Today, I am going to show you how to make it rain data with word clouds. Sure, there are free and paid resources out there with more advanced features but if all you want is something simple, private and free then why not roll your own?

You’ll need Python 3.x, wordcloud, natural language toolkit (nltk), pyperclip, collections, and regex (re).

Like many of my friends, I am in the middle of a job search and my hope is that this little demonstration might help some of you guys out in your search as well.

This python script is an attempt do the following things:

\* Generate a word cloud from a job description, filtering out stop words and common English words

\* Get the top 20 words from the word cloud.

\* Try to find keywords by searching all capitalized words and filtering out common English words

\* Get the top 20 capitalized words from the word cloud.

\* Add the Top word and Capitalized words to a csv text file for later analysis in excel

Beginning in the main part of the script, on line 45 it opens the stop words text file and reads it into the `stops` list. Stop words are common, high frequency words. WordCloud will remove stop words by default but the text can be refined further by utilizing a customizable stop word file.

Notice the `while True` loop on line 48. This is done because it takes time to load all the modules and it allows the user to generate more than one word cloud while running the script.. On line 52, the user is given a way out by allowing a few keywords that will cause it to enter the breakpoint.

On line 56, it calls the `read\_clipboard(stops)` function, passing in the stop words list. It starts by reading text from the clipboard with `pyperclip`. Then it filters out the stop words by using list comprehension on line 11. On line 12, which is currently commented out (as an optional step), it can do further filtering to remove most words from the english language. However, this isn't smart enough to recognize plurals and past/future tense forms of english verbs.

On line 57, it calls the the `get\_top\_words(text)` function, passing in the filtered `text` string from the clipboard. This function uses `Counter` (line 17) to get the most commonly occurring words in the string. The returned `top` variable is a list of tuples that will be written to a text file later on, as part of the data collection process.

Line 58 is where the first word cloud is generated by calling the `generate\_cloud(text,myfile)` function. On line 21, the `WordCloud` object is instantiated and on line 22 the text is passed to its generate method. Finally on line 23, it writes the output to an image file with the file name that was passed to it.

On line 59 it makes a function call to write csv data: `write\_tuples(myfile,'TOP',top)`. It iterates through and writes the text from each element of the tuple (i), separated by commas. It can be useful to collect this data for later analysis.

The second code block repeats the same process, except for the first step on line 61 where it calls a function `get\_capitalized\_words` which uses a regular expression on line 27 to find all the capitalized words in the originally copied text. A second word cloud is then generated from the capital words, filtered for stop words and (most) english words, leaving (mostly) specific technical terms.

I hope you will find this script useful or at least interesting enough to give it a try. See below for the complete script. Copy it and save it in a directory along with stops.txt.