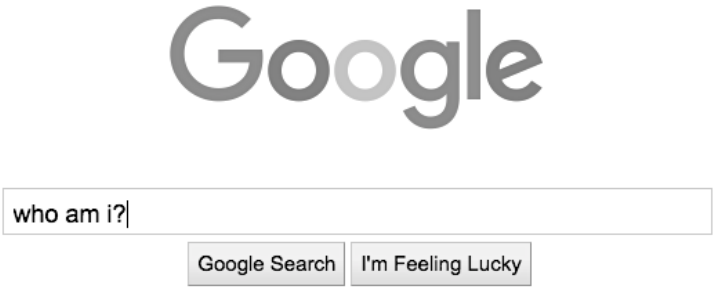
Recommended by Medium Staff and 406 others

Walker Harrison Follow
Mar 10 · 5 min read



The Search for Self

How to obtain and analyze your history of Google searches, using myself as an example.

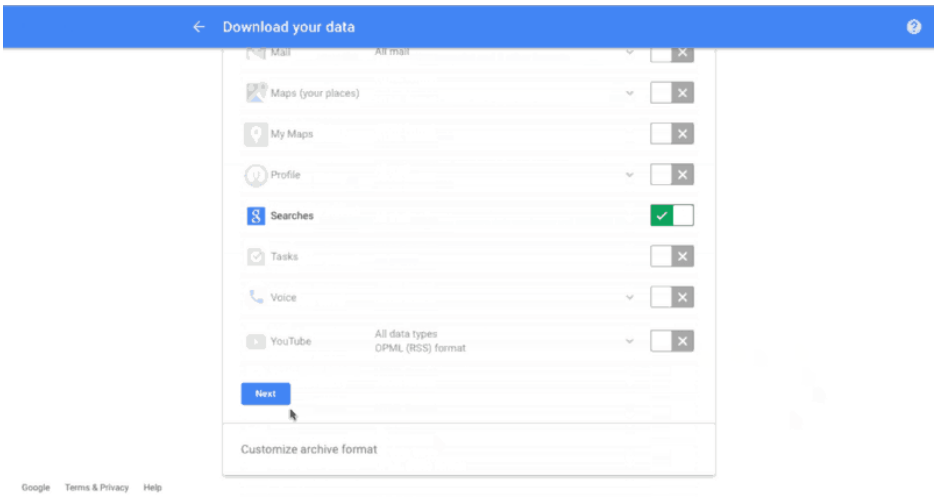
Google's search engine is so thoroughly baked into our everyday existence that it feels more like the final stage in a cognitive process than it does an independent piece of software. Modern humans don't wonder, they wonder-then-Google, with the taps of characters into your address bar as natural and legitimate a step as the original thought.

As a result, your accumulation of Google searches over a period of time acts as a reliable proxy for your state of mind, curiosities, ambitions, and fears included. Luckily (or not, depending on your definition of privacy), Google logs your searches and makes them available to you, assuming you're signed in to a Google account (often via Gmail). Here's how to find, parse, and visualize that data, starring the author as guinea pig.

1. **Download the data**

Head to https://takeout.google.com/settings/takeout, where you'll find various personal datasets available, including your GChat conversations and emails. Unselect all of them ("Select none"), then recheck Searches

and hit “Next.” On the next page you can choose a file type (*.tgz* allows for fewer files) and delivery method (I stuck with a download link sent over email). After opening that email, clicking through, downloading the archive and unzipping it, you’ll be left with a collection of files nested under the folders “Takeout” and “Searches.”



2. Prepare the data

The data is in JSON format, but is still organized in a relatively straightforward manner and can be flattened into vectors without too much trouble in Python:

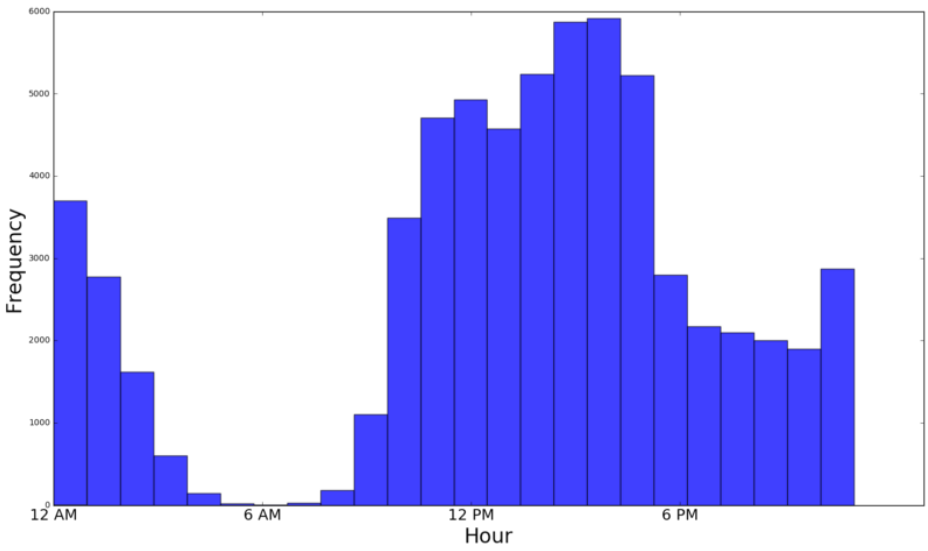
```
import json
import os
import datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from collections import Counter

files= os.listdir('Searches')
del files[0]

searches = []
dates = []
for file in files:
    with open('Searches/%s'%(file)) as json_data:
        d = json.load(json_data)
```

3. Analyze the data

We'll start with some high-level figures. In the 886 days spanning the available time period back to Fall of 2014, I executed nearly 64,000 Google searches, or over 70 per day. I use my personal laptop at work everyday, which helps explain such volume, but clearly the pervasiveness of Google searches mentioned in the intro was not overstated!

There are more patterns worth mining though. You could look at hour-by-hour trends:

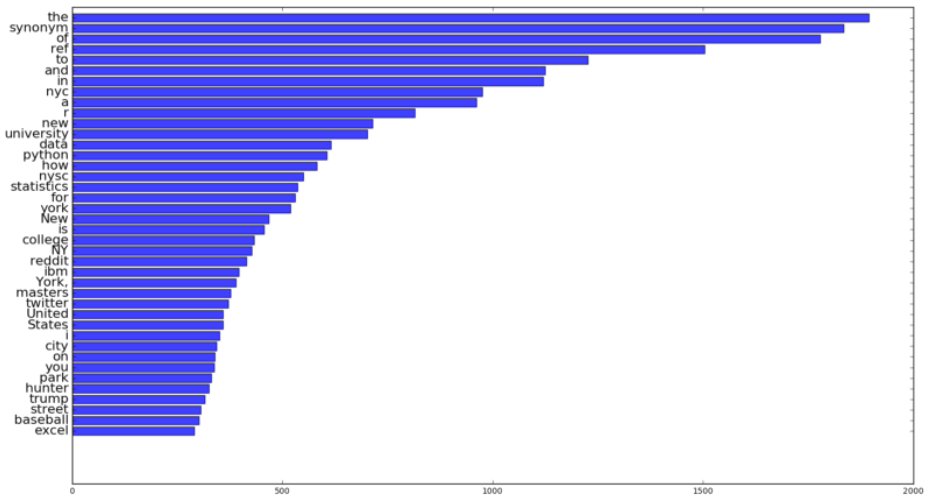
```
hours = [datetime.datetime.strptime(i, '%Y-%m-%d %H:%M:
n, bins, patches = plt.hist(hours, 24, facecolor='blue'
plt.xticks([0,6,12,18], ['12 AM','6 AM', '12 PM', '6 PM
plt.xlabel('Hour', fontsize=24)
plt.ylabel('Frequency', fontsize=24)
```



At its simplest, the hour-by-hour graph reflects my consciousness: he who does not Google is probably asleep. Soon after arriving at work though, I begin searching up a frenzy, reaching peak inquisitiveness around 3 PM. After an early evening respite, I'm back on my search grind by 10 PM and don't finish up until well past midnight (I'm a bit of a night owl).

What exactly am I Googling though? Sorting for term frequency isn't too difficult:

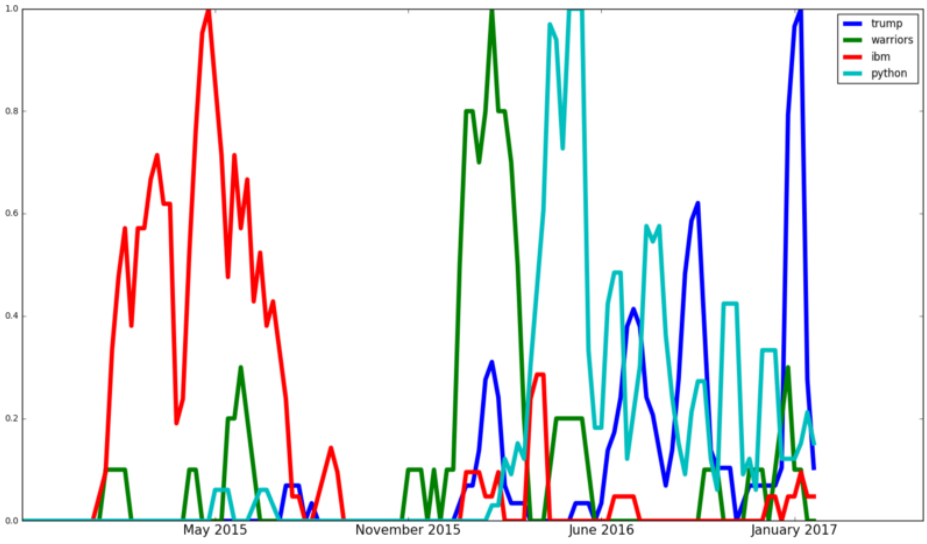
```
combo = ' '.join(searches)
freqs = Counter(combo.split())
top = freqs.most_common(40)

words = []
counts = []
for i in range(40):
    words.append(top[i][0])
    counts.append(top[i][1])

words.reverse()
counts.reverse()
```



The usual suspects from the English language like “the” and “of” dilute the list, but you can still see where my mind’s been in the last few years. I blog regularly and like to avoid overusing a word, hence the heavy reliance on searching for synonyms. I live in New York (“nyc”) and go to the gym a fair amount (“nysc”). I’m an aspiring data scientist (“data,” “python,” “r”). I’m quintessentially American (“baseball”, “States”), but also worried about what that means nowadays (“trump”).

There is, of course, a time component to each of these terms. People don’t Google the same things everyday for the same reasons they don’t think the same thoughts every day. So picking certain popular terms and examining their fluctuations over time gives us a sense of how our interests and focus change as the weeks roll by:

```
d = {"search": searches,
    "time": dates}
googled = pd.DataFrame(d)

dt = datetime.datetime(2014, 10, 1)
end = datetime.datetime(2017, 3, 5)
step = datetime.timedelta(days=7)

weekly = []

while dt < end:
    weekly.append(dt.strftime('%Y-%m-%d %H:%M:%S'))
    dt += step

# finding/smoothing/normalizing weekly data only shown
trump_weeks = []
for i in range(len(weekly)-1):
    trump_weeks.append(sum((googled['time'] > weekly[i
                (googled['time'] < weekly[i+1]) &
                (googled['search'].str.contains('trump')

term = len(trump_weeks)-1
trump_weeks_smooth = [(trump_weeks[i] + trump_weeks[i-
```



Results are smoothed and normalized to their peaks.

Without ever meeting me, you could use a graph like this to understand who I was and what I was thinking about over a long

period of time (which, of course, is what Google *does* to make gobs of money). I worked for IBM (red) after I graduated until changing jobs in summer of 2015. For months, I closely followed the Golden State Warriors' record-breaking season (green). I decided to learn Python (light blue), the programming language used for all this, in the spring of 2016. And I paid great attention to Trump (dark blue) as the election neared, took a much needed hiatus, and then replugged in for his inauguration.

. . .

Unfortunately, your lasting takeaway from this post may be a reminder of Google's omniscience. You may have noticed all the other things I didn't check when exporting my data, from maps to GChat conversations to personal calendars. There are long, complex conversations to be had about how big your digital footprint should be and who should have access to it.

One certain thing is that *you* have the right to view your past online actions, and as demonstrated above, the capability to find meaning in them. In an age where all of us are too distracted to write a journal entry before bed, Google provides something of an approximation for a diary, and one that's likely a little more honest at that.

So I would encourage you to at least download your data, and even take a shot at analyzing it. The full code is linked below and I'd be happy to lend a helping hand to those who find the syntax inaccessible and try to answer any questions you might have about the process…

…or you could just Google it. 😉

> *Thanks for reading!*

> *The code (but not the data!) can be found at https://github.com/WalkerHarrison/Google_searches.*

> *New posts every Thursday! Questions? Want to write a post? Email hello@perplex.city.*