



NLP IN PYTHON

WWW.HACKYALE.COM



NLP IN PYTHON

**WEEK 0**  
MANIPULATING TEXT

# **COURSE INTRODUCTION**

# WHO AM I?

- Nick Hathaway, BR '17
- CS major, former English major (\*gasp\*)
- Digital Humanities, Linguistics, Journalism
- Teeth Slam Poets

# NATURAL LANGUAGE PROCESSING

- Programs that:
  - “Understand” human speech or text
  - Mainly statistical
- Natural vs. artificial languages
  - Programming languages are unambiguous
  - Human speech, not so much

# YOU'LL LEARN

## ➤ Text processing

- Tokenization, stemming, built-in text corpora
- Brown, Reuters, WordNet, make your own!

## ➤ Text classification

- Naive Bayes inference, n-gram language models, sentence parsing, etc.
- Classify by topic (Reuters) or by sentiment (imdb)

# YOU'LL LEARN

- Evaluating your models
  - F-scores, MaxEnt, selecting appropriate training and testing sets
- Applications, MOOCs, and books for further study



# SOURCES

- Natural Language Processing with Python, by Steven Bird, Ewan Klein, and Edward Loper. O'Reilly Media, 978-0-596-51649-9.
- MOOC: Natural Language Processing with Dan Jurafsky and Christopher Manning (Stanford, Coursera)



# APPLICATIONS

# SPELL CHECKING

Google   

Web Images Shopping Videos News More ▾ Search tools

About 122,000,000 results (0.49 seconds)


Showing results for **i *can't spell***  
Search instead for **i cant spel**

---

**Dyslexia Warning Signs - Reading Remedy**  
[www.reading-remedy.com/dyslexiawarningsigns.html](http://www.reading-remedy.com/dyslexiawarningsigns.html) ▾  
If someone struggles with spelling, is a slow reader who has a difficult time .... spelling list long enough to pass Friday's spelling test, but they **can't spell** those ...

---

**Images for i can't spell** [Report images](#)






















[More images for i can't spell](#)

---

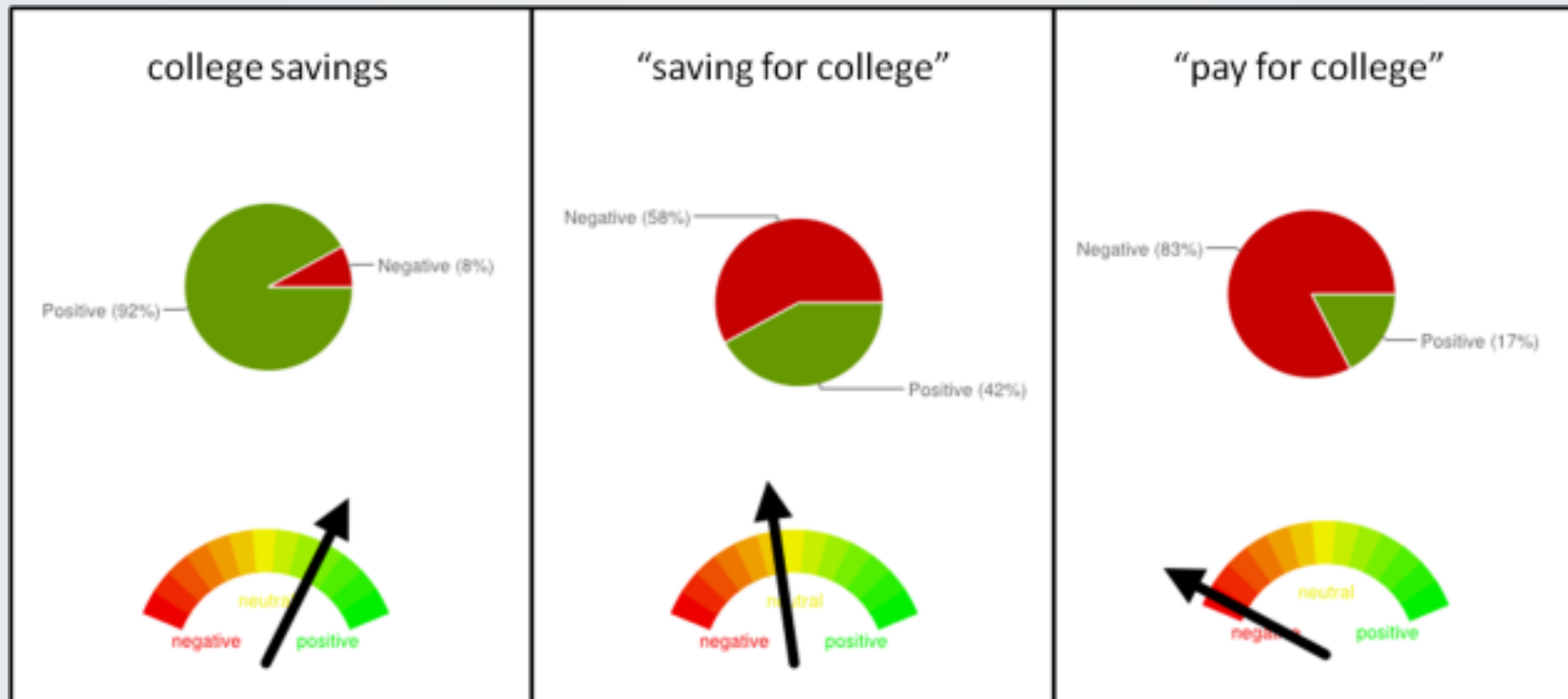
**Dyslexia - Barton Reading & Spelling System**  
<https://www.bartonreading.com/dys.html> ▾

# SPAM DETECTION

From	Subject
 Adelaide Fatimah	a \$12000 watch, we sell at \$200, Quality watches at ...
 antonino rodney	Goodiest c1alis
 Irina Gidget	FDA Approved Medications: \$1.12/pill forViagr...
 tom@messagingtime...	tom@messagingtimes.com, Up to 20% OFF
 Samantha Hickey	Enlarge, Widen and Strengthen
 churchill ravi	MSG #:19846 The world's largest online presc...
 abel yanjun	MSG #:84037 World's lowest prices on largest...
 Maureen Orr	Recapture a bit of your youth again
 nanako258@yahoo.c...	40□Î~È□ã,À□S,à□g'Ì,à-ü,â,³,ê,½,¢•û,Í[-ü,â,...
 Jerald Shook	a xmas gift to your wife is your bigger PE gs ft...
 Blanca Petty	Mit und schaffen Sie das was Frauen wollern!
 Lynne Mcneal	xp oem software
 emerson forrest	from Stella Vargas
 Revolution Jobs	Hundreds of digital careers on Revolution Jobs
 Auto Loan Department	GET APPROVED!
 jacquelyn	hi from jacquelyn
 ParkRoyalCancun	Visit Cancun With A 3 Night Free Stay - No Pur...
 Colon Cleanse Samples	View this LifeChanging Breakthrough
 o05689ok97@tom.com	40□Î~È□ã,À□S,à□g'Ì,à-ü,â,³,ê,½,¢•û,Í[-ü,â,...

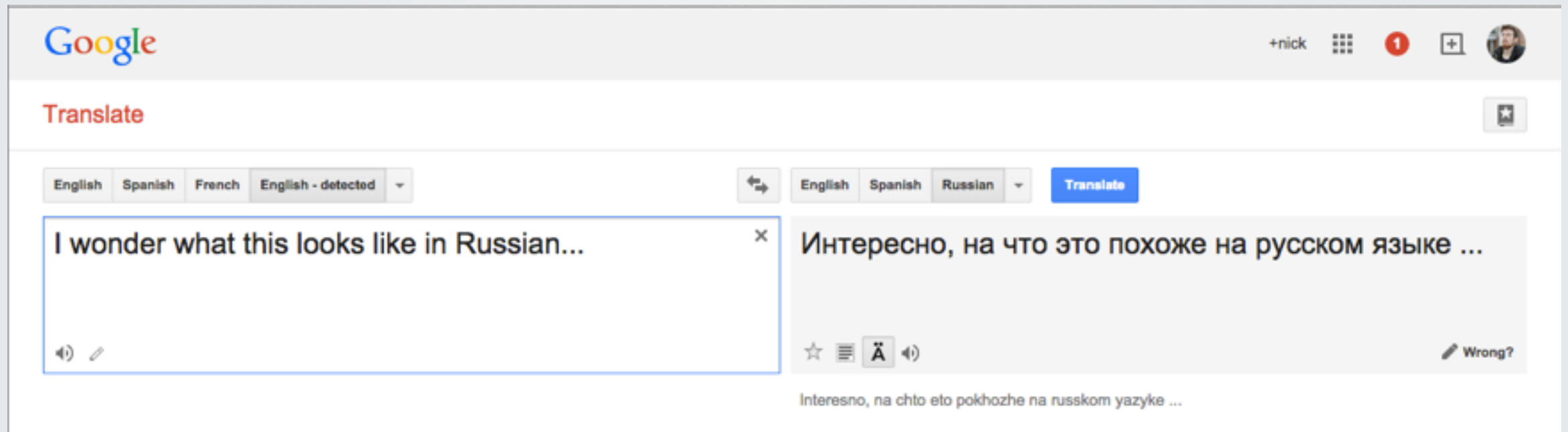
CREDIT: GROUPMAIL

# SENTIMENT ANALYSIS



CREDIT: HUTCH CARPENTER, I'M NOT ACTUALLY A GEEK BLOG  
VIA SENTIMENT140 & SENTIMENT ANALYZER

# MACHINE TRANSLATION



# INFORMATION EXTRACTION



## Ada Lovelace

Countess of Lovelace

Augusta Ada King, Countess of Lovelace, born Augusta Ada Byron and now commonly known as Ada Lovelace, was an English mathematician and writer chiefly known for her work on Charles Babbage's early ...

[Wikipedia](#)

**Born:** December 10, 1815, London, United Kingdom

**Died:** November 27, 1852, Marylebone, United Kingdom

**Full name:** Augusta Ada King

**Parents:** [George Gordon Byron](#), [Anne Isabella Byron](#), [Baroness Byron](#)

**Children:** [Anne Blunt](#), [15th Baroness Wentworth](#), [More](#)

**Siblings:** [Allegra Byron](#)

### People also search for

[View 15+ more](#)



[Charles Babbage](#)



[George Gordon Byron](#)  
Father



[Grace Hopper](#)



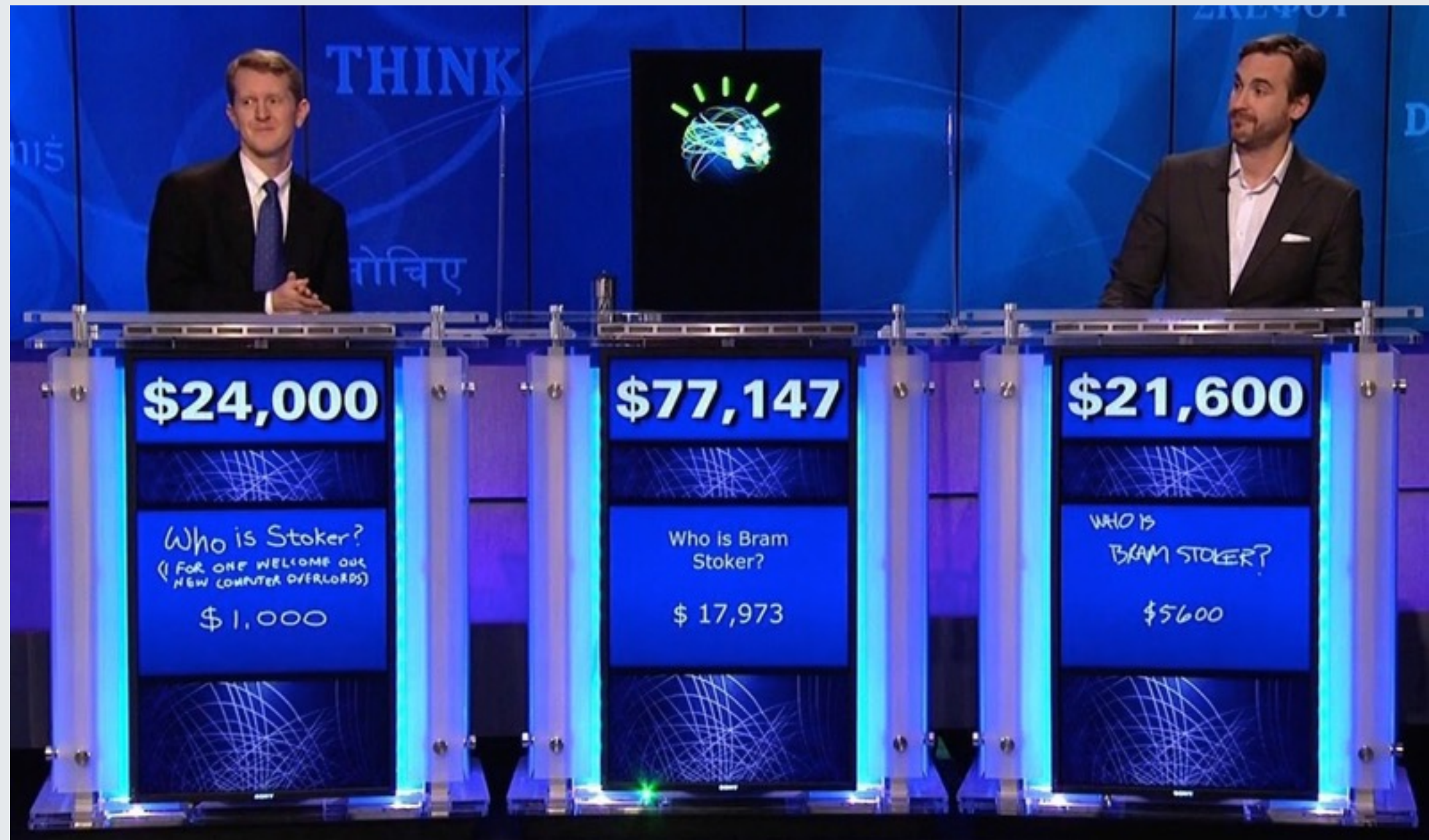
[Alan Turing](#)



[Herman Hollerith](#)



# IBM WATSON



CREDIT: CNME ONLINE

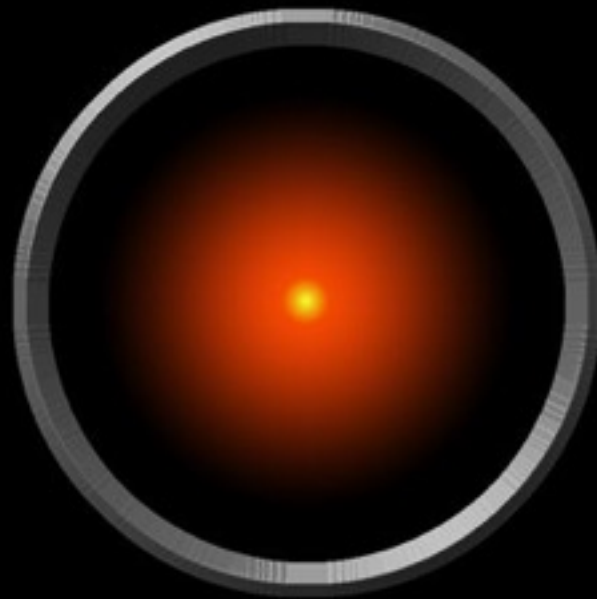
# DIALOG SYSTEMS





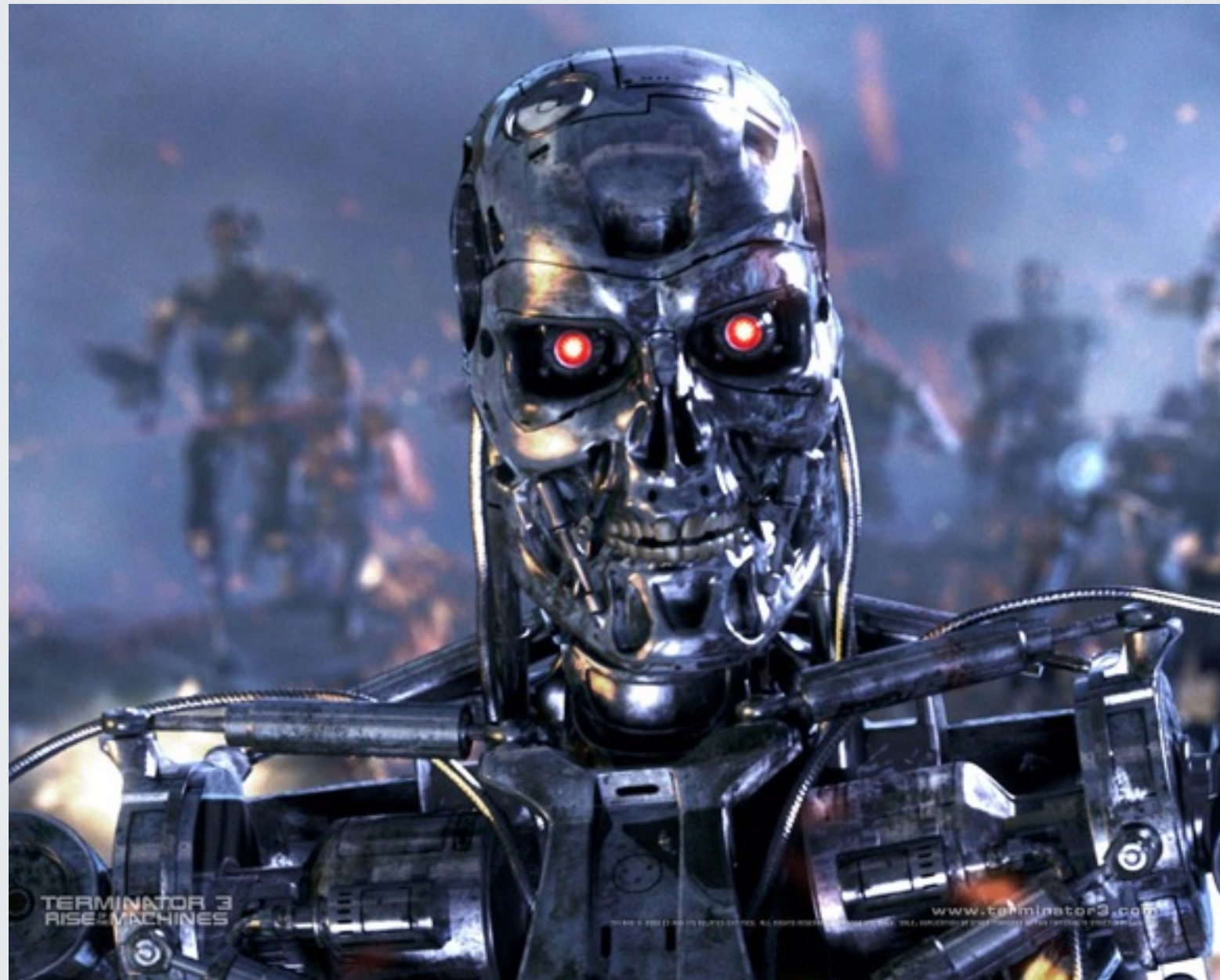
# DIALOG SYSTEMS

I'm sorry Dave,  
I'm afraid I can't do that.



*CREDIT: ABOIET COUNSELING  
& THERAPY GROUP*

# ELIMINATE ALL HUMANS



*CREDIT: TERMINATOR FRANCHISE &  
OUR FUTURE ROBOT OVERLORDS*

# GETTING STARTED

# WHY PYTHON?

- String processing methods
- Excellent statistical libraries
  - Pandas, numpy, etc.
- Natural Language Toolkit
  - Created in 2002
  - > 50 text resources

# ALTERNATIVES

- Java
  - StanfordNLP, mallet, OpenNLP
- Ruby
  - Treat, open-nlp

# STRING MANIPULATION

```
string = "This is a great sentence."
```

```
string = string.split()
```

```
string[:4]
```

```
=> ["This", "is", "a", "great"]
```

```
string[-1:]
```

```
=> "sentence."
```

```
string[::-1]
```

```
=> ["sentence.", "great", "a", "is", "This"]
```

# FILE I/O

```
f = open('sample.txt', 'r')

for line in f:
    line = line.split()
    for word in line:
        if word[-2:] == 'ly':
            print word
```

# SET-UP

NLTK: <http://www.nltk.org/install.html>

NLTK DATA:

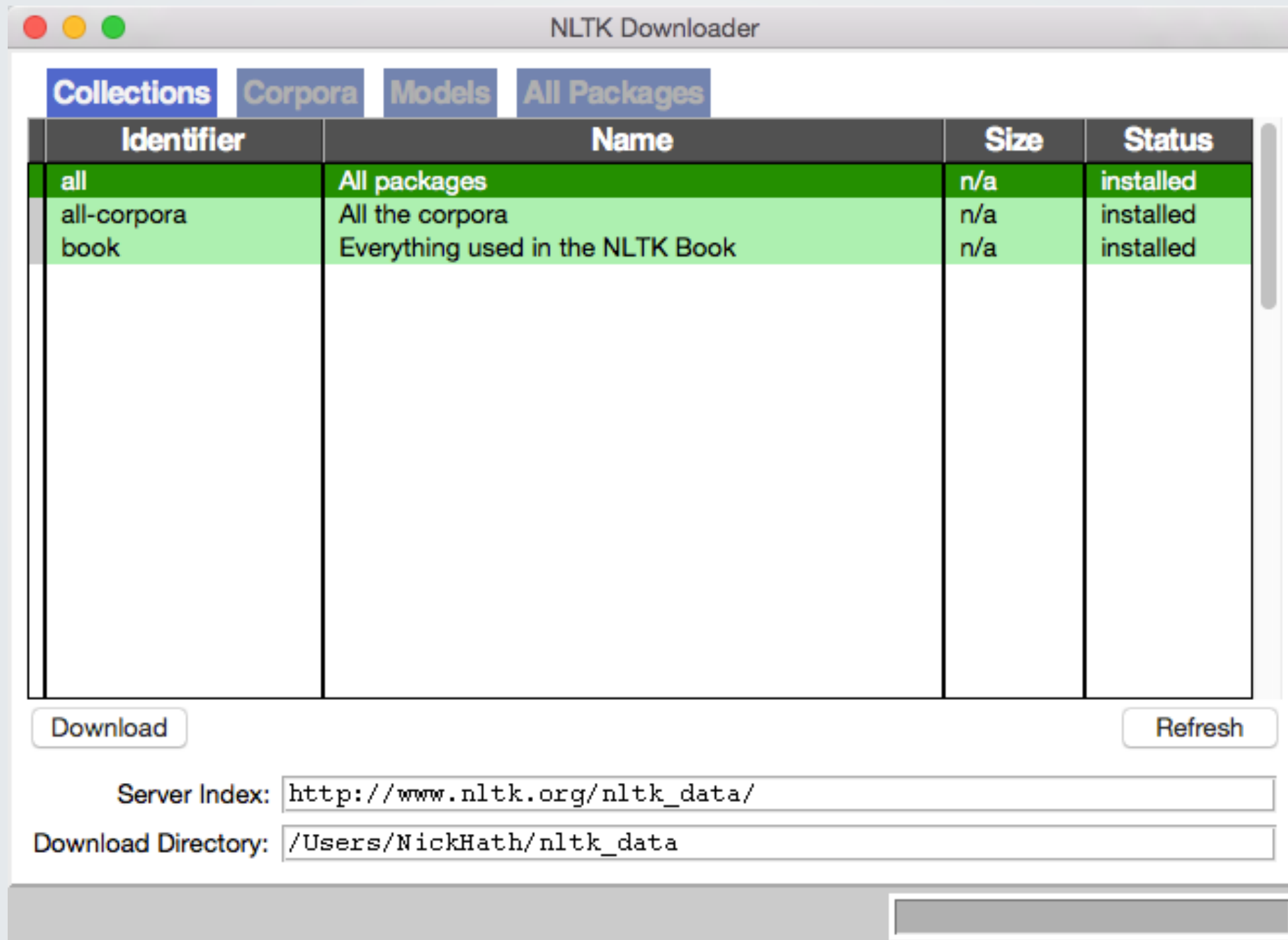
```
import nltk  
nltk.download()
```

Matplotlib:

```
$ pip install matplotlib
```



# NLTK.DOWNLOAD()



The screenshot shows the 'NLTK Downloader' application window. It features a tabbed interface with four tabs: 'Collections', 'Corpora', 'Models', and 'All Packages'. The 'Collections' tab is currently selected. Below the tabs is a table with four columns: 'Identifier', 'Name', 'Size', and 'Status'. The table contains three rows of data, all of which are highlighted in green. The first row shows 'all' as the identifier, 'All packages' as the name, 'n/a' as the size, and 'installed' as the status. The second row shows 'all-corpora' as the identifier, 'All the corpora' as the name, 'n/a' as the size, and 'installed' as the status. The third row shows 'book' as the identifier, 'Everything used in the NLTK Book' as the name, 'n/a' as the size, and 'installed' as the status. Below the table, there are two buttons: 'Download' and 'Refresh'. At the bottom of the window, there are two text input fields. The first field is labeled 'Server Index:' and contains the URL 'http://www.nltk.org/nltk\_data/'. The second field is labeled 'Download Directory:' and contains the path '/Users/NickHath/nltk\_data'.

Identifier	Name	Size	Status
all	All packages	n/a	installed
all-corpora	All the corpora	n/a	installed
book	Everything used in the NLTK Book	n/a	installed

Download Refresh

Server Index:

Download Directory:

# WEEK 0

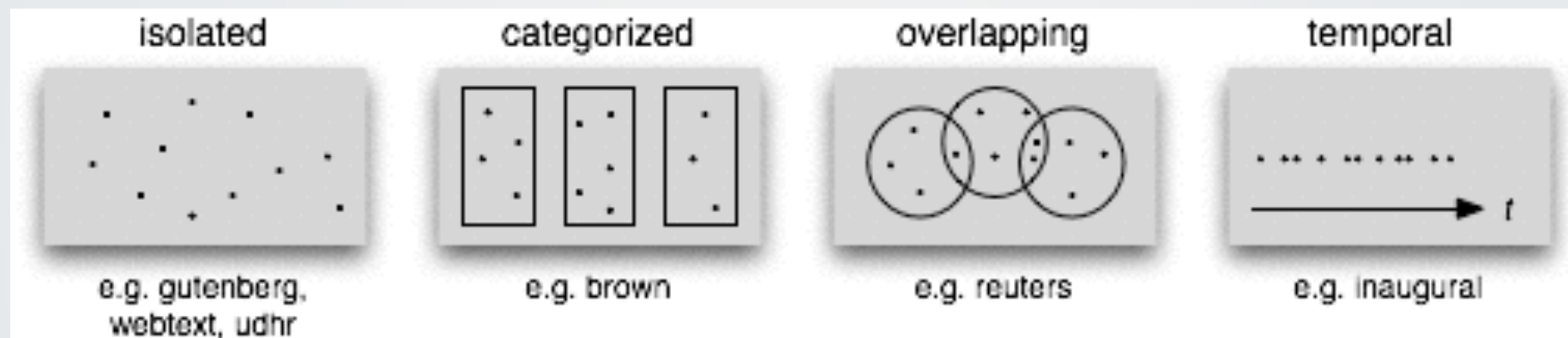
- nltk.corpus
- nltk.tokenize
- nltk.chat

# WHAT IS A CORPUS?

# TEXT CORPORA

- Large body of text
  - Some are general (Brown), others are domain specific (nps\_chat)
  - Some are tagged (Brown), others are raw
  - Some have very specific uses (movie\_reviews - sentiment analysis)

# NLTK BOOK: CORPUS CHART



*CREDIT: NLTK BOOK*

# BUILT-IN CORPORA

- Sentences:

["I", "hate", "cats", "."]

- Paragraphs:

[["I", "hate", "cats", "."], ["Dogs", "are", "worse", "."]]

- Entire file:

List of paragraphs, which are lists of sentences

# MANIPULATING CORPORA

# BROWN CORPUS

- 1960's at Brown University
- > 1,000,000 words
- 15 categories
  - News, mystery, editorial, humor, etc.
- Part-of-speech tagged
- Widely cited



# NAVIGATING CORPORA

```
from nltk.corpus import brown  
  
brown.categories()  
len(brown.categories()) # num categories  
brown.words(categories=['news', 'editorial'])  
brown.sents()  
brown.paras()  
brown.fileids()  
brown.words(fileids=['cr01'])
```

# TOKENS VS. TYPES

## ➤ Tokens

- All of the words (or whatever you choose as your smallest particle of meaning)

I really love this **song**. It's the best **song** I have ever heard.

I: 2

song: 2

heard: 1

# TOKENS VS. TYPES

## ➤ Types

- The vocabulary of your text
- Everything gets counted exactly once

I really love this song. It's the best song I have ever heard.

I: 1

song: 1

heard: 1

# LEXICAL DIVERSITY

```
from nltk.corpus import brown
from __future__ import division
```

```
tokens = brown.words()
types = set(brown.words())
```

```
# lexical diversity
len(tokens)/len(types)
=> 20.714
```

# BY GENRE?

```
# pick categories to compare from:  
brown.categories()
```

```
# calculate lexical diversity of news articles  
news = brown.words(categories='news')  
len(news) / len(set(news))
```

```
# versus fiction  
fiction = brown.words(categories='fiction')  
len(fiction) / len(set(fiction))
```

# GUTENBERG CORPUS

- Curated by the NLTK team
- Just 18 books/poetry collections
- Also look at `nltk.corpus.genesis`

# BY AUTHOR?

```
from nltk.corpus import gutenbergl  
gutenberg.fileids()  
shakespeare = gutenbergl.words(fileids='b1ah')
```

```
len(shakespeare) / len(set(shakespeare))
```

# # versus

```
len(austen) / len(set(austen))
```

# WHAT ARE WE MISSING?

- What does the Brown corpus mean by “fiction”?
  - Is this representative of the question we want to ask?
  - Is 68,488 words enough?
- Can the different categories be compared?
  - Similarly sized data?
  - Same genres of writing?



# INVESTIGATE YOUR CORPORA!

```
from nltk.corpus import brown
```

```
brown.fileids(categories='fiction')
```

```
brown.abspath('ck01')
```

```
# poke around the corpora in ../../nltk_data/corpora
```

```
# check out the README, category list, actual files
```

# TEXT OBJECTS VS. CORPORA OBJECTS

# TEXT OBJECT VS. CORPUS OBJECT

- You have to convert `some_corpus.words()` to a text object using the `Text` method

Ex:

```
from nltk.text import Text
```

```
text = Text(brown.words())
```

# TEXT OBJECT VS. CORPUS OBJECT

```
# you have to convert a corpus object to a text  
# object to access additional methods
```

```
from nltk.text import Text  
text = Text(brown.words())
```

# TEXT METHODS

```
from nltk.corpus import some_corpus
from nltk.text import Text
text = Text(some_corpus.words())
```

```
text.concordance("word")
text.similar("word")
text.common_contexts(["word1", "word2"])
text.count("word")
text.collocations()
```

# WEBTEXT CORPUS

```
from nltk.corpus import webtext
from nltk.text import Text

dating = Text(webtext.words(fileids='singles.txt'))
dating.collocations()
```

# NPS\_CHAT CORPUS

- 10,567 forum posts by age
  - Out of 500,000 collected by the naval postgraduate school
- Can we identify online sexual predators?
  - Or categorize aspects of written communication that vary by age?

# INVESTIGATING CORPORA

```
from nltk.corpus import nps_chat  
nps_chat.sents()
```



# AHHH, DOESN'T WORK

```
# if you ever get stuck, use these techniques  
# to find appropriate methods for a corpus
```

```
# gives you list of methods  
dir(nps_chat)
```

```
# gives you documentation  
help(nltk.corpus)  
help(nltk.corpus.nps_chat)
```

# FREQUENCY DISTRIBUTIONS

# FREQUENCY DISTRIBUTIONS

```
from nltk import FreqDist
from nltk.text import Text
import matplotlib
```

```
text = Text(some_corpus.words())
```

```
fdist = FreqDist(text)
```

```
=> dict of {'token1':count1, 'token2':count2,...}
```

```
fdist['the']
```

```
=> returns the # of times 'the' appears in text
```

# FREQUENCY DISTRIBUTIONS

```
# top 50 words
```

```
fdst.most_common(50)
```

```
# frequency plot
```

```
fdist.plot(25, cumulative=[True or False])
```

```
# list of all 1-count tokens
```

```
fdist.hapaxes()
```

# FREQUENCY DISTRIBUTIONS

# most frequent token

`fdist.max()`

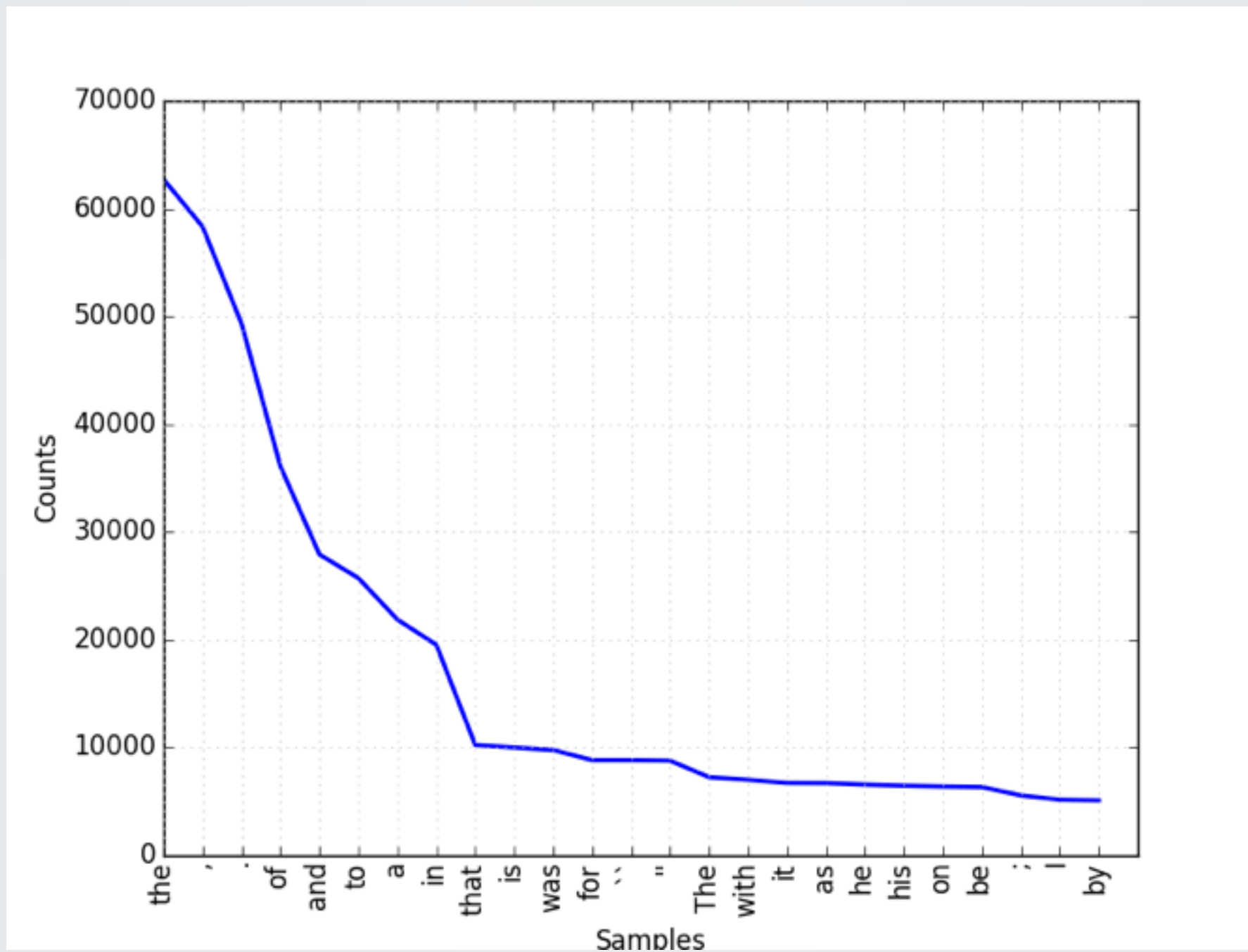
# decimal frequency of a given token

`fdist.freq('token')`

# total # of samples

`fdist.N()`

# PROBLEM



*FREQ. DIST. OF BROWN CORPUS*

# PROBLEM

- We're mostly counting words that don't tell us anything about the text
  - "The" is most frequent
  - Followed by “
- We need stopwords
  - Commonly used words that you can remove from a corpus before processing

# STOPWORDS

```
from nltk.corpus import stopwords
```

```
sw = stopwords.words('english')
```

```
old_brown = brown.words()
```

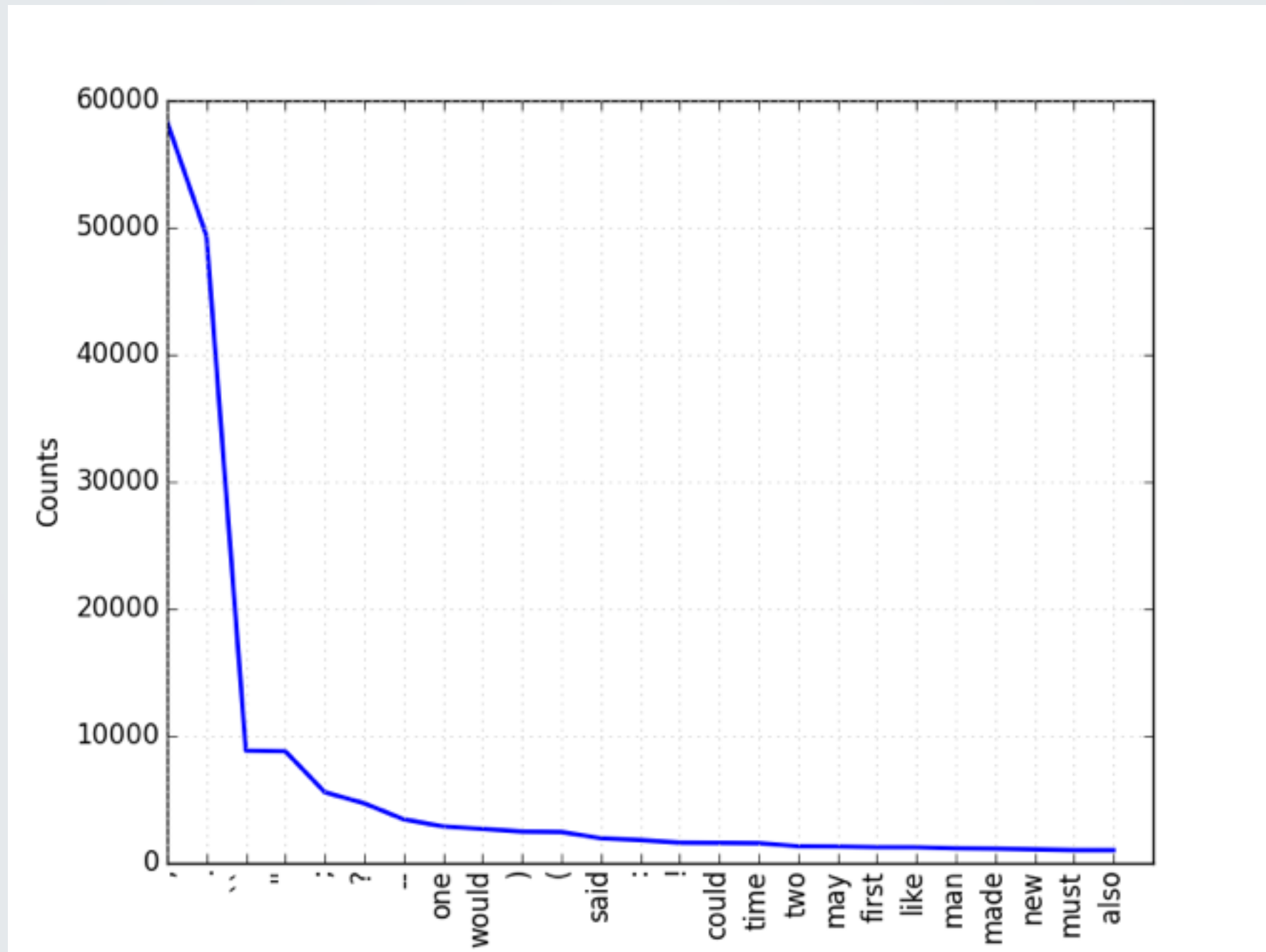
```
new = [w for w in old_brown if w.lower() not in sw]
```

```
# better, but we're still counting punctuation
```

```
# sample code - brown.py
```



# PROBLEM



*FREQ. DIST. OF BROWN CORPUS*

# STOPWORDS & PUNCTUATION

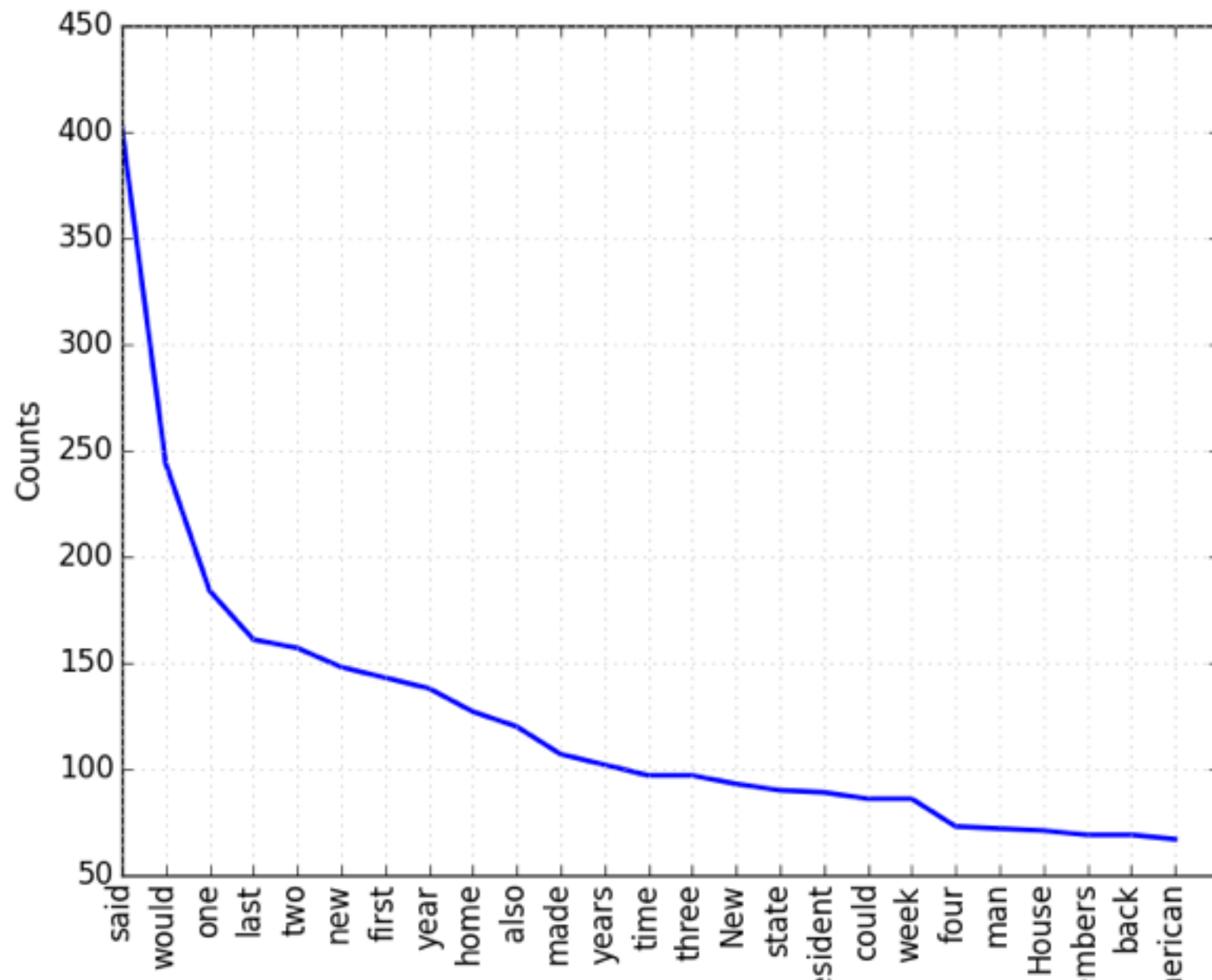
```
from nltk.corpus import stopwords
```

```
sw = stopwords.words('english')
```

```
old_brown = brown.words()
```

```
new = [w for w in old_brown if w.lower() not in sw  
                                and w.isalnum()]
```

# STILL PROBLEMS



FREQ. DIST. OF BROWN'S NEWS CORPUS

# HOMework

- Play around with new corpora
  - What are the frequent words of diff. genres?
  - Look at `webtext.fileids()` for a diversity of text files
  - Find one in your area of interest
    - `movie_reviews`, `udhr`, `wordnet`, etc.
    - `dir(some_corpus)` => how do the methods differ?
    - `some_corpus.abspath()` => what's actually in the corpus?