Design Lab Report:

Cancer Blog Text Analytics

Team Member:

Xiaye Xu

Jingxue Feng

Xinjie Guo

Minghan Wang

Date: 06/19/2018

Table of Content

1. Introduction---------------------------------------------------1
2. Data Description---------------------------------------------1
3. Methodology-------------------------------------------------4
4. Result------------------------------------------------------------5
5. **Introduction**

In addition academic paper on cancer, blog is a popular tool for people to voice their personal experience, thoughts or even professional regarding different types and stages of cancer. The goal of this project is to analyze the cancer blogs and show the result in an applicable format. This is a text-based analytics and require python coding to realize the functions. The task is to generate the closest blog outcome from the dataset once new text is give. To accomplish this task, my team decided to use machine learning in text analytics field to calculate the most similar text.

1. **Data Description**

The dataset text file from 1594 blogs. The blogs contains different length and focus on cancer. The 1594 text files and each document is treated as one object, combined in a single text file to be processed. Text preparation process such as tokenization, stop words removal and stemming will be explained in the following section.

1. **Methodology**
2. Data preprocessing

* Tokenization
* Stopwords Removal
* Stemming

1. Model Building

* Build Doc2Vec Model
* Train the model
* Get output

1. GUI Building

* Using Tkinter library, build the user interface

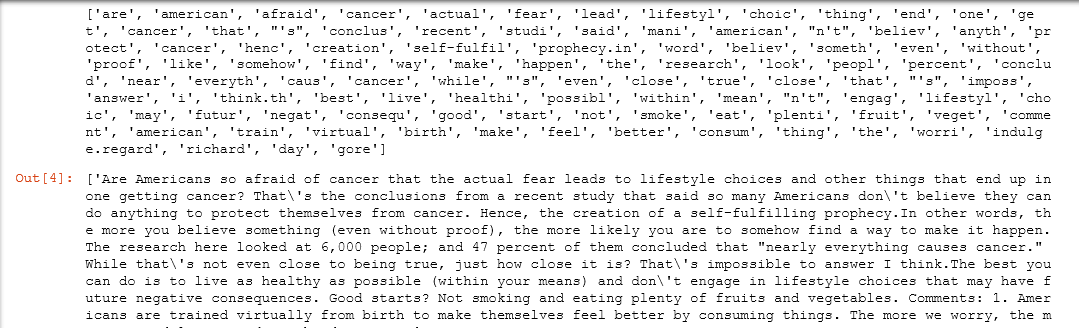


Figure 1 Result of stopwords removal and stemming for one blog

**Model Building**

As the goal of the project is to build a decision support application, the output of the application should have the closest content regarding to the query that user inputs. The best model that accomplishes such kind of task is the Doc2Vec model.

Doc2Vec model is largely developed based on another widely used model, the word2vec model. The word2vec representation is created using 2 algorithms: Continuous Bag-of-Words model and the Skip-Gram model.

CBOW model creates a sliding window around the current word, to predict it from the surrounding words. And each word is represented as a feature vector which, after training, become the word vectors.

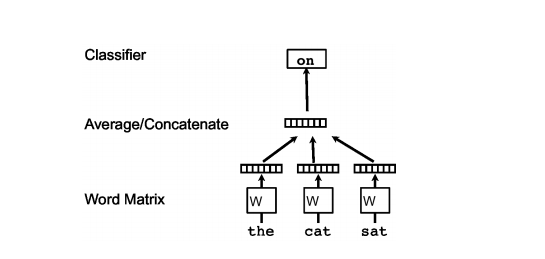


Figure 2 CBOW Model

Similar to word2vec model, whose goal is to create a numerical representation of word, doc2vec model’s goal is to create a numerical representation for each document. The idea, similar to CBOW model, is to add another vector. Instead of using just words to predict the next word, another feature vector is also added, which is document-unique. Then while the word vectors W are being trained, the document vector D is trained as well, and in the end of the training it holds a numeric representation of the document which is a vector. And this is called the PV-DM model.

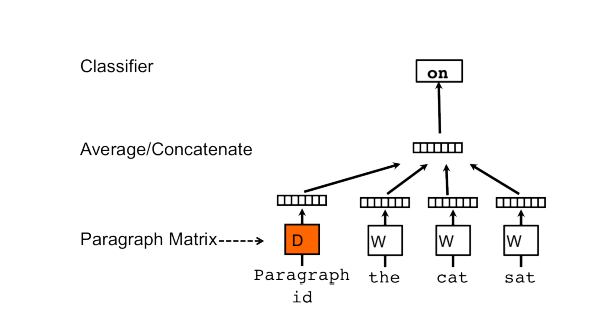


Figure 3 PV-DM Model

After getting a numeric representation for each document, the same process was done on the query data, and a numeric representation of the query data was created. Then by computing the cosine similarity between a simple mean of the projection weight vectors of the given words and the vectors for each word in the model, one can generate the most similar document regarding to the query data which in our project, generates the most similar cancer blog according to the query that user searches.

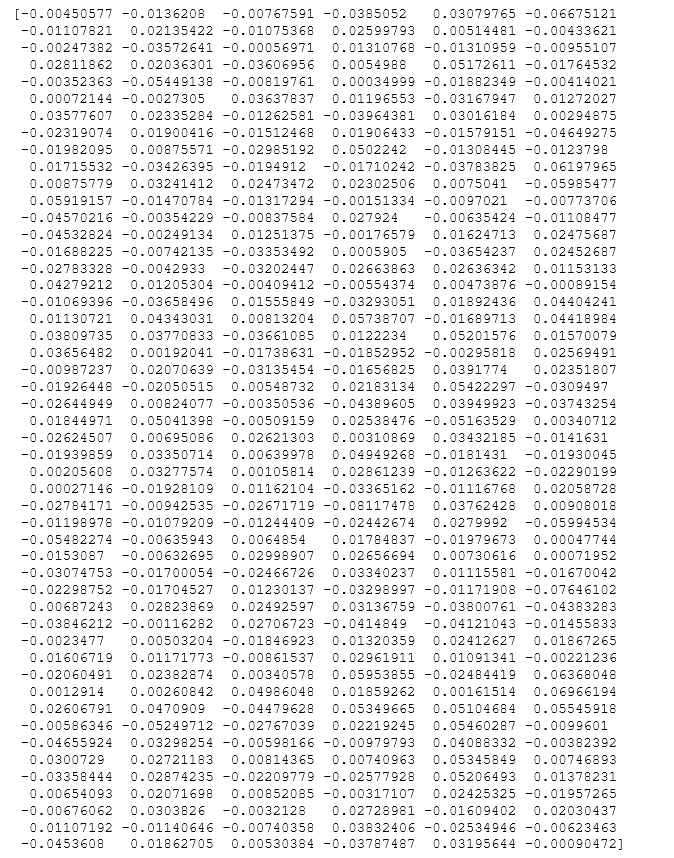


Figure 4 Vector representation of “Breast cancer cure”

Above is the vector representation of an example query “Breast cancer cure”.



This is the two most similar cancer blogs regarding to the query “breast cancer cure”. The first index in the tuple is the index of the blog in the list data structure that stores the blogs data. And the second index is the numeric representation of the similarity.

**Building the GUI**

Using Tkinter library in python, a graphical user interface was build. It is a much more user friendly interface comparing to generating outputs into the python console.

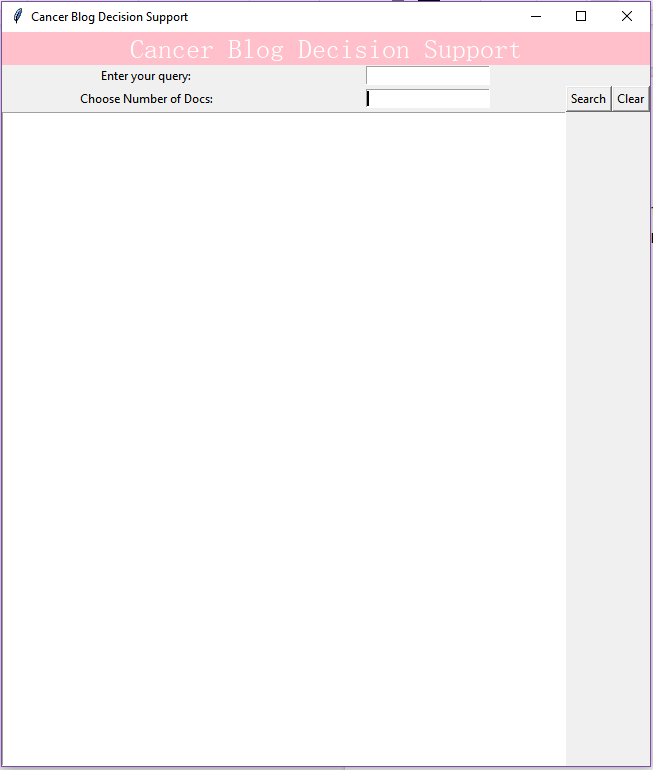


Figure 5 GUI of the APP

This is the layout of the user interface, where there are texts boxes for users to enter the query they want to search and the number of most similar blogs they want to retreive.

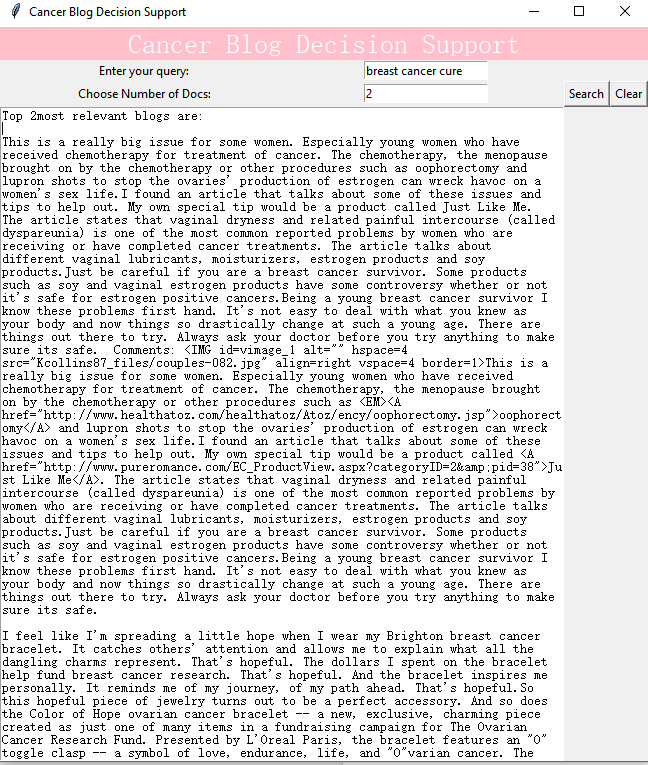


Figure 6 GUI with Example Output