In [1]:
```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import re
import nltk
from nltk.corpus import stopwords
import string
from sklearn.metrics import classification_report
```

In [2]:
```
df = pd.read_csv("NASA.csv")
df.drop(df.columns[[1,2,4,5,6,7,8,9,10,11]], axis=1, inplace=True)

df
```

Out[2]:

	Unnamed: 0	Tweet
0	0	Here's to you, Oppy. 🥂\n\nBefore you say #Good...
1	1	Are there rivers and lakes on other worlds? Yo...
2	2	We want to hear from you!\n\nJoin our series o...
3	3	The @NASAExoplanets data hint that WASP-39 b, ...
4	4	.@NASAWebb just scored another first: a full p...
...	...	...
14105	14105	The supermoon is here! Be sure to bundle up th...
14106	14106	Ever wonder how we track supermoons 🌕 and othe...
14107	14107	A supermoon is coming! Tonight, the full Moon ...
14108	14108	Happy New Year from space! Astronauts aboard t...
14109	14109	□ Send a robot to Mars\n□ Launch @NASA_Astrona...

14110 rows × 2 columns

In [3]:
```
df['length'] = df['Tweet'].str.len()
df['Tweet'][0]
```

Out[3]:
```
"Here's to you, Oppy. 🥂\n\nBefore you say #GoodNightOppy, learn more about
our Opportunity @NASAMars rover, and how a planned 90-day mission turned int
o a 15-year journey of exploration and discovery: https://t.co/KTf5ECdUrO (h
ttps://t.co/KTf5ECdUrO) https://t.co/70f9CjxGrb" (https://t.co/70f9CjxGrb")
```

In [4]:
```
df['length'].describe()
```

Out[4]:
```
count    14110.000000
mean       239.409001
std         75.212052
min          8.000000
25%        208.000000
50%        268.000000
75%        297.000000
max        453.000000
Name: length, dtype: float64
```

Conversion of Emoji

In [5]:
```
import regex
import emoji
import html.parser as html

emoticons = [':-)', ':)', '(:', '(-:', ':))', '((:', ':-D', ':D', 'X-D', 'XD'
                ';-)',
                ';)', ';-D', ';D', '(;', '(-;', ':-(', ':(', '(:', '(-:', ':
                '=)',
                '(=', '=(', ')=', '=-O', 'O-=', ':o', 'o:', 'O:', 'O:', ':-o
                ':>',
                ':<', '^_^', '^.^', '>.>', 'T_T', 'T-T', '-.-', '*.*', '~.~'
                ':-|',
                ':->', ':-<', '$_$', '8-)', ':-P', ':-p', '=P', '=p', ':*)',


def split_count(text):
    text = html.unescape(text)
    emoji_list = []
    data = regex.findall(r'\X', text)
    for ch in data:
        if any(char in emoji.EMOJI_DATA for char in ch):
            emoji_list.append(ch)
    for word in text.split(' '):
        if word in emoticons :
            emoji_list.append(word)
    return emoji_list

text = df['Tweet']

emoji_list= []
for t in text:
    emoji_list=emoji_list+split_count(t)
from collections import Counter
print(len(emoji_list))
print(emoji_list)
```

```
7026
['🥂', '🌎', '😀', '💫', '🚀', '🌕', '🌌', '🗣', '👾', '🚀', '😊',
'🎶', '👧🏿\u200d🚀', '✨', '👧🏿\u200d🚀', '🐾', '🤩', '😈', '🙌',
'🤩', '🐾', '👨\u200d🚀', '👀', '🚀', '🪐', '✨', '🚀', '🧚', '🤩',
'😀', '❤️', '😻', '📦', '🚀', '🐾', '🚀', '🌕', '🚀', '🚀', '🖤',
'📸', '🤦\u200d♀️', '🤩', '🎃', '🤩', '🤣', '🙌', '🤩', '🚀', '🎁',
'🚀', '🕷', '🗺', '❓', '🗣', '📷', '💜', '🚀', '🔭', '✨', '🤩',
'🚀', '🚨', '🕸', '🍥', '🎙', '🎧', '😊', '💻', '😍', '🚀', '🚀',
'🚀', '🌕', '🚀', '🌕', '🖤', '🌙', '🥳', '🌕', '🌕', '🚀', '🗺',
'✈️', '🚀', '🚀', '🌕', '✨', '🔴', '🤸', '🌌', '🌌', '✨', '🎉',
'🎈', '🚬', '🚨', '🚨', '😊', '🔔', '🌡', '✨', '🚀', '✨', '🚀',
'🎇', '🚀', '🗓', '⚡', '🚀', '✨', '🔎', '👱\u200d♀️', '😊', '🪐',
'➡️', '💫', '👀', '🚀', '🤩', '🚀', '✨', '😳', '🚀', '🚨', '👾',
'✨', '🌒', '👋', '✨', '🌕', '🚀', '🚀', '🚀', '🌎', '✨', '✈️',
'🌕', '🌔', '🌳', '🔴', '📷', '✨', '🤩', '🤩', '😊', '✨', '✨',
'🚀', '😊', '🚀', '🚨', '✨', '✨', '🔔', '🌎', '✨', '🌎', '🌎',
'😊', '🤩', '💫', '❄️', '🌞', '😊', '🎫', '🪐', '✨', '⬇️', '🚀',
'⭐', '😎', '🔴', '😻', '🚨', '🐾', '🚀', '🚀', '😎', '😎', '🌑',
'😊', '💫', '🤩', '🪐', '🌞', '🌕', '😎', '🌌', '⭐', '✅', '🌟',
```

In [6]:
```
import re
from emot.emo_unicode import UNICODE_EMOJI
def convert_emojis(text):
    for emot in UNICODE_EMOJI:
        text = text.replace(emot, " ".join(UNICODE_EMOJI[emot].replace(",","""
    return text

df['Tweet'] = df['Tweet'].apply(lambda x: convert_emojis(x))
df
```

Out[6]:

	Unnamed: 0	Tweet	length
0	0	Here's to you, Oppy. clinking_glasses\n\nBefor...	245
1	1	Are there rivers and lakes on other worlds? Yo...	293
2	2	We want to hear from you!\n\nJoin our series o...	302
3	3	The @NASAExoplanets data hint that WASP-39 b, ...	175
4	4	.@NASAWebb just scored another first: a full p...	189
...	...	...	...
14105	14105	The supermoon is here! Be sure to bundle up th...	174
14106	14106	Ever wonder how we track supermoons full_moon ...	140
14107	14107	A supermoon is coming! Tonight, the full Moon ...	255
14108	14108	Happy New Year from space! Astronauts aboard t...	254
14109	14109	white_medium_square Send a robot to Mars\nwhit...	272

14110 rows × 3 columns

Data Preprocessing

In [7]:
```
df.isnull().any()
```

Out[7]:
```
Unnamed: 0    False
Tweet         False
length        False
dtype: bool
```

In [8]:
```
df['dup'] = df.duplicated(subset=None, keep='first')
del df['dup']
df
```

Out[8]:

	Unnamed: 0	Tweet	length
0	0	Here's to you, Oppy. clinking_glasses\n\nBefor...	245
1	1	Are there rivers and lakes on other worlds? Yo...	293
2	2	We want to hear from you!\n\nJoin our series o...	302
3	3	The @NASAExoplanets data hint that WASP-39 b, ...	175
4	4	.@NASAWebb just scored another first: a full p...	189
...	...	...	...
14105	14105	The supermoon is here! Be sure to bundle up th...	174
14106	14106	Ever wonder how we track supermoons full_moon ...	140
14107	14107	A supermoon is coming! Tonight, the full Moon ...	255
14108	14108	Happy New Year from space! Astronauts aboard t...	254
14109	14109	white_medium_square Send a robot to Mars\nwhit...	272

14110 rows × 3 columns

In [9]:
```
def text_lowering(text):
    text = text.lower()
    return text
df['Tweet'] = df['Tweet'].apply(lambda x: text_lowering(x))
df
```

Out[9]:

	Unnamed: 0	Tweet	length
0	0	here's to you, oppy. clinking_glasses\n\nbefor...	245
1	1	are there rivers and lakes on other worlds? yo...	293
2	2	we want to hear from you!\n\njoin our series o...	302
3	3	the @nasaexoplanets data hint that wasp-39 b, ...	175
4	4	.@nasawebb just scored another first: a full p...	189
...	...	...	...
14105	14105	the supermoon is here! be sure to bundle up th...	174
14106	14106	ever wonder how we track supermoons full_moon ...	140
14107	14107	a supermoon is coming! tonight, the full moon ...	255
14108	14108	happy new year from space! astronauts aboard t...	254
14109	14109	white_medium_square send a robot to mars\nwhit...	272

14110 rows × 3 columns

In [10]:
```
def remove_html_tags(text):
    html=re.compile(r'<.*?>')
    text = html.sub(r'',text)
    return text
df['Tweet'] = df['Tweet'].apply(lambda x: remove_html_tags(x))
df
```

Out[10]:

	Unnamed: 0	Tweet	length
0	0	here's to you, oppy. clinking_glasses\n\nbefor...	245
1	1	are there rivers and lakes on other worlds? yo...	293
2	2	we want to hear from you!\n\njoin our series o...	302
3	3	the @nasaexoplanets data hint that wasp-39 b, ...	175
4	4	.@nasawebb just scored another first: a full p...	189
...	...	...	...
14105	14105	the supermoon is here! be sure to bundle up th...	174
14106	14106	ever wonder how we track supermoons full_moon ...	140
14107	14107	a supermoon is coming! tonight, the full moon ...	255
14108	14108	happy new year from space! astronauts aboard t...	254
14109	14109	white_medium_square send a robot to mars\nwhit...	272

14110 rows × 3 columns

In [11]:
```
def replace_uderScores(tweet):
    return tweet.replace("_"," ")
def remove_url_tags(text):
    text = re.sub(r"http\S+", "",text)
    return text
df['Tweet'] = df['Tweet'].apply(lambda x: remove_url_tags(x))
df['Tweet'] = df['Tweet'].apply(lambda x: replace_uderScores(x))
df
```

Out[11]:

	Unnamed: 0	Tweet	length
0	0	here's to you, oppy. clinking glasses\n\nbefor...	245
1	1	are there rivers and lakes on other worlds? yo...	293
2	2	we want to hear from you!\n\njoin our series o...	302
3	3	the @nasaexoplanets data hint that wasp-39 b, ...	175
4	4	.@nasawebb just scored another first: a full p...	189
...	...	...	...
14105	14105	the supermoon is here! be sure to bundle up th...	174
14106	14106	ever wonder how we track supermoons full moon ...	140
14107	14107	a supermoon is coming! tonight, the full moon ...	255
14108	14108	happy new year from space! astronauts aboard t...	254
14109	14109	white medium square send a robot to mars\nwhit...	272

In [12]:
```
PUNCT_TO_REMOVE = string.punctuation
print(PUNCT_TO_REMOVE)
def remove_punctuation(text):
    return text.translate(str.maketrans('', '', PUNCT_TO_REMOVE))

df["Tweet"] = df["Tweet"].apply(lambda text: remove_punctuation(text))
df
```

```
!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~
```

Out[12]:

	Unnamed: 0	Tweet	length
0	0	heres to you oppy clinking glasses\n\nbefore y...	245
1	1	are there rivers and lakes on other worlds you...	293
2	2	we want to hear from you\n\njoin our series of...	302
3	3	the nasaexoplanets data hint that wasp39 b aka...	175
4	4	nasawebb just scored another first a full prof...	189
...	...	...	...
14105	14105	the supermoon is here be sure to bundle up the...	174
14106	14106	ever wonder how we track supermoons full moon ...	140
14107	14107	a supermoon is coming tonight the full moon wi...	255
14108	14108	happy new year from space astronauts aboard th...	254
14109	14109	white medium square send a robot to mars\nwhit...	272

14110 rows × 3 columns

In [13]:
```
import nltk
nltk.download('stopwords')
import nltk
#nltk.download('omw-1.4')
STOPWORDS = set(stopwords.words('english'))
def remove_stopwords(text):
    return " ".join([word for word in str(text).split() if word not in STOPWO
df["Tweet"] = df["Tweet"].apply(lambda text: remove_stopwords(text))
df
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\peram\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[13]:

	Unnamed: 0	Tweet	length
0	0	heres oppy clinking glasses say goodnightoppy ...	245
1	1	rivers lakes worlds bet like earth saturn’s mo...	293
2	2	want hear join series virtual inperson meeting...	302
3	3	nasaexoplanets data hint wasp39 b aka bocaprin...	175
4	4	nasawebb scored another first full profile ato...	189
...	...	...	...
14105	14105	supermoon sure bundle lead “pack” outside view...	174
14106	14106	ever wonder track supermoons full moon lunar e...	140
14107	14107	supermoon coming tonight full moon near closes...	255
14108	14108	happy new year space astronauts aboard space s...	254
14109	14109	white medium square send robot mars white medi...	272

14110 rows × 3 columns

In [14]:
```
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
import nltk
nltk.download('wordnet')
# nltk.download('omw-1.4')
nltk.download('averaged_perceptron_tagger')

lemmatizer = WordNetLemmatizer()
wordnet_map = {"N":wordnet.NOUN, "V":wordnet.VERB, "J":wordnet.ADJ, "R":wordn
def lemmatize_words(text):
    pos_tagged_text = nltk.pos_tag(nltk.word_tokenize(text))
    return " ".join([lemmatizer.lemmatize(word, wordnet_map.get(pos[0], wordn

df["Tweet"] = df["Tweet"].apply(lambda text: lemmatize_words(text))
df
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\peram\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\peram\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

Out[14]:

	Unnamed: 0	Tweet	length
0	0	here oppy clink glass say goodnightoppy learn ...	245
1	1	river lakes world bet like earth saturn ' s mo...	293
2	2	want hear join series virtual inperson meeting...	302
3	3	nasaexoplanets data hint wasp39 b aka bocaprin...	175
4	4	nasawebb score another first full profile atom...	189
...	...	...	...
14105	14105	supermoon sure bundle lead " pack " outside vi...	174
14106	14106	ever wonder track supermoons full moon lunar e...	140
14107	14107	supermoon come tonight full moon near close po...	255
14108	14108	happy new year space astronauts aboard space s...	254
14109	14109	white medium square send robot mar white mediu...	272

14110 rows × 3 columns

Adding Labels

In [15]:
```
nltk.download('vader_lexicon')

from nltk.sentiment.vader import SentimentIntensityAnalyzer
sentiments = SentimentIntensityAnalyzer()
df["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in df["Tweet"]]
df["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in df["Tweet"]]
df["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in df["Tweet"]]
df['Compound'] = [sentiments.polarity_scores(i)["compound"] for i in df["Twee
score = df["Compound"].values
sentiment = []
for i in score:
    if i > 0 :
        sentiment.append(1)
    elif i < 0 :
        sentiment.append(-1)
    else:
        sentiment.append(0)
df['Sentiment'] = sentiment
df.head()
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\peram\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[15]:

	Unnamed: 0	Tweet	length	Positive	Negative	Neutral	Compound	Sentiment
0	0	here oppy clink glass say goodnightoppy learn ...	245	0.227	0.000	0.773	0.5719	1
1	1	river lakes world bet like earth saturn ' s mo...	293	0.091	0.000	0.909	0.3612	1
2	2	want hear join series virtual inperson meeting...	302	0.137	0.000	0.863	0.3612	1
3	3	nasaexoplanets data hint wasp39 b aka bocaprin...	175	0.000	0.154	0.846	-0.4767	-1
4	4	nasawebb score another first full profile atom...	189	0.162	0.000	0.838	0.4019	1

In [16]:
```
df.drop(df.columns[[3,4,5,6]], axis=1, inplace=True)
df
```

Out[16]:

	Unnamed: 0	Tweet	length	Sentiment
0	0	here oppy clink glass say goodnightoppy learn ...	245	1
1	1	river lakes world bet like earth saturn ' s mo...	293	1
2	2	want hear join series virtual inperson meeting...	302	1
3	3	nasaexoplanets data hint wasp39 b aka bocaprin...	175	-1
4	4	nasawebb score another first full profile atom...	189	1
...	...	...	...	...
14105	14105	supermoon sure bundle lead " pack " outside vi...	174	1
14106	14106	ever wonder track supermoons full moon lunar e...	140	0
14107	14107	supermoon come tonight full moon near close po...	255	0
14108	14108	happy new year space astronauts aboard space s...	254	1
14109	14109	white medium square send robot mar white mediu...	272	0

14110 rows × 4 columns

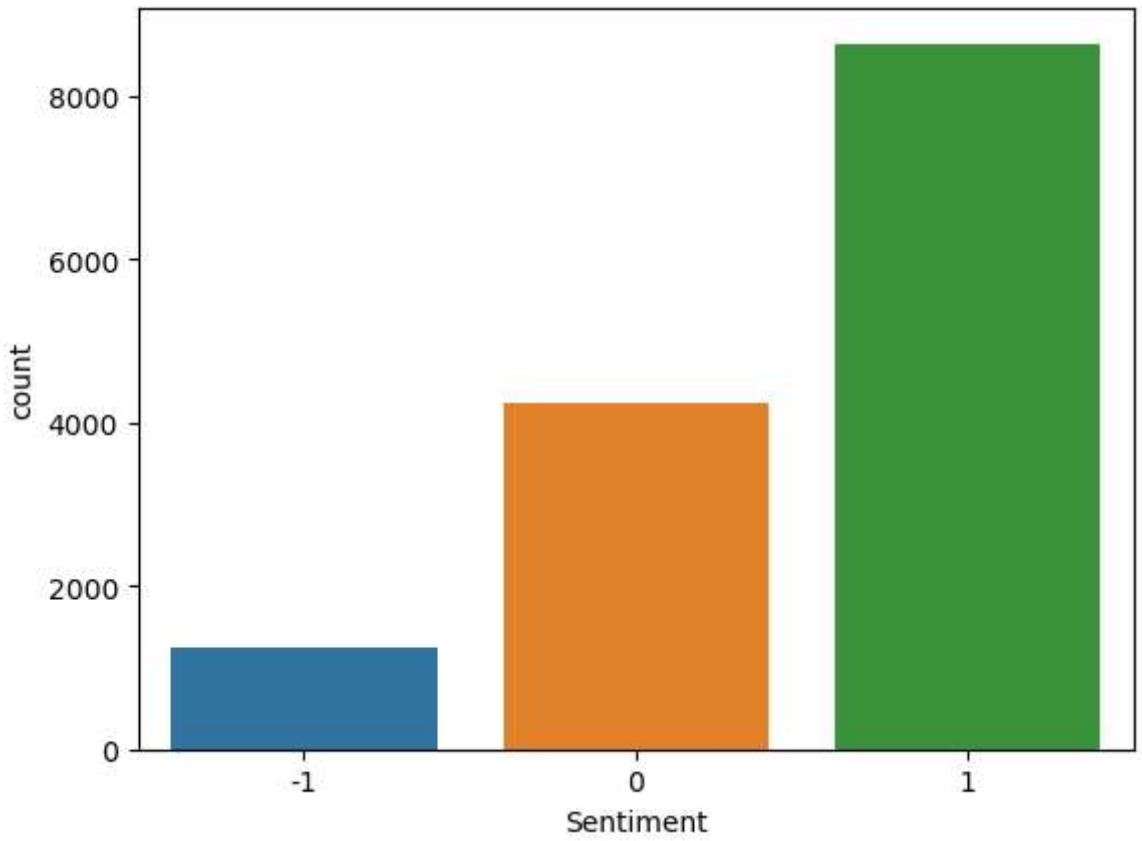
In [17]:
```
df['Sentiment'].value_counts()
```

Out[17]:
```
 1    8636
 0    4234
-1    1240
Name: Sentiment, dtype: int64
```

Data Visualization

In [18]:
```
sns.countplot(x ='Sentiment', data = df)
```

Out[18]:
```
<AxesSubplot:xlabel='Sentiment', ylabel='count'>
```



Model Building

In [19]:
```
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_df=0.90,min_df=1,max_features = 14110,stop_words='en
x = cv.fit_transform(df['Tweet'])
```

In [20]:
```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(x,df['Sentiment'],test_size=
```

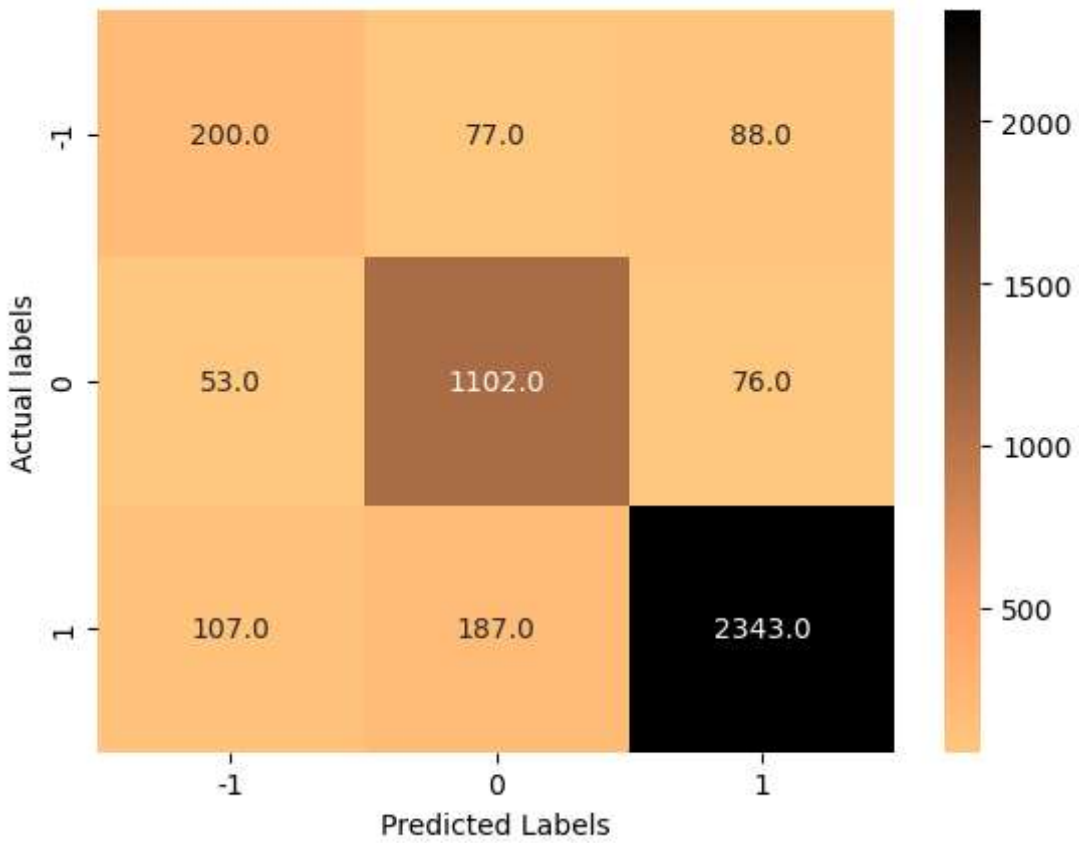
In [21]:
```
from sklearn import svm
from sklearn.metrics import accuracy_score,confusion_matrix
model1 = svm.SVC(kernel='linear', C=1)
model1.fit(X_train,y_train)
y_pred = model1.predict(X_test)
print("accuracy_score",accuracy_score(y_test,y_pred))
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm,annot=True,fmt=".1f",cmap="copper_r",xticklabels=[-1,0,1],ytic
```

```
accuracy_score 0.8610914245216159
              precision    recall  f1-score   support

          -1       0.56      0.55      0.55       365
           0       0.81      0.90      0.85      1231
           1       0.93      0.89      0.91      2637

    accuracy                           0.86      4233
   macro avg       0.77      0.78      0.77      4233
weighted avg       0.86      0.86      0.86      4233
```

Out[21]:
```
[Text(0.5, 23.52222222222222, 'Predicted Labels'),
 Text(50.722222222222214, 0.5, 'Actual labels')]
```



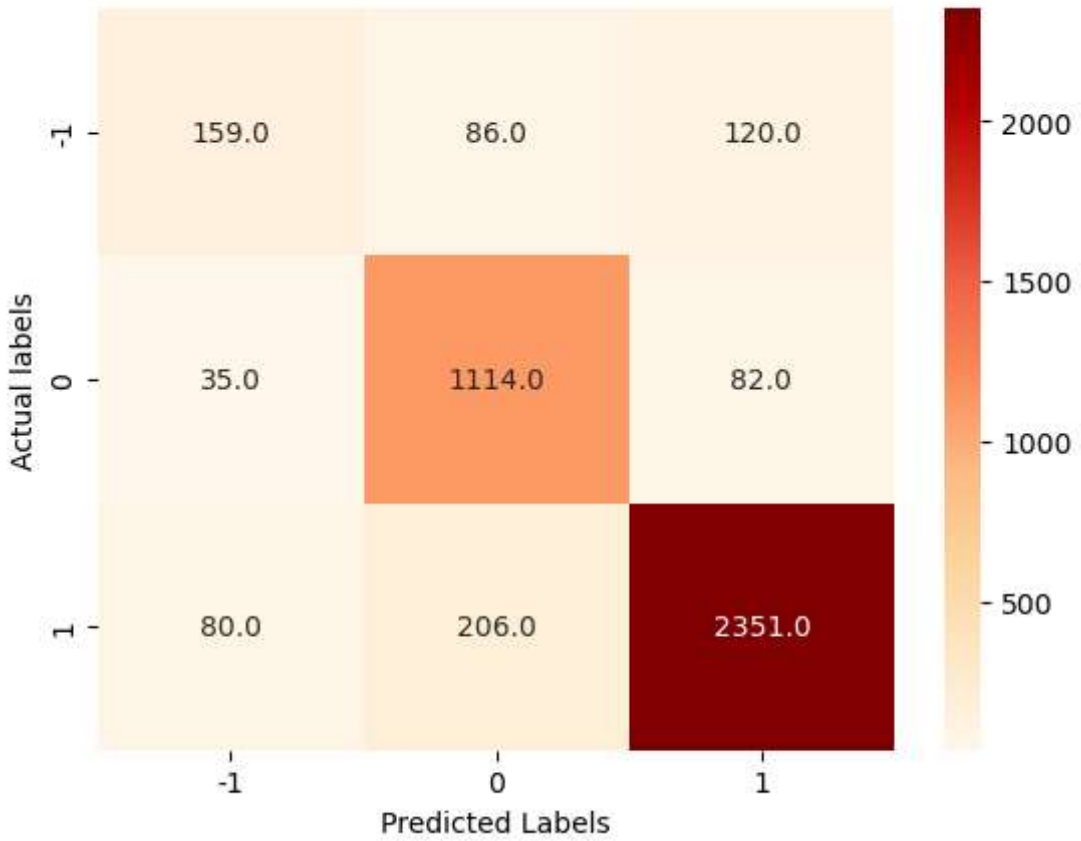
In [22]:
```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score,confusion_matrix,classification_re
model2 = DecisionTreeClassifier()
model2.fit(X_train, y_train)
y_pred = model2.predict(X_test)
print("accuracy_score",accuracy_score(y_test,y_pred))
cm = confusion_matrix(y_test, y_pred)
print(classification_report(y_test, y_pred))
sns.heatmap(cm,annot=True,fmt=".1f",cmap="OrRd",xticklabels=[-1,0,1],yticklab
```

```
accuracy_score 0.8561304039688165
              precision    recall  f1-score   support

          -1       0.58      0.44      0.50       365
           0       0.79      0.90      0.84      1231
           1       0.92      0.89      0.91      2637

    accuracy                           0.86      4233
   macro avg       0.76      0.74      0.75      4233
weighted avg       0.85      0.86      0.85      4233
```

Out[22]:
```
[Text(0.5, 23.52222222222222, 'Predicted Labels'),
 Text(50.722222222222214, 0.5, 'Actual labels')]
```



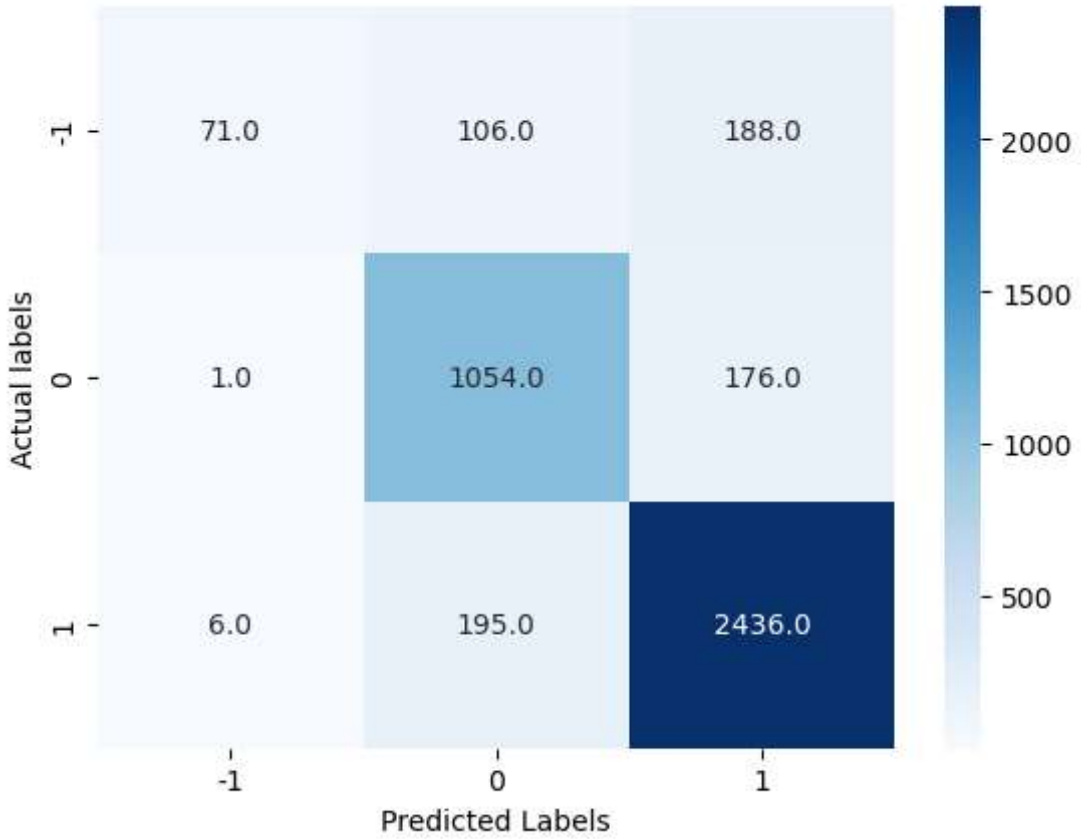
In [27]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,confusion_matrix
model3 = RandomForestClassifier()
model3.fit(X_train, y_train)
y_pred = model3.predict(X_test)
print("accuracy_score",accuracy_score(y_test,y_pred))
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm,annot=True,fmt=".1f",cmap="Blues",xticklabels=[-1,0,1],yticklal
```

```
accuracy_score 0.8412473423104181
              precision    recall  f1-score   support

          -1       0.91      0.19      0.32       365
           0       0.78      0.86      0.82      1231
           1       0.87      0.92      0.90      2637

    accuracy                           0.84      4233
   macro avg       0.85      0.66      0.68      4233
weighted avg       0.85      0.84      0.82      4233
```

Out[27]:
```
[Text(0.5, 23.52222222222222, 'Predicted Labels'),
 Text(50.722222222222214, 0.5, 'Actual labels')]
```



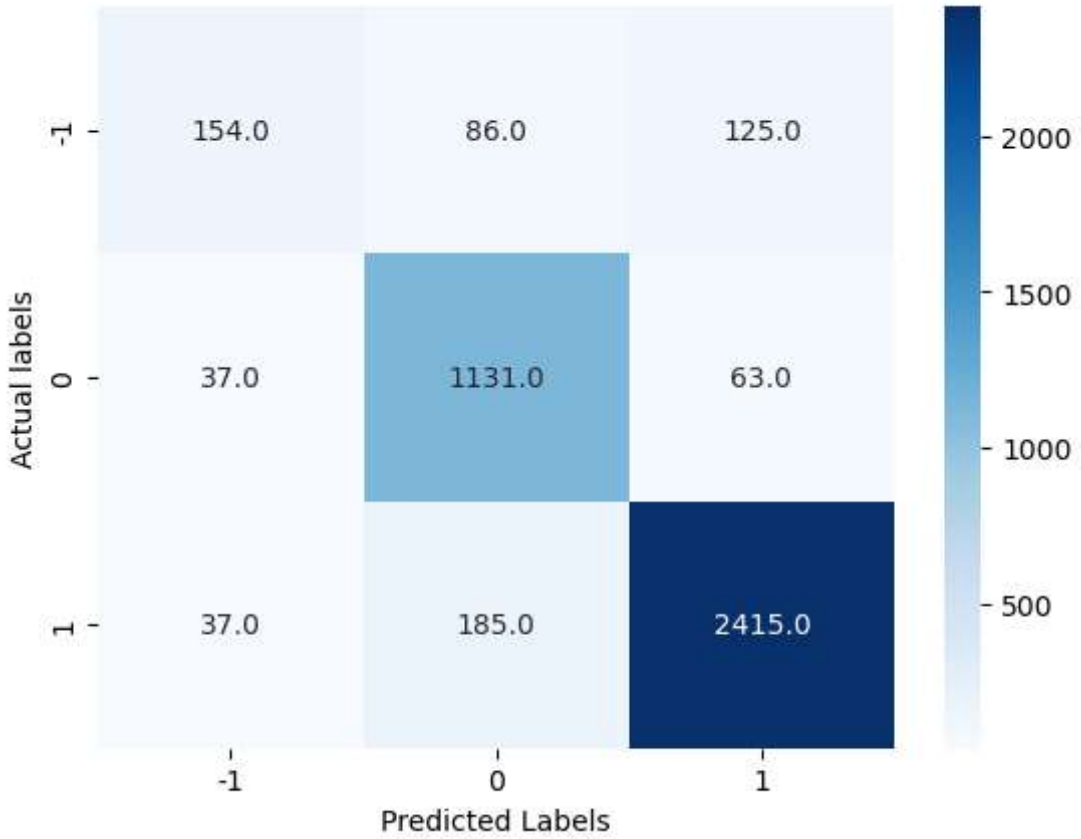
In [28]:
```
from sklearn.ensemble import VotingClassifier
from sklearn.metrics import accuracy_score
final_model = VotingClassifier(estimators=[('svm', model1), ('dt', model2), (
final_model.fit(X_train,y_train)
y_pred = final_model.predict(X_test)
print("accuracy_score",accuracy_score(y_test,y_pred))
print(classification_report(y_test, y_pred))
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm,annot=True,fmt=".1f",cmap="Blues",xticklabels=[-1,0,1],yticklal
```

```
accuracy_score 0.8740845735884716
              precision    recall  f1-score   support

          -1       0.68      0.42      0.52       365
           0       0.81      0.92      0.86      1231
           1       0.93      0.92      0.92      2637

    accuracy                           0.87      4233
   macro avg       0.80      0.75      0.77      4233
weighted avg       0.87      0.87      0.87      4233
```

Out[28]:
```
[Text(0.5, 23.52222222222222, 'Predicted Labels'),
 Text(50.722222222222214, 0.5, 'Actual labels')]
```



In [ ]:

In [ ]: