Ryan Fraser
UIUC Online MCS
CS410: Text Information Systems – Fall 2020

Investigation of Using Python NLP Libaries (NLTK and SpaCy) to Perform Sentiment Analysis on Economic Reports

Problem Description: Quickly understanding sentiment of new economic reports

Government institutions around the world routinely publish various reports on the status of their respective economies. These reports contain a wealth of information, ranging from raw data to the opinions of those most responsible for managing the economy. (One common thread through all this information is that these reports are some of the only ways to obtain this very important information.)

Investors around the world need time to digest all of the information in these reports as it can significantly affect market and investment values, so they need to be able to quickly understand if the report is going to hurt the value of their investments in order to react accordingly before the situation worsens. This problem is compounded by the fact that investors also have huge amounts of *other* data they need to interpret each day, so their resources can be stretched very thin.

Building tools that can automatically ingest the latest reports and quickly give some kind of warning signal could help investors allocate their attention and resources more effectively, which could help them preserve the values of their portfolios.

Basic structure of an example report: The United States Federal Reserve – Federal Open Market Committee Statements

The United States Federal Reserve is the central bank of the United States. It is responsible for conducting monetary policy for the world's largest economy. This has a global impact for obvious reasons, but also because the United States Dollar is the world's reserve currency of choice. The Federal Reserve's division that actually implements the monetary policy is called the Federal Open Market Committee, or FOMC. The FOMC is led by a group of highly qualified individuals that also have access to extremely sensitive and pertinent economic information.

The FOMC issues monthly statements outlining their general thoughts on the economy and how they are changing monetary policy accordingly. Per Investing.com, "it contains the outcome of the vote on interest rates, discusses the economic outlook, and offers clues on the outcome of future votes." These statements are usually released as simple PDFs or HTML webpages. As such, it is straightforward to ingest or scrape the raw text and perform sentiment analysis on it.

Overview of NLTK

NLTK (Natural Language Toolkit) is a Python library that performs core Natural Language Processing (NLP) tasks. It was originally created as an educational and research tool. Because it was originally an educational and research tool, it has a huge suite of algorithms and libraries that allow the user to perform any task necessary in an entirely customizable way. (From an academic perspective, this was ideal both for allowing users to explore and learn as well as allowing users to create something entirely self-designed for research purposes.)

Ryan Fraser
UIUC Online MCS
CS410: Text Information Systems – Fall 2020

NLTK is also easily extended and customized. Very little functionality is abstracted away, which can be tedious, but also allows for more customization and control. From a data structures and algorithms perspective, NLTK is almost entirely string and list based. Most functions ingest a string or list of strings, and most functions return the same.

Like many things in life, something's greatest strength is often its greatest weakness. To use an analogy, compare writing NLP code to building a racecar. Many users might prefer to simply take an off-the-shelf, high-performance car. However, other users might have very specific ideas and very specific needs – these users might choose to purchase a garage and a toolkit and build it themselves from scratch. Using NLTK is similar to building the racecar from scratch. This can create a very fit-for-purpose tool, but the user may have to put in a lot of work to get it there. It is up to the user to determine if this extra effort is worth it.

Overview of SpaCy

SpaCy is also a Python library that performs core NLP tasks such as tokenization, stemming, part-of-speech tagging, syntactic analysis, and more. Unlike NLTK, however, SpaCy is (aggressively) marketed as a toolkit meant to be used in production environments. It is "opinionated," in the sense that the community attempts to choose a single, best algorithm for each function, then focuses on maintaining that algorithm… until that algorithm is no longer state-of-the-art, when it is then replaced with something that is.

In addition to the approach of providing a single "best" method, almost everything in SpaCy is object oriented. Functions create and return their own objects, which then have their own internal data and methods. This can speed up development, but also means that it is not as easy to extend as NLTK, and is much harder to customize if the user has a very specific approach in mind. However, SpaCy may be ideal for users who are focused on shipping a product quickly and getting a feature out the door in a suitable manner. (In fact, one of the very first things on SpaCy's website landing page is a bold header stating, "Get things done.")

Using the racecar analogy from above, NLTK is "building a racecar from scratch," while SpaCy is "going to a dealership and buying one off the lot." This approach might not be ideal if the user is trying to "build a Formula 1 car," but may be ideal if the user is "in a hurry to get somewhere quickly."

How NLTK and SpaCy approach sentiment analysis

Unsurprisingly, NLTK has a few built-in libraries for performing sentiment analysis. The most prevalent library is VADER, which stands for "Valence Aware Dictionary for sEntiment Reasoning," which relies on a predefined dictionary of various words and features to calculate sentiment scores. In general, it seems to work well on things such as social media posts and product reviews, or really most sufficiently "broad" tasks. A user might choose to use VADER if they are really just trying to find the "general" sentiment of a large number of documents.

NLTK also has some built-in machine learning tools such as a Naïve Bayes Classifier. If a user is willing to provide labeled training data and train these models, NLTK may also be a good choice. NLTK does not currently have any native deep learning tools which represent the state-of-the-art for text classification, so the user may still need to put in (possibly substantial) work to prepare the data to be ingested by a deep learning model.

Interestingly, despite being marketed as a comprehensive tool, SpaCy does not have a single built-in library of any kind that directly supports sentiment analysis, even for general use cases. One of SpaCy's major strengths, however, is that it makes it very easy to prepare the data in such a way that its ready to be ingested by almost any machine learning or deep learning model.

SpaCy is marketed as the more self-contained, all-purpose, production-ready tool for NLP. It's interesting that it has no built-in tools for sentiment analysis, while NLTK does. However, NLTK's sentiment analysis tools are very general and may not be a great fit for some specific uses, particularly if those uses are niche.

Using NLTK and SpaCy to determine sentiment of economic reports

In terms of preparing the data to be ingested by more advanced models than either NLTK or SpaCy are able to provide, the user will need to do some investigation into figuring out just how important various niche economic terms are when determining if the sentiment of a report is positive or negative. Similar to many other areas of NLP, this will require human intervention and domain expertise. If the user finds that the inputs need to be cleaned and prepared in extremely specific ways, and is also willing to put in the effort to do that, NLTK may be a better choice. If the user finds that the inputs do not need to be cleaned beyond the scope of more general NLP tasks, it's likely the SpaCy will be the better choice given its focus on making development in production easier.

For the sentiment analysis itself, NLTK will be able to quickly and easily perform basic, general sentiment analysis on these reports. However, a user would be wise to thoroughly back test this approach on prior releases to make sure the output is well understood. The reports contain lots of arcane language that is likely not in VADER's predefined lexicon, and these terms could be strongly indicative of positive or negative sentiment. Per above, if the inputs don't need to be intensively cleaned, it may be more straightforward to use SpaCy and a separate deep learning model.

SpaCy will not be able to perform sentiment analysis at all. The user will need to use SpaCy to create the basic inputs that will then be ingested by other models – if this is out of scope of the user and the user simply needs a "quick and dirty" sentiment analysis on the reports, it's likely NLTK will be a better choice. However, this still seems unwise given arcane words are likely to be extremely material in these reports, so in the spirit of, "a job worth doing is a job worth doing right," it seems using SpaCy to prepare the data and then feeding it into a fit-for-purpose deep learning model will make the best use of time and provide reliable results.

Ryan Fraser
UIUC Online MCS
CS410: Text Information Systems – Fall 2020

Citations

Primary Sources

- https://spacy.io/
- https://spacy.io/models/en
- https://www.nltk.org/
- https://www.nltk.org/book/ch06.html
- https://www.nltk.org/howto/sentiment.html
- https://www.federalreserve.gov/monetarypolicy/fomccalendars.htm

Secondary Sources

- https://blog.thedataincubator.com/2016/04/nltk-vs-spacy-natural-language-processing-in-python/
- https://en.wikipedia.org/wiki/Natural_Language_Toolkit
- https://en.wikipedia.org/wiki/SpaCy
- https://gist.github.com/rschroll/61b20c41e984a963df2870cfc9e628ed
- https://medium.com/analytics-vidhya/simplifying-social-media-sentiment-analysis-using-vader-in-python-f9e6ec6fc52f
- https://towardsdatascience.com/sentimental-analysis-using-vader-a3415fef7664
- https://www.activestate.com/blog/natural-language-processing-nltk-vs-spacy/
- https://www.digitalocean.com/community/tutorials/how-to-perform-sentiment-analysis-in-python-3-using-the-natural-language-toolkit-nltk
- https://www.investing.com/economic-calendar/fomc-statement-398
- https://www.presentslide.in/2019/07/implementing-spacy-advanced-natural-language-processing.html
- https://www.quora.com/How-can-I-use-Spacy-to-perform-sentiment-analysis
- https://www.quora.com/Out-of-all-the-sentiment-analysis-libraries-Python-has-which-one-do-you-prefer-and-why