Applied Text Mining in Python

Handling Text in Python

Primitive constructs in Text

- Sentences / input strings
- Words or Tokens
- Characters
- Document, larger files

And their properties …

Let's try it out!

```
>>> text1 = "Ethics are built right into the ideals and objectives
of the United Nations "
>>> len(text1)
76
>>> text2 = text1.split(' ')
>>> len(text2)
13
>>> text2
['Ethics', 'are', 'built', 'right', 'into', 'the', 'ideals', 'and',
'objectives', 'of', 'the', 'United', 'Nations', '']
```

Finding specific words

- **Long words: Words that are most than 3 letters long**

```
>>> [w for w in text2 if len(w) > 3]
['Ethics', 'built', 'right', 'into', 'ideals', 'objectives', 'United',
'Nations']
```

- **Capitalized words**

```
>>> [w for w in text2 if w.istitle()]
['Ethics', 'United', 'Nations']
```

- **Words that end with s**

```
>>> [w for w in text2 if w.endswith('s')]
['Ethics', 'ideals', 'objectives', 'Nations']
```

Finding unique words: using set()

```
>>> text3 = 'To be or not to be'
>>> text4 = text3.split(' ')
>>> len(text4)
6
>>> len(set(text4))
5
>>> set(text4)
set(['not', 'To', 'or', 'to', 'be'])
>>> len(set([w.lower() for w in text4]))
4
>>> set([w.lower() for w in text4])
set(['not', 'to', 'or', 'be']
```

Some word comparison functions …

- s.startswith(t)
- s.endswith(t)
- t in s
- s.isupper(); s.islower(); s.istitle()
- s.isalpha(); s.isdigit(); s.isalnum()

String Operations

- s.lower(); s.upper(); s.titlecase()
- s.split(t)
- s.splitlines()
- s.join(t)
- s.strip(); s.rstrip()
- s.find(t); s.rfind(t)
- s.replace(u, v)

From words to characters

```
>>> text5 = 'ouagadougou'
>>> text6 = text5.split('ou')
>>> text6
['', 'agad', 'g', '']
>>> 'ou'.join(text6)
'ouagadougou'
```

```
>>> text5.split('')
Traceback (most recent call last):
  File "<stdin>", line 1, in
<module>
ValueError: empty separator
>>> list(text5)
['o', 'u', 'a', 'g', 'a', 'd',
'o', 'u', 'g', 'o', 'u']
>>> [c for c in text5]
['o', 'u', 'a', 'g', 'a', 'd',
'o', 'u', 'g', 'o', 'u']
```

Cleaning Text

```
>>> text8 = '    A quick brown fox jumped over the lazy dog. '
>>> text8.split(' ')
['', '', '\t', 'A', 'quick', 'brown', 'fox', 'jumped', 'over',
'the', 'lazy', 'dog.', '']
>>> text9 = text8.strip()
>>> text9.split(' ')
['A', 'quick', 'brown', 'fox', 'jumped', 'over', 'the', 'lazy',
'dog.']
```

Changing Text

- **Find and replace**

```
>>> text9
'A quick brown fox jumped over the lazy dog.'
>>> text9.find('o')
10
>>> text9.rfind('o')
40
>>> text9.replace('o', 'O')
'A quick brOwn fOx jumped Over the lazy dOg.'
```

Handling Larger Texts

- **Reading files line by line**

```
>>> f = open('UNDHR.txt', 'r')
>>> f.readline()
'Universal Declaration of Human Rights\n'
```

- **Reading the full file**

```
>>> f.seek(0)
>>> text12 = f.read()
>>> len(text12)
10891
>>> text13 = text12.splitlines()
>>> len(text13)
158
>>> text13[0]
'Universal Declaration of Human Rights'
```

File Operations

- f = open(*filename*, *mode*)
- f.readline(); f.read(); f.read(*n*)
- for *line* in f: doSomething(*line*)
- f.seek(*n*)
- f.write(*message*)
- f.close()
- f.closed

Issues with reading text files

```
>>> f = open('UNDHR.txt', 'r')
>>> text14 = f.readline()
'Universal Declaration of Human Rights\n'
```

- **How do you remove the last newline character?**

```
>>> text14.rstrip()
'Universal Declaration of Human Rights'
```

  - Works also for DOS newlines (`^M`) that shows up as `'\r'` or `'\r\n'`

Take Home Concepts

- Handling text sentences
- Splitting sentences into words, words into characters
- Finding unique words
- Handling text from documents