NCBI Document Recommender

Andrew Y

Problem Statement

- As students, we are often tasked to write essays on subjects that we are not professional experts in.
- To combat this, we instead research work done by people that actually are practiced in the subject in the form of reading their articles / textbooks / blog pieces.
- However, finding relevant pieces of writings can still prove to be difficult with the abundance of research in the field.

Problem Statement

SOLUTION: Create an unsupervised learning model that would recommend research articles based on a given keyword.

Executive Summary

- The goal of the project is to ease the process of finding relevant research material.
- This is a project that takes a keyword from the user as an input, and then pulls data from the National Center for Biotechnology Information.
    - Specifically from their database dedicated to hosting research papers and articles relating to the biology field.
- The data comes in a format of rows being individual, unique documents and the columns describing the document's metadata.
    - Features include data such as title, author, key terms, abstract, publication date, etc.
- The data will then be cleaned and prepared for modeling.

Executive Summary

- By using a clustering model, the program will group the list of documents into a pre-designated amount of clusters.
- Afterwards, the model will select the most referenced document in each cluster to ensure the list is made up of a number of distinct articles.
- Once everything is selected, the user will receive a list of document titles to use for their own purposes.

Related Work

- Will be able to reference the world famous Google search function.
- However it will be much smaller in scope, and be focused on recommending documents from the National Center for Biotechnology Information, or NCBI
- The NCBI has a database dedicated to papers relating to
    - Bioinformatics
    - Biomedicine
    - Biotechnology
- The NCBI database does have an existing search function, but their sorting is based on article order, publication date, journal number, or PMC live date.
    - Goal is to create a model that would recommend documents based on their contents.

Proposed Work

- Main Task 1: Utilizing Entrez
    - Entrez is NCBI's tool for data extraction, needed to read the documentation to learn how to utilize it for data scraping.
- Main Task 2: Data Cleaning, Wrangling, and Preprocessing
    - There are plenty of metadata features in the data we do not need, so can be dropped.
    - Need to undergo typical text cleaning for NLP situations on the features we do want to keep.
    - Run the data through Count Vectorizer and TF-IDF Vectorizer in preparation for modeling.
- Main Task 3: Data Exploration and Visualization
    - Create Count Plots, Bar Plots, and Distribution Plots to view the popularity of certain groups or terms.

Proposed Work

- Main Task 4: Constructing Supervised Regression Model to impute missing reference numbers.
    - Reference numbers are how often a document is referenced by other articles.
    - Will be used as a metric for document quality, though many documents are missing it.
    - Supervised model will be used to calculate it based on trends and patterns.
- Main Task 5: Feed document data into a clustering model to create multiple clusters .
    - Goal is to group the documents into N distinct clusters, so we can have N distinct documents to pull.
- Main Task 6: Pull desired documents from the clusters.
    - Will pull the document with the highest reference number from each cluster.

Evaluation

The first model involved utilizing a Supervised Regression model to input missing reference numbers for documents.

- Utilized the evaluation metric Mean Poisson Deviance due to its specialization in count data.
- Tested both Count Vectorizer and TF-IDF Vectorizer as inputs.
- Tested inputs on the models Linear Regression, Poisson Regression, Decision Tree Regression, and Random Forest Regression.
- Poisson Regression with TF-IDF inputs had the best performance, which was further boosted by GridSearchCV.

Evaluation

The second model was a clustering model used to form a list of documents to return to the user.

- Utilized the evaluation metric Silhouette Score as the priority was to have distinct clusters.
- Tested Scaling, Normalizing, Scaling then Normalizing, Normalizing then Scaling, and Normalizing then Scaling then SVD the inputs.
- Tested both K-Means and Hierarchical Clustering
- K-Means Clustering with Normalizing then Scaling then SVD the inputs had the best score.

Example Output

Example of the output when looking for 10 documents with the keyword ‘Hay Fever’

```
[Epidemiology of respiratory allergies: current data].
Allergic Rhinitis and its Impact on Asthma (ARIA) 2008 update (in collaboration with the World Health Organization, GA(2)L
EN and AllerGen).
Intranasal corticosteroids for allergic rhinitis: superior relief?
Allergic rhinitis and its pharmacology.
Intranasal fluticasone propionate. A reappraisal of its pharmacology and clinical efficacy in the treatment of rhinitis.
Safety and tolerability profiles of intranasal antihistamines and intranasal corticosteroids in the treatment of allergic
rhinitis.
Pharmacologic rationale for treating allergic and nonallergic rhinitis.
Intranasal azelastine. A review of its efficacy in the management of allergic rhinitis.
The bidirectional capacity of bacterial antigens to modulate allergy and asthma.
Cysteinyl-leukotrienes and their receptors in asthma and other inflammatory diseases: critical update and emerging trends.
```

Timeline

First milestone: Successfully pulling data, and finish data wrangling

Second milestone: Create a metric for finding if a document is significant

Third milestone: Create a method of returning a list of significant documents

Fourth milestone: Implement the models

Fifth milestone: Evaluate and Improve Performance

Discussion

Challenges

- Initial model for finding documents could not be evaluated properly
- Had to redo inputs for clustering as was having runtime issues
- Initial method for choosing documents proved to be redundant
- Overall runtime still too long compared to normal search engines

Lessons Learned

- Sanitizing, scaling, normalizing, matrix factorizing inputs makes a significant impact.

Conclusion

Although code has not been streamlined yet, the project is in a usable state.

Key Findings

- Recommender systems are difficult to evaluate when there is no ground truth or feedback to build off of.
- Linear Regression models can reach negative score if the fit to the data is horrendous enough.

Future Work and Ideas

- Find ways to further reduce runtime
- Extend data cleaning section to receive data not just from NCBI but other sources too
- Include other metadata such as Authors or Key Terms
- Refactor code and turn this into an actual application for ease of use

References

Entrez's Bio Python Package

- https://biopython.org/docs/1.76/api/Bio.Entrez.html

Github Link

- https://github.com/awyeh64/NCBI-Document-Recommender