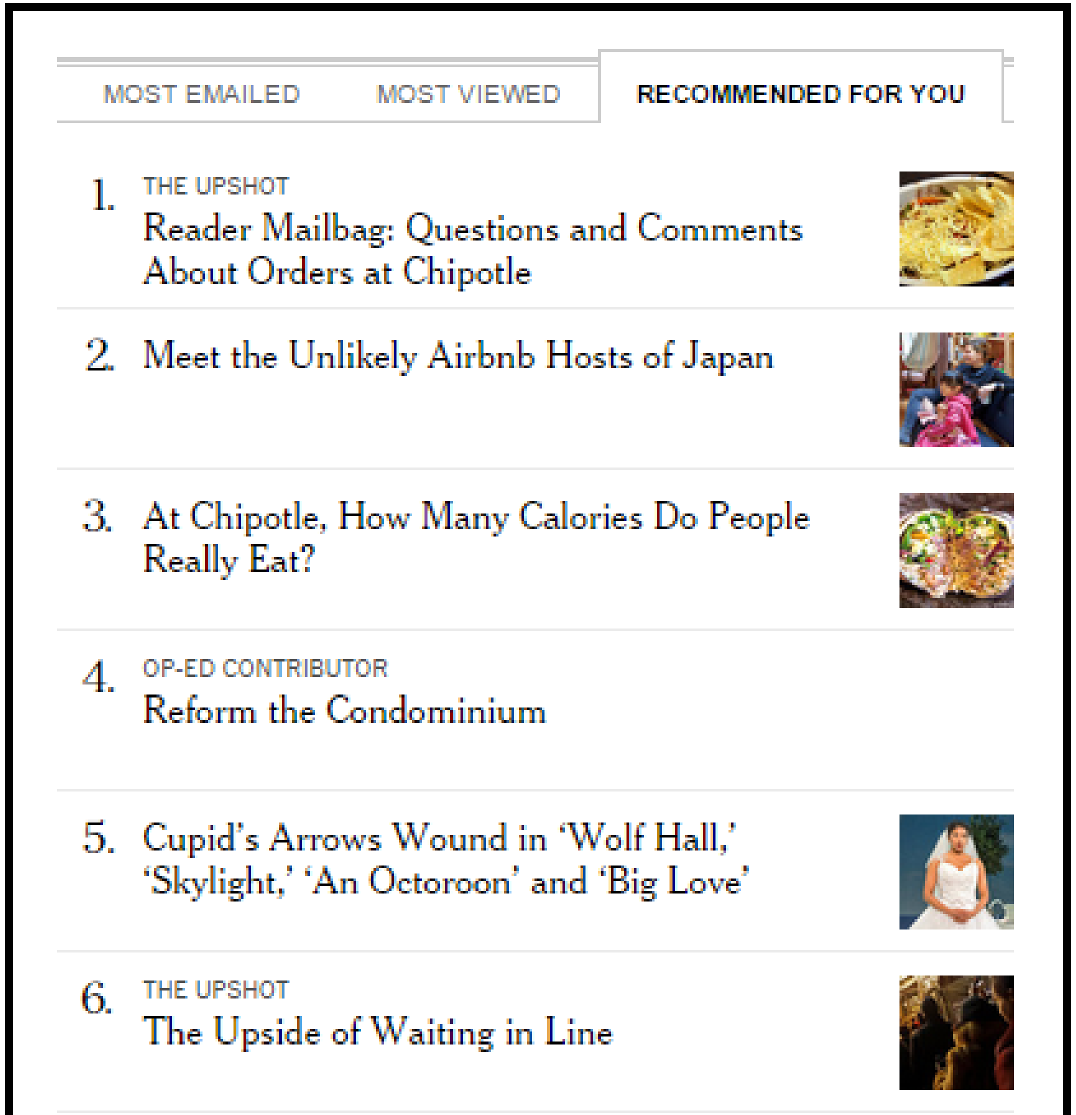
The car project is still in its prototype phase, one person said, meaning it is probably many years away from being a viable product and might never reach the mass market if the quality of the vehicle fails to impress Apple's executives.

Electric car batteries being prepared for shipment at the A123 Systems plant in Livonia, Mich in 2012. Apple has hired engineers from A123 Systems. Stephen McGee for The New York Times

It could also go nowhere if Apple struggles to find a compelling business opportunity in automobiles, a business that typically has much lower sales margins than

# Prediction tasks involving text

Which of these articles are relevant to my interests?



# Prediction tasks involving text

## Find me articles similar to this one

Meatloaf That Conquers the Mundane

FEB. 13, 2015

City Kitchen

By DAVID TANIS

Email
Share
Tweet
Pin
Save
More

I was raised on Midwestern meatloaf. My mother's dependable recipe did not vary: Ground beef, grated onion and carrot and a little oatmeal were the main ingredients, along with a dash of "seasoned salt." A ribbon of bottled chili sauce ran down a gully in the center.

Served hot, accompanied by Tater Tots, it was dinner. Served cold for lunch, it was always a sandwich on white bread, with potato chips on the side. It was usually moist and tasty but never remarkable, and there was no way you could call it anything but meatloaf.

Do I harbor a kind of nostalgia for it? Yes. But would I use that recipe now? I think not.

Evan Sung for The New York Times

I have a friend from Brussels who loves to entertain. Of his dinner party repertoire, one dish is most requested and admired. It is pain de veau, served with a vermouth-splashed mushroom sauce. In French, it sounds elegant. Translated into English — veal loaf — it sounds dull.

The Italian word for meatloaf is polpettone. (Polpette are Italian meatballs; polpettine are meatballs, too, but more diminutive.) This substantial family-size meatball, whether ovoid or elongated, plain or fancy, served with tomato sauce or not, is beloved both in Italy and in Italian communities throughout the world. Aside from its melodic, polysyllabic name, polpettone is always well seasoned, prepared with care and served with gusto.

It is usually a combination of different kinds of ground meat, typically beef, pork and veal

RELATED COVERAGE

City Kitchen: How to Make Polpettone, Step by Step FEB. 13, 2015

RECIPES FROM COOKING

Polpettone with Spinach and Provolone

By David Tanis

related articles

# Prediction tasks involving text

## Which of these reviews am I most likely to agree with or find helpful?

### Most Helpful Customer Reviews

1,900 of 1,928 people found the following review helpful

**Le Creuset on a budget**

By N. Lafond on October 24, 2007

Color Name: Caribbean Blue | Size Name: 6 qt | Verified Purchase

Enamel on cast iron cookware like this, was, until recently, only available from makers like Le Creuset. Lately, several lower cost makers have come on the scene, like Target and Innova. The new budget priced Lodge cookware is in the same price range as the low cost alternatives but completely out performs them.

I have all of the brands I have mentioned. The Lodge is the same weight as the Le Creuset which is much heavier than the other budget models. The ridge where the lid and sides meet is a matt black porcelain on the Lodge and Le Creuset but is just exposed cast iron for the other budget models (which leads to rusting if you are not careful). The porcelain resists staining (even tomato sauces) in the Lodge and Le Creuset but the other budget models stain very easily. And finally, the Lodge and Le Creuset maintain a very polished interior finish that resists sticking which others do not. So, I see no performance differences at all between the Le Creuset and the Lodge whereas the comparably priced budget models are certainly inferior.

If you plan of using these pots very heavily (every day for example) you might want to upgrade to the higher priced Lodge product. It has 4 coatings of enamel as opposed to 2 in this model. But if you use them once or twice a week I dont think you will need the added wear resistance.

47 Comments | Was this review helpful to you? Yes No

1,105 of 1,164 people found the following review helpful

**OK pot, Great Price. Some flaws.**

By J. G. Pavlovich on March 2, 2008

Color Name: Island Spice Red | Size Name: 6 qt | Verified Purchase

This is a terrific value. The quality and performance match my Le Creuset pieces at a fraction of the price. The only slight design flaw I have found is that the rounded bottom makes browning large pieces of meat awkward. Other than that I have no complaints. Even heating. Easy clean up. I use it several times a week.
UPDATE: I found a second minor problem. The inside rim of the lid has a couple of raised spots which prevent the lid from seating tightly. This causes steam to escape much faster than I would like during a long braise or stew.

# Prediction tasks involving text

## Which of these sentences best summarizes people's opinions?

# Prediction tasks involving text

## Which sentences refer to which aspect of the product?

'Partridge in a Pear Tree', brewed by 'The Bruery'

Dark brown with a light tan head, minimal lace and low retention. Excellent aroma of dark fruit, plum, raisin and red grape with light vanilla, oak, caramel and toffee. Medium thick body with low carbonation. Flavor has strong brown sugar and molasses from the start over bready yeast and a dark fruit and plum finish. Minimal alcohol presence. Actually, this is a nice quad.

Feel: 4.5 Look: 4 Smell: 4.5 Taste: 4 Overall: 4

# Today

## Using **text** to solve predictive tasks

- How to represent **documents** using **features?**
- Is text **structured** or **unstructured?**
- Does structure actually help us?
- How to account for the fact that most words may not convey much information?
- How can we find **low-dimensional** structure in text?

# Web Mining and Recommender Systems

## Bag-of-words models

# Feature vectors from text

We'd like a fixed-dimensional representation of documents, i.e., we'd like to describe them using **feature vectors**

This will allow us to compare documents, and associate weights with particular features to solve predictive tasks etc. (i.e., the kind of things we've been doing already)

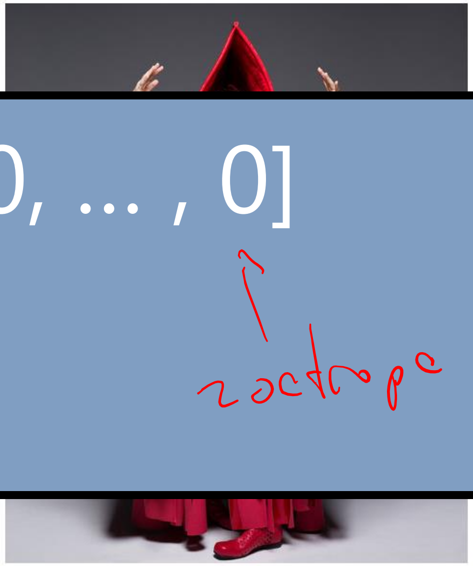
# Feature vectors from text

**Option 1:** just count how many times each word appears in each document

## The Peculiar Genius of Bjork

CULTURE | BY EMILY WITT | JANUARY 23, 2015 11:30 AM

*Solo musician or master collaborator? For her new album, Bjork has merged the two sides of her artistry to create a new experience of music — again.*



F_text = [150, 0, 0, 0, 0, 0, ... , 0]

a aardvark zoetrope

musician, who creates her music in an emotional cocoon, tinkering with technologies, concepts and feelings; and Bjork the producer and curator, who seeks out

# Feature vectors from text

## **Option 1:** just count how many times each word appears in each document

| | |
|---|---|
| Dark brown with a light tan head, minimal lace and low retention. Excellent aroma of dark fruit, plum, raisin and red grape with light vanilla, oak, caramel and toffee. Medium thick body with low carbonation. Flavor has strong brown sugar and molasses from the start over bready yeast and a dark fruit and plum finish. Minimal alcohol presence. Actually, this is a nice quad. | yeast and minimal red body thick light a Flavor sugar strong quad. grape over is molasses lace the low and caramel fruit Minimal start and toffee. dark plum, dark brown Actually, alcohol Dark oak, nice vanilla, has brown of a with presence. light carbonation. bready from retention. with finish. with and this and plum and head, fruit, low a Excellent raisin aroma Medium tan |

These two documents have **exactly** the same representation in this model, i.e., we're completely **ignoring** syntax.
This is called a "bag-of-words" model.

# Feature vectors from text

**Option 1:** just count how many times each word appears in each document

We've already seen some (potential) problems with this type of representation (dimensionality reduction), but let's see what we can do to get it working

# Feature vectors from text

50,000 reviews are available on :
http://cseweb.ucsd.edu/classes/fa20/cse258-a/data/beer_50000.json
(see course webpage)

Code on course webpage

# Feature vectors from text

## **Q1:** How many words are there?

```
wordCount = defaultdict(int)
for d in data:
  for w in d['review/text'].split():
    wordCount[w] += 1

print len(wordCount)
```

~36k

# Feature vectors from text

## **2:** What if we remove capitalization/punctuation?

```
wordCount = defaultdict(int)
punctuation = set(string.punctuation)
for d in data:
  for w in d['review/text'].split():
    w = ''.join([c for c in w.lower() if not c in punctuation])
    wordCount[w] += 1

print len(wordCount)
```

~19k

# Feature vectors from text

## **3:** What if we merge different inflections of words?

drinks → drink
drinking → drink
drinker → drink

argue → argu
arguing → argu
argues → argu
arguing → argu
argus → argu

# **3:** What if we merge different inflections of words?

This process is called “stemming”

- The first stemmer was created by Julie Beth Lovins (in 1968!!)
- The most popular stemmer was created by Martin Porter in 1980

# Feature vectors from text

## **3:** What if we merge different inflections of words?

### The algorithm is (fairly) simple but depends on a huge number of rules

**Step 1a**

```
SSES -> SS                     caresses  ->  caress
IES  -> I                      ponies    ->  poni
                               ties      ->  ti
SS   -> SS                     caress    ->  caress
S    ->                        cats      ->  cat
```

**Step 1b**

```
(m>0) EED -> EE                feed      ->  feed
                               agreed    ->  agree
(*v*) ED  ->                   plastered ->  plaster
                               bled      ->  bled
(*v*) ING ->                   motoring  ->  motor
                               sing      ->  sing
```

If the second or third of the rules in Step 1b is successful, the following is done:

```
AT -> ATE                      conflat(ed)  ->  conflate
BL -> BLE                      troubl(ed)   ->  trouble
IZ -> IZE                      siz(ed)      ->  size
(*d and not (*L or *S or *Z))
   -> single letter
                               hopp(ing)    ->  hop
                               tann(ed)     ->  tan
                               fall(ing)    ->  fall
                               hiss(ing)    ->  hiss
                               fizz(ed)     ->  fizz
(m=1 and *o) -> E              fail(ing)    ->  fail
                               fil(ing)     ->  file
```

The rule to map to a single letter causes the removal of one of the double letter pair. The -E is put back on

recognised lat

**Step 1c**

```
(*v*) Y -> I                   happy     ->  happi
```

**Step 2**

```
(m>0) ATIONAL ->  ATE          relational     ->  relate
(m>0) TIONAL  ->  TION         conditional    ->  condition
                               rational       ->  rational
(m>0) ENCI    ->  ENCE         valenci        ->  valence
(m>0) ANCI    ->  ANCE         hesitanci      ->  hesitance
(m>0) IZER    ->  IZE          digitizer      ->  digitize
(m>0) ABLI    ->  ABLE         conformabli    ->  conformable
(m>0) ALLI    ->  AL           radicalli      ->  radical
(m>0) ENTLI   ->  ENT          differentli    ->  different
(m>0) ELI     ->  E            vileli        - >  vile
(m>0) OUSLI   ->  OUS          analogousli    ->  analogous
(m>0) IZATION ->  IZE          vietnamization ->  vietnamize
(m>0) ATION   ->  ATE          predication    ->  predicate
(m>0) ATOR    ->  ATE          operator       ->  operate
(m>0) ALISM   ->  AL           feudalism      ->  feudal
(m>0) IVENESS ->  IVE          decisiveness   ->  decisive
(m>0) FULNESS ->  FUL          hopefulness    ->  hopeful
(m>0) OUSNESS ->  OUS          callousness    ->  callous
(m>0) ALITI   ->  AL           formaliti      ->  formal
(m>0) IVITI   ->  IVE          sensitiviti    ->  sensitive
(m>0) BILITI  ->  BLE          sensibiliti    ->  sensible
```

The test for the string S1 can be made fast by doing a program switch on the penultimate letter of the word being tested. This gives a fairly even breakdown of the possible values of the string S1. It will be seen in fact that the S1-strings in step 2 are presented here in the alphabetical order of their penultimate letter. Similar techniques may be applied in the other steps.

**Step 3**

```
(m>0) NESS  ->                 goodness       ->  good
```

**Step 4**

```
(m>1) AL    ->                 revival     ->  reviv
(m>1) ANCE  ->                 allowance   ->  allow
(m>1) ENCE  ->                 inference   ->  infer
(m>1) ER    ->                 airliner    ->  airlin
(m>1) IC    ->                 gyroscopic  ->  gyroscop
(m>1) ABLE  ->                 adjustable  ->  adjust
(m>1) IBLE  ->                 defensible  ->  defens
(m>1) ANT   ->                 irritant    ->  irrit
(m>1) EMENT ->                 replacement ->  replac
(m>1) MENT  ->                 adjustment  ->  adjust
(m>1) ENT   ->                 dependent   ->  depend
(m>1 and (*S or *T)) ION ->    adoption    ->  adopt
(m>1) OU    ->                 homologou   ->  homolog
(m>1) ISM   ->                 communism   ->  commun
(m>1) ATE   ->                 activate    ->  activ
(m>1) ITI   ->                 angulariti  ->  angular
(m>1) OUS   ->                 homologous  ->  homolog
(m>1) IVE   ->                 effective   ->  effect
(m>1) IZE   ->                 bowdlerize  ->  bowdler
```

The suffixes are now removed. All that remains is a little tidying up.

**Step 5a**

```
(m>1) E     ->                 probate     ->  probat
                               rate        ->  rate
(m=1 and not *o) E ->          cease       ->  ceas
```

**Step 5b**

```
ontrol
oll
```

http://telemat.det.unifi.it/book/2001/wchange/download/stem_porter.html

# Feature vectors from text

## **3:** What if we merge different inflections of words?

```
wordCount = defaultdict(int)
punctuation = set(string.punctuation)
stemmer = nltk.stem.porter.PorterStemmer()
for d in data:
  for w in d['review/text'].split():
    w = ''.join([c for c in w.lower() if not c in punctuation])
    w = stemmer.stem(w)
    wordCount[w] += 1

print len(wordCount)
```

~14k

# **3:** What if we merge different inflections of words?

- Stemming is **critical** for retrieval-type applications (e.g. we want Google to return pages with the word "cat" when we search for "cats")
- Personally I tend not to use it for predictive tasks. Words like "waste" and "wasted" may have different meanings (in beer reviews), and we're throwing that away by stemming

# Feature vectors from text

## **4:** Just discard extremely rare words…

```
counts = [(wordCount[w], w) for w in wordCount]
counts.sort()
counts.reverse()

words = [x[1] for x in counts[:1000]]
```

- Pretty unsatisfying but at least we can get to some inference now!

# Feature vectors from text

## Let's do some inference!

## **Problem 1:** Sentiment analysis

Let's build a predictor of the form:

$$f(\text{text}) \rightarrow \text{rating}$$

using a model based on linear regression:

$$\text{rating} \simeq \alpha + \sum_{w \in \text{text}} \text{count}(w) \cdot \theta_w$$

$X_i \cdot \theta$

Code on course webpage

# Feature vectors from text

## What do the parameters look like?

$$\theta_{\text{fantastic}} = 0.143$$

$$\theta_{\text{watery}} = -0.163$$

$$\theta_{\text{and}} = -0.008$$

$$\theta_{\text{me}} = -0.037$$

# Feature vectors from text

## Why might parameters associated with "and", "of", etc. have non-zero values?

- Maybe they have meaning, in that they might frequently appear slightly more often in positive/negative phrases
- Or maybe we're just measuring the length of the review…

How to fix this (and is it a problem)?

1) Add the length of the review to our feature vector
2) Remove stopwords

# Feature vectors from text

## Removing stopwords:

```
from nltk.corpus import stopwords
stopwords.words("english")
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you',
'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself',
'she', 'her', 'hers', 'herself', 'it', 'its', 'itself', 'they', 'them',
'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this',
'that', 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been',
'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing',
'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until',
'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between',
'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to',
'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again',
'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why',
'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other',
'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than',
'too', 'very', 's', 't', 'can', 'will', 'just', 'don', 'should', 'now']
```

# Feature vectors from text

## Why remove stopwords?

some (potentially inconsistent) reasons:

- They convey little information, but are a substantial fraction of the corpus, so we can reduce our corpus size by ignoring them
- They **do** convey information, but only by being correlated by a feature that we don't want in our model
- They make it more difficult to reason about which features are informative (e.g. they might make a model harder to visualize)
- We're confounding their importance with that of phrases they appear in (e.g. words like "The Matrix", "The Dark Night", "The Hobbit" might predict that an article is about movies)

so use n-grams!

# Feature vectors from text

## We can build a richer predictor by using **n-grams**

e.g. "Medium thick body with low carbonation."

unigrams: `["medium", "thick", "body", "with", "low", "carbonation"]`

bigrams: `["medium thick", "thick body", "body with", "with low", "low carbonation"]`

trigrams: `["medium thick body", "thick body with", "body with low", "with low carbonation"]`

etc.

# We can build a richer predictor by using **n-grams**

- Fixes some of the issues associated with using a bag-of-words model – namely we recover some basic **syntax** – e.g. "good" and "not good" will have different weights associated with them in a sentiment model
- Increases the **dictionary size** by a lot, and increases the sparsity in the dictionary even further
- We might end up double (or triple-)-counting some features (e.g. we'll predict that "Adam Sandler", "Adam", and "Sandler" are associated with negative ratings, even though they're all referring to the same concept)

# Feature vectors from text

## We can build a richer predictor by using **n-grams**

- This last problem (that of double counting) is bigger than it seems: We're **massively** increasing the number of features, but possibly increasing the number of **informative** features only slightly
- So, for a **fixed-length** representation (e.g. 1000 most-common words vs. 1000 most-common words+bigrams) the bigram model will quite possibly perform **worse** than the unigram model

## **Problem 2:** Classification

Let's build a predictor of the form:

$$f(\text{text}) \rightarrow \text{class label}$$

# So far…

## Bags-of-words representations of text

- Stemming & stopwords
- Unigrams & N-grams
- Sentiment analysis & text classification

# References

## Further reading:

- Original stemming paper

"Development of a stemming algorithm" (Lovins, 1968):
http://mt-archive.info/MT-1968-Lovins.pdf

- Porter's paper on stemming

"An algorithm for suffix stripping" (Porter, 1980):
http://telemat.det.unifi.it/book/2001/wchange/download/stem_porter.html

# Web Mining and Recommender Systems

## TF-IDF

# Distances and dimensionality reduction

When we studied recommender systems, we looked at:

- Approaches based on measuring similarity (cosine, jaccard, etc.)
- Approaches based on dimensionality reduction

We'll look at the same two concepts, but using textual representations

# Finding relevant terms

So far we've dealt with huge vocabularies just by identifying the **most frequently occurring** words

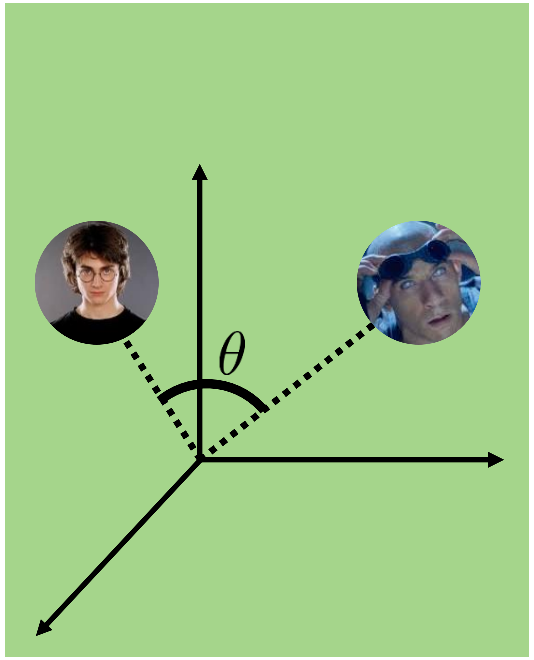
**But!** The most informative words may be those that occur very rarely, e.g.:

- Proper nouns (e.g. people's names) may predict the content of an article even though they show up rarely
- Extremely superlative (or extremely negative) language may appear rarely but be very predictive

# Finding relevant terms

e.g. imagine applying something like cosine similarity to the document representations we've seen so far

e.g. are (the features of the reviews/IMDB descriptions of) these two documents "similar", i.e., do they have high cosine similarity



# Finding relevant terms

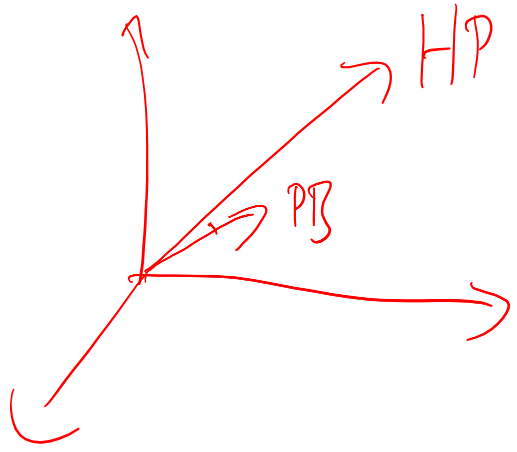
e.g. imagine applying something like cosine similarity to the document representations we've seen so far

PB = [181, 158, 161, ... 2 ... 1 ...]

"exciting"

HP = [256, 220, 201, ... 5 ...]

"magic"



# Finding relevant terms

## So how can we estimate the “relevance” of a word in a document?

### e.g. which words in this document might help us to determine its content, or to find similar documents?

Despite Taylor making moves to end her long-standing feud with Katy, HollywoodLife.com has learned exclusively that Katy isn’t ready to let things go! Looks like the bad blood between Kat Perry, 29, and Taylor Swift, 25, is going to continue brewing. A source tells HollywoodLife.com exclusively that Katy prefers that their frenemy battle lines remain drawn, and we’ve got all the scoop on why Katy is set in her ways. Will these two *ever* bury the hatchet? Katy Perry & Taylor Swift Still Fighting? “Taylor’s tried to reach out to make amends with Katy, but Katy is not going to accept it nor is she interested in having a friendship with Taylor,” a source tells HollywoodLife.com exclusively. “She wants nothing to do with Taylor. In Katy’s mind, Taylor shouldn’t even attempt to make a friendship happen. That ship has sailed.” While we love that Taylor has tried to end the feud, we can understand where Katy is coming from. If a friendship would ultimately never work, then why bother? These two have taken their feud everywhere from social media to magazines to the Super Bowl. Taylor’s managed to mend the fences with Katy’s BFF Diplo, but it looks like Taylor and Katy won’t be posing for pics together in the near future. Katy Perry & Taylor Swift: Their Drama Hits All-Time High At the very least, Katy and Taylor could tone down their feud. That’s not too much to ask

# So how can we estimate the “relevance” of a word in a document?

## e.g. which words in this document might help us to determine its content, or to find similar documents?

Despite Taylor making moves to end her long-standing feud with Katy, HollywoodLife.com has learned exclusively that Katy isn’t ready to let things go! Looks like the bad blood between Kat Perry, 29, and Taylor Swift, 25, is going to continue brewing. A source tells HollywoodLife.com exclusively that Katy prefers that their frenemy battle lines remain drawn, and we’ve got all the scoop on why Katy … ill these two *ever* bury the hatchet? Katy Perry & Taylor Swift Still Fighting? “Tay… ut to make amends with Katy, but Katy is not going to accept it nor is she inte… ndship with Taylor,” a source tells HollywoodLife.com exclusively. “She wan… Taylor. In Katy’s mind, Taylor shouldn’t even attempt to make a friendship happen. That ship has sailed.” While we love that Taylor has tried to end the feud, we can understand where Katy is coming from. If a friendship would ultimately never work, then why bother? These two have taken their feud everywhere from social media to magazines to the Super Bowl. Taylor’s managed to mend the fences with Katy’s BFF Diplo, but it looks like Taylor and Katy won’t be posing for pics together in the near future. Katy Perry & Taylor Swift: Their Drama Hits All-Time High At the very least, Katy and Taylor could tone down their feud. That’s not too much to ask

“the” appears 12 times in the document

# Finding relevant terms

## So how can we estimate the “relevance” of a word in a document?

### e.g. which words in this document might help us to determine its content, or to find similar documents?

Despite Taylor making moves to end her long-standing feud with Katy, HollywoodLife.com has learned exclusively that Katy isn’t ready to let things go! Looks like the bad blood between Kat Perry, 29, and Taylor Swift, 25, is going to continue brewing. A source tells HollywoodLife.com exclusively that Katy prefers that their frenemy battle lines remain drawn, and we’ve got all the scoop on why
Katy ... ill t ... e hatchet? Katy Perry & Taylor Swift Still Fighting?
“Tay ... ut to ... Katy, but Katy is not going to accept it nor is she
inte ... ends ... urce tells HollywoodLife.com exclusively. “She
wan ... Tay ... ylor shouldn’t even attempt to make a friendship happen. That ship has sailed.” While we love that Taylor has tried to end the feud, we can understand where Katy is coming from. If a friendship would ultimately never work, then why bother? These two have taken their feud everywhere from social media to magazines to the Super Bowl. Taylor’s managed to mend the fences with Katy’s BFF Diplo, but it looks like Taylor and Katy won’t be posing for pics together in the near future. Katy Perry & Taylor Swift: Their Drama Hits All-Time High At the very least, Katy and Taylor could tone down their feud. That’s not too much to ask

“the” appears 12 times in the document

“Taylor Swift” appears 3 times in the document

# Finding relevant terms

## So how can we estimate the “relevance” of a word in a document?

**Q:** The document discusses “the” more than it discusses “Taylor Swift”, so how might we come to the conclusion that “Taylor Swift” is the more relevant expression?

**A:** It discusses “the” **no more** than other documents do, but it discusses “Taylor Swift” **much more**

# Finding relevant terms

## Term frequency & document frequency

**Term frequency** ~ How much does the term appear in the document

**Inverse document frequency** ~ How “rare” is this term across all documents

# Finding relevant terms

## Term frequency & document frequency

$$tf(w, d) = \#\text{times } w \text{ appears in } d$$
$$= |\{t \in d \mid t = w\}|$$

$$df(w, D) = \#\text{docs that contain } d$$
$$= |\{d \in D \mid w \in d\}|$$

# Finding relevant terms

## Term frequency & document frequency

"Term frequency": $tf(t, d)$ = number of times the term *t* appears in the document *d*

e.g. tf("Taylor Swift", that news article) = 3

"Inverse document frequency": $idf(t, D) = log \frac{N}{|\{d \in D : t \in d\}|}$ $= |D|$

term (e.g. "Taylor Swift")

set of documents

"Justification": $P(t|D) = \frac{|\{d \in D : t \in d\}|}{N}$ so $idf(t, D) = -\log P(t|D)$

# Finding relevant terms

## Term frequency & document frequency

**TF-IDF** is high → this word appears much more frequently in this document compared to other documents

**TF-IDF** is low → this word appears infrequently in this document, or it appears in many documents

$$tfidf(t, d, D) = tf(t, d) \times idf(t, D)$$

# Term frequency & document frequency

**tf** is sometimes defined differently, e.g.:

$$tf'(t,d) = \delta(t \in d)$$

$$tf''(t,d) = \frac{\text{frequency of word}}{\text{frequency of most common word in document}}$$

Both of these representations are invariant to the document length, compared to the regular definition which assigns higher weights to longer documents

# Finding relevant terms

## How to use TF-IDF

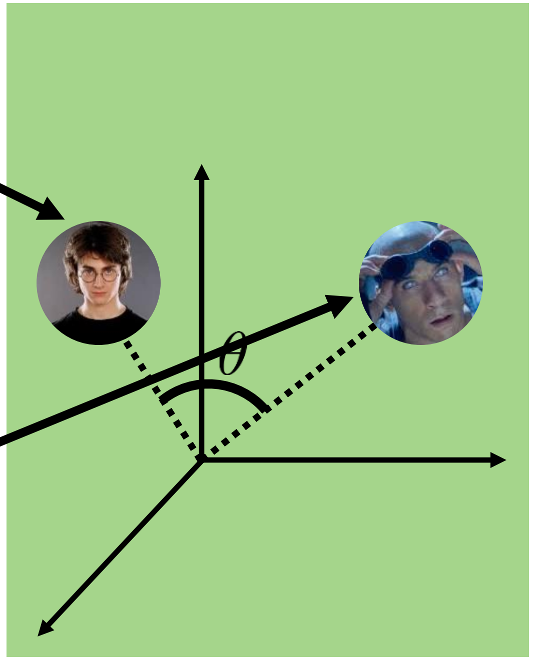
**“fantasy”**

[0,0,0.01,0,0.6,…,0.04,0,3,0,159.1,0]

“the” “and”

[180.2,0,0.01,0.5,0,…,0.02,0,0.2,0,0,0]

**“action”**




- Frequently occurring words have little impact on the similarity
- The similarity is now determined by the words that are most “characteristic” of the document

# Finding relevant terms

But what about when we're **weighting** the parameters anyway?

e.g. is:

$$\text{rating} \simeq \alpha + \sum_{w \in \text{text}} \text{count}(w) \cdot \theta_w$$

really any different from:

$$\text{rating} \simeq \alpha + \sum_{w \in \text{text}} \mathit{tfidf}(w, d, D) \cdot \theta_w$$

after we fit parameters?

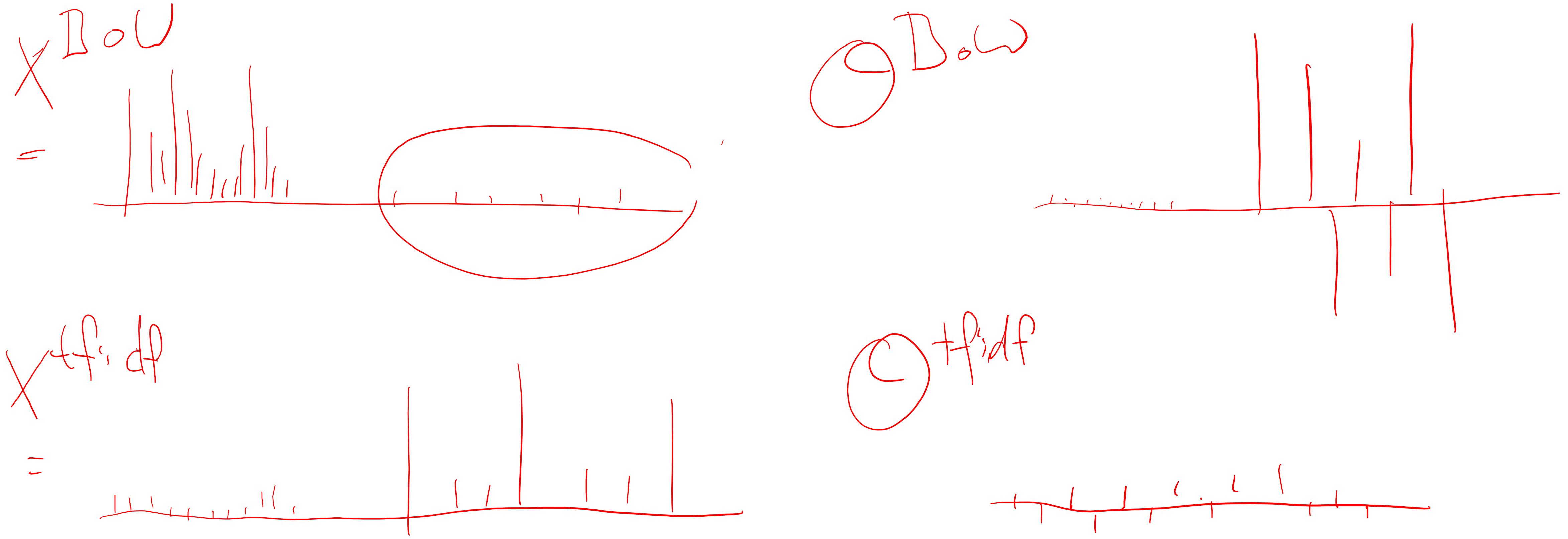
## But what about when we're **weighting** the parameters anyway?

## **Yes!**

- The **relative** weights of features is different between documents, so the two representations are not the same (up to scale)
- When we regularize, the scale of the features matters – if some "unimportant" features are very large, then the model can overfit on them "for free"

# Finding relevant terms

But what about when we're **weighting** the parameters anyway?



# Finding relevant terms

But what about when we're **weighting** the parameters anyway?

# References

## Further reading:

- Original TF-IDF paper (from 1972)

"A Statistical Interpretation of Term Specificity and Its Application in Retrieval"

http://goo.gl/1CLwUV

# Web Mining and Recommender Systems

## Dimensionality-reduction approaches to document representation

# Dimensionality reduction

## How can we find **low-dimensional structure** in documents?

### What we would like:

87 of 102 people found the following review helpful

☆☆☆☆☆ **You keep what you kill**, December 27, 2004

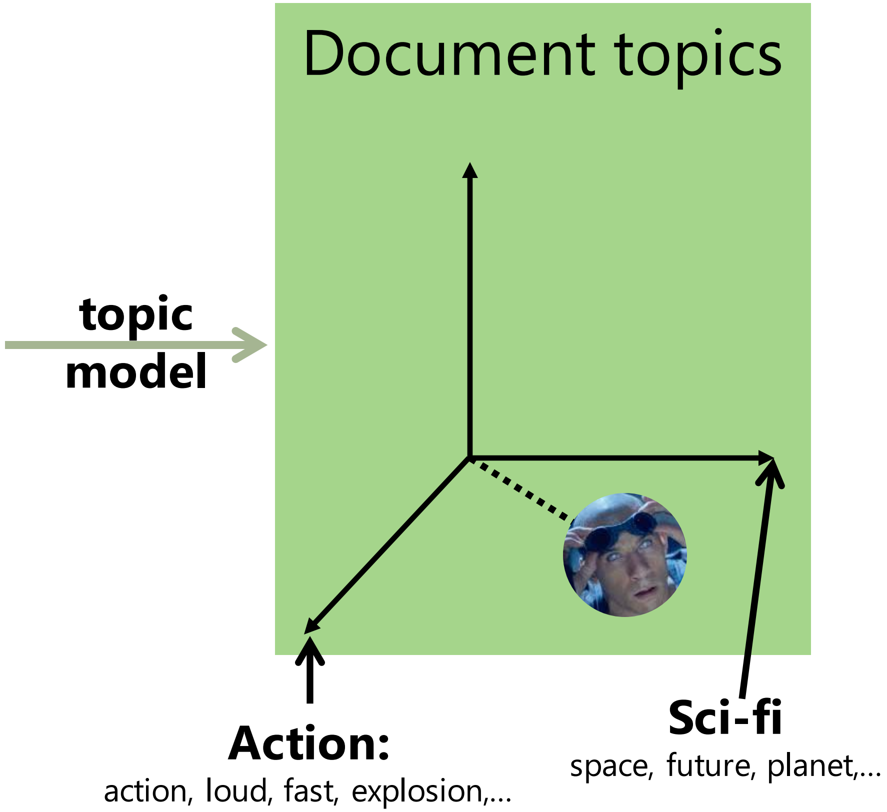
By Schtinky "Schtinky" (Washington State) - See all my reviews
VINE™ VOICE

This review is from: The Chronicles of Riddick (Widescreen Unrated Director's Cut) (DVD)

Even if I have to apologize to my Friends and Favorites, and my family, I have to admit that I really liked this movie. It's a Sci-Fi movie with a "Mad Maxx" appeal that, while changing many things, left Riddick from `Pitch Black' to be just Riddick. They did not change his attitude or soften him up or bring him out of his original character, which was very pleasing to `Pitch Black' fans like myself.

First off, let me say that when playing the DVD, the first selection to come up is Convert or Fight, and no explanation of the choices. This confused me at first, so I will mention off the bat that they are simply different menu formats, that each menu has the very same options, simply different background visuals. Select either one and continue with the movie.

(review of "The Chronicles of Riddick")



# Singular-value decomposition

## Recall (from dimensionality reduction / recommender systems)

$$R = \begin{pmatrix} 5 & 3 & \cdots & 1 \\ 4 & 2 & & 1 \\ 3 & 1 & & 3 \\ 2 & 2 & & 4 \\ 1 & 5 & & 2 \\ \vdots & & \ddots & \vdots \\ 1 & 2 & \cdots & 1 \end{pmatrix}$$

(e.g.) matrix of ratings

(square roots of) eigenvalues of $RR^T$

$$R = U \Sigma V^T$$

eigenvectors of $RR^T$

eigenvectors of $R^T R$

# Singular-value decomposition

Taking the eigenvectors corresponding to the top-K eigenvalues is then the "best" rank-K approximation

$$R = \begin{pmatrix} 5 & 3 & \cdots & 1 \\ 4 & 2 & & 1 \\ 3 & 1 & & 3 \\ 2 & 2 & & 4 \\ 1 & 5 & & 2 \\ \vdots & & \ddots & \vdots \\ 1 & 2 & \cdots & 1 \end{pmatrix}$$

(square roots of top k) eigenvalues of $RR^T$

$$R \simeq U^{(k)} \Sigma^{(k)} V^{(k)T}$$

(top k) eigenvectors of $RR^T$

(top k) eigenvectors of $R^T R$

# Singular-value decomposition

## What happens when we apply this to a matrix encoding our documents?

$$\underset{\text{document matrix}}{X} = \underbrace{\begin{pmatrix} 1 & 0 & \cdots & 4 \\ 0 & 2 & & 0 \\ 31 & 23 & & 97 \\ 0 & 98 & & 1 \\ 473 & 88 & & 347 \\ \vdots & & \ddots & \vdots \\ 11 & 34 & \cdots & 13 \end{pmatrix}}_{\text{documents}} \left.\vphantom{\begin{pmatrix} 1 \\ 0 \\ 31 \\ 0 \\ 473 \\ \vdots \\ 11 \end{pmatrix}}\right\} \text{terms}$$

*X* is a *TxD* matrix whose **columns** are bag-of-words representations of our documents

*T* = dictionary size
*D* = number of documents

# Singular-value decomposition

What happens when we apply this to a matrix encoding our documents?

$X^T X$ is a *DxD* matrix.

$U^{(k)}\sqrt{\Sigma^{(k)}}$ is a low-rank approximation of each **document**

eigenvectors of $X^T X$

$X X^T$ is a *TxT* matrix.

$V^{(k)}\sqrt{\Sigma^{(k)}}$ is a low-rank approximation of each **term**

eigenvectors of $X X^T$

# Singular-value decomposition

## What happens when we apply this to a matrix encoding our documents?

$$\begin{bmatrix} R \end{bmatrix} = \begin{bmatrix} \gamma_u \end{bmatrix} \begin{bmatrix} \gamma_I \end{bmatrix}$$

$\gamma_i$ one item

$\gamma_u$ one user

# Singular-value decomposition

## What happens when we apply this to a matrix encoding our documents?

$$T\ \underset{D}{\begin{bmatrix} BoW \end{bmatrix}} \approx \begin{bmatrix} \gamma_T \end{bmatrix} \begin{bmatrix} \gamma_D^T \end{bmatrix}$$

$\gamma_d$ one doc

$\gamma_t$ one term

# Singular-value decomposition

## Using our low rank representation of each **document** we can…

- Compare two documents by their low dimensional representations (e.g. by cosine similarity)
- To retrieve a document (by first projecting the query into the low-dimensional document space)
- Cluster similar documents according to their low-dimensional representations
- Use the low-dimensional representation as features for some other prediction task

# Singular-value decomposition

## Using our low rank representation of each **word** we can...

- Identify potential synonyms – if two words have similar low-dimensional representations then they should have similar "roles" in documents and are potentially synonyms of each other
- This idea can even be applied across languages, where similar terms in different languages ought to have similar representations in parallel corpora of translated documents

# This approach is called **latent semantic analysis**

- In practice, computing eigenvectors for matrices of the sizes in question is not practical – neither for XX^T nor X^TX (they won't even fit in memory!)
- Instead one needs to resort to some approximation of the SVD, e.g. a method based on stochastic gradient descent that never requires us to compute XX^T or X^TX directly (much as we did when approximating rating matrices with low-rank terms)

# Web Mining and Recommender Systems

word2vec

# Word2vec (Mikolov et al. 2013)

Goal: estimate the probability that a word appears *near* another (as opposed to Latent Semantic Analysis, which estimates a word count in a given document)

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t)$$

All tokens in document

Context window of *c* adjacent words

Probability that nearby word appears in the context of w_t

# Word2vec

In practice, this probability is modeled approximately by trying to maximize the score of words that cooccur and minimizes the score of words that don't:

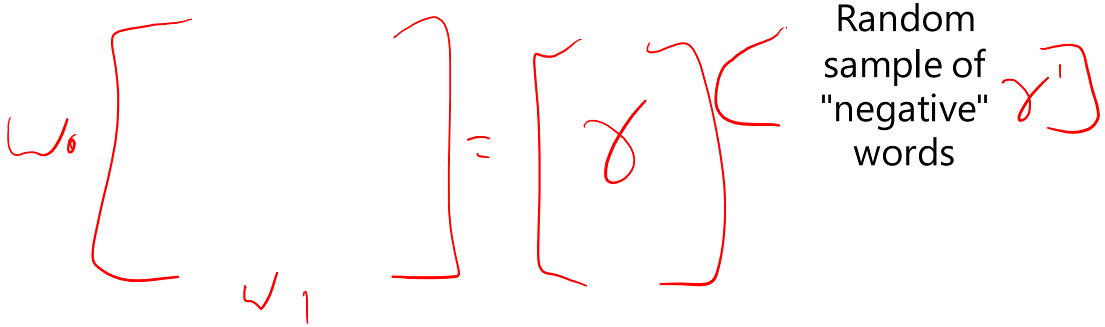
$$\log p(w_o|w_i) \simeq \sigma(\overbrace{\gamma'_{w_o} \cdot \gamma_{w_i}}^{\text{Co-occuring words should have compatible representations}}) + \sum_{w \in \mathcal{N}} \log \sigma(\overbrace{-\gamma'_{w} \cdot \gamma_{w_i}}^{\text{Words that don't co-occur should have low compatibility}})$$

Repr. of w_o

Repr. of w_i

Random sample of "negative" words

Note: Very similar to a **binary latent factor model!**



# Item2vec (Barkan and Koenigstein, 2016)

Given its similarity to a latent factor representation, this idea has been adapted to use *item* sequences rather than *word* sequences

$u_1 = [108, 16, 1, 5, 15]$

$u_2 = [1, 16, 58]$

$\vdots$

# Item2vec (Barkan and Koenigstein, 2016)

Given its similarity to a latent factor representation, this idea has been adapted to use *item* sequences rather than *word* sequences

$$\underbrace{\log p(i|j)}_{\text{Probability that item } i \text{ appears near } j} \simeq \sigma(\overbrace{\underbrace{\gamma'_i}_{\text{Repr. of item i}} \cdot \underbrace{\gamma_j}_{\text{Repr. of item j}}}^{\text{Co-occuring items should have compatible representations}}) + \sum_{\underbrace{i' \in \mathcal{N}}_{\text{Random sample of negative items}}} \log \sigma(\overbrace{-\gamma'_{i'} \cdot \gamma_j}^{\text{Items that don't co-occur should have low compatibility}})$$

# Word2Vec and Item2Vec in GenSim

(run on our 50k beer dataset)

```
from gensim.models import Word2Vec

model = Word2Vec(reviewTokens, # Tokenized documents (list of lists)
            min_count=5, # Minimum frequency before words are discarded
            size=10, # Model dimensionality K
            window=3, # Window size c
            sg=1) # Skip-gram model (what I described)

model.wv.similar_by_word("grassy")
```

$$\max_{w} \frac{\gamma_w \cdot \gamma_{grassy}}{||\gamma_w|| ||\gamma_{grassy}||}$$ = 'citrus', 'citric', 'floral', 'flowery', 'piney', 'herbal'

# Word2Vec and Item2Vec in GenSim

(run on our 50k beer dataset)

```
from gensim.models import Word2Vec

model = Word2Vec(itemSequences, # ordered sequences of items per user
          min_count=5, # Minimum frequency before items are discarded
          size=10, # Model dimensionality K
          window=3, # Window size c
          sg=1) # Skip-gram model (what I described)

model.wv.similar_by_word("Molson Canadian Light") # or really its itemID
```

Most similar items = 'Miller Light', 'Molsen Golden', 'Piels', 'Coors Extra Gold', 'Labatt Canadian Ale' (etc.)

# Word2Vec and Item2Vec in GenSim

- Note: this is a form of *item to item* recommendation, i.e., we learn which items appear in the context of other items, but there is no user representation

- This is actually a very effective way to make recommendations based on a few items a user has consumed, without having to explicitly model the user

# Web Mining and Recommender Systems

## Topic models

# Probabilistic modeling of documents

## Finally, can we represent documents in terms of the topics they describe?

**What we would like:**

87 of 102 people found the following review helpful

☆☆☆☆☆ **You keep what you kill**, December 27, 2004

By **Schtinky "Schtinky"** (Washington State) - See all my reviews

VINE™ VOICE

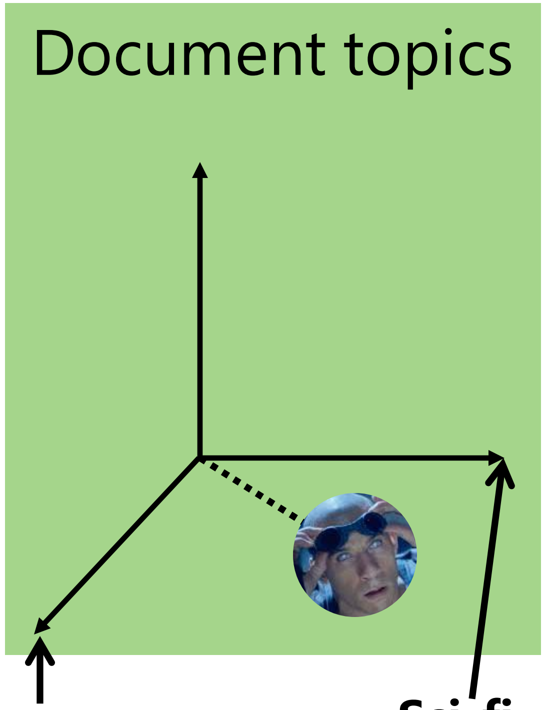
This review is from: **The Chronicles of Riddick (Widescreen Unrated Director's Cut) (DVD)**

Even if I have to apologize to my Friends and Favorites, and my family, I have to admit that I really liked this movie. It's a Sci-Fi movie with a "Mad Maxx" appeal that, while changing many things, left Riddick from `Pitch Black' to be just Riddick. They did not change his attitude or soften him up or bring him out of his original character, which was very pleasing to `Pitch Black' fans like myself.

First off, let me say that when playing the DVD, the first selection to come up is Convert or Fight, and no explanation of the choices. This confused me at first, so I will mention off the bat that they are simply different menu formats, that each menu has the very same options, simply different background visuals. Select either one and continue with the movie.

(review of "The Chronicles of Riddick")

**topic model**




**Action:**
action, loud, fast, explosion,...

**Sci-fi**
space, future, planet,...

# Probabilistic modeling of documents

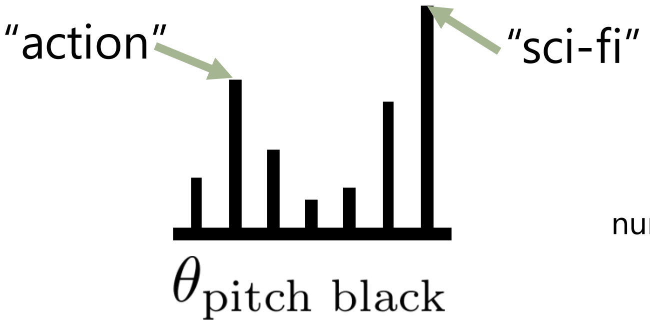
## Finally, can we represent documents in terms of the topics they describe?

- We'd like each document to be a **mixture over topics** (e.g. if movies have topics like "action", "comedy", "sci-fi", and "romance", then reviews of action/sci-fis might have representations like [0.5, 0, 0.5, 0])

action sci-fi

- Next we'd like each topic to be a **mixture over words** (e.g. a topic like "action" would have high weights for words like "fast", "loud", "explosion" and low weights for words like "funny", "romance", and "family")

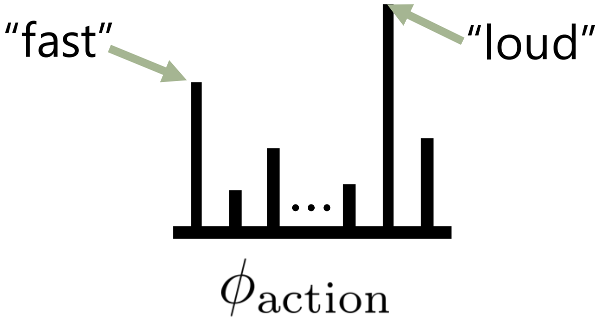
# Both of these can be represented by **multinomial distributions**




Each document has a **topic distribution** which is a mixture over the topics it discusses

number of topics

$\theta_d \in \Delta^K$ i.e., $\forall_d \sum_k \theta_{d,k} = 1$




Each topic has a **word distribution** which is a mixture over the words it discusses

number of words

$\phi_k \in \Delta^D$ i.e., $\forall_k \sum_w \phi_{k,w} = 1$

# Latent Dirichlet Allocation

## Under this model, we can estimate the probability of a particular bag-of-words appearing with a particular topic and word distribution

document

iterate over word positions

probability of this word's topic

probability of observing this word in this topic

$$p(d|\theta, \phi, z) = \prod_{j=1}^{\text{length of } d} \theta_{z_{d,j}} \phi_{z_{d,j}, w_{d,j}}$$

**Problem:** we need to estimate all this stuff before we can compute this probability!

# Latent Dirichlet Allocation

## E.g. some topics discovered from an Associated Press corpus

labels are determined manually

| "Arts" | "Budgets" | "Children" | "Education" |
|---|---|---|---|
| NEW | MILLION | CHILDREN | SCHOOL |
| FILM | TAX | WOMEN | STUDENTS |
| SHOW | PROGRAM | PEOPLE | SCHOOLS |
| MUSIC | BUDGET | CHILD | EDUCATION |
| MOVIE | BILLION | YEARS | TEACHERS |
| PLAY | FEDERAL | FAMILIES | HIGH |
| MUSICAL | YEAR | WORK | PUBLIC |
| BEST | SPENDING | PARENTS | TEACHER |
| ACTOR | NEW | SAYS | BENNETT |
| FIRST | STATE | FAMILY | MANIGAT |
| YORK | PLAN | WELFARE | NAMPHY |
| OPERA | MONEY | MEN | STATE |
| THEATER | PROGRAMS | PERCENT | PRESIDENT |
| ACTRESS | GOVERNMENT | CARE | ELEMENTARY |
| LOVE | CONGRESS | LIFE | HAITI |

# Latent Dirichlet Allocation

## And the topics most likely to have generated each word in a document

labels are determined manually

| "Arts" | "Budgets" | "Children" | "Education" |
|---|---|---|---|
| NEW | MILLION | CHILDREN | SCHOOL |
| FILM | TAX | WOMEN | STUDENTS |

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

From http://machinelearning.wustl.edu/mlpapers/paper_files/BleiNJ03.pdf

# Many many many extensions of Latent Dirichlet Allocation have been proposed:

- To handle temporally evolving data:

"Topics over time: a non-Markov continuous-time model of topical trends" (Wang & McCallum, 2006)
http://people.cs.umass.edu/~mccallum/papers/tot-kdd06.pdf

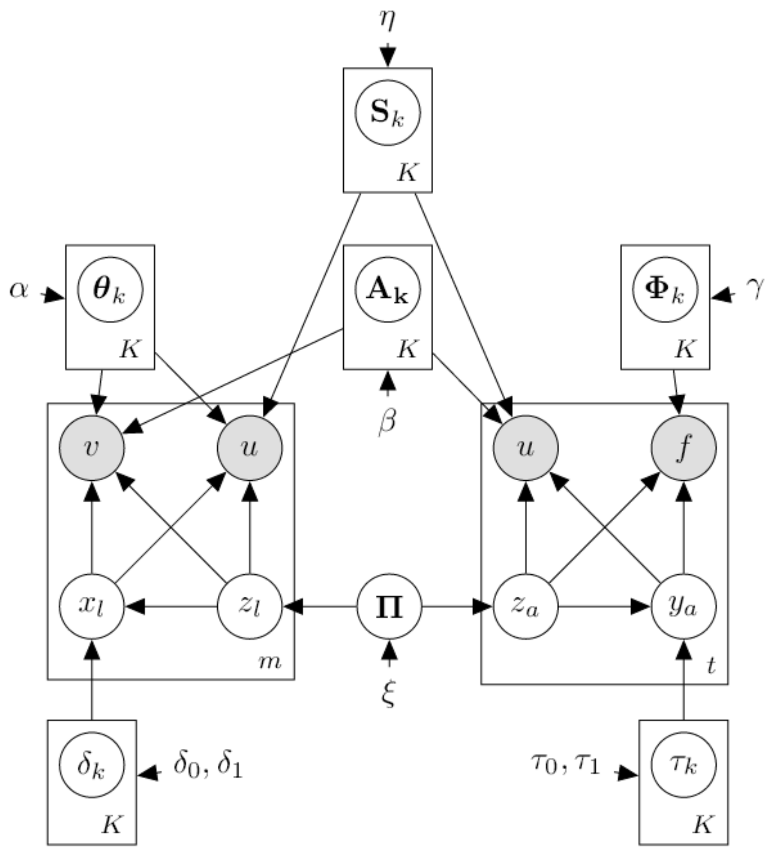
- To handle **relational** data:

"Block-LDA: Jointly modeling entity-annotated text and entity-entity links" (Balasubramanyan & Cohen, 2011)
http://www.cs.cmu.edu/~wcohen/postscript/sdm-2011-sub.pdf
"Relational topic models for document networks" (Chang & Blei, 2009)
https://www.cs.princeton.edu/~blei/papers/ChangBlei2009.pdf
"Topic-link LDA: joint models of topic and author community" (Liu, Nicelescu-Mizil, & Gryc, 2009)
http://www.niculescu-mizil.org/papers/Link-LDA2.crc.pdf

# Latent Dirichlet Allocation

## Many many many extensions of Latent Dirichlet Allocation have been proposed:

"WTFW" model (Barbieri, Bonch, & Manco, 2014), a model for relational documents



# Summary

## Using **text** to solve predictive tasks

- Representing documents using bags-of-words and TF-IDF weighted vectors
- Stemming & stopwords
- Sentiment analysis and classification

## Dimensionality reduction approaches:

- Latent Semantic Analysis
- Latent Dirichlet Allocation

# Questions?

## Further reading:

- Latent semantic analysis

"An introduction to Latent Semantic Analysis" (Landauer, Foltz, & Laham, 1998)
http://lsa.colorado.edu/papers/dp1.LSAintro.pdf

- LDA

"Latent Dirichlet Allocation" (Blei, Ng, & Jordan, 2003)
http://machinelearning.wustl.edu/mlpapers/paper_files/BleiNJ03.pdf

- Plate notation

http://en.wikipedia.org/wiki/Plate_notation
"Operations for Learning with Graphical Models" (Buntine, 1994)
http://www.cs.cmu.edu/afs/cs/project/jair/pub/volume2/buntine94a.pdf