Determining Authorship

Mar 22, 2016

Working with Strings

Check out the documentation...

- The `str` type has lots of great member functions:
  - `find()`
  - `replace()`
  - `strip()`, `lstrip()`, `rstrip()`
  - `join()` — the opposite of `split()`

Working with Strings

Check out the documentation...

- The `str` type has lots of great member functions:
  - `find()`
  - `replace()`
  - `strip()`, `lstrip()`, `rstrip()`
  - `join()` — the opposite of `split()`

find()

- Finds the first position of a word in a text

- Can start looking at some position (inclusive), stop at another position (exclusive)
  - *Optional* arguments!

```
>>> mobyString.find('me')
5
>>> mobyString.find('me', 7)
20
>>> mobyString.find('me', 22, len(mobyString))
139
```

replace()

- Replaces all occurrences of one string by another
- Can specify the maximum number of substitutions to be made
  - *Optional* argument!

Try these two:

```
>>>  mobyString.replace('I', 'YOUR-LOYAL-CS931-TEACHER’)
>>>  mobyString.replace('I', 'YOUR-LOYAL-CS931-TEACHER', 6)
```

strip(), lstrip(), rstrip()

- Removes whitespace at start and end of the string
  - `lstrip`() does that only for the *start* of the string
  - `rstrip`() does that only for the end of the string

- You can specify the string to be stripped as an optional argument (defaults to whitespace)

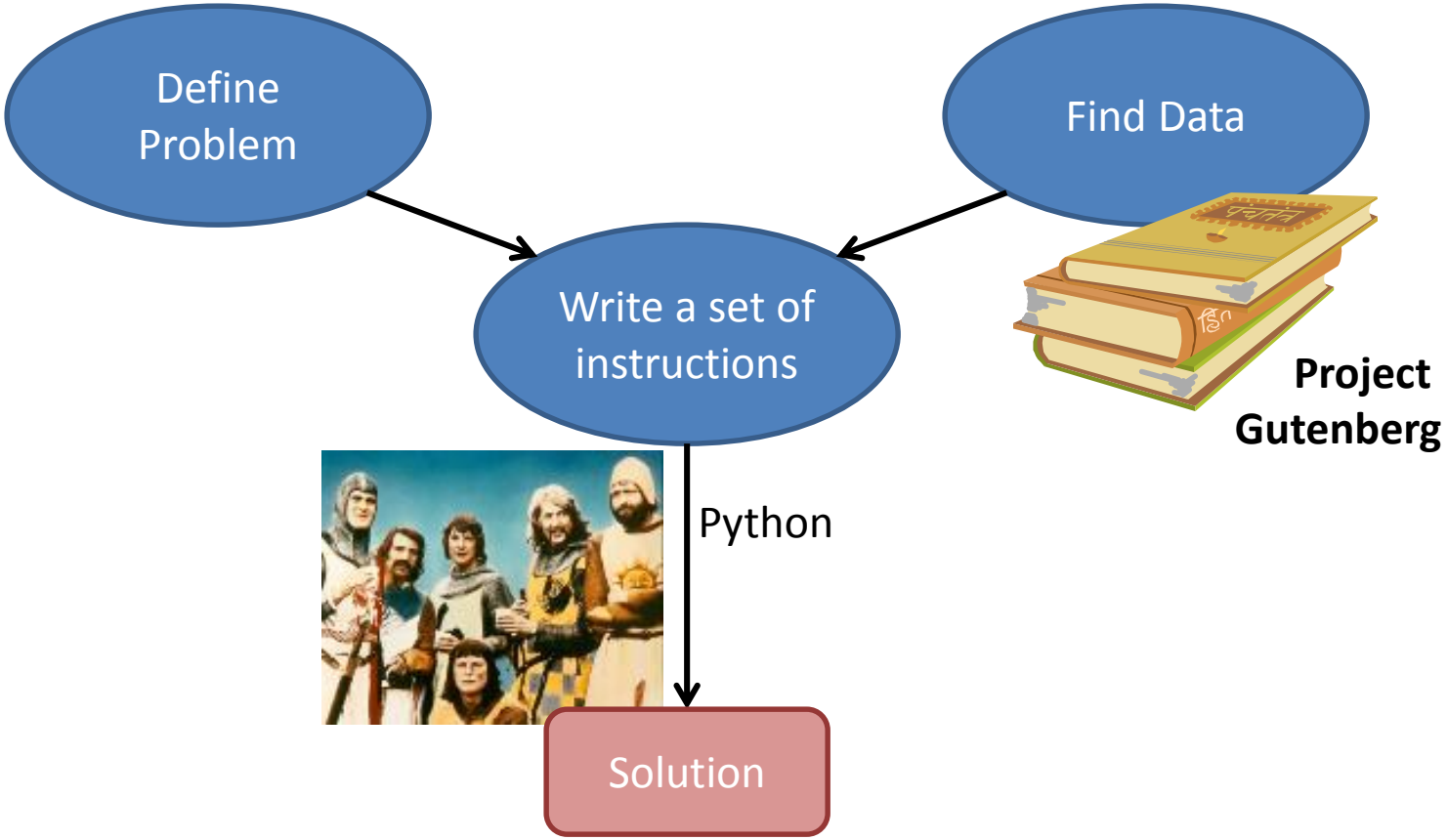
join()

- Joins a list of strings through the specified delimiter (on which the function is called)
  - If the list of words is a single string, the function treats that string as a list of characters (as usual)

Try these two:

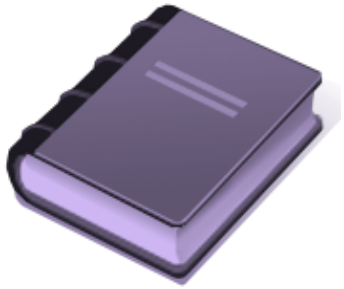
```
>>> delim = ':'
>>> delim.join(['a', 'b', 'c'])
'a:b:c'
>>> delim.join('word')
'w:o:r:d'
```

Determining Authorship



Determining Authorship: Data

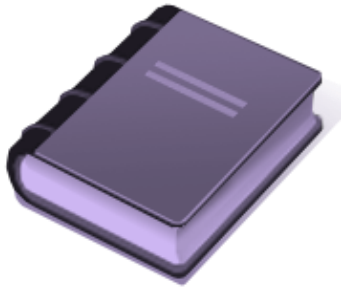
Five Books from a Famous Children's Series



One Book from a Famous Children's Series

Determining Authorship: Data

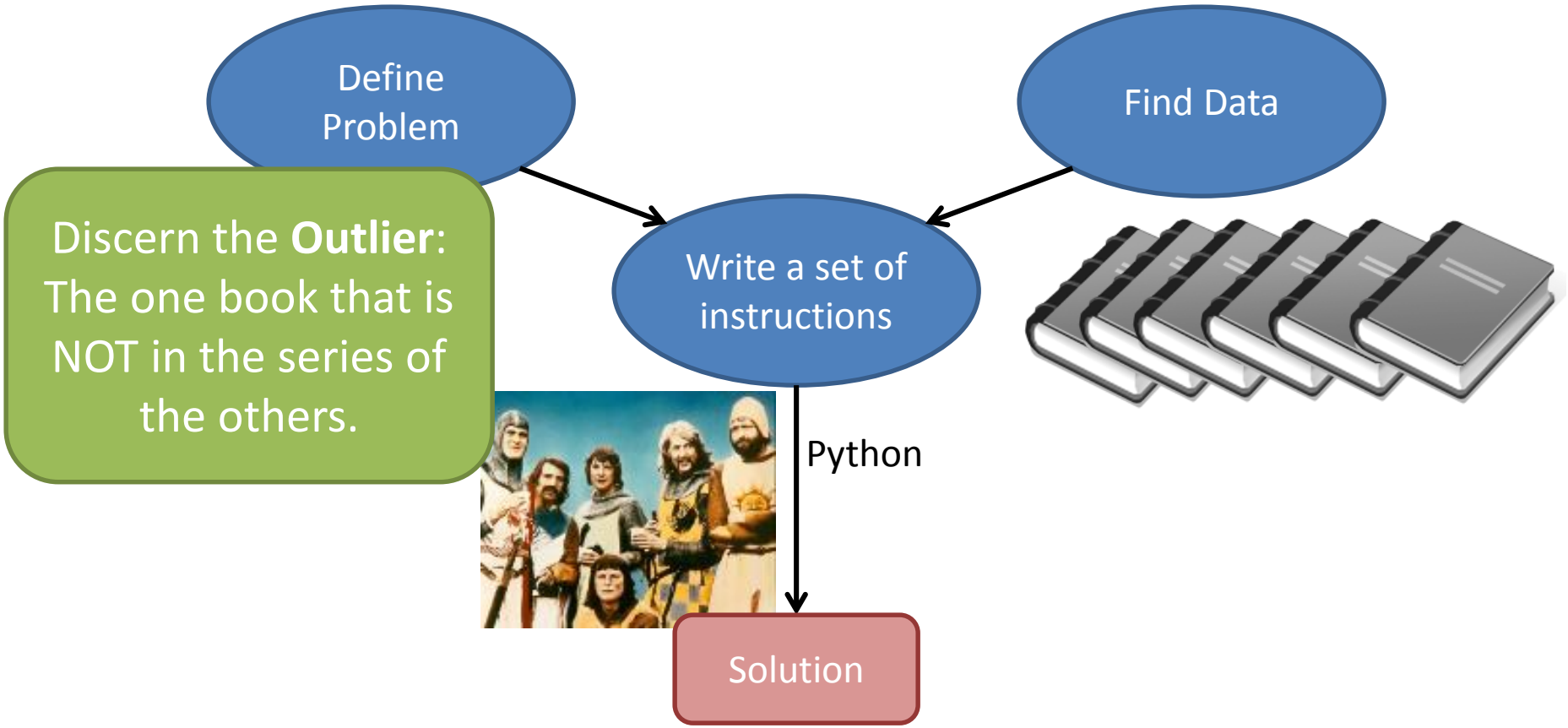
Five Books from a Famous Children's Series



One Book from a Famous Children's Series

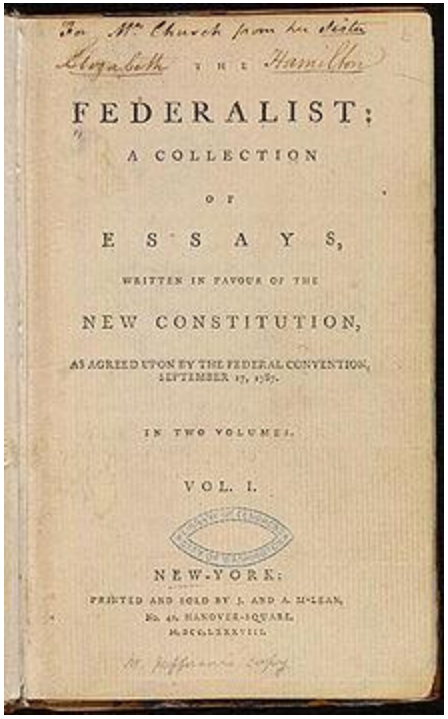
Six Books from Two Famous Children's Series

Determining Authorship



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
- Corroborated in 1964 by a computer analysis



Wikipedia

http://pages.cs.wisc.edu/~gfung/federalist.pdf

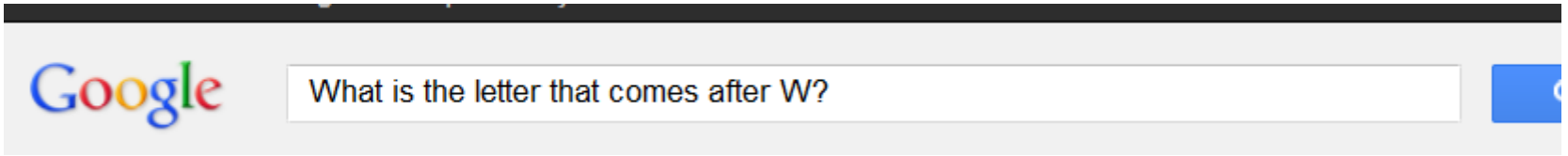
Determining Authorship

Discern the **Outlier**: The one book that is NOT in the series of the others.

vs.

Stop Words

Stop Words are words that are filtered out in natural language processing



Stop Words

Stop Words are words that are filtered out in natural language processing

Stop Words

Stop Words are words that are filtered out in natural language processing

a, able, about, across, after, all, almost, also, am, among, an, and, any, are, as, at, be, because, been, but, by, can, cannot, could, dear, did, do, does, either, else, ever, every, for, from, get, got, had, has, have, he, her, hers, him, his, how, however, i, if, in, into, is, it, its, just, least, let, like, likely, may, me, might, most, must, my, neither, no, nor, not, of, off, often, on, only, or, other, our, own, rather, said, say, says, she, should, since, so, some, than, that, the, their, them, then, there, these, they, this, tis, to, too, twas, us, wants, was, we, were, what, when, where, which, while, who, whom, why, will, with, would, yet, you, your

http://www.textfixer.com/resources/common-english-words.txt

Stop Words

Stop Words are words that are filtered out in natural language processing

a, able, about, across, after, all, almost, also, am, among, an, and, any, are, as, at, be, because, been, but, by, can, cannot, could, dear, did, do, does, either, else, ever, e ve, he, her, hers, him, his, how, however, ike, likely, may, me, might, most, must n, on, only, or, other, our, own, rather, s ome, than, that, the, their, them, then, t as, us, wants, was, we, were, what, when, where, which, while, who, whom, why, will, with, would, yet, you, your

Why should we look at the frequencies of stop words?

http://www.textfixer.com/resources/common-english-words.txt

Determining Authorship

Discern the **Outlier**: The one book that is NOT in the series of the others.

1

vs.

2

1. Calculate the word counts of the stop words in the two books

	a	able	about	across	after	...
File 1	1000	238	483	12	3	...
File 2	102	93	10	0	15	...

Determining Authorship

Discern the **Outlier**: The one book that is NOT in the series of the others.

1

vs.

2

1. Calculate the word counts of the stop words in the two books
2. Normalize to get word frequencies

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

Determining Authorship

1. Calculate the word counts of the stop words in the two books
2. Normalize to get word frequencies

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

3. Design a **metric** to compare the two files
   - A metric is a function that defines a **distance** between two things

Determining Authorship

1. Calculate the word counts of the stop words in the two books
2. Normalize to get word frequencies

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

3. Design a **metric** to compare the two files
   - A metric is a function that defines a **distance** between two things

Write a
`compareTwo(list1,list2)`
function that returns a `float`.

Determining Authorship

1. Calculate the word counts of the sto
2. Normalize to get word frequencies

	a	able	about	a		
File 1	.3	.01	.003	.0027		
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

In groups, do Task 1

3. Design a **metric** to compare the two files
   - A metric is a function that defines a **distance** between two things

Write a `compareTwo(list1,list2)` function that returns a `float`.

Determining Authorship

Download and extract `ACT2-7.zip`

Evaluate and run `testFiles('output.csv')`

Determining Authorship

Download and extract `ACT2-7.zip`

Evaluate and run `testFiles('output.csv')`

We are going to modify two things:

- `compareTwo` function
- Write distance matrix to a file

Determining Author

Download and extract `ACT2-7.zip`

Evaluate and run `testFiles('output.csv')`

We are going to modify two things:

- `compareTwo` function
- Write distance matrix to a file

Determining Author

Download and extract `ACT2-7.zip`

Evaluate and run `testFiles('output.csv')`

We are going to modify two things:

- **`compareTwo` function**
- Write distance matrix to a file

Frequency table in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

frequencies

```
[
        [.3,    .1,     .003,   .0027,  .0006,…],
        [.238,  .0932,  .0034,  .0021,  .005 ,…],
        …
        …
        …
        …
]
```

Frequency table in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

frequencies[0]

```
[
        [.3,    .1,     .003,   .0027, .0006,…],
        [.238, .0932, .0034, .0021, .005 ,…],
        …
        …
        …
        …
]
```

Frequency table in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

frequencies[1]

```
[
        [.3,    .1,     .003,   .0027,  .0006,…],
        [.238,  .0932,  .0034,  .0021,  .005 ,…],
        …
        …
        …
        …
]
```

Frequency table in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

```
[
        [.3,    .1,     .003,   .0027,  .0006,…],
        [.238,  .0932,  .0034,  .0021,  .005 ,…],
        …
        …
        …
        …
]
```

frequencies[1][0]

Frequency table in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

```
[
        [.3,    .1,     .003,   .0027,  .0006,…],
        [.238,  .0932,  .0034,  .0021,  .005 ,…],
        …
        …
        …
        …
]
```

frequencies[1][1]

Your task in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

```
[
        [.3,    .1,     .003,  .0027, .0006,…],   i
        [.238, .0932, .0034, .0021, .005 ,…],   j
        …
        …
        …
        …
]
```

Your task in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

```
[
        [.3,    .1,     .003,   .0027, .0006,…],   i
        [.238, .0932,  .0034, .0021, .005 ,…],   j
        …
        …
        …
        …
]
```

Your task in `compareTwo()`

We're representing this:

	a	able	about	across	after	...
File 1	.3	.01	.003	.0027	0.006	...
File 2	0.238	0.0932	0.0034	0.0021	0.05	...

As a list of lists!

```
[
        [.3,    .1,     .003,  .0027, .0006,…],   i
        [.238, .0932, .0034, .0021, .005 ,…],   j
        …
        …
        …
        …
]
```

Determining Author

Download and extract `ACT2-7.zip`

Evaluate and run `testFiles('output.csv')`

We are going to modify two things:

- **`compareTwo` function**
- Write distance matrix to a file

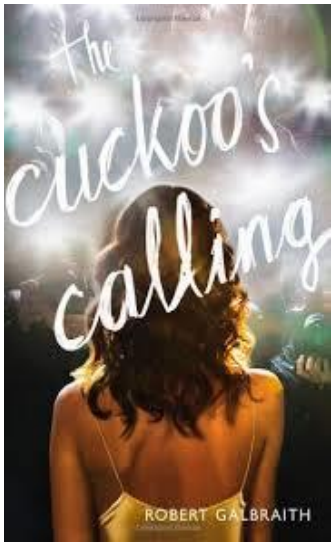
One way of doing it

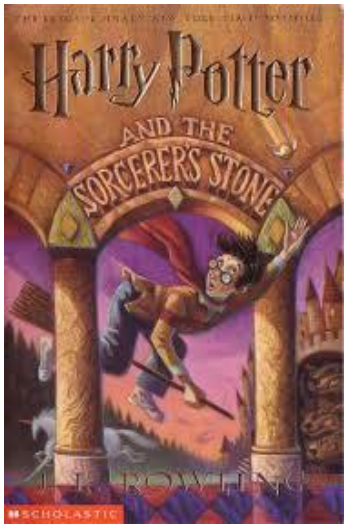
len of frequencies[i] == *len* of any other frequencies[j] == *len* of word list

```
val = 0.0

for word in range(0, len(frequencies[i])):
        freqsI = frequencies[i]
        freqsJ = frequencies[j]
        val = val + abs(freqsI[word] - freqsJ[word])

    return val
```

“How a Computer Program Helped Reveal J. K. Rowling as Author of *A Cuckoo’s Calling*”

http://www.scientificamerican.com/article/how-a-computer-program-helped-show-jk-rowling-write-a-cuckoos-calling/





Run `testFiles('output.csv')`

This matrix looks kind of familiar...

Distance Matrix

This matrix looks kind of familiar...

Instead of printing to the screen, write it to a file in CSV (comma-separated value) format.

```
myNum = 1
myFile = open('output.csv','w')
myFile.write('this is an output file\n')
myFile.write(str(myNum))
myFile.write('\n')
myFile.close()
```

Distance Matrix

This matrix looks kind of familiar...

Instead of printing to the screen, write it to a file in CSV (comma-separated value) format.

```
myNum = 1
myFile = open('output.csv','w')
myFile.write('this is an output file\n')
myFile.write(str(myNum))
myFile.write('\n')
myFile.close()
```

```
this is an output file
1
```

Distance Matrix

This matrix looks kind of familiar...

Instead of printing to the screen, write it to a file in CSV (comma-separated value) format.

```
myNum = 1
myFile = open('output.csv','w')
myFile.write('this is an output file\n')
myFile.write(str(myNum))
myFile.write('\n')
myFile.close()
```

```
this is an output file
1
```

Generating the file

```
#For each file, create a string row with all the values in the
#corresponding list row in distMatrix, with commas in between
for i in range(0,len(FILE_LIST))
     row = '' + FILE_LIST[i]

     # Loop through the columns in the current list row
     for val in distMatrix[i]:
          row = row + ',' + str(val)

     #At this point, we created our string row.
     #We want to write this row into our csv
     outFile.write(row)

     #Need a newline at the end of each string row
     outFile.write('\n')

# Finalize the new file by closing it
outFile.close()
```

Distance Matrix

This matrix looks kind of familiar...

Instead of printing to the screen, write it to a file in CSV (comma-separated value) format.

Open the CSV file in Google Spreadsheets. Use conditional formatting to look for patterns

What's Your Answer?

Discern the **Outlier**:
The one book that is NOT in the series of the others.

File	Title	Series	Author
file1.txt			
file2.txt			
file3.txt			
file4.txt			
file5.txt			
file6.txt			

What's Your Answer?

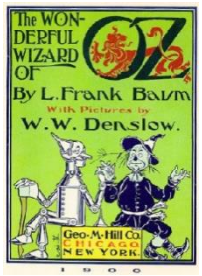
Discern the **Outlier**:
The one book that is NOT in the series of the others.

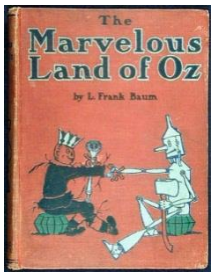
File	Title	Series	Author
file1.txt	Wonder Wizard of Oz	Oz	
file2.txt	Alice's Adventures in Wonderland	Alice in Wonderland	
file3.txt	Dorothy and the Wizard in Oz	Oz	
file4.txt	Emerald City of Oz	Oz	
file5.txt	Royal Book of Oz	Oz	
file6.txt	Glinda of Oz	Oz	

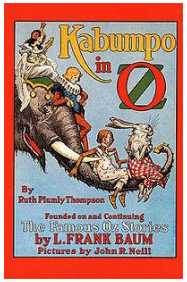
The Wizard of OZ

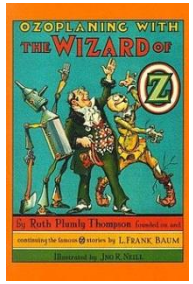
- About 40 Books, written by 7 different authors

#1 ... #14 #15 #16 ... #33









Lyman Frank Baum
(1856-1919)

Ruth Plumly Thompson

Published in 1921

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

What's Your Answer?

Discern the **Outlier**:
The one book that is NOT in the series of the others.

File	Title	Series	Author
file1.txt	Wonder Wizard of Oz	Oz	Lyman Frank Baum
file2.txt	Alice's Adventures in Wonderland	Alice in Wonderland	Lewis Carroll
file3.txt	Dorothy and the Wizard in Oz	Oz	Lyman Frank Baum
file4.txt	Emerald City of Oz	Oz	Lyman Frank Baum
file5.txt	Royal Book of Oz	Oz	Ruth Plumly Thompson
file6.txt	Glinda of Oz	Oz	Lyman Frank Baum

What's Your Answer?

	file1.txt	file2.txt	file3.txt	file4.txt	file5.txt	file6.txt
file1.txt	0	0.16824579	0.08785724	0.08832557	0.13960696	0.10485192
file2.txt	0.16824579	0	0.1688084	0.1623172	0.18100267	0.17304682
file3.txt	0.08785724	0.1688084	0	0.07196412	0.11099609	0.07941847
file4.txt	0.08832557	0.1623172	0.07196412	0	0.13234903	0.09274778
file5.txt	0.13960696	0.18100267	0.11099609	0.13234903	0	0.14705212
file6.txt	0.10485192	0.17304682	0.07941847	0.09274778	0.14705212	0

File	Title	Series	Author
file1.txt	Wonder Wizard of Oz	Oz	Lyman Frank Baum
file2.txt	Alice's Adventures in Wonderland	Alice in Wonderland	Lewis Carroll
file3.txt	Dorothy and the Wizard in Oz	Oz	Lyman Frank Baum
file4.txt	Emerald City of Oz	Oz	Lyman Frank Baum
file5.txt	Royal Book of Oz	Oz	Ruth Plumly Thompson
file6.txt	Glinda of Oz	Oz	Lyman Frank Baum

Stuff to do

- Think about Project 2 ideas for your proposal
  - Initial proposal due tomorrow
  - Revised proposal due on next Monday

- Revisit the activities we did to get more practice with Python

Tools you've learned

- Reading and writing files **(ACT 2-2, 2-7)**
- String processing: split(), find(), etc. **(ACT 2-2, 2-6)**
- Lists and dictionaries **(ACT 2-5)**
- Iterating over data: two approaches to for-loops **(HW 2-4)**
- Summaries statistics like counts, averages, min/max **(ACT 2-3)**