```python
import requests
from bs4 import BeautifulSoup
import pandas as pd
from urllib.parse import urlencode
import csv


# List of URLS to search
list_of_urls = ['https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re
                'https://www.amazon.com/Assault-Fitness-Air-Bike-AirBike/product-re


# Retrieve each of the url's HTML data and convert the data into a beautiful soup c
# Find, extract and store reviewer names and review text into a list.

names = []
reviews = []
data_string = ""

for url in list_of_urls:
    params = {'api_key': "ENTERAPIKEYHERE", 'url': url}
    response = requests.get('http://api.scraperapi.com/',   params=urlencode(params
    soup = BeautifulSoup(response.text, 'html.parser')

    for item in soup.find_all("span", class_="a-profile-name"):
      data_string = data_string + item.get_text()
      names.append(data_string)
      data_string = ""

    for item in soup.find_all("span", {"data-hook": "review-body"}):
      data_string = data_string + item.get_text()
      reviews.append(data_string)
      data_string = ""


# Create the dictionary
reviews_dict = {'Reviewer Name': names, 'Reviews': reviews}
```

```
# Print the lengths of each list.
print(len(names), len(reviews))
```

```
128 100
```

```
# Create a new dataframe
df = pd.DataFrame.from_dict(reviews_dict, orient='index')
df.head()
```

	0	1	2	3	4	5	6	7	
Reviewer Name	Scott	Goof Ball	Mike	Hickman	Amazon Customer	Amazon Customer	Dannyun	james mcleod	
Reviews	\nI road an Echo, little more expensive but th...	\nBike was shipped and came with damaged parts...		\nThis is literally the best cardio machine th...	\nquietness\n	\nEasy to assemble and REALLY good value despi...	\nNot too hard to put together at all. Great w...	\nI have used this bike for under 50 miles and...	a y u

2 rows × 128 columns

```
# Delete all the columns that have missing values
df.dropna(axis=1, inplace=True)
df.head()
```

	0	1	2	3	4	5	6	7	
Reviewer Name	Scott	Goof Ball	Mike	Hickman	Amazon Customer	Amazon Customer	Dannyun	james mcleod	
Reviews	\nI road an Echo, little more expensive but th...	\nBike was shipped and came with damaged parts...		\nThis is literally the best cardio machine th...	\nquietness\n	\nEasy to assemble and REALLY good value despi...	\nNot too hard to put together at all. Great w...	\nI have used this bike for under 50 miles and...	a y u

2 rows × 100 columns

```
# Transpose the dataframe
prod_reviews = df.T
print(prod_reviews.head(10))
```

```
     Reviewer Name                                            Reviews
0            Scott  \nI road an Echo, little more expensive but th...
1        Goof Ball  \nBike was shipped and came with damaged parts...
2             Mike
3          Hickman  \nThis is literally the best cardio machine th...
4  Amazon Customer                                    \nquietness\n
5  Amazon Customer  \nEasy to assemble and REALLY good value despi...
6          Dannyun  \nNot too hard to put together at all. Great w...
7    james  mcleod  \nI have used this bike for under 50 miles and...
8              Don  \nI've had the bike for almost a year, I use i...
9        Anonymous  \nI love the bike. It is smaller than I expect...
```

```
# Remove special characters
prod_reviews['Reviews'] = prod_reviews['Reviews'].str.replace('\n','')
prod_reviews.head(5)
```

	Reviewer Name	Reviews
0	Scott	I road an Echo, little more expensive but the ...
1	Goof Ball	Bike was shipped and came with damaged parts. ...
2	Mike	
3	Hickman	This is literally the best cardio machine that...
4	Amazon Customer	quietness

```
# Convert dataframe to CSV file
prod_reviews.to_csv('reviews.csv', index=False, header=True)
```

Sentiment Analysis

```
import pandas as pd
import nltk
nltk.download('punkt')
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
data = pd.read_csv('/content/rogue_reviews.csv')
data.head()
```

	Reviewer Name	Reviews
0	A TEAM 114	Easy to assemble
1	Aaron	Bad quality, bad customer service, no exaggera...
2	Amazon Customer	quietness
3	Amazon Customer	Easy to assemble and REALLY good value despite...
4	Amazon Customer	Real easy to assemble and comes with everythin...

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   Reviewer Name  100 non-null    object
 1   Reviews        99 non-null     object
dtypes: object(2)
memory usage: 1.7+ KB
```

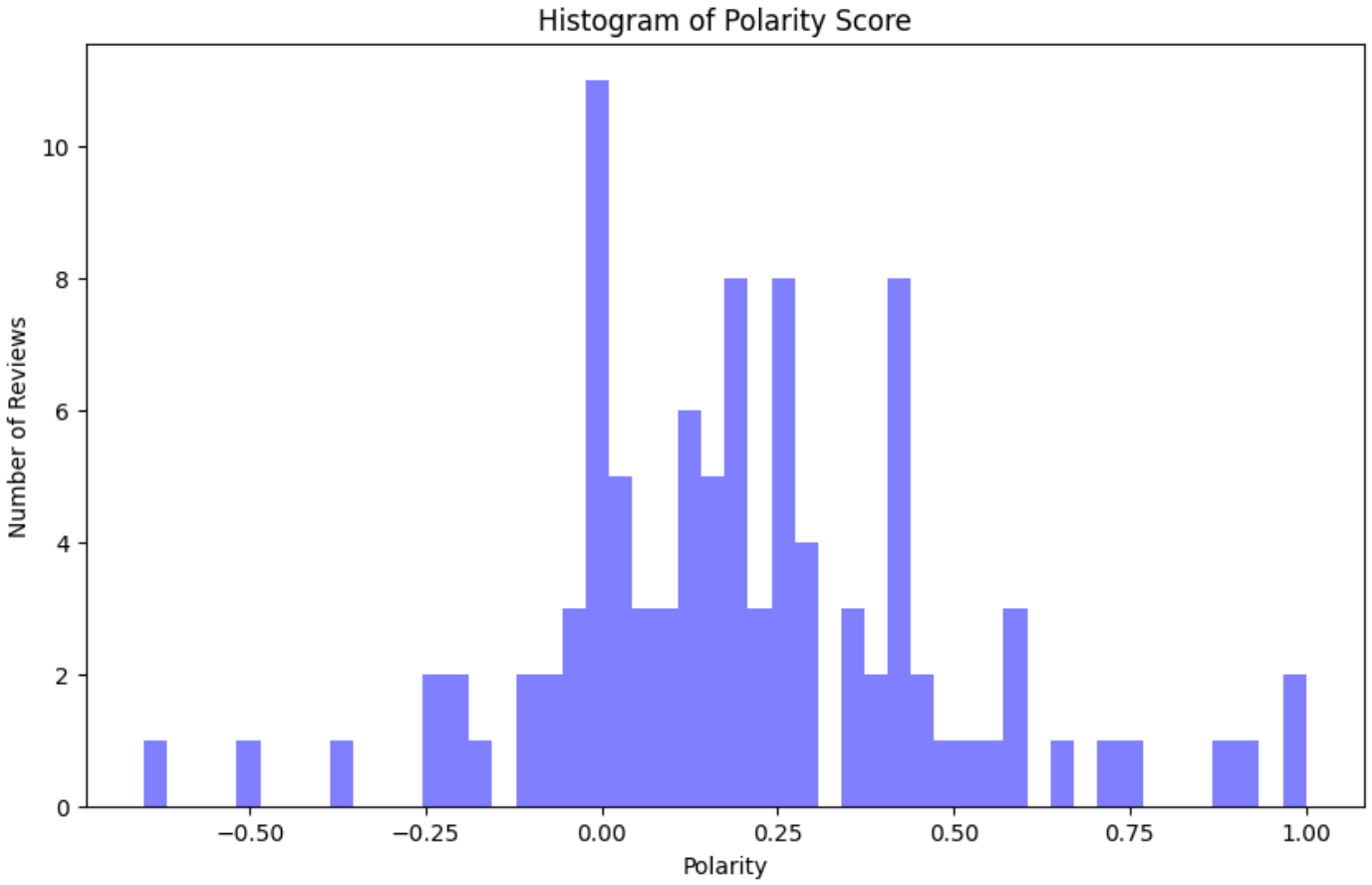
```
# drop any null values

data.dropna(inplace=True)
```

```
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
# polarity and subjectivity using wordnet and textblob based on review text
# 1 = positive
# -1 = negative
from textblob import TextBlob

# Lambda function to find the polarity of each review
data['Reviews']= data['Reviews'].astype(str)
pol = lambda x: TextBlob(x).sentiment.polarity
data['polarity'] = data['Reviews'].apply(pol)

# Plot of scores
```

```
import matplotlib.pyplot as plt
import seaborn as sns
num_bins = 50
plt.figure(figsize=(10,6))
n, bins, patches = plt.hist(data.polarity, num_bins, facecolor='blue', alpha=0.5)
plt.xlabel('Polarity')
plt.ylabel('Number of Reviews')
plt.title('Histogram of Polarity Score')
plt.show();
```



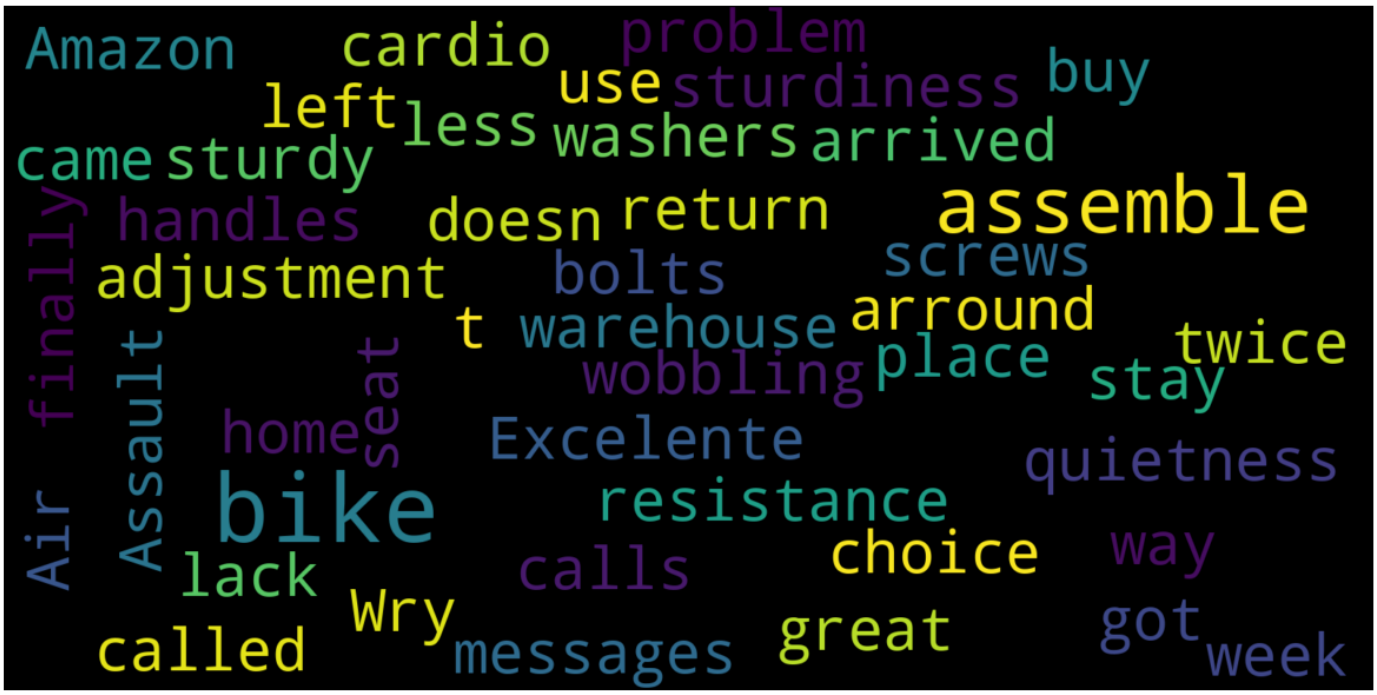
```
stp_words=stopwords.words('english')
def clean_review(review):
  cleanreview=" ".join(word for word in review.
                       split() if word not in stp_words)
  return cleanreview

data['Reviews']=data['Reviews'].apply(clean_review)


data.head()
```

	Reviewer Name	Reviews	polarity
0	A TEAM 114	Easy assemble	0.433333
1	Aaron	Bad quality, bad customer service, exaggeratio...	-0.204258
2	Amazon Customer	quietness	0.000000
3	Amazon Customer	Easy assemble REALLY good value despite price ...	0.142045
4	Amazon Customer	Real easy assemble comes everything need. Very...	0.427778

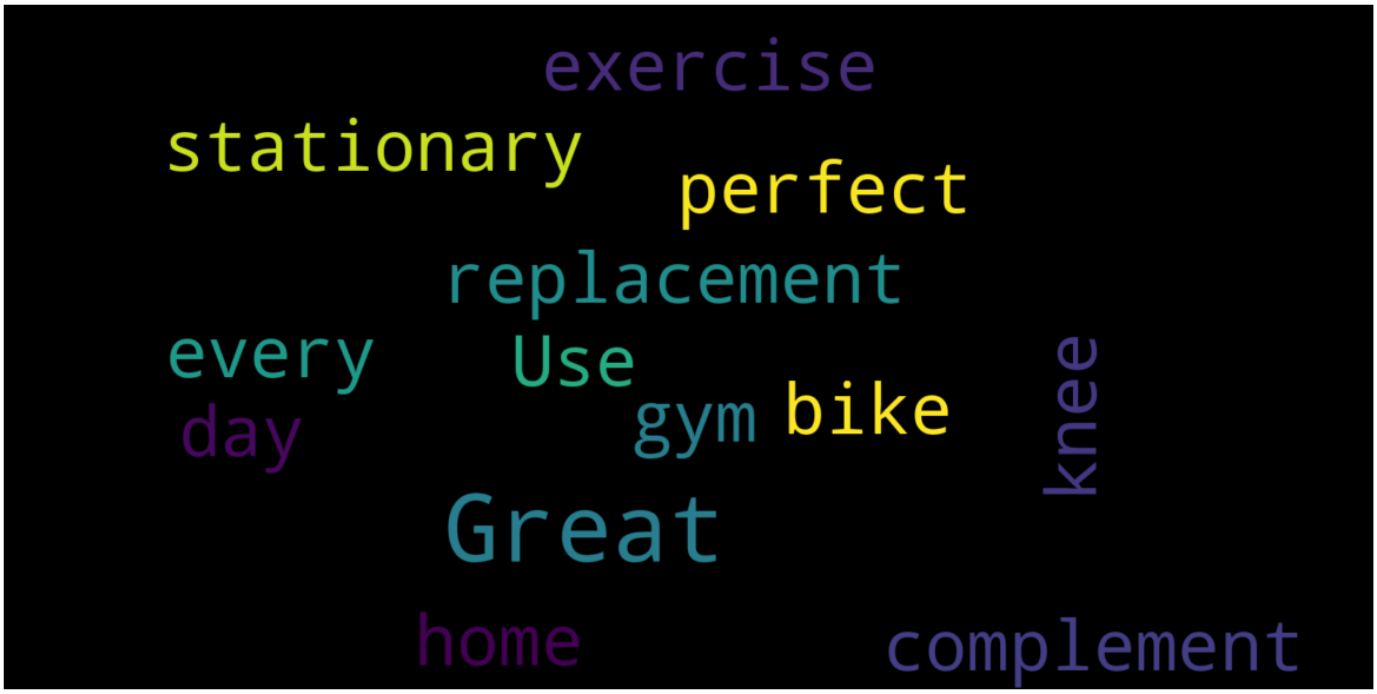
```
data['polarity'].value_counts()
```

```
 0.000000    8
 0.433333    4
 0.300000    2
 1.000000    2
 0.036364    1
            ..
-0.076923    1
 0.475000    1
 0.286967    1
 0.250000    1
 0.900000    1
Name: polarity, Length: 87, dtype: int64
```

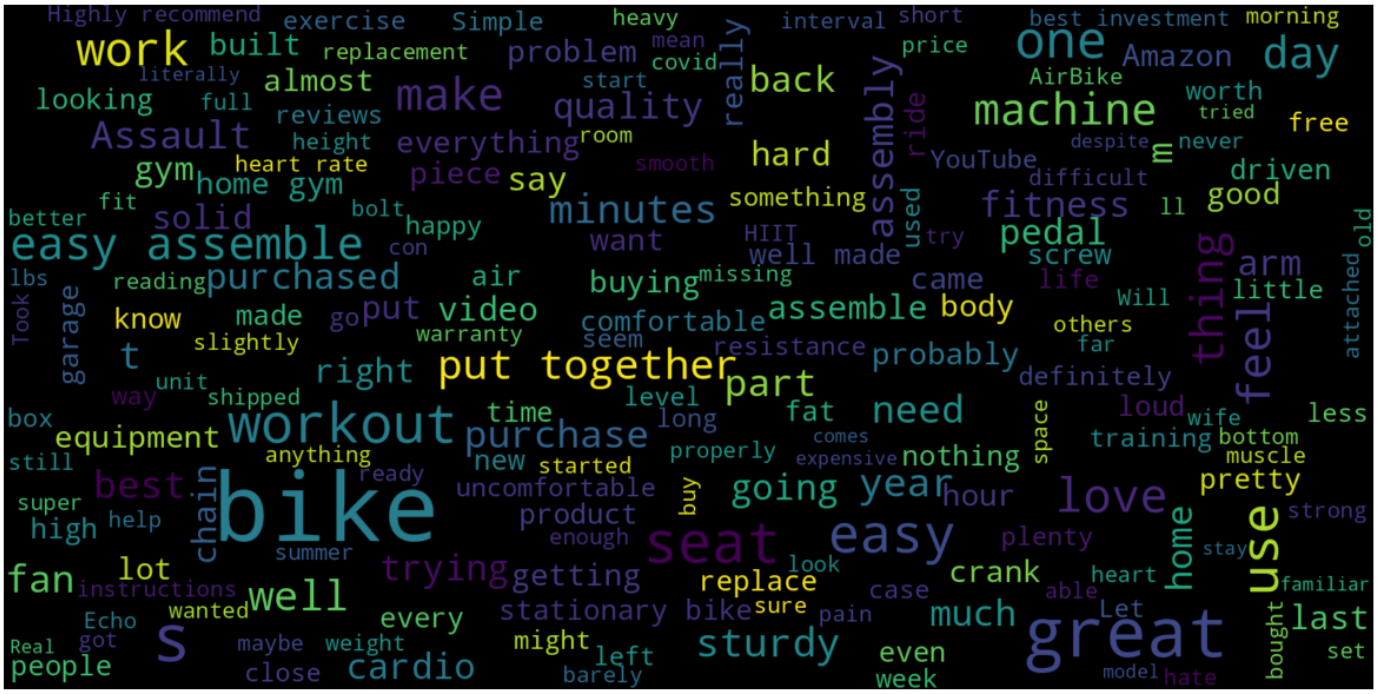
```
# neutral review word cloud - polarity score equal to "0"

from wordcloud import WordCloud

consolidated=' '.join(word for word in data['Reviews'][data['polarity']==0].astyp
wordCloud=WordCloud(width=1600,height=800,random_state=21,max_font_size=110)
plt.figure(figsize=(15,10))
plt.imshow(wordCloud.generate(consolidated),interpolation='bilinear')
plt.axis('off')
plt.show()
```



```
# positive review word cloud - polarity score equal to "1"

consolidated=' '.join(word for word in data['Reviews'][data['polarity']==1].astyp
wordCloud=WordCloud(width=1600,height=800,random_state=21,max_font_size=110)
plt.figure(figsize=(15,10))
plt.imshow(wordCloud.generate(consolidated),interpolation='bilinear')
plt.axis('off')
plt.show()
```



```
# word cloud - reviews greater than 0 (positive)

consolidated=' '.join(word for word in data['Reviews'][data['polarity']>=0].astyp
wordCloud=WordCloud(width=1600,height=800,random_state=21,max_font_size=110)
plt.figure(figsize=(15,10))
plt.imshow(wordCloud.generate(consolidated),interpolation='bilinear')
plt.axis('off')
plt.show()
```



```
# negative reviews - polarity scores less than "0"

consolidated=' '.join(word for word in data['Reviews'][data['polarity']<0].astype
wordCloud=WordCloud(width=1600,height=800,random_state=21,max_font_size=110)
plt.figure(figsize=(15,10))
plt.imshow(wordCloud.generate(consolidated),interpolation='bilinear')
plt.axis('off')
plt.show()
```