THE UNIVERSITY OF MELBOURNE

Workshop 5

COMP20008

Elements of Data Processing

Zijie Xu

Agenda

- Regular expressions
- N-grams and similarity

Regular Expressions

- Regular Expressions (RegEx) enable searching, matching, and manipulation of strings based on specific patterns
  - Python `re` module API and Tutorial
- Some useful methods
  - `re.search(`*`pattern, string`*`)`
  - `re.findall(`*`pattern, string`*`)`
  - `re.sub(`*`pattern, replacement, string`*`)`
  - `re.split(`*`pattern, string`*`)`

Regular Expressions

- Metacharacters `.^$*+?{}[]\|()`
  - Wildcard
    - `.` Matches any character
  - Anchor
    - `^` Start of string
    - `$` End of string
  - Repeats
    - `*` ≥0
    - `+` ≥1
    - `?` 0 or 1
    - `{m,n}` m ≤ # repeat ≤ n

Regular Expressions

- Metacharacters
  - Character class/set
    - `[ ]` matches any character from a class of characters
    - `[^]` '`^`' as **first character** for complementing class
    - Metacharacters (except `\`) do not work in classes and will be matched as literals
    - Some predefined classes: `\w \W \d \D \s \S`

Regular Expressions

- Alternation
  - `|` split alternative patterns
- Capture groups
  - `( )` captures the matched part for later reference

```
In [15]: text = 'To Be Or Not To Be? That is the question.'

In [16]: re.findall(r'(.+) Or Not \1', text)
Out[16]: ['To Be']
```

- Lookahead assertions
  - `x(?=y)` matches x only if it is followed by y
  - `x(?!y)` matches x only if it is not followed by y
  - Not part of the matched

Regular Expressions

- `\`
  - Escapes metacharacters

  Use raw strings to avoid typing many double backslashes

  `'\\$' == r'\$'`
  - Escapes the name of a character class `\d \w`
  - Back-references a sequence captured by a capture groups `\1 \2`

N-grams and similarity

- N-gram: a sequences of n contiguous items from text
- Letter N-gram: n-gram sequences where items are individual letters
  - '#' stands for padding
  - Bi-grams of 'crat': $G_2(\texttt{crat}) = [\texttt{\#c, cr, ra, at, t\#}]$
- We can use n-gram/letter n-gram to compare how similar two documents/strings are

N-grams and similarity

- There are many metrics for finding the similarity between two strings/documents
  - N-gram: Simple n-gram, Jaccard, Dice
  - Edit distance based: Levenshtein, Hamming
  - Geometric: Cosine
  - Distance based: Manhattan, Euclidean, Minkowski

THE UNIVERSITY OF MELBOURNE

Thank you

More Resources: Canvas