Using NLP to analyze Annual Reports

Problem Statement

If the yearly stock market trend of a company shows correlation to the sentiments expressed by the company's annual reports then this information can be extremely useful to make important future investment decisions.

Context

The length of 10-K documents (Annual reports uploaded by US companies to the Sec Edgar government website) has increased exponentially over the last decade. It is very cumbersome to extract only the useful information given in the document forgoing all the financial jargon and legal language that the documents need to adhere. This is also a highly manual process.

Currently, applications exist to extract only table data (numeric) from these documents and not the other information which is given in text format.

In order to extract information from these documents and to correlate it to stock prices we need to make use of the latest algorithms in NLP & machine learning.

Previous Research

An important research called "Lazy prices" was done to find if there was a correlation between stock prices and certain sections of the 10-k documents. This study reported that testing the similarity between these documents for a given company and plotting it against the stock prices of that company showed a correlation with a lag , hence the term "Lazy".

WhitePapers

https://arxiv.org/abs/1908.10063

https://www.nber.org/system/files/working_papers/w25084/w25084.pdf

Data retrieval

All US companies have to submit their annual reports to the government website https://www.sec.gov/ on a yearly basis. It has certain web crawling rules and by adhering to them we can successfully download the 10k forms by web scrapping using the python requests package and beautiful soup package.

A input csv file contains all the tickers (company short names as provided in the Edgar website) & their corresponding cik number. Using this we extract the urls for 10-k filings over all the years since the company has released its reports Then the filing text is extracted using beautiful soup & unwanted information like various html style tags etc is removed.

Preprocessing documents

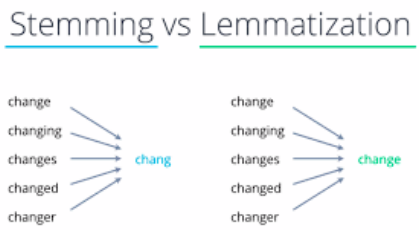
Before any NLP project is started we need to make sure that the text data used is as informative as possible. Hence the data goes through several stages as follows.

1. Remove Special characters & accented characters.
2. Remove numerical data.
3. Remove punctuations.
4. Detach words from numbers existing due to incorrect formatting.
5. Replace contractions. eg. “don’t” is replaced with “do not”.

Normalizing text

Tokenize : Each document is separated into lines and words using nltk.tokenize library.

Stemming & Lemmatization : Each word can be broken down to a short format. Stemming just removes or stems the last few characters of a word, often leading to incorrect meanings and spelling. Lemmatization considers the context and converts the word to its meaningful base form, which is called **Lemma**.



Removing Stopwords : Stop words are a set of commonly used words in any language. For example, in English, “the”, “is” and “and”, would easily qualify as stop words and eliminating these unimportant stopwords we give more accurate results during modelling.

Sample text with Stop Words	Without Stop Words
GeeksforGeeks – A Computer Science Portal for Geeks	GeeksforGeeks , Computer Science, Portal ,Geeks
Can listening be exhausting?	Listening, Exhausting
I like reading, so I read	Like, Reading, read

NLP libraries

We used a few prominent NLP libraries for their specific advantages for various NLP tasks.

- NLTK : Natural Language Toolkit
- spaCy : It is extremely fast and excels at large-scale information extraction tasks. It's written from the ground up in carefully memory-managed Cython.
- Gensim : This is also super fast and implemented in cython. It is specifically used for topic modelling.
- Sklearn : Sklearn also provides a few libraries which are a combination of NLP and machine learning algorithms.

Part of speech(POS) tagging

In corpus linguistics, part-of-speech tagging (POS tagging), is the process of marking up a word in a text (corpus) as corresponding to a particular part of speech, based on both its definition and its context. A simplified form of this is the identification of words as nouns, verbs, adjectives, adverbs, etc.

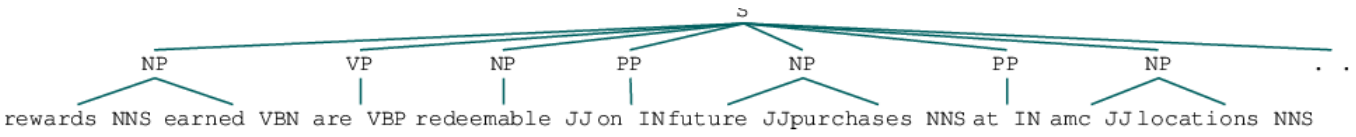
	word	tag	Pos_tag
0	rewards	NNS	NOUN
1	earned	VBN	VERB
2	are	VBP	AUX
3	redeemable	VBN	VERB
4	on	IN	ADP
5	future	JJ	ADJ
6	purchases	NNS	NOUN
7	at	IN	ADP
8	amc	NNP	PROPN
9	locations	NNS	NOUN
10	.	.	PUNCT

Chunking

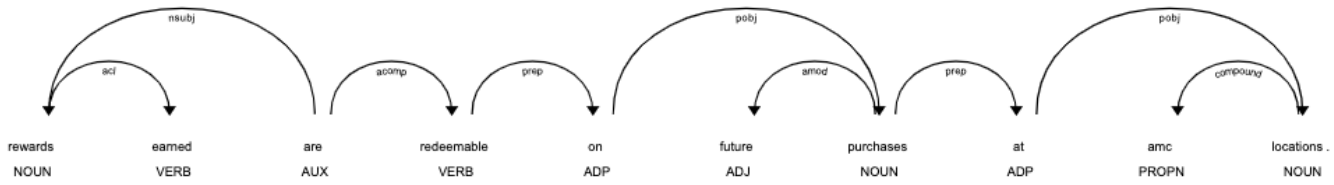
Shallow parsing (also chunking) is an analysis of a sentence which first identifies constituent parts of sentences (nouns, verbs, adjectives, etc.) and then links them to higher order units that have discrete grammatical meanings (noun groups or phrases, verb groups, etc.).

```
[('Chancellor', 'NNP', 'O'),
 ('of', 'IN', 'B-PP'),
 ('the', 'DT', 'B-NP'),
 ('Exchequer', 'NNP', 'I-NP'),
 ('Nigel', 'NNP', 'B-NP'),
 ('Lawson', 'NNP', 'I-NP'),
 ("'s", 'POS', 'B-NP'),
 ('restated', 'VBN', 'I-NP'),
 ('commitment', 'NN', 'I-NP'),
 ('to', 'TO', 'B-PP'),
 ('a', 'DT', 'B-NP'),
 ('firm', 'NN', 'I-NP'),
 ('monetary', 'JJ', 'I-NP'),
 ('policy', 'NN', 'I-NP'),
 ('has', 'VBZ', 'B-VP'),
 ('helped', 'VBN', 'I-VP'),
 ('to', 'TO', 'I-VP'),
 ('prevent', 'VB', 'I-VP'),
 ('a', 'DT', 'B-NP'),
 ('freefall', 'NN', 'I-NP'),
 ('in', 'IN', 'B-PP'),
 ('sterling', 'NN', 'B-NP'),
 ('over', 'IN', 'B-PP'),
 ('the', 'DT', 'B-NP'),
 ('past', 'JJ', 'I-NP'),
 ('week', 'NN', 'I-NP'),
 ('.', '.', 'O')]
```

Chunk Trees



nltk library treebank



spaCy library displacy

NER

Named-entity recognition (NER) (also known as (named) entity identification, entity chunking, and entity extraction) is a subtask of information extraction that seeks to locate and classify named entities mentioned in unstructured text into pre-defined categories such as person names, organizations, locations, medical codes, time expressions, quantities, monetary values, percentages, etc.

currency volatilities may continue , which may significantly impact (either positively or negatively) our reported results and consolidated trends and comparisons.for additional information about each line item addressed above , refer to item 8 CARDINAL of part ii , financial statements and supplementary data note 1 CARDINAL description of business , accounting policies , and supplemental disclosures.our annual DATE report on form 10-k QUANTITY for the year ended december 31 , 2019 DATE includes a discussion and analysis of our financial condition and results of operations for the year ended december 31 , 2018 DATE in item 7 CARDINAL of part ii , managements discussion and analysis of financial condition and results of operations.effects of covid-19 the covid-19 pandemic and resulting global disruptions have affected our businesses , as well as those of our customers , suppliers , and third ORDINAL -party sellers .

EDA with word clouds

We used sklearn CountVectorizer and Bag of words concept to build simple word cloud for some EDA on the reports data.

Fetching historical data

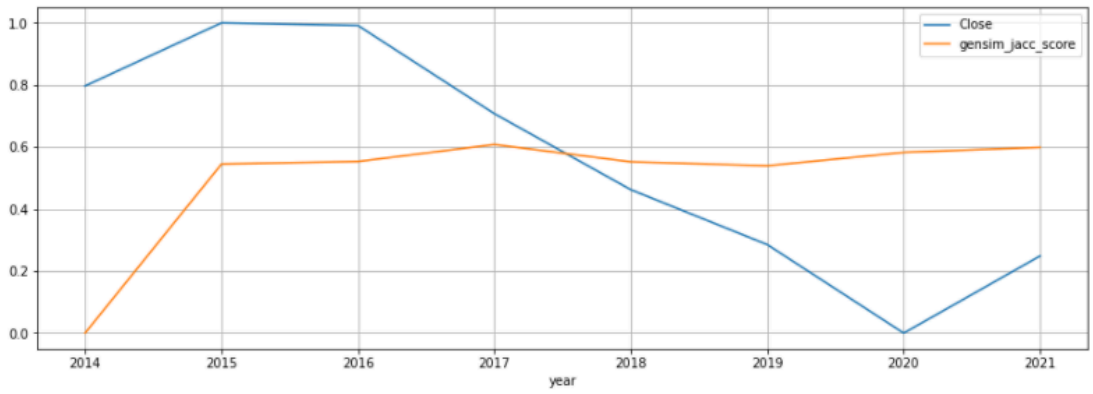
We fetch historical stock data using an open source library
https://pypi.org/project/yfinance/

Finding similarity in Documents

A dictionary/Corpus is created with all the words of all the documents.
We then use the below similarity algorithms to find similarity scores between annual reports of consecutive years. The change in similarity is then plotted against the change in stock prices for a particular company.

Bag of words & Jaccard Similarity

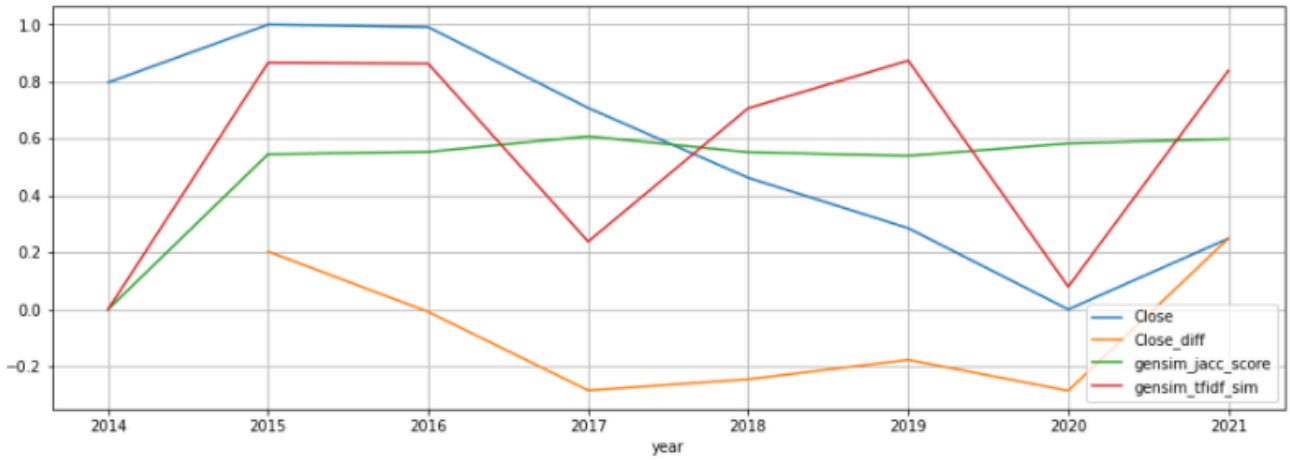
A simple algorithm would be to get the bag of words for each document and compare these bows using gensim.matutils library jaccard. This however doesn't give a very strong trend.



TF-IDF & SparseMatrixSimilarity

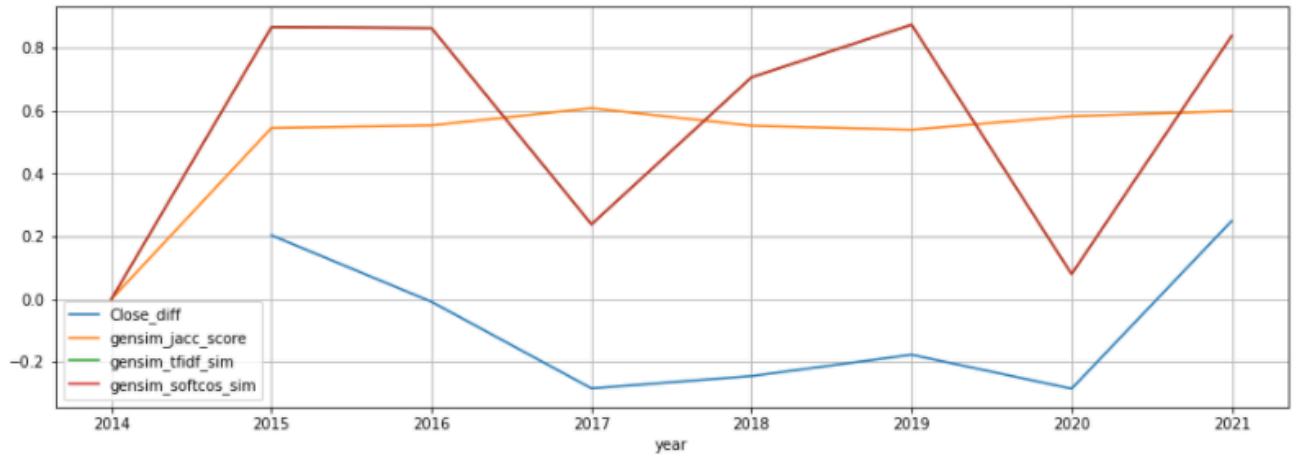
Term frequency - Inverse document frequency

TF-IDF is a statistical measure that evaluates how relevant a word is to a document in a collection of documents. We use TF-idf vectors and compare these for different documents using SparseMatrixSimilarity.

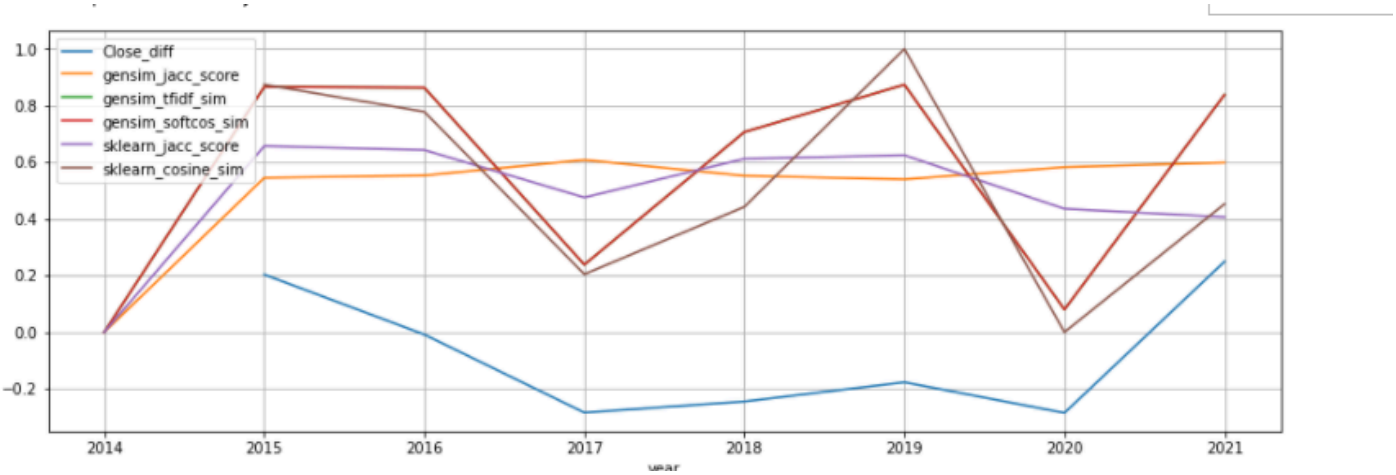


Word2Vec & SoftCosineSimilarity

This doesn't give us results similar to tf-idf.



Sklearn CountVectorizer & jaccard similarity & sklearn TfidfVectorizer & cosine similarity



Sentiment vectors

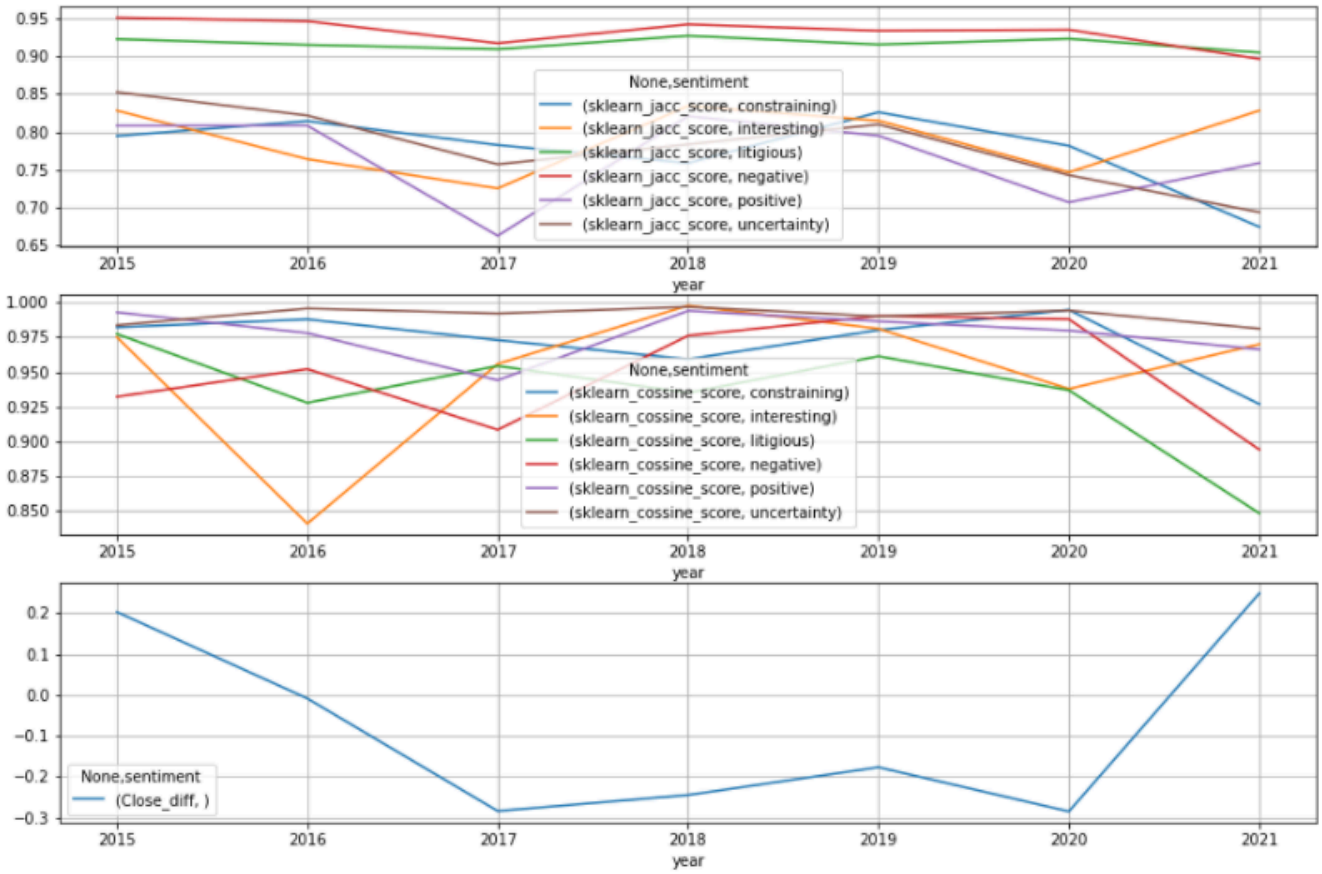
As the comparison of similarity between documents is not giving a strong trend when compared to stock prices we decided to check similarity based off sentiments.

A famous dataset of sentiments is released by LoughranMcDonald specifically for accounting and finance. Hence this is the most relevant dataset to be used.

https://sraf.nd.edu/textual-analysis/resources/

We further get the tf-idf vectors for each sentiment for each 10-k document for a company and perform document similarity of these individual vectors.

The resulting graph shows patterns in the "positive" sentiment which is similar to the stock prices trend observed in the last graph below.



Further Scope

State of the art deep learning algorithms are now being used for frequently for NLP tasks. With transformer based machine learning techniques like BERT the usage of pre-trained models on huge corpus of text data is more prevalent like that used for images. RNNs and LSTMs are on the forefront of these algorithms and promise to give better results at tasks such as sentiment analysis and document similarities.