Michael Grossberg

Intro to Data Science CS59969

Text

Machine Learning on Text

Why???

Traditional Human Knowledge

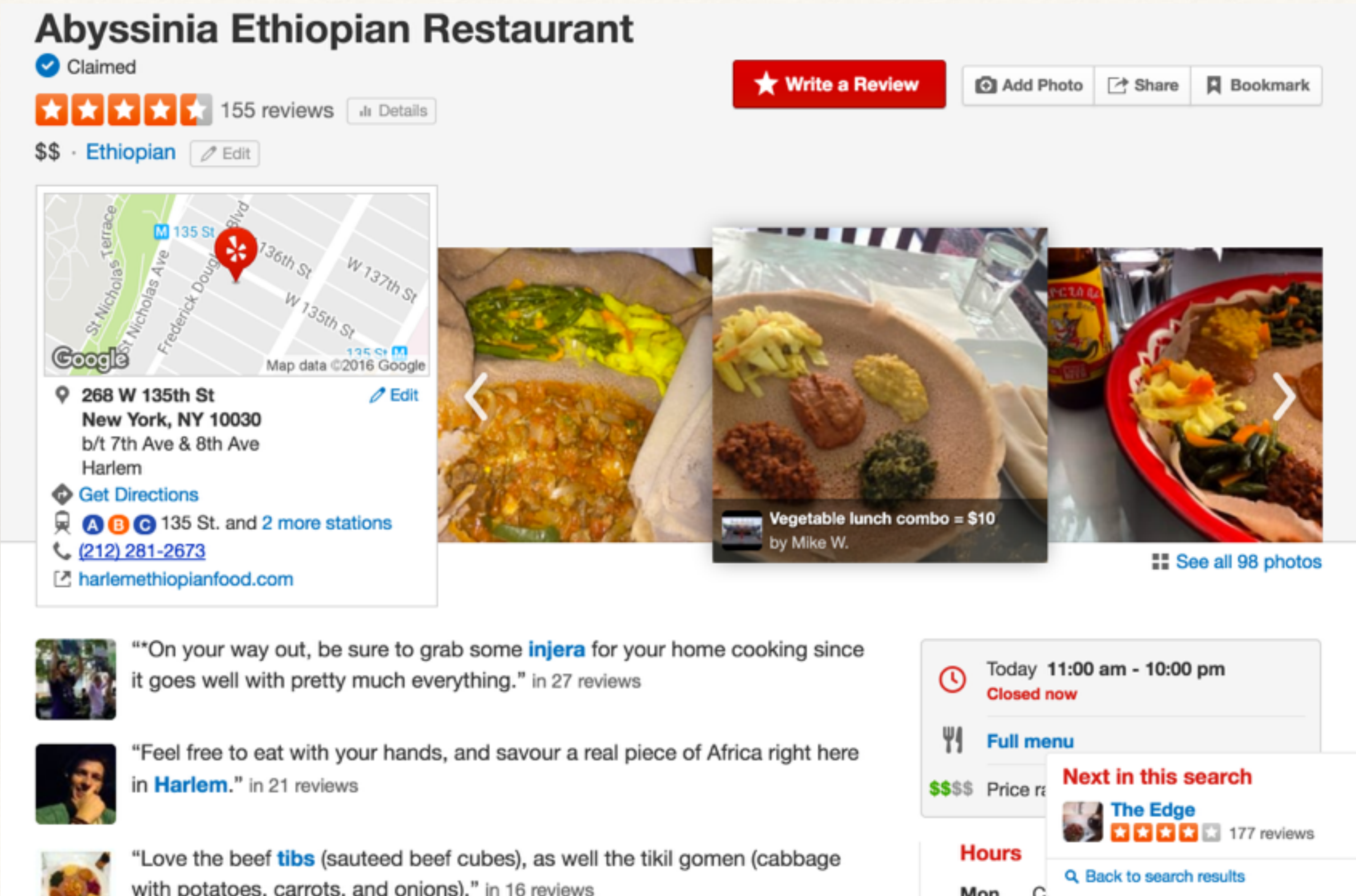
News


Le Monde
PS en plein doute
un projet pour 2007
DVD
2-FOR-1
MOTHER
Wetter
krank?

Web Content

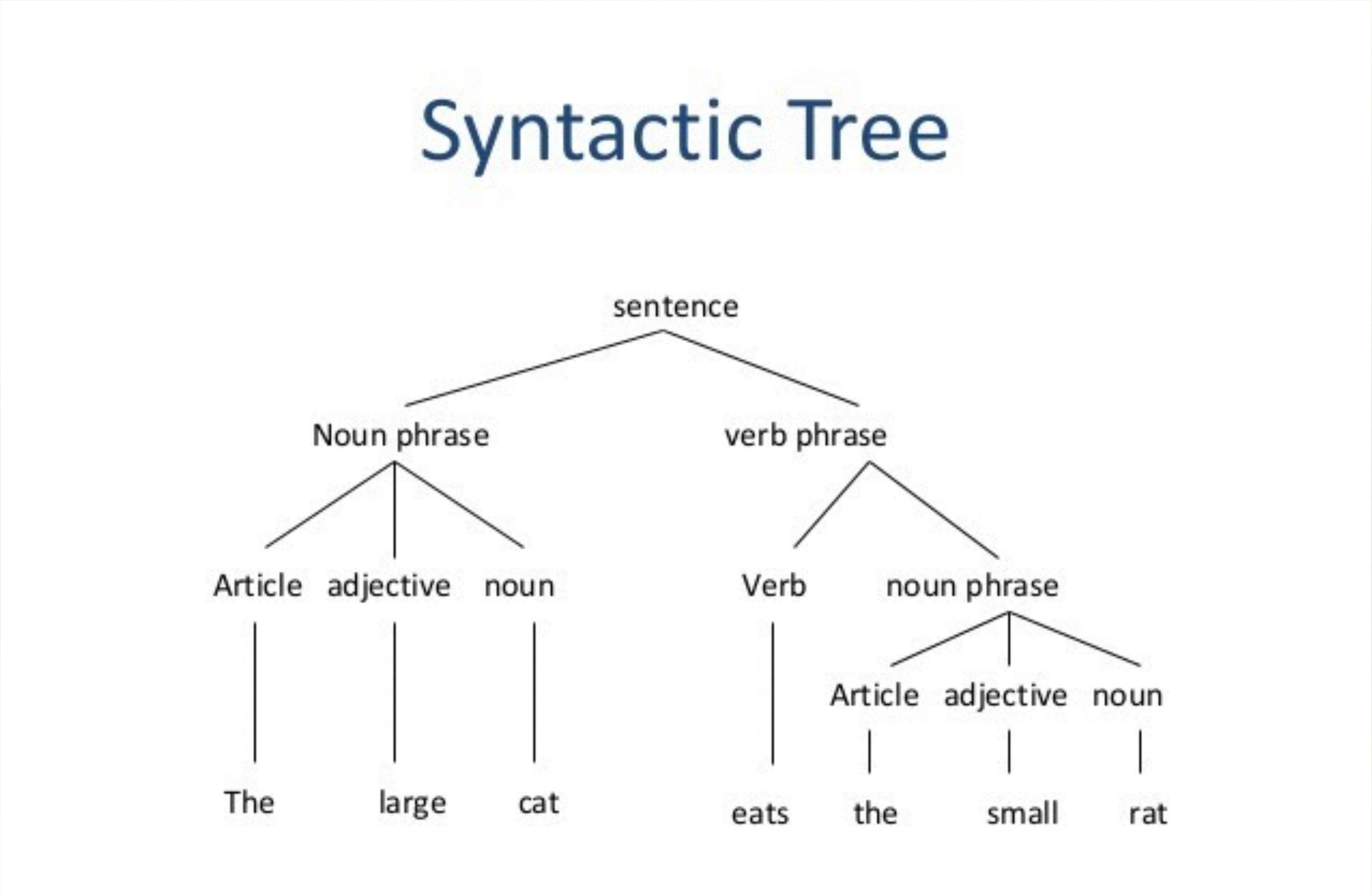

Not logged in Talk Contributions Create account Log in
Main Page Talk
Read View source View history
Search Wikipedia
Wikipedia
The Free Encyclopedia
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store
Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page
Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Welcome to Wikipedia,
the free encyclopedia that anyone can edit.
5,285,920 articles in English
Arts
Biography
Geography
History
Mathematics
Science
Society
Technology
All portals
From today's featured article
Hurricane Kate formed northeast of Puerto Rico on November 15, 1985, as the eleventh named storm of the annual hurricane season. Kate made its first landfall on the northern coast of Cuba at Category 2 intensity, then emerged as a slightly weaker storm during the evening hours of November 19. Heavy rainfall in Cuba caused numerous mudslides and flooding, killing 10 people and leading to severe agricultural damage. Wind gusts also damaged crops, and resulted in widespread power outages and significant building damage; the cost in Cuba totaled $400 million, the most from a hurricane strike on that island in many decades. Once clear of land, Kate intensified to Category 3, and the following day it attained its
In the news
An earthquake with a magnitude of 7.5 to 7.8 strikes north of Christchurch, New Zealand, triggering tsunami warnings, causing widespread damage, and killing at least two people.
Leonard Cohen
Canadian singer, songwriter, and poet Leonard Cohen (pictured) dies at the age of 82.
A tram derails on the Tramlink in Croydon, London, killing seven people and injuring more than fifty others.
Donald Trump wins the United States

Reviews/Comments/Complaints



Communication

Traditional: Natural Language Processing (NLP)



Traditional NLP: Issues

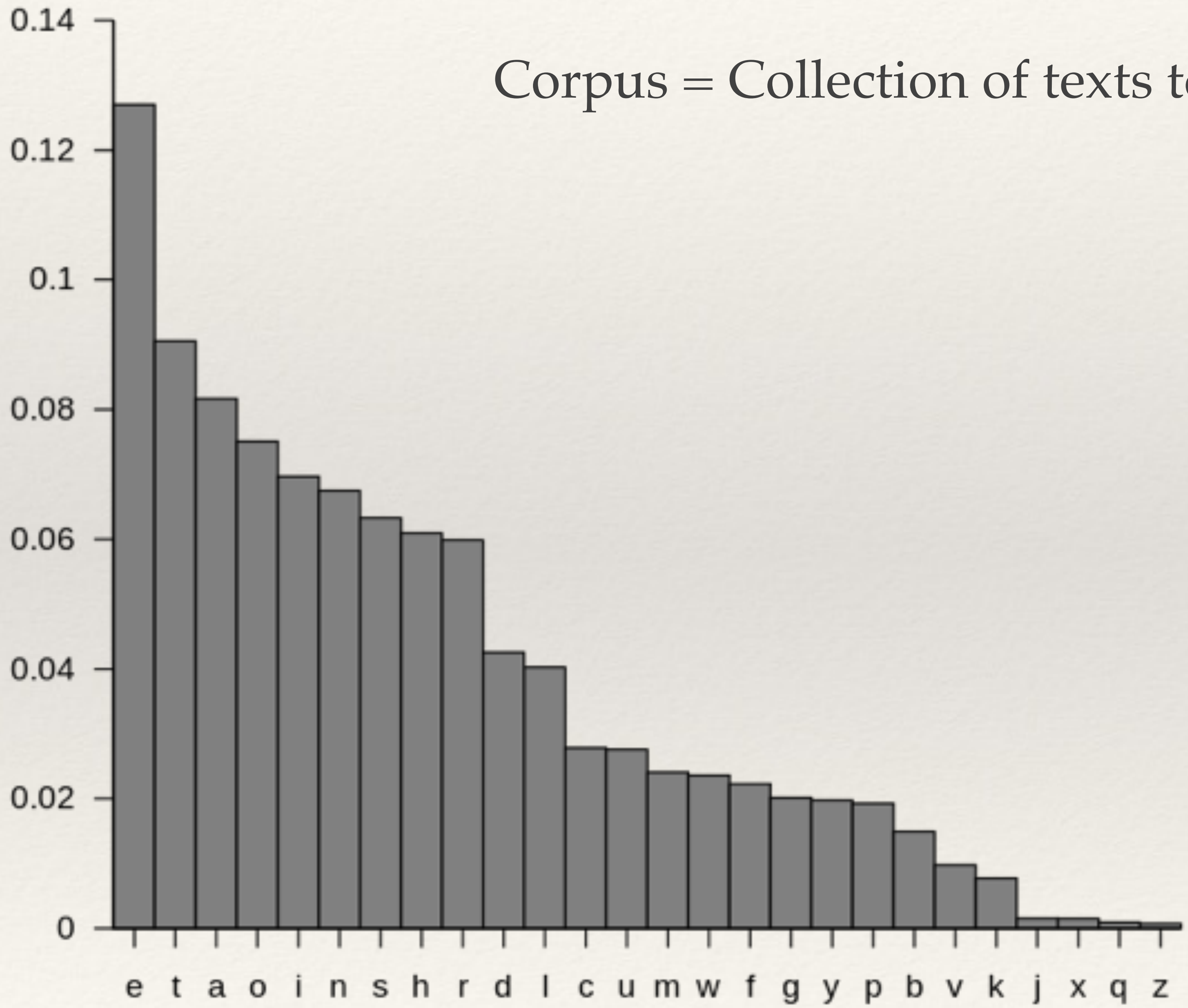
- Text often a mess (bad syntax)
- Computationally expensive
- Big gap between syntax and semantics (meaning)

Look for cheaper approaches

Task: Language Recognition

- Mixed language text, w/mis-spellings
- Looking for dominant language
- Must be very fast

Letter Frequencies in English



Corpus = Collection of texts to train on

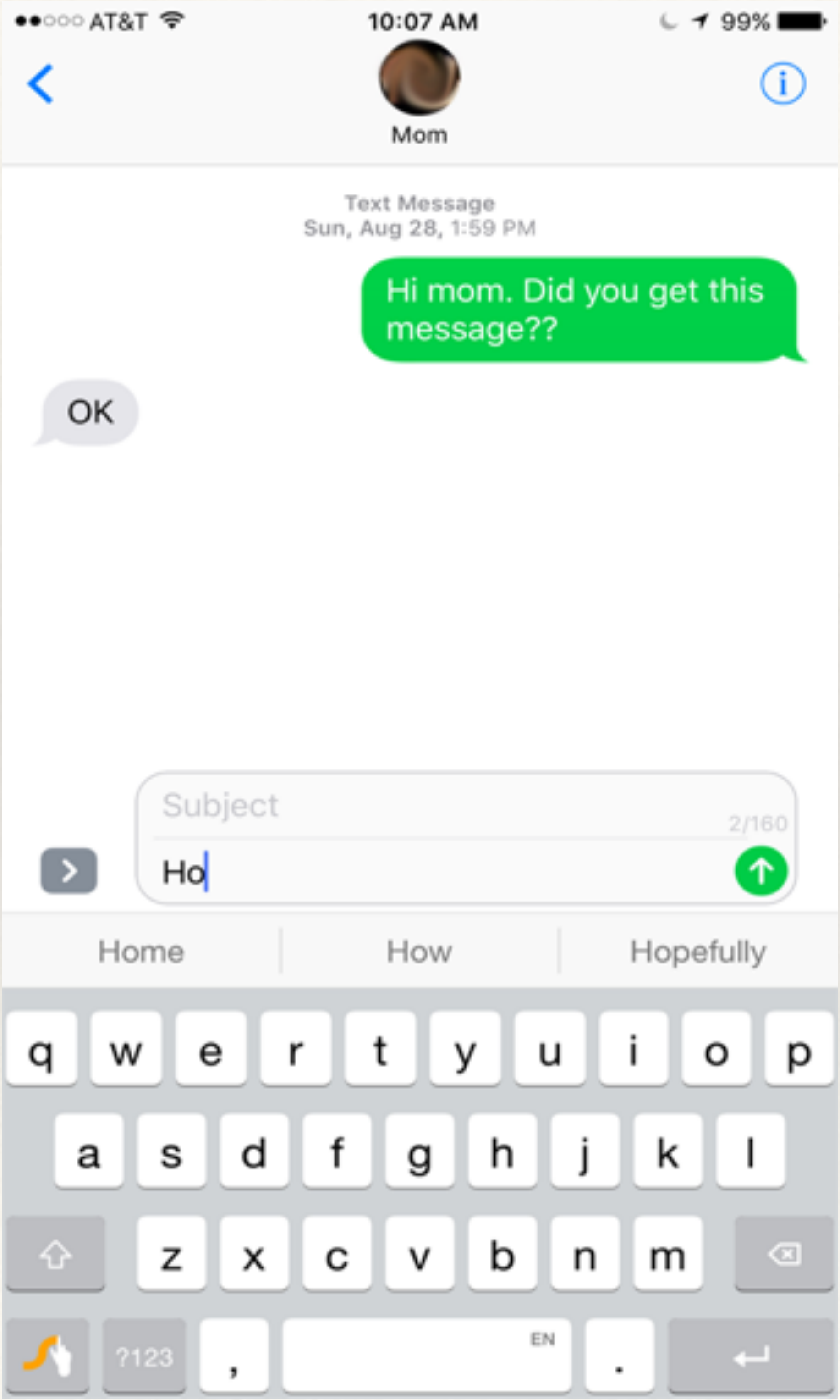
Letter Frequencies in Languages

Letter	English	French [20]	German [21]	Spanish [22]	Portuguese [23]	Esperanto [24]	Italian [25]	Turkish [26]	Swedish [27]
e	12.702%	14.715%	16.396%	12.181%	12.570%	8.995%	11.792%	9.912%	10.149%
t	9.056%	7.244%	6.154%	4.632%	4.336%	5.276%	5.623%	3.314%	7.691%
a	8.167%	7.636%	6.516%	11.525%	14.634%	12.117%	11.745%	12.920%	9.383%
o	7.507%	5.796%	2.594%	8.683%	9.735%	8.779%	9.832%	2.976%	4.482%
i	6.966%	7.529%	6.550%	6.247%	6.186%	10.012%	10.143%	9.600%*	5.817%
n	6.749%	7.095%	9.776%	6.712%	4.446%	7.955%	6.883%	7.987%	8.542%
s	6.327%	7.948%	7.270%	7.977%	6.805%	6.092%	4.981%	3.014%	6.590%
h	6.094%	0.737%	4.577%	0.703%	0.781%	0.384%	0.636%	1.212%	2.090%
r	5.987%	6.693%	7.003%	6.871%	6.530%	5.914%	6.367%	7.722%	8.431%
d	4.253%	3.669%	5.076%	5.010%	4.992%	3.044%	3.736%	5.206%	4.702%
l	4.025%	5.456%	3.437%	4.967%	2.779%	6.104%	6.510%	5.922%	5.275%
c	2.782%	3.260%	2.732%	4.019%	3.882%	0.776%	4.501%	1.463%	1.486%
u	2.758%	6.311%	4.166%	2.927%	3.639%	3.183%	3.011%	3.235%	1.919%
m	2.406%	2.968%	2.534%	3.157%	4.738%	2.994%	2.512%	3.752%	3.471%
w	2.360%	0.074%	1.921%	0.017%	0.037%	0	0.033%	0	0.142%

Simple Algorithm

- Training: Store letter pdf for each language
- Prediction:
  - Compute letter pdf for document
  - Measure distances between doc pdf and each language
  - Language with closest pdf to doc pdf is predicted

Text Completion

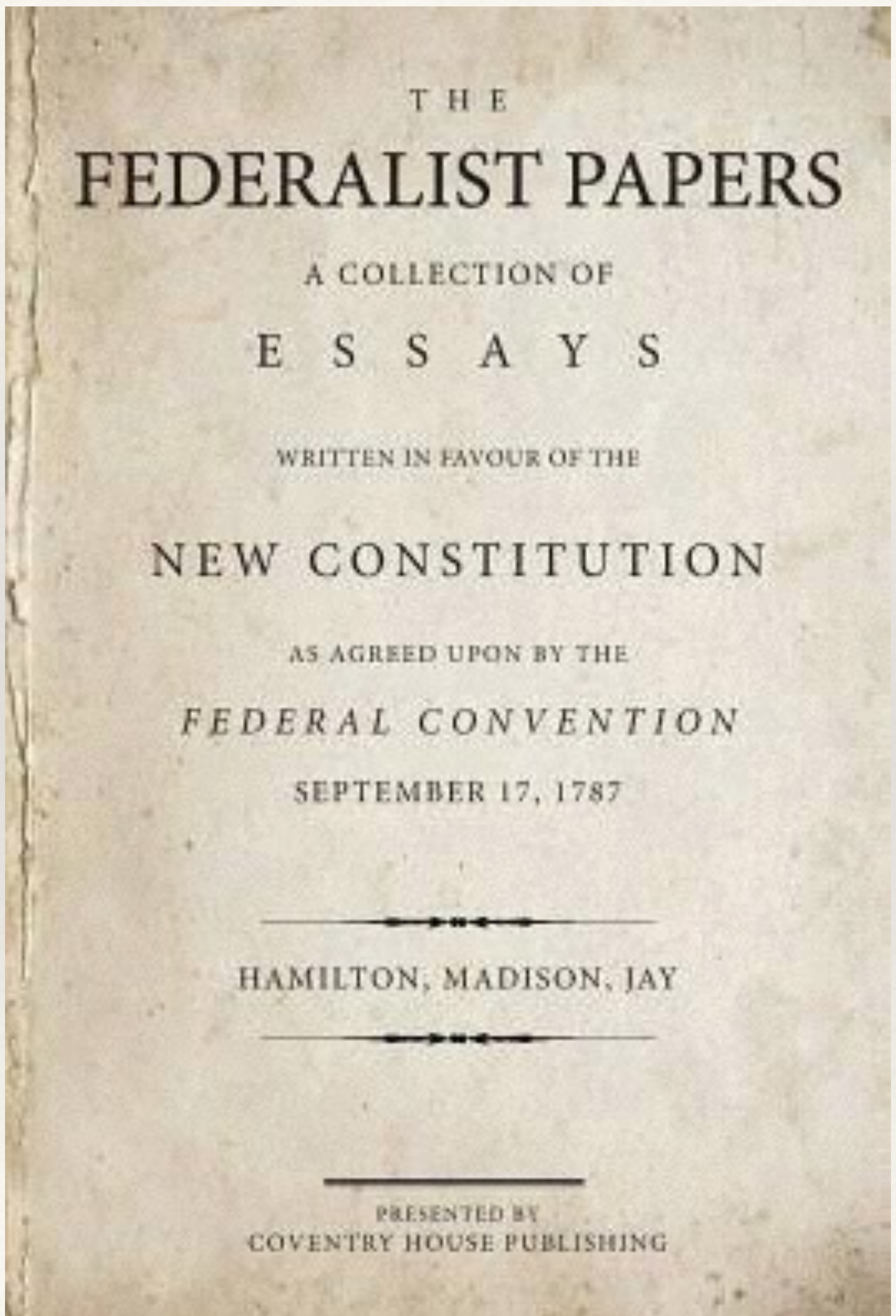


- Count Letter n-gram in Corpus
- Filter to match prefix
- Take top 3

For spelling check:

Use hamming distance rather than exact match

How Can we Match Documents?



Hamilton

Madison

John Jay

Authorship Problem

Statistics to figure out Madison vs Hamilton essays

Study words

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION

Number 302 JUNE, 1963 *Volume 58*

INFERENCE IN AN AUTHORSHIP PROBLEM[1,2]

A comparative study of discrimination methods applied to the authorship of the disputed *Federalist* papers

FREDERICK MOSTELLER
Harvard University
and
Center for Advanced Study in the Behavioral Sciences
AND
DAVID L. WALLACE
University of Chicago

This study has four purposes: to provide a comparison of discrimination methods; to explore the problems presented by techniques based strongly on Bayes' theorem when they are used in a data analysis of large scale; to solve the authorship question of *The Federalist* papers; and to propose routine methods for solving other authorship problems.

Word counts are the variables used for discrimination. Since the topic written about heavily influences the rate with which a word is used, care in selection of words is necessary. The filler words of the language such as *an*, *of*, and *upon*, and, more generally, articles, prepositions, and conjunctions provide fairly stable rates, whereas more meaningful words like *war*, *executive*, and *legislature* do not.

After an investigation of the distribution of these counts, the authors execute an analysis employing the usual discriminant function and an analysis based on Bayesian methods. The conclusions about the authorship problem are that Madison rather than Hamilton wrote all 12 of the disputed papers.

The findings about methods are presented in the closing section on conclusions.

This report, summarizing and abbreviating a forthcoming monograph [8], gives some of the results but very little of their empirical and theoretical foundation. It treats two of the four main studies presented in the monograph, and none of the side studies.

[1] This work has been faciliated by grants from The Ford Foundation, the Rockefeller Foundation, and from the National Science Foundation NSF G-13040 and G-10368, contracts with the Office of Naval Research Nonr 1866(37) and 2121(09), and the Laboratory of Social Relations, Harvard University. The work was done in part at the Massachusetts Institute of Technology Computation Center, Cambridge, Massachusetts, and at the Center for Advanced Study in the Behavioral Sciences, Stanford, California. Permission is granted for reproduction in whole or in part for purposes of the United States Government.

[2] Presented at a session of Special Papers Invited by the Presidents of The American Statistical Association, The Biometric Society (ENAR), and The Institute of Mathematical Statistics at the statistical meetings in Minneapolis, Minnesota, September 9, 1962.

Common Words

TABLE 2.5. FUNCTION WORDS AND THEIR CODE NUMBERS FOR THE FEDERALIST STUDY

1 a	8 as	15 do	22 has	29 is	36 no	43 or	50 than	57 this	64 when
2 all	9 at	16 down	23 have	30 it	37 not	44 our	51 that	58 to	65 which
3 also	10 be	17 even	24 her	31 its	38 now	45 shall	52 the	59 up	66 who
4 an	11 been	18 every	25 his	32 may	39 of	46 should	53 their	60 upon	67 will
5 and	12 but	19 for	26 if	33 more	40 on	47 so	54 then	61 was	68 with
6 any	13 by	20 from	27 in	34 must	41 one	48 some	55 there	62 were	69 would
7 are	14 can	21 had	28 into	35 my	42 only	49 such	56 thing	63 what	70 your

Common in both (and in english): “Stop Words”

Distinguishing Words

TABLE 3.1. WEIGHT-RATE ANALYSIS: WORDS, WEIGHTS, AND IMPORTANCES (TIMES 10^4)

	Weight	Importance		Weight	Importance		Weight	Importance
Group 1			*Group 3*			*Group 5*		
upon	1394	3847	as	−0140	0339	innovation	−1681	0336
			at	0247	0318	language	−1448	0304
Group 2			by	−0146	0542	vigor	2174	0543
although	−1754	0351	of	0037	0281	voice	−2159	0410
commonly	1333	0267	on	−0271	0796			
consequently	−1311	0459	there	0463	0972	*Group 6*		
considerable	0784	0251				destruction	1709	0342
enough	0683	0403	*Group 4*					
while	2708	0704	would	0085	0428			
whilst	−2206	0993						

Hamilton uses “upon” much more than Madison

Bag of Words

```
(1) John likes to watch movies. Mary likes movies too.
```

```
(2) John also likes to watch football games.
```

```
[
    "John",
    "likes",
    "to",
    "watch",
    "movies",
    "also",
    "football",
    "games",
    "Mary",
    "too"
]
```

```
(1) [1, 2, 1, 1, 2, 0, 0, 0, 1, 1]
```

```
(2) [1, 1, 1, 1, 0, 1, 1, 1, 0, 0]
```

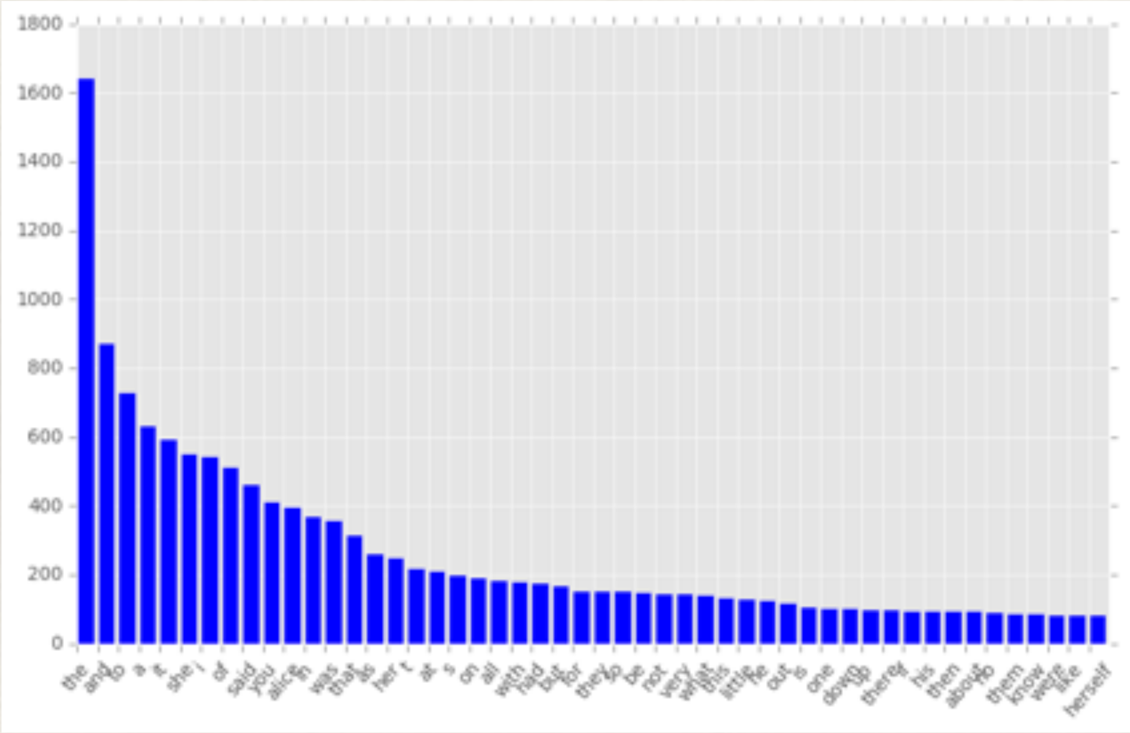
Word Counts

Feature Processing

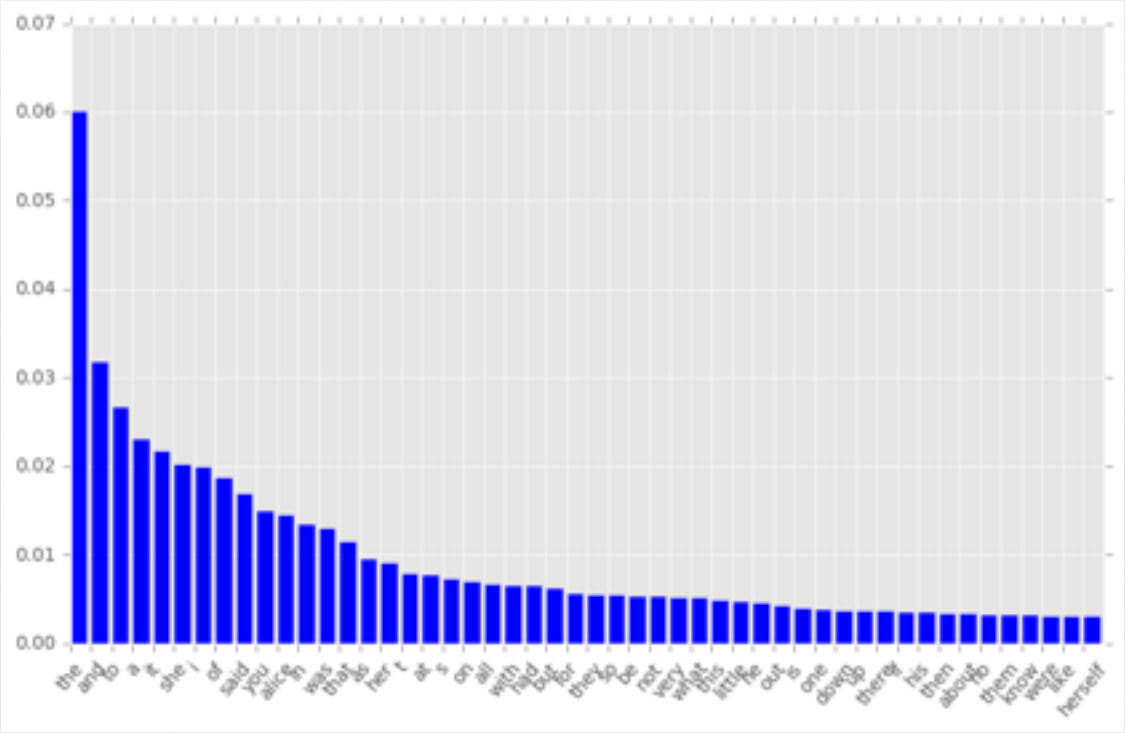
Remove Stop Words

Normalize

Counts



Relative Frequencies



Term Frequency, Inverse Document Frequency

tf = Term Frequency

Variants of TF weight

weighting scheme	TF weight
binary	$0, 1$
raw frequency	$f_{t,d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K)\frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

pdf = Inverse Document Frequency

Variants of IDF weight

weighting scheme	IDF weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t}$
inverse document frequency smooth	$\log(1 + \frac{N}{n_t})$
inverse document frequency max	$\log\left(1 + \frac{\max_{\{t' \in d\}} n_{t'}}{n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

tf-idf

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

tf-idf formulas

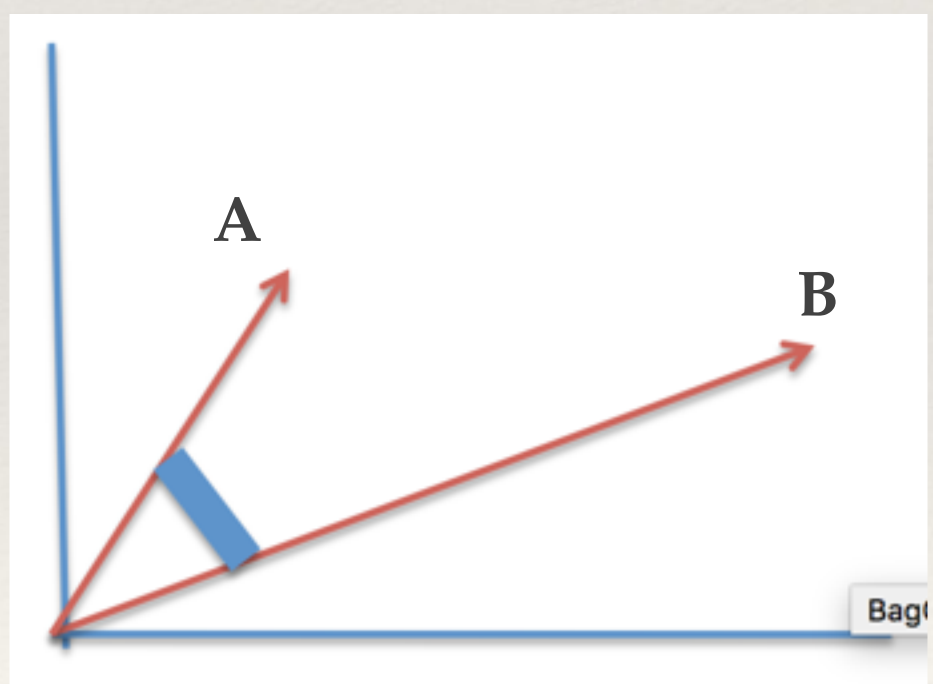
Recommended TF-IDF weighting schemes

weighting scheme	document term weight	query term weight
1	$f_{t,d} \cdot \log \frac{N}{n_t}$	$\left(0.5 + 0.5 \frac{f_{t,q}}{\max_t f_{t,q}}\right) \cdot \log \frac{N}{n_t}$
2	$1 + \log f_{t,d}$	$\log(1 + \frac{N}{n_t})$
3	$(1 + \log f_{t,d}) \cdot \log \frac{N}{n_t}$	$(1 + \log f_{t,q}) \cdot \log \frac{N}{n_t}$

Vector of tf-idf common “Bag of Words” feature

Metric for Bag of Words

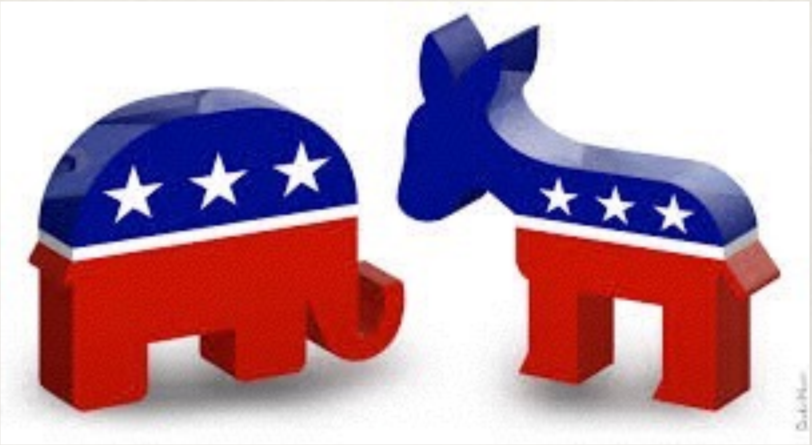
$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$



BoW

- Search … find closest document
- KNN Madison vs. Hamilton
- Use any Classifier you like
- Clustering: Group documents in to similar “themes”
    - Topic Modeling

Topic Modeling



How to Automatically
Extract topics?

Clustering Approach

- Use bag of words (or some other feature) vector
- Cluster
- Name cluster (manually?)

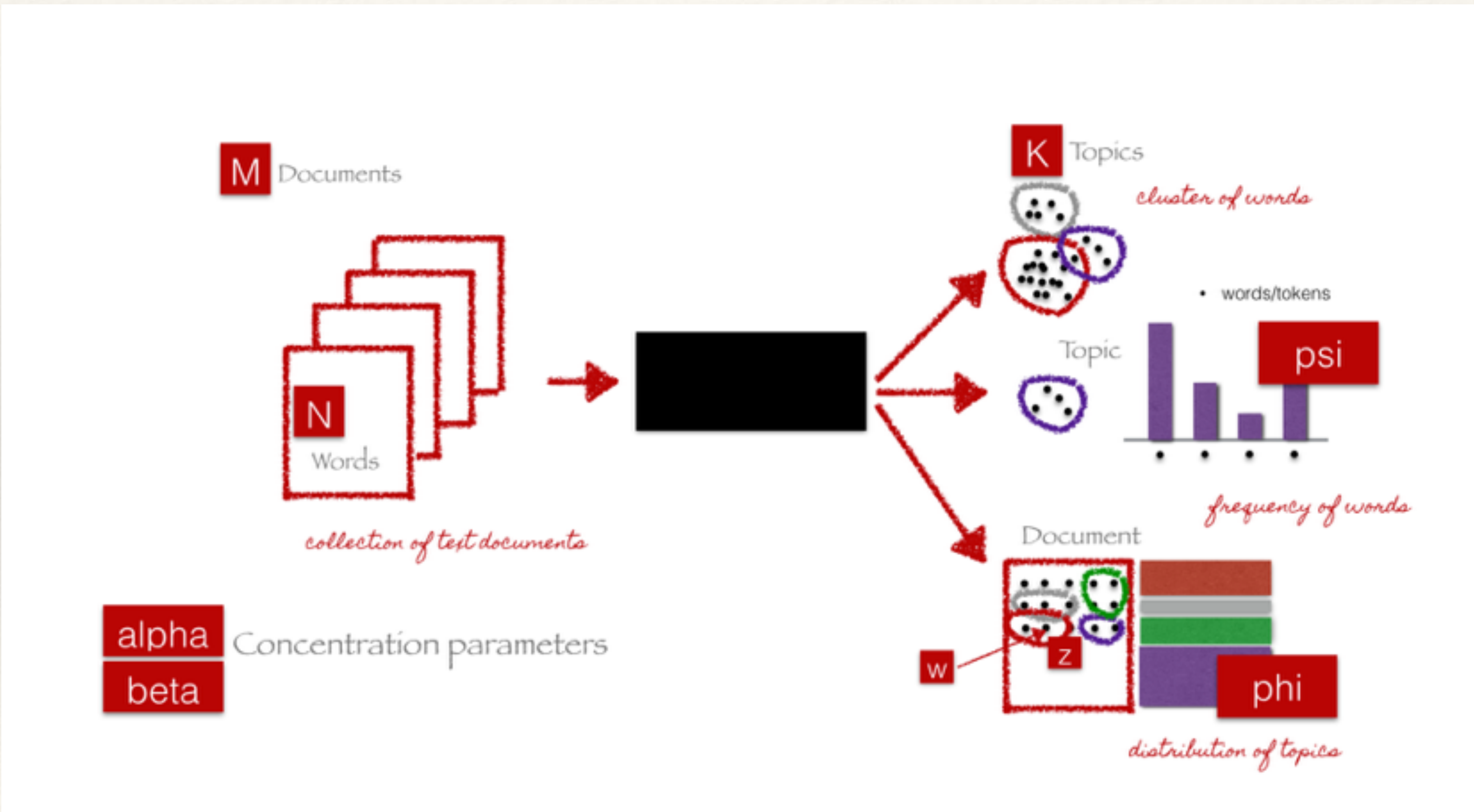
SVD

- Use bag of words (or some other feature) vector
- Decompose document data vector (take some components)
- Manually identify components

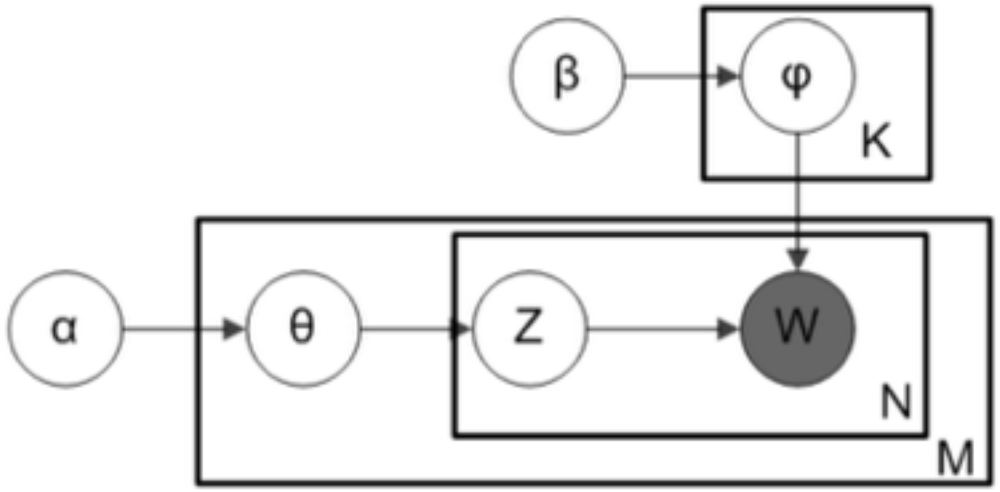
Latent Dirichlet Allocation (LDA)

- One of most popular frameworks for topic modeling
- Like clustering or SVD … discovers topics automatically

Generative Model



LDA in Plate Notation



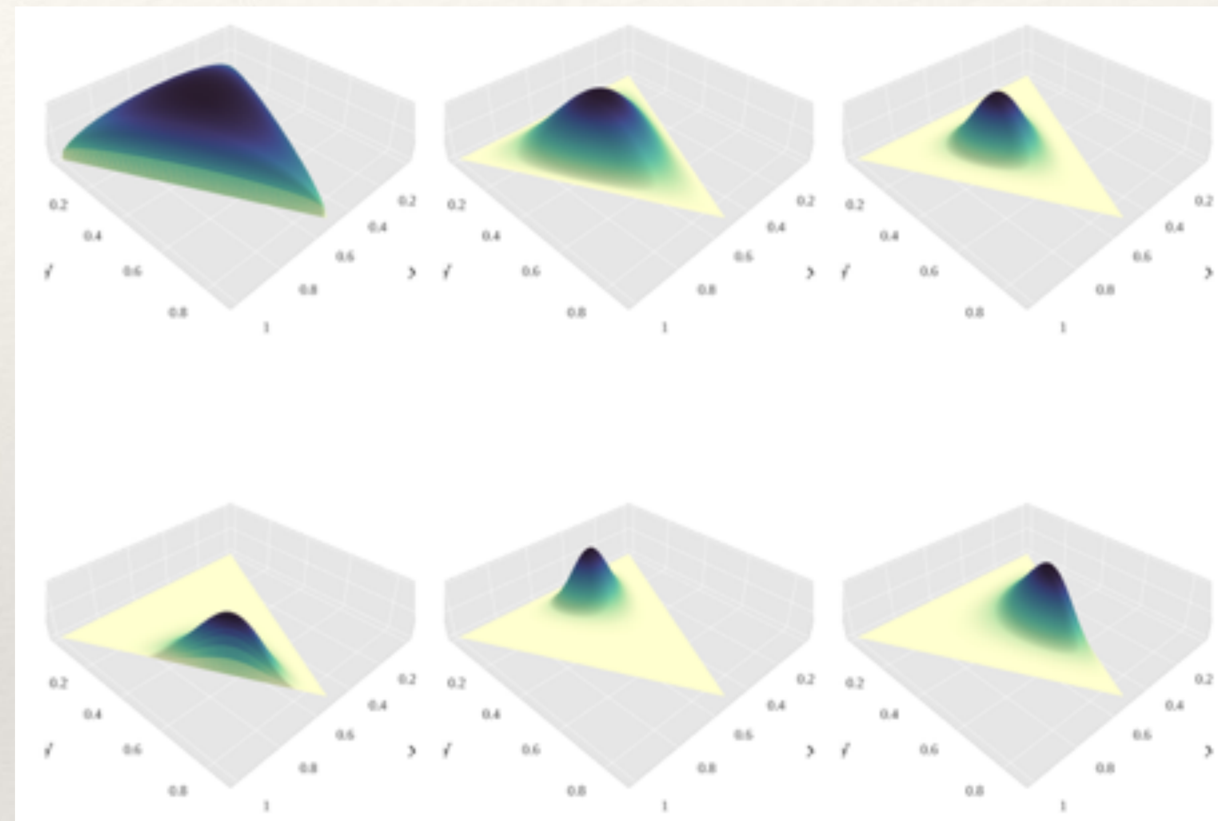
- K is the number of topics
- N is the number of words in the document
- M is the number of documents to analyse
- α is the Dirichlet-prior concentration parameter of the per-document topic distribution
- β is the same parameter of the per-topic word distribution
- ϕ(k) is the word distribution for topic k
- θ(i) is the topic distribution for document i
- z(i,j) is the topic assignment for w(i,j)
- w(i,j) is the j-th word in the i-th document
- ϕ and θ are Dirichlet distributions, z and w are multinomials.

Dirichlet Distribution

$$\frac{1}{\mathrm{B}(\boldsymbol{\alpha})} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

$$\text{where } \mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}$$

$$\text{where } \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)$$



Python Libraries

- Sklearn
- NLTK
- Gensim