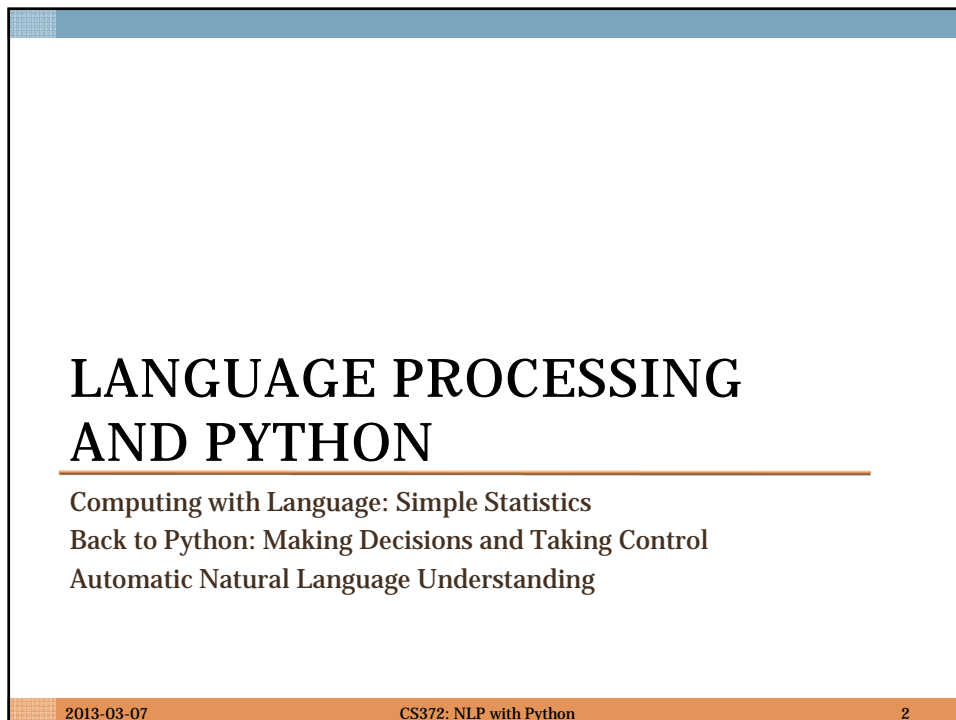


Natural Language Processing
with Python
CS372: Spring, 2013

Lecture 2

Jong C. Park
Department of Computer Science
Korea Advanced Institute of Science and Technology



**LANGUAGE PROCESSING
AND PYTHON**

Computing with Language: Simple Statistics
Back to Python: Making Decisions and Taking Control
Automatic Natural Language Understanding

2013-03-07 CS372: NLP with Python 2

Computing with Language: Simple Statistics

- Introduction
- Frequency Distributions
- Fine-Grained Selection of Words
- Collocations and Bigrams
- Counting Other Things

2013-03-07

CS372: NLP with Python

3

Introduction

- Goal
 - Use automatic methods to find characteristic words and expressions of a text.

2013-03-07

CS372: NLP with Python

4

Frequency Distributions

- Use `FreqDist` to find the 50 most frequent words of *Moby Dick*.

```
>>> fdist1 = FreqDist(text1)
>>> len(fdist1)
19317
>>> fdist1
<FreqDist with 260819 outcomes>
>>> vocabulary1 = fdist1.keys()
>>> vocabulary1[:50]
['I', 'the', '.', 'of', 'and', 'a', 'to', ';', 'in', 'that', '"', '-', 'his', 'i', 't', 'I', 's', 'is', 'he', 'with', 'was', 'as', '"', 'all', 'for', 'this', '!', 'at', 'by', 'but', 'not', '---', 'him', 'from', 'be', 'on', 'so', 'whale', 'one', 'you', 'had', 'have', 'there', 'But', 'or', 'were', 'now', 'which', '?', 'me', 'like']
>>> fdist1['whale']
906
>>>
```

2013-03-07

CS372: NLP with Python

5

Frequency Distributions

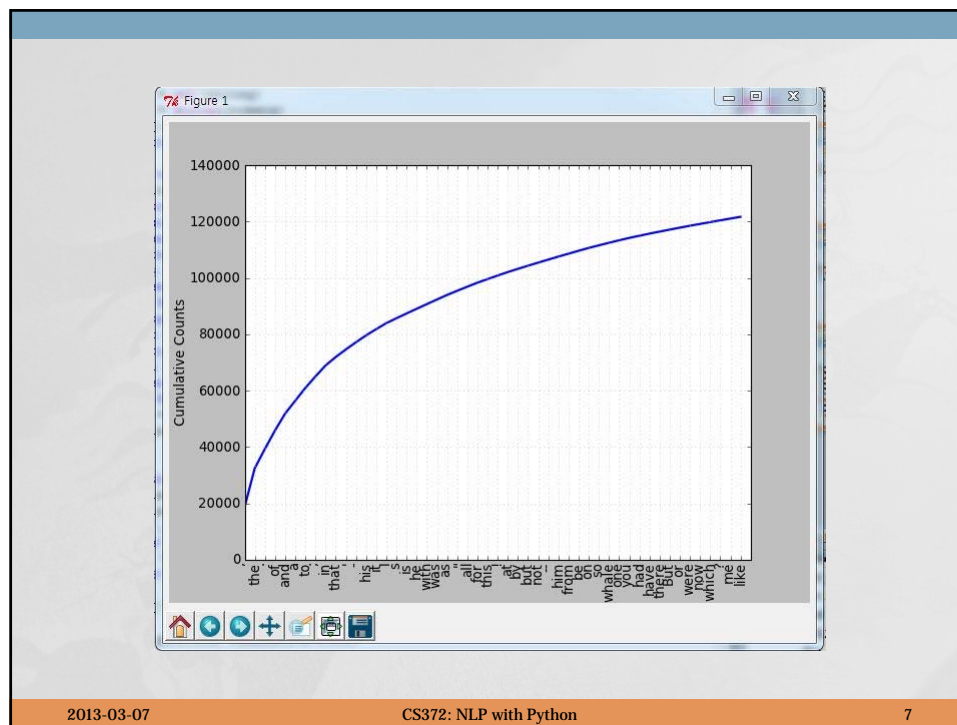
- Generate a cumulative frequency plot for the words produced in the last example.

```
>>> fdist1.plot(50, cumulative=True)
```

2013-03-07

CS372: NLP with Python

6



2013-03-07

CS372: NLP with Python

7

Frequency Distributions

- Use `fdist1.hapaxes()` to view the words that occur only once.

```
>>> list = fdist1.hapaxes()
>>> len(list)
9002
>>>
```

```
, 'Whiteness', 'Whitsuntide', 'Whole', 'Whom', 'Whose', 'Whosoever', 'Wife', 'Willingby', 'Wind', 'Winding', 'Winds', 'Winnebago', 'Wisdom', 'Wise', 'Wish', 'Wit', 'Witt', 'Won', 'Wonderfullest', 'Wondrous', 'Woo', 'Wooden', 'Woods', 'Words', 'Work', 'Works', 'Wrapped', 'Wrapping', 'Wretched', 'Wrinkled', 'X', 'XVI', 'XXXIX', 'YARD', 'YORK', 'YOUR', 'YOURS', 'Yankee', 'Year', 'Yellow', 'Yoke', 'Yon', 'Yorkshire', 'Zealanders', 'Zeuglodon', 'Zogranda', 'Zoned', 'Zoology', 'Zoroaster', ']', 'abandonedly', 'abatement', 'abashed', 'abate', 'abatement', 'abbreviate', 'abbreviation', 'abeam', 'abhorrence', 'abhorrent', 'abhorring', 'abided', 'ability', 'abjectly', 'abominate', 'abominated', 'abomination', 'aboriginally', 'aboriginalness', 'abortion', 'abortions', 'abounded', 'aboundingly', 'absorbingly', 'abstained', 'abstemious', 'abstinence', 'abstract', 'abstracted', 'abstraction', 'absurdly', 'accelerated', 'accept', 'accessible', 'accessory', 'accidental', 'accommodated', 'accommodation', 'accompanies', 'accomplishing', 'accomplishment', 'accountable', 'accountants', 'accounting', 'accumulate', 'accumulating', 'accurate', 'acerbities', 'ached', 'achieve', 'acknowledges', 'acknow
```

2013-03-07

CS372: NLP with Python

8

Fine-Grained Selection of Words

- Find the words from the vocabulary of the text that are more than 15 characters long.

```
>>> V = set(text1)
>>> long_words = [w for w in V if len(w) > 15]
>>> sorted(long_words)
['CIRCUMNAVIGATION', 'Physiognomically', 'apprehensiveness', 'cannibalistically',
 'characteristically', 'circumnavigating', 'circumnavigation', 'circumnavigatio',
 'comprehensiveness', 'hermaphroditical', 'indiscriminately', 'indispensable',
 'irresistibleness', 'physiognomically', 'preternaturalness', 'responsibil',
 'ities', 'simultaneousness', 'subterraneousness', 'supernaturalness', 'superstiti',
 'ousness', 'uncomfortableness', 'uncompromisedness', 'undiscriminating', 'uninter',
 'penetratingly']
>>>
```

2013-03-07

CS372: NLP with Python

9

Fine-Grained Selection of Words

- Find *frequently occurring* long words.

```
>>> fdist5 = FreqDist(text5)
>>> sorted([w for w in set(text5) if len(w) > 7 and fdist5[w] > 7])
['#14-19teens', '#talkoity_adults', '((((((((((((', '.....', 'Question', 'actua',
 'lly', 'anything', 'computer', 'cute.-ass', 'everyone', 'football', 'innocent', 'l',
 'listening', 'remember', 'seriously', 'something', 'together', 'tomorrow', 'watch',
 'ing']
>>>
```

2013-03-07

CS372: NLP with Python

10

Collocations and Bigrams

- Extract from a text a list of word pairs, also known as bigrams.

```
>>> bigrams(['more', 'is', 'said', 'than', 'done'])
```

- Find bigrams that occur more often than we would expect based on the frequency of individual words.

2013-03-07

CS372: NLP with Python

11

Collocations and Bigrams

```
>>> text4.collocations()
Building collocations list
United States; fellow citizens; four years; years ago; Federal
Government; General Government; American people; Vice President; Old
World; Almighty God; Fellow citizens; Chief Magistrate; Chief Justice;
God bless; every citizen; Indian tribes; public debt; one another;
foreign nations; political parties
>>> text8.collocations()
Building collocations list
would like; medium build; social drinker; quiet nights; non smoker;
long term; age open; Would like; easy going; financially secure; fun
times; similar interests; Age open; weekends away; poss rship; well
presented; never married; single mum; permanent relationship; slim
build
>>>
```

2013-03-07

CS372: NLP with Python

12

Counting Other Things

```
>>> list = [len(w) for w in text1]
>>> fdist = FreqDist(list)
>>> fdist
<FreqDist with 260819 outcomes>
>>> fdist.keys()
[3, 1, 4, 2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 20]
>>>

>>> fdist.items()
[(3, 50223), (1, 47933), (4, 42345), (2, 38513), (5, 26597), (6, 17111), (7, 14399), (8, 9966), (9, 6428), (10, 3528), (11, 1873), (12, 1053), (13, 567), (14, 177), (15, 70), (16, 22), (17, 12), (18, 1), (20, 1)]
>>> fdist.max()
3
>>> fdist[3]
50223
>>> fdist.freq(3)
0.19255882431878046
>>>
```

2013-03-07

CS372: NLP with Python

13

Back to Python: Making Decisions and Taking Control

- Conditionals
- Operating on Every Element
- Nested Code Blocks
- Looping with Conditions

2013-03-07

CS372: NLP with Python

14

Conditionals

- Use numerical comparison operators, such as `<`, `<=`, `==`, `!=`, `>`, and `>=`, to select different words from a sentence of news text.

```
>>> sent7
['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'will', 'join', 'the', 'board', 'as', 'a', 'nonexecutive', 'director', 'Nov.', '29', '.']
>>> [w for w in sent7 if len(w) < 4]
['.', '61', 'old', ',', 'the', 'as', 'a', '29', '.']
>>> [w for w in sent7 if len(w) <= 4]
['.', '61', 'old', ',', 'will', 'join', 'the', 'as', 'a', 'Nov.', '29', '.']
>>> [w for w in sent7 if len(w) == 4]
['will', 'join', 'Nov.']
>>> [w for w in sent7 if len(w) != 4]
['Pierre', 'Vinken', ',', '61', 'years', 'old', ',', 'the', 'board', 'as', 'a', 'nonexecutive', 'director', '29', '.']
>>>
```

2013-03-07

CS372: NLP with Python

15

Conditionals

- Test various properties of words using the functions such as `s.startswith(t)`, `s.endswith(t)`, `t in s`, `s.islower()`, `s.isupper()`, `s.isalpha()`, `s.isalnum()`, `s.isdigit()`, `s.istitle()`.

```
>>> sorted([w for w in set(text1) if w.endswith('ableness')])
['comfortableness', 'honourableness', 'immutableness', 'indispensableness', 'indomitableness', 'intolerableness', 'palpableness', 'reasonableness', 'uncomfortableness']
>>> sorted([term for term in set(text4) if 'gnt' in term])
['Sovereignty', 'sovereignties', 'sovereignty']
>>> sorted([item for item in set(text6) if item.istitle()])
['A', 'Aaaaaaaah', 'Aaaaaaah', 'Aaaaah', 'Aaaugh', 'Aaagh', 'Aaah', 'Aaaughh', 'Aaaugh', 'Aaaughh', 'Aagh', 'Aah', 'Aauuggghhh', 'Aauuugh', 'Aauuuugh', 'Aauuues', 'Action', 'Actually', 'African', 'Ages', 'Aggh', 'Agh', 'Ah', 'Ahh', 'Alice', 'All', 'Allo', 'Almighty', 'Alright', 'Am', 'Amen', 'An', 'Anarc', 'And', 'Angnor', 'Anthrax', 'Antioch', 'Anybody', 'Anyway', 'Apples', 'Aram', 'Are', 'Arimathea', 'Armaments', 'Arthur', 'As', 'Ask', 'Assyria', 'At', 'Attila', 'Augh', 'Autumn', 'Auuuuuuuugh', 'Away', 'Ay', 'Ayy', 'B', 'Back', 'Bad']
```

2013-03-07

CS372: NLP with Python

16

Operating on Every Element

- Count items other than words.

```
>>> [len(w) for w in text1]
[1, 4, 4, 2, 6, 8, 4, 1, 9, ...]
>>> [w.upper() for w in text1]
['[', 'MOBY', 'DICK', 'BY', 'HERMAN', 'MELVILLE',
'1851', ...]
>>> len(text1)
>>> len(set(text1))
>>> len(set([word.lower() for word in text1]))
>>> len(set([word.lower() for word in text1 if
word.isalpha()])))
```

2013-03-07

CS372: NLP with Python

17

Nested Code Blocks

```
>>> word = 'cat'
>>> if len(word) < 5:
    print 'word length is less than 5'

word length is less than 5
>>> if len(word) >= 5:
    print 'word length is greater than or equal to 5'

>>> for word in ['Call', 'me', 'Ishmael', '.']:
    print word

Call
me
Ishmael
.
>>>
```

2013-03-07

CS372: NLP with Python

18

Looping with Conditions

- Combine the if and for statements.

```
>>> sent1 = ['Call', 'me', 'Ishmael', '.']
>>> for xyzzy in sent1:
>>>     if xyzzy.endswith('l'):
>>>         print xyzzy

Call
Ishmael
```

```
>>> sent1 = ['Call', 'me', 'Ishmael', '.']
>>> for token in sent1:
>>>     if token.islower():
>>>         print token, 'is a lowercase word'
>>>     elif token.istitle():
>>>         print token, 'is a titlecase word'
>>>     else:
>>>         print token, 'is punctuation'

Call is a titlecase word
me is a lowercase word
Ishmael is a titlecase word
. is punctuation
```

2013-03-07

CS372: NLP with Python

19

Looping with Conditions

- Create a list of *cie* and *cei* words, and loop over each item and print it.

```
>>> tricky = sorted([w for w in set(text2) if 'cie' in w or 'cei' in w])
>>> for word in tricky:
>>>     print word,

ancient ceiling conceit conceited conceive conscience conscientious conscientiou
sly deceitful deceive deceived deceiving deficiencies deficiency deficient delic
acies excellencies fancied insufficiency insufficient legacies perceive perceive
d perceiving prescience prophecies receipt receive received receiving society sp
ecies sufficient sufficiently undeceive undeceiving
>>>
```

2013-03-07

CS372: NLP with Python

20

Automatic Natural Language Understanding

- Word Sense Disambiguation
- Pronoun Resolution
- Generating Language Output
- Machine Translation
- Spoken Dialogue Systems
- Textual Entailment

2013-03-07

CS372: NLP with Python

21

Word Sense Disambiguation

- *“He served a dish.”*
 - *serve*: help with food or drink; hold an office; put ball into play
 - *dish*: plate; course of a meal; communications device
- *“the book by Chesterton”, “the cup by the stove”, “submit by Friday”*

2013-03-07

CS372: NLP with Python

22

Word Sense Disambiguation

- Interpret the meaning of *by* with the meaning of the italicized word:
 - The lost children were found by the *searchers*. (agentive)
 - The lost children were found by the *mountain*. (locative)
 - The lost children were found by the *afternoon*. (temporal)

2013-03-07

CS372: NLP with Python

23

Pronoun Resolution

- Determine what was sold, caught, and found.
 - The thieves stole the paintings. They were subsequently *sold*.
 - The thieves stole the paintings. They were subsequently *caught*.
 - The thieves stole the paintings. They were subsequently *found*.

2013-03-07

CS372: NLP with Python

24

Generating Language Output

• Question answering

- *Text*: ... The thieves stole the paintings. They were subsequently sold. ...
- *Human*: Who or what was sold?
- *Machine*: The paintings.

2013-03-07

CS372: NLP with Python

25

Generating Language Output

• Machine translation and word sense

- The thieves stole the paintings. They were subsequently found.
- Les voleurs ont volé les peintures. Ils ont été trouvés plus tard. (the thieves)
- Les voleurs ont volé les peintures. Elles ont été trouvées plus tard. (the paintings)

2013-03-07

CS372: NLP with Python

26

Machine Translation

```
>>> babelize_shell()
NLTK Babelizer: type 'help' for a list of commands.
Babel> how long before the next flight to Alice Springs?
Babel> german
Babel> run
0> how long before the next flight to Alice Springs?
1> wie lang vor dem folgenden Flug zu Alice Springs?
2> how long before the following flight to Alice jump?
3> wie lang vor dem folgenden Flug zu Alice springen Sie?
4> how long before the following flight to Alice do you jump?
5> wie lang, bevor der folgende Flug zu Alice tun, Sie springen?
6> how long, before the following flight to Alice does, do you jump?
7> wie lang bevor der folgende Flug zu Alice tut, tun Sie springen?
8> how long before the following flight to Alice does, do you jump?
9> wie lang, bevor der folgende Flug zu Alice tut, tun Sie springen?
10> how long, before the following flight does to Alice, do do you jump?
11> wie lang bevor der folgende Flug zu Alice tut, Sie tun Sprung?
12> how long before the following flight does leap to Alice, does you?
Babel>
```

2013-03-07

CS372: NLP with Python

27

Machine Translation

```
Babel> The pig that John found looked happy
Babel> german
Babel> run
0> The pig that John found looked happy
1> Das Schwein, das John fand, schaute glücklich
2> The pig, which found John, looked happy
Babel>
```

2013-03-07

CS372: NLP with Python

28

Spoken Dialogue Systems

• Example dialogue

- S: How may I help you?
- U: When is Saving Private Ryan playing?
- S: For what theatre?
- U: The Paramount theater.
- S: Saving Private Ryan is not playing at the Paramount theater, but it's playing at the Madison theater at 3:00, 5:30, 8:00, and 10:30.

2013-03-07

CS372: NLP with Python

29

Textual Entailment

• Find evidence to support the hypothesis.

- Hypothesis: *Sandra Goudie was defeated by Max Purnell.*
- Text: *Sandra Goudie was first elected to Parliament in the 2002 elections, narrowly winning the seat of Coromandel by defeating Labour candidate Max Purnell and pushing incumbent Green MP Jeanette Fitzsimons into third place.*

2013-03-07

CS372: NLP with Python

30

Summary

- **Computing with Language: Simple Statistics**
 - Introduction, Frequency Distributions, Fine-Grained Selection of Words, Collocations and Bigrams, Counting Other Things
- **Back to Python: Making Decisions and Taking Control**
 - Conditionals, Operating on Every Element, Nested Code Blocks, Looping with Conditions

2013-03-07

CS372: NLP with Python

31

Summary

- **Automatic Natural Language Understanding**
 - Word Sense Disambiguation, Pronoun Resolution, Generating Language Output, Machine Translation, Spoken Dialogue Systems, Textual Entailment

2013-03-07

CS372: NLP with Python

32