Introduction to Python Dictionaries

March 12, 2013

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

Word Frequency: Inputs and Outputs

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

Word Frequency: Inputs and Outputs

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

- What is the input to `wordFreq`?

Word Frequency: Inputs and Outputs

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?

Word Frequency: Inputs and Outputs

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?

Word	Freq.
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Word Frequency: Inputs and Outputs

```
The cat had a hat. The cat sat on the hat.
```

I want to write a `wordFreq` function

- What is the input to `wordFreq`?
- What is the output of `wordFreq`?
- We *could* do this with a list. How?

Word	Freq.
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

A New Data Structure

- A *Data Structure* is simply a way to store information.
  - **Lists** are a type of data structure
    - We can have **lists** of integers, floats, strings, booleans, or any combination.
    - Organized **linearly** (indexed by a range of integers)

A New Data Structure

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

A New Data Structure

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

A New Data Structure

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

Keys

Values

A New Data Structure

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

Associate each **word** with the **frequency**.

Keys

Values

Key-Value Pairs

Key	Value
‘the’	3
‘cat’	2

A New Data Structure: Dictionaries

Keys can be *almost any* type or data structure.
Values can be *any* type or data structure.

Key Type	Value Type	Example Key	Example Value

A New Data Structure: Dictionaries

Keys can be *almost any* type or data structure.
Values can be *any* type or data structure.

Key Type	Value Type	Example Key	Example Value
`String`	`Integer`	`'the'`	`3`
`String`	`Integer`	`'cat'`	`2`

A New Data Structure: Dictionaries

Keys can be *almost any* type or data structure.
Values can be *any* type or data structure.

Key Type	Value Type	Example Key	Example Value
`String`	`Integer`	`'the'`	`3`
`String`	`Integer`	`'cat'`	`2`
`String`	`String`	`'Geisel'`	`'078-05-1120'`
`String`	`String`	`'Whitcher'`	`'552-38-5014'`

A New Data Structure: Dictionaries

Keys can be *almost any* type or data structure.
Values can be *any* type or data structure.

Key Type	Value Type	Example Key	Example Value
`String`	`Integer`	`'the'`	`3`
`String`	`Integer`	`'cat'`	`2`
`String`	`String`	`'Geisel'`	`'078-05-1120'`
`String`	`String`	`'Whitcher'`	`'552-38-5014'`
`Float`	`String`	`1.0`	`'one point oh'`
`Float`	`String`	`2.8`	`'two point eight'`

A New Data Structure: Dictionaries

Keys can be *almost any* type or data structure.
Values can be *any* type or data structure.

Key Type	Value Type	Example Key	Example Value
String	Integer	'the'	3
String	Integer	'cat'	2
String	String	'Geisel'	'078-05-1120'
String	String	'Whitcher'	'552-38-5014'
Float	String	1.0	'one point oh'
Float	String	2.8	'two point eight'
Integer	List	1638	[2, 3, 3, 7, 13]

A New Data Structure: Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}
>>> freq
{}
>>>
```

Initialize a Dictionary

A New Data Structure: Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}
>>> freq
{}
>>> freq['the'] = 3
>>> freq
{'the': 3}
>>>
```

Initialize a Dictionary

Key = 'the'
Value = 3

A New Data Structure: Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq = {}
>>> freq
{}
>>> freq['the'] = 3
>>> freq
{'the': 3}
>>> freq['cat'] = 2
>>> freq
{'the': 3, 'cat': 2}
>>>
```

Initialize a Dictionary

Key = 'the'
Value = 3

Key = 'cat'
Value = 2

A New Data Structure: Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq2 = {'the':3,'cat':2}
>>> freq2
{'the': 3, 'cat': 2}
>>>
```

Initialize a dictionary
with two key-value pairs

A New Data Structure: Dictionaries

```
The cat had a hat. The cat sat on the hat.
```

Word	Frequency
the	3
cat	2
had	1
a	1
hat	2
sat	1
on	1

```
>>> freq2 = {'the':3,'cat':2}
>>> freq2
{'the': 3, 'cat': 2}
>>>
>>> freq2['cat']
2
>>> freq2['the']
3
```

Retrieve a **value** using the **key**

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

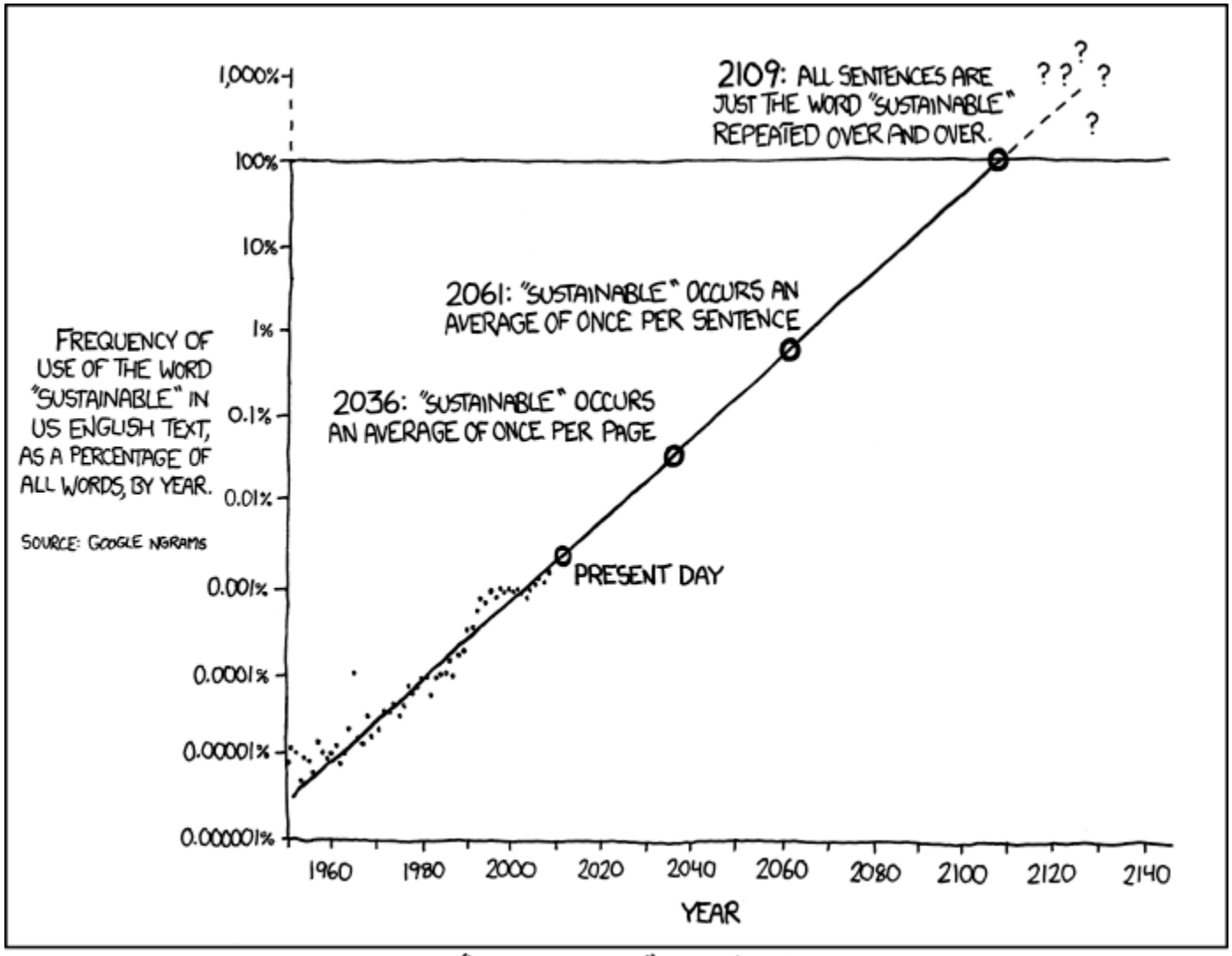
Python Dictionaries

Function (All operate on Dictionaries)	Input	Output	Example
`keys()`	None	List of keys	`>>> freq2.keys()` `['the', 'cat']`
`values()`	None	List of values	`>>> freq2.values()` `[3, 2]`
`has_key(<key>)`	Key	Boolean	`>>> freq2.has_key('missing')` `False`
`<key> in <dict>`	(means same as above)		`>>> 'the' in freq2` `True`
`del(<dict>[<key>])`	Dict. Entry	None	`>>> del(freq2['cat'])`

Do Task 1

- Keys Are Unique!
- Dictionaries are Unordered! Unlike lists, which are linear.

Break



http://xkcd.com/1007/

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

Python Dictionaries

Function (All operate on Dictionaries)	Input	Output	Example
`keys()`	None	List of keys	`>>> freq2.keys()` `['the', 'cat']`
`values()`	None	List of values	`>>> freq2.values()` `[3, 2]`
`has_key(<key>)`	Key	True or False	`>>> freq2.has_key('missing')` `False`
`<key> in <dict>`	(means same as above)		`>>> 'the' in freq2` `True`
`del(<dict>[<key>])`	Dict. Entry	None	`>>> del(freq2['cat'])`

- Keys Are Unique!
- Dictionaries are Unordered! Unlike lists, which are linear.

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

Building a Concordance

```
The cat had a hat. The cat sat on the hat.
 0   1   2  3  4    5   6   7   8  9   10
```

Word	List of Positions	Frequency
the	[0,5,9]	3
cat	[1,6]	2
had	[2]	1
a	[3]	1
hat	[4,10]	2
sat	[7]	1
on	[8]	1

The Big Picture

Overall Goal

Build a Concordance of a text

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

Today: Get Word Frequencies

- Define the inputs and the outputs
- Learn a new *data structure*
- Write a function to get word frequencies
- Go from word frequencies to a concordance (finally!)

This is part of your HW