

**SP JAIN SCHOOL OF GLOBAL MANAGEMENT**  
**BACHELOR OF DATA SCIENCE**



S P Jain  
School of Global  
Management

DUBAI • MUMBAI • SINGAPORE • SYDNEY

**TWITTER SENTIMENT ANALYSIS**  
**FOR APPAREL BRANDS: NIKE AND ADIDAS**

**Group 3**

Anh Ngo – BS20DSY035

Vu Truong – BS20DSY037

Vinh Pham – BS20DSY038

Quang Huynh – BS20DSY027

Thanh Nguyen – BS20DSY029

Sydney, March 26, 2023

# CONTENTS

<b>1. Introduction.....</b>	<b>1</b>
<b>1.1 Problem Statement.....</b>	<b>1</b>
<b>1.2 Approach.....</b>	<b>1</b>
<b>2. Used technology.....</b>	<b>2</b>
<b>2.1 Social Media Platform .....</b>	<b>2</b>
<b>2.2 Scraping Library .....</b>	<b>2</b>
<b>3. User manual.....</b>	<b>4</b>
<b>4. Testing.....</b>	<b>7</b>
<b>5. Conclusion .....</b>	<b>12</b>
<b>5.1. Generalizability .....</b>	<b>12</b>
<b>5.2 Conclusion and Future Work.....</b>	<b>12</b>

# **1. Introduction**

## **1.1 Problem Statement**

Social media like Twitter or Facebook is a platform for everyone to express their opinions and discuss hundreds of topics in the world, from politics, and events to brands, entertainment, or sports. Nowadays, it plays a vital role in assisting entrepreneurs to improve their market thanks to huge and real-time sentiment data in every tweet post. Marketers or strategic planners can implement sentiment analysis solutions through machine learning techniques to identify, classify subjective information from people which then is beneficial for businesses to increase profits and plan an effective marketing strategy.

Knowing the importance and urgency of the topic in our modern and fast-paced changing world, we have decided to come up with two renowned apparel brands Nike and Adidas to do a deep-dive sentimental analysis and learn people's judgment and feeling on the two entities shared on social media.

The purpose of this study would be to obtain a deeper understanding of how customers perceive the two companies and to identify potential areas for improvement in their marketing and brand positioning strategies. The analysis might also assist companies in comprehending the competitive landscape and identifying areas in which they can differentiate themselves from their competitors.

## **1.2 Approach**

Our target is trying to extract all statistics and information (likes, reactions, shares, etc.) on Twitter about these two entities by using Twitter API. Due to time constraints, we aim to construct a dataset that contains latest observations for each entity. Furthermore, some NLP techniques and Python library (NLTK), particularly TextBlob sentimental analysis model are applied and explained; visualization is built to understand what categories people said about Nike and Adidas and how negative or positive the verbatims in the tweets are towards these categories.

## 2. Used technology

### 2.1 Social Media Platform

Twitter is a networking site launched in 2006 that allows users to share their thoughts via tweets and connect with others on a global scale. Over time, it has become one of the most popular social media platforms for businesses, politicians, celebrities, and others to communicate with their audiences, as well as journalists to report news promptly.

Twitter is a valuable source of data for web scraping because it contains a large volume of public information on hundreds of topics, including news, events, politics, entertainment, sports, and more. It also allows developers to access tweets, user profiles, and other metadata in real-time to identify trends, understand user behaviours and perform sentiment analysis.

### 2.2 Scraping Library

#### 2.2.1. *Data Manipulation and Analysis*

**NumPy:** provides powerful functions for effectively working with arrays and numerical data. The keys advantages of utilizing this library include compact storage, fast array loops, array operations and its compatibility when it can be used with other libraries like SciPy, Pandas and Matplotlib.

**Pandas:** is a popular open-source Python library used for manipulating and analyzing data. It includes data structures for storing and processing huge datasets effectively, as well as capabilities for reading and writing data to and from various file formats.

#### 2.2.2. *Data Visualization*

**Matplotlib and Seaborn:** they are both user-friendly libraries for building interactive figures and charts. While Matplotlib is powerful in compatibility and easy to customise the visual style and configuration, Seaborn provides a high-level visual solution with more selection of visualizations and is conveniently working with complex datasets.

**WordCloud:** a natural language processing (NLP) graphical visual of word frequencies that captures the theme of main points and entities in the dataset. They usually play an important role in sentiment analysis.

#### 2.2.3. *Twitter API*

**Tweepy:** is a Python library for accessing the Twitter API. The library allows us to easily retrieve tweets, submit tweets, search for tweets, stream live tweets, and more. Tweepy offers a user-friendly interface for dealing with Twitter's API and abstracts away a significant portion of the complexity needed in performing API calls.

## 2.2.4. NLP Predefined Model and Processing tool

**TextBlob** is a well-known package for NLP that offers simple capabilities for processing and analyzing text data. These tools are extremely effective in a sentiment analysis project when their output includes polarity and subjectivity scores.

**NLTK**: (Natural Language Toolkit) is a Python library for manipulating human language data. It offers tools for tokenization, stemming, lemmatization, and part-of-speech tagging that help increase the accuracy of the sentimental model. In our project, we mostly use the library for text cleaning, such as:

- **WordNetLemmatizer**: Lemmatization
- **Porterstemmer** and **Snowballstemmer**: Stemming
- **nltk.corpus.stopwords**: remove non-meaning-generating English word

**re**: a library that identifies non-value components of a sentence by regular expression. In our project, we apply the library to remove any mentions, hashtags, punctuation, special characters and URL before doing the analysis.

**shap**: model explainability shap that assist human to interpret and understand the machine learning model work on features and come up with the scores. In this project, we apply **shap** explainer on the pre-defined **TextBlob** model to learn more about how the model comes up with the polarity score (target) based on its words in the sentence (features).

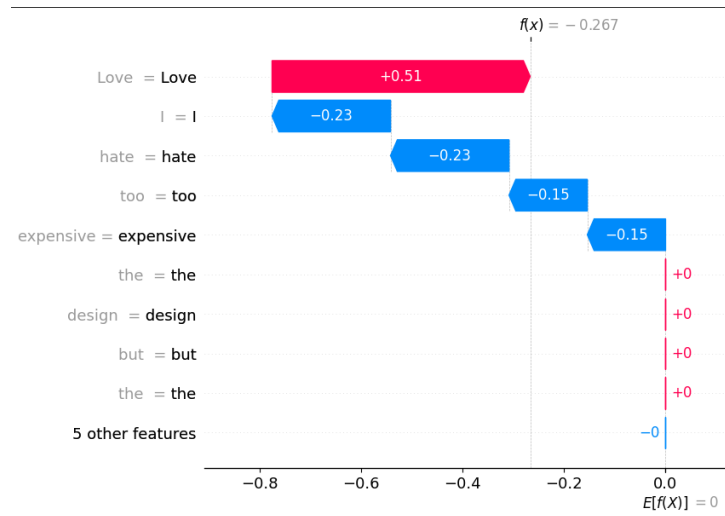


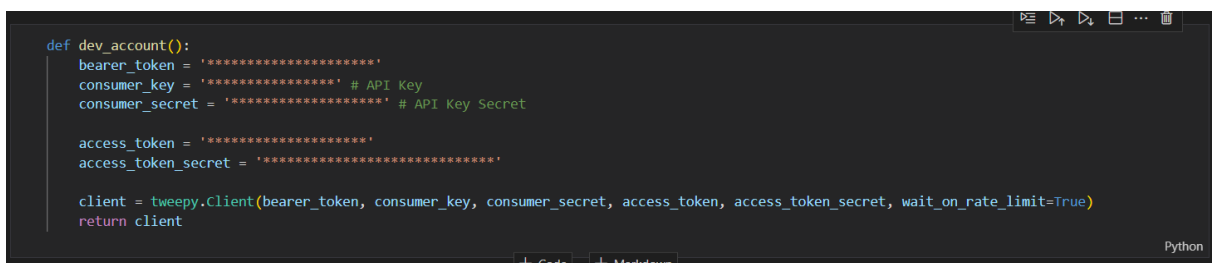
Fig 2.1: The local explainer on TextBlob model for a text sample *"Love the design but I hate the fact that the price is too expensive"*. The result shows that the text is defined as *negative* ( $-0.267 < -0.1$ ) and how words in the sentence contribute to that score

### 3. User manual

This user manual provides an overview of the code for performing sentiment analysis on tweets related to Adidas and Nike using Tweepy, TextBlob, and various natural language processing techniques. The script extracts tweets from Twitter, cleans and processes the data, performs sentiment analysis using TextBlob, and visualizes the results using word clouds, bar charts, and density plots. Additionally, SHAP (SHapley Additive exPlanations) is used to perform global and local model interpretability analysis.

#### *Prerequisites and Steps by steps to run the program:*

1. Make sure your **pip** is up-to-date  
`pip3 install -U pip`
2. Python 3. x installed on your system
3. The following Python libraries are installed: numpy, pandas, matplotlib, seaborn, tweepy, textblob, NLTK, collections, wordcloud, requests, and shap. You can install these libraries using pip command in your terminal or Anaconda prompt.  
`pip install TextBlob`  
`pip install Tweepy`  
`pip install shap`  
`pip install wordcloud`  
`pip install nltk`  
`pip install pandas`  
`pip install numpy`  
`pip install matplotlib`
4. Go to [our GitHub](#) and pull the code to your local machine. There are two files which are **main.py** and **Final\_data.csv**
5. Use the dev\_account() function to get the Twitter API access keys and tokens. Replace the placeholders with your own Twitter keys and tokens, which can be obtained by creating a developer account on Twitter.



```
def dev_account():  
    bearer_token = '*****'  
    consumer_key = '*****' # API Key  
    consumer_secret = '*****' # API Key Secret  
  
    access_token = '*****'  
    access_token_secret = '*****'  
  
    client = tweepy.Client(bearer_token, consumer_key, consumer_secret, access_token, access_token_secret, wait_on_rate_limit=True)  
    return client
```

- For the first time running the code or you don't have the dataset, you have to run the code from the beginning to the end. The **Final\_data.csv** will be saved in the same folder. When running **Lemmatization** and **Stemming**, please do uncomment **nlk.download('wordnet')** for the first time running.

## 2.3 Stemming and Lemmatization

```
#nlk.download('wordnet')
def Lemmatization(text):
    wn = WordNetLemmatizer()
    w = [wn.lemmatize(word) for word in text.split(' ')]
    return(" ".join(w))
```

- However, we have already provided you with the **Final\_file.csv** which means the data extractions and cleaning do not need to be run again. Hence, the program should begin with **EDA and Visualisation** part.

## 3. EDA and Visualisation

### 3.1 Text Visualisation

```
#All of the stats are left-skewed
df[['followers_count', 'Likes', 'Retweets', 'Reply', 'Impression']].describe()
```

[15] Python

	followers_Count	Likes	Retweets	Reply	Impression
count	4.355000e+03	4355.000000	4355.000000	4355.000000	4.355000e+03
mean	6.616520e+05	148.819518	14.859242	4.363490	2.765877e+04
std	2.454094e+06	1900.258197	181.109376	33.229528	1.719371e+05
min	1.670000e+02	0.000000	0.000000	0.000000	0.000000e+00
25%	4.699150e+04	2.000000	0.000000	0.000000	1.836500e+03
50%	2.686050e+05	8.000000	1.000000	0.000000	1.022600e+04

### *All functions in our code:*

- dev\_account(): This function returns the Twitter API access keys and tokens required to authenticate the Twitter API client.
- extract\_data(response): This function extracts the relevant information from the Twitter API response and returns a pandas DataFrame.
- extract\_hashtags(df): This function extracts the hashtags from the tweets and adds them to the DataFrame.
- clean(text): This function cleans the text by removing URLs, mentions, hashtags, punctuation, and numbers, and removes any non-ASCII characters.
- removestopwords(text): This function removes stop words from the text using NLTK.
- Lemmatization(text): This function performs lemmatization on the text using NLTK.
- Porterstemmer(text): This function performs stemming using the Porter stemmer algorithm.
- Snowballstemmer(text): This function performs stemming using the Snowball stemmer algorithm.

9. `wordcloud(wc1, wc2)`: This function generates a word cloud for both Adidas and Nike tweets using the WordCloud library.
10. `top_10_words(all_words)`: This function counts the frequency of all the words in the verbatim and returns the top 10 most frequent words.
11. `barchart_top(top1, top2)`: This function generates a bar chart of the top 10 words for both Adidas and Nike tweets.
12. `textBlob_model(X)`: This function returns the polarity scores for each tweet using the TextBlob library.
13. `shap.Explainer(model)`: This function performs SHAP global and local interpretability analysis on the sentiment analysis model.

***Outputs:***

1. A cleaned and processed dataset is saved as a CSV file named "Final\_data.csv".
2. Word clouds and bar charts for the top 10 words in Adidas and Nike tweets.
3. Density plots of the polarity and subjectivity scores for Adidas and Nike tweets.
4. SHAP bar plots and waterfall plots for the sentiment analysis model.



## 4. Testing

### Case 1 – Remove special text components:

Below is a function used to remove special text components:

```
1 def clean(text):
2     text = re.sub("https?\S+", "", text) #remove URL
3     text = re.sub("@\S+", "", text) #remove mentions
4     text = re.sub("#\S+", "", text) #remove hashtags
5     text = text.encode('ascii', 'ignore').decode() # Remove unicode characters
6     text = re.sub('[%s]' % re.escape(string.punctuation), ' ', text) # Remove punctuation
7     text = re.sub('\w*\d+\w*', '', text) #remove number
8     return text
```

#### a. Remove icons, mention and URL link

(i) Input: “@adidas is considering donating profits from leftover Yeezy stock to charity 🤖🔗https://t.co/WQhXKWtOir”

(ii) Case description: This text contains icons (🤖, 🔗), mention (@adidas) and URL link (https://t.co/WQhXKWtOir”) which are not useful for analysis

(iii) Output: “ is considering donating profits from leftover Yeezy stock to charity”

```
1 text = "@adidas is considering donating profits from leftover Yeezy stock to charity 🤖🔗https://t.co/WQhXKWtOir"
2 clean(text)
✓ 0.0s
' is considering donating profits from leftover Yeezy stock to charity '
```

#### b. Remove punctuation, hashtag, number, special characters

(i) Input: “#MARKETS | Adidas CEO warns donating Yeezys could backfire as the company struggles to get rid of its \$1.3 billion stockpile.”

(ii) Case description: This text contains punctuation (.), hashtag (#MARKETS), number (1.3) and some special characters (| and \$) which are not useful for analysis

(iii) Output: “ Adidas CEO warns donating Yeezys could backfire as the company struggles to get rid of its billion stockpile “

```
1 text = '#MARKETS | Adidas CEO warns donating Yeezys could backfire as the company struggles to get rid of its $1.3 billion stockpile.'
2 clean(text)
✓ 0.0s
' Adidas CEO warns donating Yeezys could backfire as the company struggles to get rid of its billion stockpile '
```

### Case 2 – Remove stop words:

Below is a function used to remove stop words:

```
1 def removestopwords(text):
2     # Convert text to lowercase and split to a list of words
3     tokens = word_tokenize(text.lower())
4     # Remove stop words
5     stop_words = set(stopwords.words('english'))
6     token = [t for t in tokens if t not in stop_words]
7     return(" ".join(token))
```

(i) Input: “The company has yet to decide what to do with unsold Yeezy inventory”

(ii) Case description: Yeezy has a capital letter, and the text contains some stop words (the, has, to, what, do, with)

(iii) Output: “company yet decide unsold yeezy inventory “

```
1 text = 'The company has yet to decide what to do with unsold Yeezy inventory'
2 removestopwords(text)
✓ 0.1s
'company yet decide unsold yeezy inventory'
```

### Case 3 – Stemming and Lemmatization:

```
1 #nltk.download('wordnet')
2 def Lemmatization(text):
3     wn = WordNetLemmatizer()
4     w = [wn.lemmatize(word) for word in text.split(' ')]
5     return(" ".join(w))
6
7 def Snowballstemmer(text):
8     ss = SnowballStemmer(language = 'english')
9     w = [ss.stem(word) for word in text.split(' ')]
10    return(" ".join(w))
```

(i) Input: “I just received a lovely wonderful shoes order with free shipping. My cousins also love it”

(ii) Case description: Lemmatization remove inflectional endings only and to return the base or dictionary form of a word while Stemming is to reduce inflected word to word stem format

(iii) Output:

Lemmatization: “I just received a lovely wonderful shoe order with free shipping. My cousin also love it “

Stemming: “i just receiv a love wonder shoe order with free shipping. my cousin also love it”

```
1 text = 'I just received a lovely wonderful shoes order with free shipping. My cousins also love it'
2 Lemmatization(text)
✓ 3.7s
'I just received a lovely wonderful shoe order with free shipping. My cousin also love it'

1 text = 'I just received a lovely wonderful shoes order with free shipping. My cousins also love it'
2 Snowballstemmer(text)
✓ 0.1s
'i just receiv a love wonder shoe order with free shipping. my cousin also love it'
```

## Case 4 – Sentiment Analysis:

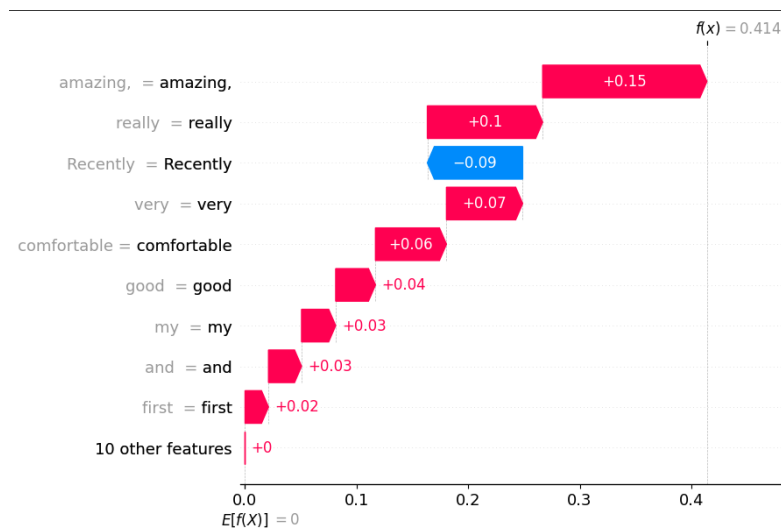
### 1. Testing the result of TextBlob model

#### a. Positive

(i) Input: “Recently I bought my first branded shoes from NIKE and it's really amazing, good looking, and very comfortable”.

(ii) Case description: The expected result should be positive since there are many positive words in the sentence like “amazing”, “good looking” or “comfortable”.

(iii) Output:  $f(x) = 0.414 > 0.1$ , hence, it is positive

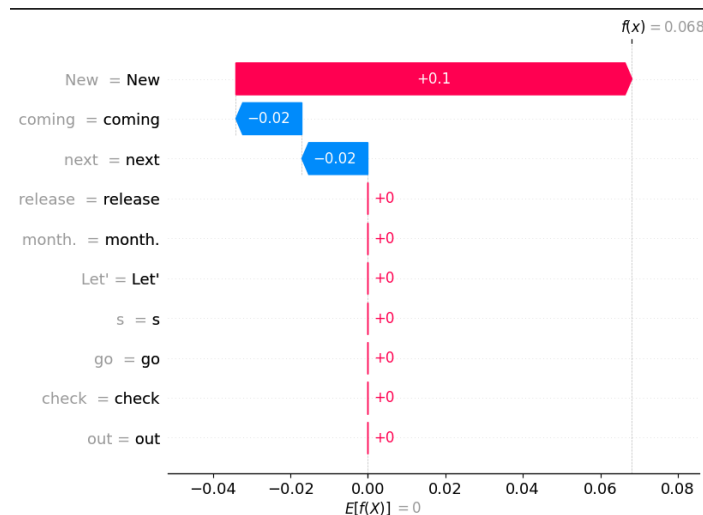


#### b. Neutral

(i) Input: “New release coming next month. Let's go check out”

(ii) Case description: This is an announcement for the new release.

(iii) Output:  $f(x) = -0.1 < 0.068 < 0.1$ , hence, it is neutral.

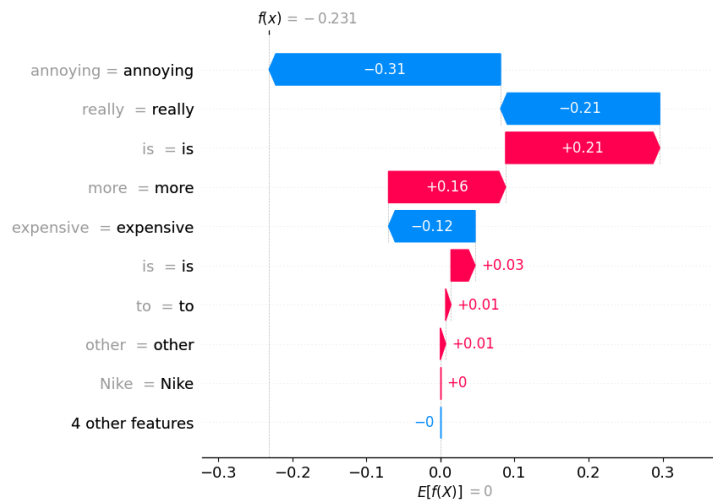


### c. Negative

(i) Input: “Nike shoes is more expensive compared to other brands which is really annoying”.

(ii) Case description: The expected result should be negative since it has some negative words like “expensive” or “annoying”.

(iii) Output:  $f(x) = -0.231 < -0.1$ , hence, it is negative.



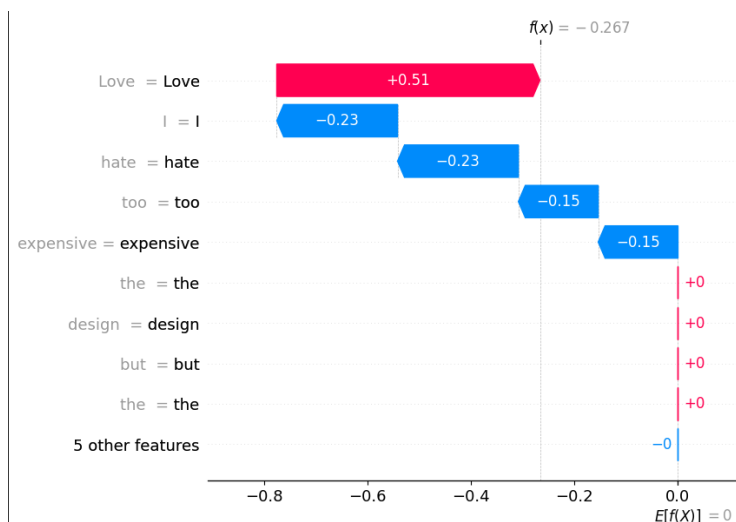
## 2. Testing the reliability of TextBlob Model

### a. Emphasize negative words

(i) Input: “Love the design but I hate the fact that the price is too expensive”.

(ii) Case description: The expected result should be negative since the strongly negative word "hate" after “but” is emphasized.

(iii) Output:  $f(x) = -0.267 < -0.1$ , hence, it is negative

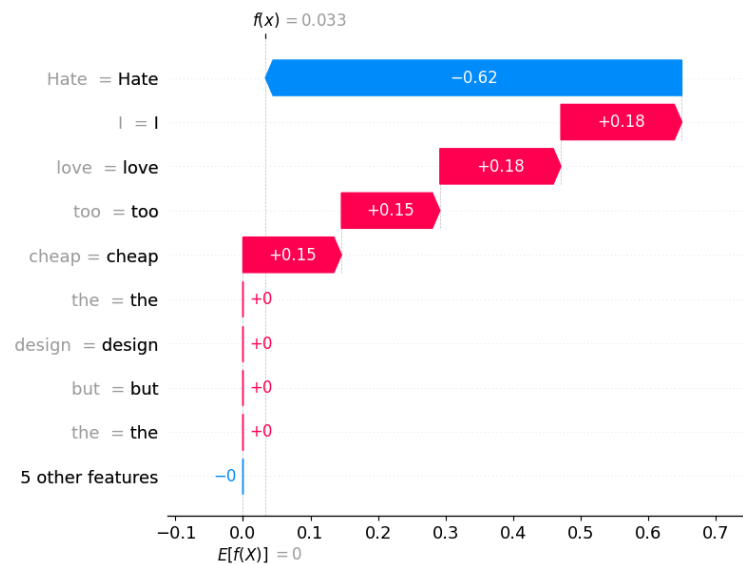


## b. Emphasize positive words

(i) Input: “Hate the design but I love the fact that the price is too cheap”

(ii) Case description: The expected result should be positive since the strongly positive word "love" after “but” is emphasized.

(iii) Output:  $f(x) = -0.1 < 0.033 < 0.1$ , the verbatim turns out to neutral. Hence, TextBlob does not efficiently do sentimental analysis in all cases. Better model should be improved and finalized to get correct results.



## 5. Conclusion

### 5.1. Generalizability

In the project above, we have done data crawling using Twitter API - Tweepy and sentiment analysis by several techniques, in which TextBlob performed the best. Generalizing from this research, an approach to other apparel brands or other industries could be done. However, the generalizability of sentiment analysis among apparel companies is affected by several factors, including the training data used, the features chosen, and the classification method used.

If the sentiment analysis model is trained on a big and diversified text dataset containing a vast variety of clothing brands, the model's generalizability should be fairly good. Nevertheless, if the training dataset is biased towards certain brands or types of clothing, the accuracy and generalizability of the sentiment analysis model may suffer.

Another key consideration is the context in which the sentiment analysis is carried out. The mood portrayed in social media posts or product evaluations about clothing labels may be different from that represented in fashion publications or commercials.

It's also worth mentioning that different clothes brands may have distinct customer demographics, brand values, and product offerings, which might influence how people feel about them. Sentiment towards sporty brands like Nike or Adidas, for example, may differ from opinions of luxury apparel brands, such as Louis Vuitton or Gucci.

To ensure the quality and generalizability of sentiment analysis on apparel brands, it is critical to employ a broad dataset and evaluate the context and features of each brand.

### 5.2 Conclusion and Future Work

Based on the visualisation and sentiment analysis as shown in the code, we can make some conclusions as below:

- People are talking about Nike compared to Addidas on Twitter within the 14-day timeframe.
- The top things mentioned in tweet posts are about various kinds of shoes of the two brands and their e-commerce platforms. Hence, the marketer can use this data to plan their strategy in doing the marketing campaigns.
- Most of the verbatim are judgement statements; not many people express their opinions towards the two brands, especially Adidas
- TextBlob is comparatively good but does not perfectly yield the right results in many cases.

In the future, we will generalize our sentimental analysis and dashboard for other apparel brands. The application program would assist marketers and strategic planners in deciding on their marketing campaign and understanding customer demand. To that end, a better sentiment model like BERT or DIETClassifier would be researched and developed instead of using a pre-defined TextBlob model to boost the accuracy of our prediction.