Categorizing Companies by Stated Risk Factors in 10-K Filings

Evie Wan, Nick Mosca, Eric South

[Draft] Final Report

Introduction

We developed a web-scraping tool that extracts financial documents from EDGAR, a database hosted by the Securities and Exchange Commission (SEC). Publicly traded companies must annually submit 10-K filings to the SEC. These 10-K documents are intended to be comprehensive references for current and potential investors, where financial performance, forward looking statements, and risk factor sections are detailed across multiple sections. Each 10-K document must include a risk section, often labeled as “Item 1A Risk Section,” which often details distinct vulnerabilities a company is working to mitigate. 10-K risk sections often consist of multiple paragraphs, and may emphasize technical, market, or supply chain risk, among other factors. We hypothesized that vulnerabilities described in Item 1A could be used to categorize companies into distinct groups, where underlying sentiments across paragraphs of text would be dependent on a company’s general risk profile. We assumed that risk profiles were not made equal, and that different company vulnerabilities were sensitive to different external factors. Given these assumptions, we were curious whether the economic disruption that followed COVID-19 had a disproportionate impact on biotechnology companies with certain risk profiles. Although biotechnology companies in general have performed well during the pandemic, we asked whether a company’s financial performance (e.g., fold change in revenue between 2019 and 2020) could be linked to their self-stated risk factors (found in their 2019 10-K filings).

Our project explored whether natural language processing techniques, such as topic modelling, could be used to differentiate companies by their stated risk factors. We developed a corpus of 10-K risk sections for hundreds of biotech companies by both adapting an API that interacts with the EDGAR database and developing an HTML web scraper that identified, cleaned, and aggregated paragraphs found in Item 1A subsections. Aggregated risk texts were then subject to Latent Dirichlet Allocation (LDA), an unsupervised learning, probabilistic algorithm which can represent a corpus of text as an underlying set of topics.

Once companies are grouped into risk categories, financial performance (e.g., net profit, changes in stock, etc.) will be comparing pre- and post- COVID-19. Among biotechnology companies, we are interested in the relationship between 1) types of business model risk (as stated in SEC filings) and 2) financial growth during a pandemic. Our initial pipeline compared a handful of companies from IBB. However, we have expanded our pipeline to handle the entire index for both 2019 and 2020. We have preliminary topic modeling results for the entire IBB index and are continuing to tune our pipeline . Ideally we would like to be able to identify distinct risk topics from 2019 and 2020 and couple that result with financial metrics to see how companies performed over the last year.

Approach

To conduct our analysis on the IBB index we decided to build a function to bulk download financial documents directly from the Securities and Exchange Commission. We specifically

targeted 10-K documents because of the specific investor related details that are provided within this annual report. Our Bulk_extraction function allowed us to select the type of financial document and the year of interest and with a provided list of ticker symbols we were able to download hundreds of 10-K documents for both 2019 and 2020. Data is nicely organized per ticker symbol in a predetermined location that the user selects.

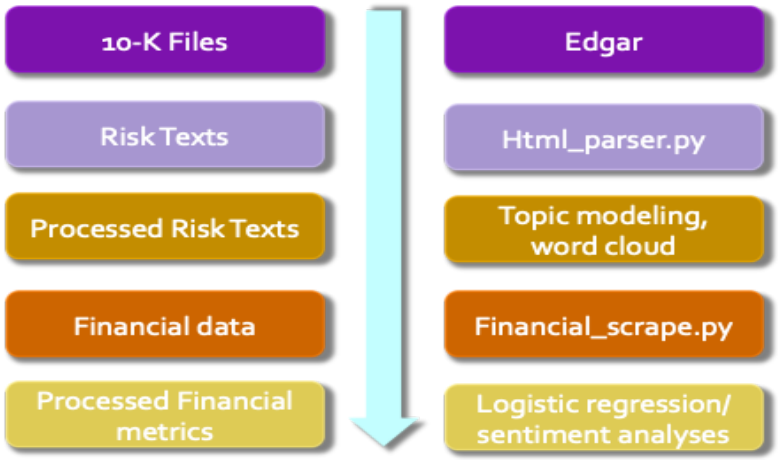

Figure 1. Overview of our analytical pipeline.

In order to process our raw data we had to develop a function that managed file paths locally. The objective was to create a set of file paths that correlate to each company's pre download 10-K for 2019 and 2020. That file path would be used as an input for our preprocessing html_parser function. To accomplish this we implemented a function called file_paths within our path_mover.py module. The file_paths function utilizes python's pathlib package which is designed to use regular expression to extract paths based on file types. File type extensions coupled with ticker symbols embedded inside the file name allowed us to generate file paths for every downloaded 10-K document.

The format of a 10-K document is similar at first glance but differs in terms of HTML structure. Our html_parser.py module is able to capture the majority of the risk sections from companies within the IBB index. Within the parser module we have implemented a number of preprocessing functions. File path outputs from the file_paths function and used to import individual 10-K documents. Section 1A Risk Factors were located for each company and extracted as strings via the grab_section_text function which takes advantage of Python's BeautifulSoup package. The isolated raw 10-K risk sections are processed in our clean_strings function. All html tags, whitespaces, and end of line characters were removed. In addition the text was tokenized, lemmatized , stripped of common stopwords using the nltk package.

To improve plumbing between modules, we developed a function for bulk downloading of all 10-k financial documents from the IBB index (2019 & 2020). We've also written supporting modules (path_mover.py and structuring_data.py), which generate lists of directory paths (and help plumb all our local HTML files to our HTML parser).

We connected our web scraping module (10K_extraction.py) to our HTML parsing module (html_parser.py), which enabled the scanning and extraction of hundreds of 'Risk Sections' (i.e. paragraphs of strings found between Item 1A. and Item 1B.) from SEC 10-K filings. We then queried the SEC EDGAR database, and uploaded a slew of financial documents onto our local machines, parsed these files, and produced a CSV which contains 1) company ID and 2) cleaned risk text. Obtaining this CSV was a milestone, as it enabled downstream topic modelling and EDA (see below).

Given our initial corpus of text (i.e. 200 risk sections--paragraphs containing self-prescribed vulnerabilities by company executives), we have developed topic_model.py, which learns and extracts topics across a collection of documents. The module converts our company CSV file (see above) into 1) a corpus dictionary (i.e. unique set of words) and 2) corpus of text (i.e. term-frequencies for each risk document). This corpus of text feeds into an LDA model, where we specify the number of components (topics) a priori. As a starting point, we've selected 5 topics based on the literature surrounding business risk theory. According to Investopedia, business risks can be categorized as either: market, liquidity, credit, or operational. We've processed our preliminary risk corpus with LDA and produced word clouds, which visualize the top 15 words in each topic. Our output topics have considerable overlap, which tells us we need to either 1) tune our model or 2) refine preprocessing steps (e.g. adding stopwords or improving upstream string cleaning).

[FIGURES ARE PLACEHOLDERS -- MODEL NEEDS TO BE REFINED]

Table 1: Subset of our processed text prior to LDA model.

Ticker	Risk Text
KRYS	['item', '1a', 'risk', 'factor', 'risk', 'related', 'to', 'our', 'fi…
KURA	['following', 'the', 'closing', 'of', 'our', 'initial', 'public', 'o…
LGND	['item', '1a', 'risk', 'factor', '17', 'government', 'regulation', '…
LMNX	['item', '1a', 'risk', 'factor', 'the', 'life', 'science', 'and', 'd…
LRMR	['other', 'risk', 'and', 'uncertainty', 'including', 'those', 'liste…
MCRB	['item', '1a', 'risk', 'factor', 'our', 'business', 'face', 'signifi…
MEDP	['a', 'of', 'december', '31', '2018', 'our', 'backlog', 'increased',…

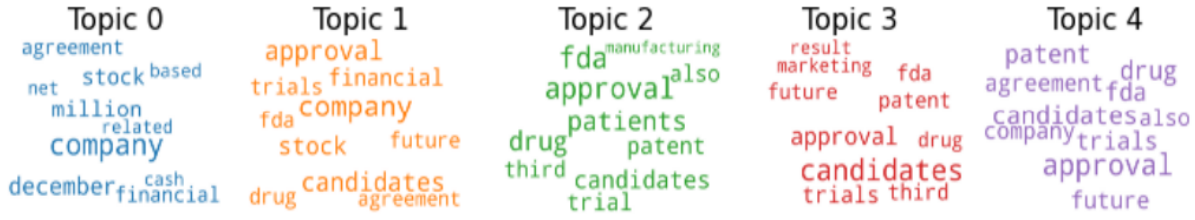

Figure 2. Word clouds generated from 2019 processed text, which then feeds into our LDA model and designates a dominant topic to each document in the corpus. We identified the dominant topic (& its percentage contribution) for each document, and then created word clouds of top 15 words in each topic.

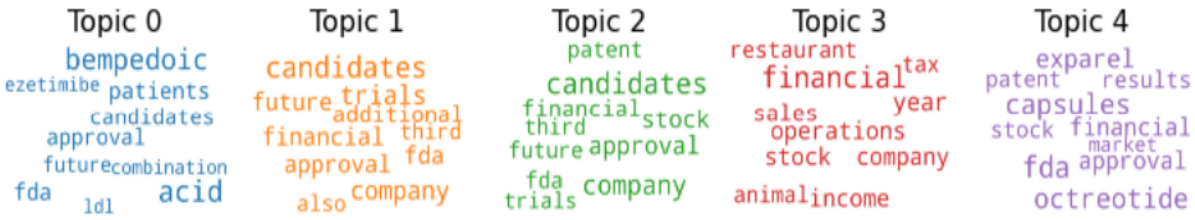

Figure 3. Word clouds generated from 2019 processed text (post-LDA). The issue is we see little differentiation among topic groups.

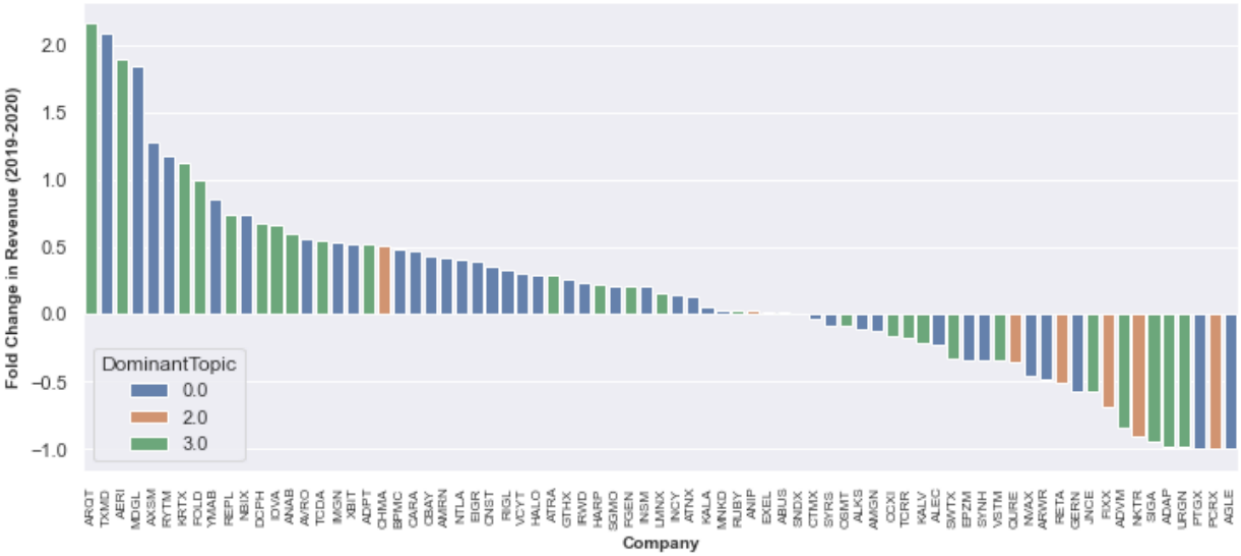

Figure 4. Incorporating 10-K revenue data for each company with dominant topics produced by the LDA model.

[paragraph on LDA struggles]
Our LDA model is currently a blackbox. We merged LDA results (i.e. the dominant topic # associated to each company) with revenue data. I added a column fc, which indicates fold change [(new_revenue - old_revenue) / old_revenue)]. Despite specifying 5 distinct topics in the LDA, only 3 topic groups (groups 0, 2, or 3) were ever dominant in a document. Oddly, LDA results led to extreme data skew towards one dominant topic (document's often had one topic predominant, opposed to an even distribution of 'dominant topics' across the entire corpus of documents. The problem is that uneven labelling of topics makes revenue comparison difficult. Our next steps are to tweak the LDA model to achieve more 'even' outcomes.

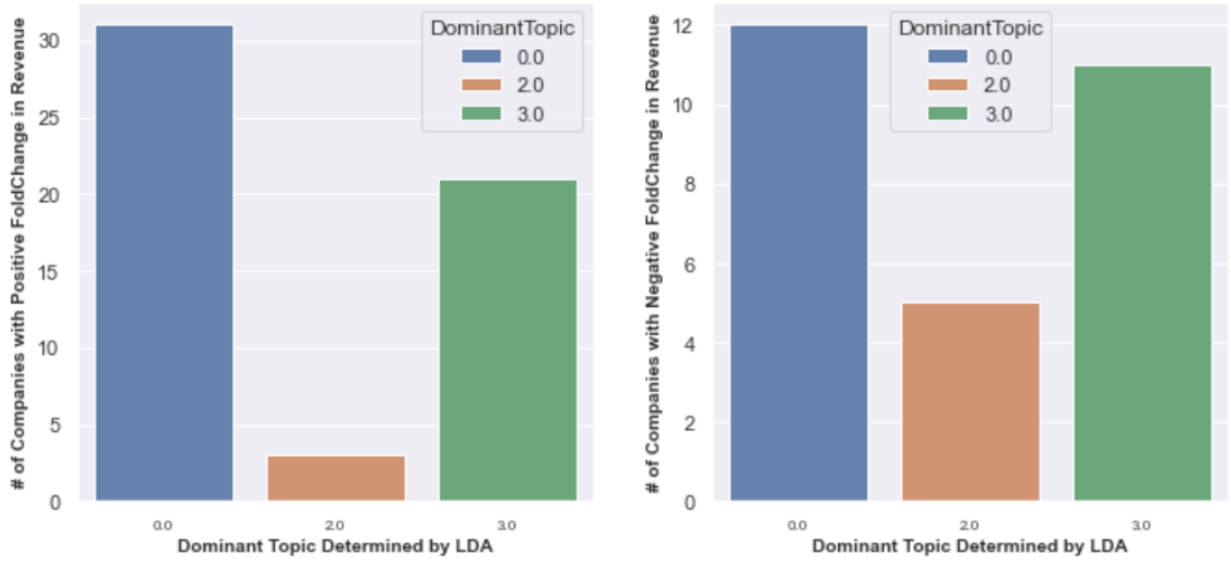

Figure 5. For each topic group, number of companies with positive fold change (i.e., how many companies performed well financially between 2019 and 2020, and are these numbers trending between topic groups?).

[paragraph on gathering financial metrics]
[paragraph on overall exploratory data analysis]

Discussion

What did we accomplish?
What did we do well?
What did we not do so well?

Evaluating Performance of our HTML Parser

We've found that our HTML parser is fairly generalizable (i.e. can successfully extract risk sections from a heterogeneous mix of html data). Although 10-K filings are purportedly standardized, the underlying HTML tree can vary, and thus developing a scalable method for isolating specific text sections is non-trivial. We've found that our current html_parser.py returns nan values for a proportion of 10-K filings-- indicating unbeknownst bugs in our scraping algorithm. Despite these issues, our web scraper can return over 200 Risk Sections for companies between 2019 and 2020 (which we're satisfied with, as it'll provide an initial corpus for our topic modeling efforts).

Limitations

Although we were excited that our LDA model works, we were concerned that our compiled risk sections did not form distinct 'topic groups'. Upon initial analysis, many of the top words among topic groups were standard biotech business words (e.g., 'product', 'develop', 'regulatory', 'clinical'). We sought to refine our model to form more differentiable groupings.

Subject to change

If we cannot do this, we'll need to focus less on finding differences among documents and more on exploring trends among our risk corpus. Either way, the next phase of our project will predominantly focus on exploratory data analysis.

Navigating our Repository

Tools and Methods

Scraping: beautiful soup, Autoscraper, or EDGAR specific packages will be used to aggregate data from yahoo finance webpages.

- Scraping EDGAR with Python (https://doi.org/10.1080/08832323.2017.1323720)

Pandas supported cleaning and preprocessing efforts, scikit learn for cosine similarity and clustering, and both Matplotlib and Seaborn for data visualization.