Problem Statement 2 : Associating pmid tags

Overview of PMID Tags:

- A PMID, also known as the PubMed reference number, is a number assigned by the National Library of Medicine to papers indexed in PubMed
- A PMID is the unique identifier number used in PubMed for each article.
- The PMID is assigned to each article record when it enters the PubMed system.
- An "in press" publication will not have one unless it is issued as an electronic pre-publication.
- The PMID# is always found at the end of a PubMed citation

Approach: [Rough Draft]

Get PubMed Data

- Download XML from NLM FTP. Approx 200GB. Benefits: all the data all the time. Negatives: with addition of new publications the list becomes out of date quickly, requires a lot of memory
- API. Slow. Doesn't require server space. Can't get everything.
- Pubrunner
- EDirect Local Data Cache

Extract useful metadata from XML

- Metadata was extracted from PubMed XML files using python's lmxl model. The following features were extracted:
  - PMIDs – Considering to go only with this ☺
  - Title
  - Abstract Text
  - MeSH Major Headings
  - MeSH Subheadings

- Keywords

- The title and abstract are concatenated together, this is for older PubMed records with no abstract, and then are cleaned for processing. The cleaning process includes tokenizing, lowercasing, stemming and the removal of stop words. ** Note: full text was not included in this extraction. **

NLP Modeling on Title and Abstract

- TF-IDF is run on both Abstract+Title
- Document embeddings (Doc2Vec) are run on documents for features to measure similarity across documents
- Latent Dirichlet Allocation (LDA) is run for Topic Modeling purposes on Title and Abstract

NLP Modeling on MeSH terms

- TF-IDF across MeSH terms to find co-occurence
- Word embeddings (Word2vec) are run across the entire corpus of words

Retrieve most similar documents to initial ones provided

- Determining document similarity through NLP feature vectors and similarity metric (e.g. cosine similarity)
- apply a nearest-neighbors approach (K-D Tree) to retrieve the most similar documents to initial list of PMIDs
- Retrieve documents using pre-set number PMIDs (e.g. n=200)

Test against known PMIDs

- Use a corpus of a previously performed systematic review to validate results (Gout)
- Embed relevant articles (~300) among a larger semi-random corpus of ~10,000 other PubMed abstracts
- Measure performance through standard measures of precision and recall

Map MeSH & Keyword Strings associated with newest retrieved documents

- Using the newly retrieve PMIDs, map to the associated MeSH terms and identify terms most relevant to the original search documents

- Apply a similar nearest-neighbor approach as done with PMIDs abstracts, but now retrieving new MeSH terms
- Traverse the hierarchical structure of MeSH ID strings and subset terms based on their similarity (i.e. Levenshtein Distance)
- Combine the results of both approaches (nearest neighbor search, MeSH string matching) to form a final subset list of MeSH terms

Create Shortest Precise Search String

- Penalize longer search strings - apply higher weight to MeSH terms deeper in MeSH tree
- Sort and prioritize final MeSH terms based on frequency found within retrieved PubMed articles

*****END*****

Using Metapub 0.4.3.6,a Python library that provides python objects fetched via eutils that represent papers and concepts found within the NLM[National Library of Medicine].

```
from Bio import Entrez

Entrez.email = "bharathkumar@live.co.uk"

for single_term in ["Pre Frontal Cortex", "Posterior Parietal Cortex", "PFC"]:

    data = Entrez.esearch(db="pubmed",term = single_term)

    res = Entrez.read(data)

    pmids = res["IdList"]

    print 'PMIDs for %s: %s' % (single_term, pmids)


from metapub import PubMedFetcher

fetch = PubMedFetcher()

# get the first 15000 pmids matching "Pre Frontal Cortex", "Posterior Parietal Cortex", "PFC" keyword search

pmids = fetch.pmids_for_query('breast neoplasm', retmax=15000)
```