# Workshop 4

**COMP20008**

Elements of Data Processing
Zijie Xu

## Agenda

- Text pre-processing and Bag of Words
  - Regular expressions
- Data visualisation

## Text pre-processing

1. Sentence splitting
   - Goal: split word strings into sentences
   - Problem: Numbers ($3.50) and abbreviations (a.k.a.)
2. Word splitting
   - Goal: split sentences into word tokens
   - Problem: punctuations (it’s), cases (US & us) and other non-canonical text forms ($1,234.56 & €1.234,56    13:30 & 1.30 pm)
   - Tool: regular expression

# Text pre-processing

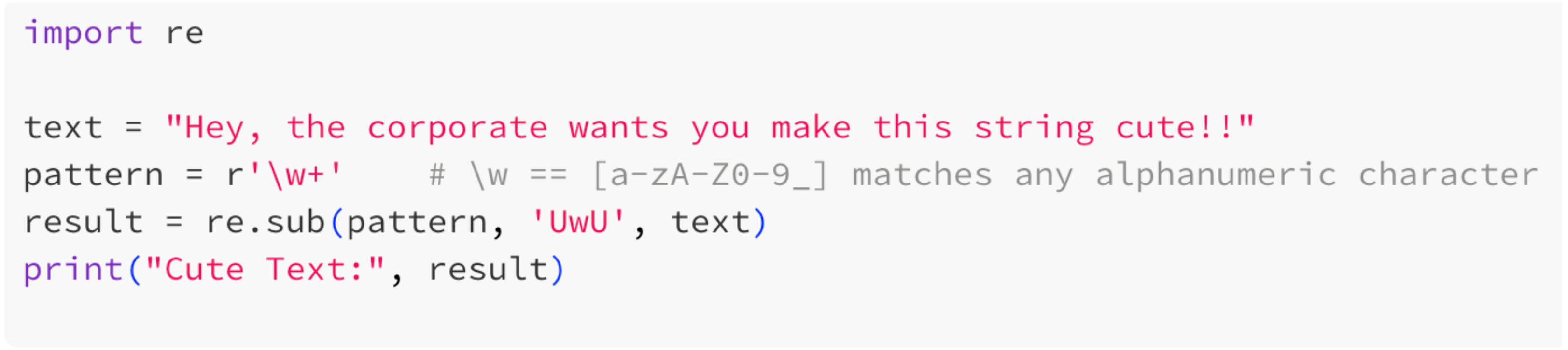
3. Word regularisation
   - Goal: find one representation for many morphologies of a word
   - Problem: number, tense, aspect; root + more parts
   - Tools: stemming (Porter stemmer), Lemmatising (WordNet)
4. Bag of words
   - Representing a document as a count of every word it contains
   - N-gram: N successive word(s) as a unit
   - One way to represent text as a numeric vector

# Regular expression

- Regular expression (RegEx) defines a search pattern that can be used for pattern matching and text manipulation tasks

- `re.sub(pattern, replacement, string)`

  A function from the `re` module that searches for occurrences of `pattern` in `string` and replaces them with `replacement`



```python
import re

text = "Hey, the corporate wants you make this string cute!!"
pattern = r'\w+'    # \w == [a-zA-Z0-9_] matches any alphanumeric character
result = re.sub(pattern, 'UwU', text)
print("Cute Text:", result)
```

```
Cute Text: UwU, UwU UwU UwU UwU UwU UwU UwU UwU!!
```

# Regular expression

- `+` repeat for one or more times
- `[]` match any character from a set (class) of characters
  - `[^]` add ‘^’ as **first character** for complementing set
- `\` escape special characters: `"\\w" == r"\w"`

  define charcter sets: `\s == [ \t\n\r\f\v]`
- `(?= )` specify what comes immediately after a match

```python
import re

text = "A regular chad is here writing some code with a regular chad."
pattern = "regular chad(?= is here)" # Match `regular chad` immediately before ` is here`
result = re.sub(pattern, 'GIGACHAD', text)
print("After replacement:", result)
```

```
After replacement: A GIGACHAD is here writing some code with a regular chad.
```

## Data visualisation

- Some common charts

| | |
|---|---|
| **Line Charts** | Display trends over time |
| **Histograms** | Represent the distribution of data |
| **Bar Charts** | Used to compare values across categories |
| **Scatter Plots** | Display relationships between two variables |
| **Heatmaps** | Represent data using colours on a grid |

- Both Matplotlib and Seaborn are great packages to plot visualisation with python
  - `import matplotlib.pyplot as plt`
  - `import seaborn as sns`

THE UNIVERSITY OF MELBOURNE

## Thank you

More Resources: Canvas