Application of Big Data in Social Science

week 4

Heidi Hyeseung Choi
Fall 2022
HYSIS
heidichoi@hanyang.ac.kr

Announcements

~~9.1~~	~~1~~	~~Introduction~~
~~9.8~~	~~2~~	~~Web Scraping~~
~~9.15~~	~~3~~	
9.22	4	**Natural Language Processing**
9.29	5	
10.6	6	Text Analysis (recorded lecture on week 7)
10.13	7	
10.20	8	Mid term exam (as school schedule)
10.27	9	Social Network Analysis
11.3	10	
11.10	11	Machine Learning: Supervised Learning
11.17	12	
11.24	13	Machine Learning: Unsupervised Learning
12.1	14	
12.8	15	Data Visualization
12.15	16	Final Exam

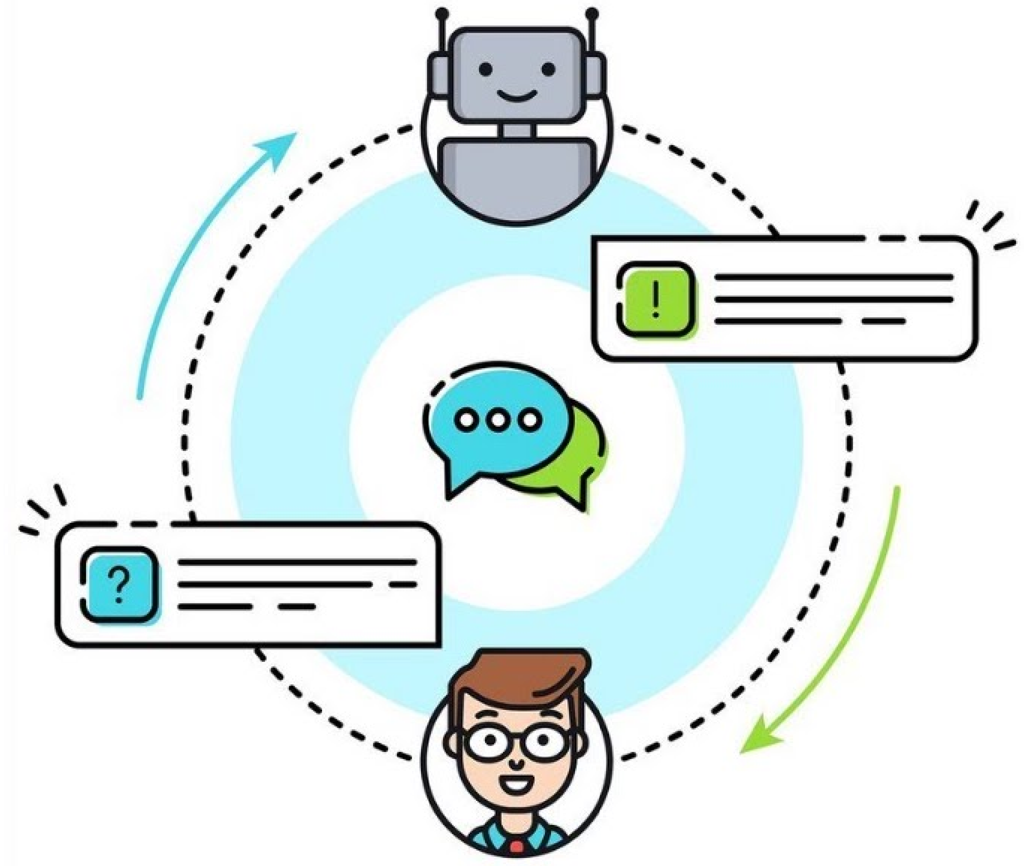
on how to run the class

- On emails: if I do not reply you back, please forward me the email after 2-3 days!
- I will go more slowly (especially when coding)
- I will take a lot of short breaks in between.
- Those who get stuck in the middle:
  1) should wait for that short break
  2) while waiting you should be re-looking at the codes or
  3) search for the solution on google.
  4) ask me questions during short breaks.

Natural Language Processing

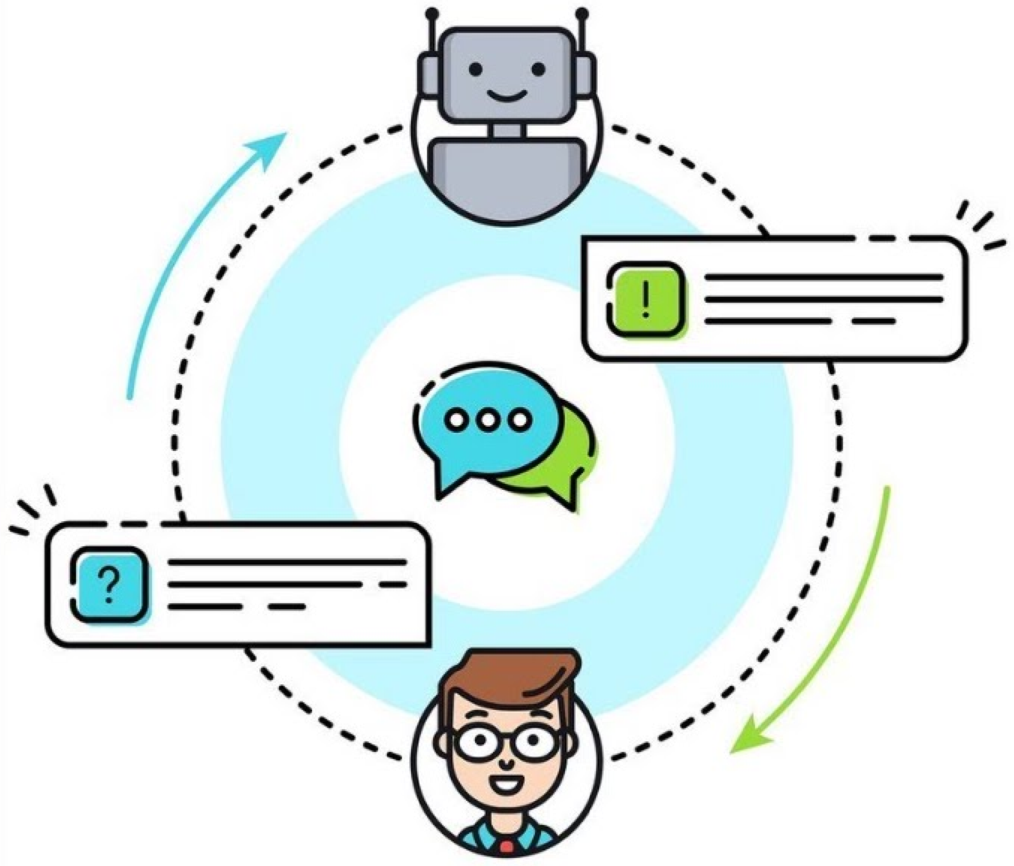
What do they have in common?

- Spellcheck and autocorrect
- Amazon Alexa and Apple's Siri
- Online translators
- News sites' suggested articles
- Chatbots
- filtering E-mail spams
- Customer feelings



What do they have in common?

- Spellcheck and autocorrect
- Amazon Alexa and Apple’s Siri
- Online translators
- News sites’ suggested articles
- Chatbots
- filtering E-mail spams
- Customer feelings



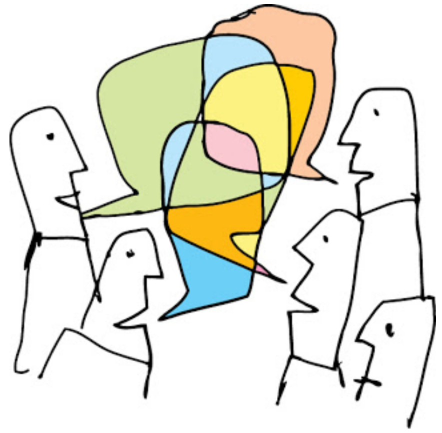
They all use Natural Language Processing techniques!

What is Natural Language Processing (NLP)?

- ‘Natural language’ is a language that humans use everyday for communication purposes.
- A lot of information is contained from our ‘words’.
- However, our daily language is in highly unstructured format.
- That is when NLP comes in handy.
- NLP is the sub-field of AI that focuses on enabling computers to understand and process human language that is highly unstructured.
- NLP is at the intersection of linguistics, artificial intelligence, and computer science.
- The main goal of NLP is to read, decipher, understand and analyze the human language.

Challenges of NLP

- Language does not adhere to structured or regular syntax and patterns.
- A language is concerned with the semantics and the nature of meaning itself.
- A use of language concerns how language is used when speaking, including the speaker's intent, tone, content and actions involved in expressing their words.
- intonation
- be / is / are/ was / were
- go / goes / went



Libraries to be used in this class

- Natural Language Toolkit(NLTK)
- Main modules of NLTK

Language processing task	NLTK modules	Functionality
Accessing corpora	nltk.corpus	Standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	Tokenizers, sentence tokenizers, stemmers
Collocation discovery	nltk.collocation	t-test, chi-squared, point-wise mutual information
Part-of-speech (POS) tagging	nltk.tag	n-gram, backoff, Brill, HMM,
Classification	nltk.classify, nltk.cluster	Decision tree, maximum entropy, naïve Bayes, k-means
Chunking	nltk.chunk	Regular expression, n-gram, named entity
Parsing	nltk.parse	chart, feature-based, unification, probabilistic, dependency

Text Corpora

- Text corpus is a large and structured collection of natural language texts. They usually consists of a body of written or spoken text, often stored in electronic format. It can be large or small, depending on the text.
- A corpus can be broken down into categories of documents or individual documents. Documents contained by a corpus can vary in size, from tweets to books, but they contain text (and sometimes metadata) and a set of related ideas.
- The main purpose of text corpora is to leverage them for linguistic as well as statistical analysis. Also, they are used as learning data for building natural language processing tools.

Commonly used methods and techniques

- POS tagging:
  - the process of marking up a word in a text as corresponding to a particular part of speech. (Noun, adjective, verb etc)
- Word stemming and lemmatization:
  - go, goes, went => go
  - dog, dogs, dog’s, dogs’ => dog
  - baby, babies => baby
- Dependency / constituency grammar:
  - finding out the various relationship among the components in sentences and annotating the dependencies among them.
  - add syntactic annotation to sentences based on their constituents, including phrases and clauses.

Before moving on…

1. Regular Expressions
2. List comprehension

Regular Expressions,
a.k.a.
regex

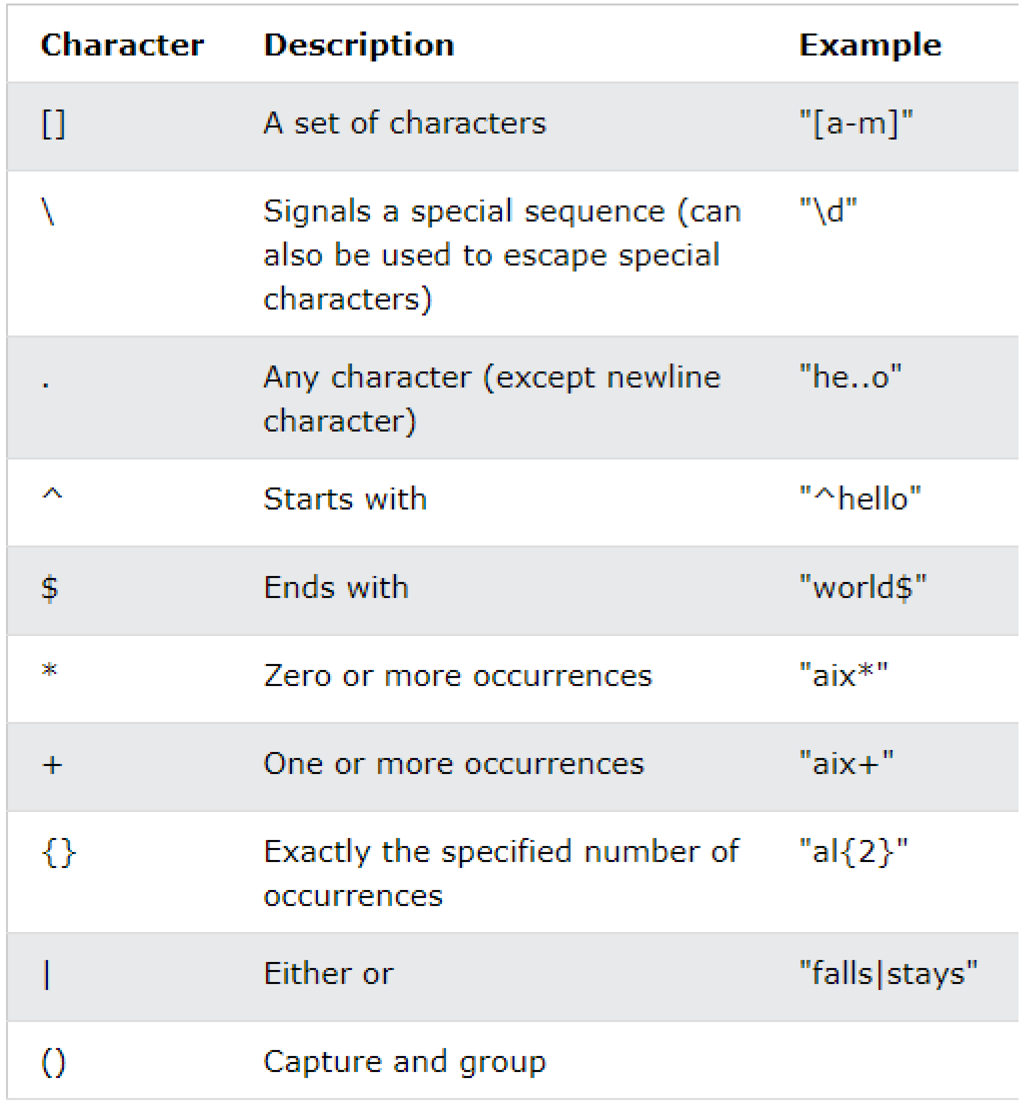
Regular Expressions

Reference link

- Useful in extracting information from any text.
- Is a pattern describing a certain amount of text.

- Metacharacters: characters that are interpreted in a special way by regex.

eg) []/^$*+?{}()\|

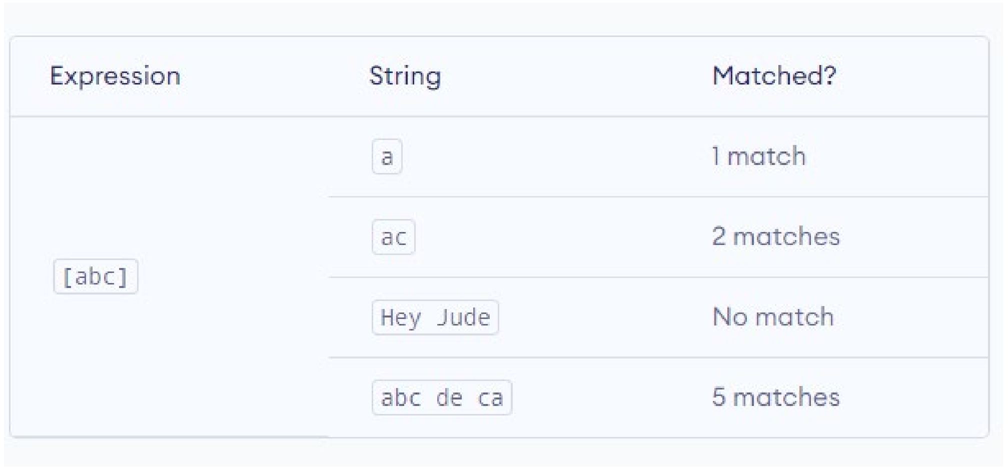


Character	Description	Example
[]	A set of characters	"[a-m]"
\	Signals a special sequence (can also be used to escape special characters)	"\d"
.	Any character (except newline character)	"he..o"
^	Starts with	"^hello"
$	Ends with	"world$"
*	Zero or more occurrences	"aix*"
+	One or more occurrences	"aix+"
{}	Exactly the specified number of occurrences	"al{2}"
|	Either or	"falls|stays"
()	Capture and group	

Regular Expressions

- [abc] will match if the strings that you are trying to match contains any of the a,b, or c.
- You can use – to specify a range.
- [a-d] is same as [abcd]
- [1-4] is same as [1234]
- You can also use caret (^) at the start to indicate "excluding".
- [^a-d] means any characters except a,b,c or d.

Link to website



Expression	String	Matched?
[abc]	a	1 match
	ac	2 matches
	Hey Jude	No match
	abc de ca	5 matches

Regular Expressions

Regex explanation

Regex documentation

Regex Syntax	What it means
[^...]	matches a character not present in the square brackets after the ^ symbol
|	OR operator
+	matches one or more cases of the previous mentioned regex before the + symbol
[A-Za-z]	matches all alphabets (upper and lower cases)
[a-zA-Z0-9_] Same as `\w`	matches any word character. (alpha-numeric characters)
[^a-zA-Z0-9_] Same as `\W`	Returns a match where the string DOES NOT contain any word characters

re.sub(*pattern*, *replace*, *string*)

This method is to substitute a specified regex pattern in a string with a replacement string.

List comprehension

List comprehension

List comprehension provide a concise way to crate a list

Simplest format:

```
>>> [expression for item in iterable]

>>> for item in iterable:
>>>             expression
>>> # example
>>> [x for x in range (1,10)]
>>> [1,2,3,4,5,6,7,8,9]
```

List comprehension

List comprehension with if-conditions (WITH ONLY IF)

You can add an *If* statement at the **end** to return only items which satisfy a certain condition.

```
>>> [expression for item in list if conditional]

>>> for item in list:
>>>         if conditional:
>>>     expression

>>> [x for x in range(1,10) if x %2 ==0]
>>> [2,4,6,8]
```

List comprehension

List comprehension with if-else conditions (with if/else)

Note that you *must* have both *if* and *else* keywords otherwise a SyntaxError is thrown. *elif* does not apply here.

```
# compare the two
>>> [expression for item in list if condition]
>>> [expression if condition else expression for item in list]


>>> for item in list:
>>>        if conditional:
>>>    expression
>>>      else:
>>>             expression
```

List comprehension

List comprehension with append option

```
>>> new_list= [expression for item in list (if conditional)]
```

```
>>> x= []
>>> my_list = [1,2,3]
>>> for a in my_list:
>>>       x.append(a*2)
```

```
x=[a*2 for a in my_list]
```

List comprehension examples

```
>>> s=[]
>>> for x in range (10):
>>>      x**2
>>>      s.append(x)
>>> print(s)
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
>>> t=[]
>>> for x in range (10):
>>>      if x%2 == 0:
>>>           x = x*10
>>>           s.append(x)
>>> print(s)
[0, 20, 40, 60, 80]
```

```
>>> s = [x**2 for x in range(10)]
>>> print(t)
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

```
>>> t = [x*10 for x in range(10) if x%2==0]
>>> print(t)
[0, 20, 40, 60, 80]
```

What we will do today..

1. Scrape the text off a site.
2. Remove HTML tag
3. Tokenization
4. Remove unnecessary tokens and stopwords
5. Add some new stopwords of your choice.

Tokenization

- A paragraph of a text consists of several components, such as sentences, phrases and words.
- Tokenization is to break down or split a text into smaller pieces, called *tokens*.
- A token is an instance of a sequence of characters in a text document that are grouped together as a semantic unit for processing.
- Popular tokenization techniques include sentence and word tokenization, which breaks down a text document (or corpus) into sentences, and each sentence into words.

Welcome to the class

Tokenization

- A paragraph of a text consists of several components, such as sentences, phrases and words.
- Tokenization is to break down or split a text into smaller pieces, called *tokens*.
- A token is an instance of a sequence of characters in a text document that are grouped together as a semantic unit for processing.
- Popular tokenization techniques include sentence and word tokenization, which breaks down a text document (or corpus) into sentences, and each sentence into words.

Welcome	to	the	class

Tokenization

Sentence tokenization:

- *Sentence tokenization* is the process of breaking down a text corpus into sentences as the first level of token.
- There are various ways to tokenize into sentences, such as looking for a specific delimiters between sentences such as a period (.) or a newline character (\n) or a semicolon (;) depending on the document.

Word Tokenization:

- *Word tokenization* is the process of splitting sentences into their words
- Word tokenization is really important in many processes, especially in cleaning and normalizing text where operations like stemming and lemmatization work on each individual word based on its respective stems and lemma.

Stopwords

- Stopwords are words that have little or no significant meaning when analyzing text data.
- Therefore, they are usually removed from text when processing text.
- This is done to retain maximum significance to those that are to be analyzed.
- Words such as ‘a’, ‘the’, ‘and’, are stopwords.
- There is no set list of stopwords, it depends on the domain, language and researchers to set its own stopwords.

Let’s go to Jupyter notebook ☺