Mark McCarthy

5/3/2021

CS697-AK

**Keywords and Ad Revenue**

**Introduction:**

The issue this project attempts to resolve revolves around the question of what kind of returns one can expect from the keywords one chooses. An example of where this issue comes to light is easily seen when tackling the problem of titling, tagging, or monetizing web content from the perspective of a sole proprietor content creator. As a content creator, one has much depending on effective keyword selection and, in many cases, the revenue brought in by advertisements. Some of the applications of solutions to this issue include furthering the effectiveness of SEO (Search Engine Optimization), progress in NLP (Natural Language Processing), development of new marketing strategies, and many other more abstract applications (the field of psychology would be an interesting place to apply ideas in keyword research). As for the motivation behind choosing this subject for my term project, I have always been interested in how headlines are written and how these headlines impact business as well as how they are involved in the psychology of economics that drive revenue.

**Background:**

The background material and literature surrounding keyword choice is, as I have found, very pertinent to today’s SEO and NLP markets. As such most of the ideologies and methods behind current practices are very difficult to ascertain without paying for or subscribing to SEO services. The best information discovered was in personal blogs and GitHub repositories. However, for this project, none of the resources found were utilized. Therefore, this project has no background materials to source. If the reader is interested in studying further on SEO, NLP, and related methodologies, I would suggest starting with GitHub repositories returned from searching for “keyword” or “keyword optimization.”

**Methodology:**

The approach developed in this project focuses mainly on deriving and assigning a normalized earning potential for each of the keywords present in the data set. To begin, a description of the data set is necessary. The data set used in this project is sourced from and credited to a site at the URL. <https://www.mondovo.com/keywords/most-asked-questions-on-google>. This site lists the 1000 most asked questions on google worldwide for a month-long time step (I could not find if the month time step is actually updated on a monthly basis). The data points available include the rank of the question asked as assigned by Mondovo, the question asked, the global monthly searches (which appear to be what Mondovo largely based the rankings from), the cost per click (CPC), and a link to related keywords on a paid SEO tool page benefiting Mondovo. These links to related keywords could be very useful in future iterations of this project but presently they are omitted from the data set. The rank, questions, global searches, global CPC, and string version of the links were scrapped from the website using the BeautifulSoup library in Python.

A brief overview of the method for scrapping will be covered here. If the reader requires a more in-depth explanation of the scrapping process, the code is attached in the appendix. The BeautifulSoup library allows the Python script to make a URL request given a target URL, then reads in the content of the webpage as a structured “Soup” based on the html format. From here, depending on the inputs given, the “Soup” is parsed, and information can be extracted. For this project this was done by finding all the “td” tags (the input) in the html “Soup” and extracting the required data based on further inputs available to view in the code. After extracting the data from the “Soup” it is placed into a list to be preprocessed into a format that accommodates entry into a data frame. A note to the reader, the “td” tag used produced excess unwanted data as there are more items on the webpage with the tag used than pertained to the needed data. To mitigate this, the entirety of the data associated with the search tag was printed out on individual lines. This allowed for locating the terminating entry of the relevant data. From here the relevant data was copied and pasted into a notepad application to get a line count for the data needed. Based on this count, the data collection process was altered to terminate collection at the appropriate location. The data was then placed into a list for formatting before being loaded into a data frame.

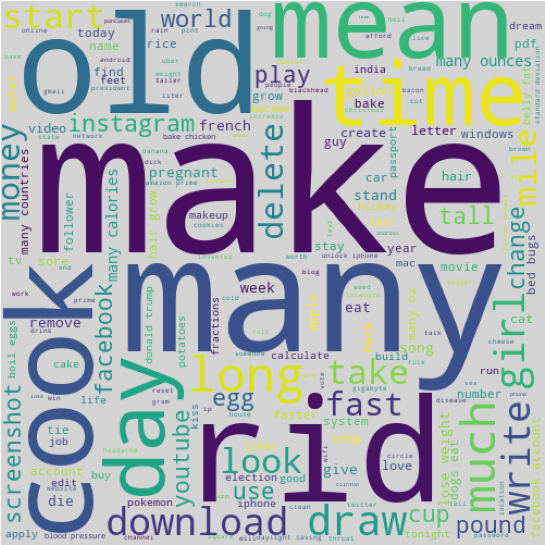
Once the data had been scrapped and formatted it was loaded into a data frame. After being loaded into the data frame, the data required a fair amount of cleaning to be utilized. Initially, there were gaps in the scrapped data for advertisements on the Mondovo site that were removed manually before the data was entered into the data frame. Most of the additional string manipulation was implemented with lambda functions to modify the data in place. These modifications can be described as small changes to strings and conversion of data types (e.g., Removing “$” from the global CPC column or converting the global monthly search column to integers from strings). These cleaning methods are all described in the comments of the code.

From here I needed to begin constructing a method for transforming the global monthly search and the global CPC into new data describing the ad revenue and then further transforming the data into a weight associated with each keyword. Creating the ad revenue data was simple enough. I took the assumption here that a search implied a click. Then I was able to multiply the global monthly search by the global CPC to yield the monthly potential ad revenue for each question. As a side note this data showed a rather staggering earning potential capping out at $2,598,600 for a single month’s potential ad revenue. Now that the potential ad revenue had been calculated, I needed to formulate a system for giving a weight to each keyword. This was accomplished in three stages. The first stage was to remove all the “StopWords” from the questions. “StopWords” are a category of small grammatically necessary words like “is”, “are”, “the”, “and”, etc. This category is a portion of the NLTK library. These words are removed by appending each of the questions to a single string delimiting with a space. Then the aggregate string is tokenized into a list with the NLTK library. This tokenized list is then checked against the list of “StopWords” and common words are removed. After removing the “StopWords”, the list of filtered keywords is used to create a dictionary where the key-value pairs are defined as {keyword:frequency}, where the frequency pertains to the total frequency of each keyword as they appear in the entire dataset.

For stage two, with the keywords having been tokenized, filtered, and counted, a method for assigning a weight to each keyword can be determined. The approach used to assign weights is through utilizing the Minmax function to produce a normalized ad revenue value for each question in the dataset, then assign this normalized ad revenue value to each keyword in the respective question. There is some thought behind this method to unpack. Since the data is ranked by global monthly search, the higher ranked questions should, on average, have a higher ad revenue. Based on this notion the first appearance of a keyword as you traverse the questions from the top rank down gives the presumably highest weight to each keyword as it appears in the questions. So, the weight associated with the first question a keyword appears in is assigned to the keyword as its initial weight. From here each subsequent appearance of a keyword modifies the weight on a percentage basis. The percentage method developed here is based on the idea that as you traverse the questions from the top rank down, each subsequent occurrence of a keyword will have a lower rank which implies that, on average, the normalized ad revenue associated with the lower ranked question will be lower. In other words, if a keyword is found in a lower ranked question, it presumably has less earning potential than the first time the keyword is found. With this in mind, the weight associated with the subsequent question is divided by the initial weight of the keyword yielding a percentage of the initial weight. This percentage is then added to the initial weight. This methodology will give the highest weights to the keywords that have the most searches and the highest CPC at the same time. This gives insight into the trends of the user searches and trends of what advertisers will pay the most for. This structure is implemented with a dictionary defined as {keyword:weight}.

Now that each keyword has a representative weight associated with it, we can move to stage three and utilize the dictionary containing the keyword frequencies along with the dictionary of weights to produce a ranking of each keyword in the dataset. this is achieved by simply multiplying a keyword’s weight with the respective keyword’s frequency. This gives a basis to rank each word on. The data produced in these last few steps was put into a second data frame for ease of manipulation.

**Results:**

From here EDA was begun and will comprise the results section of this paper. Numerous graphs were rendered using the Matplotlib library attempting to discover correlations between the Mondovo ranking and other features. There was not much correlation discovered between the Mondovo ranking and features other than the presumed ranking basis of global monthly searches. An interesting phenomenon that occurred in the graph of the derived ad revenue and the Mondovo ranking were recurring spikes and exponential decay as the ranking decreased. No empirical explanation was discovered for this phenomenon, however speculation involving the methodologies of Mondovo’s ranking system may lead to an explanation. As for the core result of this project, the goal of creating an effectiveness measure for individual keywords was a success. The best way found to represent the results of this measure was to utilize a word cloud based on the effectiveness of each keyword. The code to generate the word cloud in this project was found in a GitHub repository at the URL <https://github.com/kavgan/word_cloud> . 

(Note: data contains explicit content)

**Conclusion & Discussion:**

The conclusion reached as a result of this project is that it is possible to amass a database of keywords and their earning potentials based on data from search engines. This project is large enough to provide a proof of concept, but a much larger sample size would be necessary to make any industry judgements based on the data provided. Along with the larger sample size, other improvements to be made entail including the “StopWords” in the derivation of the final earning potential to provide more context yielding a more accurate representation of the use of each keyword, utilizing unsupervised ML models to attempt categorizing the keywords by subject which would factor into context and provide more accuracy, and finally an application that would take as input a keyword or keyword phrase and return the earning potential and “score” of the input. This application was the original intention of this project, but due to time constraints and depth of study of the course, this outcome could not be reached. I intend to continue work on this project and hope to develop the application mentioned above to a relatively rigorous functional level.

**Sources:**

* <https://www.mondovo.com/keywords/most-asked-questions-on-google>
* Word Cloud code is from [kavgan/word\_cloud: Python word cloud library for use within Jupyter notebook and Python apps. (github.com)](https://github.com/kavgan/word_cloud)

**Appendix:**

Refer to attached Jupyter Notebook for code and graphs as well as attached PowerPoint for necessary information