UPES

UNIVERSITY WITH A PURPOSE

Computational Linguistic and Natural Language Processing

Course Outcomes

. Understand the basic concepts of text classification and lexical analysis.

. Understanding the basic concepts of Natural Language Processing.

. Understanding the basic concepts of information retrieval

Credit subject (2)

2 Quiz

2 Assignments

2 Test

UP

UNIVERSITY WITH A

Text Processing

- Theory and practice of automating the creation or manipulation of electronic text.
- Text: Alphanumeric characters specified on the keyboard.
- Processing: Automated or mechanized processing.
- Representation of data:

  Text

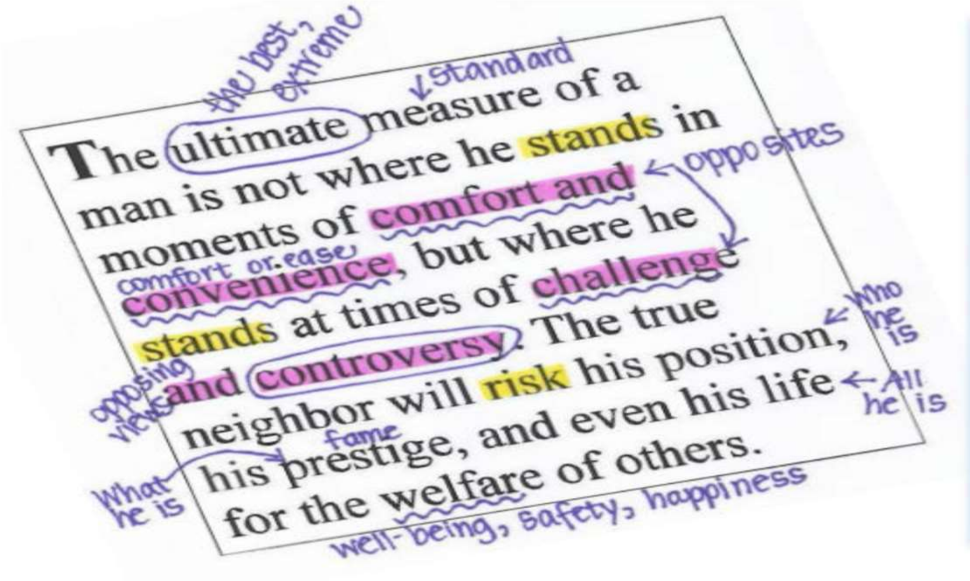
  Images

  Audio

  Videos
- Analysing the data which may be structured or unstructured to obtain structured information.

What Is text processing?

- The textual information: Processed, analyzed and manipulated-machines learn.
- **Text extraction and text classification**
- Extracting individual and small bits of information from large text data is called as text extraction.
- Assigning values to the text data depending upon the content is called as text classification.



What Is text processing?

- How is text processing used?
  - Topic analysis
  - Sentiment analysis
  - Intent detection
  - Language classification
- How can text processing generate business value?
  - Surveys and reviews
  - Support tickets

UP
UNIVERSITY WITH A

Text analysis vs. Text mining vs. Text analytics

- Used to obtain data by statistical pattern learning.
- Both text analysis and text mining are qualitative processes.
- Text Analytics is quantitative process.

Example: Banking service: Customer satisfaction.

- Text analysis: Individual performance of the customer support executive. Text used in the feedback like "good", "bad”.
- Text analytics:
  - Overall performance of all the support executives.
  - Graph for visualizing the performance of the entire support team.

Scope of text analysis/processing

Large documents

- Refer for a context
- Cross examine multiple documents

Individual sentences

- Gathering specific information
- Identify the emotional or intentional activities.

Parts of the sentences

- Sentiments of the words can be analyzed
- Better understanding of the natural language
- Provided for machine to analyse and understand

Importance of text analysis

Business growth:

– Extraction of information to identify the customer

Real time analysis:

– Urgent requirements or complaint handled on a real-time basis.

– Categorized as priority

– Require multiple analysis

Checking for consistency:

– Analysing

– Understanding

– Sharing of the available data accurately

Working principles of text analysis

Data gathering

Data preparation

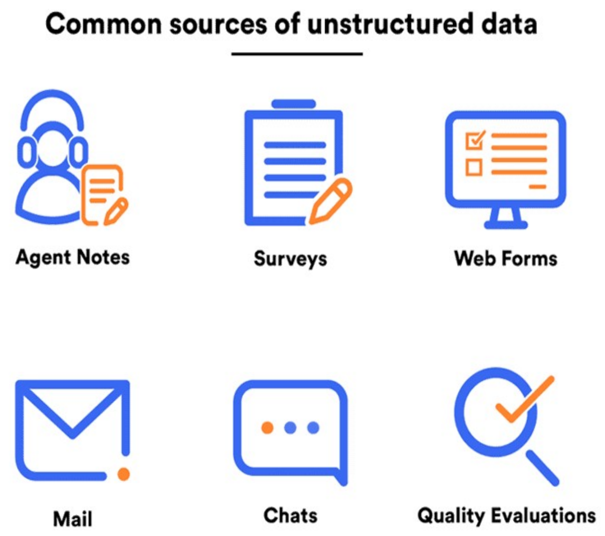
Data analysis

Data gathering

Text analysis: Gathering the required data that need to be analyzed.

Internal data:

- Email
- Chat messages
- CRM tools
- Data bases
- Surveys
- Spread sheets
- Product analysis report



External data: The external data do not belong to the organization and are available fre through other sources.

- Web scraping tools.
- Open data.

Data preparation

Before text is analysed by any machine learning algorithm, it needs to be prepared.

Regular expressions

- A formal language for specifying text strings
- How can we search for any of these?
- woodchuck
- woodchucks
- Woodchuck
- Woodchucks

Regular Expressions: Disjunctions

Letters inside square brackets []

Pattern	Matches
[wW]oodchuck	Woodchuck, woodchuck
[1234567890]	Any digit

Ranges [A-Z]

Pattern	Matches	
[A-Z]	An upper case letter	Drenched Blossoms
[a-z]	A lower case letter	my beans were impatient
[0-9]	A single digit	Chapter 1: Down the Rabbit Hole

Regular Expressions: Negation in Disjunctions

- Negations `[^Ss]`
  - Carat means negation only when first in []

Pattern	Matches	
`[^A-Z]`	Not an upper case letter	Oyfn pripetchik
`[^Ss]`	Neither 'S' nor 's'	I have no exquisite reason"
`[^e^]`	Neither e nor ^	Look here
`a^b`	The pattern a carat b	Look up a^b now

Regular Expressions: More in Disjunctions

- Woodchucks is another name for groundhog!
- The pipe | for disjunction

Pattern	Matches
groundhog|woodchuck	
yours|mine	yours mine
a|b|c	= [abc]
[gG]roundhog|[Ww]oodchuck	

Regular Expressions: ? * + .

Pattern	Matches	
colou?r	Optional previous char	color colour
oo*h!	0 or more of previous char	oh! ooh! oooh! ooooh!
o+h!	1 or more of previous char	oh! ooh! oooh! ooooh!
baa+		baa baaa baaaa baaaaa
beg.n		begin begun begun beg3n

Regular Expressions: Anchors ^ $

Pattern	Matches
^[A-Z]	Palo Alto
^[^A-Za-z]	1 "Hello"
\.$	The end.
.$	The end? The end!

Example

Find me all instances of the word “the” in a text.

`the`

Misses capitalized examples

`[tT]he`

Incorrectly returns `other` or `theology`

`[^a-zA-Z][tT]he[^a-zA-Z]`

Errors

- The process we just went through was based on fixing two kinds of errors
  - Matching strings that we should not have matched (there, then, other)
    - False positives (Type I)
  - Not matching things that we should have matched (The)
    - False negatives (Type II)

Errors

- In NLP we are always dealing with these kinds of errors.
- Reducing the error rate for an application often involves two antagonistic efforts:
  - Increasing accuracy or precision (minimizing false positives)
  - Increasing coverage or recall (minimizing false negatives).

Summary

- Regular expressions play a surprisingly large role
  - Sophisticated sequences of regular expressions are often the first model for any text processing text
- For many hard tasks, we use machine learning classifiers
  - But regular expressions are used as features in the classifiers
  - Can be very useful in capturing generalizations

Word tokenization

Text Normalization

Every NLP task needs to do text normalization:

1. Segmenting/tokenizing words in running text

2. Normalizing word formats

3. Segmenting sentences in running text

How many words?

- I do uh main- mainly business data processing
    - Fragments, filled pauses
- Seuss's cat in the hat is different from other cats!
    - **Lemma**: same stem, part of speech, rough word sense
        - cat and cats = same lemma
    - **Wordform**: the full inflected surface form
        - cat and cats = different wordforms

How many words?

they lay back on the San Francisco grass and looked at the stars and their

- **Type**: an element of the vocabulary.
- **Token**: an instance of that type in running text.
- How many?
  - 15 tokens (or 14)
  - 13 types (or 12) (or 11?)

How many words?

N = number of tokens

V = vocabulary = set of types

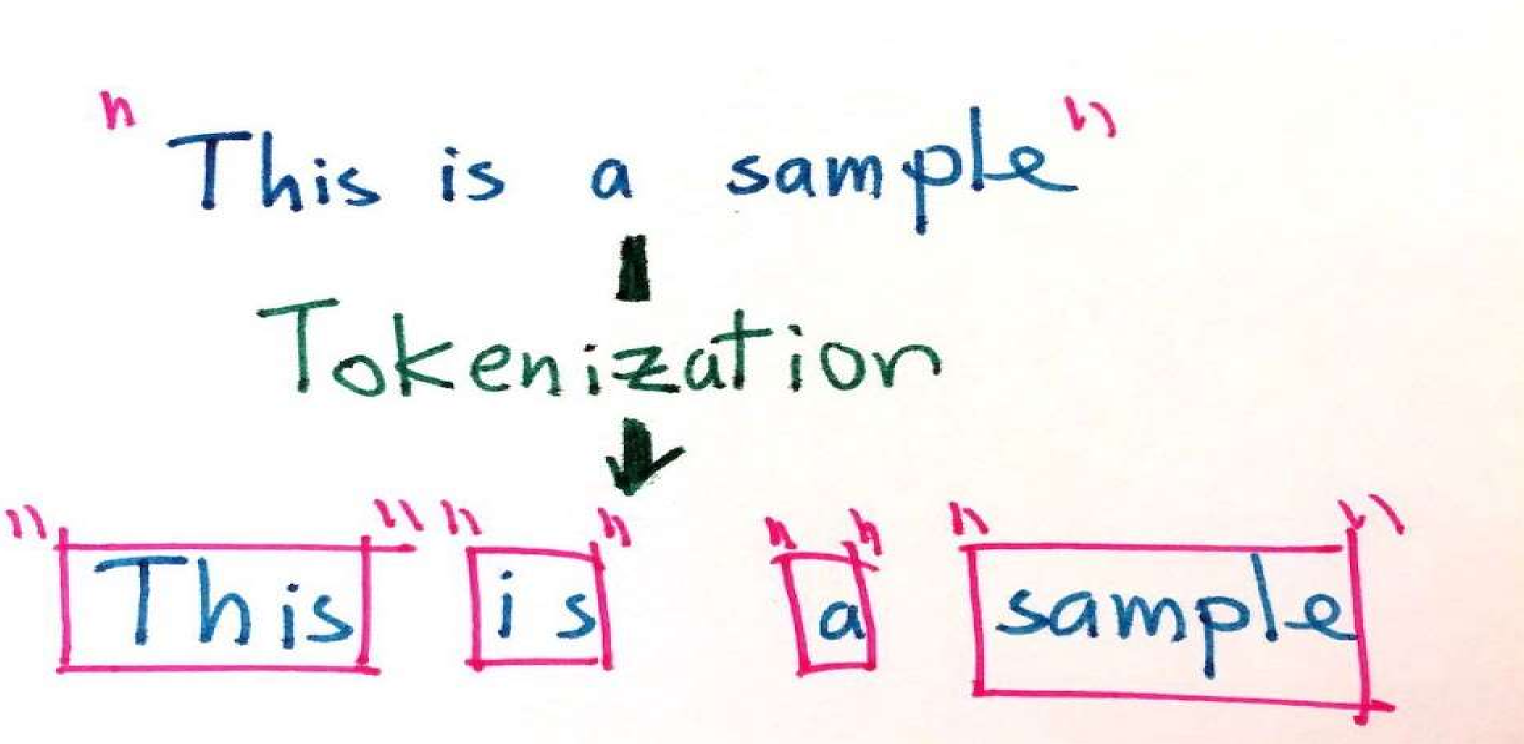
$|V|$ is the size of the vocabulary

Church and Gale (1990): $|V| > O(N^{½})$

	Tokens = N	Types = |V|
Switchboard phone conversations	2.4 million	20 thousand
Shakespeare	884,000	31 thousand
Google N-grams	1 trillion	13 million

Tokenization

Tokenization is the process of breaking down the given text in natural language processing into the smallest unit in a sentence called a token.



Tokenization with NLTK

Tokenization is the process of breaking down the given text in natural language processing into the smallest unit in a sentence called a token.

```
import nltk
from nltk import sent_tokenize
from nltk import word_tokenize
```

```
tokens_sents = nltk.sent_tokenize(text)
print(tokens)
```

```
['Hello everyone!', 'Welcome to my blog post on Medium.', 'We are studying Natural Language
Processing.']
```

```
tokens_words = nltk.word_tokenize(text)
print(tokens_words)
```

```
['Hello', 'everyone', '!', 'Welcome', 'to', 'my', 'blog', 'post', 'on', 'Medium', '.', 'We',
'are', 'studying', 'Natural', 'Language', 'Processing', '.']
```

Issues in Tokenization

- Finland's capital → Finland Finlands Finland's ?
- what're, I'm, isn't → What are, I am, is not
- Hewlett-Packard → Hewlett Packard ?
- state-of-the-art → state of the art ?
- Lowercase → lower-case lowercase lower case ?
- San Francisco → one token or two?
- m.p.h., PhD. → ??

Tokenization: language issues

- French: L'ensemble→ one token or two?
- German noun compounds are not segmented.
- Chinese and Japanese no spaces between words
- Maximum Matching Word Segmentation Algorithm
- Ex.
- Thecatinthehat (the cat in the hat)
- Thetabledownthere (the table down there) (theta bled own ther
- Doesn’t generally work in English!

Word Normalization

- In the field of linguistics and NLP, Morpheme is defined as a base form of the word. A token is basically made up of two components one is morphemes and the other is inflectional form like prefix or suffix.
- For example, consider the word Antinationalist (Anti + national+ ist) which is mad up of Anti and ist as inflectional forms and national as the morpheme.

Word Normalization

- Need to “normalize” terms
  - Information Retrieval: indexed text & query terms must have same form.
    - We want to match ***U.S.A.*** and ***USA***
- We implicitly define equivalence classes of terms
  - e.g., deleting periods in a term
- Alternative: asymmetric expansion:
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***
- Potentially more powerful, but less efficient

Stemming

- Stemming is the process of finding the root of words.
- Stemming is definitely the simpler of the two approaches. With stemming, words ar reduced to their word stems. A word stem need not be the same root as a dictionary based morphological root, it just is an equal to or smaller form of the word.
- It is an elementary rule-based process for removing inflationary forms from a given token.

Form	Suffix	Stem
studies	-es	studi
studying	-ing	study
niñas	-as	niñ
niñez	-ez	niñ

Stemming

- Reduce terms to their stems in information retrieval
- *Stemming* is crude chopping of affixes
  - language dependent
  - e.g., ***automate(s), automatic, automation*** all reduced to ***automat***.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Stemming

- Stemming is not a good process for normalization. since sometimes it can produce non-meaningful words which are not present in the dictionary. Consider the sentenc
” His teams are not winning”. After stemming we get “Hi team are not winn ” .
- We can examine the stemming example with two different algorithms.
- Porter Stemmer
- Snowball Stemmer

Stemming

```
from nltk.stem import PorterStemmer
```

```
ps = PorterStemmer()
word = ("civilization")
ps.stem(word)
```

```
'civil'
```

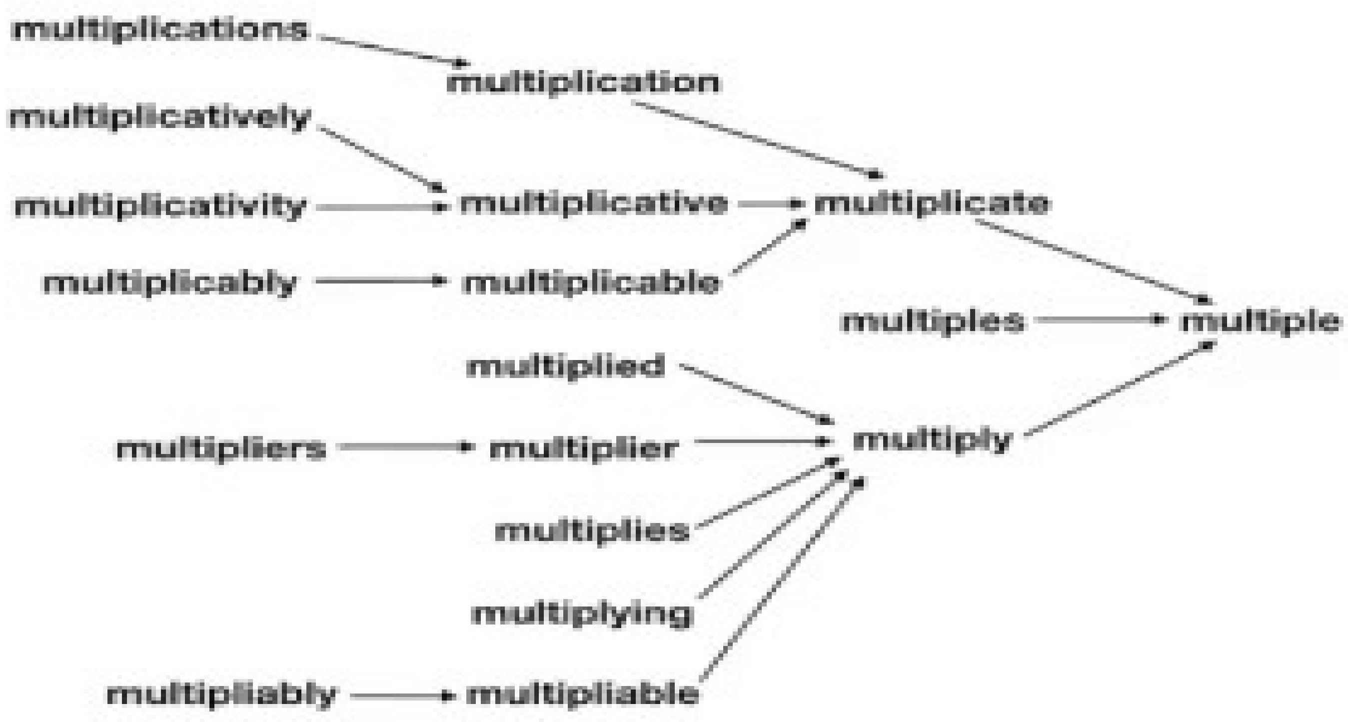
```
from nltk.stem.snowball import SnowballStemmer
```

```
stemmer = SnowballStemmer(language = "english")
word = "civilization"
stemmer.stem(word)
```

```
'civil'
```

Lemmatization

Lemmatization is a systematic process of removing the inflectional form of a token transform it into a lemma. It makes use of word structure, vocabulary, part of speech t and grammar relations.



Lemmatization

- Reduce inflections or variant forms to base form
  - *am, are, is* → *be*
  - *car, cars, car's, cars'* → *car*
- *the boy's cars are different colors* → *the boy car be different color*
- Lemmatization: have to find correct dictionary headword form
- Machine translation
  - Spanish quiero ('I want'), quieres ('you want') same lemma as querer 'want'

Lemmatization

The output of lemmatization is a root word called a lemma. for example “am”, “are”, will be converted to “be”. Similarly, running runs, ‘ran’ will be replaced by ‘run’.

```
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
```

```
# Lemmatize single word

print(lemmatizer.lemmatize("workers"))
print(lemmatizer.lemmatize("beeches"))
```

```
worker
beech
```

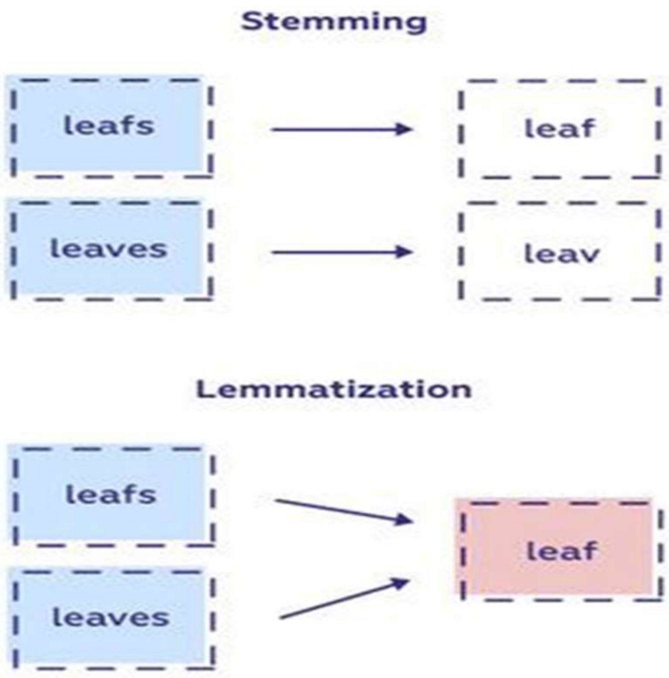
```
text = "Let's lemmatize a simple sentence. We first tokenize the sentence into words using nltk.wo
word_list = nltk.word_tokenize(text)
print(word_list)
```

```
['Let', "'", 's', 'lemmatize', 'a', 'simple', 'sentence', '.', 'We', 'first', 'tokenize', 'the',
'sentence', 'into', 'words', 'using', 'nltk.word_tokenize', 'and', 'then', 'we', 'will', 'call',
'lemmatizer.lemmatize', '(', ')', 'on', 'each', 'word', '.']
```

```
lemmatized_output = ' '.join([lemmatizer.lemmatize(w) for w in word_list])
print(lemmatized_output)
```

```
Let ' s lemmatize a simple sentence . We first tokenize the sentence into word using
nltk.word_tokenize and then we will call lemmatizer.lemmatize ( ) on each word .
```

Lemmatization



Porter's algorithm

The Porter stemming algorithm (or 'Porter stemmer') is a process for removing commoner morphological and inflexional endings from words in English.

Step 1a

```
sses → ss   caresses → caress
ies  → i    ponies   → poni
ss   → ss   caress   → caress
s    → ø    cats     → cat
```

Step 1b

```
(*v*)ing → ø  walking   → walk
              sing      → sing
(*v*)ed  → ø  plastered → plaster
```

Step 2 (for long stems)

```
ational→ ate  relational→ relate
izer→ ize     digitizer → digitize
ator→ ate     operator  → operate
…
```

Step 3 (for longer stems)

```
al   → ø  revival    → reviv
able → ø  adjustable → adjust
ate  → ø  activate   → activ
```

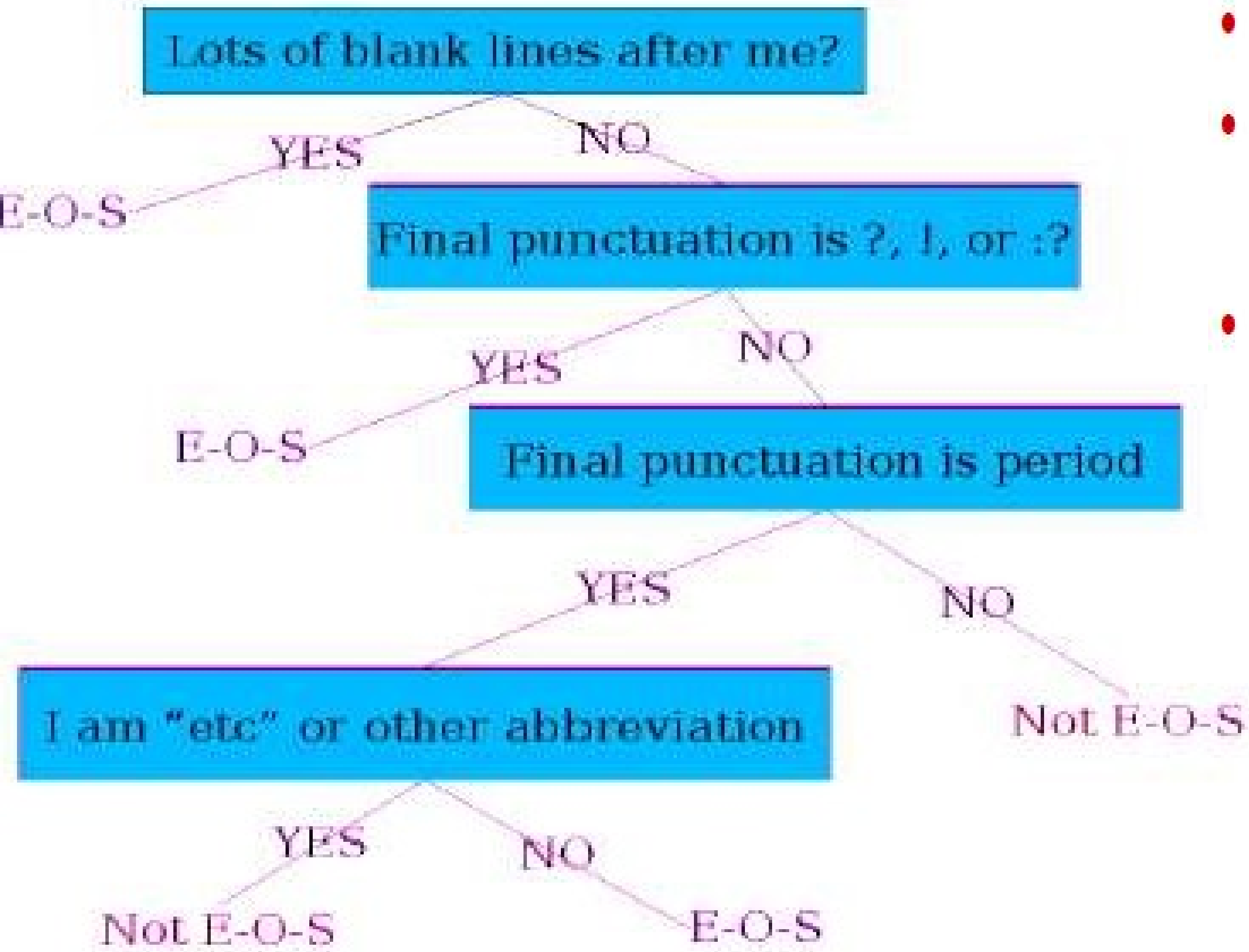
Sentence Segmentation

Sentence tokenization (also called sentence segmentation) is the problem of dividin string of written language into its component sentences.

"sentences = nltk.sent_tokenize(text)"

- !, ? are relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
  - Numbers like .02% or 4.3
- Build a binary classifier
  - Looks at a "."
  - Decides EndOfSentence/NotEndOfSentence
  - Classifiers: hand-written rules, regular expressions, or machine-learning

Determining if a word is end--of--sentence: a Decision Tree



- Case of word with ".": Upper, Lower, Cap, Nu
- Case of word after ".": Upper, Lower, Cap, Nu
- Numeric features
  - Length of word with "."
  - Probability(word with "." occurs at end-of-s)
  - Probability(word after "." occurs at beginning-of-s)

Determining if a word is end--of--sentence: a Decision Tree

- A Decision tree is just an if-then-else statement.
- The interesting research in a decision tree is choosing the features.
- These features could be exploited by any kind of classifier like Logistic Regression, SVMs etc.

Data analysis

- Machine learning based systems.
- Predict based upon the past observations.
- Require multiple samples of text and tags.
- Converted into vectors before training.
- Vectors: Extract the features that are relevant.
- Two major processes: Text classification and Text extraction.