Analyzing the Impacts of SuperBowl Ads

Spring 2022 - DS4A Data Engineering Cohort 1, Team 3

1.0 Introduction

Business Problem

On this day Super Bowl Sunday, advertisers spend millions of dollars to have their commercial played during the broadcast. Super Bowl ads are television commercials featured during the broadcast of the Super Bowl NFL Championship game that draws in an average of 92 million viewers. According to the President and CEO of EDO, Inc, he stated that “The Super Bowl is the single most engaging TV environment advertisers will see all year. TV viewers who saw an ad during the Super Bowl were over three times more likely to search online for that advertiser compared to everyday primetime tv.”(1)

Our team wants to analyze the impact of Super Bowl ads on a company’s brand and how consumer sentiment relates to these advertisements as well as the impact on stock prices. However, numbers are just numbers until the full story of how consumers react to an ad, and how they feel about the message, the product or company is also analyzed to give those numbers meaning.

Essentially, we are analyzing the effects of television advertising in the setting of the NFL Super Bowl broadcast.

Business Impact

It was reported by executive Dan Lovinger of NBC Sports Group that in 2022 the price for a 30-second ad went for upwards of $6.5 million. It is important to understand the impact and measure the effectiveness of an investment of such magnitude in order to maximize the media spending efforts and make informed strategic decisions(2).

Monitoring the response of the target audience for previous advertising campaigns can help uncover what matters to that audience, serving as guidance for the development of future distinctive campaigns. By understanding what sort of feedback and responses can be invoked by future ad viewers before the ad gets published, content editors or content managers can foresee and prevent any possible misunderstandings and better direct their marketing effort.

The results from a digital presence analysis can help to understand the brand perception and changes in brand recognition that result from an ad campaign. By analyzing how customers or prospect customers are responding to new advertisements, the firm can keep track of potential negative, positive or viral trends involving the company after an ad release, and also understand what the effect on the stock price is during ad periods if any.

2.0 Data Analysis and Computation

For our project, we utilized three primary sources of data: Twitter, Google Trends, and stock price. We included data from January-March within the years 2015-2022. As Superbowl happens around February, including data within this date range helped us understand brand awareness, stock values, and sentiments around the brand both pre-Superbowl ad and post-Super Bowl ad. This also assisted us in understanding if Superbowl ads influence the brand by mostly looking at companies that had a Superbowl ad in 2022 and looking at previous years if the same companies had Superbowl ads in the past. The details of each of the datasets are below:

- **Twitter Data(snscrape)**: We used snscrape to gather tweets that mention the brand.
  - **Dataset Name:** Twitter Data (snscrape)

- **Dataset description**: Twitter data pulled using snscrape TwitterSearchScraper. Data was filtered to download only english tweets, tweets that received engagements and tweets for specific date ranges only.
- **Rounded number of rows:** Max 750 per year per brand but the number varies depending on how much data the API was able to find for each brand term given the date ranges and filter. This was all sample data.
- **Number of relevant columns**: 15
- **Size:** 42MB (on average 6 years per brand). Using these files we created 3 different data frames:
  - With data for 2 days before Superbowl (14.6MB)

Average of numeric variables per company:

```
before_df.groupby('Company_Name').mean().filter(regex='Count$',axis=1)
```

Company_Name	Retweet Count	Reply Count	Like Count	Followers Count	Friends Count
Avocado from Mexico	7.245000	2.620000	6.650000	175989.195000	17258.600000
Budlight	13.476765	3.681955	65.061557	361102.984309	2996.409777
Budweiser	9.104816	2.132011	24.156941	316883.579037	4885.215297
Coca-Cola	12.992099	1.700903	31.886757	287951.140331	7035.234763
Doritos	14.318605	2.049612	98.542636	220235.940310	3370.938760
Jeep	6.201747	1.341930	24.570300	46794.937188	2486.914725
Mars	14.345556	2.261556	86.350222	213877.997778	4262.438222
Pepsi	15.380939	5.153717	59.462549	303302.615428	4258.319173
T-Mobile	17.715569	3.726689	86.783576	402266.858426	4999.523524
Tide	8.780923	1.524098	35.724638	89269.651500	4194.474553
Toyota	10.728829	2.193563	53.391235	201923.424104	3716.802556
amazon alexa	4.341463	0.670732	9.835366	297795.983740	6212.674797
pringles	8.546816	3.179775	42.898876	660119.464419	2701.273408
sprint	4.962222	0.663111	15.204889	94718.465333	1826.213333
squarespace	5.490741	0.805556	37.513889	145110.837963	5161.949074
turbotax	8.675462	5.184697	15.817942	165868.335092	7730.812665
uber eats	16.252708	4.699158	87.330927	370199.460890	4349.078219
weather tech	3.541667	1.791667	30.552083	102307.979167	3918.312500
wix	4.900000	0.931250	15.093750	167993.325000	6029.181250

Sum of numeric variables per company:

```
before_df.groupby('Company_Name').sum().filter(regex='Count$',axis=1)
```

Company_Name	Retweet Count	Reply Count	Like Count	Followers Count	Friends Count
Avocado from Mexico	1449	524	1330	35197839	3451720
Budlight	22331	6101	107807	598347645	4965051
Budweiser	16070	3763	42637	559299517	8622405
Coca-Cola	34533	4521	84755	765374131	18699654
Doritos	18471	2644	127120	284104363	4348511
Jeep	14909	3226	59067	112495029	5978543
Mars	64555	10177	388576	962450990	19180972
Pepsi	55033	18440	212757	1085216758	15236266
T-Mobile	41419	8713	202900	940499915	11688886
Tide	26053	4522	105995	264863056	12445006
Toyota	47003	9610	233907	884626521	16283312
amazon alexa	2136	330	4839	146515624	3056636
pringles	2282	849	11454	176251897	721240
sprint	11165	1492	34211	213116547	4108980
squarespace	1186	174	8103	31343941	1114981
turbotax	3288	1965	5995	62864099	2929978
uber eats	13506	3905	72572	307635752	3614084
weather tech	340	172	2933	9821566	376158
wix	784	149	2415	26878932	964669

Sum of numeric variables per company per year:

	Company_Name	Year	Retweet Count	Reply Count	Like Count	Followers Count	Friends Count
0	Mars	2018	26240	2969	109296	226958527	2483630
1	T-Mobile	2021	21228	4447	129149	77505391	1576775
2	Coca-Cola	2020	17337	1166	39144	82906963	1966534
3	Pepsi	2015	15057	2139	20632	124483867	3166859
4	Toyota	2022	14383	4450	69698	177956523	1921719
...	...	...	...	...	...	...	...
87	weather tech	2016	69	18	106	1609593	132362
88	weather tech	2022	58	23	461	645375	46910
89	weather tech	2018	48	15	111	1082882	130858
90	Avocado from Mexico	2018	24	3	57	13418	7807
91	Avocado from Mexico	2022	7	3	20	3825	5227

92 rows × 7 columns

Average of numeric variables per company:

```
after_df.groupby('Company_Name').mean().filter(regex='Count$',axis=1)
```

Company_Name	Retweet Count	Reply Count	Like Count	Followers Count	Friends Count
Avocado from Mexico	7.191904	2.259370	17.219640	2.478452e+05	5948.222639
Budlight	13.925171	3.357402	63.965914	1.964389e+05	2884.718233
Budweiser	26.420755	2.447484	100.568868	9.852033e+04	3929.434277
Coca-Cola	11.083294	1.477327	32.898091	1.149393e+06	20977.045823
Doritos	28.838797	4.508478	133.415555	1.394784e+05	3675.637124
Jeep	8.602133	2.314400	36.438933	1.136707e+05	4301.316533
Mars	11.784000	1.924889	51.903778	1.676772e+05	3206.970222
Pepsi	14.153778	2.732889	49.945111	1.914844e+05	4538.964000
T-Mobile	16.315431	3.939682	44.693452	5.238889e+05	5519.112993
Tide	9.708333	1.821333	67.296000	1.333936e+05	3947.286667
Toyota	9.678152	2.289350	36.759984	1.512429e+05	4189.750196
amazon alexa	5.615578	1.359296	29.694724	2.103804e+05	7893.133166
pringles	8.160291	2.229508	45.500911	2.878679e+05	2713.440801
sprint	9.244000	1.202667	16.490222	1.183010e+05	4452.500889
squarespace	9.825516	1.463415	28.346154	1.379584e+05	5326.975610
turbotax	10.129477	2.185491	21.375574	1.168477e+05	3442.832874
uber eats	38.535961	4.487552	233.738589	1.567512e+05	3008.665975
weather tech	10.601869	1.912150	14.528972	1.433798e+05	5346.927103
wix	14.992032	1.264940	11.874502	1.905277e+05	3708.673307

Sum of numeric variables per company:

```
after_df.groupby('Company_Name').sum().filter(regex='Count$',axis=1)
```

Company_Name	Retweet Count	Reply Count	Like Count	Followers Count	Friends Count
Avocado from Mexico	9594	3014	22971	330625471	7934929
Budlight	69041	16646	317143	973943997	14302433
Budweiser	84018	7783	319809	313294658	12495601
Coca-Cola	46439	6190	137843	4815956816	87893822
Doritos	127554	19941	590097	616912817	16257343
Jeep	32258	8679	136646	426265229	16129937
Mars	53028	8662	233567	754547250	14431366
Pepsi	63692	12298	224753	861679950	20425338
T-Mobile	78983	19072	216361	2536146224	26718026
Tide	29125	5464	201888	400180732	11841860
Toyota	49436	11694	187770	772548681	21401244
amazon alexa	4470	1082	23637	167462786	6282934
pringles	4480	1224	24980	158039480	1489679
sprint	20799	2706	37103	266177245	10018127
squarespace	10474	1560	30217	147063655	5678556
turbotax	11031	2380	23278	127247177	3749245
uber eats	55723	6489	337986	226662247	4350531
weather tech	5672	1023	7773	76708183	2860606
wix	7526	635	5961	95644914	1861754

Field	Type	Description
url	str	Permalink pointing to tweet location
date	timestamp	Date tweet was created
content	str	Text content of tweet
renderedContent	str	Appears to be a cleaner tweet content
id	Int	id of tweet
retweetCount	int	Count of retweets
replyCount	int	Count of replies to the tweet
likeCount	int	Count of likes
User.followersCount	int	Count of followers for the user
User.friendsCount	int	Count of friends for the user
User.username	str	Name shown for user
User.displayname	str	Unique name for user

- **Google Trends(pytrends):** To collect Google Trends data we used pytrends. Pytrends is an API to automate downloading reports from Google Trends.

    - **Dataset Name:** Google Trends (pytrends)
    - **Dataset description**: Google Trends pulled using pytrends. Interest: Interest will be on a scale of 1-100, with 100 being the highest interest. Dates of interest Jan-Mar. Trends compared by Product Type.
    - **Rounded number of rows:** 100 per year per product
    - **Number of relevant columns**: 2
    - **Size:** 2KB per year per product (on average 6 years per brand)

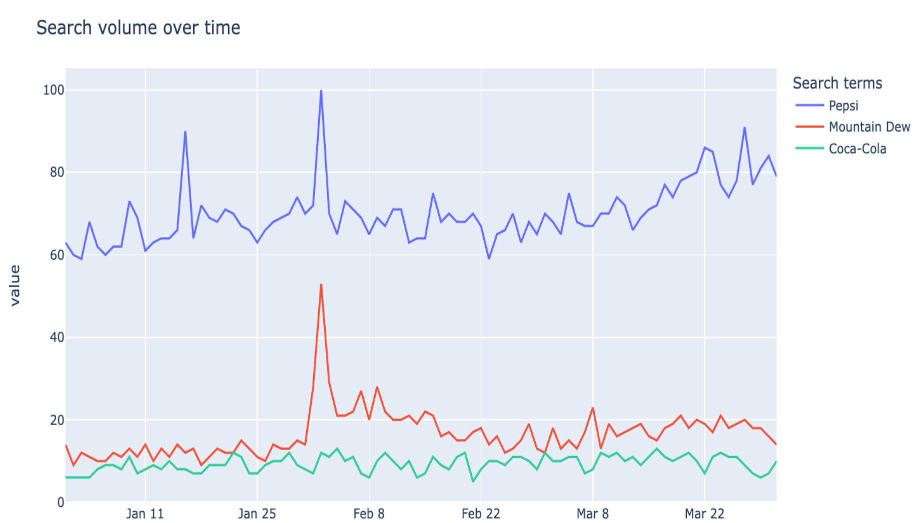
Field	Type	Description
date	timestamp	Time frame of trends Jan-March
Brand	Int	Interest Number

	Pepsi	Mountain Dew	Coca-Cola
date			
2015-01-01	63	14	6
2015-01-02	60	9	6
2015-01-03	59	12	6
2015-01-04	68	11	6
2015-01-05	62	10	8

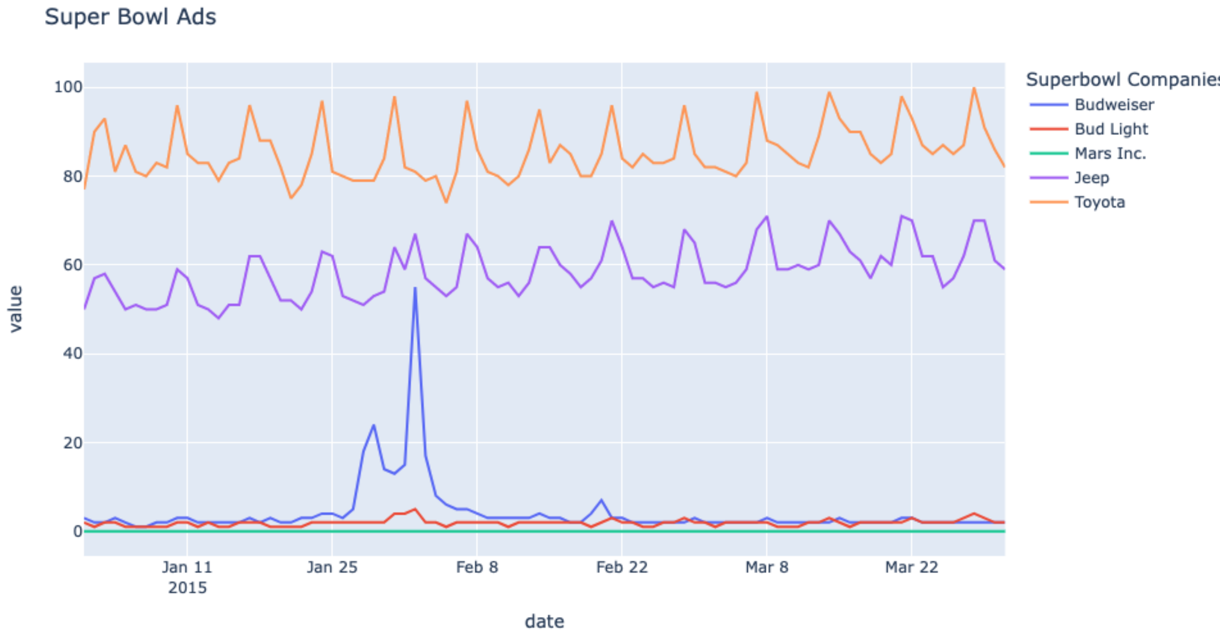
date	Pepsi	Mountain Dew	Coca-Cola
2015-03-27	91	20	9
2015-03-28	77	18	7
2015-03-29	81	18	6
2015-03-30	84	16	7
2015-03-31	79	14	10

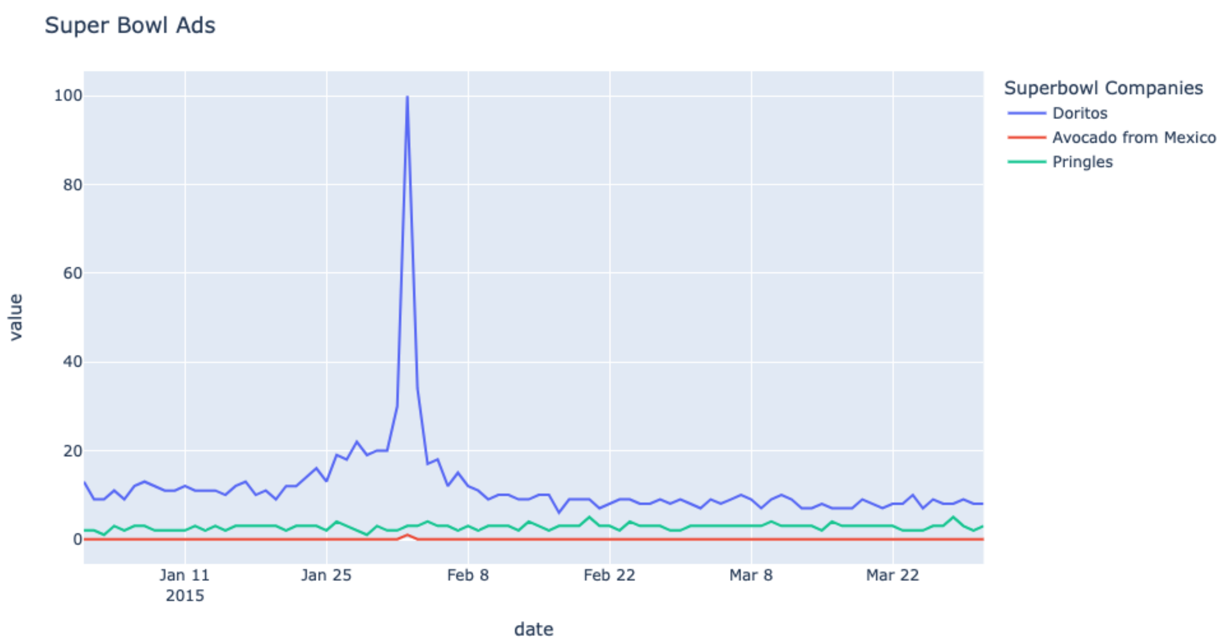
date	Budweiser	Bud Light	Mars Inc.	Jeep	Toyota
2015-01-01	3	2	0	50	77
2015-01-02	2	1	0	57	90
2015-01-03	3	2	0	58	93
2015-01-04	3	2	0	54	81
2015-01-05	2	1	0	50	87

Interest Over time of Google Trend for Soft Drinks: 2KB per year per product (on average 6 years per brand)



Interest over time of Google Trend for Beer, Candy, Car, and Food Companies





- **Yahoo Finance(yfinance)**: To pull stock data, we will be using stock trend data from Yahoo Finance using yfinance. yfinance is an open-source tool that uses Yahoo's publicly available APIs to collect data from yahoo finance. Data will be collected for all brands that had a Superbowl ad during the period. The data source will give us access to the following:

  - Date: The date period of stock prices to be downloaded
  - Open: The price of the stock when the market opens

- Close: The price of the stock when the market closed
- High: Highest price the stock reached during that period
- Low: Lowest price the stock is traded during that
- Volume: The total amount of stocks traded in the period considered

We have downloaded the yearly stock prices data using the Yahoo finance API functionality. It's a seven-year data capturing Open, High, Low, Close, and Volume as described above. We also looked at stock prices between Jan - March of each year in consideration as the Superbowl starting from 2015 occurs mostly in February.

Super Bowl Observances **Showing:** 2017–2027

Year	Weekday	Date	Name	Holiday Type
2017	Sun	Feb 5	Super Bowl	Sporting event
2018	Sun	Feb 4	Super Bowl	Sporting event
2019	Sun	Feb 3	Super Bowl	Sporting event
2020	Sun	Feb 2	Super Bowl	Sporting event
2021	Sun	Feb 7	Super Bowl	Sporting event
2022	Sun	Feb 13	Super Bowl	Sporting event
2023	Sun	Feb 12	Super Bowl	Sporting event
2024	Sun	Feb 11	Super Bowl	Sporting event
2025	Sun	Feb 9	Super Bowl	Sporting event
2026	Sun	Feb 8	Super Bowl	Sporting event
2027	Sun	Feb 14	Super Bowl	Sporting event

Stock Ticker Symbols:

Budweiser/Budlight = BUD

Mars Inc = MNBP

Jeep = STLA

Toyota = TM

Doritos / Pepsi = PEP

Avocado from Mexico = CVGW

Pringles(Kellogg) = K

Coca Cola = KO

T-mobile = TMUS

Tide (Procter Gamble) = PG

Turbo Tax(Intuit Inc.) = INTU

Wix.com = WIX

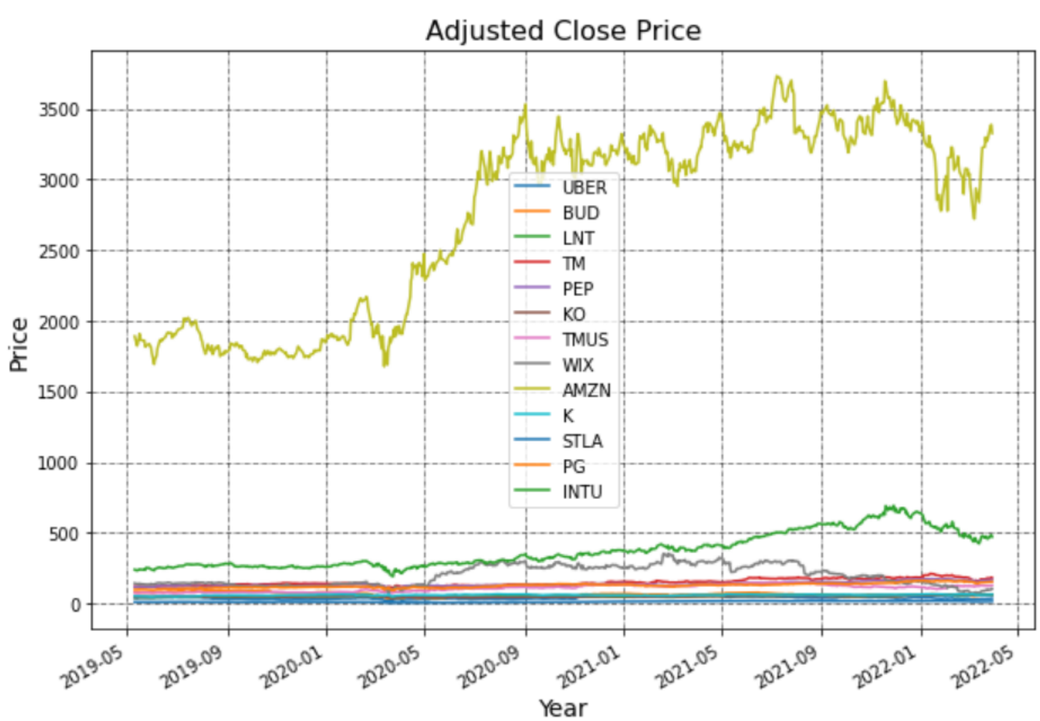
SquareSpace = SQSP * Went public in 2022

Alexa(Amazon) = AMZN

UberEats(Uber Technologies Inc.) = UBER * Went public in 2020

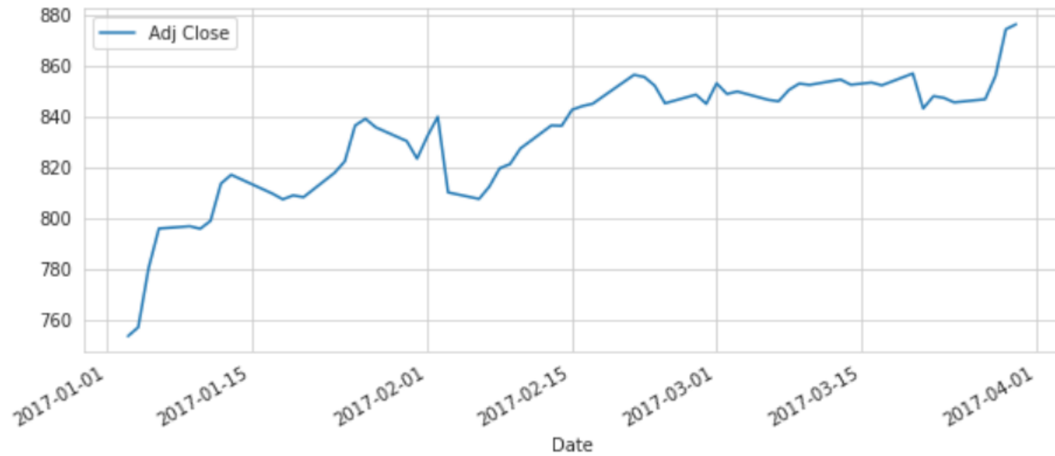
WeatherTech is not on the on the stock market

Adjusted Closing prices for all Stocks under consideration 2015 - 2022

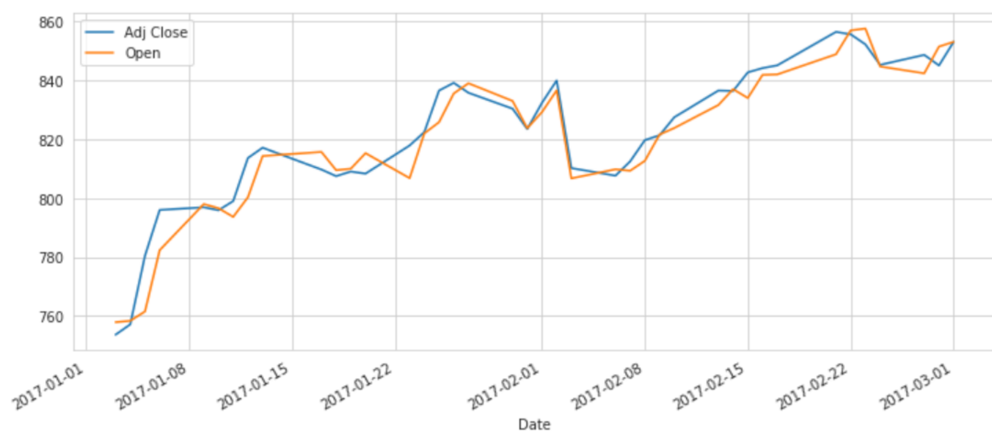


[23]
```python
# Historical view of the closing price of Amazon stock
AMZN['Adj Close'].plot(legend=True,figsize=(10,4))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5720377e10>
```

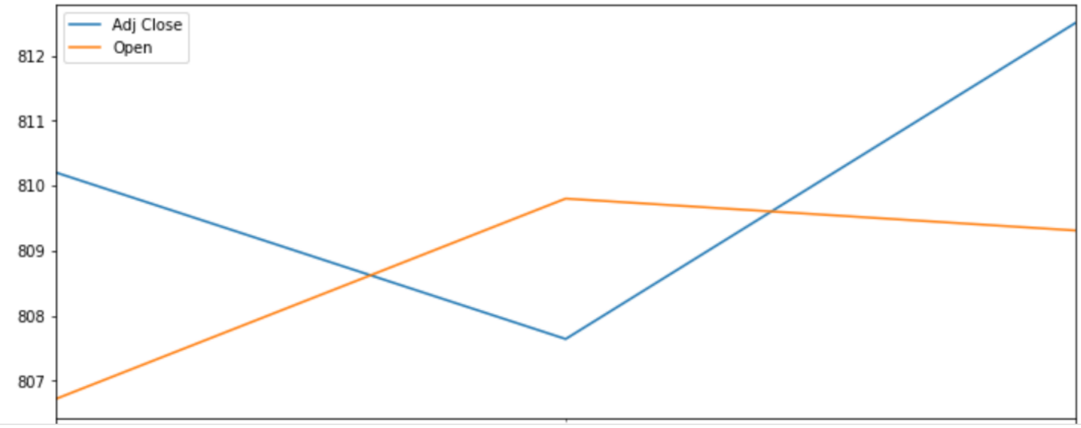



```python
# Plotting the stock's adjusted closing price using pandas before and after Superbowl 2017
AMZN.truncate(before='2017-01-01', after='2017-03-01')['Adj Close'].plot(legend=True,figsize=(12,5))
AMZN.truncate(before='2017-01-01', after='2017-03-01')['Open'].plot(legend=True,figsize=(12,5))
plt.legend();
```



```
# Plotting the stock's adjusted closing price using pandas day before and day after Superbowl 2017
AMZN.truncate(before='2017-02-03', after='2017-02-07')['Adj Close'].plot(legend=True,figsize=(12,5))
AMZN.truncate(before='2017-02-03', after='2017-02-07')['Open'].plot(legend=True,figsize=(12,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69dae4c210>
```

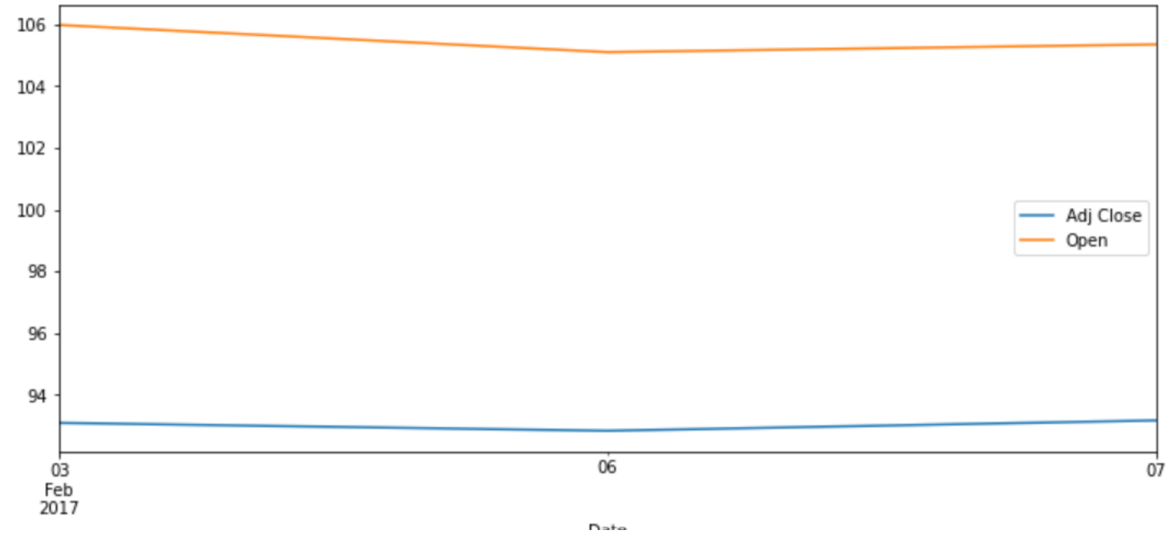



```
# Historical view of the closing price of BUD stock
BUD['Adj Close'].plot(legend=True,figsize=(10,4))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f571fe3b850>
```

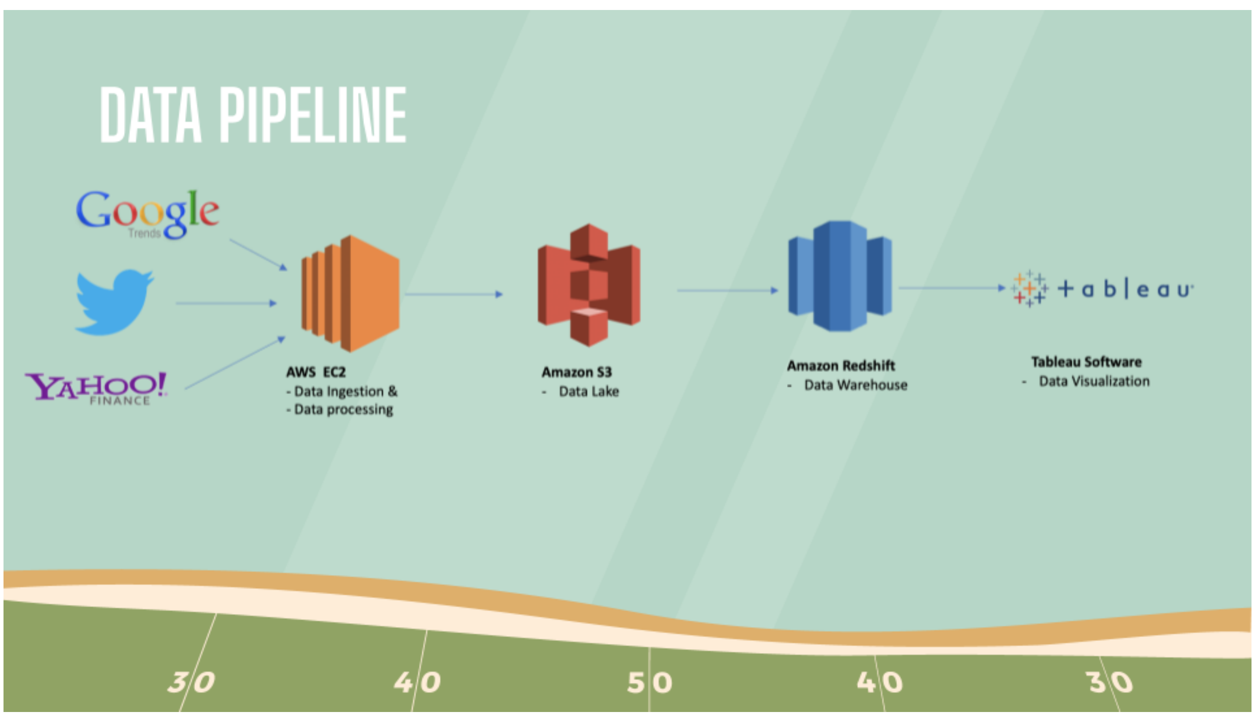
```
BUD.truncate(before='2017-02-03', after='2017-02-07')['Open'].plot(legend=True,figsize=(12,5))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f69daf43750>
```



3.0 Data Pipeline and Data Warehouse

a. Pipeline Diagram

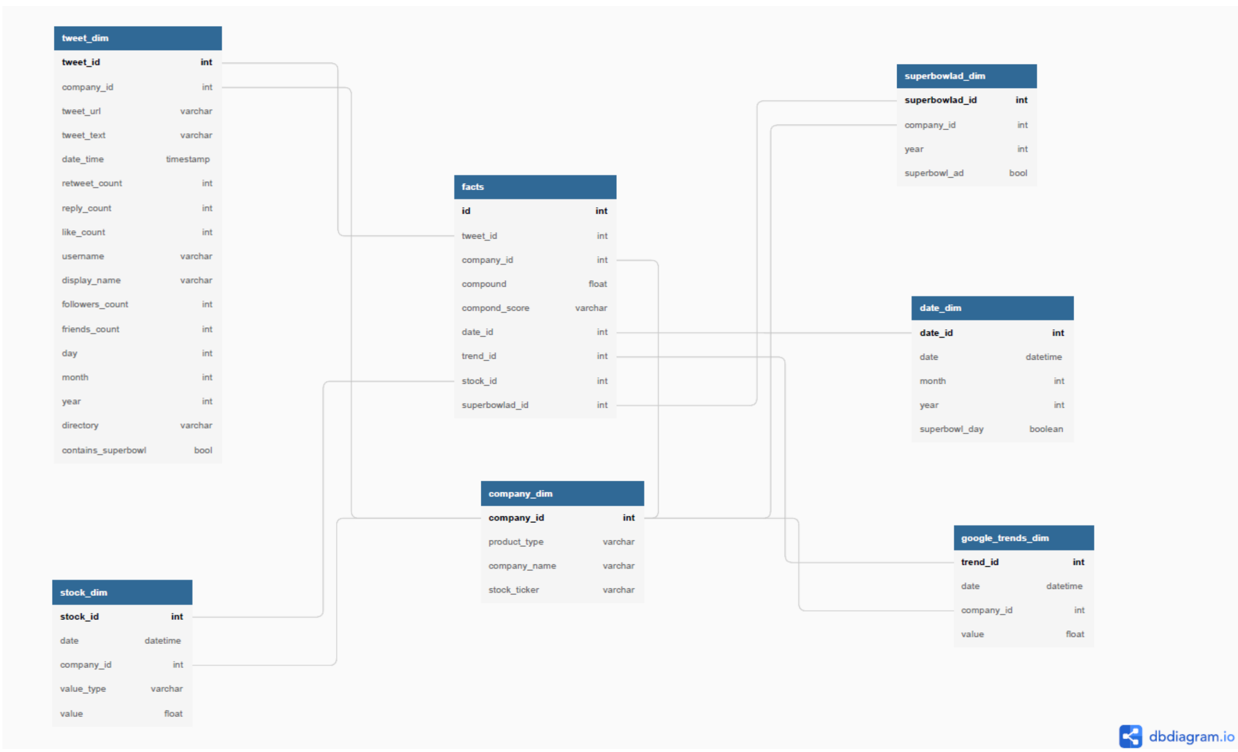


For the Data Pipeline, our data was ingested and processed into an AWS EC2. The AWS S3 bucket includes a prefix which identifies a specific day, month, year or even an entire entity.

AWS Redshift reads S3 objects and asynchronously processes the data to be used to create tables. The raw data is then processed into structured formats and aggregations that are visualized in Tableau which is hosted on an EC2.

We have currently included historical data within the s3 bucket. To make sure that data is updated during future superbowl years, we included a python script within EC2 that would update the data for Jan - Mar every year moving forward. This way brands can compare the effect of superbowl ads using google trends, stock and twitter data.

b. Data Schema



3.1 Data Details:

We used the data schema mentioned above when designing the data warehouse on redshift. The following is a detailed description of the different tables.

Tweet_dim:

Data from Twitter. Data contains tweets with company names two days before the superbowl and two days after the superbowl. Data is pulled from 2015 - 2022.

- Tweet_id(PK, int) : id of tweets
- Company_id(FK, int): id of companies

- Tweet_url(varchar): url of tweets
- Tweet_text(varchar): text of tweets that contains the company names
- Date_time(datetime): time at which the tweet is generated
- Retweet_count(int): number of retweets
- Reply_count(int): number of replies
- Like_count(int): number of liked
- Username(varchar): username of where the tweet come from
- Display_name(varchar): display name where the tweet came from
- Followers_count(int): number of followers of the user
- Friends_count(int): number of friends of the user
- Day(int): day of the month
- Month(int): month number
- Year(int): year
- Directory(varchar): labeled “before” and “after” superbowl
- Contains_superbowl(bool): contains if the tweet has the word “superbowl” or not

Facts:

- Id(PK, int): id of table
- Tweet_id(FK, int): id of tweets
- Company_id(FK, int):id of company
- Compound: sentiment score between -1 and 1
- Compound_score: labeled as pos, neg, neu based on sentiment score
- Date_id(FK, int): id of date
- Trend_id(FK, int):id of trend data
- Stock_id(FK, int): id of stock data
- Superbowl_ad(bool): identifies if there was a superbowl ad for that company in that specific year

Company_dim

- Company_id(PK, int): id of company
- Company_name(varchar): name of company
- Stock_ticker(varchar): stock ticker of the company. Values are included in the stock data

Stock_dim:

Data from yahoo finance stocks. Contains data from Jan 2015 - March 2022

- Stock_id(PK, int): id of stock data
- Date(date): data from which the data is pulled
- Company_id(FK int): id of companies

- Value_type(varchar): Value labels of the data. Data is labeled as: Close, high, low, open, volume
- Value(float): values of the the different value_types

Date_dim:

- Date_id(PK, int): id of date
- Date(date): all dates from 2015 -2022
- Month(varchar): month of the dates
- Year(int): year of the dates
- Superbowl_day(bool): contains values to see if superbowl happened on the day or not

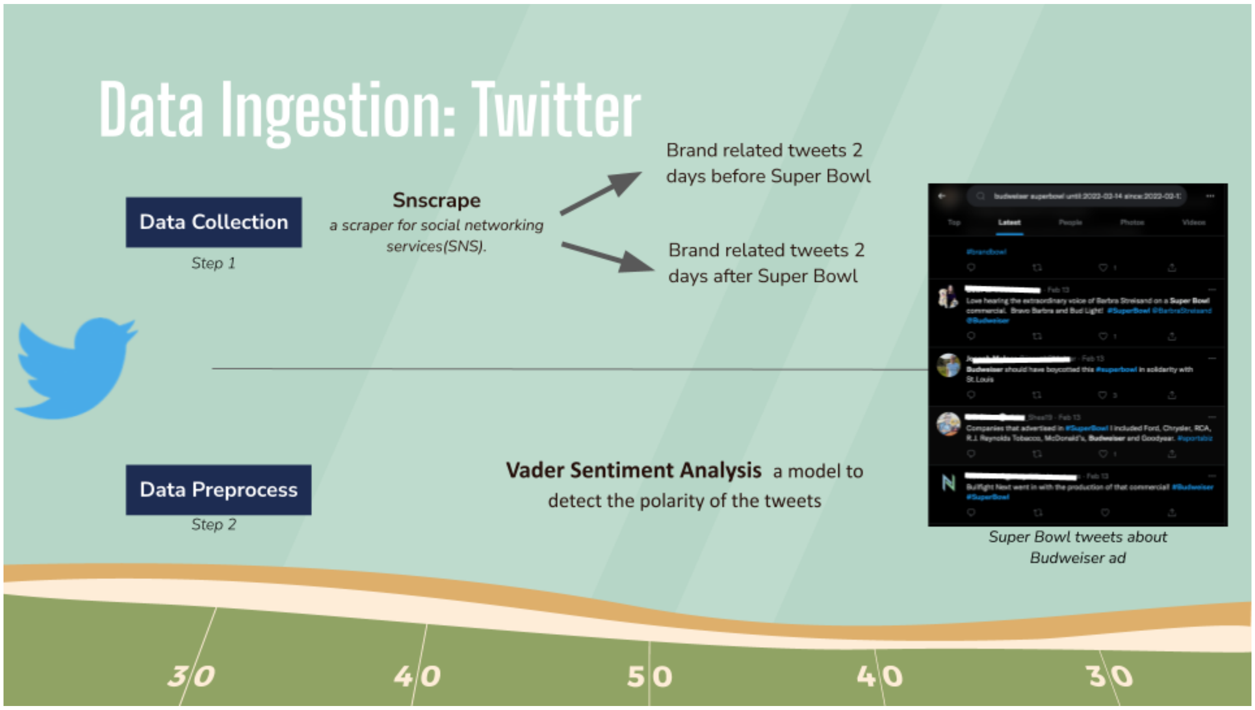
Google_trends_dim:

Data from google trends that contains company names. Data is pulled from January - March for every year from 2015-2022

- Trend_id(PK, int): id of the trends data
- Date(date): dates when the trend data was pulled
- Company_id(FB, int): id of companies
- Value(int): interest of the term during that time

3.2 Data preprocessing steps

A. Twitter



Data Preprocessing: After completing the data ingestion from Twitter and performing the preprocessing methodologies, we then performed a sentiment analysis to detect the polarity of the tweets. For this purpose we have used the Vader package, and we will be evaluating two different approaches.

The Vader package calculated the polarity scores as a dictionary of negative, positive, neutral and compound scores for each tweet. For our analysis we used the compound score, which was added to each tweet during the data preprocessing stage.

url	Datetime	Tweet Id	Text	Retweet Cou	Reply Count	Like Count	Username	Display Nam	Followers Co	Friends Coun	Day	Month	Year	brand	directory	compound	comp_score	contains_superbowl
https://twitt	2016-02-06 :	6.9612E+17	When I threv	2	1	1	katyh_13	katy	133	101	6	2	2016	Pepsi	before	0	neu	FALSE
https://twitt	2016-02-06 :	6.9612E+17	□üôå□üèæ tc	6	1	37	MattForte22	Matt Forte	399357	601	6	2	2016	Pepsi	before	0	neu	FALSE
https://twitt	2016-02-06 :	6.9612E+17	My brother F	1	0	6	yaboyluke2tu	the neighbor	487	302	6	2	2016	Pepsi	before	-0.5423	neg	FALSE
https://twitt	2016-02-06 :	6.9612E+17	The always-p	2	0	5	NickUniverse	Nickelodeon	3674	914	6	2	2016	Pepsi	before	0	neu	FALSE
https://twitt	2016-02-06 :	6.9612E+17	No. Pepsi is i	3	1	9	SmamSmani	Sam Bam	352	228	6	2	2016	Pepsi	before	-0.1511	neg	FALSE

Sample of twitter data in data lake

In order to understand if superbowl ads had an impact on brand sentiment or not, we looked at the sentiment scores closely. We have tested with two different approaches:

Analyzing mean sentiment score approach:

We calculated the compound score per brand per date using the following function by calculating the mean of the compound scores per brand per date.

```
def calculate_average_compound_score_per_brand_per_date(df):

    # Get the brand name from the brand colum
    brand = df['brand']

     # Convert the datatime to date
    df['Date'] = df['Datetime'].apply(lambda x: datetime.strptime(x, '%Y-%m-%d %H:%M:%S%z').date())

    #  Create a new data frame that shows the average compound score per date
    summary_df = df.groupby(['Date', 'brand']).mean()['compound'].reset_index()

    #  Round compound score to 3 decimals
    summary_df['compound'] = summary_df['compound'].apply(lambda x: round(x,3))

    # Rename compound column to average_compound_score
    summary_df.rename(columns={'compound':'average_compund_score'}, inplace=True)

    return summary_df
```

The expected output: A summary data frame showing the average compound score per brand per date. Dates include 2 days before the superbowl, the day of the superbowl and a day after the superbowl.

	Date	average_compund_score	brand
0	2015-01-30	0.448	Avocado from Mexico
1	2015-01-31	0.325	Avocado from Mexico
2	2015-02-01	0.376	Avocado from Mexico
3	2015-02-02	0.295	Avocado from Mexico
4	2016-02-05	0.479	Avocado from Mexico
...	...	...	...
314	2017-02-06	0.226	wix
315	2018-02-02	0.498	wix
316	2018-02-03	0.355	wix
317	2018-02-04	0.564	wix
318	2018-02-05	0.318	wix

<u>Analyzing percentage positive, negative, neutral sentiments approach:</u>
First we calculated the number of total tweets per company and per year for before and after the superbowl ad separately. Then we calculated the number of positive, negative and neutral sentiments per brand per year. This number was then used to find the percentage positive, negative and neutral tweets per brand per year. To understand the effect of the superbowl, we observed if the percentage positive/negative sentiments have increased or decreased after superbowl ads.

```python
join_before = pd.merge(before_df.groupby(['Year', 'Company_Name'])['Text'].count().reset_index().rename({'Text': 'Total_Tweets'}, axis = 1),
        before_df.groupby(['comp_score', 'Company_Name', 'Year'])['Text'].count().reset_index(),
        how='left',
        left_on=['Company_Name','Year'],
        right_on = ['Company_Name','Year'])

join_before['perc'] = join_before['Text']/join_before['Total_Tweets']

join_after = pd.merge(after_df.groupby(['Year', 'Company_Name'])['Text'].count().reset_index().rename({'Text': 'Total_Tweets'}, axis = 1),
        after_df.groupby(['comp_score', 'Company_Name', 'Year'])['Text'].count().reset_index(),
        how='left',
        left_on=['Company_Name','Year'],
        right_on = ['Company_Name','Year'])

join_after['perc'] = join_after['Text']/join_after['Total_Tweets']


join_after = join_after.rename({'perc': 'after_perc'}, axis = 1)
join_before = join_before.rename({'perc': 'before_perc'}, axis = 1)

join_final = pd.merge(join_before, join_after,
        how='left',
        left_on=['Company_Name','Year', 'comp_score'],
        right_on = ['Company_Name','Year', 'comp_score'])

def categorize(row):
    if row['after_perc'] > row['before_perc']:
        return 'increase'
    return 'decrease'

join_final['result'] = join_final.apply(lambda row: categorize(row), axis = 1)

join_final.head()
```

	Year	Company_Name	Total_Tweets_x	comp_score	Text_x	before_perc	Total_Tweets_y	Text_y	after_perc	result
0	2015	Avocado from Mexico	20	neu	6	0.300000	295	80	0.271186	decrease
1	2015	Avocado from Mexico	20	pos	14	0.700000	295	182	0.616949	decrease
2	2015	Budlight	385	neg	33	0.085714	750	106	0.141333	increase
3	2015	Budlight	385	neu	170	0.441558	750	217	0.289333	decrease

B. Google Trends:

In order to grab data from Google Trends, we are using Pytrends which is an API package to download reports from Google trends. We are collecting the data from companies that were ads in the Superbowl. To calculate the volatility of the companies, we did the interest over time.

After the API has been initialized, we query the keyword term we want to search for which is beer, candy, food, soft_drinks, phones, tech, etc. We use the method build_payload to tell the API which keywords we want and the timeframe which is from January 01, 2015 - March 31, 2015. We repeated the same process for the timeframe for 2016, 2017, 2018, 2019, 2020, 2021, & 2022.

	date	Budweiser	Bud Light	Mars	Jeep	Toyota	Doritos	Avocado fron	pringles	Pepsi	Mountain De	Coca-Cola	T-Mobile	sprint	Tide	weather tech	turbotax	wix.com	squarespace	amazon alex	uber eats
0	1/1/15	2	1	33	27	74	19	0	15	63	14	6	23	94	27	1	8	5	20	1	0
1	1/2/15	1	1	30	31	92	17	0	14	60	9	6	29	100	46	1	12	6	29	2	0
2	1/3/15	2	1	32	32	92	17	0	13	59	12	6	25	91	22	1	10	9	34	0	0
3	1/4/15	2	1	30	30	88	17	0	11	68	12	5	24	86	20	1	10	8	35	1	0
4	1/5/15	1	1	29	29	92	15	0	11	62	11	8	30	83	15	1	13	11	45	1	0
5	1/6/15	1	1	29	29	90	17	0	9	61	10	9	29	85	15	1	12	9	48	0	0
6	1/7/15	1	1	28	28	88	18	0	10	62	12	9	27	79	14	1	12	11	47	0	0
7	1/8/15	2	0	30	27	92	16	0	9	63	12	8	29	82	15	1	13	11	36	1	0
8	1/9/15	2	1	31	29	88	16	0	9	73	13	11	27	81	15	1	14	10	38	1	0
9	1/10/15	2	1	33	32	94	15	0	10	69	10	7	25	85	18	1	14	8	33	2	0

Sample of google trends data in data lake

```
##Code for only 2015. Same code will be used to run 2016-2022 for trends
company = pd.read_csv('Superbowl_Companies.csv')
company
# List of brands used for trends
beer = ['Budweiser', 'Bud Light']
candy = ['Mars Inc']
cars = ['Jeep', 'Toyota']
food = ['Doritos', 'Avacodo from Mexico', 'pringles']
soft_drinks = ['Pepsi', 'Mountain Dew', 'Coca-Cola']
phones = ['T-Mobile', 'sprint']
tech = ['turbotax', 'wix.com', 'squarespace', 'amazon alexa']
other = ['Tide', 'weather tech', 'uber eats']
normalize = True
```

```
#pytrends request
pytrend1 = TrendReq()
pytrend2 = TrendReq()
pytrend3 = TrendReq()
pytrend4 = TrendReq()
pytrend5 = TrendReq()
pytrend6 = TrendReq()
pytrend7 = TrendReq()
pytrend8 = TrendReq()
```

```
#building payload by product type and timeframe. we are looking at three months, Jan 1st to March 31st for the
#years 2015-2022. As an example, here is 2015.
pytrend1.build_payload(beer, timeframe='2015-01-01 2015-03-31')
pytrend2.build_payload(candy , timeframe='2015-01-01 2015-03-31')
pytrend3.build_payload(cars, timeframe='2015-01-01 2015-03-31')
pytrend4.build_payload(food, timeframe='2015-01-01 2015-03-31')
pytrend5.build_payload(soft_drinks, timeframe='2015-01-01 2015-03-31')
pytrend6.build_payload(phones, timeframe='2015-01-01 2015-03-31')
pytrend7.build_payload(tech, timeframe='2015-01-01 2015-03-31')
pytrend8.build_payload(other, timeframe='2015-01-01 2015-03-31')
```

Afterwards, we created a data frame that holds the time series data for this query. This will return historical, indexed data for the companies we are searching for.

```
#this puts the data into a dataframe
df1 = pytrend1.interest_over_time().drop(columns='isPartial')
df2 = pytrend2.interest_over_time().drop(columns='isPartial')
df3 = pytrend3.interest_over_time().drop(columns='isPartial')
df4 = pytrend4.interest_over_time().drop(columns='isPartial')
df5 = pytrend5.interest_over_time().drop(columns='isPartial')
df6 = pytrend6.interest_over_time().drop(columns='isPartial')
df7 = pytrend7.interest_over_time().drop(columns='isPartial')
df8 = pytrend8.interest_over_time().drop(columns='isPartial')
```

Lastly, we merged all the pytrends into one dataset then exported it to a csv which is placed in S3 bucket.

```
#merge all pytrends into one
All_2015 = df1.merge(df2,on='date').merge(df3,on='date').merge(df4,on='date').merge(df5,on='date').merge
(df6,on='date').merge(df7,on='date').merge(df8,on='date')
print(All_2015)

#export to csv
All_2015 .to_csv('All_2015 .csv', sep=',', index=True)
```

With the trends data, we then analyzed the percentage change of the variables/trends. The pct_change() method of DataFrame class in pandas computes the percentage change between the rows of data. We also calculated the mean interest before superbowl ads and compared it against the mean interest after superbowl using percentage change method per brand per year. This was then compared to both the twitter and stocks data.

C. **Yahoo Finance Stock**:

To analyze stock prices for 20 brands, we will be pulling stock trend data from Yahoo Finance using yfinance. Data was collected for all brands that had a Superbowl ad during 2015 - 2022. To find brand stock tickers, we had to use yahoo finance website to search for each brand as explained in the Data computation process above.

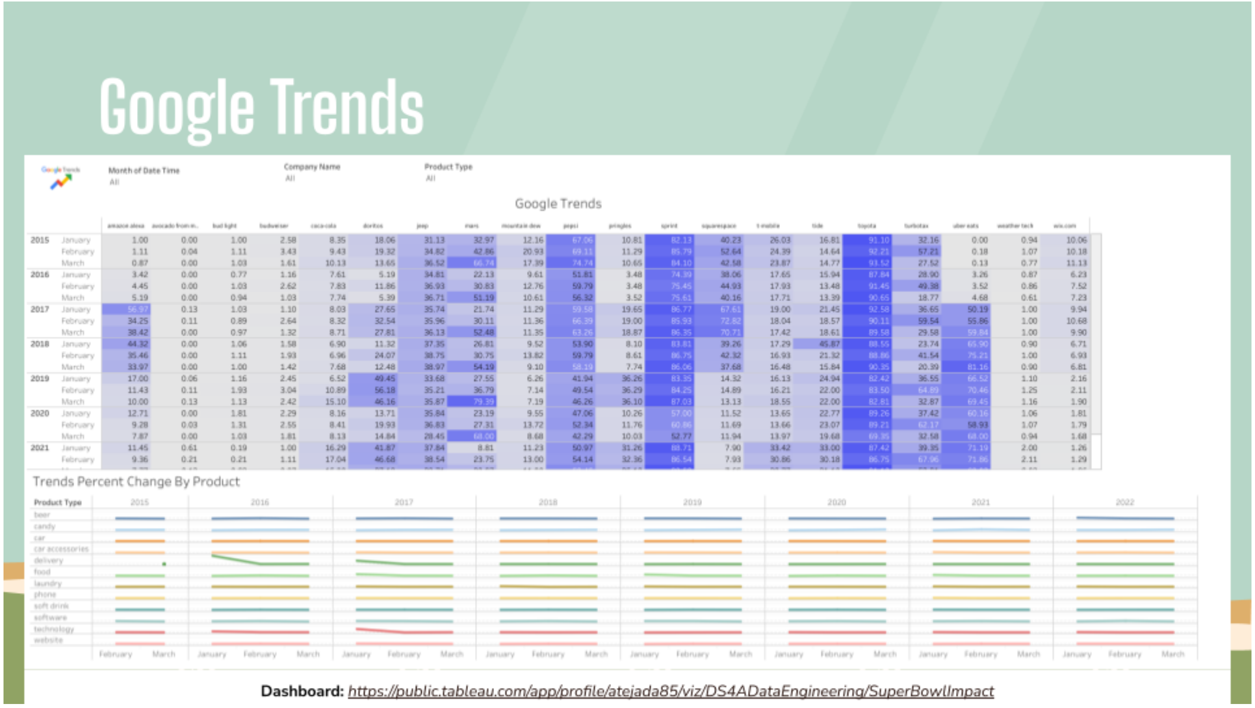
stock_id	Date	Ticker Name	Value Type	Values
0	5/10/19	UBER	High	45
1	5/13/19	UBER	High	39.2400017
2	5/14/19	UBER	High	39.9599991
3	5/15/19	UBER	High	41.8800011
4	5/16/19	UBER	High	44.0600014
5	5/17/19	UBER	High	43.2900009
6	5/20/19	UBER	High	41.6800003
7	5/21/19	UBER	High	42.2400017
8	5/22/19	UBER	High	41.2799988
9	5/23/19	UBER	High	41.0900002
10	5/24/19	UBER	High	41.5099983
11	5/28/19	UBER	High	41.7999992
12	5/29/19	UBER	High	40.7099991
13	5/30/19	UBER	High	40.3800011
14	5/31/19	UBER	High	41.5699997
15	6/3/19	UBER	High	41.8499985
16	6/4/19	UBER	High	42.8800011
17	6/5/19	UBER	High	45.6599999
18	6/6/19	UBER	High	45.75
19	6/7/19	UBER	High	45.6699982

Sample of stock data in data lake

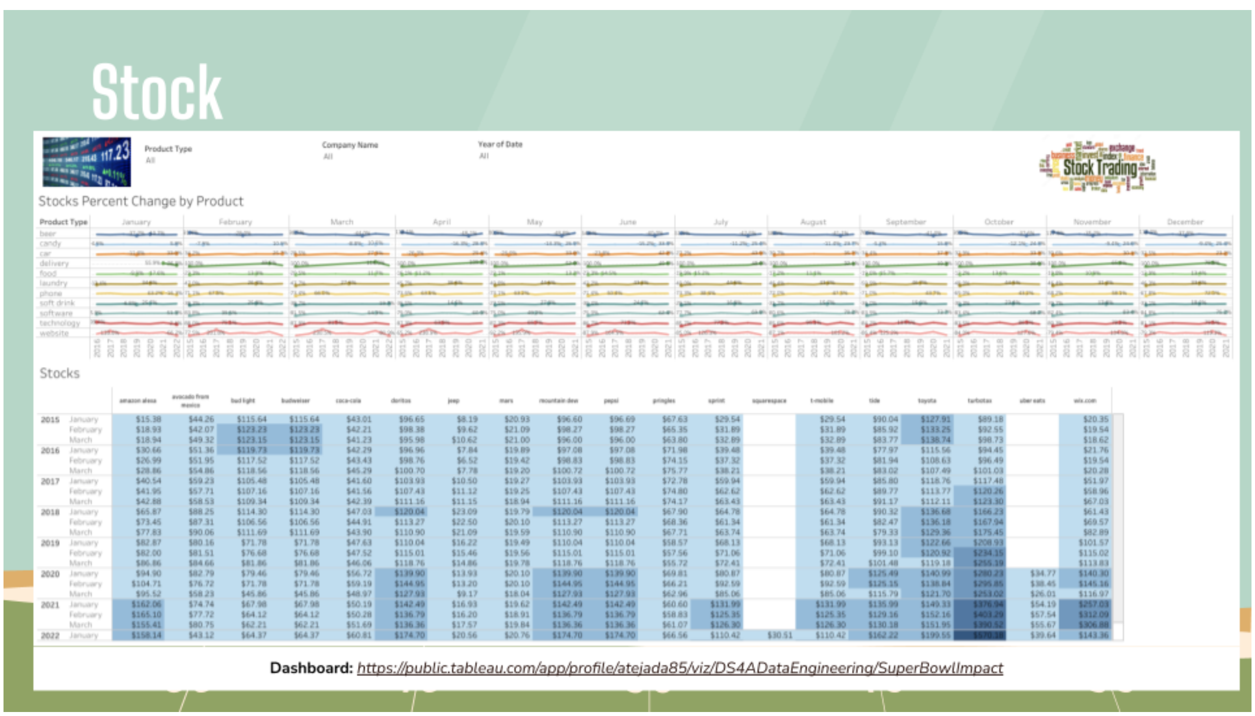
4.0 Data Dashboard

A. Use Case for Dashboard

The Dashboard for the twitter data showcases the compound score broken down by company and industry. We can also see how many retweets and likes each company has before and after superbowl.



The Google trends dashboard showcases which brand was trending one month before, the month of superbowl and one month after superbowl for each year using a time series chart to understand if the brand has search interest or not during, before and after superbowl months.



The Stock prices dashboard showcases and tracks stock prices for each brand for months before, during and after the superbowl.

B. Highlights from the Dashboard

It was observed that brands such as Toyota, Sprint, Pepsi were trending all eight years during the Super Bowl while Amazon Alexa, Uber eats, Jeep and Doritos started to trend more frequently in the later years. Turbotax, Toyota, Pepsi, Doritos, Bud Light and Budweiser all saw increased stock prices all eight years during the Super Bowl.

All of the companies we researched had a positive score in their sentiments. We also saw that Budweiser, Doritos, Mars, Pepsi and Toyota had the most retweet counts for all eight years two days before and two days after the Super Bowl in the years considered.
Another aspect we looked at was the number of likes a tweet received before and after the Super Bowl. In this case, Mars and Toyota had the most likes and retweets before the Superbowl while Doritos and Uber Eats had the most likes and retweets after the Superbowl for all eight years. Based on all of the data, Google Trends, Yahoo Finance and Tweets, Superbowl ads do have an impact for companies who advertise.

Link to Dashboard:

https://public.tableau.com/views/DS4ADataEngineering/SuperBowlImpact?:language=en-US&:display_count=n&:origin=viz_share_link

5.0 Conclusion and Future work

The first things that come to mind when you think about the superbowl are the game, the audience and Superbowl ads. Super Bowl ads draw in an average of 92 million viewers. In 2022 alone, the price for a 30-second ad went for upwards of $6.5 million and to justify the investment of the ads, we need to understand the impact these ads are having on the brands.

In this project, we concluded that analyzing the impact of superbowl ads on company brands is effective for the following:

- Help to uncover what matters to the audience
- Better direct marketing efforts
- Keep track of positive, negative and viral trends
- Understand effect on stock price

REFERENCES

1. https://www.globenewswire.com/news-release/2022/02/14/2384508/0/en/Super-Bowl-LVI-Sees-Advertising-Boom-as-Marketers-Get-Back-in-the-Game.html

2. https://www.sportingnews.com/us/nfl/news/super-bowl-commercials-cost-2022/v9ytfqzx74pjrcdvxyhevlzd
3. https://pypi.org/project/pytrends/
4. https://www.timeanddate.com/holidays/us/super-bowl

5. https://pypi.org/project/yfinance/