BAIT 508 2025W1 Group Project

Section BA2 - Group 35

Group Members:

Name: Rohan Jasani
Email: jasani.rohan@gmail.com
Student ID: 26972711
Role: Responsible for:

- Completing Part 1: Quantitative Analysis of the Industry Sector
- Final code file compilation
- Submission

Name: Anmol Saluja
Email: anmolsaluja045@gmail.com
Student ID: 21028691
Role: Responsible for completing Part 2 with formatting: Text Analysis on the Industry Sector

Name: Haho Presto
Email: hansjoshua.presto@gmail.com
Student ID: 55149413
Role: Responsible for:

- Completing Part 3: Analysis of One Sample Firm,
- Final document compilation, formatting, and editing

Github : https://github.com/goatcheese98/BAIT-508-Group-Project

Project Report

Project Purpose:
The purpose of this project is to perform an analysis of public US-listed firms within a selected industry sector using a variety of tools and techniques learned throughout the course. The following sections will discuss the findings from our quantitative, text, and focused comprehensive analysis.

Part 1- Quantitative Analysis of the Industry Sector

Part A - Industry Sector Selection and Data Filtering

For this report, our team has decided to focus on the **Health Services** sector. From the provided project data file 'major_groups.csv', we confirmed that the description of the sector matches our interpretation of the industry, and then filtered the dataset using the major group prefix on SIC codes to only keep firms with codes that start with "80".

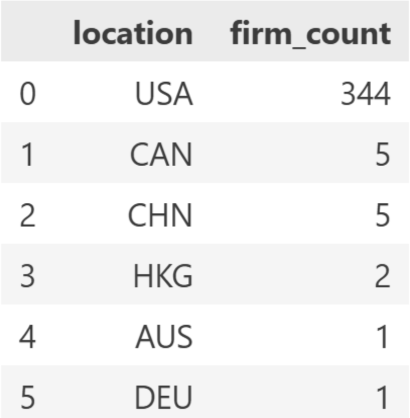
With the filtered dataset, we found that there were **3,064 unique firm-year observations** between fiscal years 1994 and 2020. Within this sample, we found that there were **358 unique firms**, of which only **2 firms** ("*Tenet Healthcare Corp*" and "*DaVita Inc*") have complete records across all 27 fiscal years.

Part B - Preliminary Analysis

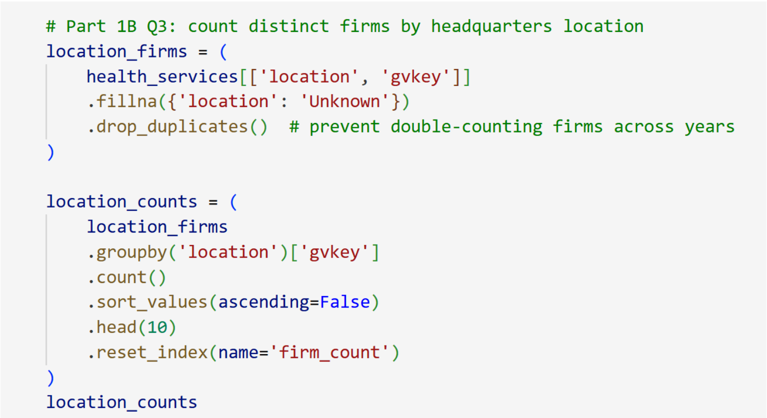
To get an understanding of the industry, the team looked to identify companies with the highest stock prices in the latest fiscal year available to see who the biggest players currently within it are. We see that the company with the highest current stock price in this sector (**Chemed Corp** at **$532.61/share**) has an abnormally high price, whereas the next 9 are in the $117/share to $293/share stock prices. By multiplying the share price (column = "prcc_c") with the shares outstanding (in millions) (column = "ch"), we are able to assess each company's market capitalization.

The team analyzed and observed what revenues have been achieved historically to get a rough estimate of the kind of revenue scale companies at the top of this industry are working with. **Figure 5** and **Figure 6** show that the top 10 revenues achieved historically (from **2010 - 2020**) by any company ranges from **$30.7B USD** to **$50.1B USD**. Interestingly, they seem to be from one firm, indicating strong market dominance by one player in the sector. The output table can be seen in the below figure.

The team also was looking at the geographical distribution of these companies to see what companies and markets are represented in this dataset and sector. The output below shows that almost the entirety of the companies in this filtered dataset are in the US (**344 firms** or **96%** of the unique firms), with smaller clusters in Canada (**5 firms**), China (**5 firms**), Hong Kong (**2 firms**) and single entries from Australia and Germany.

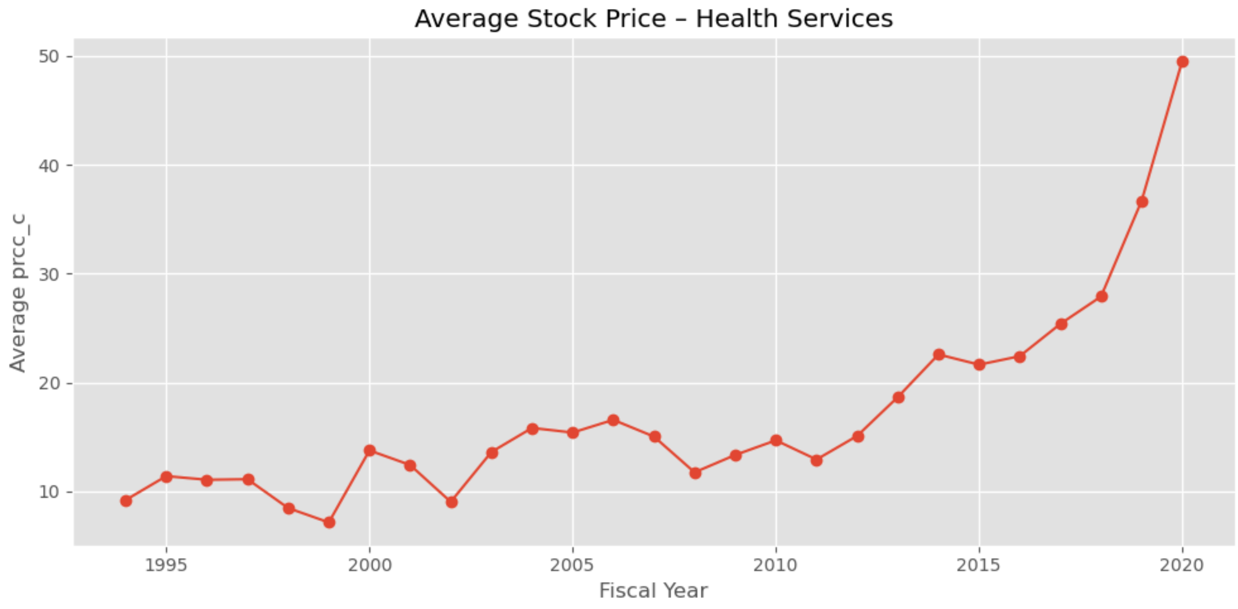


	location	firm_count
0	USA	344
1	CAN	5
2	CHN	5
3	HKG	2
4	AUS	1
5	DEU	1



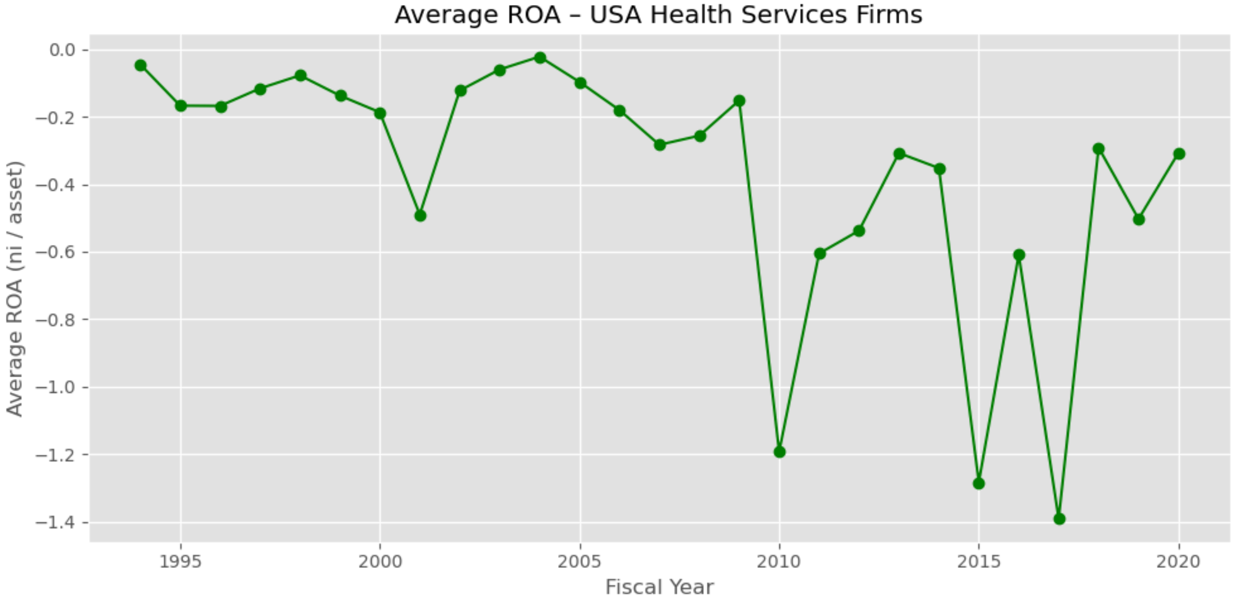
```
# Part 1B Q3: count distinct firms by headquarters location
location_firms = (
    health_services[['location', 'gvkey']]
    .fillna({'location': 'Unknown'})
    .drop_duplicates()  # prevent double-counting firms across years
)

location_counts = (
    location_firms
    .groupby('location')['gvkey']
    .count()
    .sort_values(ascending=False)
    .head(10)
    .reset_index(name='firm_count')
)
location_counts
```

To get an idea and visual of the growth of the industry, we compared the average stock price in 1994 (about **$9/share**) to the average stock price in 2020 (about **$50/share**) in the industry, which on a geometric average (or via the industry term “CAGR” Compound Annual Growth Rate) of **6.82%**. This shows that over the long term, the sector is growing despite some year-to-year variation and dips as shown in figure below, though slightly underperforming the broad market (using the S&P 500 as a reference index).



Within the sector, the company “*Insight Health Services Holding”* suffered the largest stock-price decrease during the financial crisis of 2008, falling from a 2007 stock price of **$3.00** to literal pennies in 2008 at **$0.02**, representing a **99.3%** drop in stock price in one year.

Interestingly, across the sector in the USA, average ROA remains negative through the historical data from **(-) 4%** in 1994 to **- (31%)** in 2020 as shown in the figure below. Initially, it might seem that it is an unattractive industry to invest in, but the average stock price increasing shows a different story. The team’s hypothesis of the data, combined with our understanding of the industry, is that a better explanation might be that the sector is highly asset intensive, where returns on assets are realized over decades and perhaps lost in the nuance of year to year.



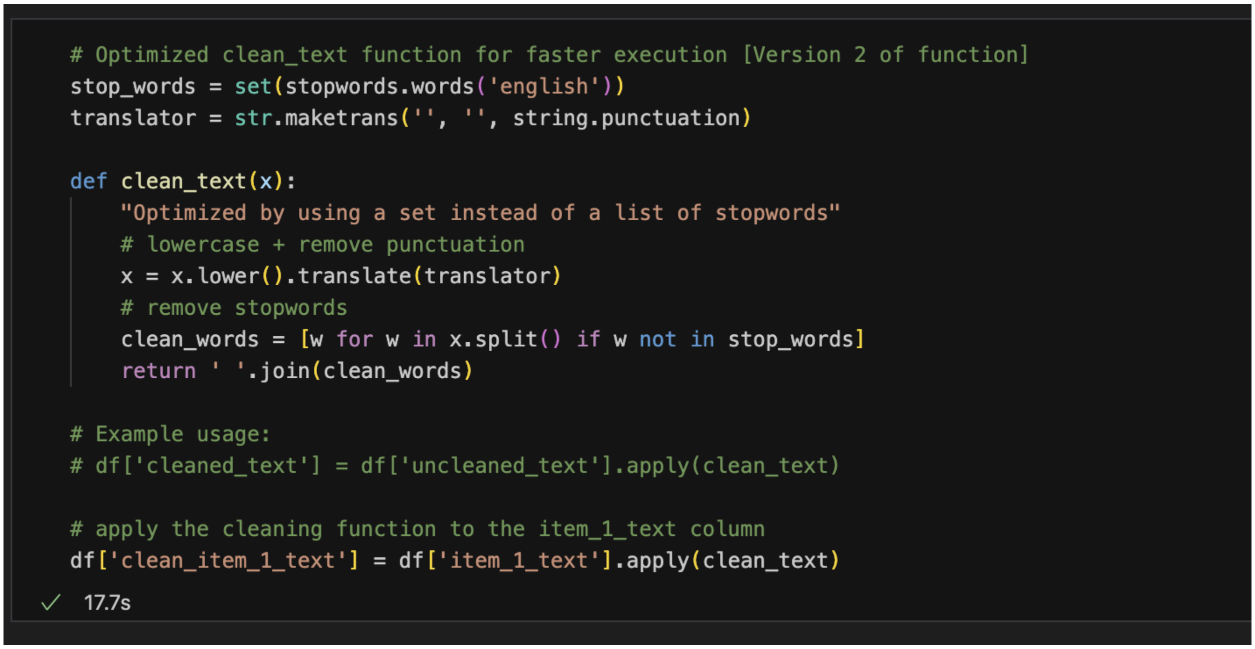
Part 2 - Text Analysis on the Industry Sector

The team sought to understand the sector through a more qualitative lens using the tools of text analysis. The following sections describe the methodologies the team used and the findings of sector sentiment that were discovered.

Part C - Text Cleaning

Text cleaning, and more broadly, data cleaning, is an essential step that should be performed before any deep analysis is conducted otherwise, the resulting output may contain inexplicable errors due to the data not being properly cleaned. Data from the file "**2020_10K_item1_full.csv**" was first loaded into a Pandas dataframe and subsequently cleaned to facilitate the efficient running of the analysis. The team followed the steps to clean the text in the '**item_1_text**' column using a user-defined Python function, '**clean_text()**'. The function operates as follows:

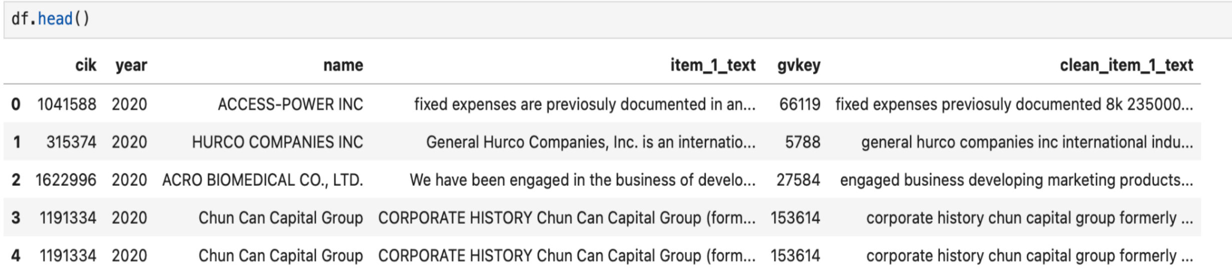
1. It first **lowercases** the text
2. It then removes all **punctuation**
3. Lastly, it removes **stopwords** using the nltk package.



```
# Optimized clean_text function for faster execution [Version 2 of function]
stop_words = set(stopwords.words('english'))
translator = str.maketrans('', '', string.punctuation)

def clean_text(x):
    "Optimized by using a set instead of a list of stopwords"
    # lowercase + remove punctuation
    x = x.lower().translate(translator)
    # remove stopwords
    clean_words = [w for w in x.split() if w not in stop_words]
    return ' '.join(clean_words)

# Example usage:
# df['cleaned_text'] = df['uncleaned_text'].apply(clean_text)

# apply the cleaning function to the item_1_text column
df['clean_item_1_text'] = df['item_1_text'].apply(clean_text)
```

✓ 17.7s

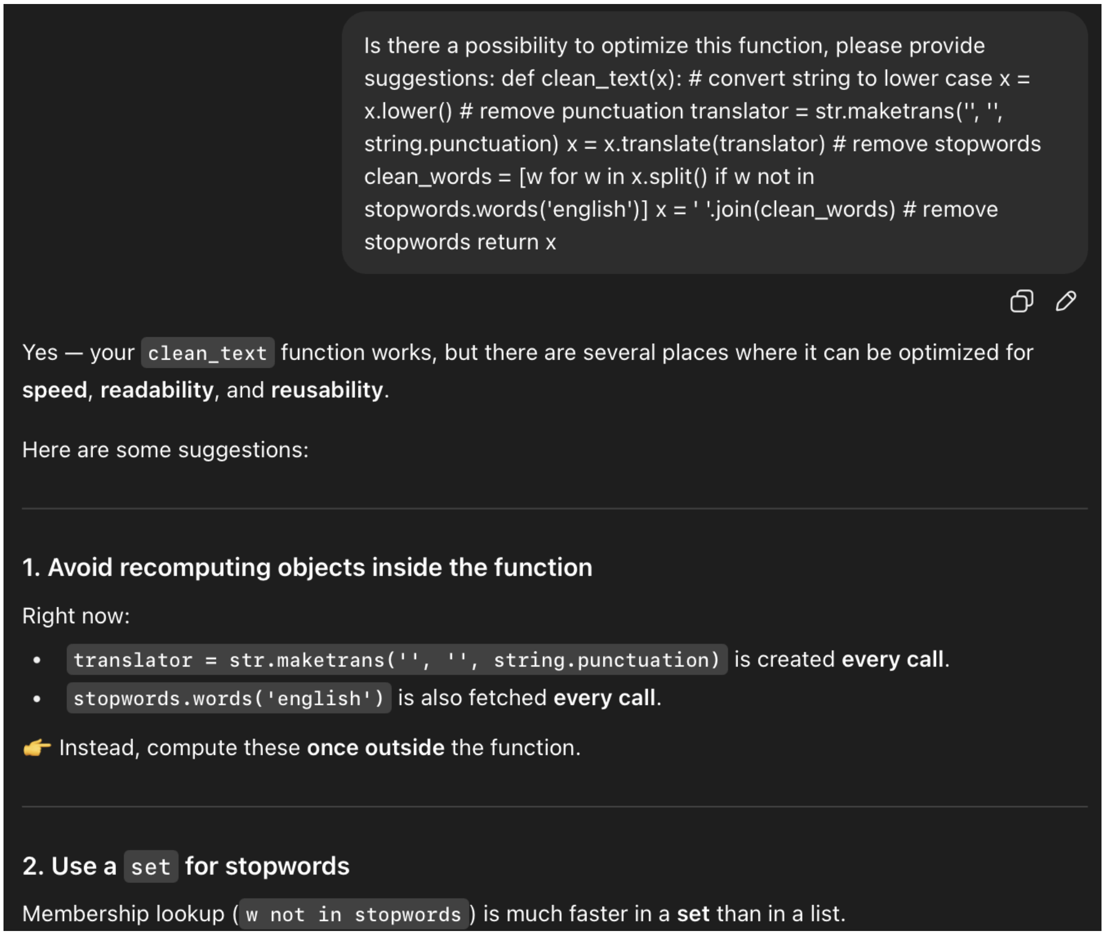
Here is the snapshot of the dataframe that includes the new clean column named "**clean_item_1_text**".



```
df.head()
```

	cik	year	name	item_1_text	gvkey	clean_item_1_text
0	1041588	2020	ACCESS-POWER INC	fixed expenses are previously documented in an...	66119	fixed expenses previously documented 8k 235000...
1	315374	2020	HURCO COMPANIES INC	General Hurco Companies, Inc. is an internatio...	5788	general hurco companies inc international indu...
2	1622996	2020	ACRO BIOMEDICAL CO., LTD.	We have been engaged in the business of develo...	27584	engaged business developing marketing products...
3	1191334	2020	Chun Can Capital Group	CORPORATE HISTORY Chun Can Capital Group (form...	153614	corporate history chun capital group formerly ...
4	1191334	2020	Chun Can Capital Group	CORPORATE HISTORY Chun Can Capital Group (form...	153614	corporate history chun capital group formerly ...

The team noticed that the initial version of the function created was longer than expected to run, so they optimized the function by asking ChatGPT to provide a more optimized version, which accomplishes the same task using fewer computational resources. Here is the snapshot of the suggestion:



Is there a possibility to optimize this function, please provide suggestions: def clean_text(x): # convert string to lower case x = x.lower() # remove punctuation translator = str.maketrans('', '', string.punctuation) x = x.translate(translator) # remove stopwords clean_words = [w for w in x.split() if w not in stopwords.words('english')] x = ' '.join(clean_words) # remove stopwords return x

Yes — your `clean_text` function works, but there are several places where it can be optimized for **speed**, **readability**, and **reusability**.

Here are some suggestions:

1. Avoid recomputing objects inside the function

Right now:

- `translator = str.maketrans('', '', string.punctuation)` is created **every call**.
- `stopwords.words('english')` is also fetched **every call**.

👉 Instead, compute these **once outside** the function.

2. Use a `set` for stopwords

Membership lookup (`w not in stopwords`) is much faster in a **set** than in a list.

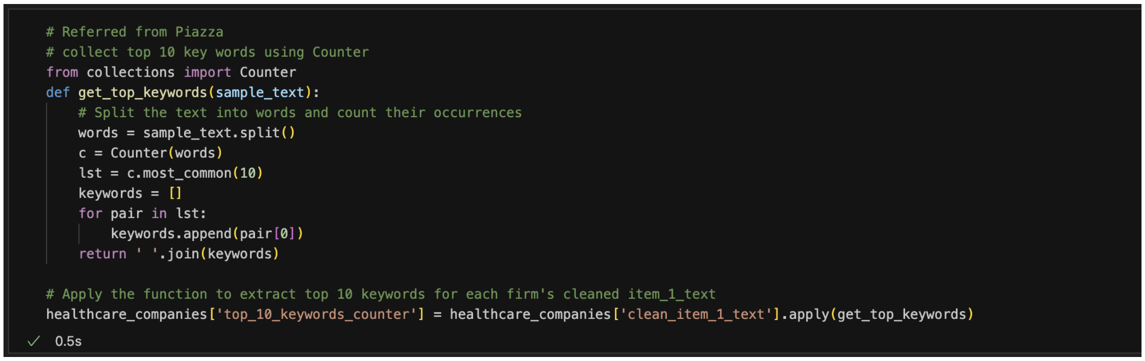
The final run-time of the function was reduced to a few **minutes** from **hours** after implementing the suggestion from ChatGPT.

Part D - Keyword Analysis

This next section describes the steps taken to perform a keyword analysis, aiming to provide a qualitative representation of the sector using words that reflect the human sentiment of the sector.

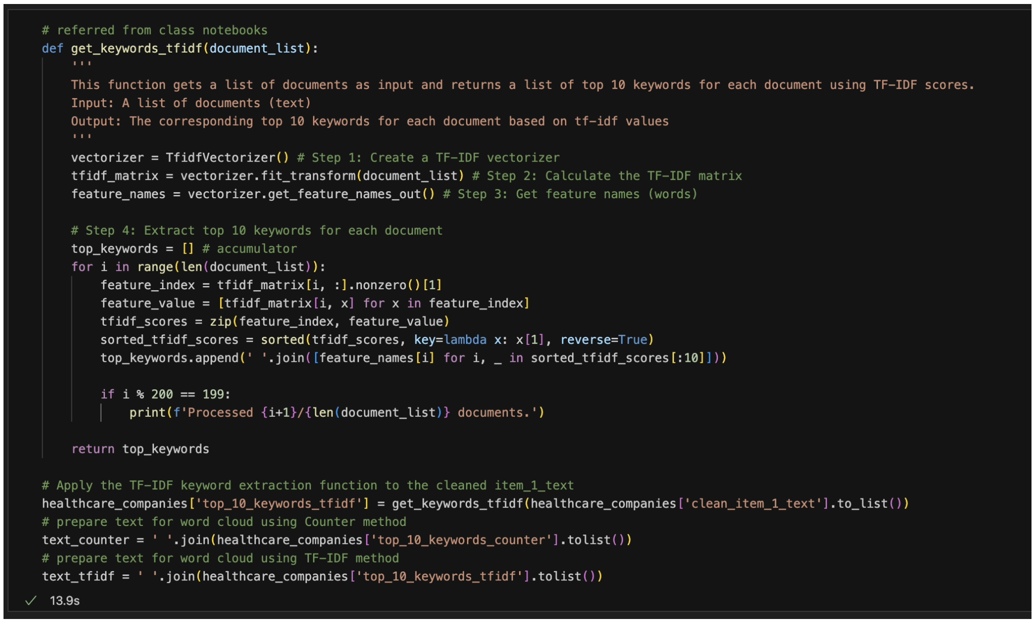
The data was first loaded from the “**public_firm.csv**” file into a Pandas DataFrame and filtered for rows where the SIC code matched the SIC code for Health Services. An inner join was used to link the cleaned DataFrame created in Part C, filtering only the public_firms dataset to retain firms within the sector of interest.
This new, filtered DataFrame was named **'healthcare_companies'**, which we then use as a base to perform the analysis on. Two new columns were created to contain the top 10 words in each cleaned text present in “**clean_item_1_text”** using two different approaches:

Method 1: Word Counter approach
Method 2: TF-IDF score approach



```
# Referred from Piazza
# collect top 10 key words using Counter
from collections import Counter
def get_top_keywords(sample_text):
    # Split the text into words and count their occurrences
    words = sample_text.split()
    c = Counter(words)
    lst = c.most_common(10)
    keywords = []
    for pair in lst:
        keywords.append(pair[0])
    return ' '.join(keywords)

# Apply the function to extract top 10 keywords for each firm's cleaned item_1_text
healthcare_companies['top_10_keywords_counter'] = healthcare_companies['clean_item_1_text'].apply(get_top_keywords)
```
✓ 0.5s

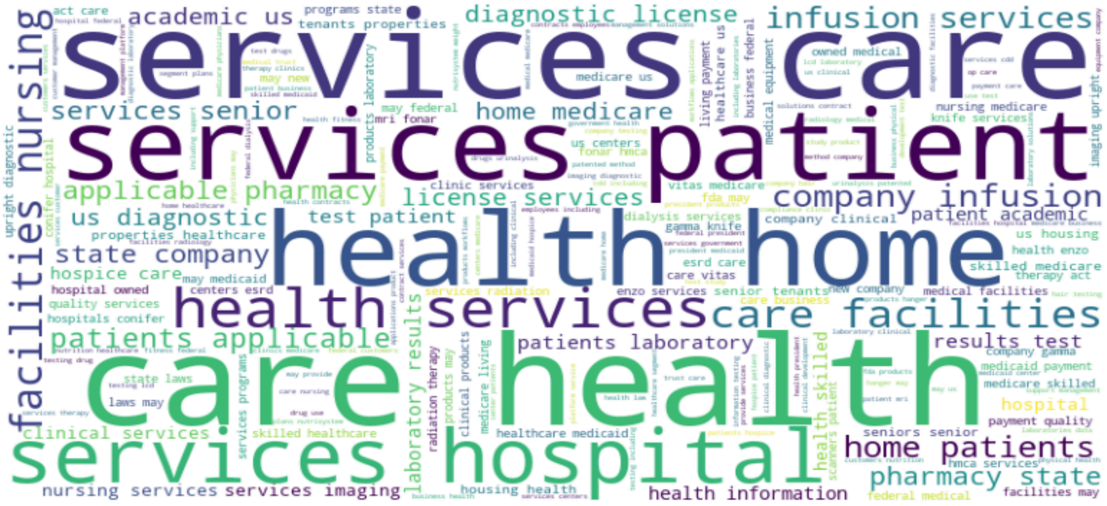
Snapshot of the function for the **Method 1**:



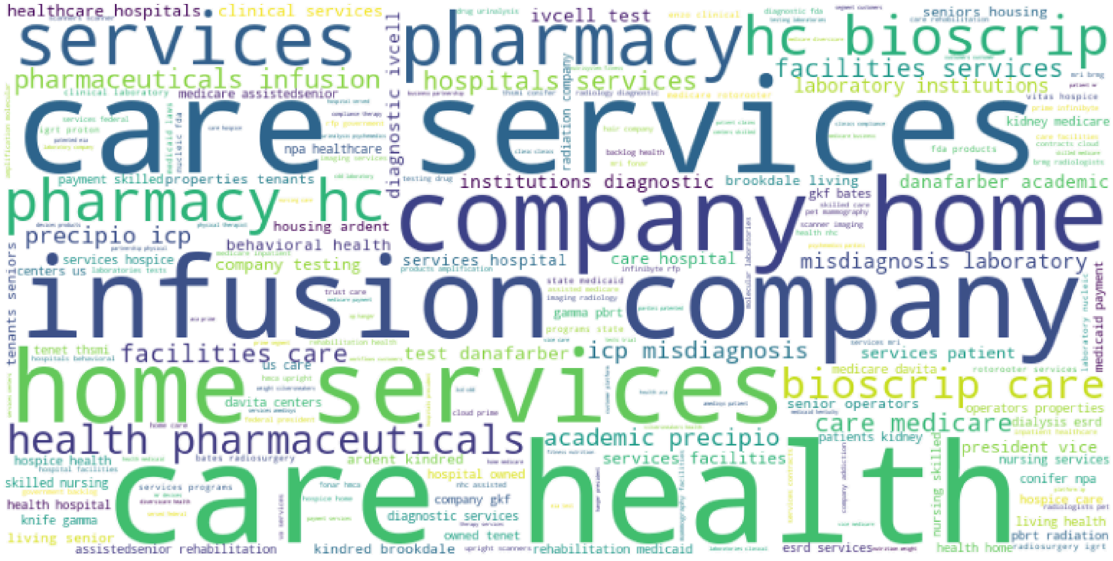
```
# referred from class notebooks
def get_keywords_tfidf(document_list):
    '''
    This function gets a list of documents as input and returns a list of top 10 keywords for each document using TF-IDF scores.
    Input: A list of documents (text)
    Output: The corresponding top 10 keywords for each document based on tf-idf values
    '''
    vectorizer = TfidfVectorizer() # Step 1: Create a TF-IDF vectorizer
    tfidf_matrix = vectorizer.fit_transform(document_list) # Step 2: Calculate the TF-IDF matrix
    feature_names = vectorizer.get_feature_names_out() # Step 3: Get feature names (words)

    # Step 4: Extract top 10 keywords for each document
    top_keywords = [] # accumulator
    for i in range(len(document_list)):
        feature_index = tfidf_matrix[i, :].nonzero()[1]
        feature_value = [tfidf_matrix[i, x] for x in feature_index]
        tfidf_scores = zip(feature_index, feature_value)
        sorted_tfidf_scores = sorted(tfidf_scores, key=lambda x: x[1], reverse=True)
        top_keywords.append(' '.join([feature_names[i] for i, _ in sorted_tfidf_scores[:10]]))

        if i % 200 == 199:
            print(f'Processed {i+1}/{len(document_list)} documents.')

    return top_keywords

# Apply the TF-IDF keyword extraction function to the cleaned item_1_text
healthcare_companies['top_10_keywords_tfidf'] = get_keywords_tfidf(healthcare_companies['clean_item_1_text'].to_list())
# prepare text for word cloud using Counter method
text_counter = ' '.join(healthcare_companies['top_10_keywords_counter'].tolist())
# prepare text for word cloud using TF-IDF method
text_tfidf = ' '.join(healthcare_companies['top_10_keywords_tfidf'].tolist())
```
✓ 13.9s

Snapshot of the function for the **Method 2**:

Based on the top 10 word list, the team created two different word clouds, which help visualize the differences in the top listed words from the two approaches. The two word clouds are shown below:



Wordcloud generated from **Method 1: Counter Approach**



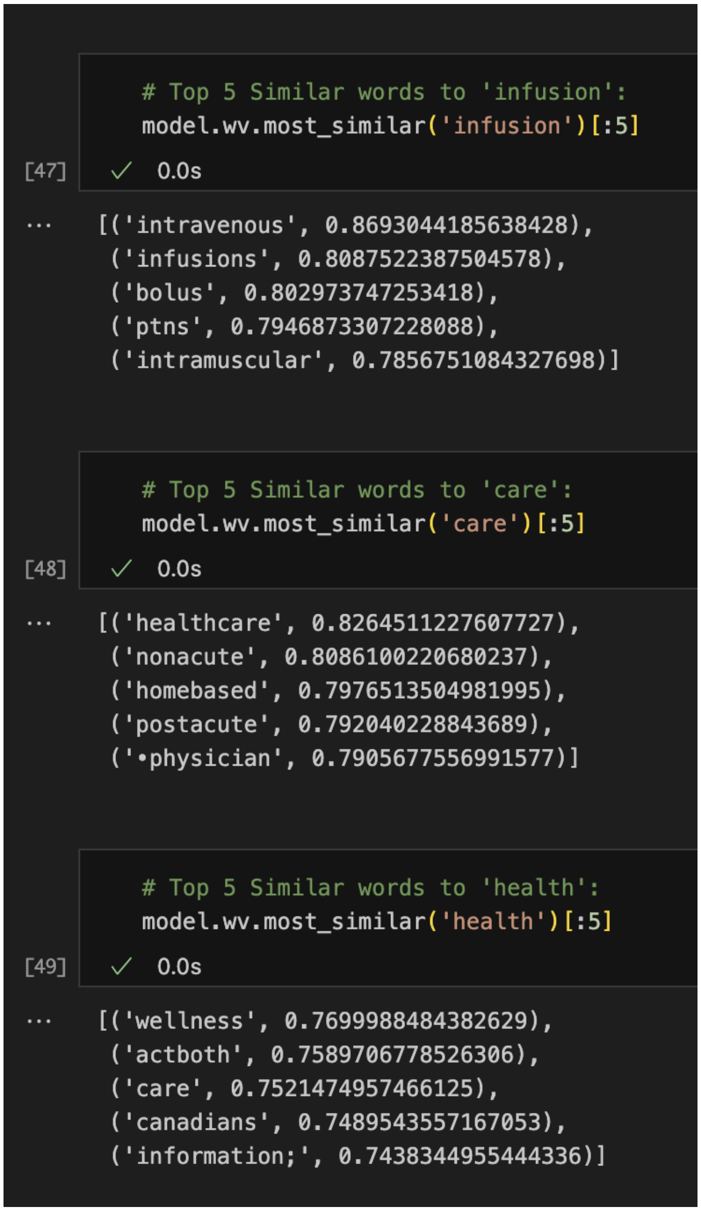
Wordcloud generated from **Method 2: TF-IDF Score Approach**

It can be seen that “**services**”, “**care**”, and “**health**” seem to be mentioned more frequently than “**infusion”**, based on word count. While TF-IDF score also prioritizes “**infusion”** with “**health”**, “**care”**, and “**services”**. This shows the difference and intuition between the two approaches: **Method 1** focuses on volume, while **Method 2** values the significance of the word in the document.

Part E - Word Embedding

A Word2Vec model was trained with the following configuration:
min_count=5; vector_size=50; workers=3; window=5; sg = 1

The team selected three different words from the word cloud created in Part D: “**infusion**”, “**care**”, and “**health**”, which will be used later in the report to find companies with similar sentiments. The Word2Vec model was then used to identify the 5 most similar words for each of the three selected terms. Looking through similar words, the team found some that we could relate to intuitively, but other word relationships were surprising, such as “**care**” and “**HCBS**”. Examining these similar words provided the team with a notional, sentimental understanding of the words used in this sector, which can help the team better contextualize some of the quantitative findings.



```
     # Top 5 Similar words to 'infusion':
     model.wv.most_similar('infusion')[:5]
[47]  ✓  0.0s

...  [('intravenous', 0.8693044185638428),
      ('infusions', 0.8087522387504578),
      ('bolus', 0.802973747253418),
      ('ptns', 0.7946873307228088),
      ('intramuscular', 0.7856751084327698)]

     # Top 5 Similar words to 'care':
     model.wv.most_similar('care')[:5]
[48]  ✓  0.0s

...  [('healthcare', 0.8264511227607727),
      ('nonacute', 0.8086100220680237),
      ('homebased', 0.7976513504981995),
      ('postacute', 0.792040228843689),
      ('•physician', 0.7905677556991577)]

     # Top 5 Similar words to 'health':
     model.wv.most_similar('health')[:5]
[49]  ✓  0.0s

...  [('wellness', 0.7699988484382629),
      ('actboth', 0.7589706778526306),
      ('care', 0.7521474957466125),
      ('canadians', 0.7489543557167053),
      ('information;', 0.7438344955444336)]
```

Part 3 - Comprehensive Analysis of One Firm

Part F - Firm Analysis and Strategy Suggestion

Identifying Focal Firm's Competing Firms
The team was initially looking to identify a firm to focus on. With no particular personal preferences for any company in the dataset, the team elected to choose at random a company in the top 20 by market share. From here, the team selected the company "TELADOC HEALTH INC" (gvkey = 24249).

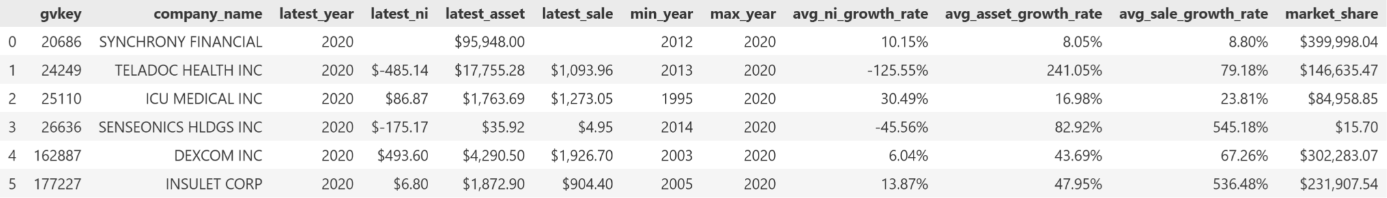
Using the full dataset, the team trained a Word2Vec model on the cleaned data. The team then used keywords suggested in the previous section ["**infusion**", "**care**", "**health**"] to create and assign vectors to words. The top 5 most similar words to this input group, came out to be the following: ["**healthcare**", "**employerfunded**", "**homebased**", "**accountable**", and "**dietitian**"].

Using these vectors and embeddings, the team summed and assigned the total embedding values to each firm. Using the the focal firm's unique identifier (gvkey = 24249) as an input, the team identified the 5 most similar companies as quantified by total embedding values which are the following: ["**INSULET CORP**", "**Senseonics Holdings, Inc.**", "**DEXCOM Inc.**", "**Synchrony Financial**", "**ICU MEDICAL INC/DE**"]

Competitive Analysis
Having identified the firms which the focus firm is competing with (at least from a sentiment similarity perspective), we can use the public_firms.csv dataset to compare various metrics including latest revenues, market capitalization, asset value, and return on assets. Further, we looked at the arithmetic average growth rates, which can be seen in the figure below.

Relative to other firms, Teladoc Health Inc., the team observes that they:

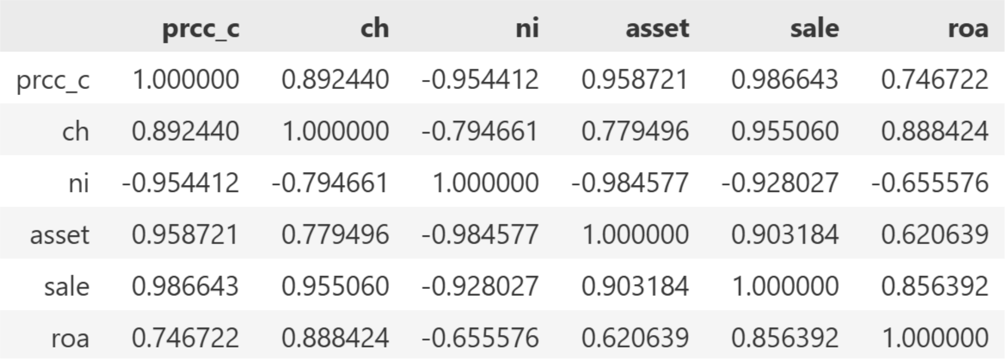
- in the most recent year, have the lowest annual earnings of **-$485.14M**
- Also saw a big increase in losses with a negative net income growth rate of **`125.55%**
- have dramatically increased their asset value by over 3x (**241.05%**) where the next highest asset growth firm only went up by **82.92%.**
- have increased their revenues by **79.18**% from the previous year, which for company is good but actually markedly lower than other competitors
- have a market share of **$146M,** which is about middle of the pack in this group of companies



	gvkey	company_name	latest_year	latest_ni	latest_asset	latest_sale	min_year	max_year	avg_ni_growth_rate	avg_asset_growth_rate	avg_sale_growth_rate	market_share
0	20686	SYNCHRONY FINANCIAL	2020		$95,948.00		2012	2020	10.15%	8.05%	8.80%	$399,998.04
1	24249	TELADOC HEALTH INC	2020	$-485.14	$17,755.28	$1,093.96	2013	2020	-125.55%	241.05%	79.18%	$146,635.47
2	25110	ICU MEDICAL INC	2020	$86.87	$1,763.69	$1,273.05	1995	2020	30.49%	16.98%	23.81%	$84,958.85
3	26636	SENSEONICS HLDGS INC	2020	$-175.17	$35.92	$4.95	2014	2020	-45.56%	82.92%	545.18%	$15.70
4	162887	DEXCOM INC	2020	$493.60	$4,290.50	$1,926.70	2003	2020	6.04%	43.69%	67.26%	$302,283.07
5	177227	INSULET CORP	2020	$6.80	$1,872.90	$904.40	2005	2020	13.87%	47.95%	536.48%	$231,907.54

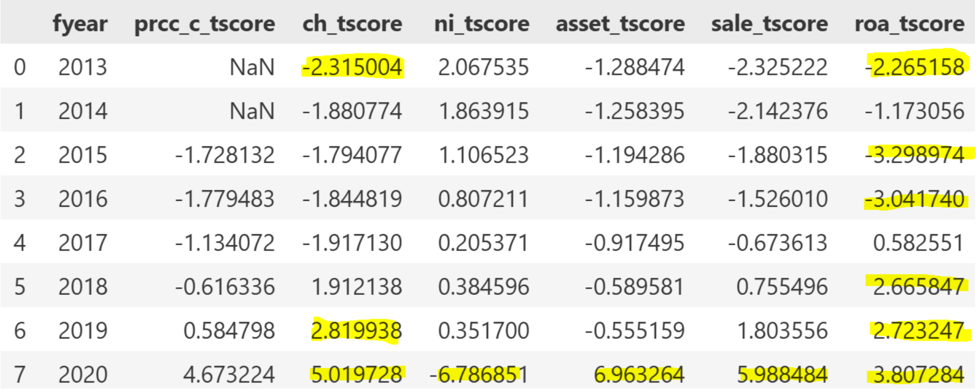
Focus Firm Analysis:
Looking at the historical data of just the focus firm as shown in **Figure X,** the team was interested to see if there are any correlations between the data columns. Given that some of these columns are dependent on each other (for example, return on assets = net income / assets), the team expected some auto correlation.

In creating a correlation matrix in the figure below, we see that indeed there is a strong positive correlation between share price and sales/revenue, almost near perfect at **0.986643.** In fact, all the data fields are strongly or very strongly correlated to each other.



	prcc_c	ch	ni	asset	sale	roa
prcc_c	1.000000	0.892440	-0.954412	0.958721	0.986643	0.746722
ch	0.892440	1.000000	-0.794661	0.779496	0.955060	0.888424
ni	-0.954412	-0.794661	1.000000	-0.984577	-0.928027	-0.655576
asset	0.958721	0.779496	-0.984577	1.000000	0.903184	0.620639
sale	0.986643	0.955060	-0.928027	0.903184	1.000000	0.856392
roa	0.746722	0.888424	-0.655576	0.620639	0.856392	1.000000

The team also wanted to see if there were any noteworthy changes in any of the years for the firm. To understand this, the team calculated the t-scores (as n=8 years of data) to get a rough estimate of which years and which data columns are far outside what that data suggest. Indeed, we see some fields such as **ROA in 2015, 2016** having t-scores of >|3|, indicating that it had less than 0.5% of happening per the central limit theorem. Fiscal year 2020 was collectively an outlier across all measured variables, indicating it was likely a year of both high capital investment and raising capital to make that investment.



	fyear	prcc_c_tscore	ch_tscore	ni_tscore	asset_tscore	sale_tscore	roa_tscore
0	2013	NaN	-2.315004	2.067535	-1.288474	-2.325222	-2.265158
1	2014	NaN	-1.880774	1.863915	-1.258395	-2.142376	-1.173056
2	2015	-1.728132	-1.794077	1.106523	-1.194286	-1.880315	-3.298974
3	2016	-1.779483	-1.844819	0.807211	-1.159873	-1.526010	-3.041740
4	2017	-1.134072	-1.917130	0.205371	-0.917495	-0.673613	0.582551
5	2018	-0.616336	1.912138	0.384596	-0.589581	0.755496	2.665847
6	2019	0.584798	2.819938	0.351700	-0.555159	1.803556	2.723247
7	2020	4.673224	5.019728	-6.786851	6.963264	5.988484	3.807284

Suggestion to the Focal Firm:
The team's suggestion to the firm is quite simple, and that is to **not make any further capital investments** for the next **5 years**. Instead, focus on **reducing the firm's operating costs** while **maintaining** or **slowly scaling current revenue** levels, such that their net income and ROA trend positively into the future.

Generative AI Attribution:

The following section discloses and attributes the team's use of Generative AI during the writing of this project. Attribution has been organized by report parts.

Part 1

Name, version, company of AI Tool: Codex CLI(by OpenAI), Cursor Tab completion
Chat Objective: You are a professional code reviewer. You will review my code, the data files and the attached pdf (questions for the group). I want you to come up better ways to write/optimize my code. I will provide you with snippets and you will review them and come up with suggestions.
Use of AI-generated content: The AI provided optimized code suggestions, best practices, and efficiency improvements for the function. The code and the functions were verified using https://pandas.pydata.org/docs/ for documentation.
URL of chat history: (Codex CLI cannot share chat history)
Time and date of chat: September 30, 2025

Part 2

Name, version, company of AI Tool: ChatGPT, GPT-5, OpenAI
Chat Objective: To review and optimize a Python clean_text function for performance and efficiency.
Use of AI-generated content: The AI provided optimized code suggestions, best practices, and efficiency improvements for the function.
URL of chat history: https://chatgpt.com/c/68dd73ec-eacc-8328-8b80-318216ea1ee1
Time and date of chat: October 1, 2025, 11:34 AM

Part 3

Name, version, company of AI Tool: Microsoft CoPilot in VS Studio, GPT-4.1
Chat Objective:

- To assist with co-author the generation of the code for more complicated analysis and methods such as:
  - Training a word2vec model
  - Creating summation and statistical summary columns such as arithmetic/geometric averages, t-scores, z-scores

Use of AI-generated content: The AI took my ideas and directions and provided code suggestions and which were further optimized when prompted. Outputs were verified against manual excel-based sample calculations for accuracy
URL of Chat History: (Co-Pilot within Visual Studio cannot share chat history)
Time and Dates of Chat(s):

- [September 30, 2025 7pm; October 1, 2025 8pm; October 2, 2025 8pm]