Project Milestones (2)
Recommendation of similar articles from journal abstract analysis

Misty M. Giles
https://github.com/OhThatMisty

Overview

Scientists need to spend a lot of time researching others' work to advance their careers. I'm creating a recommendation system to help astrophysicists at Marshall Space Flight Center find articles related to a specific topic. This system searches abstracts hosted by arXiv and uses the document similarity technique TFIDF to return related abstracts.

Previous reports are available on GitHub. They cover the choice of data and the procedure to obtain, clean, and analyze the abstracts. An additional notebook called ArXiv_data_usertest.ipynb has also been provided so users can download a sample for testing. (A pre-obtained sample is also available in the data folder.)

Additional Analysis

As usual in this step of the analysis, I discovered that certain aspects of the data didn't play well inside the models. Text analysis requires preprocessing, and most of the troublesome aspects of physics text could be resolved then.

In addition to the usual text preprocessing steps of lowercasing, lemmatizing, and removing punctuation, I needed to account for physics-specific lexicology. First, I used a converter to remove letters with accents (like a ç, ñ, or ü that could appear in a name) and replace as many as possible with ascii equivalents.

Scientists like equations, and they like putting those equations in their abstracts with LaTeX formatting or the matplotlib mathtext module. PDFs or HTML can process this formatting

and show exactly what the author intended to write. In plain text, however, this puts symbols, numbers, and noisy words all over the place. Here's a sample, first how the text is written and second how it [appears on arXiv]():

```
much higher S/N.  The upper limit of the $\\gamma$ discrepancy set by
such an extensively-observed and well-modeled source is as follows:
$\\gamma_{radio}-\\gamma_{\\gamma-ray}<3.28\\times10^{-9}$ at the energy
difference of $E_{\\gamma-ray}/E_{radio}\\sim10^{13}$,
$\\gamma_{radio}-\\gamma_{X-ray}<4.01\\times10^{-9}$ at the energy
difference of $E_{X-ray}/E_{radio}\\sim10^{9}$,
$\\gamma_{radio}-\\gamma_{optical}<2.63\\times10^{-9}$ at
$E_{optical}/E_{radio}\\sim10^{5}$, and
$\\gamma_{optical}-\\gamma_{\\gamma-ray}<3.03\\times10^{-10}$ at
$E_{\\gamma-ray}/E_{optical}\\sim10^8$.
```

much higher S/N. The upper limit of the γ discrepancy set by such an extensively-observed and well-modeled source is as follows: $\gamma_{radio} - \gamma_{\gamma-ray} < 3.28 \times 10^{-9}$ at the energy difference of $E_{\gamma-ray}/E_{radio} \sim 10^{13}$, $\gamma_{radio} - \gamma_{X-ray} < 4.01 \times 10^{-9}$ at the energy difference of $E_{X-ray}/E_{radio} \sim 10^{9}$, $\gamma_{radio} - \gamma_{optical} < 2.63 \times 10^{-9}$ at $E_{optical}/E_{radio} \sim 10^{5}$, and $\gamma_{optical} - \gamma_{\gamma-ray} < 3.03 \times 10^{-10}$ at $E_{\gamma-ray}/E_{optical} \sim 10^{8}$.

I don't know what any of those formulas mean, but this example illustrates a punctuation-removal problem. First, removing the punctuation would leave behind remnants of mathtext. Those remnants would include a lot of "gamma" - a word that can be very important in classifying astrophysics articles but is math noise here. Other Greek letters like alpha, lambda, and sigma present the same issue. While I was unable to find a way to remove the mathtext/LaTeX as a whole from text files, I solved this with a series of regular expressions (regex). The text in the example could be handed by selecting all "words" that fit a pattern, '\$\S*\$', and then substituting them with a space. (The pattern stands for "find a $ and then find another $, and select everything between them as long as there aren't any spaces." This pattern would select a string like "$mathtext$" but wouldn't select "$math text$.") Some terms didn't have surrounding $s, so I utilized regex to replace patterns like "\\math." A handful of words also needed to be hard-coded into the removal code. After this, I was able to remove the remaining punctuation.

I also used regex to test another perplexing problem. Astrophysicists use numbers to name stars and events. Some names seem so prevalent in research that they could come

up in at least 1% of my abstracts -- Cyg X-1, GW170817. After several tests on the full set of data, I only found one name that came near 1%: GW170817 at 0.68%. I still needed to remove single characters left over from the math formatting, which could have had the unintended effect of removing "x-ray," a significant term. My solution was to substitute "x_" for any "x-" before removing the digits and any non-underscore punctuation.

The rest of my text normalization was fairly standard: remove stopwords, lemmatize the tokens, and join the tokens back together as one sentence per abstract in a text file that the model could read.

Modeling

I selected TFIDF (term frequency–inverse document frequency) as a good model for the abstracts. It's easy to manipulate, and I can adjust the parameters to filter out typos and common words. TFIDF measures how frequently a word occurs in a document and compares that to how *infrequently* it appears in the rest of the documents. I used a floor of 400 abstracts (pushing the limit of my computing power) to help reduce the vocabulary to something manageable.

The only other parameter I found reason to change was the ngram_range. Ngrams are groups of words that appear together at least as frequently as the selected floor. I tested ngram_ranges of one-to-three, one-to-four, and one-to-five words. Fewer than ten results over three words met the threshold to be included in the vocabulary. Using (1, 3) -- or unigrams, bigrams, and trigrams -- provided a vocabulary of about 2,200 words and nearly 7 million stopwords, the words and phrases that didn't meet the threshold.

TFIDF in sklearn is so inexpensive that I was able to test different parameter combinations until I was satisfied with the results.

Recommendation Engine

While this is an unsupervised learning project, I started with a training group (98%) and a testing group (2%). To ensure that the training data covered all dates, I shuffled the rows of the dataframe with pandas' .shuffle() ability before data normalization. I didn't attempt any stratification and make no claims about the dates in the testing group.

The TFIDF model had been fit on 58,772 abstracts, leaving 1,200 that were only transformed. These 1,200 serve as the "user input" for the recommendation engine. This engine would recommend abstracts similar to the one being read, using cosine distances calculated by the TFIDF model. It currently operates by choosing a random abstract from the test group and calculating the distances between that abstract and the abstracts in the larger group.

The engine first prints out a small table. As seen in the following screenshot, the table provides the current index (shuffle doesn't preserve the original indices), abstract, title, terms (arXiv categories to which the article was submitted), and document similarity. This is where I use domain knowledge to check that I've been fed reasonable responses.

	abstract	title	terms	document_similarity
31861	Nearby star-forming galaxies offer a unique en...	Different generations of HMXBs: clues about th...	astro-ph.HE	0.503205
2748	We have identified 55 candidate high-mass X-ra...	Formation Timescales for High-Mass X-ray Binar...	astro-ph.HE|astro-ph.GA	0.479234
1955	[abridged] How does a star cluster of more tha...	NGC 346: Looking in the Cradle of a Massive St...	astro-ph.GA	0.455667
23459	We present 15 high mass X-ray binary (HMXB) ca...	Young Accreting Compact Objects in M31: The Co...	astro-ph.HE	0.446389
23715	The 30 Doradus star-forming region in the Larg...	An excess of massive stars in the local 30 Dor...	astro-ph.SR|astro-ph.GA	0.440245
28188	The objective of this work is to study how act...	Quenching by gas compression and consumption: ...	astro-ph.GA|astro-ph.CO	0.431887

I first check the document similarities. In this particular example, they're all between 0.43 and 0.50. As long as these numbers are fairly close together, I'm satisfied. The similarity numbers themselves don't matter as much -- I've seen the first result range anywhere from 0.74 to 0.38 in testing as the abstracts range from esoteric and extremely specialized to shallow and intended for general consumption -- but they do offer a sense of fit. If the similarity is too high, the engine might have provided a list of abstracts about a project, written by the project team, and created from a project template. (The Cherenkov Telescope Array (CTA) abstracts in particular will have high similarity to another CTA abstract.)

The second check involves the categories to which the author submitted the work. If every abstract showed up with the same category represented in all rows, I'd be fairly confident that the engine had the general idea of the sample abstract and skip to scanning over the abstracts and titles. Each row in this screenshot, however, lists either high-energy astrophysics (astro-ph.HE) or physics of galaxies (astro-ph.GA). My domain knowledge is

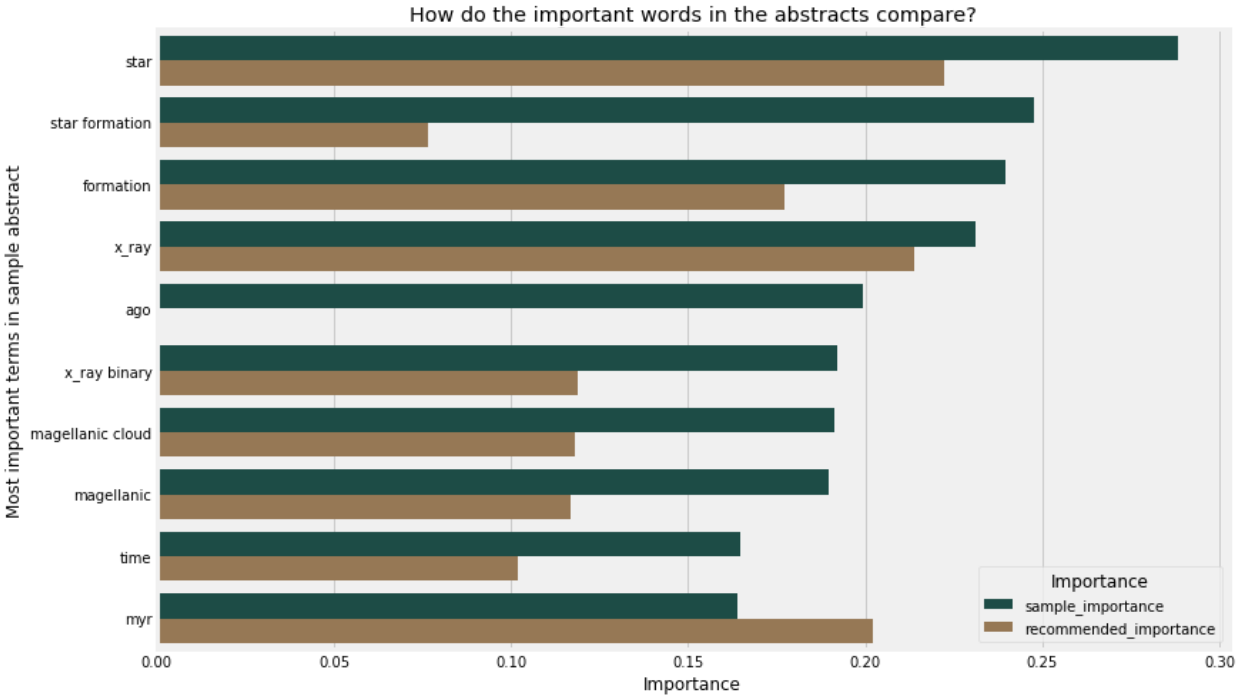
limited, so I'll check over the abstract previews and the titles. For the two marked astro-ph.HE, I see that 31861 starts with "star-forming galaxies" and 23459 mentions "M31," which sounds like a galaxy. An internet search confirms that it's another name for the Andromeda Galaxy. Now I know that all six recommendations involve galaxies.

In the next screenshot, the feature importances for the sample abstract are sorted, and the top ten features are listed. Next to them are feature importances of the same terms from the recommended abstract (first result on earlier table of six) for comparison.

	feature	sample_importance	recommended_importance
1986	star	0.288187	0.222307
1991	star formation	0.247674	0.076422
793	formation	0.239517	0.177373
2272	x_ray	0.230971	0.213804
58	ago	0.199327	0.000000
2273	x_ray binary	0.192241	0.118635
1204	magellanic cloud	0.191314	0.118063
1203	magellanic	0.189575	0.116991
2128	time	0.164884	0.101753
1355	myr	0.163825	0.202199

This example shows only one term, "ago," that isn't in both abstracts, but most recommendations in testing showed fewer matching terms. This table doesn't provide much basis for checking the accuracy of the engine, but it does provide a way to compare which words were more impactful.

Here I've plotted the relative strength of the two sets of features. The terms on the y-axis are the ten most influential sample abstract features, and the values are plotted in dark blue. The tan bars are the strength of the recommended abstract's features, for the terms they share. Reminder: This plot is unusual for showing this many matching features, and many other abstract pairs share fewer.



The five highest features of the recommendation are printed to the screen, as well. Finally, both the sample abstract and the recommended abstract are printed, along with their titles, URLs, and arXiv categories. The abstracts used in this example are available. Sample: Star-formation history and X-ray binary populations: the case of the Large Magellanic Cloud and Recommended: Different generations of HMXBs: clues about their formation efficiency from Magellanic Clouds studies.

In most production cases, articles would be added to the model when they were uploaded to a website with this type of system, and the TFIDF matrix would be updated regularly. I chose this sandboxed approach to see how the 1,200 test abstracts would fare. While writing the code, I tested with abstracts that had been in the tfidf.fit_transform() and abstracts that had only been through tfidf.transform() and didn't observe a difference in how the recommendation engine performed.