Textual Analysis & Introduction to Python

Oct 1 2015
Today’s Class

• Intro to text analysis problems
• Intro to Python
Text Analysis and Python

We’re starting a new unit in our course!
Textual Analysis

Define Problem

Find Data

Write a set of instructions

Solu4on

Computer

Solution
Textual Analysis

1. Define Problem
2. Find Data
3. Write a set of instructions
4. Solution

ACTACGTCGACTACGATCA
CGATCGCGCGATCACGTAT
TTACGATCAGCTACGATCG
ATCTACGATCGTAGCTGTG
Textual Analysis

Build a Concordance of a text
- *Locations* of words
- *Frequency* of words

Define Problem

Find Data

Write a set of instructions

Solution

Computer

ACTACGTGACTACGATCA
CGATCGCGCGGATCACGTAT
TTACGATCAGCTACGATCG
ATCTACGATCGTAGCTGTG
Concordances

Alphabetical index of all words in a text

<table>
<thead>
<tr>
<th>Word</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>4,7,10,27</td>
</tr>
<tr>
<td>Banana</td>
<td>77,110,130</td>
</tr>
<tr>
<td>Carrot</td>
<td>50,101</td>
</tr>
<tr>
<td>Date</td>
<td>9</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Concordances

• Before computers, was a huge pain.
• What texts might have had concordances?

http://en.wikipedia.org/wiki/Concordance_(publishing)
Concordances

• Before computers, was a huge pain.
• What texts might have had concordances?
  – The Bible
  – The Quran
  – The Vedas
  – Shakespeare
Concordances

• Before computers, was a huge pain.
• What texts might have had concordances?
  – The Bible
  – The Quran
  – The Vedas
  – Shakespeare

Not a “New” Problem: First Bible Concordance completed in 1230

http://en.wikipedia.org/wiki/Concordance_(publishing)
Concordances

• How long would the King James Bible take us?
  – 783,137 words

http://agards-bible-timeline.com/q10_bible-facts.html
Concordances

• How long would the King James Bible take us?
  – 783,137 words

\[
800,000 \times (3 \text{ min. to look up word and put page #}) = 2,400,000 \text{ minutes} \\
= 40,000 \text{ hours} \\
= 1,667 \text{ days} \\
= 4.5 \text{ years}
\]
Concordances

• How long would the King James Bible take us?
  – 783,137 words

800,000 * (3 min. to look up word and put page #) = 2,400,000 minutes
  = 40,000 hours
  = 1,667 days
  = 4.5 years

Takes 70 hours to read the King James Bible aloud

http://agards-bible-timeline.com/q10_bible-facts.html
Strong’s Concordance

- Concordance of the King James Bible
- Published in 1890 by James Strong

| ANT | Prv 6: 6 Go to the a, thou sluggard; consider her | Jas 1:21 Wherefore lay a all filthiness and |
| G5244 | G5259 |
| ANTICHRIST | 1Jn 2:18 as ye have heard that a shall come, even G500 | APELLES | Ro 16:10 Salute A approved in Christ. Salute them |
| | 22 is a, that denieth the Father and the Son. G500 | | G559 |
| | 4: 3 this is that spirit of a, whereof ye have G500 | | |
| | 2Jn 7 in the flesh. This is a deceiver and an a. G500 | | |
| ANTICHRISTS | 1Jn 2:18 are there many a; whereby we know that G500 | APES | 1Ki 10:22 and silver, ivory, and a, and peacocks. H6971 |
| | | | 2Ch 9:21 and silver, ivory, and a, and peacocks. H6971 |
| | | | |
| | | APHARSACHITES | Ezr 5: 6 companions the A, which were on this |
| | | | H671 |
| | | 6: 6 companions the A, which are beyond the |
| | | H671 |
| | | APHARSATICHITES | Ezr 4: 9 the Dinaites, the A, the Tarpelites, the |
| | | | H671 |
| | | APHARITES | Ezr 4: 9 the Tarpelites, the A, the Archevites, the |
| | | | H670 |
| | | | |
| | | APHEK | Jos 12:18 The king of A, one; the king of Lasharon, |
| | | | H663 |
| | | 13: 4 unto A, to the borders of the Amorites: |
| | | | H663 |
| | | 19:30 Ummah also, and A, and Rehob: twenty |
| | | | H663 |
| | | 1Sa 4: 1 and the Philistines pitched in A. |
| | | | H663 |
| | | 29: 1 all their armies to A; and the Israelites |
| | | | H663 |
| | | 1Ki 20:26 and went up to A, to fight against Israel. |
| | | | H663 |
| | | | |

CSCI 0931 - Intro. to Comp. for the Humanities and Social Sciences

http://www.cities.indiana.edu/academics/degree-programs/humanities/science/
From Concordance to Word Frequency

Suppose our text has 1000 words total.

<table>
<thead>
<tr>
<th>Word</th>
<th>Page Numbers</th>
<th># of Occurrences</th>
<th>Word Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>4,7,10,27</td>
<td>4</td>
<td>4/1000</td>
</tr>
<tr>
<td>Banana</td>
<td>77,110,130</td>
<td>3</td>
<td>3/1000</td>
</tr>
<tr>
<td>Carrot</td>
<td>50,101</td>
<td>2</td>
<td>2/1000</td>
</tr>
<tr>
<td>Date</td>
<td>9</td>
<td>1</td>
<td>1/1000</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>

CSCI 0931 - Intro. to Comp. for the Humanities and Social Sciences
Google Ngrams

• Google (verb) “Google n-grams”
• ngram: a set of $n$ words
  – “hello” is a 1-gram
  – “hello there” is a 2-gram

• Click on “Google Ngram viewer” for more information
• Question: what is the data source here?
Textual Analysis

Define Problem

Find Data

Write a set of instructions

Solution

Build a Concordance of a text
• *Locations* of words
• *Frequency* of words

• Word frequencies across time

ACTACGTCGACTACGATCA CGATCGCGCGATCACGTAT TTACGATCAGCTACGATCG ATCTACGATCGTAGCTGTG
The Wizard of OZ

• About 40 Books, written by 7 different authors

Lyman Frank Baum

Ruth Plumly Thompson

http://www.ssc.wisc.edu/~zzeng/soc357/OZ.pdf
The Wizard of OZ

• About 40 Books, written by 7 different authors

Lyman Frank Baum (1856-1919)

Published in 1921

Ruth Plumly Thompson

http://www.ssc.wisc.edu/~zzeng/soc357/OZ.pdf
The Wizard of OZ

- About 40 Books, written by 7 different authors

Lyman Frank Baum (1856-1919)

Ruth Plumly Thompson

Published in 1921

http://www.ssc.wisc.edu/~zzeng/soc357/OZ.pdf
The Federalist Papers

• 85 articles written in 1787 to promote the ratification of the US Constitution

• In 1944, Douglass Adair guessed authorship
  – Alexander Hamilton (51)
  – James Madison (26)
  – John Jay (5)
  – 3 were a collaboration

• Confirmed in 1964 by a computer analysis

Textual Analysis

Define Problem

Find Data

Write a set of instructions

Solution

Build a Concordance of a text
- Locations of words
- Frequency of words
  - Word frequencies across time
  - Determine authorship

ACTACGTGCAGCTACGATCA
CGATCGCGCGATCACGTAT
TTACGATCGCTACGATCG
ATCTACGATCGTAGCTGTG
Textual Analysis

Define Problem

Find Data

Write a set of instructions

Build a Concordance of a text

- Locations of words
- Frequency of words

- Word frequencies across time
  - Determine authorship
  - Count labels to determine liberal media bias

Solution

Computer

ACTACGTCGACTACGATCA
CGATCGCGCGATCACGTAT
TTACGATCAGCTACGATCG
ATCTACGATCGTAGCTGTG
How are we going to analyze texts?

Excel

Numerical Data
How are we going to analyze texts?

Excel

Numerical Data

Textual Data

firehow.com
How are we going to analyze texts?

Makita Cordless Chain Saw, $270
How are we going to analyze texts?

Python: A Programming Language
Free!

Textual Data

9poundhammer.blogspot.com
Textual Analysis

Define Problem

Find Data

Write a set of instructions

Solution

Python

Build a Concordance of a text

- **Locations** of words
- **Frequency** of words

- Word frequencies across time
  - Determine authorship
  - Count labels to determine liberal media bias

ACTACGTCGACTACGATCA
CGATCGGCGGATACGAT
TTACGATCAGCTACGATCG
ATCTACGATCGTAGCTGTG
“Python”
“Python”

- A language for giving the computer instructions. It has syntax and semantics.
“Python”

• **A language** for giving the computer instructions. It has syntax and semantics.

• Might say “write a Python program”, meaning “write instructions in the Python language”
“Python”

- **A language** for giving the computer instructions. It has syntax and semantics.

- Might say “write a Python program”, meaning “write instructions in the Python language”

- There is an **interpreter** (e.g., IDLE) that takes Python instructions and executes them with the CPU, etc.
Install

- **Let’s install** Python 3.4.x
- [www.python.org/downloads/](http://www.python.org/downloads/)
Install – Mac OS X

• On Mac OS X double click the pkg file you downloaded. Follow the instructions by agreeing and click next.
Install - Windows

[Image of Python 3.4.3 Setup window showing the option to add `python.exe to Path` and its installation options.]
Let’s open IDLE

• On Mac OS X open a terminal window
  Type: idle3

• On Windows
  Start ->
  All Programs ->
  Python ->
  IDLE
Introduction to Python

• **Expressions** are *inputs* that Python evaluates
  – Expressions return an *output*
  – Like using a *calculator*

1. **Expressions**
2. **Assignments**
   a) Variables
3. **Types**
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Expressions** are *inputs* that Python evaluates
  – Expressions return an *output*
  – Like using a **calculator**

Type the expressions below after ‘>>>’ and hit Enter

```
>>> 4+2
6
>>> 4-2
2
>>> 4*2
8
>>> 4/2
2.0
```
Introduction to Python

• **Assignments** do not have an output, they are *stored in memory.*
Introduction to Python

• **Assignments** do not have an output, they are stored in memory.

  – We’ve done this kind of thing with spreadsheets

  ![Spreadsheet Example](example.png)

  We have *assigned* the number 1 to cell A1.

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Assignments** do not have an output, they are stored in memory.
  – We’ve done this kind of thing with spreadsheets

![Spreadsheet Image]

We have *assigned* the number 1 to cell A1.

Let’s rename cell A1 to *x*.

![Spreadsheet Image]

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Assignments** do not have an output, they are stored in memory.
  – We’ve done this kind of thing in Spreadsheets

```python
>>> x = 1
```

Let’s rename cell A1 to x.

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Assignments** do not have an output, they are *stored in memory.*

  – We’ve done this kind of thing in Spreadsheets

```
>>> x = 1
variable = value
```

<table>
<thead>
<tr>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable Name</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
<tr>
<td>x</td>
</tr>
</tbody>
</table>
Assignment do not have an output, they are stored in memory.

- We’ve done this kind of thing in Spreadsheets

\[ \text{>>> } x = 1 \]

- We can now use \( x \) in expressions!

\[ \text{>>> } x+1 \]
\[ 2 \]
\[ \text{>>> } (x+2)*3 \]
\[ 9 \]

**Memory**

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x )</td>
<td>1</td>
</tr>
</tbody>
</table>
Introduction to Python

• You can name your variables anything
  ```
  >>> numberOfEggs = 100
  >>> myNumber = 12345
  >>> noninteger = 4.75
  ```

• Well, *almost* anything
  – No spaces, operators, punctuation, number in the first position

• Variables usually start with a lowercase letter and, if useful, describe something about the value.

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Choices

• Why are *those* the rules for names?
• Someone thought about it and made a choice
• Usually based on years of experience
• Many choices seem crazy...
  – Until one day you see they’re obviously correct
Introduction to Python

• Try this: \[
>>> \frac{3}{2}
\]

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• Try this: `>>> 3/2`

• There are *two* types of numbers in Python. The `type()` function is useful.

```
>>> type(3)
<class 'int'>
>>> type(3/2)
<class 'float'>
```

---

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• Try this: >>> 3/2

• There are two types of numbers in Python. The type() function is useful.

  >>> type(3)
  <class 'int'>
  >>> type(3/2)
  <class 'float'>

• Floats are numbers that display with decimal points.

  >>> 3.0/2.0
  1.5

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• Try this: >>> 3/2

• There are two types of numbers in Python. The type() function is useful.

>>> type(3)
<class 'int'>
>>> type(1.5)
<class 'float'>

• Floats are decimals.

>>> 3.0/2.0
1.5

General Rule: Expressions for a particular type will output that same type! Except for the division operator (/)

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

- **Strings** are sequences of characters, surrounded by single quotes.

```python
>>> 'hi'
'hi'
>>> myString = 'hi there'
>>> myString
'hi there'
```
Introduction to Python

- **Strings** are sequences of characters, surrounded by single quotes.

  ```python
  >>> 'hi'
  'hi'
  >>> myString = 'hi there'
  >>> myString
  'hi there'
  ```

- The `+` operator concatenates

  General Rule: Expressions for a particular type will *output* that same type!

1. **Expressions**
2. **Assignments**
   a) Variables
3. **Types**
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Strings** are sequences of characters, surrounded by single quotes.

```python
>>> 'hi'
'hi'
>>> myString = 'hi there'
>>> myString
'hi there'
```

• The `+` operator concatenates

```python
>>> endString = ' class!'
>>> myString + endString
'hi there class!'
>>> newString = myString + endString
>>> newString
'hi there class!'
```
Introduction to Python

• **Lists** are an ordered collection of items

```python
>>> [5,10,15]
[5, 10, 15]
>>> myList = [5,10,15]
>>> myList
[5, 10, 15]
>>> stringList = ['hi','there','class']
>>> stringList
['hi', 'there', 'class']
```

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• **Lists** are an ordered collection of items

>>> [5, 10, 15]
[5, 10, 15]

>>> myList = [5, 10, 15]
>>> myList
[5, 10, 15]

>>> stringList = ['hi', 'there', 'class']
>>> stringList
['hi', 'there', 'class']

• Individual items are *elements*

• The + operator concatenates

>>> myList + stringList
[5, 10, 15, 'hi', 'there', 'class']

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
d) Lists
Introduction to Python

• To get an element from a list, use the expression `>>> myList[i]` where `i` is the index. Often spoken: “myList sub i”

• List indices start at 0!

```
>>> myList[0]
5
>>> myList[1]
10
>>> myList[2]
15
```

• What does `>>> myList[1] = 4` do?

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
Introduction to Python

• To get a range of elements from a list, use the expression `>>> myList[i:j]` where `i` is the start index (inclusive) and `j` is the end index (exclusive).

```python
>>> myList
[5, 4, 15]
>>> myList[0:2]
[5, 4]
>>> myList[1:3]
[4, 15]
>>> newList = [2,5,29,1,9,59,3]
>>> newList
[2, 5, 29, 1, 9, 59, 3]
>>> newList[2:6]
[29, 1, 9, 59]
```
Introduction to Python

• **Indexing and ranges** also work on Strings.

```python
>>> myString
'hi there'
>>> myString[0]
'h'
>>> myString[5]
'e'
>>> myString[6]
'r'
>>> myString[0:6]
'hi the'
```

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) **Strings**
   d) Lists
Introduction to Python

• Remember what assignments do

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1</td>
</tr>
<tr>
<td>amountOfEggs</td>
<td>100</td>
</tr>
<tr>
<td>myNumber</td>
<td>12345</td>
</tr>
<tr>
<td>noninteger</td>
<td>4.75</td>
</tr>
<tr>
<td>myString</td>
<td>'hi there'</td>
</tr>
<tr>
<td>endString</td>
<td>' class!'</td>
</tr>
<tr>
<td>myList</td>
<td>[5, 4, 15]</td>
</tr>
<tr>
<td>stringList</td>
<td>['hi', 'there', 'class']</td>
</tr>
<tr>
<td>newList</td>
<td>[2, 5, 29, 1, 9, 59, 3]</td>
</tr>
</tbody>
</table>

1. Expressions
2. Assignments
   a) Variables
3. Types
   a) Integers
   b) Floats
   c) Strings
   d) Lists
1. Expressions
   • Evaluate *input* and returns some *output* (calculator)
2. Variable Assignments: `<variable> = <expression>`
   • Store the value of the expression in the variable instead of outputting the value.
   • There is *always* an equals sign in an assignment
   • Variables can be named many things
   • List assignments: `<listvar>[<index>] = <expression>`
3. Types
   • Integers vs. Floats (Decimals)
   • Strings in single quotes
   • Lists are sets of other types
   • We can index into Strings & Lists
   
**Expressions for a particular type will output that same type!** Floats have a higher priority.
A brief review of things you didn’t know you’d learned

• In a spreadsheet, there are many **types** of data
• Numbers (start with +/- or a digit)
• Strings (nondigit-start, or start with ‘ ’)
• Formulas (start with = )
• Ranges (B2, B2:B4, B2:D5)
• Errors (#N/A)
• Blanks
What shows up in a cell

• If a formula evaluates to a number or string, that number or string
• If it evaluates to a range, the value in the first cell of that range ...sometimes
  – If you write =A1:A6, you get A1
  – If you write =OFFSET(A1:A6, 0, 0), Gsheets fills in adjacent cells; excel just fills in one cell
• If evaluation leads to an error, then #N/A
• Mostly, we never notice any of this
• In Python, the rules have greater consistency, and because results aren’t instantly visible, knowing the rules matters more