Python in 10 minutes

Part 3:

Dr. Mark Williamson

Purpose:

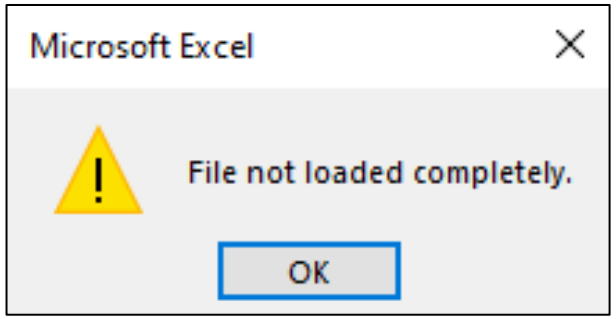
- Quick, bite-size guides to basic usage and tasks in Python
- I’m no expert, I’ve just used it for various tasks, and it has made my life easier and allowed me to do things I couldn’t manually
- I’d like to share that working knowledge with you

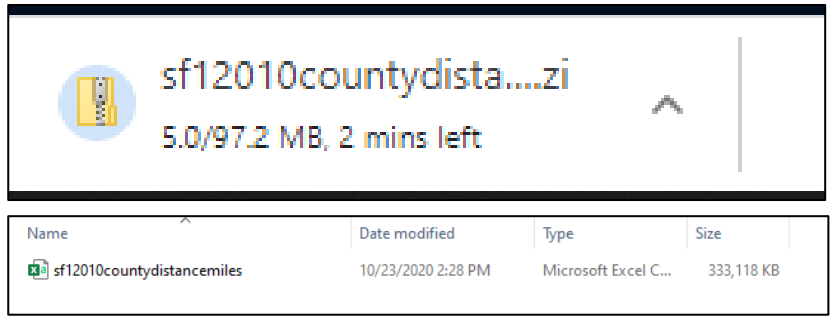
Lesson 3: Exploring a Large Dataset

 Today, we'll be using Python to parse through and explore a large dataset. This is a very useful technique because normal tools like Excel can't fully open a file if it is too large. Instead, the data will be cut off after a certain size. We can use Python to determine how large the file it, as well as other basic characteristics. This can be the first step in condensing or sub-setting the data for further work.

Lesson 3: Getting the Data

- We'll be using county distances
  - Great-circle distances of all counties from the National Bureau of Economic Research
  - https://data.nber.org/data/county-distance-database.html
  - Download the **csv** version of the **2010** Year for infinite distance
    - It might take a while to download
  - Unzip and try to open in Excel
    - Should get a warning

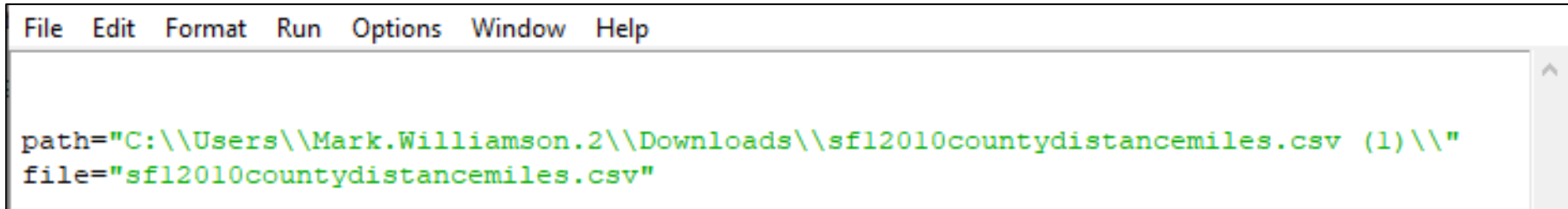




Year	Source	County Name				Distance in Miles / (CSV file size)																			
						25				50				100 (23M)				500 (330M)				∞ (670M)			
		Stata	SAS	CSV	Desc	Stata	SAS	CSV	Desc	Stata	SAS	CSV	Desc	Stata	SAS	CSV	Desc	Stata	SAS	CSV	Desc	Stata	SAS	CSV	Desc
2010	SF1	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc
2000	SF1	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc
1990	Gazetteer	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc	dta	sas7bdat	csv	desc

Lesson 3: Getting File Information

- Open Python and start new file
- Locate the file path for the county distance csv
  - Yours will be different from mine (my example below)
  - Can also find the location by right clicking on csv file and selecting ‘Properties’
- Create a variable called **path** with the file path as a string
  - Need to enclose in quotation marks
  - Also, need to add a second backslash (\) to each backslash and two more at the end
- Created another variable called **file** with the file name as a string
  - Should be **sf12010countydistancemiles**
  - Include **.csv** at the end and enclose in quotation marks



Lesson 3: Opening and Checking File Length

- Create a variable called **tally** and set it to zero (0)
- Start a for-loop
  - For-loops iterate over a sequence
  - Real world example: “For each number from 1-10, say the number out loud.”
  - Python example: for x in [1,2,3,4,5,6,7,8,9,10]: print(x)
- This loop with go through each line and add one to the tally
  - The for-loop line needs to end with a colon (:)
  - The argument inside the loop needs to be indented
- Then, print out the tally’s final count
  - This will give the total number of lines
  - Number should be 10371621
    - 1 header line
    - 10,371,620 lines of observations
    - The distance from every county to every other county
    - That’s a lot of observations

File Edit Format Run Options Window Help

```
path="C:\\Users\\Mark.Williamson.2\\Downloads\\sf12010countydistancemiles.csv (1)\\"
file="sf12010countydistancemiles.csv"

tally=0
for line in open(path+file):
    tally +=1

print(tally)
```

Lesson 3: Checking the First Ten Entries

- Create a variable called **tally2** and set it to zero (0)
- Create another for-loop
- Inside the for-loop, create an if-else statement
  - If-else statements check a condition and then do something based on that condition
  - Real world example: “If it is sunny, I am going outside. Else, I’m staying inside.”
  - Python example: if variable==sunny”: print(“Going outside”): else: print(“Stayin’ in”)
- In the if line, if the tally is not yet to 10, print the line
- Otherwise break, which will stop the whole for-loop
- After the if-else lines, add one to the tally2 variable
  - Keep the indentation the same as the if and else
- Should be one header line and nine observations
- Each line has a FIPS code for a county, the distance to a second county, and a FIPS code for that county
  - The FIPS is unique for each county

```
"county1","mi_to_county","county2"

"01001",22.4629943022086,"01021"

"01001",26.8446865669988,"01085"

"01001",29.5175849409829,"01051"

"01001",30.7763708418057,"01047"

"01001",34.4934427264388,"01101"

"01001",35.8927585637374,"01037"

"01001",38.4608685606685,"01105"

"01001",43.4032527287312,"01007"

"01001",50.2379456016557,"01117"

>>>
```

File Edit Format Run Options Window Help

```
path="C:\\Users\\Mark.Williamson.2\\Downloads\\sf12010countydistancemiles.csv (1)\\"
file="sf12010countydistancemiles.csv"

tally=0
for line in open(path+file):
    tally +=1

print(tally)


tally2=0
for line in open(path+file):
    if tally2<10:
        print(line)
    else:
        break
    tally2+=1
```

Lesson 3: Counting Your County

- Find your county's FIPS code at the link below
  - https://www.nrcs.usda.gov/wps/portal/nrcs/detail/national/home/?cid=nrcs143_013697
  - Hint, find your state first (alphabetical order)
  - Example: Grand Forks County is **38035**
- Create a variable called **own_county_code** and set it to your county's FIPS code (include quotations)
- Create another variable called **own_county_tally** and set it to zero (0)
- Run a for-loop through each line
- Add an if-statement inside the for-loop (indent) that will add 1 to the tally if your county code is in that line
  - Don't need an else-statement if you only want to do one thing
  - The else is automatic then, nothing happens
- Finally, print out a string that has the number of lines your county is in
  - Need to turn your tally into a string by using str(own_count_tally)
  - Can add strings together using the + sign

```
Grand Forks County is in 7258 lines
>>>
```

File Edit Format Run Options Window Help

```
path="C:\\Users\\Mark.Williamson.2\\Downloads\\sf12010countydistancemiles.csv (1)\\"
file="sf12010countydistancemiles.csv"

tally=0
for line in open(path+file):
    tally +=1

print(tally)


tally2=0
for line in open(path+file):
    if tally2<10:
        print(line)
    else:
        break
    tally2+=1


own_county_code="38035"
own_county_tally=0
for line in open(path+file):
    if own_county_code in line:
        own_county_tally+=1

print("Grand Forks County is in " + str(own_county_tally) + " lines")
```

Lesson 3: Your County to Anywhere

- Let's wrap this up by determining the distance from your county to anywhere else
  - I'll used Grand Forks, ND (38035) and New York, NY (36061)
- Create a variable called **destination_county_code** and set it to a county of your choice
- Run a final for-loop
- Inside the for-loop, create a new variable called **distance**
  - Because it is inside the for-loop, it will be updated ever line
  - Make distance=line.split(',')[1]
    - This will split the line up by the comma into three pieces and set distance to the second piece because iterations in python start at 0 so 1 is the second item
    - The second piece each line is the distance in miles from the first county to the second
- Create an if-statement to see if both your county code and your destination's county code is in the line
  - If so, print out the distance in a string
  - Your if-statement will need two conditions
- Will need to use the and operator
  - Both conditions will need to be fulfilled for the if-statement to be valid
- The line should print twice
  - Once for when your county is in the first column and the destination is in the second
  - A second time for when the destination is first, and your county is second

```
Grand Forks County is 1254.52234789588 miles from New York County
Grand Forks County is 1254.52234789588 miles from New York County
>>>
```

File Edit Format Run Options Window Help

```
path="C:\\Users\\Mark.Williamson.2\\Downloads\\sf12010countydistancemiles.csv (1)\\"
file="sf12010countydistancemiles.csv"

tally=0
for line in open(path+file):
    tally +=1

print(tally)


tally2=0
for line in open(path+file):
    if tally2<10:
        print(line)
    else:
        break
    tally2+=1


own_county_code="38035"
own_county_tally=0
for line in open(path+file):
    if own_county_code in line:
        own_county_tally+=1

print("Grand Forks County is in " + str(own_county_tally) + " lines")

destination_county_code="36061"
for line in open(path+file):
    distance=line.split(',')[1]
    if own_county_code in line and destination_county_code in line:
        print("Grand Forks County is " +str(distance) + " miles from New York County")
```

Lesson 3: Summary

- Python can quickly parse through large datasets to make tallies, check variables, compare variables, etc.
- To do so, you can use for-loops, if-else statements, mathematical operators, and other functions and methods to aid you
- To learn more for-loops and if-else statements, check out the following resources:
  - https://www.tutorialspoint.com/python/python_for_loop.htm
  - https://www.tutorialspoint.com/python/python_if_else.htm