BAIT 508 Group Project: Industry Analysis
Final Report

Part 1. Quantitative Analysis of the Industry Sector

The goal of this project was to conduct an in-depth analysis of public US firms within the food stores sector using various data analyses and natural language processing (NLP) techniques.

A. Industry Sector Selection and Data Filtering

The file *data/major_groups.csv* contains a list of major industry sectors and their corresponding major group codes. For this project, we've chosen major group **54** — **food stores** — as our industry of interest.

We first imported the data set and stored it as a pandas dataframe ('data'). Then, we filtered the data to only include the firms in major group 54. The dataframe was filtered to only include rows where the 'sic' column starts with '54', which represents firms in the food stores sector. This was done by converting the 'sic' column values into string and filtering the values for those starting with the string '54'. We obtained the *str.startswith()* code from Parewa Labs Pvt. Ltd (n.d.). This filtered dataframe was named 'food_data'. We used *nunique()* to count **27 unique fiscal years** based on the 'fyear' column and **104 unique firms** based on the 'gvkey' column. To find out which firms had data for all 27 years, we calculated the difference between the latest and earliest years for each firm and filtered the data set for firms where the range was 26 (i.e., 2020 - 1994 = 26). **Only one firm, namely Caseys General Stores Inc., possessed data spanning all 27 years from 1994 to 2020**.

B. Preliminary Analysis

Top 10 firms with the highest stock price in 2020

To find the top 10 firms with the highest stock price in 2020, we filtered 'food_data' for data in the year 2020 based on the 'fyear' column and then displayed the top 10 firms with the highest stock price ('prcc_c') in descending order.

1. Caseys General Stores Inc	178.62
2. Weis Markets Inc	47.81
3. Ingles Markets Inc - Cl A	42.66
4. Grocery Outlet Hldng Corp	39.25
5. Kroger Co	31.76
6. Koninklijke Ahold Delhaize	28.28
7. Village Super Market - Cl A	22.06
8. Sprouts Farmers Market	20.10
9. Albertsons Cos Inc	17.58
10. Companhia Brasileira De Dist	14.32

Top 10 firms with the highest sales from 1994 - 2020

Similarly, to find the top 10 firms with the highest sales in the entire history of the dataset, we first grouped 'food_data' by the column 'conm' and calculated the sum of each firm's 'sale' column. This data was stored in the new column ‘total_sales’. We then displayed the top 10 firms with the highest total sales in descending order.

1. Kroger Co	1,959,875.36
2. Koninklijke Ahold Delhaize	1,285,153.20
3. Safeway Inc	650,414.60
4. Publix Super Markets Inc	568,923.55
5. Delhaize Group - Ets Dlhz Fr	420,677.60
6. Albertson's Inc	332,439.04
7. Ito-Yokado Co Ltd	302,131.83
8. Companhia Brasileira De Dist	274,569.64
9. Albertsons Cos Inc	192,680.00
10. Winn-Dixie Stores Inc	180,545.87

Top 10 locations with the highest number of firms

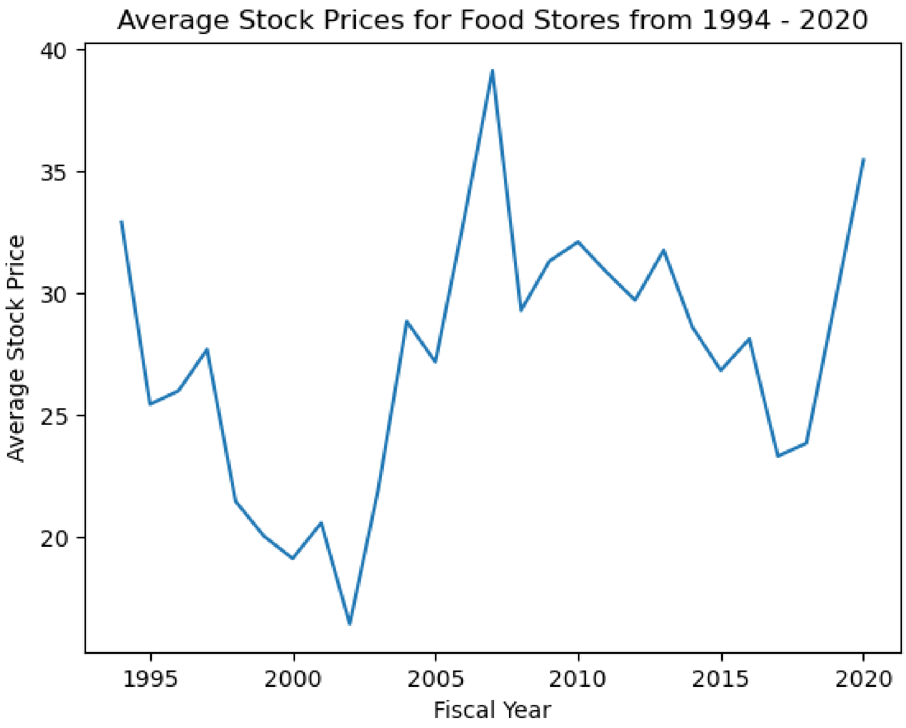
Likewise, to count the number of unique firms in each location in ‘food_data’, we grouped the data by ‘location’ and used *nunique()* to count the unique firms (column ‘conm’) in each location. This information was stored in the new 'firm_count' column. We displayed the top 10 locations with the highest number of unique firms based on this count in descending order.

1. USA (United States of America)	88
2. CHL (Chile)	4
3. CHN (China)	3
4. BRA (Brazil)	2
5. JPN (Japan)	2
6. ARG (Argentina)	1
7. BEL (Belgium)	1
8. CAN (Canada)	1
9. ISR (Israel)	1
10. NLD (Netherlands)	1

It was interesting to see that the United States of America (USA) stood out with the highest number of firms, totalling 88, while the other countries had significantly fewer firms, ranging from just 1 to 4.

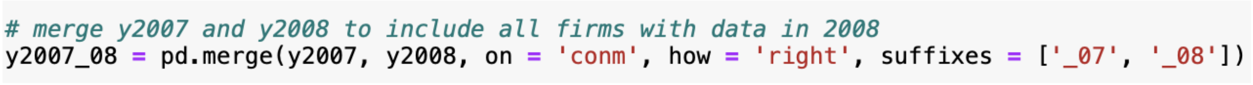
Average stock price across the years

The average stock price of all firms across the years was calculated by grouping ‘food_data’ by ‘fyear’ and storing the mean of ‘prcc_c’ in the pandas dataframe. A line plot with the fiscal year on the x-axis and average stock price on the y-axis was created. The line chart below illustrates the average stock price in the food stores industry sector from 1994 to 2020.



The Effect of the 2008 Financial Crisis

To investigate the effect of the 2008 Financial Crisis, we filtered ‘food_data’ by ‘fyear’ to create two separate dataframes for the year 2007 and the year 2008. We then merged the 2007 and 2008 dataframes on the ‘conm’ column and used a right merge to keep observations that had data for 2008.

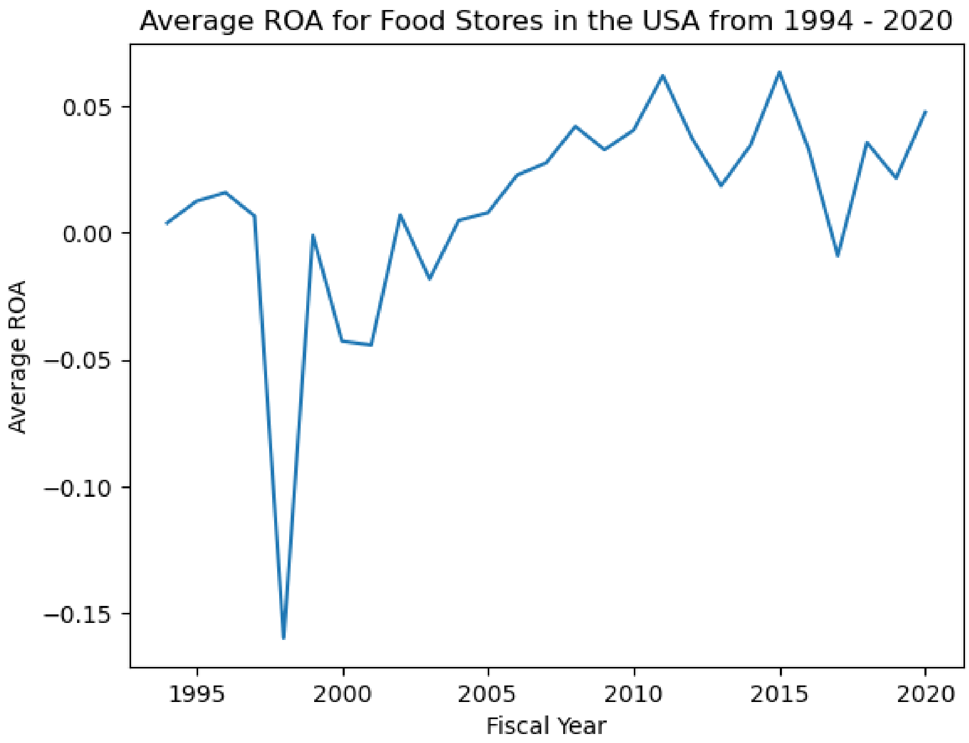


```
# merge y2007 and y2008 to include all firms with data in 2008
y2007_08 = pd.merge(y2007, y2008, on = 'conm', how = 'right', suffixes = ['_07', '_08'])
```

The difference between stock prices for each firm from 2007 to 2008 was calculated as a percentage of their 2007 stock price, and stored in a new column ‘prcc_c_drop’. Finally, the observation with the greatest ‘prcc_c_drop’ value was displayed. The firm that was affected the most by the 2008 Financial Crisis, as measured by the percentage drop in stock price from 2007 to 2008, was PENN TRAFFIC CO. Their stock price experienced a significant drop of approximately -94.81% during that period.

Average ROA for USA firms

To plot the average return on assets (ROA) of firms located in the USA across the years, we first filtered ‘food_data’ to retain observations associated with the USA (‘location’ == ‘USA’). This filtered data was stored in a new dataframe named ‘usa_data’. To compute the average ROA for USA firms across the years, we grouped ‘usa_data’ by ‘fyear’ and calculated the mean of the ‘roa’ column for each fiscal year group, storing this data in the dataframe ‘avg_usa_roa’. Finally, we generated a line plot to visualize the trend of the average ROA for firms in the USA from the year 1994 to 2020, with the fiscal year on the x-axis and average ROA on the y-axis, providing insights into the performance of these firms over this period.

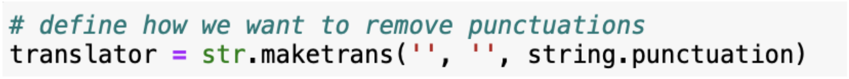


Part 2. Text Analysis on the Industry Sector

C. Text Cleaning

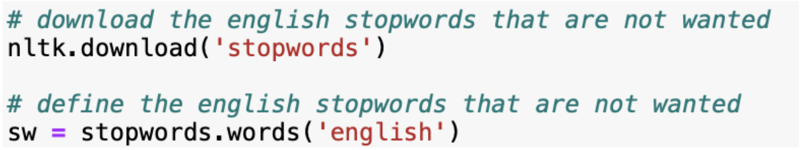
The file *data/2020_10K_item1_full.csv* contained a sample of 5,988 firms and the item 1 content of their 2020 10-K reports. The dataset was loaded as a pandas dataframe and named ‘report_data’. Duplicates were removed using *drop_duplicates* (NumFocus, Inc., 2023). We aimed to clean the text content in the column ‘item_1_text’. To do so, we defined the function *clean_text*, which takes a string as input and cleans it by converting it into lowercase, removing punctuations, and deleting stopwords.

We first defined how we wanted to remove punctuation using *string.punctuation*, which gives the following punctuation: !"#$%&'()*+, -./:;<=>?@[\]^_`{|}~



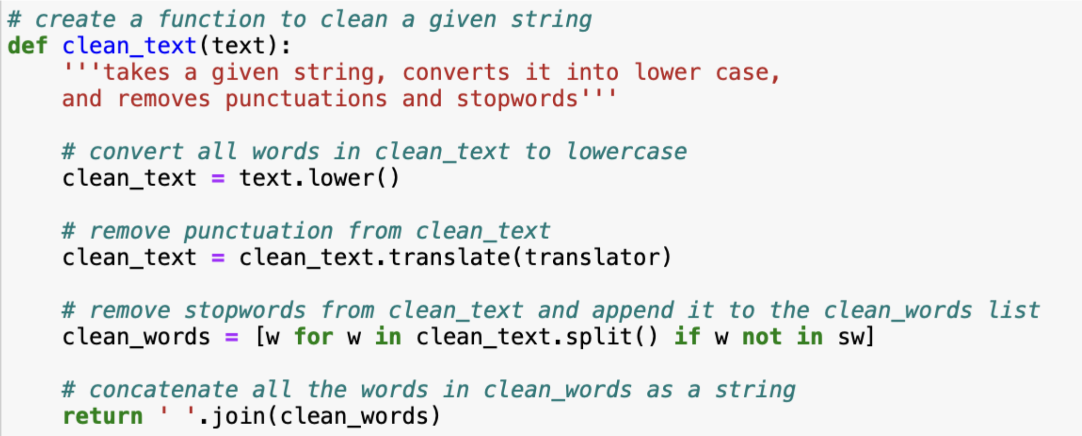
```
# define how we want to remove punctuations
translator = str.maketrans('', '', string.punctuation)
```

English stopwords were downloaded from NLTK, and stored in ‘sw’.



```
# download the english stopwords that are not wanted
nltk.download('stopwords')

# define the english stopwords that are not wanted
sw = stopwords.words('english')
```

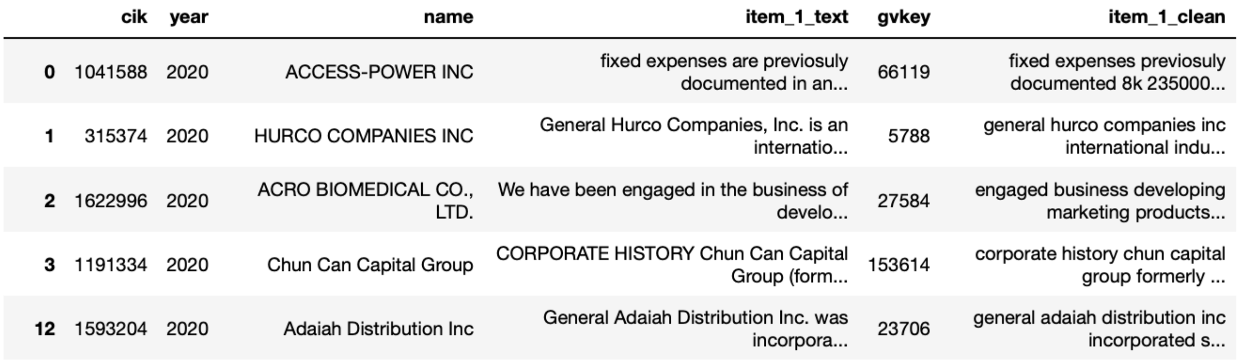
The code used to define *clean_text* is shown below:



```
# create a function to clean a given string
def clean_text(text):
    '''takes a given string, converts it into lower case,
    and removes punctuations and stopwords'''

    # convert all words in clean_text to lowercase
    clean_text = text.lower()

    # remove punctuation from clean_text
    clean_text = clean_text.translate(translator)

    # remove stopwords from clean_text and append it to the clean_words list
    clean_words = [w for w in clean_text.split() if w not in sw]

    # concatenate all the words in clean_words as a string
    return ' '.join(clean_words)
```

In the *clean_text* function, the string input 'text' is first converted into lowercase. The *translate* method is then applied to the lowercase text to remove punctuations. Next, each word in the split text is compared against the pre-defined list of stopwords ('sw'). If a word is not in the stopwords list, it is appended to the accumulator 'clean_words'. Finally, all the words in 'clean_words' are concatenated into a string.

The 'clean_text' function was applied to each row of 'item_1_text' in 'report_data' and the cleaned text for each observation was stored in the new column 'item_1_clean'.



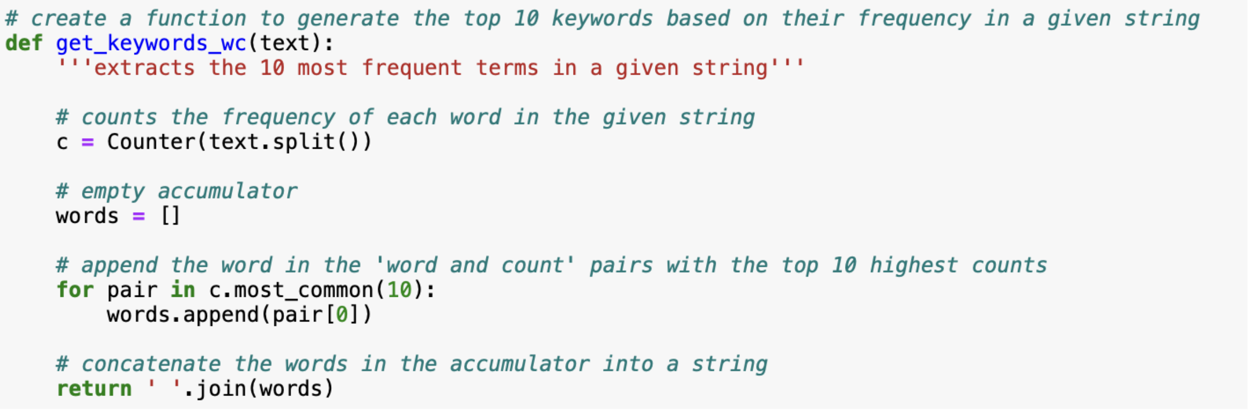
	cik	year	name	item_1_text	gvkey	item_1_clean
0	1041588	2020	ACCESS-POWER INC	fixed expenses are previosuly documented in an...	66119	fixed expenses previosuly documented 8k 235000...
1	315374	2020	HURCO COMPANIES INC	General Hurco Companies, Inc. is an internatio...	5788	general hurco companies inc international 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 ...
12	1593204	2020	Adaiah Distribution Inc	General Adaiah Distribution Inc. was incorpora...	23706	general adaiah distribution inc incorporated s...

D. Keyword Analysis

To conduct keywords analysis on the food stores, we aimed to create a new dataframe containing reports only by firms in the food stores sector. To do this, we merged 'report_data' and 'food_data' on the 'gvkey' column using an inner merge so that only observations with data in both datasets remained. This merged data was stored in a dataframe named 'food_reports'. Next, we filtered 'food_reports' for the fiscal year 2020. After merging and filtering the data, we noticed that out of the 104 food stores, only 9 had 2020 10-K reports available in the 'report_data' dataframe. To keep things tidy, we subset the 'food_reports' dataframe to include only the 'name', 'gvkey', and 'item_1_clean' columns. We were interested in generating the top 10 keywords for each firm using two methods: word counts and TF-IDF score.

Keywords by word count

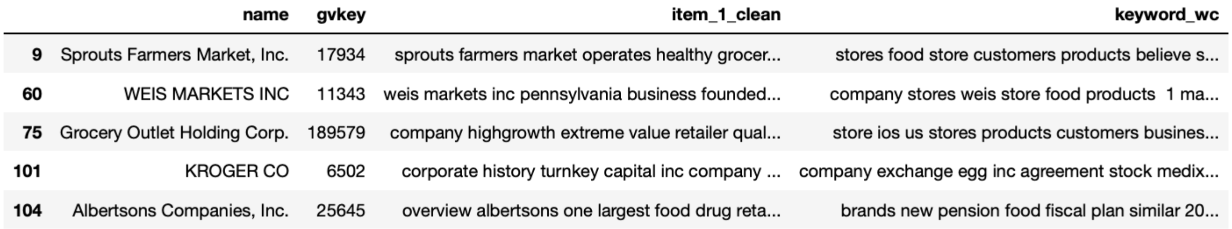
To find the keywords based on word count, we defined a function *get_keywords_wc*, which takes a string as input and extracts the 10 most frequent terms.



```
# create a function to generate the top 10 keywords based on their frequency in a given string
def get_keywords_wc(text):
    '''extracts the 10 most frequent terms in a given string'''

    # counts the frequency of each word in the given string
    c = Counter(text.split())

    # empty accumulator
    words = []

    # append the word in the 'word and count' pairs with the top 10 highest counts
    for pair in c.most_common(10):
        words.append(pair[0])

    # concatenate the words in the accumulator into a string
    return ' '.join(words)
```

In the above function, the frequency of each word in the given string (‘text’) is tabulated and paired with its word in ‘c’. The function then identifies the top 10 word and frequency pairs with the highest frequencies, and appends the first element of these 10 pairs (i.e., the word) into the accumulator ‘words’. Finally, all the words in the accumulator are concatenated into a single string.

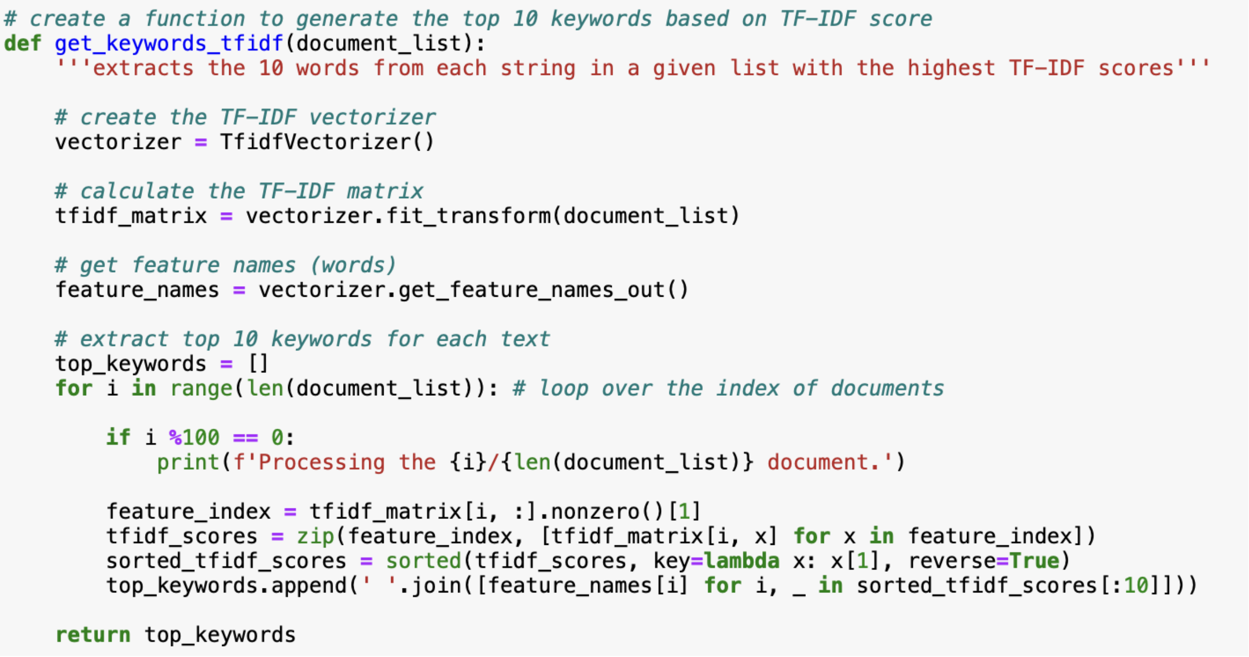
We applied *get_keywords_wc* to the text in the ‘item_1_clean’ column of the ‘food_reports’ dataframe. The generated keywords for each firm’s item 1 text were stored in a new column (‘keyword_wc’). Below we display the first 5 rows of the resulting dataframe.



	name	gvkey	item_1_clean	keyword_wc
9	Sprouts Farmers Market, Inc.	17934	sprouts farmers market operates healthy grocer...	stores food store customers products believe s...
60	WEIS MARKETS INC	11343	weis markets inc pennsylvania business founded...	company stores weis store food products 1 ma...
75	Grocery Outlet Holding Corp.	189579	company highgrowth extreme value retailer qual...	store ios us stores products customers busines...
101	KROGER CO	6502	corporate history turnkey capital inc company ...	company exchange egg inc agreement stock medix...
104	Albertsons Companies, Inc.	25645	overview albertsons one largest food drug reta...	brands new pension food fiscal plan similar 20...

Keywords based on TF-IDF score

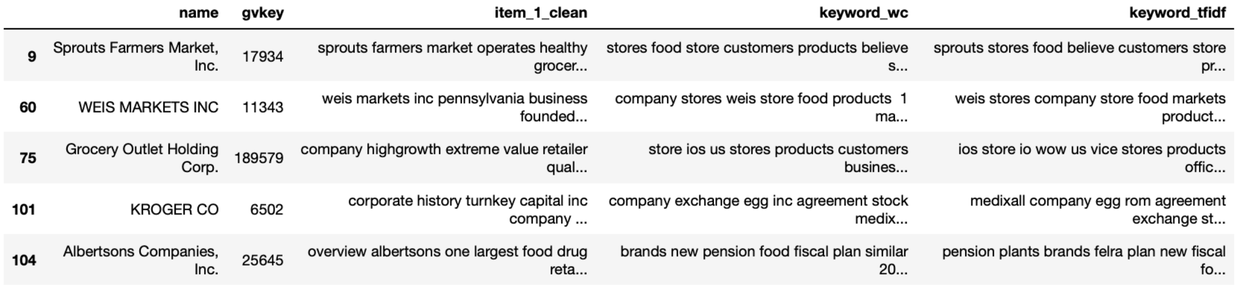
To find the keywords based on TF-IDF score, we defined a function *get_keywords_tfidf*, which takes a list of strings as input and extracts the 10 words from each string with the highest TF-IDF scores relative to the words in the other lists of strings.



```
# create a function to generate the top 10 keywords based on TF-IDF score
def get_keywords_tfidf(document_list):
    '''extracts the 10 words from each string in a given list with the highest TF-IDF scores'''

    # create the TF-IDF vectorizer
    vectorizer = TfidfVectorizer()

    # calculate the TF-IDF matrix
    tfidf_matrix = vectorizer.fit_transform(document_list)

    # get feature names (words)
    feature_names = vectorizer.get_feature_names_out()

    # extract top 10 keywords for each text
    top_keywords = []
    for i in range(len(document_list)): # loop over the index of documents

        if i %100 == 0:
            print(f'Processing the {i}/{len(document_list)} document.')

        feature_index = tfidf_matrix[i, :].nonzero()[1]
        tfidf_scores = zip(feature_index, [tfidf_matrix[i, x] for x in feature_index])
        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]]))

    return top_keywords
```

This function takes a list of strings (‘document_list’) as input and extracts the 10 words from each string with the highest TF-IDF scores relative to the words in the other lists of strings. Then, it returns a list of lists containing the extracted keywords.

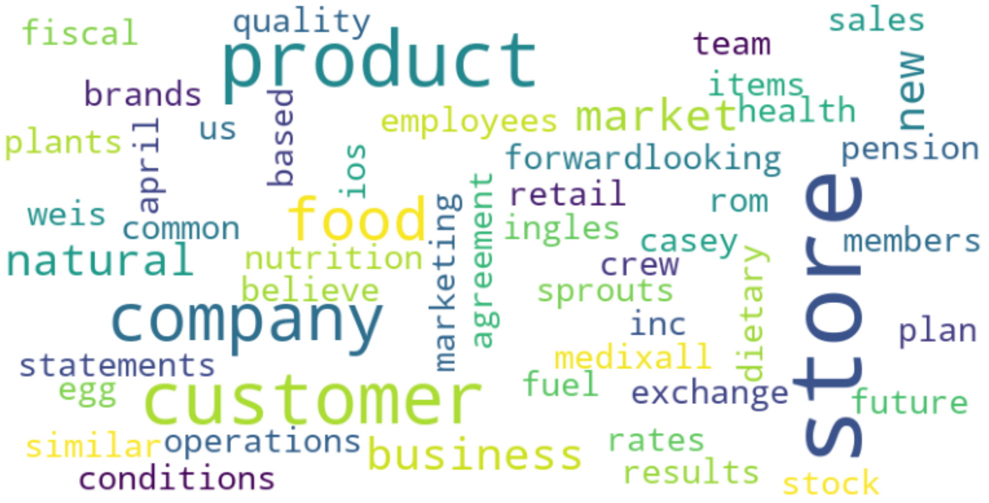
After converting the data in the ‘item_1_clean’ column of ‘food_reports’ into a list of strings, we applied *get_keywords_tfidf* to the data. The output was then stored in the new column ‘keyword_tfidf’. Below we display the first 5 rows of the resulting dataframe.



	name	gvkey	item_1_clean	keyword_wc	keyword_tfidf
9	Sprouts Farmers Market, Inc.	17934	sprouts farmers market operates healthy grocer...	stores food store customers products believe s...	sprouts stores food believe customers store pr...
60	WEIS MARKETS INC	11343	weis markets inc pennsylvania business founded...	company stores weis store food products 1 ma...	weis stores company store food markets product...
75	Grocery Outlet Holding Corp.	189579	company highgrowth extreme value retailer qual...	store ios us stores products customers busines...	ios store io wow us vice stores products offic...
101	KROGER CO	6502	corporate history turnkey capital inc company ...	company exchange egg inc agreement stock medix...	medixall company egg rom agreement exchange st...
104	Albertsons Companies, Inc.	25645	overview albertsons one largest food drug reta...	brands new pension food fiscal plan similar 20...	pension plants brands felra plan new fiscal fo...

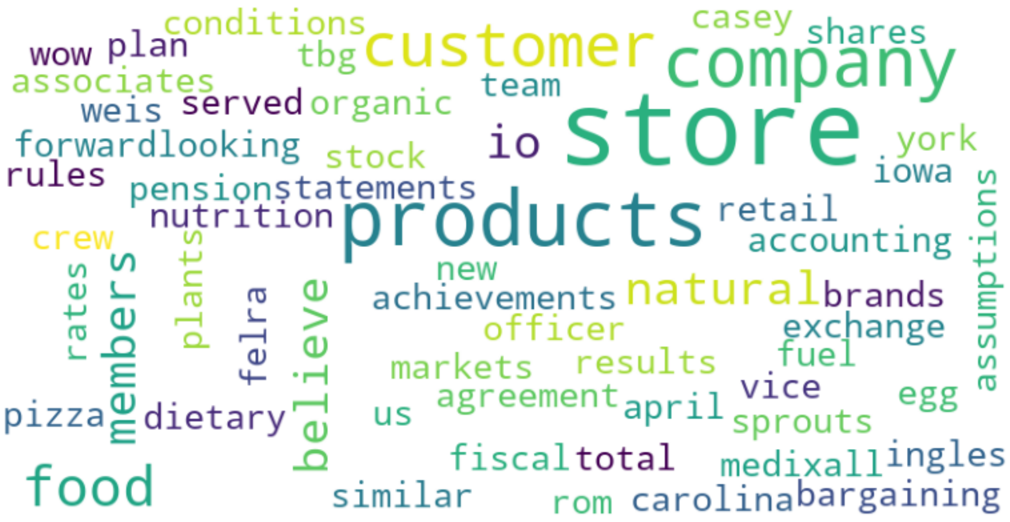
Creating a word cloud based on word count

To create a word cloud based on word count, we first concatenated the words in the 'keyword_wc' column from ‘food_reports’ into a list (‘text_wc’). We then used the *WordCloud* function to generate a word cloud visualization using content from ‘text_wc’. The visualization was configured with an 800-pixel width, 400-pixel height, a white background color, and a maximum font size of 80. To present this word cloud visually, we used a Matplotlib figure of 10 inches in width and 5 inches in height. For a cleaner appearance, we turned off the axis labels and ticks within the Matplotlib plot. Below is the word cloud visualization of the top 10 keywords across all firms in the food stores sector.



Creating a word cloud based on TF-IDF scores

Similarly, we created a word cloud based on TF-IDF scores by utilizing the 'keyword_tfidf' column in our dataset, following the same process as before. Below is the word cloud visualization of the top 10 keywords based on TF-IDF scores across all firms in the food stores sector.



E. Word Embedding

A word2vec model was used to analyze the cleaned item 1 text of the food stores. First, each report's 'item_1_clean' data in the full 'report_data' dataframe was split into a list of words, and the data across all firms was stored in a list of lists ('sent'). The word2vec model was trained using 'sent', saved, then loaded.

We identified 3 words from the TF-IDF wordcloud generated in Part D that were representative of the food stores sector: ‘nutrition’, ‘food’, and ‘markets’. Then, we used the trained word2vec model to find the most relevant 5 words for each of the chosen keywords.

For 'nutrition,' the most relevant words were:
1. Nutritional
2. Wellness
3. Nutraceuticals
4. Beverage
5. Beauty

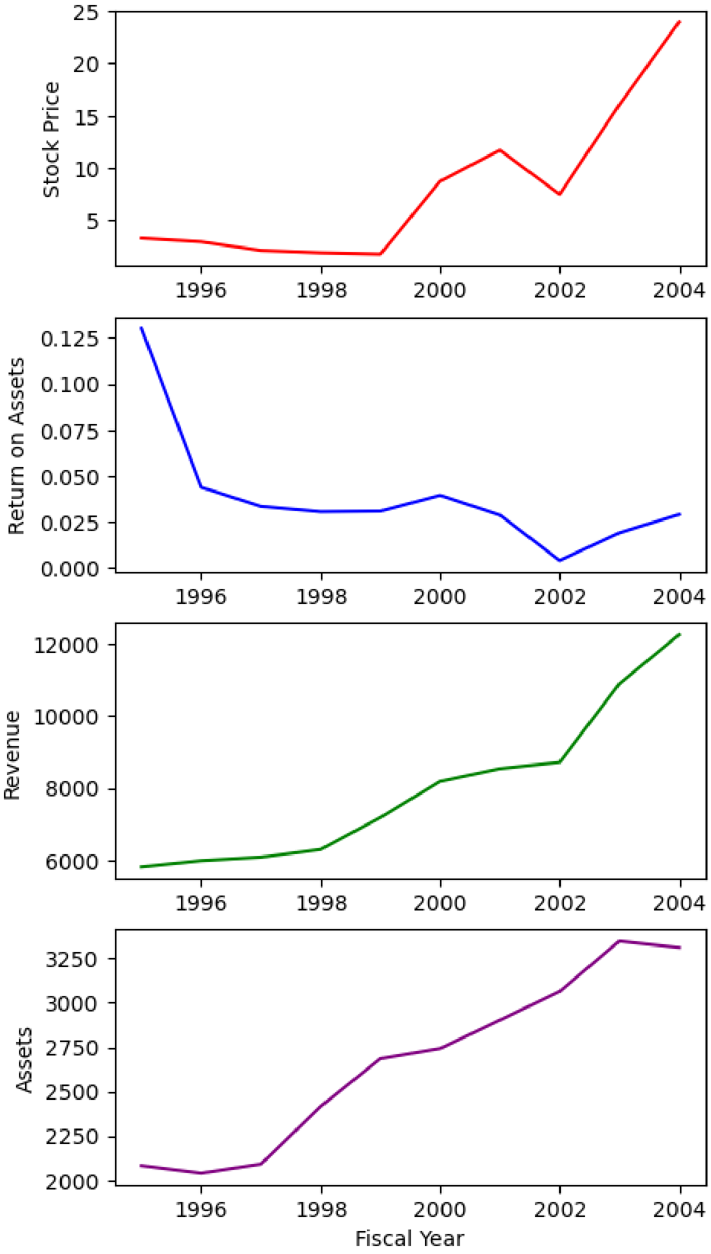
For 'food,' the most relevant words were:
1. Beverage
2. Meat
3. Cosmetic
4. Perishable
5. Fruitbased

For 'markets,' the most relevant words were:
1. Market
2. Endmarkets
3. Subsegments
4. Sectors
5. Geographies

Part 3. Comprehensive Analysis of One Sample Firm

F. Firm Analysis and Strategy Suggestion

As 7-ELEVEN is such a prevalent and familiar store to most of us, we decided to conduct a comprehensive analysis of its market status with the goal of providing one suggestion to the firm based on our results. We visualized the historical stock prices, ROA, revenue, and assets of the company. The code to generate this plot was obtained from Data Camp’s ‘Introduction to Data Visualization with Matplotlib’ course (Rokem, n.d.).

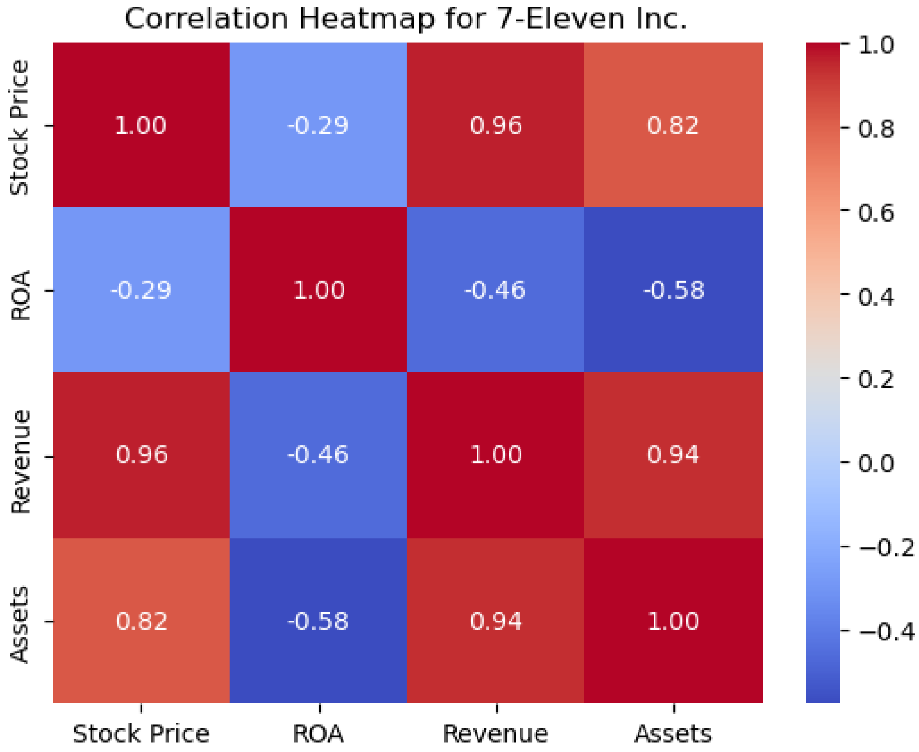


Based on the above plot, it is evident that the year 2002 marked a significant turning point for 7-ELEVEN INC. From 2001 to 2002, there was a noticeable decline in key performance indicators, including stock prices, ROA, and revenue. This downturn can largely be attributed to the economic recession, which had a widespread impact on businesses across various sectors.

However, what makes the year 2002 particularly noteworthy is 7-ELEVEN INC's strategic response to these challenges. In this crucial year, they executed a highly effective marketing strategy by introducing a national free Slurpee day as part of their 75th-anniversary celebration. This innovative initiative garnered substantial media attention and created significant buzz not only in North America but also in Australia. As a result, 7-ELEVEN INC gave away a staggering 500,000 gallons of Slurpees in North America and

270,000 Slurpees in Australia (Lammle, 2018). This promotion not only generated a surge in customer interest but also contributed substantially to revenue growth and an uptick in stock prices.

After plotting the graphs for the company's stock prices, ROA, revenue, and assets, we decided to do a correlation analysis to understand how these variables are related. We constructed a correlation heatmap based on code obtained from Geeks for Geeks (Vanshikagoyal43, 2020). The heatmap below displays the strength and direction of these correlations, with warmer red colors indicating stronger positive relationships and cooler blue representing negative or weaker correlations. The values indicate correlation coefficients.



Analyzing the heatmap, we observed positive correlations between stock price and revenue, stock price and assets, as well as revenue and assets. Conversely, we noticed negative correlations between ROA and stock price, as well as ROA and revenue.

Given this analysis, **our valuable suggestion to 7-ELEVEN INC is to recognize the potential of staying relevant in the media landscape**. To achieve this, we recommend continued investment in marketing and customer engagement strategies. For instance, the company could consider implementing customer loyalty programs to incentivize regular customers and explore the creation of new traditions or events that can generate media attention. By doing so, 7-ELEVEN INC can reinforce its brand presence in the minds of consumers, sustain growth, and navigate future challenges more effectively, ultimately leading to potential boosts in revenue and stock prices, both of which have been shown to have a positive impact on the company’s return on assets.

References

Lammle, R. (2018, July 18). *11 facts about 7-Eleven.* Mental Floss. Retrieved September, 2023. https://www.mentalfloss.com/article/51629/11-facts-about-7-eleven-711

NumFocus, Inc. (2023). *pandas.DataFrame.drop_duplicates.* pandas. Retrieved September, 2023. https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.drop_duplicates.html

Parewa Labs Pvt. Ltd. (n.d.) *Python String startswith().* Programiz. Retrieved September, 2023. https://www.programiz.com/python-programming/methods/string/startswith

Rokem, Ariel. (n.d.). *Introduction to Data Visualization with Matplotlib.* Data Camp. https://app.datacamp.com/learn/courses/introduction-to-data-visualization-with-matplotlib

Vanshikagoyal43. (2020, November 12). *How to create a seaborn correlation heatmap in Python?* Geeks for Geeks. Retrieved September, 2023. https://www.geeksforgeeks.org/how-to-create-a-seaborn-correlation-heatmap-in-python/

Acknowledgements

The authors would like to thank Gene Moo Lee, Jaecheol Park, and Xiaoke Zhang for their valuable guidance and support. Much of the code in this project was obtained from the UBC Master of Business Analytics BAIT 508 course.

Packages Used

- pandas
- matplotlib
- nltk
- string
- stopwords from nltk.corpus
- Counter from collections
- TfidVectorizer from sklearn.feature_extraction.text
- WordCloud from wordcloud
- Word2Vec from gensim.models
- Seaborn