Python Odds & Ends

April 23, 2015
Today

• Web Interaction and Forms
• Graphical User Interfaces
• Natural Language Processing
Today

• Web Interaction and Forms
• Graphical User Interfaces
• Natural Language Processing
Basic Web Interaction, ~1995

http://www.weather.com/pvd

<HTML>
  <HEAD>
    ...
  </HEAD>
  ...
</HTML>
Basic Web Interaction, 2015

<HTML>
  <SCRIPT>
    javascript program here, which is executed
  </SCRIPT>
  <HEAD>
    Google Results
  </HEAD>
  <BODY>
    Text, text, text, text
  </BODY>
</HTML>

http://www.google.com/...
Interaction Through Forms

http://www.azlyrics.com/

Welcome to AZLyrics!

It's a place where all searches end!

We have a large, legal, every day growing universe of lyrics where stars of all genres and ages shine.
Fancier web Interaction


<HTML>
<HEAD>
Paul Simon
</HEAD>
<BODY>
Graceland...
</BODY>...
</HTML>
Details

• “Inspect Element” on the search box on www.azlyrics.com

• Look at the call to “search.azlyrics.com”
  – With input parameter “q” being the song name
Details

• “Inspect Element” on the search box on www.azlyrics.com

• Look at the call to “search.azlyrics.com”
  – With input parameter “q” being the song name
import urllib

url = 'http://search.azlyrics.com/search.php?'
params = {'q': 'Paul Simon Graceland'}

paramsEncode = urllib.urlencode(params)

remoteFile = urllib.urlopen(url + paramsEncode)
contents = remoteFile.read()
remoteFile.close()
Results!

Song results:
[1-1 of 1 total Songs found]

1. <a href="http://www.azlyrics.com/lyrics/paulsimon/graceland.html">Graceland</a> by <b>PAUL SIMON</b>

The Mississippi Delta was shining
Like a National guitar ...

Which is displayed as...
Results!

OK, now we grab the song using the link to it
Method

• Use lxml or equivalent to transform the string you read into an “XML Tree” and...
  – Use an xpath to grab the piece you want
  – Or grab a specific node, and then walk around its children

• Tougher for simple cases, really helpful for complex ones

• You can do this (Lecture 3-3)
Today

• Web Interaction and Forms
• Graphical User Interfaces
• Natural Language Processing
Graphical User Interfaces

• This is the way most of us *interact* with the computer
  – Windows containing text forms and buttons
  – Dialog boxes

• Not good for everything:
  – Say you want to convert files from PPT to PDF
    • But you have 10000 of them
    • Better make a Python program, right?

• But great for most cases
easygui

• Python module for... well... easy... GUIs

• To install on the MacOS
  – Open your “Terminal”
    • “sudo easy_install easygui”

• To install on Windows
  – Open your “Command Prompt”
    • “cd \Python27\Lib\site-packages
    • “.\easy_install easygui”
Basic Boxes

```
from easygui import *

# Message Box
msgbox("Hello World!")

# Enter Box
message = "Please give me any input"
title = "Title"
something = enterbox(message, title)

# Password Box
message = "Please give me your password"
title = "Title"
password = passwordbox(message, title)
```
Basic Boxes

Hello World!

Please give me any input

Please give me your password
A Fancier Way to Select Files

# File Box
message = "Please select a file"
title = "Title"

filetypes = ["*.txt", "*.htm", "*.html"]
filename = fileopenbox(message, title, ",*, filetypes)

print filename
A Fancier Way to Select Files
Choices & Questions

# Choice Box
choices = ["Yes", "No", "It's kind of ugly"]
reply = choicebox("Do you like easygui?", choices=choices)

print reply

# Continue/Cancel Box
message = "Do you want to continue?"
title = "Please Confirm"
if ccbox(message, title):
    print "Do actions for continue"
else:
    print "Do actions for cancel"
Choices & Questions
message = "Enter your personal information"
title = "Credit Card Application"
fieldNames = ["Name","Street Address","City","State","ZipCode"]
fieldValues = []
fieldValues = multenterbox(message ,title, fieldNames)

print "Here are the captured fields:", fieldValues
User Input Forms

Credit Card Application

Enter your personal information

Name
Street Address
City
State
Zip Code

OK
Cancel
message = "Enter your personal information"
title = "Credit Card Application"
fieldNames = ["Name","Street Address","City","State","ZipCode"]
fieldValues = []
fieldValues = multenterbox(message ,title, fieldNames)

print "Here are the captured fields:", fieldValues

But we still have to validate the data!
message = "Enter your personal information"
title = "Credit Card Application"
fieldNames = ["Name","Street Address","City","State","ZipCode"]
fieldValues = []
fieldValues = multenterbox(message ,title, fieldNames)

# Keep asking for things until all the fields have been filled
if fieldValues == None:
    done = True
else:
    done = False

while not done:
    # Check for empty fields
    someIsEmpty = False
    for i in range(len(fieldNames)):
        if fieldValues[i].strip() == "":
            someIsEmpty = True
    if someIsEmpty:
        fieldValues = multenterbox("Please fill all fields",
                                   title, fieldNames, fieldValues)
        if fieldValues == None:
            done = True
        else:
            done = True
    print "Here are the captured fields:", fieldValues
There are *so many* alternatives...

• Other GUI modules
  – PyGUI
  – WxPython
  – PyQt

• Some GUI modules use “native” windows from your operating system
  – That is, they look exactly as you’d expect
Today

• Web Interaction and Forms
• Graphical User Interfaces
• Natural Language Processing
Natural Language Processing

• You want to do “high level” work
  – Examine tone, setting, structure...
  – Use parts of speech to derive meaning

• You don’t want to lose time with details
  – Break words in German (tokenization)
  – Transform “running” into “run” (stemming)

  – These things are ready for you!
NLTK

• Natural Language ToolKit

• To install on the MacOS
  – Open your “Terminal”
    • “sudo easy_install nltk”

• To install on Windows
  – Open your “Command Prompt”
    • “cd \Python27\Lib\site-packages
    • “.\easy_install nltk”
The Brown Corpus

• Compiled in the 1960s
• 500 samples of English-language text
• About a million words
• Words are tagged with their part of speech
  – “JJ”: adjective
  – “NN”: singular or mass noun
  – “NNS$”: possessive plural form
• See http://en.wikipedia.org/wiki/Brown_Corpus
To install the *Brown Corpus* on NLTK

- You can install the Brown Corpus component in NLTK by:
  - In your python program, “import nltk”
  - Call “nltk.download()”
<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>Size</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>abc</td>
<td>Australian Broadcasting Commission 2006</td>
<td>1.4 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>alpino</td>
<td>Alpino Dutch Treebank</td>
<td>2.7 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>biocreative_ppi</td>
<td>BioCreAtIvE (Critical Assessment of Information Extr)</td>
<td>218.3 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>brown</td>
<td>Brown Corpus</td>
<td>3.2 MB</td>
<td>installed</td>
</tr>
<tr>
<td>brown_tei</td>
<td>Brown Corpus (TEI XML Version)</td>
<td>8.3 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>cess_cat</td>
<td>CESS-CAT Treebank</td>
<td>5.1 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>cess_esp</td>
<td>CESS-ESP Treebank</td>
<td>2.1 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>chat80</td>
<td>Chat-80 Data Files</td>
<td>18.8 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>city_database</td>
<td>City Database</td>
<td>1.7 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>cmudict</td>
<td>The Carnegie Mellon Pronouncing Dictionary (0.6)</td>
<td>875.1 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>comtrans</td>
<td>ComTrans Corpus Sample</td>
<td>11.4 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>conll2000</td>
<td>CONLL 2000 Chunking Corpus</td>
<td>738.9 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>conll2002</td>
<td>CONLL 2002 Named Entity Recognition Corpus</td>
<td>1.8 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>conll2007</td>
<td>Dependency Treebanks from CoNLL 2007 (Catalan and Spanish)</td>
<td>1.2 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>crubadan</td>
<td>Crubadan Corpus</td>
<td>5.0 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>dependency_treebank</td>
<td>Dependency Parsed Treebank</td>
<td>446.7 KB</td>
<td>not installed</td>
</tr>
</tbody>
</table>

Server Index: [http://www.nlTK.org/nltk_data/](http://www.nlTK.org/nltk_data/)

Download Directory: /Users/hmendes/nltk_data

Finished installing brown
Also install the **stopword** corpus
Also install the **wordnet** corpus
Install a *tokenizer*

• Allows to break up words and sentences
  – We did this for “nicely formatted English text”
  
  – In German, even “nicely formatted” text requires extra work to separate words
    • Since many words are “glued” together

• A tokenizer supports many other languages
Install a **tokenizer**

![NLTK Downloader screenshot](image)

- **Identifier**: basque_grammars, bllip_wsj_no_aux, book_grammars, hmm_treebank_pos_taggers, large_grammars, maxent_ne_chunker, maxent_treebank_pos_taggers, moses_sample
- **Name**: Grammars for Basque, BLLIP Parser: WSJ Model, Grammars from NLTK Book, Treebank Part of Speech Tagger (HMM), Large context-free and feature-based grammars for ACE Named Entity Chunker (Maximum entropy), Treebank Part of Speech Tagger (Maximum entropy), Moses Sample Models
- **Size**: 4.6 KB, 23.4 MB, 8.9 KB, 733.3 KB, 277.1 KB, 12.8 MB, 9.7 MB, 10.5 MB
- **Status**: not installed, not installed, not installed, not installed, not installed, not installed, not installed, not installed

- **Identifier**: punkt
- **Name**: Punkt Tokenizer Models
- **Size**: 13.0 MB
- **Status**: installed

**Server Index**: [http://www.nltk.org/nltk_data/](http://www.nltk.org/nltk_data/)

**Download Directory**: `/Users/hmendes/nltk_data`
Install a *part of speech* tagger

- Does what the name implies:
- Tags words with their part of speech
  - “JJ”: adjective
  - “NN”: singular or mass noun
  - “NNS$”: possessive plural form
  - ... (and many others)
Install a **part of speech** tagger

![Screenshot of NLTK Downloader with maxent_treebank_pos tagger installed](image)

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Name</th>
<th>Size</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>basque_grammars</td>
<td>Grammars for Basque</td>
<td>4.6 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>bllip_wsj_no_aux</td>
<td>BLLIP Parser: WSJ Model</td>
<td>23.4 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>book_grammars</td>
<td>Grammars from NLP Book</td>
<td>8.9 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>hmm_treebank_pos_tagger</td>
<td>Large context-free and feature-based grammars for Treebank Part of Speech Tagger (HMM)</td>
<td>733.3 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>large_grammars</td>
<td>ACE Named Entity Chunker (Maximum entropy)</td>
<td>277.1 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>maxent_ne_chunker</td>
<td>Treebank Part of Speech Tagger (Maximum entropy)</td>
<td>12.8 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>moses_sample</td>
<td>Moses Sample Models</td>
<td>10.5 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>punkt</td>
<td>Punkt Tokenizer Models</td>
<td>13.0 MB</td>
<td>installed</td>
</tr>
<tr>
<td>rslp</td>
<td>RSLP Stemmer (Remover de Sufixos da Lingua P)</td>
<td>3.7 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>sample_grammars</td>
<td>Sample Grammars</td>
<td>19.8 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>snowball_data</td>
<td>Snowball Data</td>
<td>6.5 MB</td>
<td>not installed</td>
</tr>
<tr>
<td>spanish_grammars</td>
<td>Grammars for Spanish</td>
<td>4.0 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>tagsets</td>
<td>Help on Tagsets</td>
<td>33.7 KB</td>
<td>not installed</td>
</tr>
<tr>
<td>universal_tagset</td>
<td>Mappings to the Universal Part-of-Speech Tagset</td>
<td>14.1 KB</td>
<td>not installed</td>
</tr>
</tbody>
</table>

**Server Index:** [http://www.nlTK.org/nltk_data/](http://www.nlTK.org/nltk_data/)

**Download Directory:** `/Users/hmendes/nltk_data`

**Finished installing maxent_treebank_pos_tagger**
import nltk

text = "Sarah Cynthia Sylvia Stout…"

wordList = nltk.word_tokenize(text)
sentenceList = nltk.sent_tokenize(text)
Tagging

As simple as that...

taggedList = nltk.pos_tag(wordList)
partOfSpeechElementsFound = {}
for (taggedWord, partOfSpeech) in taggedList:
    print taggedWord + " is a " + partOfSpeech
    partOfSpeechElementsFound[partOfSpeech] = 1

# Print help on the tagsets
print "KEYS:
for partOfSpeech in partOfSpeechElementsFound:
    nltk.help.brown_tagset(partOfSpeech)
Stemming and Lemmatization

• Words appear in many forms
  – In English, consider ‘talk’, ‘talked’, ‘talks’, ‘talking’
  – We want to stem these variations into ‘talk’

• Lemmatisation
  – More powerful approach to stemming
  – Operates with knowledge of context, based on a corpus
    • Can discriminate words that have a different meaning depending on their contextual part-of-speech
  – They are more accurate… but also more slower

(from the http://textminingonline.com)
from nltk.stem.porter import PorterStemmer

porter_stemmer = PorterStemmer()
porter_stemmer.stem('running')
porter_stemmer.stem('owed')

# See the result here, and then check the lemmatization below
porter_stemmer.stem('calculi')
Lemmatization

As simple as that...

```python
from nltk.stem import WordNetLemmatizer

wordnet_lemmatizer = WordNetLemmatizer()
wordnet_lemmatizer.lemmatize('running')
wordnet_lemmatizer.lemmatize('owed')

# Check the result now
wordnet_lemmatizer.lemmatize('calculi')
```
Want to do X in Python?

• Try googling X in Python
  – You’ll be surprised!