assignment-7

April 17, 2024

0.1 Problem Statement

Consider the Amazon Alexa Reviews Dataset. This dataset consists of a nearly 3000 Amazon customer reviews (input text), star ratings, date of review, variant and feedback of various amazon Alexa products like Alexa Echo, Echo dots, Alexa Firesticks etc. Perform following operations on this dataset. * (i) Plot a graph of Positive and Negative Feedback (1 = Positive Feedback, 0 = Negative Feedback) * (ii) Plot the graph of Ratings distribution. * (iii) Convert the review text into lowercase * (iv) Remove all punctuations from review text. * (v) Remove emoticons and emojis from the text * (vi) Tokenize the review text into words. * (vii) Remove the Stopwords from the tokenized text. * (viii) Perform stemming & lemmatization on the review text. * (ix) Perform the word vectorization on review text using Bag of Words technique. * (x) Create representation of Review Text by calculating Term Frequency and Inverse Document Frequency (TF-IDF)

[2]:
```
!pip install emoji
```

```
Collecting emoji
  Downloading emoji-2.11.0-py2.py3-none-any.whl (433 kB)
                             433.8/433.8
kB 3.1 MB/s eta 0:00:00
Installing collected packages: emoji
Successfully installed emoji-2.11.0
```

[3]:
```
import pandas as pd
import matplotlib.pyplot as plt
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import PorterStemmer, WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
import string
import emoji
```

[43]:
```
df = pd.read_csv('/content/Alexa-Dataset.csv')
```

[44]:
```
df.isna().sum()
```

[44]:
```
rating            0
date              0
```

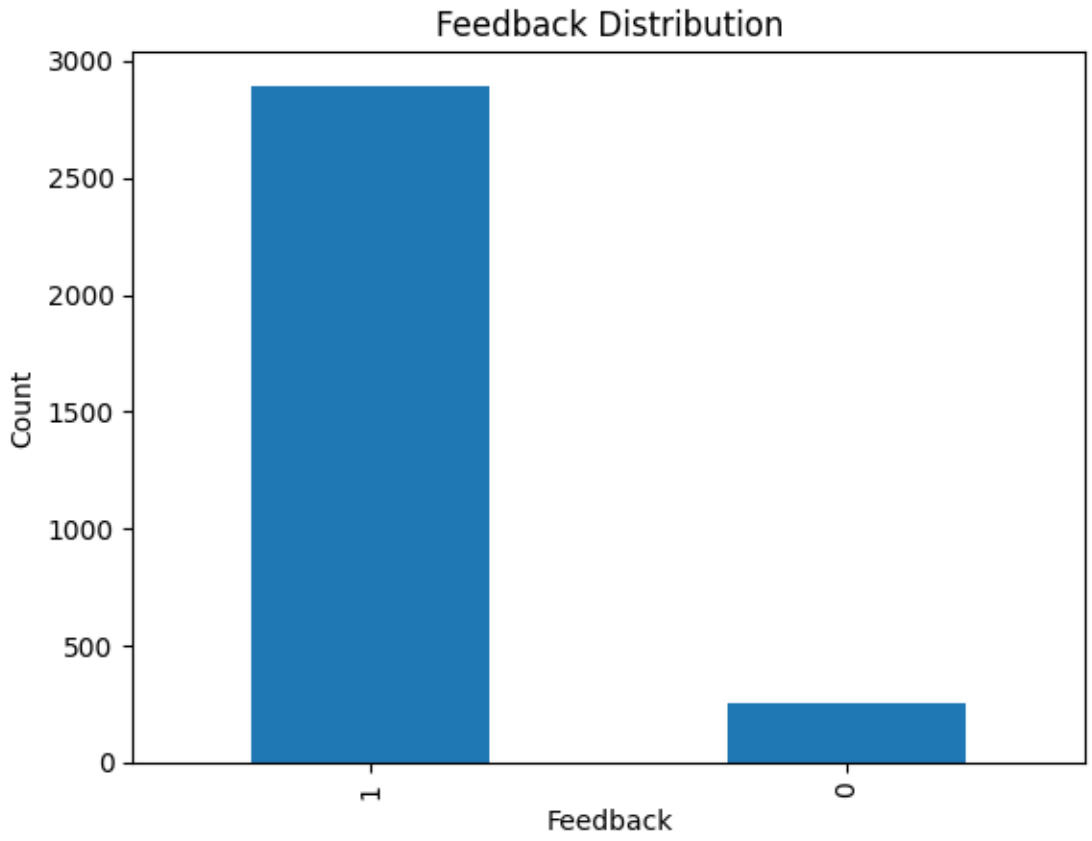
```
variation           0
verified_reviews    1
feedback            0
dtype: int64
```

[45]:
```
df['verified_reviews'] = df['verified_reviews'].fillna('')
```

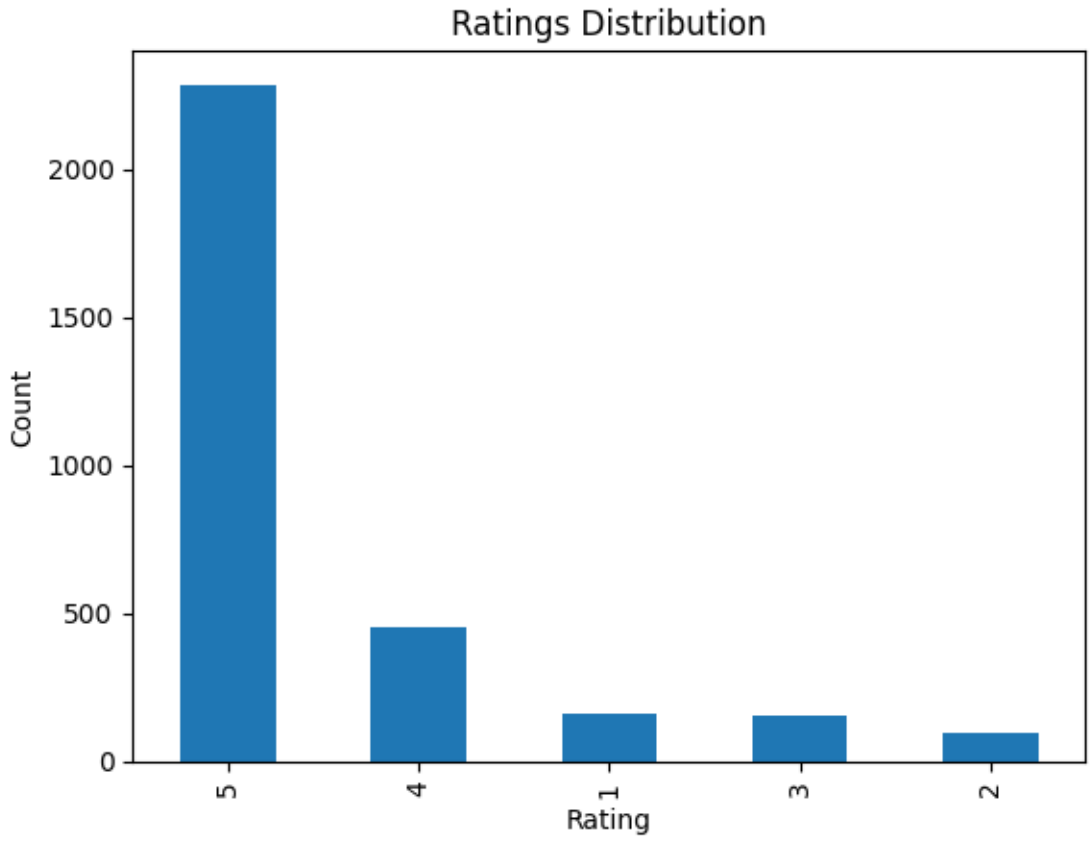
[46]:
```
df.isna().sum()
```

[46]:
```
rating              0
date                0
variation           0
verified_reviews    0
feedback            0
dtype: int64
```

[47]:
```
# (i) Plot a graph of Positive and Negative Feedback
df['feedback'].value_counts().plot(kind='bar')
plt.title('Feedback Distribution')
plt.xlabel('Feedback')
plt.ylabel('Count')
plt.show()
```



[48]:
```
# (ii) Plot the graph of Ratings distribution
df['rating'].value_counts().plot(kind='bar')
plt.title('Ratings Distribution')
plt.xlabel('Rating')
plt.ylabel('Count')
plt.show()
```



[49]:
```
# (iii) Convert the review text into lowercase
df['verified_reviews'] = df['verified_reviews'].str.lower()
```

[50]:
```
df['verified_reviews'].head()
```

[50]:
```
0                                        love my echo!
1                                            loved it!
2    sometimes while playing a game, you can answer…
3    i have had a lot of fun with this thing. my 4 …
4                                               music
Name: verified_reviews, dtype: object
```

[51]:
```
# (iv) Remove all punctuations from review text
df['verified_reviews'] = df['verified_reviews'].apply(lambda x: x.translate(str.
 ↪maketrans('', '', string.punctuation)))
df['verified_reviews'].head()
```

[51]:
```
0                                          love my echo
1                                              loved it
2    sometimes while playing a game you can answer …
3    i have had a lot of fun with this thing my 4 y…
4                                                music
Name: verified_reviews, dtype: object
```

[52]:
```
# (v) Remove emoticons and emojis from the text
def remove_emoji(text):
    return text.encode('ascii', 'ignore').decode('ascii')

df['verified_reviews'] = df['verified_reviews'].apply(remove_emoji)
df['verified_reviews'].head()
```

[52]:
```
0                                          love my echo
1                                              loved it
2    sometimes while playing a game you can answer …
3    i have had a lot of fun with this thing my 4 y…
4                                                music
Name: verified_reviews, dtype: object
```

[53]:
```
import nltk
nltk.download('punkt')
```

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

[53]:
```
True
```

[54]:
```
# (vi) Tokenize the review text into words
df['tokenized_text'] = df['verified_reviews'].apply(lambda x: x.split())
df['tokenized_text']
```

[54]:
```
0                                       [love, my, echo]
1                                            [loved, it]
2       [sometimes, while, playing, a, game, you, can,…
3       [i, have, had, a, lot, of, fun, with, this, th…
4                                              [music]
                              …
3145    [perfect, for, kids, adults, and, everyone, in…
3146    [listening, to, music, searching, locations, c…
3147    [i, do, love, these, things, i, have, them, ru…
```

```
3148    [only, complaint, i, have, is, that, the, soun…
3149                                                [good]
Name: tokenized_text, Length: 3150, dtype: object
```

[55]:
```
import nltk
nltk.download('stopwords')
```

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

[55]:
```
True
```

[56]:
```
# (vii) Remove the Stopwords from the tokenized text
stop_words = set(stopwords.words('english'))
df['filtered_text'] = df['tokenized_text'].apply(lambda x: [word for word in x␣
 ↪if word not in stop_words])
df['filtered_text']
```

[56]:
```
0                                          [love, echo]
1                                               [loved]
2       [sometimes, playing, game, answer, question, c…
3       [lot, fun, thing, 4, yr, old, learns, dinosaur…
4                                               [music]
                              …
3145                    [perfect, kids, adults, everyone]
3146    [listening, music, searching, locations, check…
3147    [love, things, running, entire, home, tv, ligh…
3148    [complaint, sound, quality, isnt, great, mostl…
3149                                                [good]
Name: filtered_text, Length: 3150, dtype: object
```

[57]:
```
import nltk
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

[57]:
```
True
```

[58]:
```
# (viii) Perform stemming & lemmatization on the review text
porter = PorterStemmer()
wordnet_lemmatizer = WordNetLemmatizer()

df['stemmed_text'] = df['filtered_text'].apply(lambda x: [porter.stem(word) for␣
 ↪word in x])
df['lemmatized_text'] = df['filtered_text'].apply(lambda x: [wordnet_lemmatizer.
 ↪lemmatize(word) for word in x])
```

[59]:
```
df['stemmed_text']
```

[59]:
```
0                                              [love, echo]
1                                                   [love]
2       [sometim, play, game, answer, question, correc…
3       [lot, fun, thing, 4, yr, old, learn, dinosaur,…
4                                                  [music]
                              …
3145                       [perfect, kid, adult, everyon]
3146    [listen, music, search, locat, check, time, lo…
3147    [love, thing, run, entir, home, tv, light, the…
3148    [complaint, sound, qualiti, isnt, great, mostl…
3149                                               [good]
Name: stemmed_text, Length: 3150, dtype: object
```

[60]:
```
df['lemmatized_text']
```

[60]:
```
0                                              [love, echo]
1                                                  [loved]
2       [sometimes, playing, game, answer, question, c…
3       [lot, fun, thing, 4, yr, old, learns, dinosaur…
4                                                  [music]
                              …
3145                      [perfect, kid, adult, everyone]
3146    [listening, music, searching, location, checki…
3147    [love, thing, running, entire, home, tv, light…
3148    [complaint, sound, quality, isnt, great, mostl…
3149                                               [good]
Name: lemmatized_text, Length: 3150, dtype: object
```

[61]:
```
# (ix) Perform the word vectorization on review text using Bag of Words␣
 ↪technique
vectorizer = CountVectorizer()
X_bow = vectorizer.fit_transform(df['lemmatized_text'].apply(lambda x: ' '.
 ↪join(x)))
```

[62]:
```
# Convert the BoW matrix to a DataFrame
bow_df = pd.DataFrame(X_bow.toarray(), columns=vectorizer.
 ↪get_feature_names_out())

# View the BoW DataFrame
print(bow_df)
```

```
      072318  10  100  1000  100x  1010  1030pm  11  1100sf  1220  …  \
0          0   0    0     0     0     0       0   0       0     0  …
1          0   0    0     0     0     0       0   0       0     0  …
2          0   0    0     0     0     0       0   0       0     0  …
```

```
3          0   0    0     0     0     0       0   0       0     0  …
4          0   0    0     0     0     0       0   0       0     0  …
…      …   ..  …    …   …   …     …     …   ..      …   …   …
3145       0   0    0     0     0     0       0   0       0     0  …
3146       0   0    0     0     0     0       0   0       0     0  …
3147       0   0    0     0     0     0       0   0       0     0  …
3148       0   0    0     0     0     0       0   0       0     0  …
3149       0   0    0     0     0     0       0   0       0     0  …

      youtubes  youve  yr  yup  zero  zigbee  zonkedout  zwave  zzzz  zzzzzzz
0            0      0   0    0     0       0          0      0     0        0
1            0      0   0    0     0       0          0      0     0        0
2            0      0   0    0     0       0          0      0     0        0
3            0      0   1    0     0       0          0      0     0        0
4            0      0   0    0     0       0          0      0     0        0
…        …   …   ..  …   …     …       …   …      …   …      …
3145         0      0   0    0     0       0          0      0     0        0
3146         0      0   0    0     0       0          0      0     0        0
3147         0      0   0    0     0       0          0      0     0        0
3148         0      0   0    0     0       0          0      0     0        0
3149         0      0   0    0     0       0          0      0     0        0

[3150 rows x 3999 columns]
```

[63]:
```
# (x) Create representation of Review Text by calculating Term Frequency and␣
 ↪Inverse Document Frequency (TF-IDF)
tfidf_vectorizer = TfidfVectorizer()
X_tfidf = tfidf_vectorizer.fit_transform(df['lemmatized_text'].apply(lambda x:␣
 ↪' '.join(x)))
```

[64]:
```
# Convert the TF-IDF matrix to a DataFrame
tfidf_df = pd.DataFrame(X_tfidf.toarray(), columns=tfidf_vectorizer.
 ↪get_feature_names_out())

# View the TF-IDF DataFrame
print(tfidf_df)
```

```
      072318   10  100  1000  100x  1010  1030pm   11  1100sf  1220  …  \
0        0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
1        0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
2        0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
3        0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
4        0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
…      …   …   …   …   …   …     …   …      …   …   …
3145     0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
3146     0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
3147     0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
3148     0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …
```

```
3149     0.0  0.0  0.0   0.0   0.0   0.0     0.0  0.0     0.0   0.0  …

      youtubes  youve        yr  yup  zero  zigbee  zonkedout  zwave  zzzz  \
0          0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
1          0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
2          0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
3          0.0    0.0  0.327532  0.0   0.0     0.0        0.0    0.0   0.0
4          0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
…          …      …        …  …   …     …       …      …   …
3145       0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
3146       0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
3147       0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
3148       0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0
3149       0.0    0.0  0.000000  0.0   0.0     0.0        0.0    0.0   0.0

      zzzzzzz
0         0.0
1         0.0
2         0.0
3         0.0
4         0.0
…         …
3145      0.0
3146      0.0
3147      0.0
3148      0.0
3149      0.0

[3150 rows x 3999 columns]
```

[ ]: