Data Mining: Philosophy Game in Wikipedia

L. J. Hachmann

Project Overview

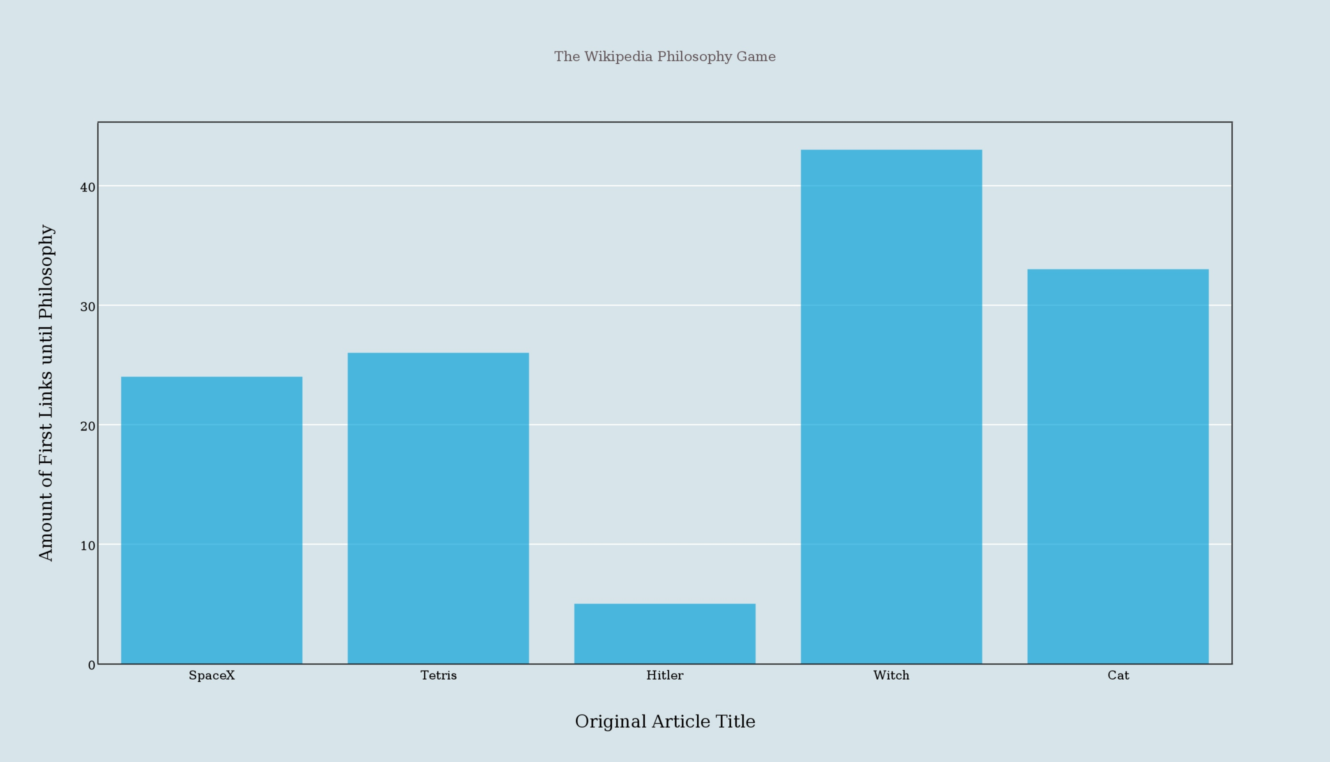
I used Wikipedia in order to play the Philosophy game: How many times do you have to click the first link in Wikipedia to get to the “Philosophy” page? I had to use Pattern BeautifulSoup python packages in order to create data, and Plotly python in order to make the graph. I hoped to analyze the differences in the amount of links used for each- for instance, see that 'Hitler' had a 5 link patter and Witch had a 42 link pattern.

Implementation

Early on, I decided that the easiest way to navigate until the article page was “Philosophy” would be a recursive function. However, besides checking for equal names and therefore infinite links loops, there was very little computation involved. When I realized that Patterns was returning links in alphabetical order, I made the decision to incorporate BeautifulSoup instead of going deeper into the possible data that Pattern could have possibly given me.

Due to the time that some words can take within the code, I decided to not create a dictionary of articles with solutions to the Wikipedia Philosophy Game. Some words were taking 2 minutes, while others could go on for 10 minutes with no sign of stopping. So instead, I printed one each time I ran the script and then manually entered them into a new script that would then graph them as a bar chart onto Plotly. Therefore, if any word took over 5 minutes, I could interrupt the script and try another word.

Results



The above figure shows the amount of links are needed to navigate from the original article listed to “Philosophy”. It was interesting to see that 'Hitler' had a short pathway to 'Philosophy' (5), 'Witches' had the largest amount that I tested (43).
What I accomplished was multiple page navigation of Wikipedia through Python, while collecting data about every page and keeping an overall count. I know this program is limited to seeing and avoiding files and audio translations, but Wikipedia has other things that might be in front of a random article, so this program doesn't work on every word yet. It's very hard to verify if sometimes the program gets stuck in one title, recognizing other links or not, so further work would have to be done with more testing.

Reflection

This wasn't the project I originally decided on: Initially, I wanted to create sentiments of friends' facebook posts over large amounts of time, to see how growing up had affected their sentiment. However, facebook had internal issues (401 Authorization) that I couldn't solve within the expected time frame, so I changed project ideas to the Wikipedia Philosophy game. The project is a little underscoped, and I could vastly improve the script if it could test multiple words during the same script. I wish I had known that Facebook's authorization would be harder to work with than the actual project, and just started on Wikipedia earlier.

In doing the Wikipedia Philosophy game, I learned a lot about BeautifulSoup. Pattern automatically gives back an alphabetized list of links, so there was absolutely no guarantee I would ever get to Philosophy. However, even with BeautifulSoup, the code won't work for every sense. The Wikipedia page source code has many different cases, and I couldn't account for all of them. It was very useful to learn a little bit about BeautifulSoup and recognizing html through Python.