BAIT 508 BA1 Group 8
Industry Analysis

Group Member:

Jiayi Tao
ID: 97892764
Role: Report writing & analysis (Part 3)

Wanlin (Cara) Zhou
ID: 37079019
Role: Codes writing & analysis (Part 1, Part 3) & report writing

Yuehan Wang
ID: 21233218
Role: Codes writing & analysis (Part 2, Part 3)

Part 1. Quantitative Analysis of the Industry Sector

The project will utilize three datasets:

- major_groups.csv
- 2020_10K_item1_full.csv
- public_firms.csv

To start with, we imported the data sets as follows:

```
#import dataa
os.listdir("data")
K10 = pd.read_csv(r'data/2020_10K_item1_full.csv')
m_g = pd.read_csv(r'data/major_groups.csv')
p_f = pd.read_csv(r'data/public_firms.csv')
```

A. Industry Sector Selection and Data Filtering.

This report aims to provide an in-depth analysis of GAP Inc (gvkey: 4990) in the apparel and accessory sector (sic starting with "56"). To start with, we created the column 'major_group' in the DataFrame "public_firms". Then we merged DataFrames "m_g" and "p_f" by inner-join with 'major_group', and filtered based on major group number (56 for the apparel sector) to retain data of this industry. To make it clearer, we included the 'description' column from major_groups data, notifying the companies are in the Apparel and Accessory Stores sector.

In [4]:
```
p_f['major_group'] = p_f['sic'].astype(str).str[:2]

#convert major_group in m_g to str so that we can merge m_g with p_f
m_g['major_group'] = m_g['major_group'].astype(str)
newp_f = pd.merge(p_f, m_g, on = 'major_group', how = 'left')
newp_f.head()
```

Out[4]:

	gvkey	fyear	location	conm	ipodate	sic	prcc_c	ch	ni	asset	sale	roa	major_group	description
0	1004	1994	USA	AAR CORP	1988/01/01	5080	13.375	22.487	10.463	425.814	451.395	0.024572	50	Wholesale Trade-durable Goods
1	1004	1995	USA	AAR CORP	1988/01/01	5080	22.000	33.606	16.012	437.846	504.990	0.036570	50	Wholesale Trade-durable Goods
2	1004	1996	USA	AAR CORP	1988/01/01	5080	30.250	51.705	23.025	529.584	589.328	0.043478	50	Wholesale Trade-durable Goods
3	1004	1997	USA	AAR CORP	1988/01/01	5080	38.750	17.222	35.657	670.559	782.123	0.053175	50	Wholesale Trade-durable Goods
4	1004	1998	USA	AAR CORP	1988/01/01	5080	23.875	8.250	41.671	726.630	918.036	0.057348	50	Wholesale Trade-durable Goods

In [6]:
```
df = newp_f[newp_f['major_group']== '56']
df.head()
```

Out[6]:

	gvkey	fyear	location	conm	ipodate	sic	prcc_c	ch	ni	asset	sale	roa	major_group	description
5923	2484	1995	USA	BURLINGTON COAT FACTORY INVS	NaN	5651	10.2500	14.520	14.866	735.269	1584.942	0.020218	56	Apparel And Accessory Stores
5924	2484	1996	USA	BURLINGTON COAT FACTORY INVS	NaN	5651	13.0000	73.560	29.013	704.731	1591.964	0.041169	56	Apparel And Accessory Stores
5925	2484	1997	USA	BURLINGTON COAT FACTORY INVS	NaN	5651	16.4370	157.394	56.515	775.077	1758.368	0.072915	56	Apparel And Accessory Stores
5926	2484	1998	USA	BURLINGTON COAT FACTORY INVS	NaN	5651	16.3125	106.952	47.783	941.635	1988.513	0.050745	56	Apparel And Accessory Stores
5927	2484	1999	USA	BURLINGTON COAT FACTORY INVS	NaN	5651	13.8750	127.818	61.120	1046.047	2198.696	0.058429	56	Apparel And Accessory Stores

There are 27 unique firm-year observations, and 105 unique firms in the filtered dataset. They are found using the len(df.unique()). By finding the firms who have “1994” and “2020” in column “fyear” and inner joining them, we found that there are 11 firms in the industry that have records over all 27 years.

```
a = len(df['fyear'].unique())

print('3.1: There are '+ str(a) + ' unique firm-year observations in the filtered dataset.')
```

```
3.1: There are 27 unique firm-year observations in the filtered dataset.
```

```
b = len(df['gvkey'].unique())
print('3.2: There are '+ str(b)+' unique firms in the filtered dataset.')
```

```
3.2: There are 105 unique firms in the filtered dataset.
```

```
h94 = df[df['fyear'] == 1994]
h20 = df[df['fyear'] == 2020]
ttl27 = pd.merge(h94, h20, on = 'gvkey', how = 'inner')
c = len(ttl27['gvkey'].unique())

print('3.3: There are '+ str(c)+' firms in the filtered dataset have records over all 27 years.')
```

```
3.3: There are 11 firms in the filtered dataset have records over all 27 years.
```

B. Preliminary Analysis

1. Top 10 firms with highest stock prices in 2020

In the year 2020, the top 10 firms with the highest stock prices are found by filtering the data of 2020, and sorting the firms based on the descending orders of stock prices using the df.sort_values (dataset, ascending=False). Then, we reset the index of the data frame, dropping the old index and shifting the index by 1 to start from 1 instead of 0. For clarity, we renamed the index to 'Rank'.

```
#select the top 10 firms from the dataframe
top10_prcc = df[df['fyear']==2020].sort_values('prcc_c', ascending=False)[['conm','prcc_c']][0:10]
```

```
top10_prcc.reset_index(drop=True, inplace=True)
top10_prcc.index += 1
top10_prcc.rename_axis('Rank', inplace=True)
top10_prcc
```

Rank	conm	prcc_c
1	BURLINGTON STORES INC	261.55
2	ROSS STORES INC	122.81
3	TJX COS INC (THE)	68.29
4	CHILDRENS PLACE INC	50.10
5	CITI TRENDS INC	49.68
6	BOOT BARN HOLDINGS INC	43.36
7	FOOT LOCKER INC	40.44
8	SHOE CARNIVAL INC	39.18
9	BATH & BODY WORKS INC	37.19
10	ZUMIEZ INC	36.78

2. Top 10 companies in sales in the history

To find sales of each firm, we first grouped “df” by the column 'conm' and calculated the sum of 'sale' for each group. Then, we sorted the grouped data in descending order based on the total 'sale'. Lastly, we select the top 10 rows based on the 'sale' column. The resulting 'filter_sale' Series contains the total sales values for the top 10 companies.

```
filter_sale = df.groupby('conm').sum('sale').sort_values('sale',ascending=False)[0:10]['sale']
filter_sale
```

```
conm
TJX COS INC (THE)            531354.915
GAP INC                      362527.300
BATH & BODY WORKS INC        274942.175
NORDSTROM INC                248159.506
ROSS STORES INC              188529.105
FOOT LOCKER INC              167706.000
ABERCROMBIE & FITCH  -CL A    67874.646
ASCENA RETAIL GROUP INC       65366.513
AMERN EAGLE OUTFITTERS INC    63138.850
DESIGNER BRANDS INC           57096.129
Name: sale, dtype: float64
```

3. The geographical distribution of all the firms

The geographical distribution of all the firms is shown below. We selected 'conm' and 'location' columns from the DataFrame “df”, removing duplicate rows. The resulting DataFrame “c_l” contains unique combinations of company names ('conm') and corresponding locations ('location').

	conm	location
5923	BURLINGTON COAT FACTORY INVS	USA
6468	CACHE INC	USA
7076	CATO CORP -CL A	USA
7564	CHARMING SHOPPES INC	USA
8310	CLAIRES STORES INC	USA
...	...	...
199846	EXPRESS INC	USA
200847	G-ESTATE LIQUIDATION STORES	USA
200914	BODY CENTRAL CORP	USA
203001	FHC HOLDINGS CORP	USA
203501	TILLY'S INC	USA

104 of them are in US and 1 of them is in Canada. The top 10 locations are just USA and Canada. The numbers are calculated using the value_counts based on location column.

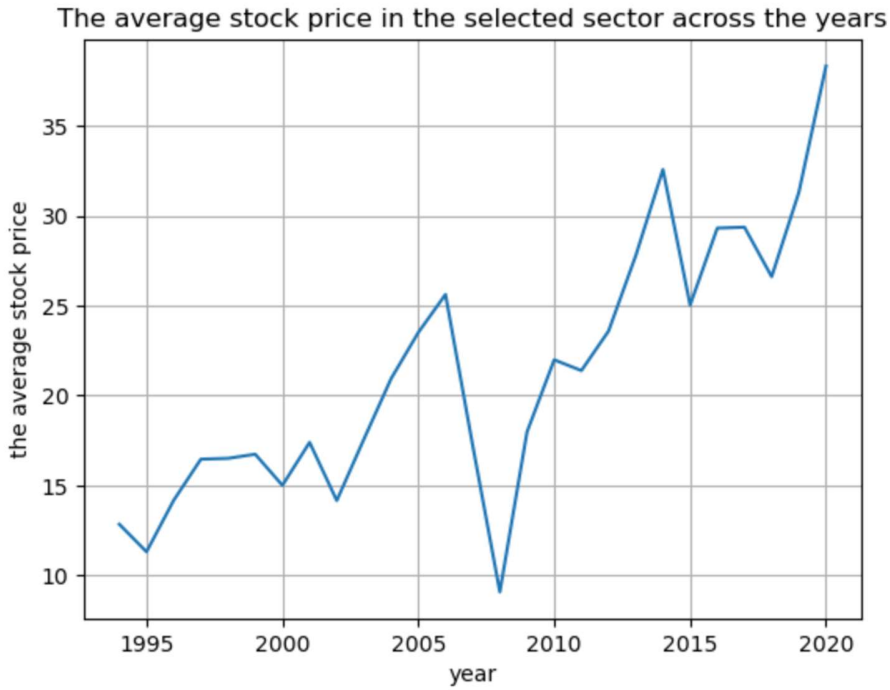
```
top10_location = c_l.value_counts('location')[0:10]
top10_location
```

```
location
USA    104
CAN      1
dtype: int64
```

4. Line chart to show the average stock price in the selected sector across the years

We firstly grouped ‘df' by the 'fyear' (financial year) column and calculated the mean of the 'prcc_c' (closing stock price) for each year. We stored the resulting Dataframe “ave_stock_aprice”, which contains average stock prices per year. Then, we plotted the average stock prices across the years using a line plot. To make it clearer, we added title to the plot as “The average stock price in the selected sector across year”, and set the labels of x,y axis. We also enabled grid lines. At last, we used plt.show() to present the plot.

```
ave_stock_price = df.groupby(['fyear'])[['prcc_c']].mean()
plt.plot(ave_stock_price)
plt.title('The average stock price in the selected sector across the years')
plt.xlabel('year')
plt.ylabel('the average stock price')
plt.grid(True)
plt.show()
```



5. The firm which was affected the most by the 2008 Financial Crisis:

Firstly, we found the companies that exist during 2007 to 2008 using df.isin().

```
#Codes for calculating percentage changes are inspired by ChatGPT 3.5
filtered_df = df[df['fyear'].isin([2007,2008])]
filtered_df
```

Then we add a new column 'yearlychange' to "filtered_df" and calculate the percentage change in 'prcc_c' column for each company grouped by 'conm'. Then, we sorted the affected firm's Dataframe by 'yearlychange' column in ascending order and selected the first row using df.iloc[] to get the most negatively affected firm

EDDIE BAUER HOLDINGS INC is the firm that was affected the most by the 2008 Financial Crisis, as measured by the percentage drop (-91.97%) in stock price from 2007 to 2008.

```
In [17]: affacted_firm = filtered_df.sort_values('yearlychange',ascending = True).iloc[0,3]
         print(f'{affacted_firm} is the firm that was affected the most by the 2008 Financial Crisis, as measured by the percentage drop in stock pr

         EDDIE BAUER HOLDINGS INC is the firm that was affected the most by the 2008 Financial Crisis, as measured by the percentage drop in stoc
         k price from 2007 to 2008

In [18]: filtered_df[filtered_df['conm'] == affacted_firm]
```

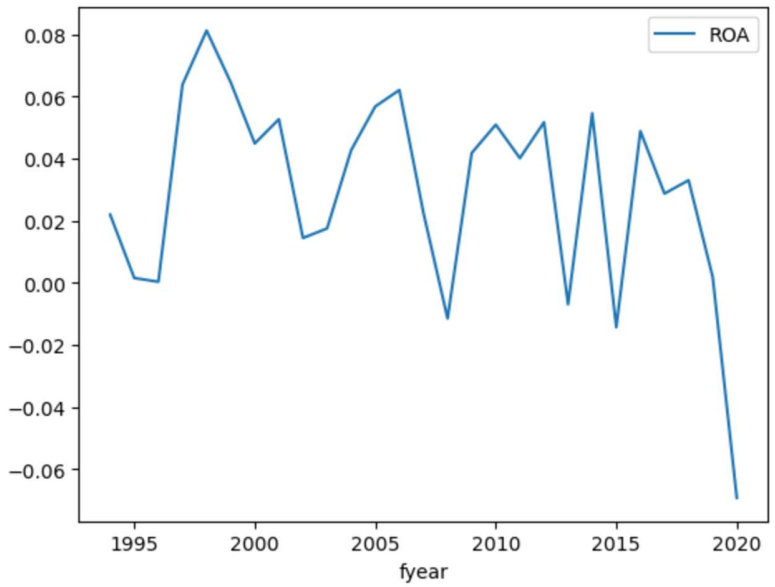
Out[18]:

	gvkey	fyear	location	conm	ipodate	sic	prcc_c	ch	ni	asset	sale	roa	major_group	description	yearlychange
185813	164058	2007	USA	EDDIE BAUER HOLDINGS INC	NaN	5600	6.35	27.596	-101.718	811.432	1044.353	-0.125356	56	Apparel And Accessory Stores	NaN
185814	164058	2008	USA	EDDIE BAUER HOLDINGS INC	NaN	5600	0.51	60.425	-165.529	596.920	1023.437	-0.277305	56	Apparel And Accessory Stores	-91.968504

6. **Plot the average Return on Assets (ROA) for the firms located in the "USA" across the years. ROA is calculated as ni/asset.**

We calculated the ROA first, then found the means of ROA of USA firms for each year.

```
df['ROA'] = df['ni']/df['asset']
dfROA = df[df['location']=='USA'].groupby('fyear').mean('ROA')
dfROA= dfROA.reset_index()
dfROA.plot(x='fyear', y='ROA')
```



Part 2. Text Analysis on the Industry Sector

C. Text Cleaning

The file "data/2020_10K_item1_full.csv" contains a sample of 5,988 firms and their "item 1" content in their 10-K reports in the year 2020.1 Load the dataset as a DataFrame and create a new column containing the cleaned text for each "item1" content.

1. Convert all words to lowercase.

To start with, words are converted to lowercase uising str.lower() function.

2. Remove punctuations.

Then, we removed the punctuations using replace function:

```
#2.Remove punctuations.
import string
K10['item1'] = K10['item1'].str.replace('[{}]'.format(string.punctuation), '')
K10['item1']
```

3. Remove stop words based on the list of English stop words in NLTK.

We imported the Natural Language Toolkit (nltk) library and downloaded the stopwords corpus. We got the set of English stopwords from nltk.corpus. Then we used the lambda function to remove stopwords from 'item1' column in K10.

```
K10['item1'] = K10['item1'].apply(lambda x: ' '.join([word for word in x.split() if word not in stop_words]))
K10['item1']
```

D. Keyword Analysis:

1. Create a new DataFrame that includes only firms in your selected industry sectors. Ensure that you merge the 10-K data with the previous "public_firm.csv" data using an inner join.

We created "tgt", a merged DataFrame of "df" and "K10" based on 'gvkey'.

```
tgt = pd.merge(df, K10, on = 'gvkey', how = 'inner')
tgt[['gvkey','fyear','conm','sale','item1']].head()
```

	gvkey	fyear	conm	sale	item1
0	2818	1994	CATO CORP -CL A	476.186	general company founded 1946 operated 1281 fas...
1	2818	1995	CATO CORP -CL A	489.995	general company founded 1946 operated 1281 fas...
2	2818	1996	CATO CORP -CL A	491.509	general company founded 1946 operated 1281 fas...
3	2818	1997	CATO CORP -CL A	512.448	general company founded 1946 operated 1281 fas...
4	2818	1998	CATO CORP -CL A	543.664	general company founded 1946 operated 1281 fas...

2. **Generate the top 10 keywords for each firm based on two different methods: word counts and TF-IDF score.**

Firstly, we defined the function get_keywords_wc to extract top 10 keywords using word counts from the given text using a for-loop.

Then, we defined the function get_keywords_tfidf to extract top 10 keywords using TF-IDF scores from the given list of documents.

```
def get_keywords_wc(text):
    c = Counter(text.split())
    words = []
    for pair in c.most_common(10):
        words.append(pair[0])
    return ' '.join(words)

def get_keywords_tfidf(document_list):
    '''
    Input: A list of documents (text)
    Output: The corresponding top 10 keywords for each document based on tf-idf values
    '''

    # Step 1: Create the TF-IDF vectorizer
    vectorizer = TfidfVectorizer()

    # Step 2: Calculate the TF-IDF matrix
    tfidf_matrix = vectorizer.fit_transform(document_list)

    # Step 3: Get feature names (words)
    feature_names = vectorizer.get_feature_names_out()

    # Step 4: Extract top 10 keywords for each text
    top_keywords = []
    for i in range(len(document_list)):

        if i %50 == 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]]))
```

We calculated top keywords by applying get_key_words_wc() to the 'item1' column and the top keywords using the word count method. We stored the calculated keywords in column 'keywords1'

The function get_keywords_tfidf() computes the top 10 keywords based on TF-IDF scores. Lastly, we assigned the calculated keywords to a new column 'keyword_clean_tfidf'.

```
In [38]: tgt[['conm','keywords1','keyword_clean_tfidf']].head()
```

Out[38]:

	conm	keywords1	keyword_clean_tfidf
0	CATO CORP -CL A	company merchandise stores sales credit store ...	company merchandise layaway stores credit sale...
1	CATO CORP -CL A	company merchandise stores sales credit store ...	company merchandise layaway stores credit sale...
2	CATO CORP -CL A	company merchandise stores sales credit store ...	company merchandise layaway stores credit sale...
3	CATO CORP -CL A	company merchandise stores sales credit store ...	company merchandise layaway stores credit sale...
4	CATO CORP -CL A	company merchandise stores sales credit store ...	company merchandise layaway stores credit sale...

3. Create two wordclouds to visualize the keywords across all firms in the selected sector: one based on the word count of keywords and another based on the TF-IDF score.

We created two wordclouds to visualize the keywords across all firms in the selected sector: one based on the word count of keywords, and another based on the TF-IDF score.

To visualize the keywords, we first prepared the text by merging the key words generated by the two methods.

```
# prepare text
forcould = tgt[['conm','keywords1','keyword_clean_tfidf']].drop_duplicates()

kwbycnt = ' '.join(forcould['keywords1'].tolist())
kwbytf = ' '.join(forcould['keyword_clean_tfidf'].tolist())
```

The list variable kwbycnt contains keywords data that is used to create the word cloud. The final word cloud is stored as a PNG file with the name "keyword_wc.png." The size of the words in this picture indicates how frequently they occur in the text. Some of the words are that frequently used in the 10-K reports in this industry are “store”, “merchandise”, and “customer”.

```
# lower max_font_size
wordcloud1 = WordCloud(width=800, height=400, max_font_size= 100, background_color='white').generate(kwbycnt)
```

The list variable kwbytf contains keywords data that is used to create the word cloud. The generated word cloud is stored as a PNG file with the name 'keyword_tfidf.png'. Manually compared with the one generated with keywords by word count, the top 3 frequent words here are basically the same.

```
# lower max_font_size
wordcloud2 = WordCloud(width=800, height=400, background_color='white').generate(kwbytf)
```

E. **Word embedding**

1. Train a word2vec model with the full 10-K sample (e.g., "data/2020_10K_item1_full.csv").

We processed the text data, trained a Word2Vec model using the provided parameters, and saved the trained model for later use.

```
# training word2vec model using the list of words in `sent`
model = Word2Vec(sent, min_count=1, vector_size=50, \
                 workers=3, window=3, sg = 1)
model.save("word2vec.model")
```

2. Manually inspect the word clouds you generated in D.3 and choose three representative keywords that are relevant to the industry sector of your interest. Utilize the trained word2vec model to find the most relevant five words for each of these three keywords.

We used the Word2Vec model to find the most similar words for each of the keywords “store”, “customer”, and “merchandise”, which are the representative keywords we selected from the word clouds generated previously. We printed the top 5 most relevant words for each keywords with a for-loop.

The top 5 most relevant words of keyword “store” are “stores”, “restaurant”, “showroom”, “bsg”, “restaurants”.

The top 5 most relevant words of keyword “customer” are “client”, “endcustomer”, “enduser”, “customers”, “vendor”.

The top 5 most relevant words of keyword “merchandise” are “inseason”, “assortment”, “instore”, “assortments”, “retailer”.

```
wc1 = ['store', 'customer','merchandise']
for i in wc1:
    print(model.wv.most_similar(i)[:5],'\n')
```

```
[('stores', 0.8809576034545898), ('restaurant', 0.8649138808250427), ('showroom', 0.854566216468811), ('bsg', 0.84544
90303993225), ('restaurants', 0.8427577018737793)]

[('client', 0.9410847425460815), ('endcustomer', 0.9170929193496704), ('enduser', 0.8756552338600159), ('customers',
0.86241614818573), ('•vendor', 0.8473394513130188)]

[('inseason', 0.8866264820098877), ('assortment', 0.8430270552635193), ('instore', 0.8329434394836426), ('assortment
s', 0.8316181898117065), ('retailer', 0.8265424370765686)]
```

Part 3. Comprehensive Analysis of One Sample Firm

3. **[Firm Analysis and Strategy Suggestion; 10 points] This is an open question. Pick <u>one</u> firm that you are interested in and try to analyze its market status. The ultimate goal is to provide <u>one valuable suggestion</u> to the firm based on your analysis. Some directions you might consider are, but not limited to:**

   1. **Convert the keywords extracted in D.2 into word embeddings with the word2vec model trained in E.1. Add up the embeddings for each firm to create the firm-level embeddings. Use the firm-level embeddings to find the focal firm's competing firms (or, most similar firms).**
   2. **Compare the revenue, market share, and ROA of the focal firm to its competitors and provide suggestions accordingly.**
   3. **Perform an analysis of the historical stock prices, ROA, revenue, and assets of the chosen company. Investigate potential correlations and address noteworthy decreases and increases.**

1. We used the firm-level embeddings to find the focal firm's competing firms (or, most similar firms). We imported the DocumentSimilarity module, then used a DataFrame "tgta" that had unique TF-IDF keywords for each firm (instead of repeated by different 'fyear's) to train the model.

```
# Load the wrapper the instructor team prepared for you
from DocumentSimilarity import DocumentSimilarity
tgta = tgt[['gvkey','conm','keyword_clean_tfidf']].drop_duplicates()
# Create an instance
d = DocumentSimilarity(model = model, gvkeys=tgta['gvkey'], \
                       conm = tgta['conm'],
                       keywordslist = tgta['keyword_clean_tfidf'])
```

We computed the top 20 most similar firm with the function model.most_similar(), and we selected 'AMERN EAGLE OUTFITTERS INC' and 'ABERCROMBIE & FITCH -CL A' as the competitors.

```
d.most_similar(firm = 4990, topn = 20)
```

```
[(30059, 'AMERN EAGLE OUTFITTERS INC', 0.91699326),
 (63643, 'ABERCROMBIE & FITCH  -CL A', 0.91054213),
 (25234, 'BUCKLE INC', 0.89101213),
 (27938, 'SHOE CARNIVAL INC', 0.8857952),
 (5109, 'GENESCO INC', 0.82513076),
 (25167, 'TAILORED BRANDS INC', 0.81526124),
 (25108, 'CHRISTOPHER & BANKS CORP', 0.8058994),
 (6733, 'BATH & BODY WORKS INC', 0.79280674),
 (27981, 'CHICOS FAS INC', 0.7833563),
 (184323, 'EXPRESS INC', 0.7730822),
 (25186, 'STEIN MART INC', 0.763908),
 (187041, 'FHC HOLDINGS CORP', 0.7616397),
 (163601, 'DSW INC-OLD', 0.75947845),
 (24171, 'DESIGNER BRANDS INC', 0.75947845),
 (18675, 'BURLINGTON STORES INC', 0.75481117),
 (2818, 'CATO CORP  -CL A', 0.7488928),
 (65430, 'CHILDRENS PLACE INC', 0.7458205),
 (29150, 'URBAN OUTFITTERS INC', 0.7428187),
 (63874, 'STAGE STORES INC', 0.73431695),
 (9248, 'ROSS STORES INC', 0.72437686)]
```

We generated a DataFrame "compet" to store competitive firms' data.

	gvkey	fyear	location	conm	ipodate	sic	prcc_c	ch	ni	asset	...	description	ROA
53	4990	2020	USA	GAP INC	NaN	5651	20.19	1988.000	-665.000	13769.000	...	Apparel And Accessory Stores	-0.048297
469	30059	2020	USA	AMERN EAGLE OUTFITTERS INC	1994/04/13	5600	20.07	850.477	-209.274	3434.806	...	Apparel And Accessory Stores	-0.060927
502	63643	2020	USA	ABERCROMBIE & FITCH -CL A	1996/09/25	5651	20.36	1104.862	-114.021	3314.902	...	Apparel And Accessory Stores	-0.034396

2. Based on the 'sale' column, we computed the market share for each company in the “tgt” by calculating the sales of each firm's divided by total market sales in 2020.

   We generated a DataFrame “compet1” to compare revenue, market share, and ROA. To calculate the market share, we first calculated the total market sales by summing the sales of each firm in the selected section. Then we calculated the market share by dividing each firm’s sales with total market share in 2020.

```
compet1=compet[['gvkey','conm','prcc_c','asset']]
#Compare the revenue
compet1['revenue']=compet['sale']
#Compare the market_share
total_market_sale = tgt[tgt['fyear']==2020]['sale'].sum()
compet1['market_share']=compet['sale']/total_market_sale
#Compare the ROA
compet1['ROA']=compet['roa']

compet1.reset_index(drop=True, inplace=True)
compet1
```

	gvkey	conm	prcc_c	asset	revenue	market_share	ROA
0	4990	GAP INC	20.19	13769.000	13800.000	0.158596	-0.048297
1	30059	AMERN EAGLE OUTFITTERS INC	20.07	3434.806	3759.113	0.043201	-0.060927
2	63643	ABERCROMBIE & FITCH -CL A	20.36	3314.902	3125.384	0.035918	-0.034396

We also ranked these firms according to the three metrics using sort_values().

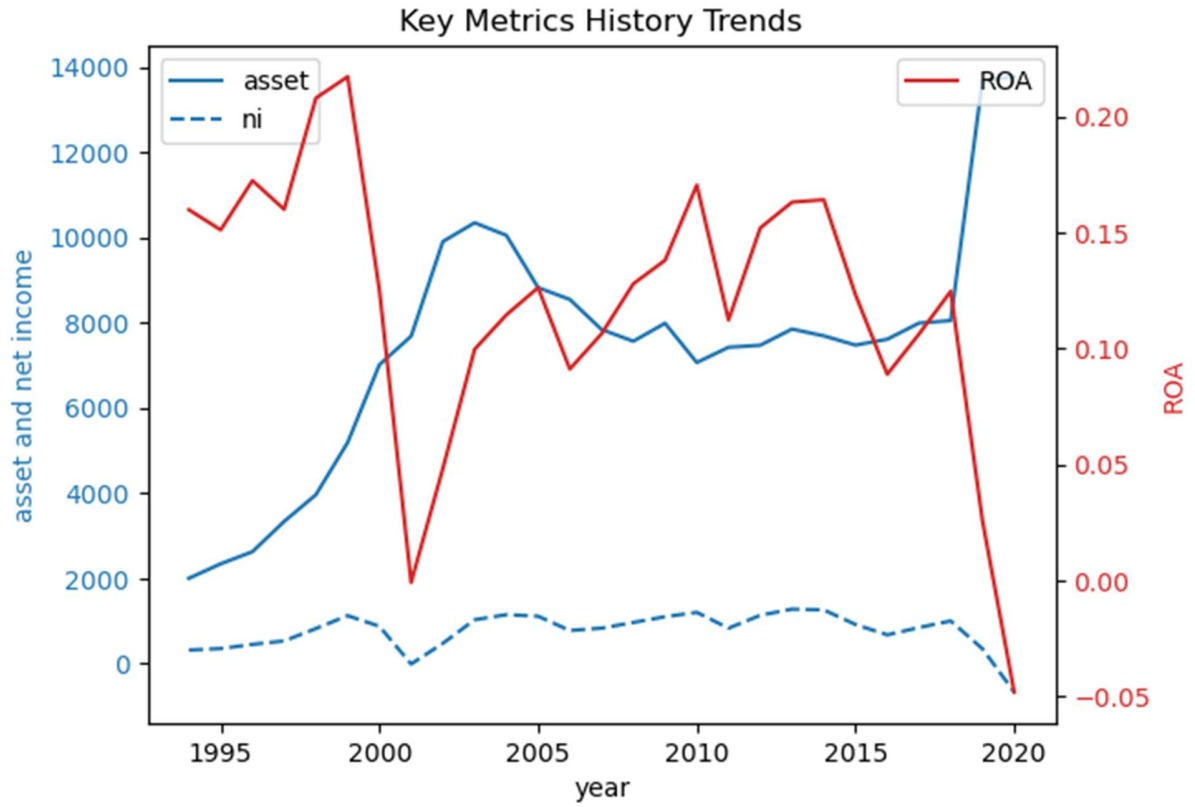
```
Sorted by Revenue:
                          conm    revenue
0                      GAP INC  13800.000
1  AMERN EAGLE OUTFITTERS INC   3759.113
2  ABERCROMBIE & FITCH  -CL A   3125.384

Sorted by Market Share:
                          conm  market_share
0                      GAP INC      0.158596
1  AMERN EAGLE OUTFITTERS INC      0.043201
2  ABERCROMBIE & FITCH  -CL A      0.035918

Sorted by ROA:
                          conm       ROA
0  ABERCROMBIE & FITCH  -CL A -0.034396
1                      GAP INC -0.048297
2  AMERN EAGLE OUTFITTERS INC -0.060927
```

The GAP INC performed the best in terms of revenue and market share among the competitors. But its ROA ranked after 'ABERCROMBIE & FITCH -CL A'. Thus, we decided to dive deeper into the historical trends of ROA and related metrics.

3. We plotted the lines showing the trends for ROA, net income, and assets. We noticed that in 2020, the ROA decreased significantly with a soaring in assets. Thus, we considered the main cause of decrease of ROA is the rapid rise in assets.



Based on the above analysis, our recommendation is that GAP INC should reduce the amount of assets, focusing on low-capital-intensive initiatives rather than bulky fixed assets. Lower fixed assets may help the firm feel less stressed in terms of managing and storing inventory.

GAP INC should invest more into encouraging creativity and brand strength may create more value for the firm. By adopting this strategy, GAP INC can link upstream and downstream supply chains more effectively, and better match production with demand. Hopefully, by doing so, the company will maintain its market presence while reducing financial risks.