Applied Text Mining in Python

Internationalization

World of Languages

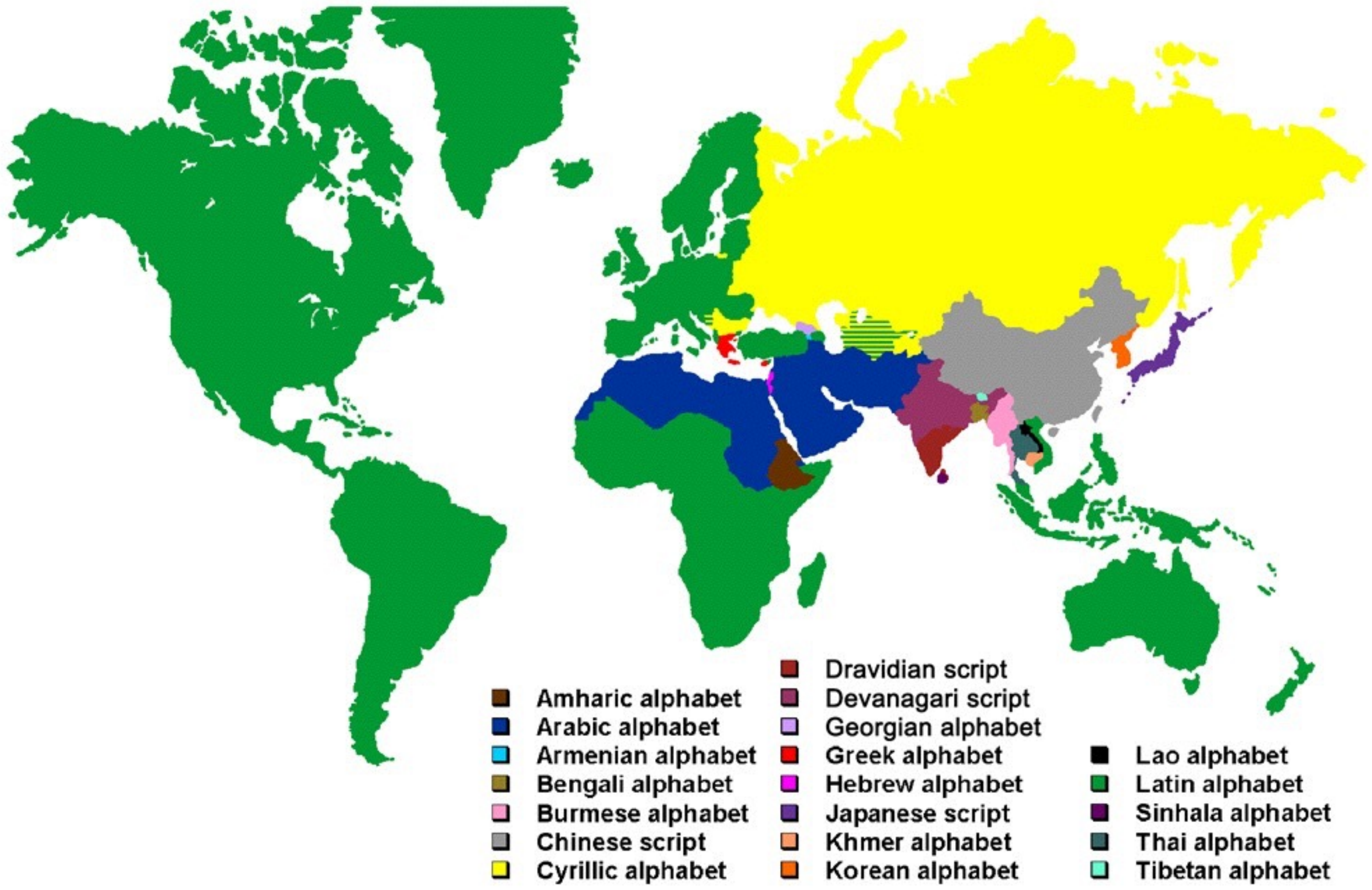
English and ASCII

- **ASCII: American Standard Code for Information Interchange**
  - 7-bit character encoding standard: 128 valid codes
  - Range: 0x00 – 0x7F [$(0000\ 0000)_2$ to $(0111\ 1111)_2$]
  - Includes alphabets (upper and lower cases), digits, punctuations, common symbols, control characters
  - Worked (relatively) well for English typewriting

Resume vs. Résumé

- Diacritics
  - résumé :: resume
  - naïve :: naive
  - café :: cafe
  - Québec
  - Zürich
  - Fédération Internationale de Football Association (FIFA)
- International languages
  - 基本上 सहायक αϭπαϭθ универсальной
  - ♪ ♬ ♩
  - ☺ ☹

Written Scripts

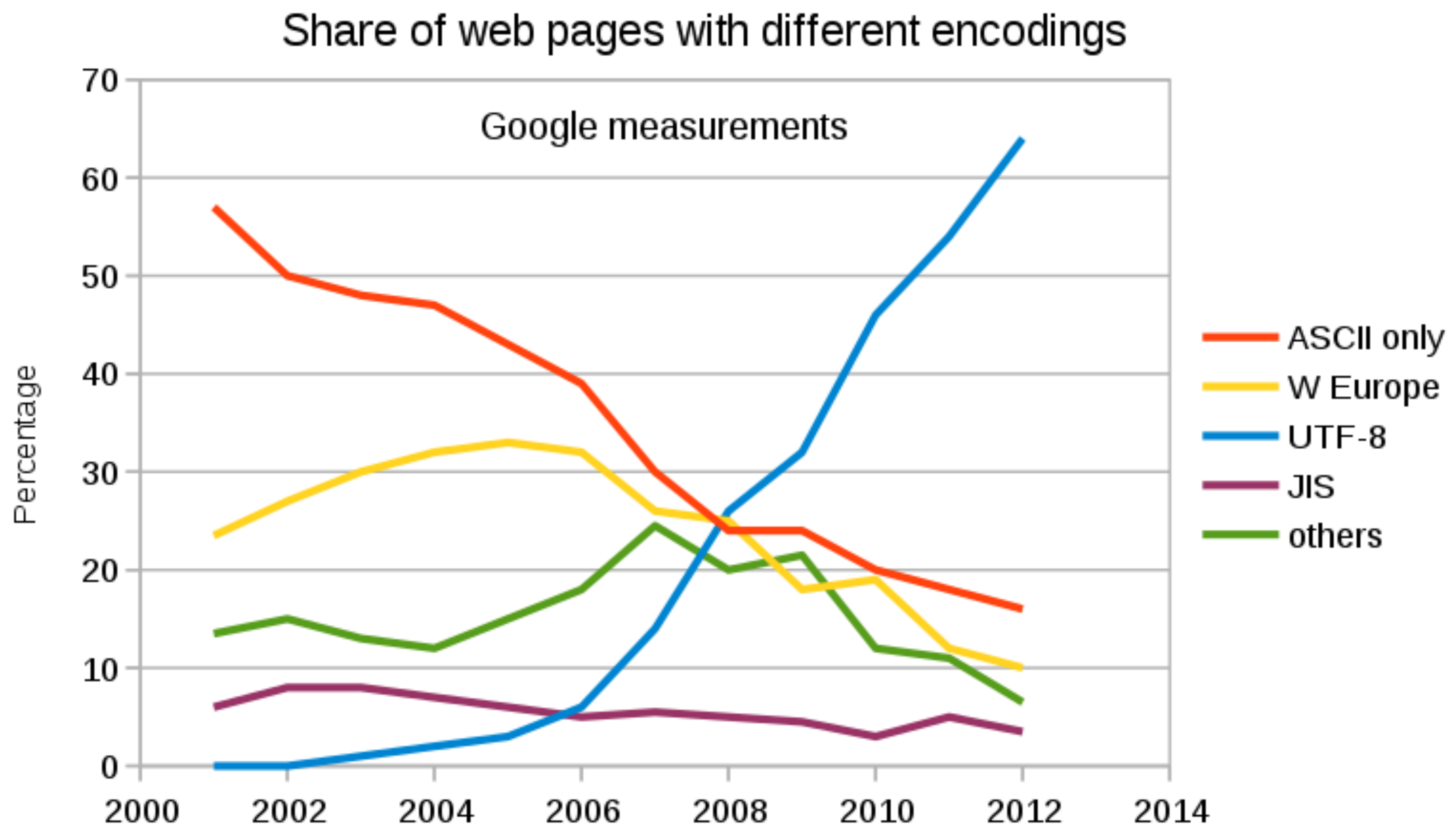


- Latin: 36% (2.6B people)
- Chinese: 18% (1.3B)
- Devanagari: 14% (1B)
- Arabic: 14% (1B)
- Cyrillic: 4% (0.3B)
- Dravidian: 3.5% (0.25B)

Other Character Encodings

- IBM EBCDIC
- Latin-1
- JIS: Japanese Industrial Standards
- CCCII: Chinese Character Code for Information Interchange
- EUC: Extended Unix Code
- Numerous other national standards

- Unicode and UTF-8


Share of web pages with different encodings
Google measurements
Percentage
0
10
20
30
40
50
60
70
2000
2002
2004
2006
2008
2010
2012
2014
ASCII only
W Europe
UTF-8
JIS
others

Unicode

- Industry standard for encoding and representing text
- Over 128,000 characters from 130+ scripts and symbol sets
- Can be implemented by different character endings
  - UTF-8: One byte to up to four bytes
  - UTF-16: One or two 16-bit code units
  - UTF-32: One 32-bit code unit

UTF-8

- Unicode Transformational Format – 8-bits
- Variable length encoding: One to four bytes
- Backward compatible with ASCII
  - One byte codes same as ASCII
- Dominant character encoding for the Web

- How to handle in Python?
  - Default in Python 3
  - In Python 2:

```
# -*- coding: utf-8 -*-
```

Let’s see an example: Résumé

Python 3

```
>>> text1="Résumé"
>>> len(text1)
6
>>> text1
'Résumé'

>>> [c for c in text1]
['R', 'é', 's', 'u', 'm', 'é']
```

Python 2

```
>>> text1="Résumé"
>>> len(text1)
8
>>> text1
‘R\xc3\xa9sum\xc3\xa9'

>>> [c for c in text1]
['R', '\xc3', '\xa9', 's', 'u', 'm', '\xc3', ‘\xa9']

>>> text2=u'Résumé'
>>> len(text2)
6
>>> text2
u'R\xe9sum\xe9'
>>> [c for c in text2]
[u'R', u'\xe9', u's', u'u', u'm', u'\xe9']
```

Take Home Concepts

- Diversity in Text
- ASCII and other character encodings
- Handling text in UTF-8