# Text Mining Born Digital Archives As Part of Data Science@NLM

Teleworking during the pandemic in the fall of 2019 provided me an excellent opportunity to participate in the NLM Data Science Fundamentals course. NLM describes this 12-week/60-hour intensive training course as

an all-virtual, rigorous, hands-on course intended to support and challenge you in developing data science skills. Specifically, this course will provide you with technical skills in computational statistics, data visualization, data modeling, and machine learning that will allow you to extract knowledge and insights from large, complex datasets and make better informed decisions.

I was one of 25 classmates from across a broad spectrum of NLM staff that met for 3 hours twice a week via Zoom who through theory, demonstration, and independent practice used [Jupyter Notebooks](https://jupyter.org/) to learn and be tested in the [Python programming language](https://en.wikipedia.org/wiki/Python_(programming_language)) for data science, exploratory data analysis, data modeling foundations, and machine learning, culminating in a capstone project of our choosing. As our instructor noted, we were indeed scientists wherein the Jupyter Notebook was our laboratory where we applied the scientific method of hypothesizing, experimenting, and testing to reach measurable and verifiable conclusions about data.

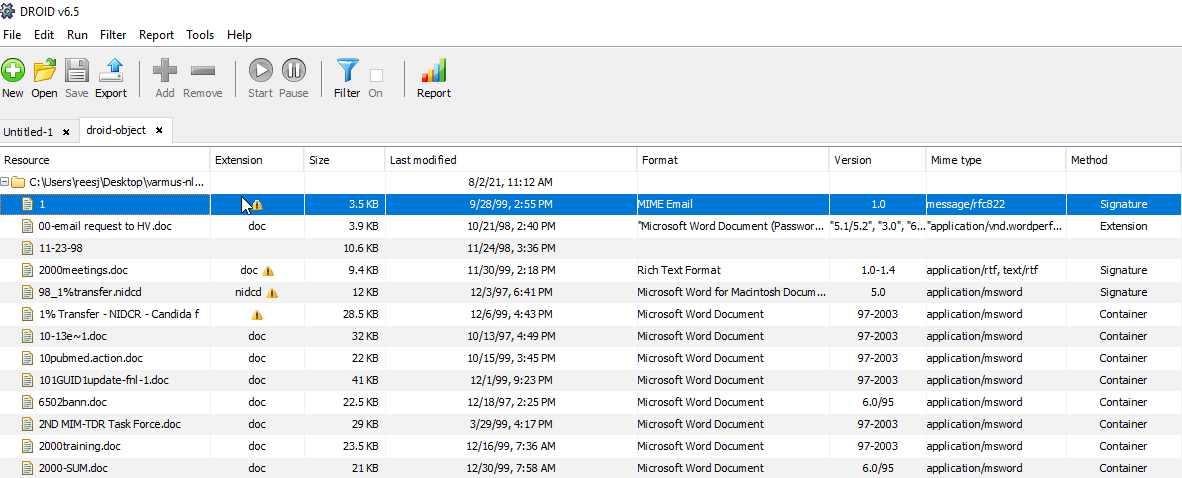
My goal for the class was to learn about [natural language processing](https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1) (NLP) techniques and bring them to bear across several areas of the archival enterprise such as improving back-office collections processing and automated collections description, but also using machine learning to perhaps better understand and contextualize our existing finding aid descriptions and the full texts of our digitized collections. I had visions of applying natural language processing text mining techniques that, magically, would reveal new knowledge about the topics, people, and places locked away in the bits and bytes of data we’d never have the time to read through ourselves as processing archivists or users.

Using the tools learned from my [Fundamentals Course capstone project](https://github.com/John-Rees/ds-class-prezi/blob/main/jrees_unit4_prezi_copy.ipynb), I conducted an experiment performing some common [text mining](https://en.wikipedia.org/wiki/Text_mining) tasks -- topic modeling, keyword extraction, and named entity recognition -- against a corpus of born digital documents from our personal papers collection of [Dr. Harold Varmus](https://profiles.nlm.nih.gov/spotlight/mv), former NIH and NCI director. My hypothesis is the results will aid in contextualizing the content to help the processing archivist make selection decisions and enhanced collection description for the finding aid. Let’s take a walk through that experiment (find the Jupyter notebooks [here](https://github.com/John-Rees/NLP-experiments)).

# Data Preparation

Understanding and preparing the data set is often the most challenging and time-consuming activity of a data science experiment. In most cases the data scientist is not the domain expert, meaning we have no contextual knowledge about our data. Dr. Varmus donated a copy of his backed-up computer files on CD-ROM, one of which was a directory titled “Files from Office” consisting of 1,266 files dating between 1996-2000. Our data preparation step reveals one large problem at the outset – what file types are these? We need to know this in order to decide which data extraction tools to use – we can’t just process files randomly. There are few file extensions like .doc or .xls which we now come to expect to identify the creating application. The lack of normalized file extensions is quite common in pre-2000 desktop computer operating systems.

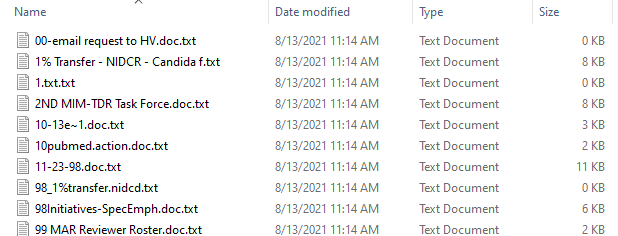
We use a format identification tool named [DROID](https://www.nationalarchives.gov.uk/information-management/manage-information/policy-process/digital-continuity/file-profiling-tool-droid/) to package our born digital collections for just this purpose.



*Figure 1*

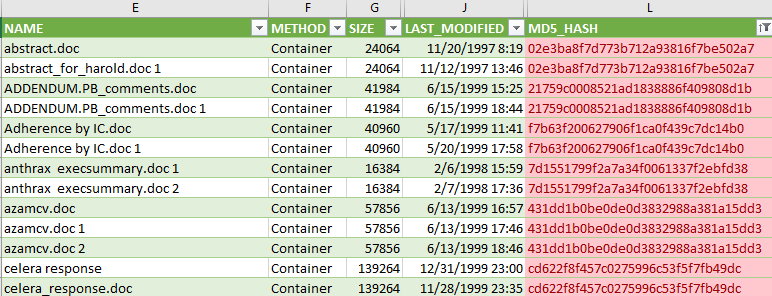
Here we identified that the majority of the files are indeed a mix of 1995 and 1997 Microsoft Word versions.

This also tells us we can’t use text extraction tools like [Word2Text](https://github.com/rthomascloud/word2text). MS Word .docx documents created with version 2000 and after are UTF-8 XML files under the hood. It’s relatively simple to use Doc2Text in a Jupyter Notebook Python code cell to strip away the XML tags and leave the unformatted text behind. Pre-2000 MS Word files are binary files, meaning they have to be de-compiled and reassembled into ASCII or UTF-8 characters to be interpreted as plain text. To solve this problem we used Apache Tika to walk through the Files From Home directory and convert the files to plain text, but unfortunately not without error. Tika could not always interpret the binaries resulting in several hundred empty text files and many incompletely processed ones.



*Figure 2*

We also suspiciously notice many similar or repeating files names – could these be duplicates? We don’t want our NLP results biased by duplications. Generating [cryptographic hashes](https://en.wikipedia.org/wiki/Cryptographic_hash_function) using DROID for each file reveals that, yes, there are over 600 duplicate files even though they may have different file names. In these cases we kept either the largest file or the latest creation date file.

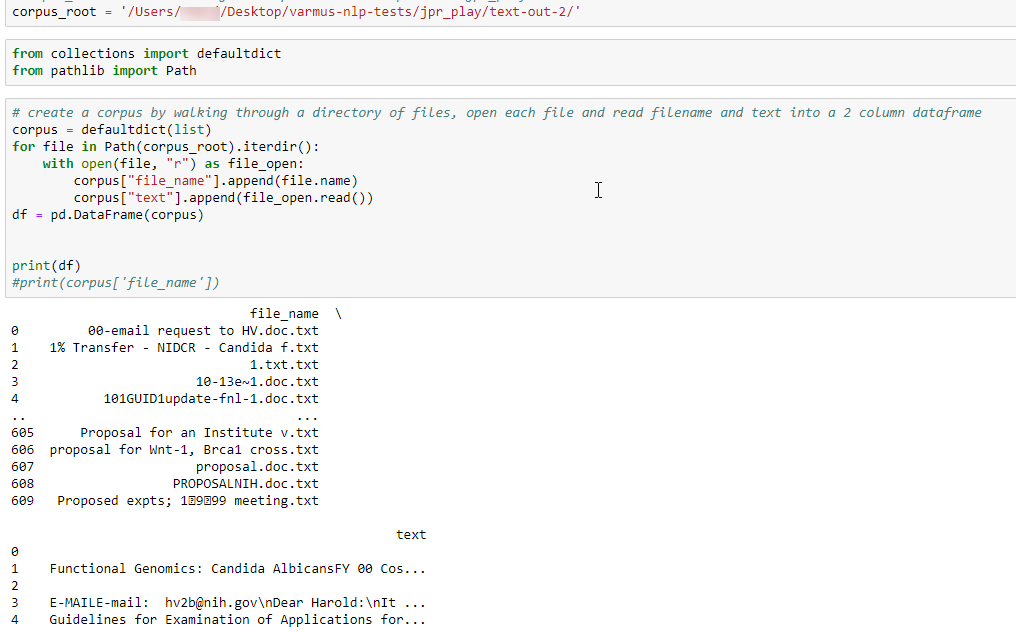


*Figure 3*

We’ve trimmed our data set to 610 unique files. Now the fun can begin!

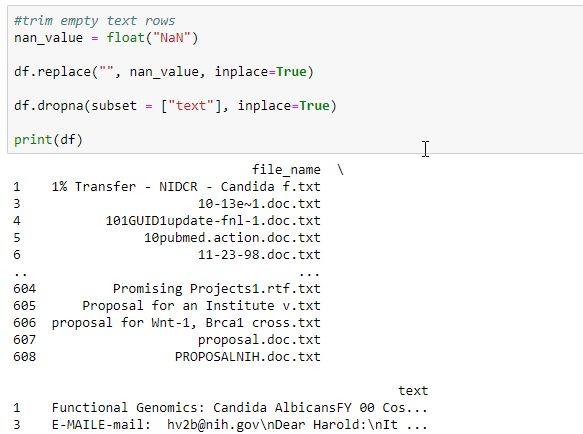
# Exploratory Data Analysis (EDA)

Our first step is to load our texts into a dataframe our tools understand and perform some initial data investigations, or EDA, to spot patterns or anomalies, check assumptions, see our data at a high level with the help of summary statistics and graphical representations.

Our dataframe has a column for the file name and one holding the extracted text  


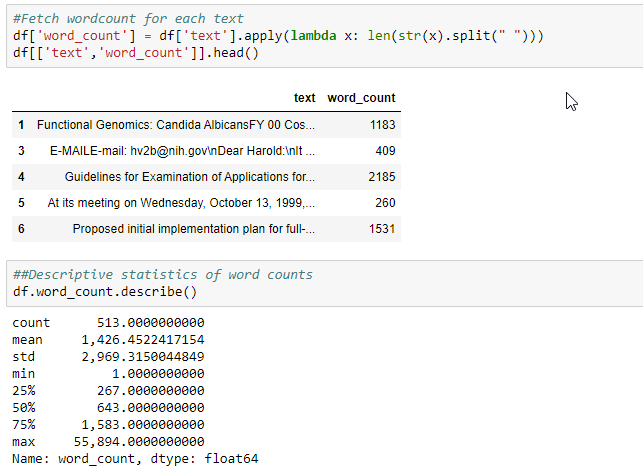
*Figure 4*

Let’s exclude those empty text files



*Figure 5*

And run some basic statistics



*Figure 6*

We’ve trimmed out corpus to 513 documents with an average word count of 1,426 per document. The smallest document has 1 word; the largest 55,894 – quite a spread!

Word clouds are popular visualizations, though the visual communications literature says they aren’t actually very helpful communicating information. We’ll use Wordcloud and apply a few different stop word dictionaries to filter out commonly occurring parts of speech like articles, prepositions, and pronouns.



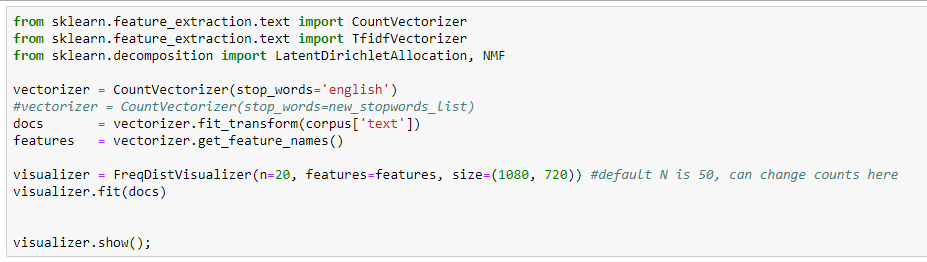
*Figure 7*



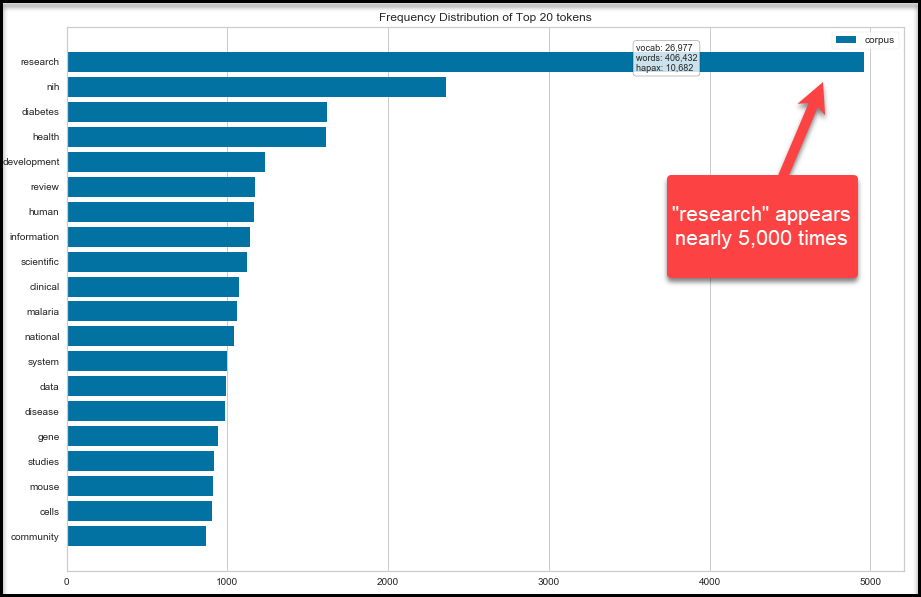
*Figure 8*

The word cloud is not especially helpful. All the words have similar sizes, so none predominate. And the words themselves are pretty common, though “E Biomed”, “diabetes”, and “malaria” pop out a bit for me.

A bar chart of the top 20 words may prove more useful.



*Figure 9*



*Figure 10*

# What is Natural Language Processing?

At a high-level, NLP relies on converting words into numbers, called tokens, so we can apply math and statistics to them.

In this simple example we have three words -- animal, zoo, panda -- and an algorithm assigned them the tokens 0, 1 and 2, and put them into an array.

{animal,0; zoo,1; panda,2}

The algorithm can then do things like count the number of occurrences across the documents in our corpus – here there are 400 of the zero tokens, meaning the word animal occurs 400 times; zoo occurs 50 times, etc.

{0,400; 1,50; 2,10}

# Topic Modeling

Topic modeling is where we want to get an idea of what subjects (topics) we have in our dataset. A topic is nothing more than a collection of words that describe the overall theme. For example, in case of news articles, we might think of topics as politics, sports etc.

Topic modeling won’t directly give you names of the topics, but rather a set of most probable words that might describe a topic. It is up to us to determine what topic the set of words might refer to.

Topic models are built around the idea that the semantics of our document are actually being governed by some hidden, or “latent,” variables that we are not observing. As a result, the goal of topic modeling is to uncover these latent variables — topics — that shape the meaning of our document and corpus.

[Latent Semantic Analysis (LSA)](https://towardsdatascience.com/latent-semantic-analysis-intuition-math-implementation-a194aff870f8) is a statistical method for generating topics from words correlated to their context within a corpus. LSA uses simple averages -- the meaning of any word is an average of all the documents it appears in and the topic is a sum of the meaning averages. And words can generally only be associated with one topic. But words are not unambiguous. For example, the word "bank" used in the context of words like "mortgage", "loans" or "rates", probably refers to a financial institution. It could refer to a river bank when used together with "lures", "casting", and "fish". In order to find the meanings or concepts *behind* the words, LSA attempts to map both words and their contexts into a "concept space" matrix in which different meanings can be statistically compared.

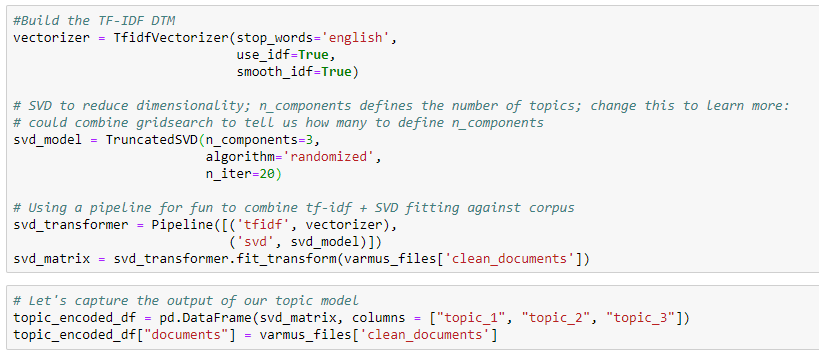
Using Pandas again, let’s create a new dataframe and further normalize our Dr. Varmus texts:



*Figure 11*

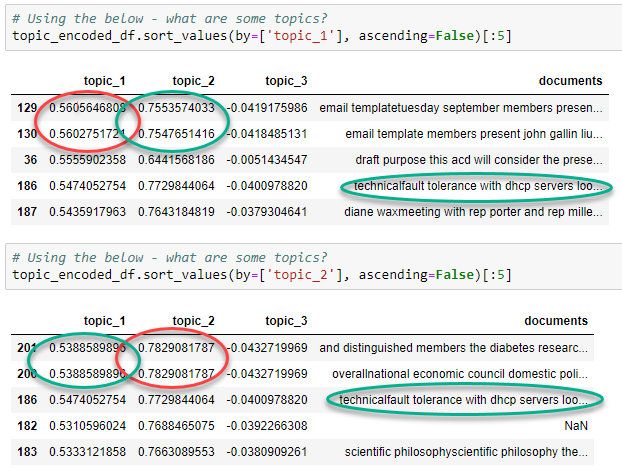
Let’s create our document/term matrix using Term Frequency – Inverse Document Frequency algorithm. Each row represents a word and each column a document, and the cells of the matrix contain the word frequencies where rare words are weighted more heavily than often used words. Terms are reduced to their stem and stop words are excluded.

We’ll also apply the Singular Value Decomposition (SVD) algorithm to further reduce dimensionality of the TFIDF-matrix. SVD emphasizes the strongest relationships of words and documents and discards the unneeded noise, thus reducing bias. Iterating over the corpus 30 times produced more meaningful topics than just one pass. Three topics is generally a good starting point, but we can add or subtract *n\_components* as we like:



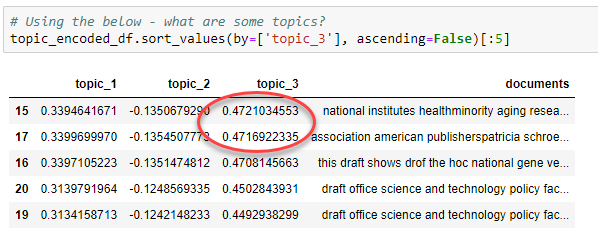
*Figure 12*

In the resulting chart we see three topics represented by the chunks of words from our Dr. Varmus cleaned texts on the right. The topic columns show the confidence value metrics ranging from 0 to 1. So we can say 2 highest ranked documents for topic 1 contain words like “email”, “members” and “john gallin”. The 2 highest ranked documents for topic 2 have words like “diabetes” and “economic council”. However, we also see the same topics have strong correlations to each other in that documents score high for both topics 1 and 2. We even see one document belonging to both topics even though that probably shouldn’t happen based on the TF-IDF modeling rules. This leads me to believe there aren’t enough distinguishing features in this corpus to surface any meaningful topic categories. But maybe the processing archivist can make some inferences about what these topics might be based on what they know about the paper contents of the collection.



*Figure 13*

Our overall confidence values plummet for topic 3, however the actual words from each document might be the most interesting from a collections/historical research perspective -- “minority aging”, “association of american publishers”, “draft office of science and technology policy”. The processing archivist might want to explore these in more detail across both the digital and paper portions of the collection.



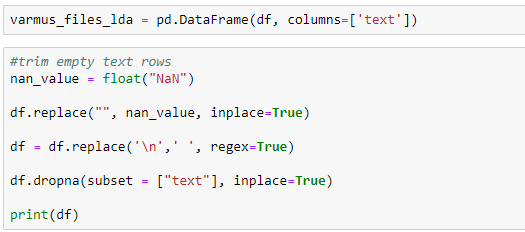
*Figure 14*

# Latent Dirichlet Allocation (LDA)

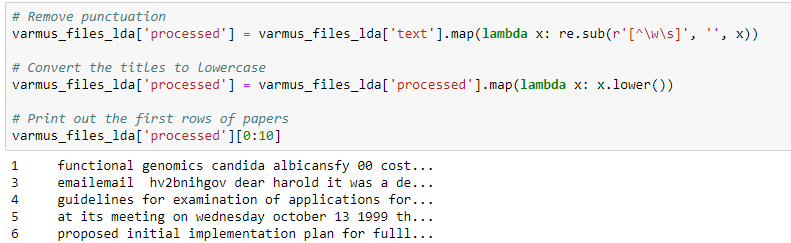
[LDA](https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2) is a model that uses probabilities rather than LSA’s simple averages. LDA allows multiple words and documents to be associated with multiple topics and introduces fuzzy logic. It builds a topic per document model and words per topic model, modeled as distributions. Each document is modeled as a distribution of topics and each topic is modeled as a distribution of words. LDA assumes that every chunk of text will contain words that are somehow related. It also assumes documents are produced from a mixture of topics. Those topics then generate words based on their probability distribution. Topic probabilities provide an explicit representation of a document.

We can apply more sophisticated NLP tools as well, such as [spaCY](https://en.wikipedia.org/wiki/SpaCy). spaCY is an advanced natural language processing application that automatically performs low-level operations on a corpus like tokenization and is intelligent enough to ignore punctuation, can find if the word is an entity such as a company, place, building, currency, institution, and label words as parts of speech like nouns, verbs, and adjectives. spaCY also uses its lemmatization factory to reduce the feature space. [Lemmatization](https://en.wikipedia.org/wiki/Lemmatisationhttps:/en.wikipedia.org/wiki/Lemmatisationhttps:/en.wikipedia.org/wiki/Lemmatisation)is the process of grouping together the inflected forms of a word so they can be analyzed as a single item, identified by the word's [lemma](https://en.wikipedia.org/wiki/Lemma_(morphology)), or dictionary form. For example, the word "better" has "good" as its lemma. Lemmatization also accounts for a word’s meaning based on its part of speech and context and usage within its sentence, surrounding sentences, or an entire corpus.

First we’ll create our dataframe as before and perform the same data cleanups:

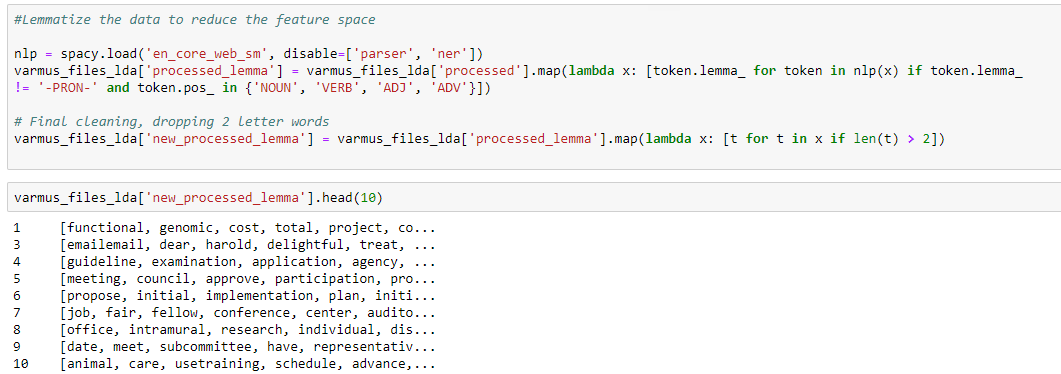


*Figure 15*



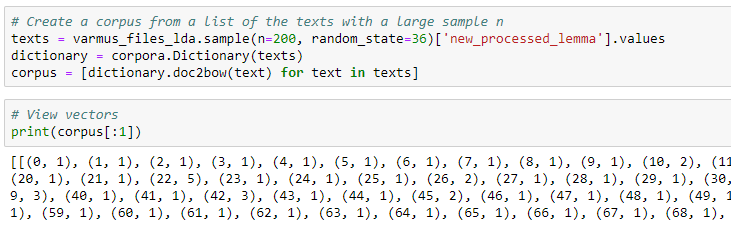
*Figure 16*

We’ll load spaCY, and lemmatize the feature space with some output to see preliminary results. We get a small glimpse of the lemmas – ‘genomics’ in document 1 is now ‘genomic’; document 6 ‘proposed’ is now ‘propose’, stop words like ‘at’ and ‘its’ are removed, etc.



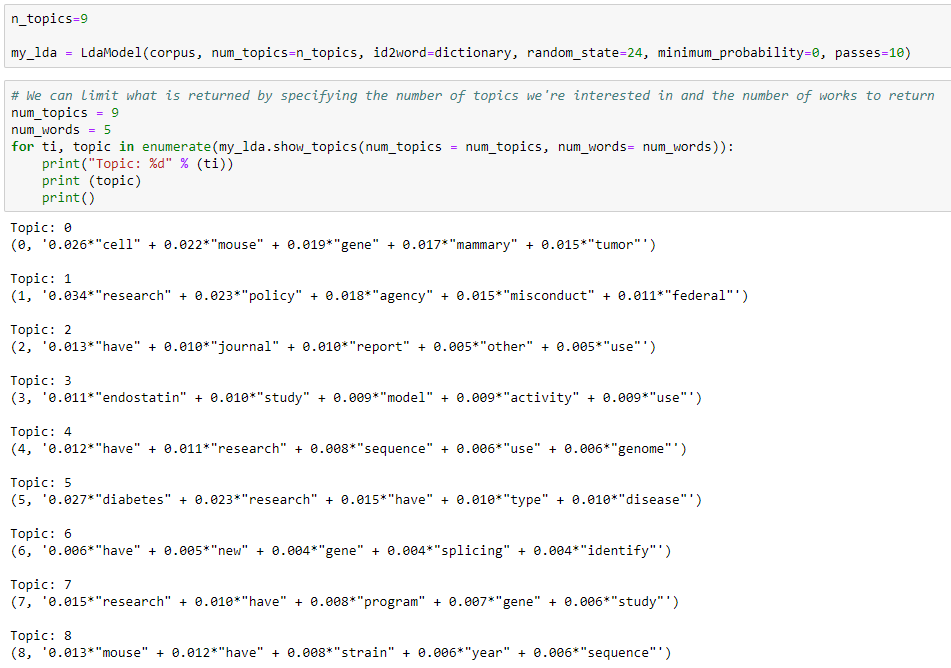
*Figure 17*

Let’s create our corpus using the lemmatized documents and apply LDA. We can see the vectors created for document 1:



*Figure 18*

Let’s not forget we’re seeking some topic modeling magic! We don’t want too many to start with – six to ten is a good industry standard starting point. We’ll generate nine topics consisting of five words each from across the entire corpus. Let’s run the LDA model 10 times to get a good sampling. Our result will show the words associated with each topic along with a numeric saliency score between 0 and 1.



*Figure 19*

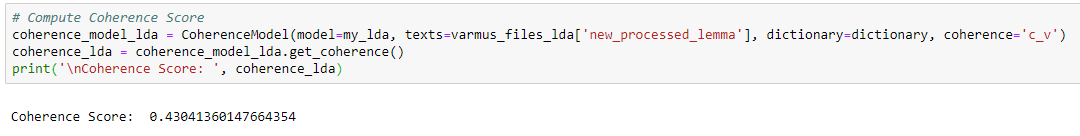
We see some familiar groupings like ‘diabetes research’, but now we see ’mouse’ associated with ‘mammary’ and ‘tumor’ and some new topics like ‘federal research misconduct policy’ and ‘genome sequence research’.

We can also validate the results with some statistical rigor by computing coherence scores – how can I prove the topics are valid apart from using my eyeball test. Coherence measures score a single topic by measuring the degree of semantic similarity between high scoring words in the topic. These measurements help distinguish between topics that are semantically understandable versus topics that are mere statistical inferences (unintelligible to a human).

LDA c\_v coherence is one method. On a scale of 0-1, a rule of thumb for understanding whether a c\_v is good or bad is:

* .3 is bad
* .4 is low
* .55 is okay
* .65 might be as good as it is going to get
* .7 is nice
* .8 is unlikely and
* .9 is probably wrong

Let’s generate a base-line c\_v on our lemma corpus:

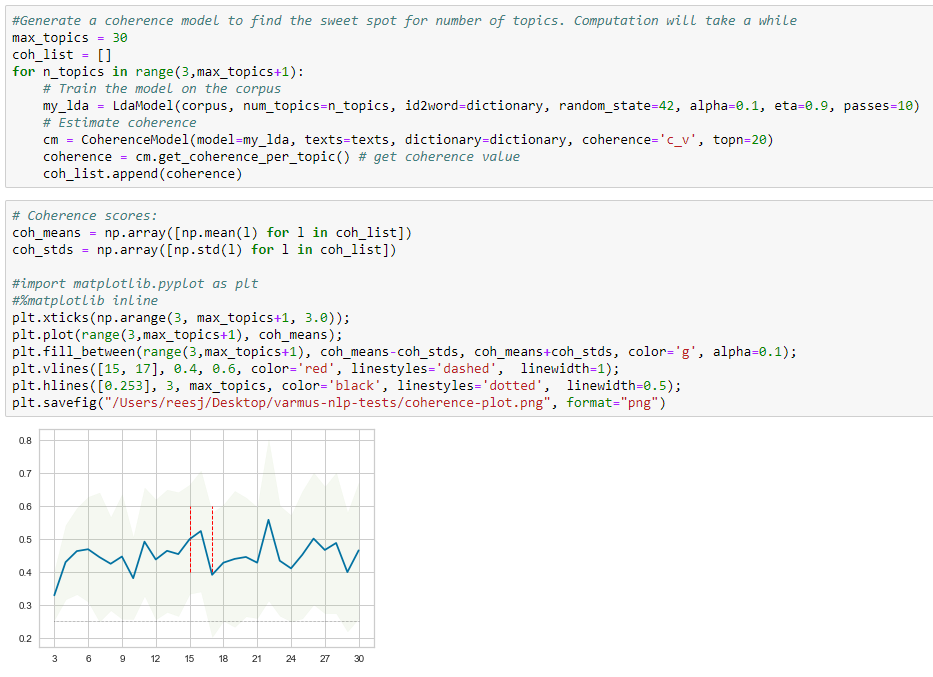


*Figure 20*

0.43 is low, meaning it will be hard for us humans to discern semantically useful topics from the list of topics.

How can we increase our coherence score? We can get better data, or adjust our model’s parameters like the number of topics and the density of documents and words per topic (alpha and beta).

We can programmatically build and test multiple models to find the optimal topic number. Choosing a ‘k’ that marks the end of a rapid growth of topic coherence usually offers meaningful and interpretable topics. Picking an even higher value can sometimes provide more granular sub-topics.



*Figure 21*

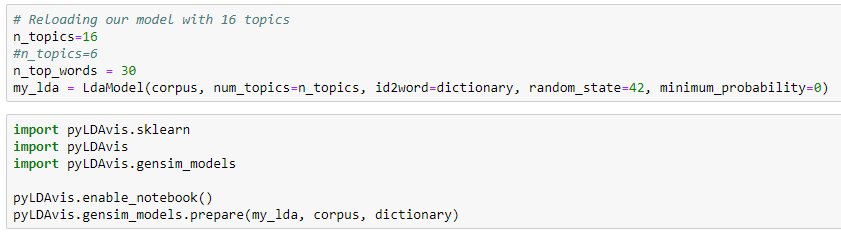
Looking at our test plotted on a chart it looks like either 16 or 22 topics will get us the most bang for our buck, but our highest coherence will still only climb to the ‘just ok’ range of ~0.55.

Now that we have a trained model let’s visualize the topics for interpretability. To do so, we’ll use a popular visualization package, pyLDAvis which is designed to help interactively with:

1. Better understanding and interpreting individual topics, and
2. Better understanding the relationships between the topics.

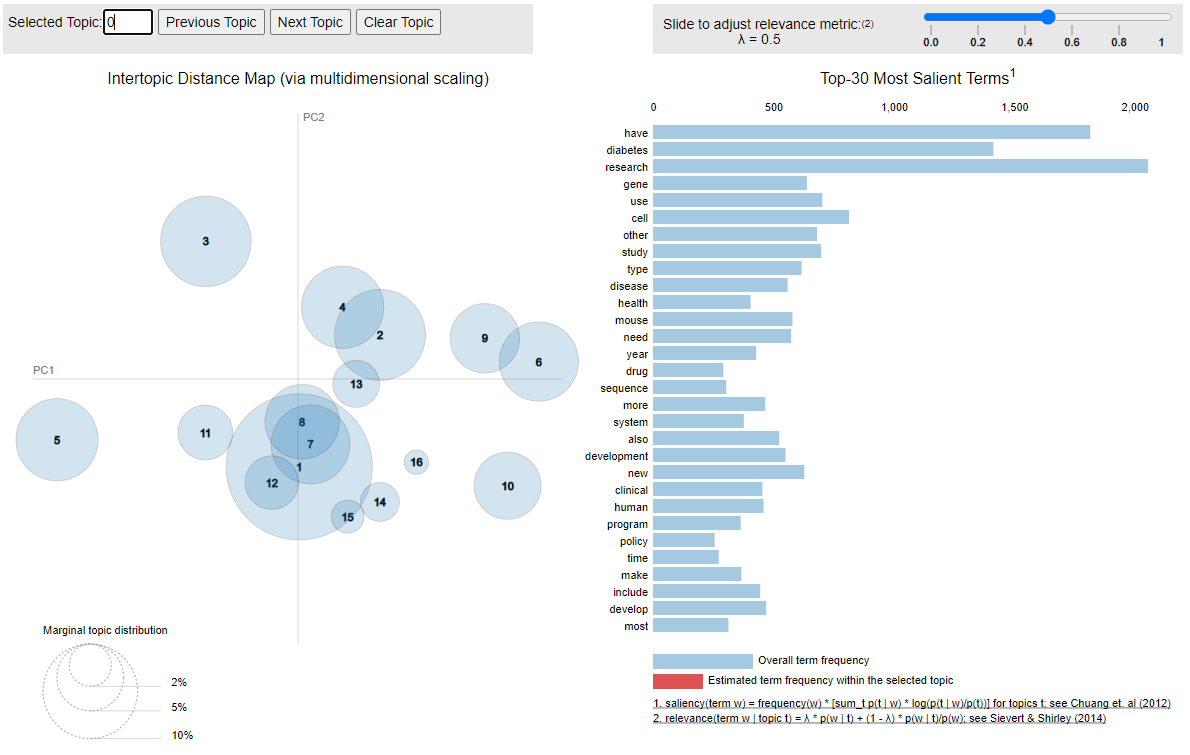
For (1), you can manually select each topic to view its top most frequent and/or “relevant” terms, using different values of the λ (alpha) parameter. This can help when you’re trying to assign a human interpretable name or “meaning” to each topic.

For (2), exploring the Intertopic Distance Plot can help you learn about how topics relate to each other, including potential higher-level structure between groups of topics.



*Figure 22*

<iframe>varmus-lda-16-topics.html</iframe>



In this interactive visualization of our data we see 6 topic bubbles on the left and the top 30 terms on the right.

The larger the bubble and closer it is to the center of the X-Y axis shows how meaningful the topic is, how many meaningful words make up the topic, and how relevant the topic is to the corpus. So it looks like topic 1, 7-8, and 12 are pretty similar since they overlap each other and are centered in our axis meaning they are highly relevant to the corpus. Bubbles 3 and 5 are medium-sizes, but sit by themselves on the axis so they contain some meaningful words. The topics aren’t similar and are less relevant to the corpus overall, but here we find our research misconduct topics. They are also located on the edges so maybe they will reveal some more topic fuzziness if we lay with the λ (alpha) parameter.

If we click each bubble we can see the terms used to define the topic on the right. The red bar shows how frequently the words appear in the topic. Clicking 1, 7-8, and 12 we see our familiar “diabetes research” topics, plus the new breast cancer topic. I also see ‘ebiome’ now, which through research I learned is actually ‘ebiomed’ before it was lemmatized, so some new knowledge! Ebiomed was the first idea for what became PubMed.

And we can increase the overall fuzziness to surface less frequently occurring terms by adjusting the λ (alpha) parameter that calculates the scoring weight for any given term used to generate the topics. A lower weight means the term is more unique to that topic. We can also scroll over the word list to see how valuable that word is across all the topics.

# Keyword Extraction

Topic modeling can be a bit of an unnecessary abstraction, especially now that we’re learning our corpus of Dr. Varmus’s texts are minimally related with few easily discernable major topics. It may be that simply extracting keywords is good enough. Generating [N-grams](https://en.wikipedia.org/wiki/N-gram), or word-grams, is one text mining technique. We can also use word stemming and lemmatization to reduce spelling variations and add similes. Adding bi-grams and tri-grams will also help add context. We’ll re-use the scikitlearn tool from our original EDA stage and add the NLTK NLP toolkit’s stemming tools Porterstemmer and Snowballstemmer, and WordNetLemmatizer. Perhaps we’ll get different results compared to the LDA lemmas?

Starting with our same root texts directory, let’s generate the same corpus, do some cleaning-up, and apply the stemmer and lemmatize. We ran both Porterstemmer and Snowballstemmer with little discernable difference, but we’ll stick with Snowball since it is the more modern tool:



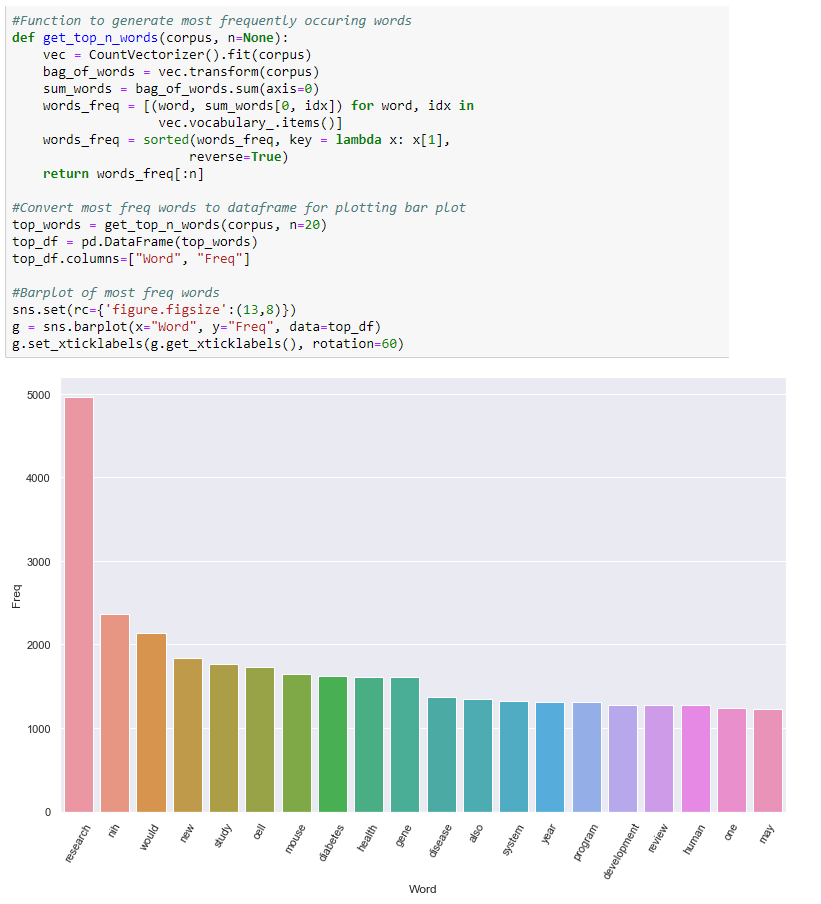
*Figure 23*

At this point we could generate a Wordcloud or perform some EDA, but let’s press on with the keyword extraction.

We’ll use scikitlearn’s Countvectorizer again against our stemmed and lemmatized corpus, this time as a function so we can automate the keyword generation. We can run these independently against any corpus. We’ll visualize our output with a bar graph each time.

## Single Words (Uni-grams)

We’ve generated uni-grams already in a few of our EDA and other exercises. We’ve determined uni-grams have limited utility in our use cases. In this example the results look familiar to past exercises. At least they serve to re-confirm the data produced. from previous our work.

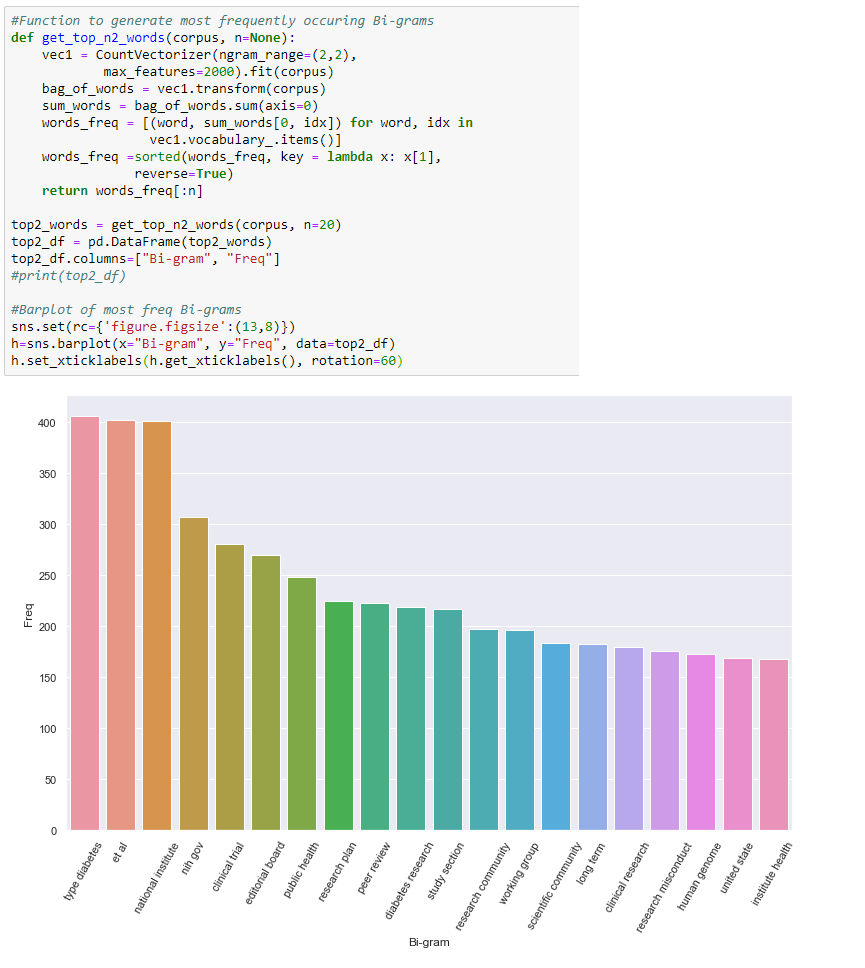


*Figure 24*

## Two Words (Bi-grams)

Bi-grams reveal more context and start to seem more like topics. They aren’t just the words before or after our uni-grams, either. Their generation depends on other words and phrases around them and their context within the documents across the corpus.

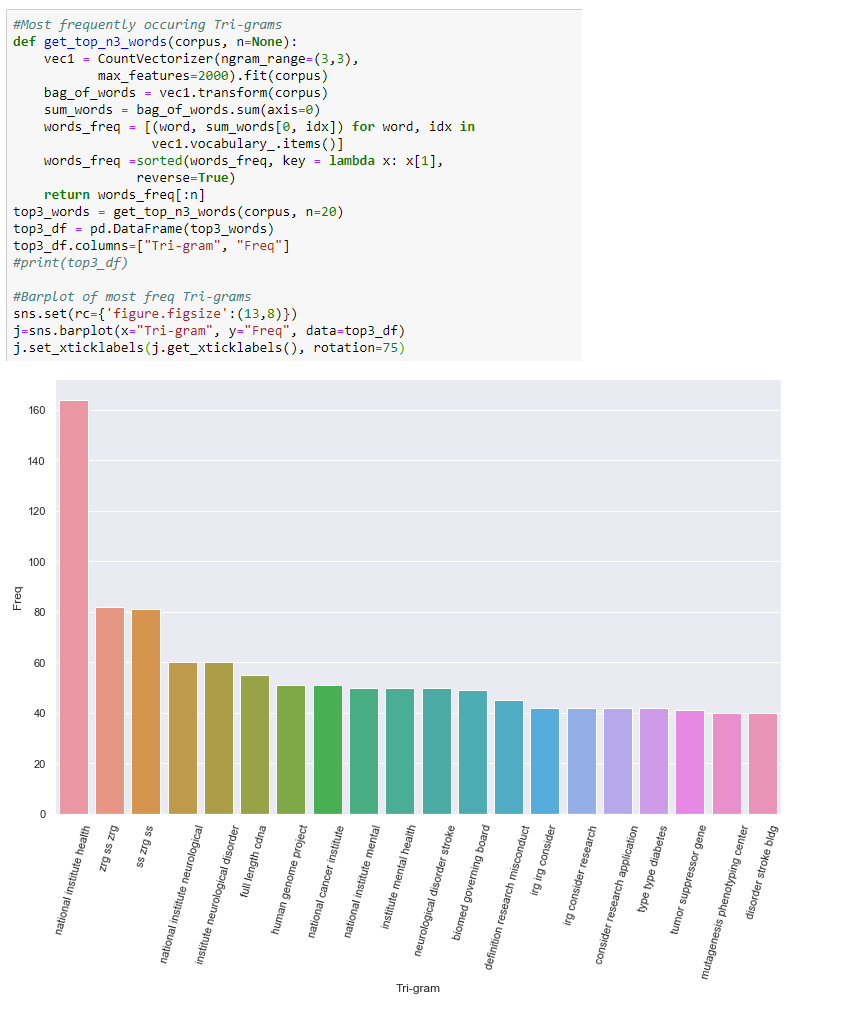
Bi-grams like ‘et al’ and ‘editorial board’ might be describing the research publications Varmus was working on and editing. ‘Research misconduct’, ‘diabetes research’, and ‘human genome’ reaffirm some of our LDA topic modeling results.



*Figure 25*

## Three Words (Tri-grams)

Like bi-grams, tri-grams are generated independently from our corpus. Here we see some problem data revealing again that our original conversion from Word documents text is imperfect. But we do see some NIH institute names again, demonstrating the preponderance of Varmus’s interaction with certain institutes and thus perhaps their relative importance to NIH or their research agendas. We also see that ‘human genome project’ and DNA research is an actual activity of significance different from the generic ‘human genome’, more affirmation that E-biomed is also an activity, and tumor cancer research is likely an important topic with the corpus.



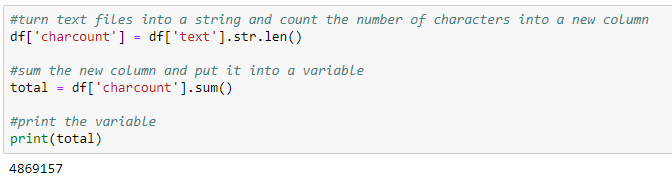
*Figure 26*

# Named Entity Recognition (NER)

Our last experiment is another common text mining task – extracting named entities like personal and organizational names, geopolitical entities, products, cardinal numbers, proper nouns, etc. using [NER](https://towardsdatascience.com/named-entity-recognition-with-nltk-and-spacy-8c4a7d88e7da) algorithms. We learned in our previous lemmatization exercise that spaCY brings some artificial intelligence to bear and can identify and tag parts of speech. We can use spaCY to also try and extract some NERs from our set of Dr. Varmus’s office files and add more context to our understanding of them. Perhaps this will also surface some names of people, places, things, and organizations to explore as potential subject terms for our processing archivist to use in their cataloging and collection description work.

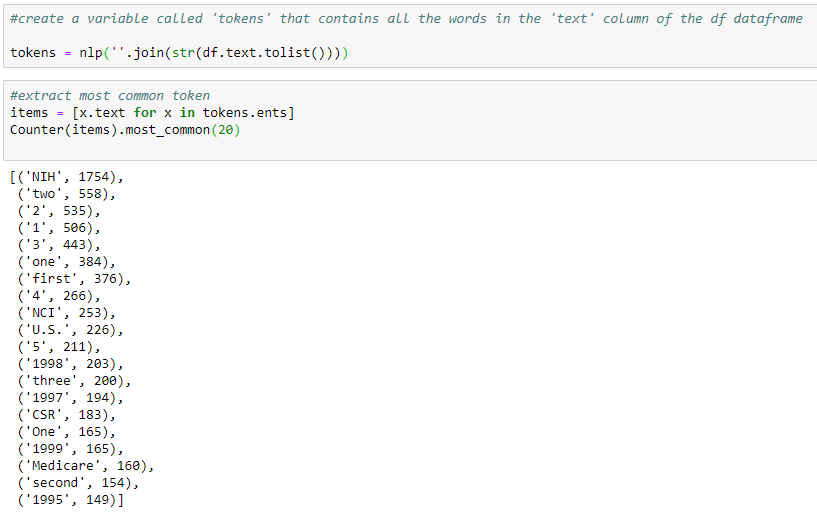
Let have spaCY generate a set of tokens from our same set of 513 processed texts. spaCY by default can only process 1 million characters and can be computationally expensive. Recall from our initial EDA our word count was well over 400K, so I imagine our character count will also be well over 1M. We’ll probably need to increase spaCY’s maximum character length close to our corpus’s total and hope our PC’s RAM can handle it.

We can easily count and sum all the characters. It turns out we have over 4.8 million:



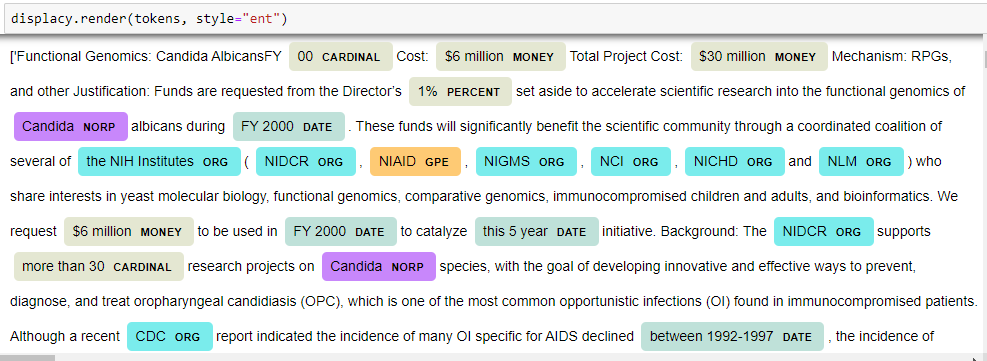
*Figure 27*

Now let’s have spaCY parse and extract NER’s from our texts and print out the 20 most common to check the work. This will take a while:



*Figure 28*

Not too exciting, but spaCY has a cool visualization package that can help us see our NER tokens in context:



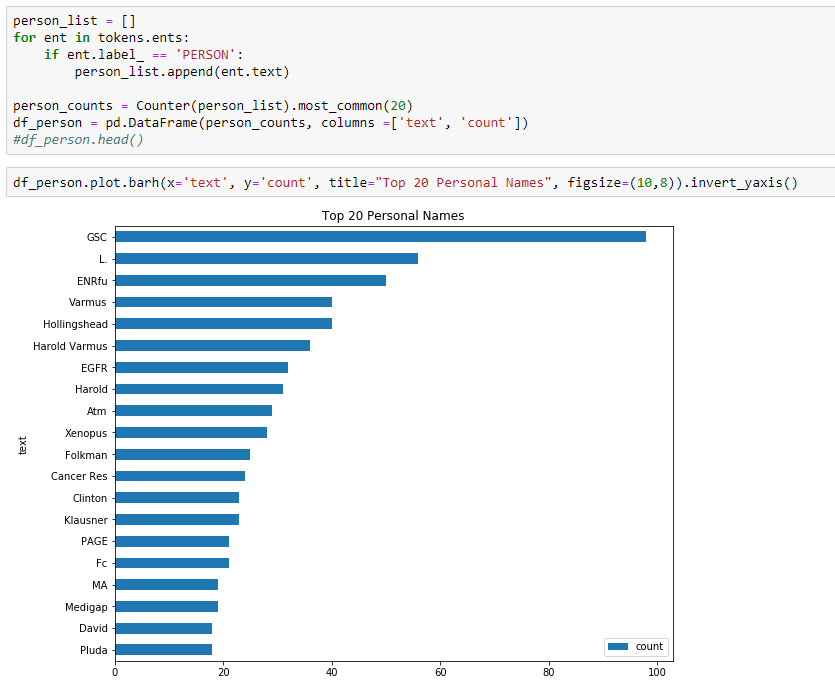
*Figure 29*

The first few lines of our first file now shows a wide-range of entities along with their NER codes like ORG for organization name, GPE for geopolitical entity, DATE, MONEY, etc., but clearly spaCY is not perfect. NIAID is not a geopolitical entity and the pathogenic yeast *Candida Albicans* is not a nationality or religious or political group (NORP). I expect we could tune our corpus and spaCY for better results, but this is a good enough start for our experiment.

Now that we have a list of NER tokens, we can isolate the most common NER’s for our use cases, count occurrences, and produce some bar charts for a simple visualization.

## Personal Names

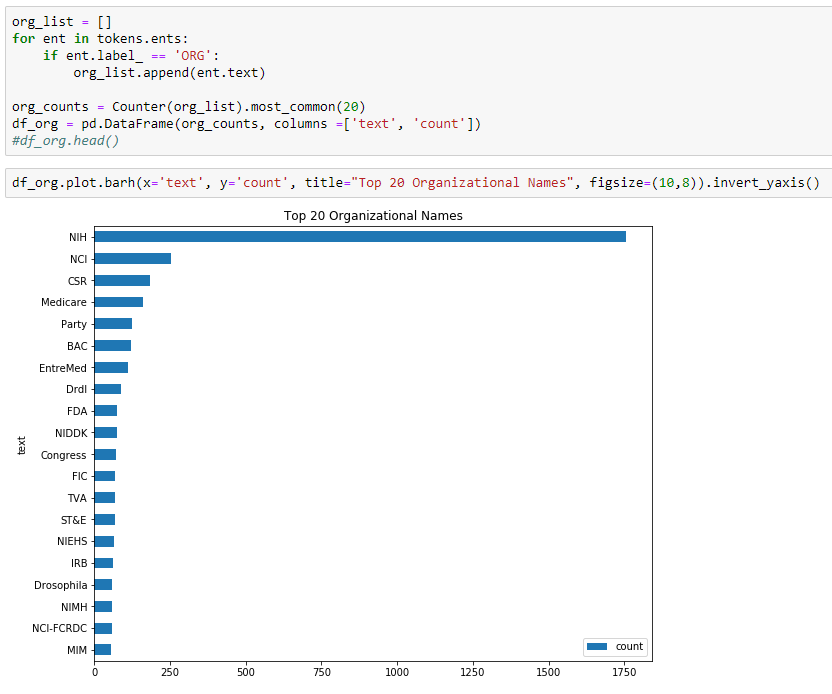
We don’t see many fully-formed personal names and there are several mis-identified terms, but the terms are different from other keywords we’ve generated before, so while they aren’t likely to produce major cataloging terms they do provide additional context. Many of the names are the last names of Varmus’s co-authors on a few research papers, which does flesh out the diabetes research topic. Clinton and Medigap are new terms which with a bit of background research probably relates to changes in Medigap/Medicare legislation Varmus was helping President Clinton’s administration formulate. David refers to David Lipman and work on E-Biomed.



*Figure 30*

## Organizational Names

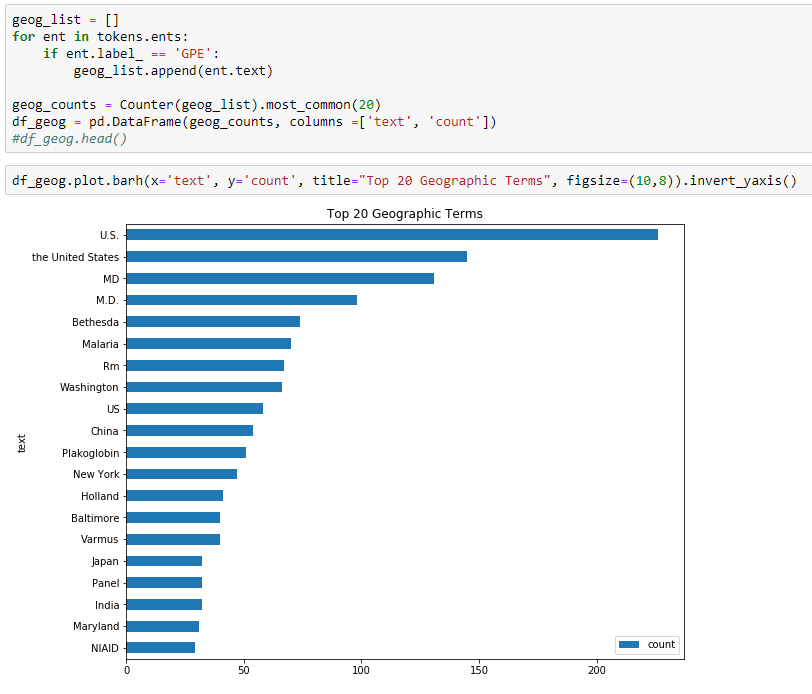
We do get better results with organizational names where we see abbreviations for lots of NIH Institutes. These are familiar to us at NLM since we work daily in the realm of NIH acronyms, but these may not be as obvious to the casual reader. Some more background research shows that Congress probably is used in conjunction with these Institute names as related to annual funding needs and research agendas. CSR (Center for Scientific Review) and IRB (Internal Review Board) perhaps relates to the research misconduct policy drafting revealed in our topic modeling and tri-gram exercises. EntreMed is a new term. Background research reveals it may be the Rockville, Md. biotech firm EntreMed (now CASI Pharmaceuticals) who in 1998 was reportedly on the verge of curing cancer based on promising mice-model research and the drugs angiostatin and endostatin. Perhaps our breast cancer/genetic research/mouse study topic and terms are related?



*Figure 31*

## Geopolitical Entities

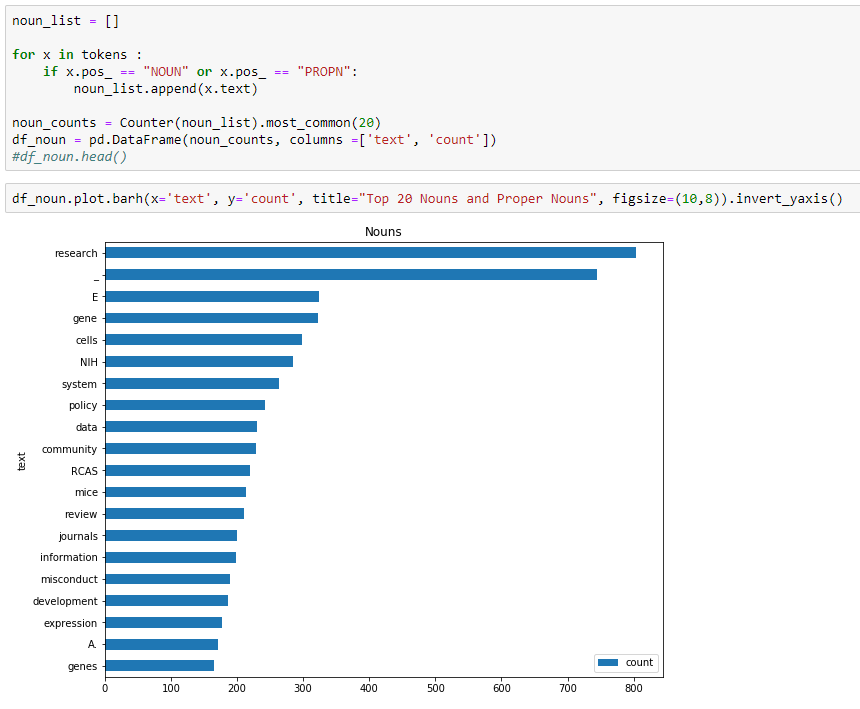
We do little cataloging of geographic entities for our archival collections – most of our collections relate to policy work and research performed here in the District, Maryland and Virginia triangle. The low number of other geographic locations compared to the total number of documents probably doesn’t justify much further investigation.



*Figure 32*

## Nouns and Proper Nouns

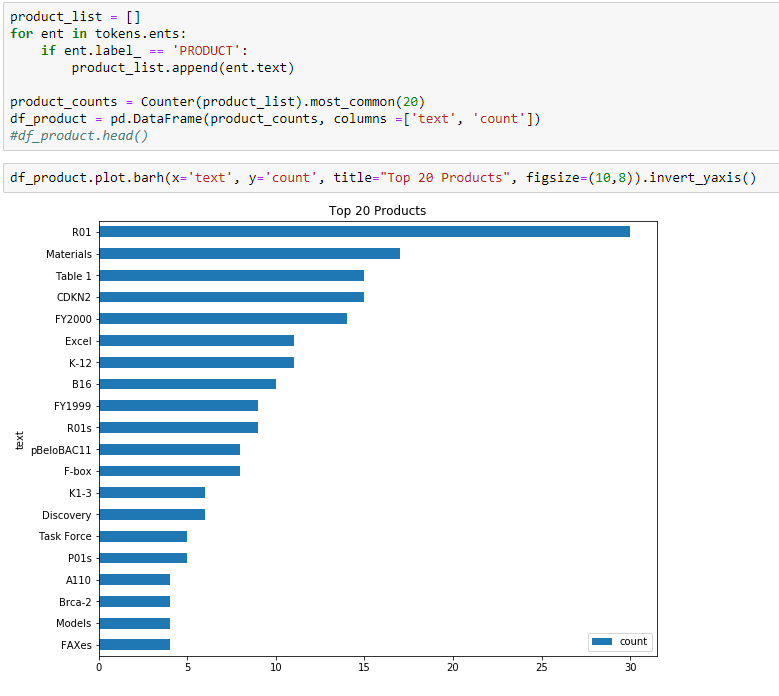
The nouns and proper nouns re-confirm the validity of our other keyword exercises.



*Figure 33*

## Products

For fun, what does the product entity label reveal? They aren’t commercial products, but it looks like we have some more terms relating to research and research funding and maybe that breast cancer cure research with terms like R01 and P01 (research grant terminology) and BRCA-2and CDKN2 (a tumor-suppressing gene).



*Figure 34*

# Conclusions

We’ve conducted a wide range of common text mining Natural Language Processing experiments against a sub-set of word processing documents produced or collected by Dr. Harold Varmus during his tenure as NIH director. Our tools and process worked from a technical perspective, although with mixed results for our cataloging and description use cases. We found low statistical coherence scores with our topic modeling experiments. Our keyword extraction numbers were generally low in relation to the number of total unique documents. Taken together I would say these show there was little cohesiveness between the documents, that their topics are widely dispersed and varied. And perhaps that is the nature of the documentation coming from a high-level director’s office and indicates the wide-variety of issues and debates facing a Nobel Prize winning scientific leader of Dr. Varmus’s stature. However, we did surface several interesting topics through the keyword and bi-gram/tri-gram exercises that we can report to our processing archivist. Several of these topics are already described in the descriptions of the analog paper collection contents such as the origins of E-Biomed and the Human Genome Project. Hopefully these electronic records add to that documentation or add context to the additional 38 boxes of still-unprocessed materials from Dr. Varmus’s tenure as NCI director. Though the outputs were imperfect in many ways and could use refinement, they are likely to be “good enough” for the use cases we defined in our hypothesis and the refinements an aspirational future research goal. The hard work of developing this data science pipeline should be easier to deploy against additional word processing collection materials and pay more dividends in the future as our data science tooling advances and text mining experiments continue.