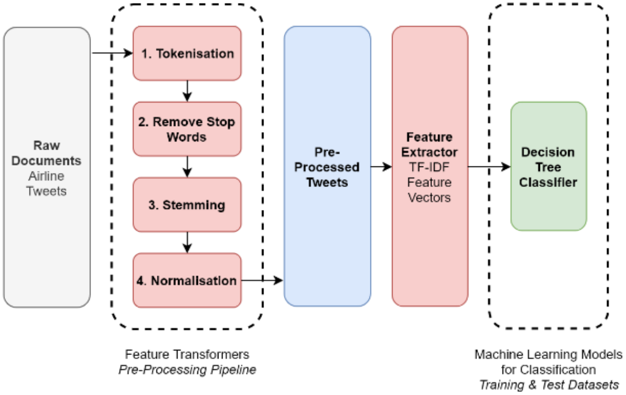
NATURAL LANGUAGE PROCESSING (PRACTICE)

NLP 242 - Lab 2: EXPLORING AND PREPROCESSING TEXT DATA

Department of Computer Science and Engineering
Ho Chi Minh University of Technology, VNU-HCM

Data Preprocessing

Data Preprocessing



- Data preprocessing is a crucial step
- 70%-80% of effort goes into data processing

Why is text preprocessing necessary?

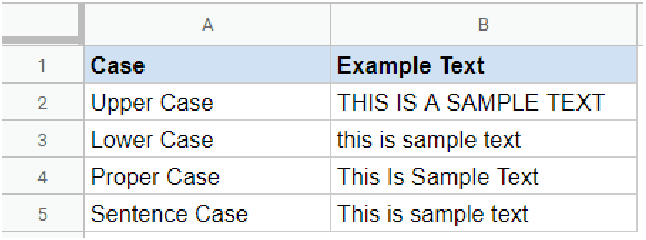
- Multiple sources: web, HTML, documents...
- Contains noise and unclean data
- Goal: Convert to understandable format

Text Data Preprocessing Techniques

Text Data Preprocessing Techniques

- Convert to lowercase
- Remove punctuation
- Remove stop words
- Standardize text
- Correct spelling
- Tokenize
- Lemmatization and stemming
- Explore text data
- Build a preprocessing program

Converting to Lowercase

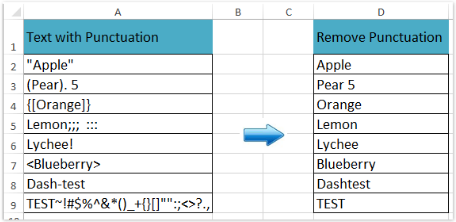


	A	B
1	**Case**	**Example Text**
2	Upper Case	THIS IS A SAMPLE TEXT
3	Lower Case	this is sample text
4	Proper Case	This Is Sample Text
5	Sentence Case	This is sample text

Problem: All text data needs consistent formatting, ensures ”NLP” and ”nlp” are treated the same

- Input: Text content
- Output: Text converted to lowercase
- Solution: Use lower() function in Python

Removing Punctuation



	A	B	C	D
1	Text with Punctuation			Remove Punctuation
2	"Apple"			Apple
3	(Pear). 5			Pear 5
4	{[Orange]}			Orange
5	Lemon;;; :::			Lemon
6	Lychee!			Lychee
7	<Blueberry>			Blueberry
8	Dash-test			Dashtest
9	TEST~!#$%^&*()_+{}[]"":;<>?.,			TEST

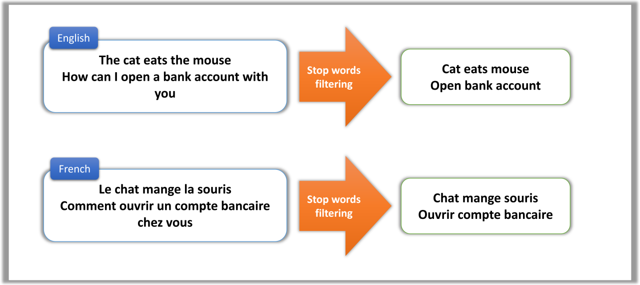
Target:

- Important because punctuation doesn't add information
- Reduces data size while improving computational efficiency

Problem

- Input: Text content
- Output: Text with punctuation removed
- Solution: Use regular expressions and replace() function in Python

Removing Stop Words



Stop words: common, but not very meaningful. Removing them helps to: reduce data size and potentially improve model performance.

- Solution:
  - Use NLTK library
  - Build a list of stop words, then use it to remove the stop words present in the text. (VNese Ex: https://github.com/stopwords/vietnamese-stopwords)

Standardizing Text

Raw	**Normalized**
2moro	tomorrow
2mrrw	tomorrow
2morrow	tomorrow
tomrw	tomorrow
b4	before
otw	on the way
:)	smile
:-)	smile
;-)	smile

- Input: Text content
- Output: Standardized text
- Solution: Create dictionary to look up abbreviations and short forms

Correcting Spelling

Common Examples

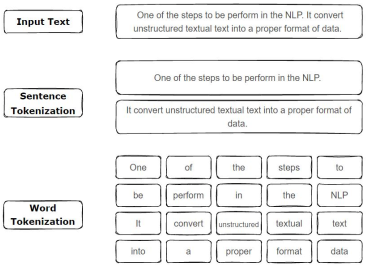
studying → studying
intresting → interesting
aquire → acquire

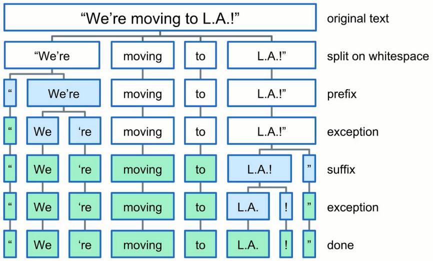
- Text contains spelling errors (user evaluations, blogs, tweets, etc.)
- Reduce the number of duplicates of words. For example, if not fixed, ”studing” and ”studying” would be considered two different words.

Problem:

- Input: A text containing content.
- Output: The text with corrected spelling errors.
- Solution: Use **TextBlob** library

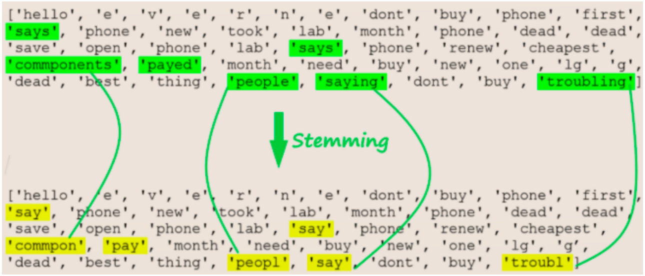
Tokenizing


Input Text
One of the steps to be perform in the NLP. It convert unstructured textual text into a proper format of data.
Sentence Tokenization
One of the steps to be perform in the NLP.
It convert unstructured textual text into a proper format of data.
Word Tokenization
One of the steps to be be perform in the NLP It convert unstructured textual text into a proper format data



- Input: Text content
- Output: Separated sentences or words
- Solutions:
  - For English: NLTK, SpaCy, TextBlob
  - For Vietnamese: VnCoreNLP, underthesea, coccoc-tokenizer

Stemming and Lemmatization



```
['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',
'says', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead',
'save', 'open', 'phone', 'lab', 'says', 'phone', 'renew', 'cheapest',
'commponents', 'payed', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',
'dead', 'best', 'thing', 'people', 'saying', 'dont', 'buy', 'troubling']
```

```
['hello', 'e', 'v', 'e', 'r', 'n', 'e', 'dont', 'buy', 'phone', 'first',
'say', 'phone', 'new', 'took', 'lab', 'month', 'phone', 'dead', 'dead',
'save', 'open', 'phone', 'lab', 'say', 'phone', 'renew', 'cheapest',
'commpon', 'pay', 'month', 'need', 'buy', 'new', 'one', 'lg', 'g',
'dead', 'best', 'thing', 'peopl', 'say', 'dont', 'buy', 'troubl']
```

Stemming

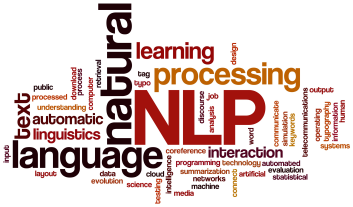
- Input: "fishing", "fishes"
- Output: Root "fish"
- Removes prefixes/suffixes

Lemmatization

- Input: "good", "best"
- Output: Root "good"
- Considers semantics

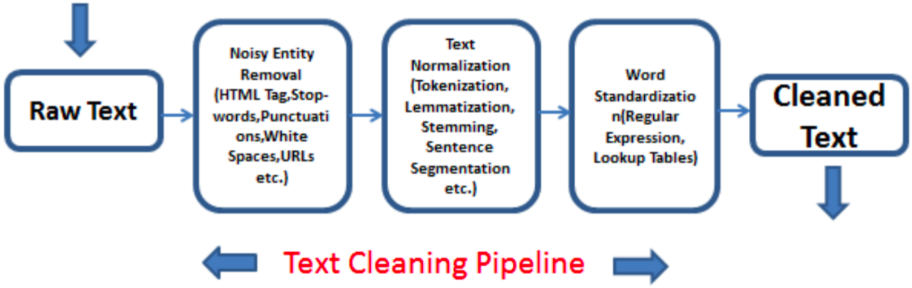
Solution: Use **NLTK** or **TextBlob**

Exploring Text Data



- **Input:** Text content
- **Output:**
  - Word count
  - Word frequency
  - Distribution of words longer than 3 characters
  - Word cloud
- **Solution:** Use NLTK or TextBlob

Building Preprocessing Program


Raw Text
Noisy Entity Removal (HTML Tag,Stop-words,Punctuations,White Spaces,URLs etc.)
Text Normalization (Tokenization, Lemmatization, Stemming, Sentence Segmentation etc.)
Word Standardization(Regular Expression, Lookup Tables)
Cleaned Text
Text Cleaning Pipeline

- **Input:** Text content
- **Output:** Preprocessed text
- **Solution:** Create preprocessing function incorporating all techniques discussed

THANKS FOR LISTENING!