Project 5 CS 111 Law: Predictive Policing in Chicago

Release Date: Thursday, November 29, 2018.
Release version 1.0 Due: **10:00 pm** Sunday, December 9, 2018 (last day of classes)

Learning Objectives: Working with dictionaries, nested lists, and heatmaps in Python, and working with real City of Chicago data.

Submit your assignment on Blackboard in a single .zip file called [your NetID]Project5.zip

In this homework, you will work with real data from the City of Chicago to develop heatmaps that show the prevalence of conditions of your choice in the 50 wards of Chicago.

Remind students about low_memory=False issue for really big files with pandas.

Overview

We will develop a visualization of some Chicago data by ward, making the simplifying assumption that the 50 wards are equal size little rectangles, laid out in big rectangle of 10 rows of 5 columns.

You will work with some (large, real) datasets we have downloaded for you from the City of Chicago Data Portal. They are all in CSV ASCII format. You will use pandas to get each into a dataframe.

We are supplying you with a function `ward_count(df, col, value, unique_id, ward_num)` that will extract the count of how many unique instances there are in dataframe `df` with:

- Entries in column with column name col equal to value; for example, entries in the column "Primary Type" equal to "HOMICIDE" (in the crime data), and
- With the unique identifier of the things we want to count unique_id; for instance, in the Supreme Court by Justice database this would have been 'docketId' and in the crime database this is 'Case Number', and
- In specifically Ward number ward_num. Chicago Wards are numbered from 1 to 50. (UIC is in the 25th Ward, and the 25th Ward Alderman is Danny Solis.)

The function is:

```python
def ward_count(df, col, value, unique_id, ward_num):
    '''Returns count of unique_id in df ward_num rows w/entry value in column col'''
    return df[df[col] == value].groupby("Ward")[unique_id].count()[ward_num]
```

What's impressive about pandas: That function will run in under a second on a 3-year-old laptop for the Chicago crime database from 2010 to 2018, which has over two million rows!

(If you go back and look at the information on pandas in Lecture 15 you can understand the ward_count function, but it is not a requirement of this assignment or this course that you understand it.)

The data files

We have downloaded three data files for you from the City of Chicago Data Portal at https://data.cityofchicago.org/. (We downloaded them for you so everybody starts with the same file with the same ASCII CSV encodings.)

1. Crimes_-_2010_to_present.csv: This has an entry for every crime reported to the Chicago police between 2010 and about a week or two ago. Important column headers for you:
   - 'Case Number': A unique ID for each case
   - 'Ward'
   - 'Primary Type': The main type of the crime. You'll probably want to work with one of the values for this column of 'HOMICIDE', 'NARCOTICS', OR 'ASSAULT'.

2. 311_Service_Requests_-_Vacant_and_Abandoned_Buildings_Reported.csv. This has an entry for every vacant/abandoned building reported to the city since 2008. Important column headers for you:
   - 'SERVICE REQUEST NUMBER': A unique ID
   - 'Ward'
   - 'SERVICE REQUEST TYPE': The type of record, the only ones we care about will have the value 'Vacant/Abandoned Building' (and all or almost all these entries in this database have this value)

3. 311*Service*Requests-Graffiti_Removal.csv. This has an entry for every graffiti removal request submitted to the city of Chicago since 2011. Same entries as the previous file except:

   - For SERVICE REQUEST TYPE' the value we care about is 'Graffiti Removal'.

First task: Getting started with a dictionary

Required Write a function called `make_ward_dictionary` that uses the `ward_count` function to make and return a dictionary whose keys are ward numbers (integers from 1 to 50) and whose values are the count

of the specified thing for each ward.

The inputs to `make_ward_dictionary`, in order should be:

- df: A pandas dataframe (that must contain a column named "Ward")
- col: The name of the column of interest (in addition to the Ward column)
- value: The value of interest for that column
- unique_id: The name of the column with the unique identifier (e.g., Case Number, DocketId, Report Number) we want to count.

Your function should return the dictionary you made.

If you do it right this function can be written in under 10 lines.

The rest of the task: Heatmaps

Your goal is to produce at least three heatmaps, each showing something by Chicago Ward. In all cases, you will be plotting a heatmap from a 2-D 10 x 5 list (array) of intensity values, one for each of Chicago's 50 wards, which are numbered 1 to 50 (in shocking disregard of Python convention by Chicago's founders in the 19th century!).

The 2-D array should be set up so that the first row (row 0) has the entries for Wards 1 to 5, the second row (row 1) has the entries for Wards 6 to 10, and so on. When you display your heatmap created with `plt.pcolor(ward_array)` the first row (row 0) will display at the *bottom* of the figure.

You should make a minimum of two simple heatmaps, and one more complicated heatmap.

You need to put the image of each plot into the zip file that you turn in, as well as submitting all your Python code.

You must give all plots a title on the plot that indicates what the plot is (e.g., homicides by ward).

All heatmaps should include a color bar so viewers know the scale. This can be accomplished by the statement `plt.colorbar()`

You can choose among the sequential ("Sequential" and "Sequential (2)") colormap choices available and listed at http://matplotlib.org/examples/color/colormaps_reference.html. I'm fond of summer for things where low is good (green) and high is of concern. To set the summer color scheme, use `plt.summer()`, and similarly for other color schemes.

The two simple heatmaps

1. Plot the counts of at least one primary type of crime by ward. Choose a type of crime that you consider to be fairly serious.
2. Plot either the counts of 'Vacant/Abandoned Building' by ward, or the counts of graffiti reports by ward (or both)

Here is an example for the primary category of crime 'CRIMINAL DAMAGE', which is not especially serious (relative to universe of all crimes in Chicago that is. *Call a good criminal lawyer immediately if you get arrested for it!*):

The more complicated plot: Where would you put more police?

The goal of predictive policing is to use data to predict where crime is most likely to occur, so policing can be proactively moved into those areas.

We want you to make a prediction (by ward) and show us your heatmap.

You need to combine *scaled* data from two (or more if you like) of the simple heatmaps you did. For each input, you should divide all the original values by the maximum value over all the wards to get a new set of values that will range between 0 and 1. The dictionary values() method will be useful for this, combined with the .max() method for lists. Remember that to get a list of dictionary d's values you need `list(d.values())`.

Why scaling? So we can combine different factors. For a given ward, 40 is a lot of homicides, 4000 is a lot of narcotics incidents, and 12,000 is a lot of abandoned buildings. If we just added those three factors without scaling, then the number of abandoned buildings would overwhelm the other factors.

How to combine? That's up to you and your insight. You could simply add two factors, but maybe you want to apply some weighting: Maybe you think the number of murders is 2.5 times as predictive as the the number of abandoned buildings. For graffiti reports, probably you want to add them, but you could consider subtracting them, because maybe neighborhoods where residents actually report graffiti have less crime.

Modules to be sure to import

You will need the following modules:

- matplotlib.pyplot
- pandas

Things to submit in your zip file

- At least 3 heatmap images
- Your Python code, in a file named [your NetID]Project8.py (Whatever Python code you write into a file; it's okay if some of your work is done only at the command line. The function `make_ward_dictionary` must be in the file.)
- A short text file, explaining the choices you made for your heatmap and why.