Scott Elmore

2/10/20

DSC 575

**Project Proposal**

For my project I am seeking to build a search/retrieval system centered around retrieving financial news. I have money invested in several stocks on Robinhood and I would like to view financial news related to the stocks I own, news related to the sector of which the stock is a part of, and/or news that would affect the financial market at large, i.e. “Coronavirus”. Robinhood is a free, zero fee, online trading platform that allows API access, so via Python (<https://github.com/jmfernandes/robin_stocks>) I can access the names and symbols of the stocks I own. Currently, Robinhood does have a news aggregation service tailored around the stocks in my portfolio, but it only shows articles that mention the stock by name, and it tends to have duplicate information. Given the names and symbols of stocks in my portfolio I want to locate news stories with any relationship. The source of the news will be from an online API called News API (<https://newsapi.org/s/us-business-news-api>). This API allows a user to aggregate articles based on a category, so I can query all articles related to “business”. The API returns a JSON object with source names, i.e. “CNBC”, title, description, and content. Using this JSON object I will be able to create an inverted dictionary using the stemmed terms from the content of the articles. I plan on using stemming and stop words to filter out the irrelevant terms and using TF-IDF weights to give appropriate weighting to the terms. I can then use the several similarity measures such as Cosine, Jaccard, and Dice to find the documents with the closest similarity to my stocks. The next step will be to use term co-occurrence to come up with the sector of which the stock belongs and find competitors. This will allow me to do a query expansion so instead of just querying for “Tesla” I can also query for “cars, energy, electric, etc..”. I believe it will be prudent to use Automatic Local Analysis to do query expansion because it will save computation time but also prevent superfluous terms from affecting the results of the query. Once results are returned, I plan on listing the top “n” articles daily related to the stocks in my current portfolio. I also plan on having a time-based filter to filter out old news and perhaps give more recent news more weight. I will store the inverted indexes of old news articles offline, so my program won’t have to recreate the dictionary every time I need to run a query against my stock names.