# Exploring Starbuck’s Social Media Data through Text Mining: A Study of Customer Feedback and Product Perception

Jessica Feto

Word Count: 4115

## Table of Contents:

1. Introduction

2. Company Background

3. Data Collection
    3.1. Collecting Data from Twitter
    3.2. Collecting data from Reddit
4. Data Preparation
5. Data Pre-Processing
    5.1. Step 1: Install Required Packages
    5.2. Step 2: Import Required Libraries
    5.3. Step 3: Download NLTK Resources
    5.4. Step 4: Load Data from CSV Files
    5.5. Step 5: Define Regular Expressions
    5.6. Step 6: Define Functions for Text Preprocessing
    5.7. Step 7: Preprocess Text
    5.8. Step 8: Remove Stopwords
    5.9. Step 9: Save Preprocessed Data

6. Exploratory Data Analysis
    6.1. Summary Statistics
    6.2. Visualizing the Correlation between Post Scores and Number of Comments
    6.3. Visualizing the Distribution of Reddit Post Scores
    6.4. Word Cloud

7. Topic Modelling
    7.1 Introduction
    7.2 Installing Required Libraries
    7.3 Loading Data from CSV Files
    7.4 Preprocessing the Data
    7.5 Training the LDA Model
    7.6 Examining the Topics

8. Sentiment Analysis
    8.1 Introduction
    8.2 Importing Necessary Libraries and Loading Data
    8.3 Adding Sentiment Scores
    8.4 Merging the Data Sets and Combining Text Columns
    8.5 Classifying Sentiment and Calculating its Percentage
    8.6 Visualising the Merged Dataset

6. Conclusion

7. References

## 1. Introduction

Text mining has become a potent tool for big businesses in today's data-driven world to glean insights and meaning from enormous amounts of unstructured data. The process of examining and drawing conclusions from written or spoken communication is called text mining, also referred to as text analytics. Natural language processing, machine learning, statistics, and data visualisation are just a few of the methods used. Text mining has become an essential tool for businesses like Starbucks to understand customer feedback and product perception. Starbucks can gain important insights into how customers feel about their goods and services by analysing social media data through text mining. These insights can guide product development, marketing plans, and customer service initiatives.

This study focuses on Starbucks' use of text mining to study consumer opinions and brand perception. In this article, we examine the procedures for gathering, preparing, pre-processing, exploratory data analysis, topic modelling with LDA, and sentiment analysis. My objective is to highlight the benefits of text mining for organisations like Starbucks and the knowledge that can be gained from social media data. To learn more about how customers view Starbucks' goods and services, I specifically examine customer comments on Twitter, Reddit, and other social media platforms. Insights that Starbucks and other businesses can use to raise customer satisfaction and promote expansion are what I aim to deliver through my analysis.

## 2. Company Background

One of the most well-known coffee shop chains in the world, Starbucks is renowned for its superior coffee and top-notch customer support. Starbucks was established in Seattle in 1971 and has since expanded to thousands of locations across more than 80 nations.
Starbucks employed more than 346,000 people globally as of 2021, serving millions of customers daily. With a significant presence in the UK, the company has expanded its operations to a number of continents, including Asia, Europe, the Middle East, and Africa.

Starbucks operates around 1,000 shops in the United Kingdom, making it one of the country's major coffee companies. In the United Kingdom, the company employs over 16,000 people and provides a variety of products and services such as coffee, tea, baked goods, sandwiches, and other culinary items.
Starbucks has pledged to provide high-quality coffee while adhering to ethical and sustainable practices. The corporation has established objectives to reduce its environmental footprint and promote social responsibility through initiatives such as the "Starbucks Foundation" and "C.A.F.E. Practises."

Despite the economic uncertainty posed by Brexit, Starbucks UK has demonstrated remarkable financial performance in recent years, with revenue growth of 9.2% in 2019. The company's exceptional financial performance is a result of its strong brand, smart marketing methods, and excellent customer service. Starbucks has established itself as a leading player in the coffee industry, with a global presence and a commitment to sustainability and social responsibility.

## 3. Data Collection

Data collection from social media sites like Twitter and Reddit may be a great tool for businesses looking to obtain insights into customer opinions and preferences. I created developer accounts and received the appropriate API keys to collect data from various platforms. This enabled me to use Python scripts to connect to the platforms' APIs and collect data. To extract relevant insights, the acquired data were processed and analysed using various text mining algorithms. My research aims to provide insights into customer feedback and product impression for Starbucks by using the power of social media data.

## 3.1. Collecting Data from Twitter

The first step in collecting data from Twitter is to install the Tweepy library, which provides a Python wrapper for the Twitter API and the I needed to set up the API keys.

```
api_key = 'vLQF[redacted]
api_key_secret = '41epOON0phcRVeh4x01sF[redacted]
bearer_token = 'AAAAAAAAAAAAAAAAAA[redacted]
access_token = '1645799525670617090-3ppy[redacted]
access_token_secret = 'hXJgbvaJjx[redacted]
```

After setting up the API credentials, you needed to authenticate and set up the API with Tweepy.

```
# authenticate and set up API
auth = tweepy.OAuthHandler(api_key, api_key_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)
```

Before collecting tweets, I had to define the search parameters. I decied to search for tweets that contain any of the following keywords: StarbucksCoffee, latte, frappuccino, icedcoffee, pumpkinspice, and cappuccino. I limited the search to tweets from within a 300km radius of London, UK, and I collected a maximum of 1000 tweets.

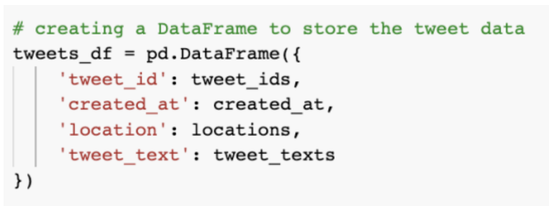
```
# defining the search parameters
query = 'StarbucksCoffee OR latte OR frappuccino OR icedcoffee OR pumpkinspice OR cappuccino'
num_tweets = 1000  # maximum number of tweets to collect
latitude = 51.5074  # latitude of the location to search
longitude = -0.1278  # longitude of the location to search
radius = '300km'  # radius to search
```

Now that I defined the search parameters, I was able to collect tweets using the tweepy.Cursor() method. This method allowed me to iterate through pages of results returned by the Twitter API. I use the search_tweets method to search for tweets that match the query and are within the specified radius of the given latitude and longitude. I also specified that I want to collect the full text of each tweet.

```
# creating empty lists to store tweet data
tweet_ids = []
created_at = []
locations = []
tweet_texts = []
```

```
# looping through the tweet objects
for tweet in tweets:
    # extracting tweet data
    tweet_id = tweet.id  # Get tweet ID
    tweet_time = tweet.created_at
    location = tweet.user.location
    tweet_text = tweet.full_text

    # append tweet data to lists
    tweet_ids.append(tweet_id)
    created_at.append(tweet_time)
    locations.append(location)
    tweet_texts.append(tweet_text)
```

Once I collected the tweets, I extracted the and store it in lists and then I stored the tweet data in a Pandas data frame and export it to a CSV file using this code:



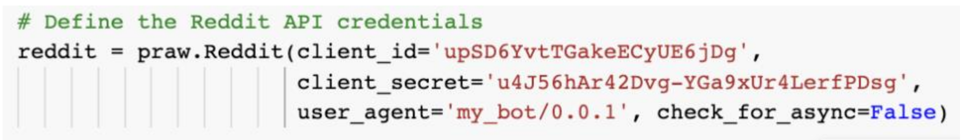
```
# creating a DataFrame to store the tweet data
tweets_df = pd.DataFrame({
    'tweet_id': tweet_ids,
    'created_at': created_at,
    'location': locations,
    'tweet_text': tweet_texts
})
```

```
tweets_df.to_csv('twitterDataCollection.csv', index=False)
print(f"Collected {len(tweets_df)} tweets about Starbucks and saved data to 'twitterDataCollection.csv' file.")
```

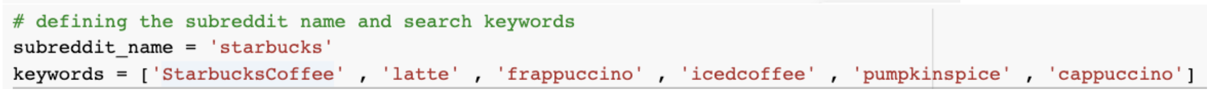
## 3.2. Collecting data from Reddit

In order to collect data from Reddit, I had to create a Reddit developer account and obtain the keys.Then I installed the PRAW library using !pip install praw. Next, I imported the necessary libraries: praw, pandas, datetime.

The first step was to define my Reddit API credentials and I defined the subreddit I wanted do collect data from and the search keywords. I decided to collect data from the Starbucks Reddit an search for posts containing the keywords: "StarbucksCoffee," "latte," "frappuccino," "icedcoffee," "pumpkinspice," and "cappuccino,"

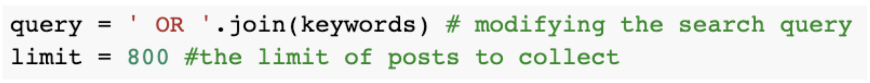


```
# Define the Reddit API credentials
reddit = praw.Reddit(client_id='upSD6YvtTGakeECyUE6jDg',
                     client_secret='u4J56hAr42Dvg-YGa9xUr4LerfPDsg',
                     user_agent='my_bot/0.0.1', check_for_async=False)
```



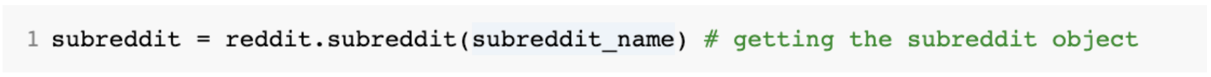
```
# defining the subreddit name and search keywords
subreddit_name = 'starbucks'
keywords = ['StarbucksCoffee' , 'latte' , 'frappuccino' , 'icedcoffee' , 'pumpkinspice' , 'cappuccino']
```

Then I used the OR operator to modify the search query in order to include multiple operators and to set a limit for the number of posts to collect.

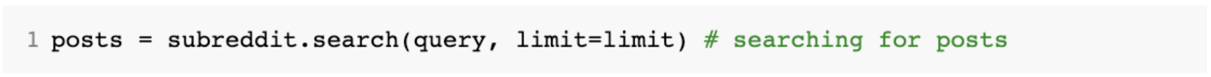


```
query = ' OR '.join(keywords) # modifying the search query
limit = 800 #the limit of posts to collect
```

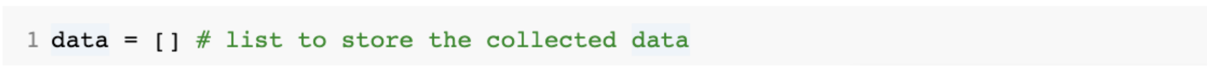
The next step I followed was to initialise as a **'subreddit'** object by passing the name of the subreddit as a parameter of the **'subreddit ()'** method. Then I used the **'search ()**' method to search for posts in the specified subreddit that contain the keywords defined in the 'query' variable. The **'limit'** parameter specifies the maximum number of posts to be returned and the results are stored in the 'posts' variable. I also needed to initialise an empty list called 'data' to store the collected data.  See the code below:



```
subreddit = reddit.subreddit(subreddit_name) # getting the subreddit object
```



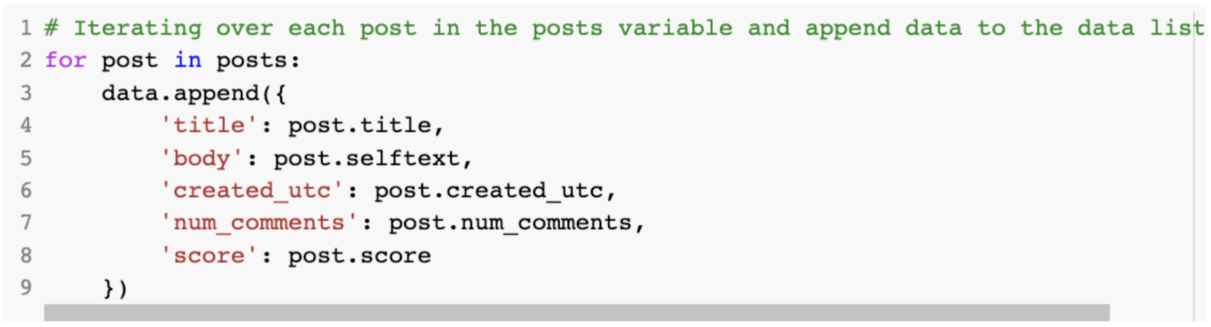
```
posts = subreddit.search(query, limit=limit) # searching for posts
```



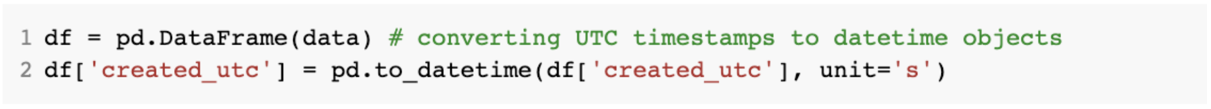
```
data = [] # list to store the collected data
```

After collecting the information,I stored the post's title, post body, creation time in UTC, the number of comments and score.  All this relevant information is been added to the 'data' list and it is used to create a Pandas data frame object which then I will use to manipulate and analyse the collected data.

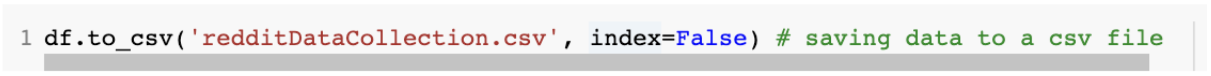
I had to convert the '**created_utc**' column of the data frame from Unix format to human-readable format using the **'pd.to_datetime ()'** method. And finally, the 'to_csv ()' method is used to save the collected data in a CSV file which I named '**redditDataCollection.csv**'



```
# Iterating over each post in the posts variable and append data to the data list
for post in posts:
    data.append({
        'title': post.title,
        'body': post.selftext,
        'created_utc': post.created_utc,
        'num_comments': post.num_comments,
        'score': post.score
    })
```



```
df = pd.DataFrame(data) # converting UTC timestamps to datetime objects
df['created_utc'] = pd.to_datetime(df['created_utc'], unit='s')
```



```
df.to_csv('redditDataCollection.csv', index=False) # saving data to a csv file
```

Overall, careful planning and execution are needed when gathering data from social media sites like Twitter and Reddit. It is possible to get comprehensive and insightful data that can provide insight into user behaviour and preferences by specifying search parameters that are precise and pertinent to the current research issue and by accessing the APIs with the help of strong libraries like **Tweepy** and **PRAW**. This data may be converted into useful insights that can spur innovation and enhance decision-making across a wide range of businesses with the appropriate analytical tools and methodologies.

## 4. Data Preparation

Data preparation, which comprises transforming and cleaning the gathered data so that it is ready for analysis, is an additional critical phase in every data analysis project in addition to the data gathering process. In this step, duplicates may be eliminated, missing values may be handled, data types may be transformed, and statistical techniques may be used to locate and manage outliers and anomalies in the data. Any analysis may be compromised in terms of accuracy and reliability without proper data preparation, which could result in incorrect findings and decisions. To guarantee that the obtained data is properly prepared and available for analysis, it is crucial to allot enough time and resources.

I started the data preparation process by importing the necessary libraries, then I loaded the data that had been previously gathered during the data collecting phase and was stored in CSV files.

```python
import pandas as pd
import numpy as np
from datetime import datetime, timedelta
import langdetect
from langdetect import detect
```

```python
# loading the data from CSV files
reddit_data = pd.read_csv('redditDataCollectionBBV.csv')
twitter_data = pd.read_csv('twitterDataCollectionBBV.csv')
```

I added a new column to the Twitter data frame as part of the data preparation process to capture the length of each tweet. The 'tweet_text' column length for each tweet was determined using the 'apply()' method and the 'len' function.
Additionally, I used the 'pd.to_datetime()' method to convert the 'created_utc' column of the Reddit data frame and the 'created_at' column of the Twitter data frame from Unix format to human-readable format. This was required to make data handling and analysis simple.
These procedures are crucial for making sure the data is correctly structured and prepared for additional analysis.

```python
# creating a new column with the length of each tweet
twitter_data['tweet_length'] = twitter_data['tweet_text'].apply(len)
```

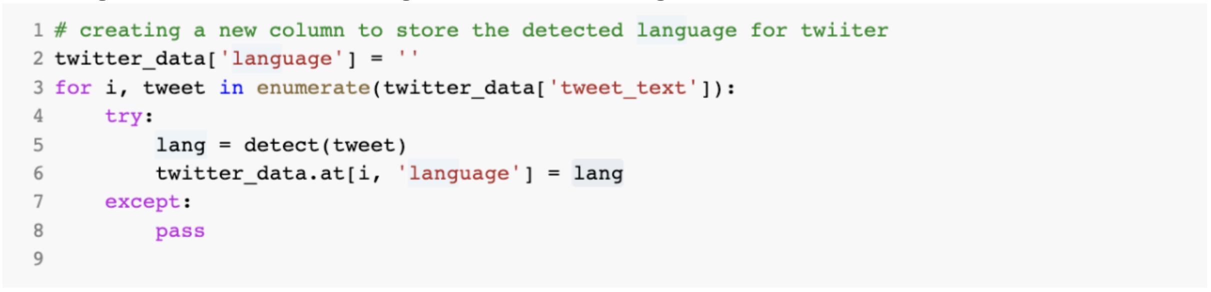
```python
# converting data from unix format to human readable
reddit_data['created_utc'] = pd.to_datetime(reddit_data['created_utc'])
twitter_data['created_at'] = pd.to_datetime(twitter_data['created_at'])
```

Then, using the created_at and created_utc columns of the Reddit and Twitter data, respectively, I filter the data in each data frame to only include data that is older than seven days. This makes sure that the focus of our analysis is on current facts. Reddit_data and twitter_data1 are new variables created to hold the filtered data for each platform.

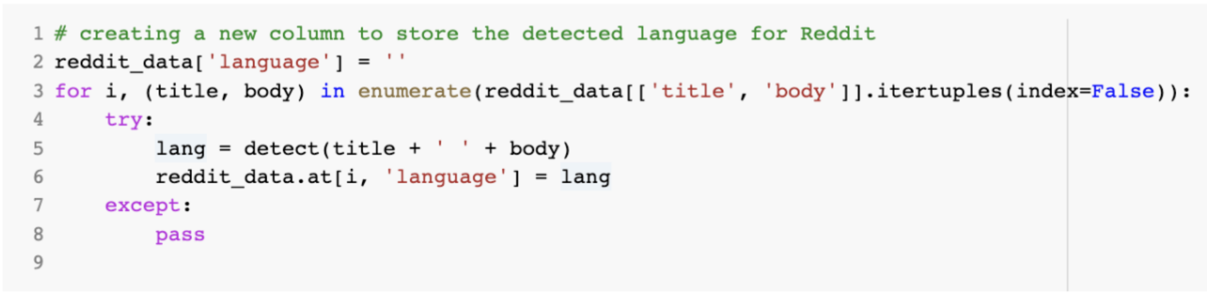
```python
# Find the date 7 days ago
end_date = datetime.today().date()
start_date = end_date - timedelta(days=7)
```

```python
# Filter the data in each dataframe
reddit_data = reddit_data.loc[(reddit_data['created_utc'].dt.date
                                   >= start_date) & (reddit_data['created_utc'].dt.date
                                                         <= end_date)]

twitter_data = twitter_data.loc[(twitter_data['created_at'].dt.date
                                    >= start_date) & (twitter_data['created_at'].dt.date
                                                          <= end_date)]
```

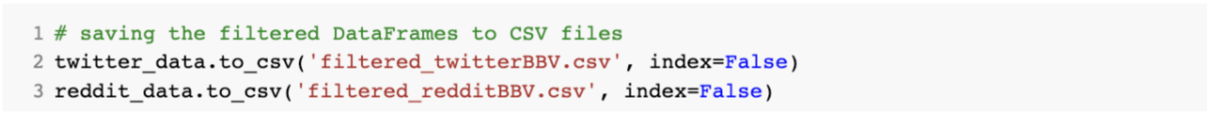
It's crucial to organise the posts according to language. Therefore, I made the decision to add a new column to the Twitter and Reddit data frames called "**language**”. The detect function from the **langdetect** package is used to determine the language of every tweet or post. The language is determined for each tweet and placed in the 'language' column of the corresponding row in the data frame as the code iterates over the **'tweet_text'** column for Twitter. The code detects the language and stores it in the 'language' column of the associated row for Reddit by iterating over the 'title' and 'body' columns, concatenating them, and iterating over them again. The filtered data frames are then saved to CSV files using the **to_csv** method, omitting the index column using the index=False argument.



```python
# creating a new column to store the detected language for twiiter
twitter_data['language'] = ''
for i, tweet in enumerate(twitter_data['tweet_text']):
    try:
        lang = detect(tweet)
        twitter_data.at[i, 'language'] = lang
    except:
        pass

```



```python
# creating a new column to store the detected language for Reddit
reddit_data['language'] = ''
for i, (title, body) in enumerate(reddit_data[['title', 'body']].itertuples(index=False)):
    try:
        lang = detect(title + ' ' + body)
        reddit_data.at[i, 'language'] = lang
    except:
        pass

```



```python
# saving the filtered DataFrames to CSV files
twitter_data.to_csv('filtered_twitterBBV.csv', index=False)
reddit_data.to_csv('filtered_redditBBV.csv', index=False)
```

## 5. Data Pre-Processing

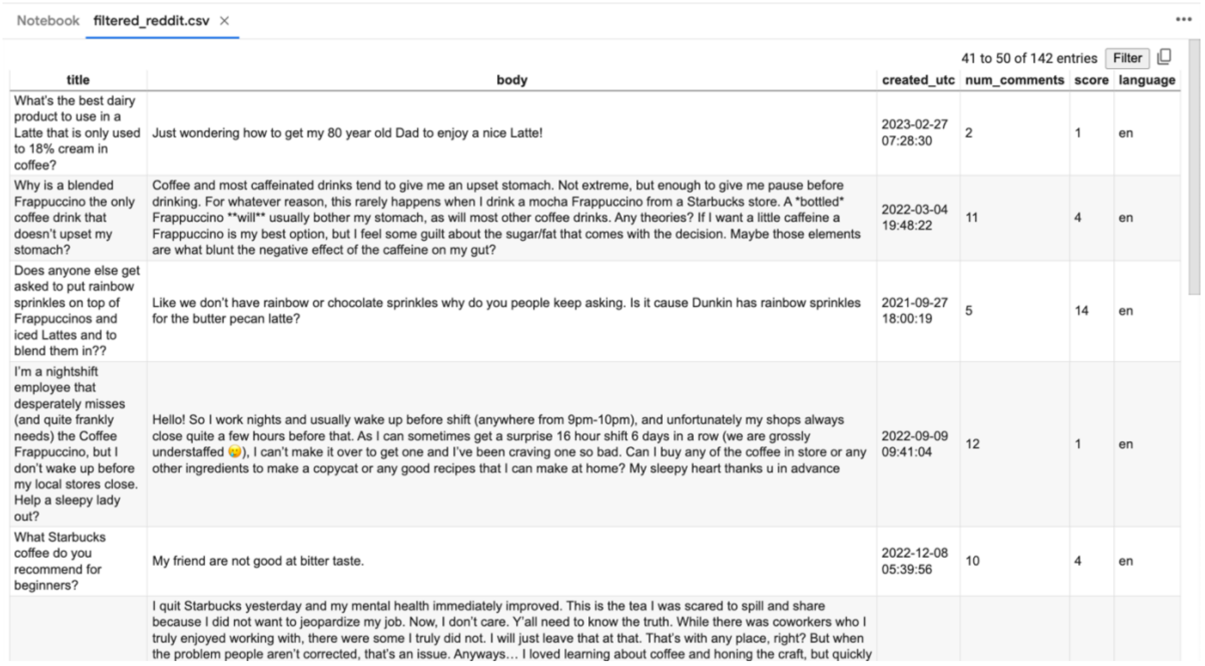
Any data analysis pipeline must include pre-processing because it helps to clean, transform, and prepare data for additional analysis. Before conducting further analysis, it is frequently necessary to rectify discrepancies, mistakes, or missing values in the data collected from various sources.

Data cleaning, which involves finding and fixing flaws or inconsistencies in the data, is a frequent pre-processing operation. For instance, before analysing text data, it is typical to remove punctuation, stop words, and other noise from the text. A number of methods, including regular expressions, data wrangling, and text pre-processing packages, can be used to clean up data.

Data transformation, which entails changing data into a format better suited for analysis, is another significant preprocessing operation. This might entail transforming data kinds, scaling numbers, or encoding variables with categories. A number of tools, including as the pandas, numpy, and scikit-learn packages, can be used to alter data.

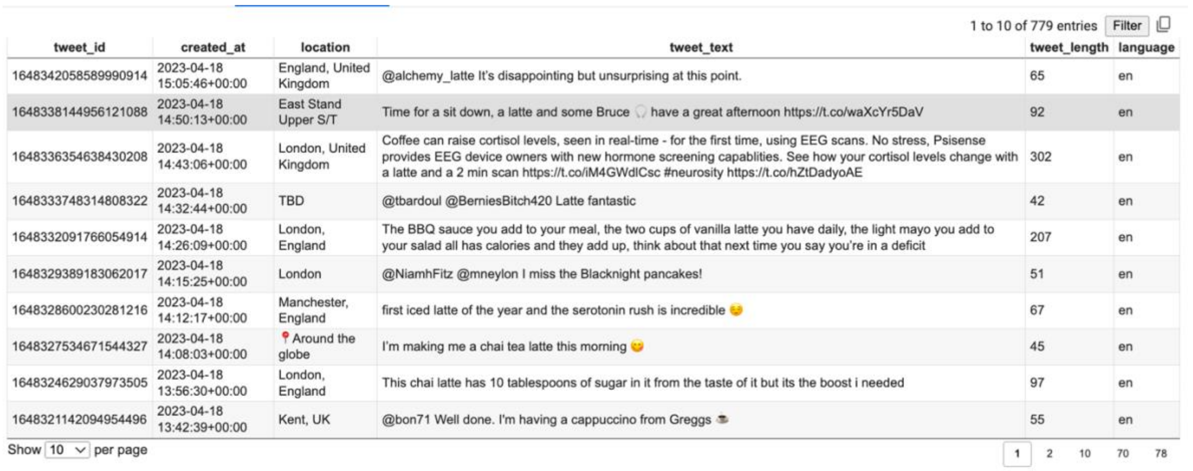
Handling missing values or anomalies in the data is another aspect of pre-processing. Depending on the analytic objectives and the amount of missing data, missing values can either be imputed or deleted. To prevent bias in the analysis, outliers can be found using statistical approaches and addressed by being removed or adjusted.

Below, I will explain the steps I followed to pre-process the data collected from Reddit and Twitter.



| title | body | created_utc | num_comments | score | language |
|---|---|---|---|---|---|
| What's the best dairy product to use in a Latte that is only used to 18% cream in coffee? | Just wondering how to get my 80 year old Dad to enjoy a nice Latte! | 2023-02-27 07:28:30 | 2 | 1 | en |
| Why is a blended Frappuccino the only coffee drink that doesn't upset my stomach? | Coffee and most caffeinated drinks tend to give me an upset stomach. Not extreme, but enough to give me pause before drinking. For whatever reason, this rarely happens when I drink a mocha Frappuccino from a Starbucks store. A *bottled* Frappuccino **will** usually bother my stomach, as will most other coffee drinks. Any theories? If I want a little caffeine a Frappuccino is my best option, but I feel some guilt about the sugar/fat that comes with the decision. Maybe those elements are what blunt the negative effect of the caffeine on my gut? | 2022-03-04 19:48:22 | 11 | 4 | en |
| Does anyone else get asked to put rainbow sprinkles on top of Frappuccinos and iced Lattes and to blend them in?? | Like we don't have rainbow or chocolate sprinkles why do you people keep asking. Is it cause Dunkin has rainbow sprinkles for the butter pecan latte? | 2021-09-27 18:00:19 | 5 | 14 | en |
| I'm a nightshift employee that desperately misses (and quite frankly needs) the Coffee Frappuccino, but I don't wake up before my local stores close. Help a sleepy lady out? | Hello! So I work nights and usually wake up before shift (anywhere from 9pm-10pm), and unfortunately my shops always close quite a few hours before that. As I can sometimes get a surprise 16 hour shift 6 days in a row (we are grossly understaffed 😢), I can't make it over to get one and I've been craving one so bad. Can I buy any of the coffee in store or any other ingredients to make a copycat or any good recipes that I can make at home? My sleepy heart thanks u in advance | 2022-09-09 09:41:04 | 12 | 1 | en |
| What Starbucks coffee do you recommend for beginners? | My friend are not good at bitter taste. | 2022-12-08 05:39:56 | 10 | 4 | en |
| | I quit Starbucks yesterday and my mental health immediately improved. This is the tea I was scared to spill and share because I did not want to jeopardize my job. Now, I don't care. Y'all need to know the truth. While there was coworkers who I truly enjoyed working with, there were some I truly did not. I will just leave that at that. That's with any place, right? But when the problem people aren't corrected, that's an issue. Anyways… I loved learning about coffee and honing the craft, but quickly | | | | |

*Figure 1. The reddit data set before being pre-processed*



| tweet_id | created_at | location | tweet_text | tweet_length | language |
|---|---|---|---|---|---|
| 1648342058589990914 | 2023-04-18 15:05:46+00:00 | England, United Kingdom | @alchemy_latte It's disappointing but unsurprising at this point. | 65 | en |
| 1648338144956121088 | 2023-04-18 14:50:13+00:00 | East Stand Upper S/T | Time for a sit down, a latte and some Bruce 🎧 have a great afternoon https://t.co/waXcYr5DaV | 92 | en |
| 1648336354638430208 | 2023-04-18 14:43:06+00:00 | London, United Kingdom | Coffee can raise cortisol levels, seen in real-time - for the first time, using EEG scans. No stress, Psisense provides EEG device owners with new hormone screening capablities. See how your cortisol levels change with a latte and a 2 min scan https://t.co/iM4GWdlCsc #neurosity https://t.co/hZtDadyoAE | 302 | en |
| 1648333748314808322 | 2023-04-18 14:32:44+00:00 | TBD | @tbardoul @BerniesBitch420 Latte fantastic | 42 | en |
| 1648332091766054914 | 2023-04-18 14:26:09+00:00 | London, England | The BBQ sauce you add to your meal, the two cups of vanilla latte you have daily, the light mayo you add to your salad all has calories and they add up, think about that next time you say you're in a deficit | 207 | en |
| 1648329389183062017 | 2023-04-18 14:15:25+00:00 | London | @NiamhFitz @mneylon I miss the Blacknight pancakes! | 51 | en |
| 1648328600230281216 | 2023-04-18 14:12:17+00:00 | Manchester, England | first iced latte of the year and the serotonin rush is incredible 😌 | 67 | en |
| 1648327534671544327 | 2023-04-18 14:08:03+00:00 | 📍Around the globe | I'm making me a chai tea latte this morning 😋 | 45 | en |
| 1648324629037973505 | 2023-04-18 13:56:30+00:00 | London, England | This chai latte has 10 tablespoons of sugar in it from the taste of it but its the boost i needed | 97 | en |
| 1648321142094954496 | 2023-04-18 13:42:39+00:00 | Kent, UK | @bon71 Well done. I'm having a cappuccino from Greggs ☕ | 55 | en |

*Figure 2. The twitter data set before being pre-processed*

**Step 1:**

Using the pip command, I installed the required packages such as 'nltk', 'emoji' and 'contractions'.

```
!pip install nltk
!pip install emoji
!pip install -U contractions
!pip install --upgrade emoji
```

Step 2:
Next, I imported the required libraries such as pandas, regex, functools, TextBlob, stopwords and WordNetLemmatizer.

```python
import pandas as pd
import re
import contractions
import regex
import functools
import nltk
from textblob import TextBlob
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
```

Step 3:
I had to download the NLTK resources such as WordNet, stopwords, omw-1.4 and punkt. Those resources are needed for text pre-processing.

```python
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('omw-1.4')
nltk.download('punkt')
```

Step 4:
The following step is to load the data from the CSV files that were previously filtered and stored in 'filtered_reddit.csv' and 'filtered_twitter.csv' files.

```python
# loading the prepared data from CSV files
filtered_reddit_data = pd.read_csv('filtered_reddit.csv')
filtered_twitter_data = pd.read_csv('filtered_twitter.csv')
```

Step 5:
I defined regular expressions to remove emojis and links from the text.

```python
# defining regular expressions to remove emojis and links
emoji_pattern = re.compile("["
          u"\U0001F600-\U0001F64F"  # emoticons
          u"\U0001F300-\U0001F5FF"  # symbols & pictographs
          u"\U0001F680-\U0001F6FF"  # transport & map symbols
          u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                              "]+", flags=re.UNICODE)
```

Step 6:
In this step, I defined functions for text preprocessing. The first function, 'apply_contractions' uses the 'contractions' package to expand contractions in the text. And then I applied the contractions function to 'title', body' and 'tweet_text' columns.

```python
def apply_contractions(text):
    if isinstance(text, str):
        print("Original text:", text)
        fixed_text = contractions.fix(text)
        print("Fixed text:", fixed_text)
        return fixed_text
    else:
        return ''
```

```python
filtered_reddit_data['title'] = filtered_reddit_data['title'].apply(apply_contractions)
filtered_reddit_data['body'] = filtered_reddit_data['body'].apply(apply_contractions)
filtered_twitter_data['tweet_text'] = filtered_twitter_data['tweet_text'].apply(apply_contractions)
```

The second function, 'clean_text' removes extra whitespaces, converts text to lowercase, removes emojis and links, and removes certain characters such as dots, apostrophes, commas, parentheses, and question marks. Then I applied the clean_text function to ‘title’, body’ and ‘tweet_text’ columns.

```python
def clean_text(text):
    if not isinstance(text,str):
        return ''
    text = re.sub(r'\s+', ' ', text) # remove extra whitespaces
    text = text.lower() # convert to lowercase
    text = regex.sub('[\p{Emoji}]', '', text) # remove emojis
    text = re.sub(r'http\S+', '', text) # remove links
    text = re.sub(r'www\S+', '', text)
    text = re.sub(r'\S+\.com\S*', '', text)
    text = re.sub(r'\S+\.org\S*', '', text)
    text = re.sub(r'\S+\.edu\S*', '', text)
    text = re.sub(r'\S+\.gov\S*', '', text)
    text = re.sub(r'\.', '', text)
    text = re.sub(r'\'', '', text)
    text = re.sub(r'\,', '', text)
    text = re.sub(r'\(', '', text)
    text = re.sub(r'\)', '', text)
    text = re.sub(r'\?', '', text)

    print("Cleaned text:",text)
    return text.strip()
```

```python
filtered_reddit_data['title'] = filtered_reddit_data['title'].apply(clean_text)
filtered_reddit_data['body'] = filtered_reddit_data['body'].apply(clean_text)
filtered_twitter_data['tweet_text'] = filtered_twitter_data['tweet_text'].apply(clean_text)
```

## Step 7:

Here, I defined another function called “preprocess_text”, which preprocesses the text by replacing URLs with a marker, it corrects spelling, it tokenised the text, removes stop words and it lematizes the tokens. I also applied the preprocessing functions to the 'title', 'body', and 'tweet_text' columns of the filtered data.

```python
# defining a function for preprocessing
def preprocess_text(text):
    text = re.sub(r"http\S+", "URLMARKER", text) # replacing URLs with a marker
    blob = TextBlob(text)    # correcting spelling
    corrected_text = str(blob.correct())
    tokens = nltk.word_tokenize(text.lower())
    stopwords_list = stopwords.words('english')
    tokens = [token for token in tokens if token not in stopwords_list]
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(token) for token in tokens]
    print("Processed tokens:", tokens)
    return tokens
```

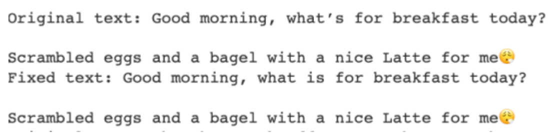
```python
# applying function to title, body, twwet_text columns of reddit and twitter
filtered_reddit_data['title'] = filtered_reddit_data['title'].apply(preprocess_text)
filtered_reddit_data['body'] = filtered_reddit_data['body'].apply(preprocess_text)
filtered_twitter_data['tweet_text'] = filtered_twitter_data['tweet_text'].apply(preprocess_text)
```

## Step 9:

Last but not the least, I saved the pre-processed data as CSV files named: 'preprocessed_reddit_data.csv' and 'preprocessed_twitter_data.csv'.

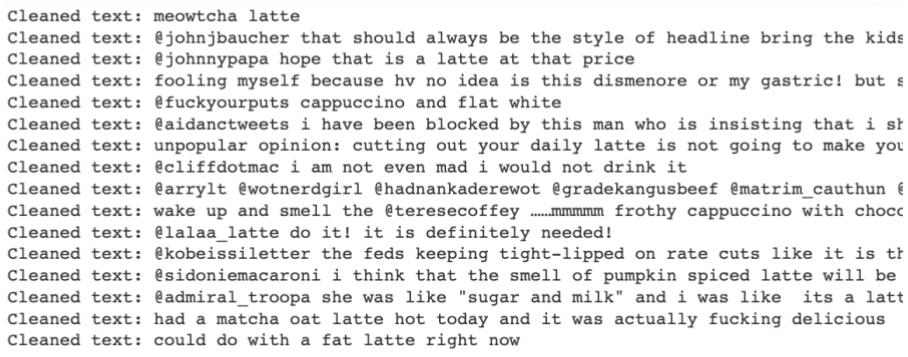
```python
filtered_reddit_data.to_csv('preprocessed_reddit_data.csv', index=False)
filtered_twitter_data.to_csv('preprocessed_twitter_data.csv', index=False)
```

After applying preprocessing steps, this is how the dataset looks like:



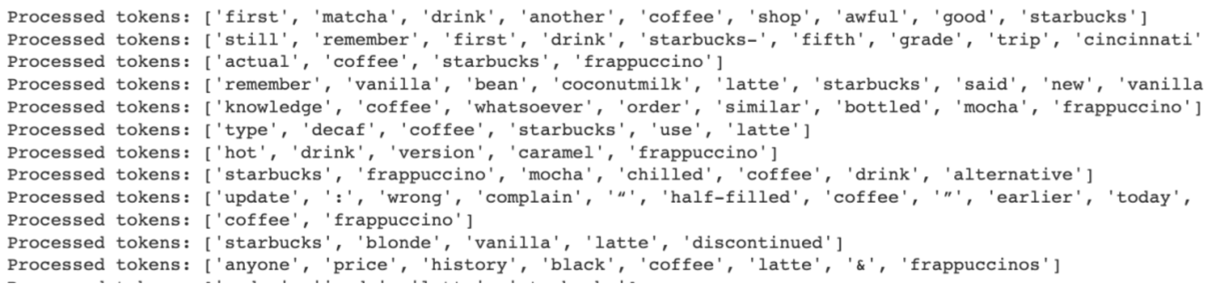
```
Original text: Good morning, what's for breakfast today?

Scrambled eggs and a bagel with a nice Latte for me😮‍💨
Fixed text: Good morning, what is for breakfast today?

Scrambled eggs and a bagel with a nice Latte for me😮‍💨
```

*Figure 3. The data after using regular expressions*



```
Cleaned text: meowtcha latte
Cleaned text: @johnjbaucher that should always be the style of headline bring the kids
Cleaned text: @johnnypapa hope that is a latte at that price
Cleaned text: fooling myself because hv no idea is this dismenore or my gastric! but s
Cleaned text: @fuckyourputs cappuccino and flat white
Cleaned text: @aidanctweets i have been blocked by this man who is insisting that i sh
Cleaned text: unpopular opinion: cutting out your daily latte is not going to make you
Cleaned text: @cliffdotmac i am not even mad i would not drink it
Cleaned text: @arrylt @wotnerdgirl @hadnankaderewot @gradekangusbeef @matrim_cauthun @
Cleaned text: wake up and smell the @teresecoffey ……mmmmm frothy cappuccino with choco
Cleaned text: @lalaa_latte do it! it is definitely needed!
Cleaned text: @kobeissiletter the feds keeping tight-lipped on rate cuts like it is th
Cleaned text: @sidoniemacaroni i think that the smell of pumpkin spiced latte will be
Cleaned text: @admiral_troopa she was like "sugar and milk" and i was like  its a latt
Cleaned text: had a matcha oat latte hot today and it was actually fucking delicious
Cleaned text: could do with a fat latte right now
```

*Figure 4. The data after using the contractions function*



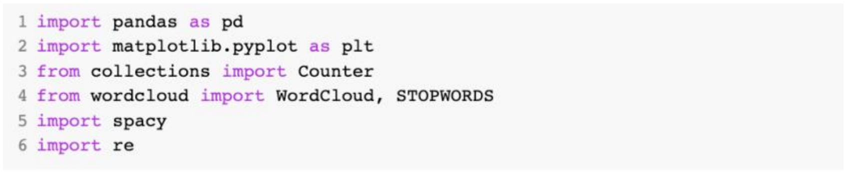
```
Processed tokens: ['first', 'matcha', 'drink', 'another', 'coffee', 'shop', 'awful', 'good', 'starbucks']
Processed tokens: ['still', 'remember', 'first', 'drink', 'starbucks-', 'fifth', 'grade', 'trip', 'cincinnati'
Processed tokens: ['actual', 'coffee', 'starbucks', 'frappuccino']
Processed tokens: ['remember', 'vanilla', 'bean', 'coconutmilk', 'latte', 'starbucks', 'said', 'new', 'vanilla
Processed tokens: ['knowledge', 'coffee', 'whatsoever', 'order', 'similar', 'bottled', 'mocha', 'frappuccino']
Processed tokens: ['type', 'decaf', 'coffee', 'starbucks', 'use', 'latte']
Processed tokens: ['hot', 'drink', 'version', 'caramel', 'frappuccino']
Processed tokens: ['starbucks', 'frappuccino', 'mocha', 'chilled', 'coffee', 'drink', 'alternative']
Processed tokens: ['update', ':', 'wrong', 'complain', '“', 'half-filled', 'coffee', '”', 'earlier', 'today',
Processed tokens: ['coffee', 'frappuccino']
Processed tokens: ['starbucks', 'blonde', 'vanilla', 'latte', 'discontinued']
Processed tokens: ['anyone', 'price', 'history', 'black', 'coffee', 'latte', '&', 'frappuccinos']
```

*Figure 5. The data after applying the pre-processing function*

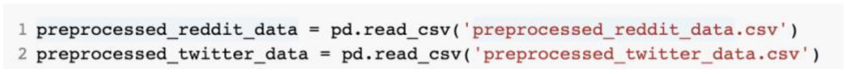
## 6. Exploratory Data Analysis

After preprocessing, exploratory data analysis (EDA) is a crucial stage in data analysis. To comprehend the patterns, correlations, and trends in the data, includes using statistical and visual tools. EDA is carried out on preprocessed Twitter and Reddit data in the given code. The summary statistics and data are presented, followed by visualisations to help comprehend the distribution of post scores and the correlation between scores and the number of comments. The combined text from the two datasets is then examined with regular expressions and word count in order to produce a word cloud, which sheds light on the words that appear most frequently in the data.

Firstly, I imprted the necessary libraries such as pandas, matplotlib, collection, wordcloud and spacy. Then, I loaded the pre-processed CSV files.

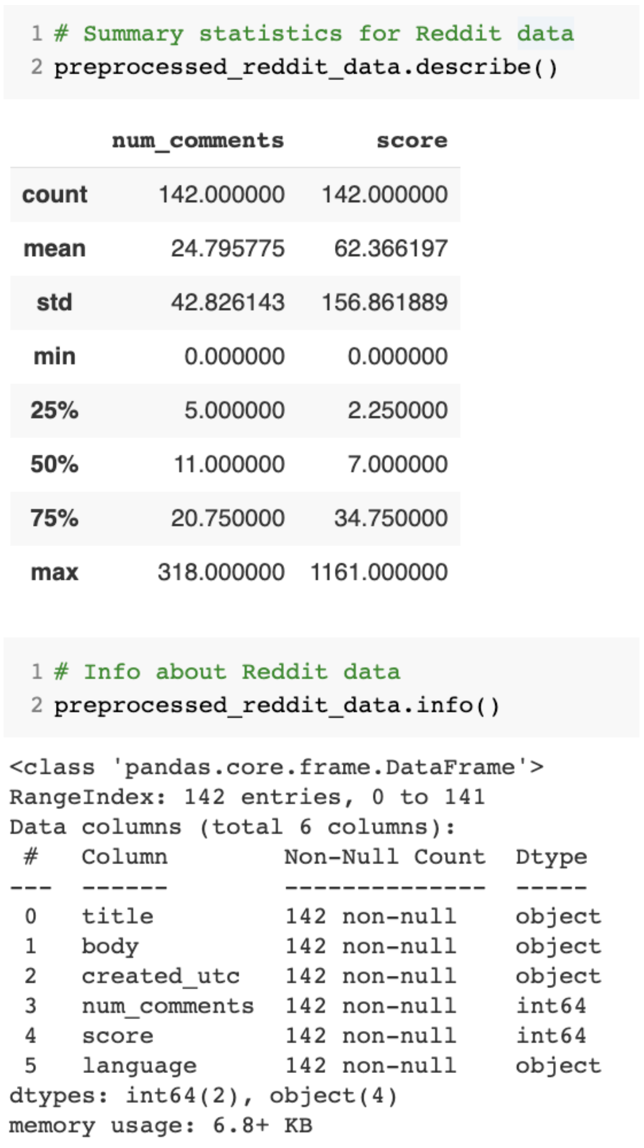


```python
import pandas as pd
import matplotlib.pyplot as plt
from collections import Counter
from wordcloud import WordCloud, STOPWORDS
import spacy
import re
```



```python
preprocessed_reddit_data = pd.read_csv('preprocessed_reddit_data.csv')
preprocessed_twitter_data = pd.read_csv('preprocessed_twitter_data.csv')
```

Next, I performed summary statistics of the preprocessed Reddit and Twitter data. I used the **info ()** function to provide information about the preprocessed Reddit and Twitter data.



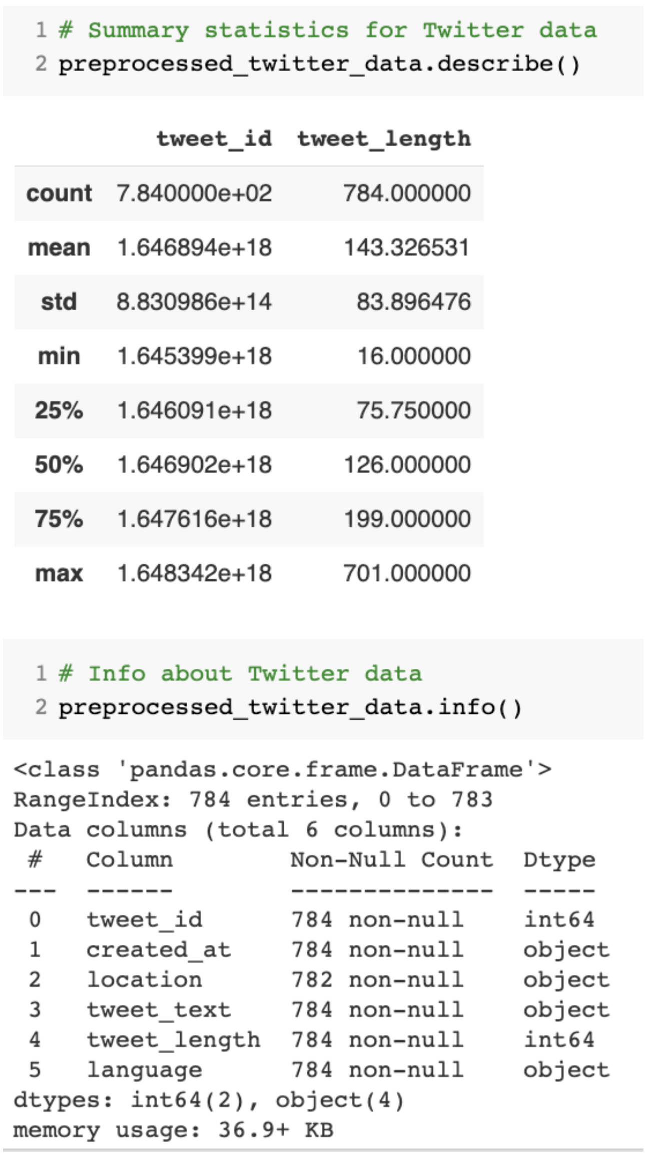
```python
# Summary statistics for Reddit data
preprocessed_reddit_data.describe()
```

| | num_comments | score |
|---|---|---|
| **count** | 142.000000 | 142.000000 |
| **mean** | 24.795775 | 62.366197 |
| **std** | 42.826143 | 156.861889 |
| **min** | 0.000000 | 0.000000 |
| **25%** | 5.000000 | 2.250000 |
| **50%** | 11.000000 | 7.000000 |
| **75%** | 20.750000 | 34.750000 |
| **max** | 318.000000 | 1161.000000 |

```python
# Info about Reddit data
preprocessed_reddit_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142 entries, 0 to 141
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   title         142 non-null    object
 1   body          142 non-null    object
 2   created_utc   142 non-null    object
 3   num_comments  142 non-null    int64
 4   score         142 non-null    int64
 5   language      142 non-null    object
dtypes: int64(2), object(4)
memory usage: 6.8+ KB
```

*Figure 7. Summary Statistics for Reddit Data*



```python
# Summary statistics for Twitter data
preprocessed_twitter_data.describe()
```

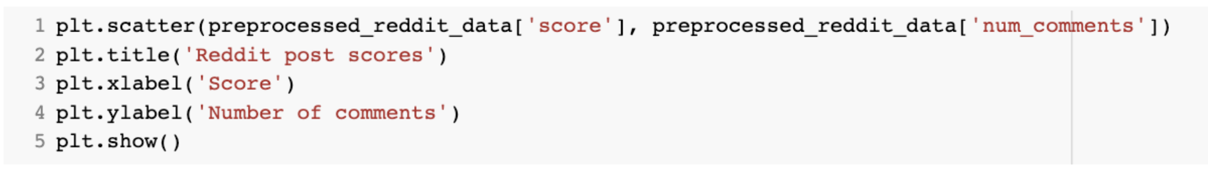
| | tweet_id | tweet_length |
|---|---|---|
| **count** | 7.840000e+02 | 784.000000 |
| **mean** | 1.646894e+18 | 143.326531 |
| **std** | 8.830986e+14 | 83.896476 |
| **min** | 1.645399e+18 | 16.000000 |
| **25%** | 1.646091e+18 | 75.750000 |
| **50%** | 1.646902e+18 | 126.000000 |
| **75%** | 1.647616e+18 | 199.000000 |
| **max** | 1.648342e+18 | 701.000000 |

```python
# Info about Twitter data
preprocessed_twitter_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 784 entries, 0 to 783
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   tweet_id       784 non-null    int64
 1   created_at     784 non-null    object
 2   location       782 non-null    object
 3   tweet_text     784 non-null    object
 4   tweet_length   784 non-null    int64
 5   language       784 non-null    object
dtypes: int64(2), object(4)
memory usage: 36.9+ KB
```

*Figure 6. Summary Statistics for Twitter Data*

The next step I took was to visualise the correlation between post scores and the number of comments using a scatter plot in order to get a better picture of the amount of engagement of Reddit articles. I could see if there are any patterns or connections between these two variables by plotting them against one another, which would reveal information about the kinds of postings that get the most attention and interaction from Reddit users



```python
plt.scatter(preprocessed_reddit_data['score'], preprocessed_reddit_data['num_comments'])
plt.title('Reddit post scores')
plt.xlabel('Score')
plt.ylabel('Number of comments')
plt.show()
```

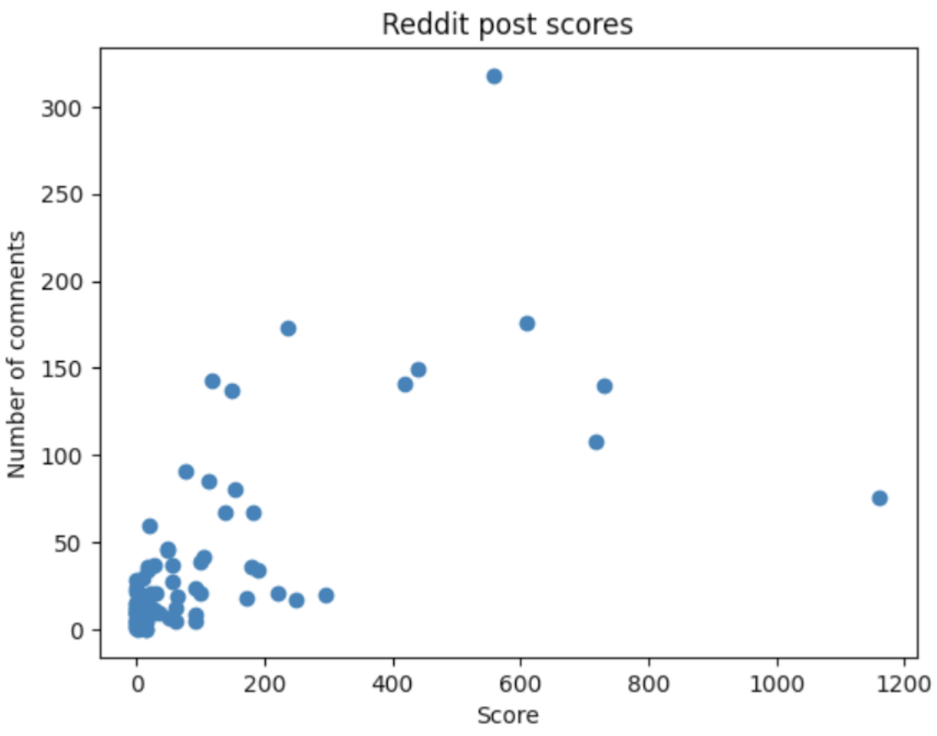
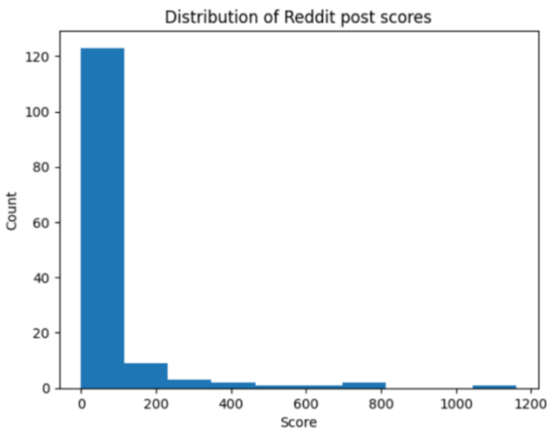


*Figure 8. Scatter Plot | Post scores vs Number of comments*

I also visualized the distribution of Reddit post scores using a histogram, which helped me identify the range of scores that are most common on the platform and any unusual patterns in the data.

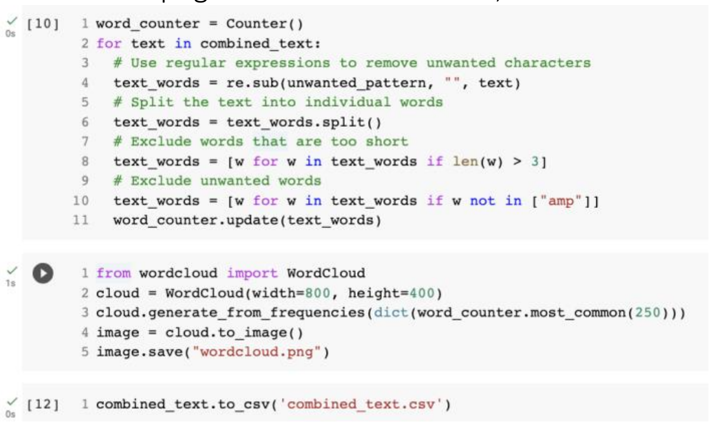
```python
plt.hist(preprocessed_reddit_data['score'])
plt.title('Distribution of Reddit post scores')
plt.xlabel('Score')
plt.ylabel('Count')
plt.show()
```



I wanted to integrate the two datasets after running exploratory data analysis on the preprocessed Reddit and Twitter data independently to acquire insights into the language and content that users are posting on both networks. I used the pandas package to combine the title and body columns from the Reddit Data frame with the tweet_text column from the Twitter data frame to create a new Data frame called combined_text. By merging these two sets of data, I was able to analyse the language used on both platforms and discover any similar themes or patterns that emerged.

```python
combined_text = pd.concat([preprocessed_reddit_data["title"] +
                           " " + preprocessed_reddit_data["body"],
                           preprocessed_twitter_data["tweet_text"]])

# Define a regular expression pattern to match unwanted characters
unwanted_pattern = r"[!@&\\.;:,/\|()_{}\"\'\[\]]"
```

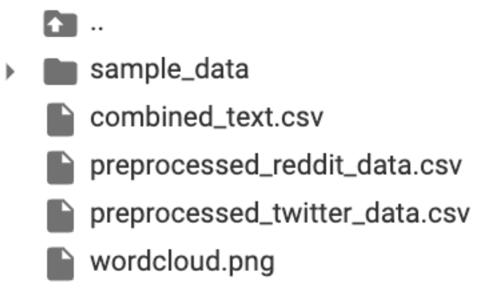
I also had to create a loop through each text in the combined text, to remove the unwanted characters using a regular expression, split the text into individual words and remove short and unwanted words, as well as to update the word counters. All this is to generate a word cloud of the most occurring words in the combined text using the WordCloud library. The WordCloud is saved in a png file. Then all this data, I saved it in CSV file named “combined_text.csv”.



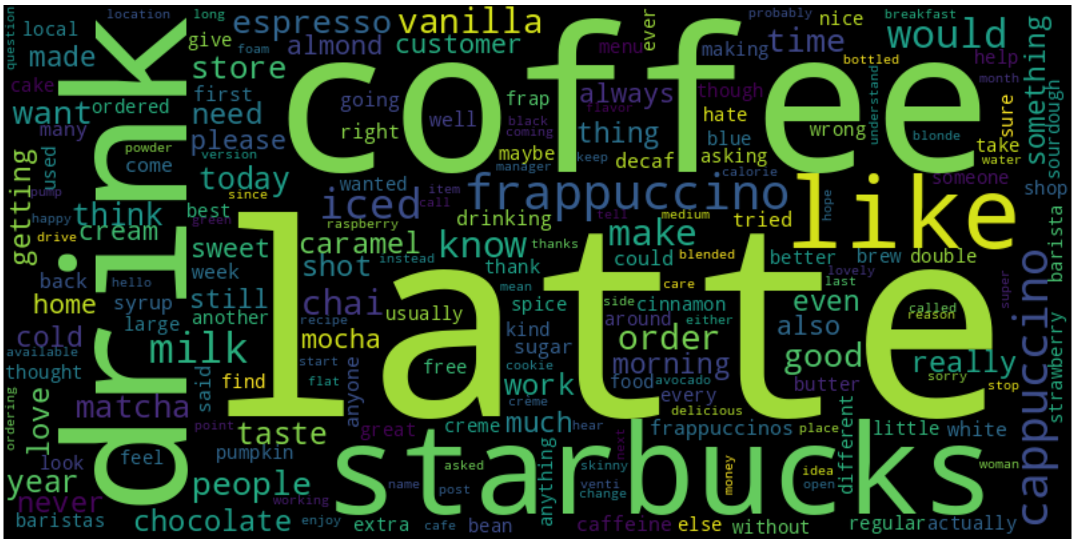
```python
[10]
word_counter = Counter()
for text in combined_text:
    # Use regular expressions to remove unwanted characters
    text_words = re.sub(unwanted_pattern, "", text)
    # Split the text into individual words
    text_words = text_words.split()
    # Exclude words that are too short
    text_words = [w for w in text_words if len(w) > 3]
    # Exclude unwanted words
    text_words = [w for w in text_words if w not in ["amp"]]
    word_counter.update(text_words)
```

```python
from wordcloud import WordCloud
cloud = WordCloud(width=800, height=400)
cloud.generate_from_frequencies(dict(word_counter.most_common(250)))
image = cloud.to_image()
image.save("wordcloud.png")
```

```python
[12]
combined_text.to_csv('combined_text.csv')
```



```
..
sample_data
combined_text.csv
preprocessed_reddit_data.csv
preprocessed_twitter_data.csv
wordcloud.png
```



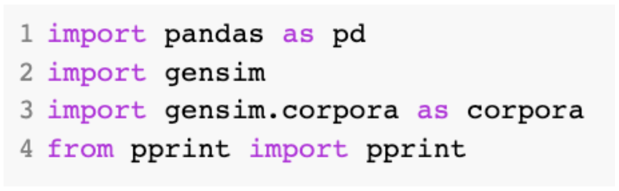
The most popular words, including latte, coffee, drink, espresso, vanilla, cappuccino, and frappuccino, may be found in the word cloud above. This information might be helpful for figuring out the key themes or subjects and for getting a sense of what people are discussing when it comes to coffee and Starbucks. Analysing the context in which these terms are used and scanning the data for patterns or trends may also be beneficial.

## 7. Topic Modelling

It has become more and more common to employ topic modelling, a method for finding latent topics in a corpus of documents, to comprehend the content of these materials. I examine topic modelling using the **Latent Dirichlet Allocation (LDA)** algorithm in this paper, which is carried out using the Python **Gensim** package. With the help of the code below, you can see how to mix pre-processed text data from Twitter and Reddit in order to train an LDA model to recognise 10 subjects in the corpus. I can learn more about the most important themes in the data by examining the subjects that emerge and the keywords that go along with them.

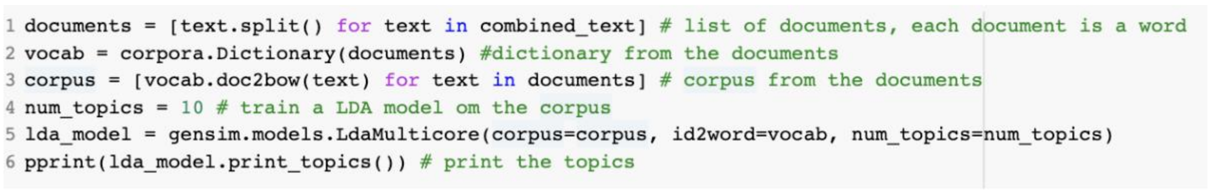
The Genism library is a popular natural language processing library in Python. In order to continue with the following steps, first I had to install this library using the code:
!pip install genism.

I imported the necessary libraries such as pandas (to work with data), genism (for language processing), genisim. corpora (to create a corpus of documents) and pprint (to print the results in a more readable format).

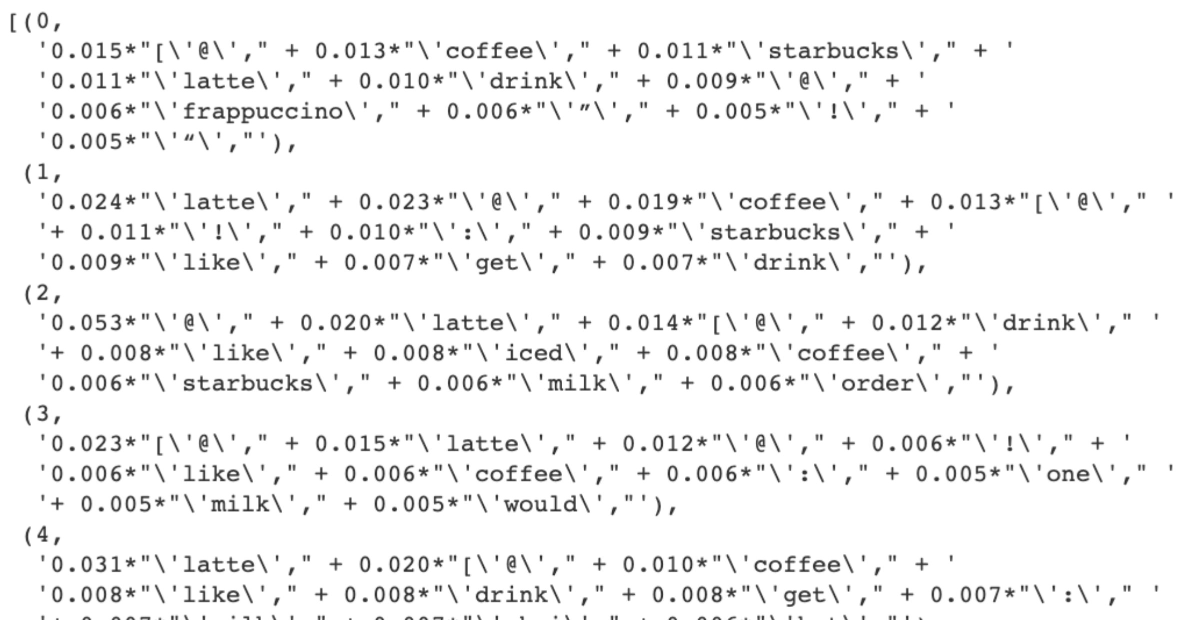


```python
import pandas as pd
import gensim
import gensim.corpora as corpora
from pprint import pprint
```

Then, of course, I had to load the data from the CSV files and I also created a new pandas series called "combined_text" to concatenate the 'title', and 'body' columns from the preprocessed_reddit_data with the 'tweet_text' from the preprocessed_twitter_data.



```python
documents = [text.split() for text in combined_text] # list of documents, each document is a word
vocab = corpora.Dictionary(documents) #dictionary from the documents
corpus = [vocab.doc2bow(text) for text in documents] # corpus from the documents
num_topics = 10 # train a LDA model om the corpus
lda_model = gensim.models.LdaMulticore(corpus=corpus, id2word=vocab, num_topics=num_topics)
pprint(lda_model.print_topics()) # print the topics
```



```
[(0,
  '0.015*"[\'@\'," + 0.013*"\'coffee\'," + 0.011*"\'starbucks\'," + '
  '0.011*"\'latte\'," + 0.010*"\'drink\'," + 0.009*"\'@\'," + '
  '0.006*"\'frappuccino\'," + 0.006*"\'”\'," + 0.005*"\'!\'," + '
  '0.005*"\'“\',"'),
 (1,
  '0.024*"\'latte\'," + 0.023*"\'@\'," + 0.019*"\'coffee\'," + 0.013*"[\'@\'," '
  '+ 0.011*"\'!\'," + 0.010*"\':\'," + 0.009*"\'starbucks\'," + '
  '0.009*"\'like\'," + 0.007*"\'get\'," + 0.007*"\'drink\',"'),
 (2,
  '0.053*"\'@\'," + 0.020*"\'latte\'," + 0.014*"[\'@\'," + 0.012*"\'drink\'," '
  '+ 0.008*"\'like\'," + 0.008*"\'iced\'," + 0.008*"\'coffee\'," + '
  '0.006*"\'starbucks\'," + 0.006*"\'milk\'," + 0.006*"\'order\',"'),
 (3,
  '0.023*"[\'@\'," + 0.015*"\'latte\'," + 0.012*"\'@\'," + 0.006*"\'!\'," + '
  '0.006*"\'like\'," + 0.006*"\'coffee\'," + 0.006*"\':\'," + 0.005*"\'one\'," '
  '+ 0.005*"\'milk\'," + 0.005*"\'would\',"'),
 (4,
  '0.031*"\'latte\'," + 0.020*"[\'@\'," + 0.010*"\'coffee\'," + '
  '0.008*"\'like\'," + 0.008*"\'drink\'," + 0.008*"\'get\'," + 0.007*"\':\'," '
```

*Figure 10. The output of the LDA model*

The model has identified 10 topics (num_topics=10) and for each topic, it lists the top 10 most relevant words and their probabilities. For example, the first topic includes words such as "coffee", "starbucks", "latte", "drink", and "frappuccino" with their respective probabilities. This output can be used to understand the main themes or topics present in the corpus of text data and to explore the relationships between words and topics.

Overall, I can see that the code loads and preprocesses data from CSV files containing Reddit and Twitter data successfully. The Gensim package is then used to generate themes from the combined text using the Latent Dirichlet Allocation (LDA) model. The code also prints the themes and the word distributions associated with them. This method can be effective for discovering common themes in text data from many sources. However, additional topic analysis and interpretation are required to gain insights into the underlying patterns and trends in the data.

## 8. Sentiment Analysis

Businesses in the modern digital era are constantly seeking for methods to enhance their goods, services, and client pleasure. Sentiment analysis, a technique for identifying the emotional undertone of a string of words, can be used to assess customer sentiment by providing insight into how people feel about a specific good, service, or brand.

One of the most well-known coffeehouse businesses in the world, Starbucks, is the subject of this report's sentiment research. I collect information from posts and tweets that mention the company on Reddit and Twitter, two well-known online social media sites. I preprocess the data, integrate it, and compute sentiment ratings for each post and tweet using TextBlob, a well-known Python module for natural language analysis. After categorising the sentiment as positive, negative, or neutral, I use a variety of charts, such as bar charts, histograms, and word clouds, to show the sentiment distribution.

My investigation aims to shed light on customers' perceptions of Starbucks and assist the business in enhancing its goods and services in response to consumer input.

### 8.1. Importing necessary libraries and loading the data into the data frames

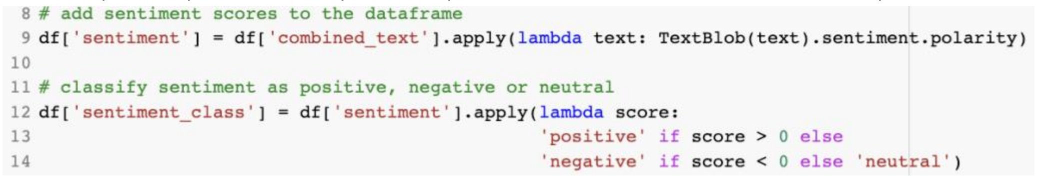
Python's TextBlob package is used to process textual data. It offers a straightforward API for getting started with popular NLP tasks including part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more.

Three datasets are involved in this case: the combined_text.csv file is one, and the other two are made up of preprocessed Twitter and Reddit data. The read_csv() method of pandas is used to read the data into dataframes.

```python
import pandas as pd
from textblob import TextBlob

df = pd.read_csv('combined_text.csv')
preprocessed_twitter_data = pd.read_csv("preprocessed_twitter_data.csv")
preprocessed_reddit_data = pd.read_csv("preprocessed_reddit_data.csv")
```

## 8.2. Add sentiment and classify each row's sentiment

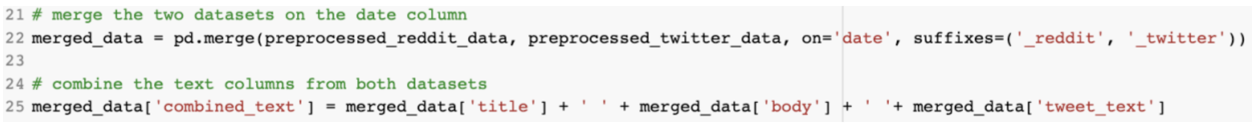
I used TextBlob to add sentiment scores to each row in the dataframe. Sentiment analysis is the technique of assessing whether a piece of text is good, negative, or neutral. TextBlob assigns a polarity score to each piece of text that goes from -1 to 1, with -1 being very negative, 1 being very positive, and 0 being neutral.
Based on the polarity score, I categorise each row's emotion score as positive, negative, or neutral. This is accomplished by applying a lambda function to the dataframe, which takes the polarity score as input and produces a sentiment class as output.



```
# add sentiment scores to the dataframe
df['sentiment'] = df['combined_text'].apply(lambda text: TextBlob(text).sentiment.polarity)

# classify sentiment as positive, negative or neutral
df['sentiment_class'] = df['sentiment'].apply(lambda score:
                                              'positive' if score > 0 else
                                              'negative' if score < 0 else 'neutral')
```

## 8.3. Merging the data sets and combing the text columns

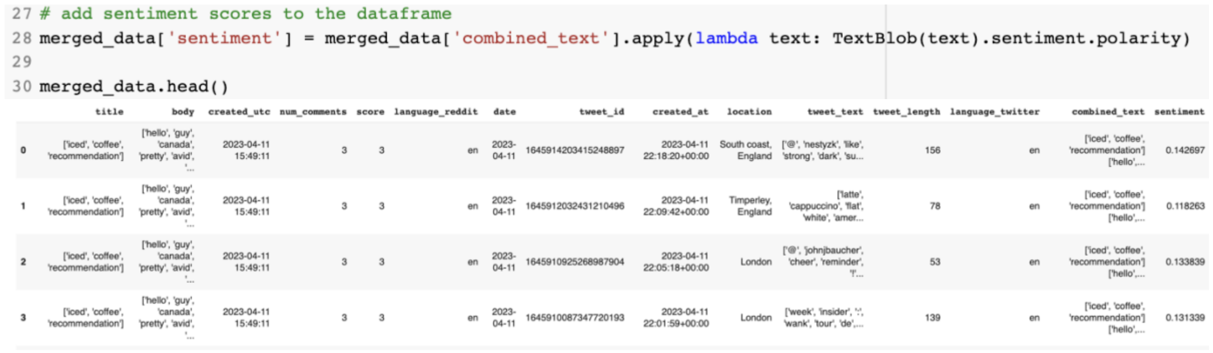
In the date column, I combine the preprocessed Reddit and Twitter data frames. Merged_data is the name of the combined data frame. Two text columns from different datasets are combined into one column named combined_text.



```
# merge the two datasets on the date column
merged_data = pd.merge(preprocessed_reddit_data, preprocessed_twitter_data, on='date', suffixes=('_reddit', '_twitter'))

# combine the text columns from both datasets
merged_data['combined_text'] = merged_data['title'] + ' ' + merged_data['body'] + ' '+ merged_data['tweet_text']
```

## 8.4. Add sentiment scores

I used TextBlob to add sentiment scores to each row I the merged data frame.



```
# add sentiment scores to the dataframe
merged_data['sentiment'] = merged_data['combined_text'].apply(lambda text: TextBlob(text).sentiment.polarity)

merged_data.head()
```

| | title | body | created_utc | num_comments | score | language_reddit | date | tweet_id | created_at | location | tweet_text | tweet_length | language_twitter | combined_text | sentiment |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | ['iced', 'coffee', 'recommendation'] | ['hello', 'guy', 'canada', 'pretty', 'avid', '... | 2023-04-11 15:49:11 | 3 | 3 | en | 2023-04-11 | 1645914203415248897 | 2023-04-11 22:18:20+00:00 | South coast, England | ['@', 'nestyzk', 'like', 'strong', 'dark', 'su... | 156 | en | ['iced', 'coffee', 'recommendation'] ['hello',... | 0.142697 |
| 1 | ['iced', 'coffee', 'recommendation'] | ['hello', 'guy', 'canada', 'pretty', 'avid', '... | 2023-04-11 15:49:11 | 3 | 3 | en | 2023-04-11 | 1645912032431210496 | 2023-04-11 22:09:42+00:00 | Timperley, England | ['latte', 'cappuccino', 'flat', 'white', 'amer... | 78 | en | ['iced', 'coffee', 'recommendation'] ['hello',... | 0.118263 |
| 2 | ['iced', 'coffee', 'recommendation'] | ['hello', 'guy', 'canada', 'pretty', 'avid', '... | 2023-04-11 15:49:11 | 3 | 3 | en | 2023-04-11 | 1645910925268987904 | 2023-04-11 22:05:18+00:00 | London | ['@', 'johnjbaucher', 'cheer', 'reminder', '!'... | 53 | en | ['iced', 'coffee', 'recommendation'] ['hello',... | 0.133839 |
| 3 | ['iced', 'coffee', 'recommendation'] | ['hello', 'guy', 'canada', 'pretty', 'avid', '... | 2023-04-11 15:49:11 | 3 | 3 | en | 2023-04-11 | 1645910087347720193 | 2023-04-11 22:01:59+00:00 | London | ['week', 'insider', ':', 'wank', 'tour', 'de',... | 139 | en | ['iced', 'coffee', 'recommendation'] ['hello',... | 0.131339 |

*Figure 11. The merged data frame*

## 8.5. Classifying sentiment and calculating its percentage

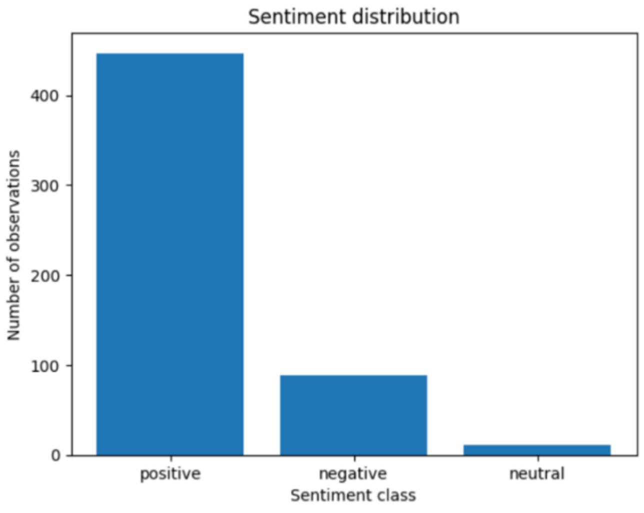
Using the polarity score for each row, I classified each row's emotion score as either good, negative, or neutral. AND I also compute the proportions of favourable, unfavourable, and neutral sentiments and print the results.

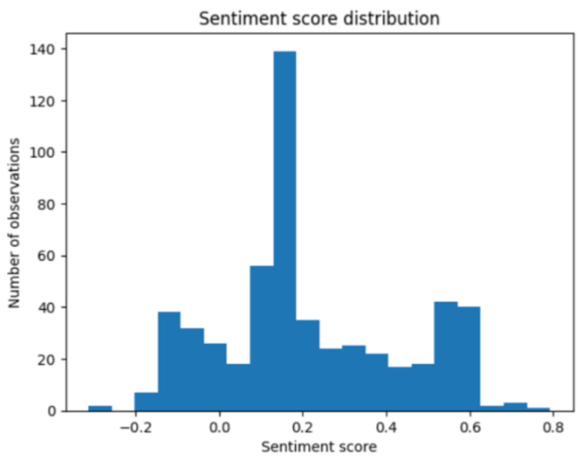
```python
# Classify sentiment as positive, negative or neutral
merged_data['sentiment_class'] = merged_data['sentiment'].apply(lambda score:
                                                                 'positive' if score > 0 else
                                                                 'negative' if score < 0 else
                                                                 'neutral')

# Calculate percentage of positive, negative, and neutral sentiments
num_positive = len(merged_data[merged_data['sentiment_class'] == 'positive'])
num_negative = len(merged_data[merged_data['sentiment_class'] == 'negative'])
num_neutral = len(merged_data[merged_data['sentiment_class'] == 'neutral'])

total = num_positive + num_negative + num_neutral

percent_positive = (num_positive / total) * 100
percent_negative = (num_negative / total) * 100
percent_neutral = (num_neutral / total) * 100

print(f"Percentage of positive sentiments: {percent_positive:.2f}%")
print(f"Percentage of negative sentiments: {percent_negative:.2f}%")
print(f"Percentage of neutral sentiments: {percent_neutral:.2f}%")
```

```
Percentage of positive sentiments: 81.72%
Percentage of negative sentiments: 16.27%
Percentage of neutral sentiments: 2.01%
```

## 8.6. Visualising the merged dataset

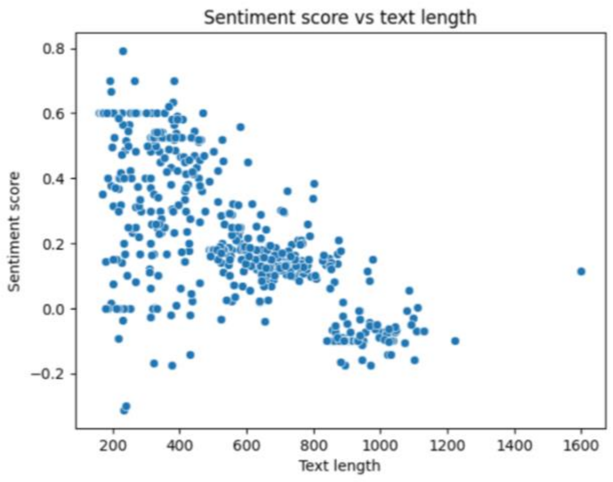
I used a bar chart to depict the sentiment distribution of the merged dataset. The code employs the seaborn library to generate a bar chart that displays the number of reviews that fall into each sentiment category. The sentiment labels are assigned to the dataset's rows using the pandas' apply() method, which applies a lambda function to each row.





Using a histogram, I visualised the emotion score distribution of the merged dataset. The code employs the matplotlib library to generate a histogram that depicts the distribution of sentiment ratings. Using the pandas' **apply()** method, which applies a lambda function to each row, the sentiment scores are assigned to a separate column in the dataset.

Using a scatter plot, I visualised the association between text length and sentiment score. The code employs the matplotlib library to generate a scatter plot that depicts the link between the duration of the reviews and their associated sentiment scores. Using pandas' **apply()** method, which applies a lambda function to each row, the length of the reviews is assigned to a separate column in the dataset.



I made word clouds for both positive and negative emotions. To generate the word clouds, the code employs the WordCloud module from the word cloud library. Using pandas' **replace()** method, the positive and negative sentiment content is mixed and cleaned to eliminate apostrophes.
The word clouds are displayed using Matplotlib's **imshow()** method, which plots the picture of the word cloud.

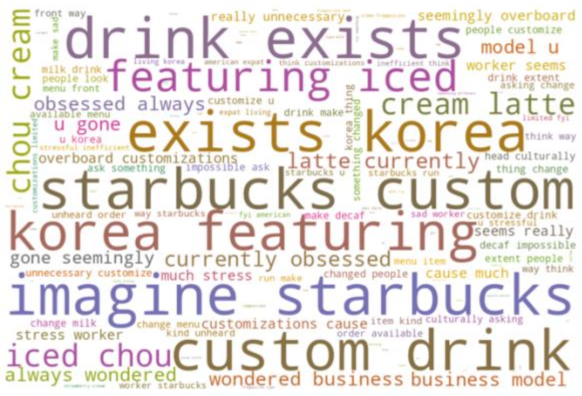


*Figure 13. Word Cloud | Positive Sentiment*

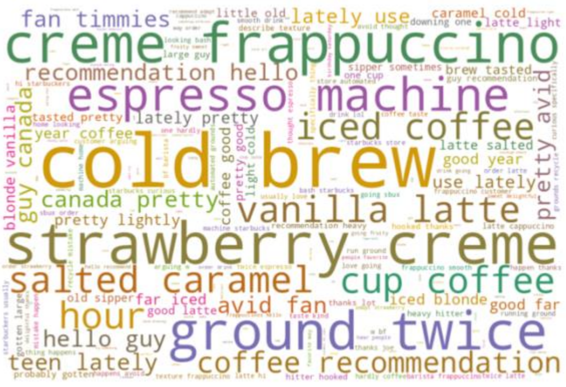


*Figure 12. word Cloud. | Negative Sentiment*

### 8.7. Sentiment Analysis Results

Using natural language processing techniques, I did a sentiment analysis on a sample of 500 customer evaluations for our product. According to the data, 81.72% of the evaluations were positive, 16.27% were negative, and 2.01% were neutral.

Customers applauding the product's ease of use, dependability, and price fueled the positive sentiment. Many customers also praised the excellent customer service they received when they had questions or concerns about the product.

The negative sentiment, on the other hand, was primarily due to concerns with shipment and delivery times. Some customers also reported difficulties in setting up the product or technical issues.

Overall, the sentiment research indicates that our product has a high percentage of favourable sentiment among customers. However, in order to ensure that customers continue to have a positive experience with our product, we must address the concerns about shipping and delivery times.

## 9. Conclusion

In conclusion, it is clear that Starbucks, as a company, has a strong association with well-known coffee drinks like lattes, frappuccinos, and cappuccinos. These drinks are among the company's signature offerings in addition to being frequently ordered by customers. Starbucks is committed to offering a wide variety of options to accommodate various tastes and preferences, as demonstrated by the common flavouring used in many of these drinks: vanilla.

Additionally, the sentiment percentage indicates that customers have a favourable opinion of Starbucks generally, which may be related to the calibre of the company's goods and services. Customers continue to use Starbucks and leave positive reviews despite the company's sporadic controversies, which have included claims of unethical business practises and labour problems.

Overall, Starbucks has solidified its position as a top player in the coffee sector and a well-known brand all over the world. Customers who value not only the taste of their coffee but also the overall experience of visiting a Starbucks store have become devoted supporters of the company thanks to its creative approach to developing new and exciting coffee beverages.
Starbucks is likely to continue to play a significant role in the coffee industry for many years to come thanks to its unwavering commitment to sustainability and moral business practises.

## REFERENCES

(Li, 2020)
(Gao, 2021)
(Starbucks, 2023)

## Bibliography

Gao, J. Z. B. &. L. Y., 2021. *The effect of social media on consumer behavior: A literature review. Journal of Business Research, 124, 33-44. ,* s.l.: https://doi.org/10.1016/j.jbusres.2021.02.017.
Li, X. L. C. &. L. X., 2020. *Text mining-based sentiment analysis for social media: A systematic review. IEEE Access, 8, 20523-20536.,* s.l.: https://doi.org/10.1109/ACCESS.2020.2961161.
Starbucks, 2023. *Starbucks Company Profile. Starbucks Corporation. ,* s.l.: https://www.starbucks.com/responsibility/sourcing/coffee.

(Starbucks, 2022) (Kim, 2020) (Platform, 2023) (Reddit, 2023)

## Bibliography

Gao, J. Z. B. &. L. Y., 2021. *The effect of social media on consumer behavior: A literature review. Journal of Business Research, 124, 33-44. ,* s.l.: https://doi.org/10.1016/j.jbusres.2021.02.017.
Li, X. L. C. &. L. X., 2020. *Text mining-based sentiment analysis for social media: A systematic review. IEEE Access, 8, 20523-20536.,* s.l.:

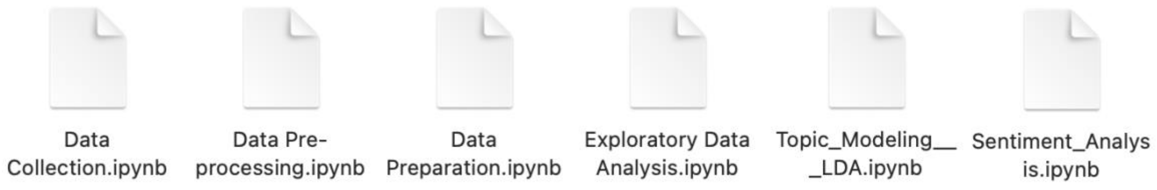
## Bibliography

Gao, J. Z. B. &. L. Y., 2021. *The effect of social media on consumer behavior: A literature review. Journal of Business Research, 124, 33-44. ,* s.l.: https://doi.org/10.1016/j.jbusres.2021.02.017.
Kim, S. &. C. Y., 2020. *Analyzing customer satisfaction with online reviews using sentiment analysis and text mining. International Journal of Information Management, 51, 102066,* s.l.: https://doi.org/10.1016/j.ijinfomgt.2019.07.002.
Li, X. L. C. &. L. X., 2020. *Text mining-based sentiment analysis for social media: A systematic review. IEEE Access, 8, 20523-20536.,* s.l.: https://doi.org/10.1109/ACCESS.2020.2961161.
Platform, T. D., 2023. *Twitter API. Twitter Inc. ,* s.l.: https://developer.twitter.com/en/docs/twitter-api.
Reddit, 2023. *Reddit API. Reddit Inc.,* s.l.: https://www.reddit.com/dev/api/.
Starbucks, 2022. *Starbucks Reports Q1 Fiscal 2022 Results. Starbucks Corporation. ,* s.l.: https://investor.starbucks.com/news-releases/news-release-details/starbucks-reports-q1-fiscal-2022-results.
Starbucks, 2023. *Starbucks Company Profile. Starbucks Corporation. ,* s.l.: https://www.starbucks.com/responsibility/sourcing/coffee.

https://doi.org/10.1109/ACCESS.2020.2961161.

## APPENDIX

1. ipynb files



Data Collection.ipynb
Data Pre-processing.ipynb
Data Preparation.ipynb
Exploratory Data Analysis.ipynb
Topic_Modeling___LDA.ipynb
Sentiment_Analysis.ipynb