Headline

Reading Data and Data Preprocessing

Importing libraries and reading our data

In [1]:
```
#Importing necessary libraries
import numpy as np
import pandas as pd
import re
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import string
import matplotlib.ticker as mtick
import matplotlib.pyplot as plt
from gensim.models import Word2Vec, KeyedVectors
import nltk
import seaborn as sns
from sklearn import preprocessing
from imblearn.over_sampling import SMOTE
from sklearn.pipeline import Pipeline
import time
from sklearn import metrics
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.model_selection import train_test_split, StratifiedKFold
import pickle
import collections
```

In [2]:
```
#Read dataset and create a dataframe
df = pd.read_csv('Data/finaldataset.csv')
```

In [3]:
```
#Overview of our data
df
```

Out[3]:

	data	title_x	title_y	title	value	tag
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	-1	-1	2	-1	কিছুটা নেতিবাচক
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	-1	-1	-1	-1	কিছুটা নেতিবাচক
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	-1	-2	-2	-5	নিশ্চিত নেতিবাচক
3	চুরি নয় লুটপাট।	-2	-2	-2	-6	নিশ্চিত নেতিবাচক
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	0	-1	0	0	নিরপেক্ষ
...	...	...	...	...	...	...
13797	ভালভাবে নির্বাচন দেন।	0	0	0	0	নিরপেক্ষ
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	0	0	0	0	নিরপেক্ষ
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	0	0	0	0	নিরপেক্ষ
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	0	0	2	0	নিরপেক্ষ
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	0	0	-2	0	নিরপেক্ষ

13802 rows × 6 columns

Data Preprocessing

In [4]:
```
#Remove unwanted columns
df = df.drop(columns=['title_x', 'title_y','title','value'])

#Rename columns
df = df.rename(columns={"data": "comment", "tag": "annotation"})

#Overview of our data
df
```

Out[4]:

	comment	annotation
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	কিছুটা নেতিবাচক
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	কিছুটা নেতিবাচক
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	নিশ্চিত নেতিবাচক
3	চুরি নয় লুটপাট।	নিশ্চিত নেতিবাচক
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	নিরপেক্ষ
...	...	...
13797	ভালভাবে নির্বাচন দেন।	নিরপেক্ষ
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	নিরপেক্ষ
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	নিরপেক্ষ
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	নিরপেক্ষ
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	নিরপেক্ষ

13802 rows × 2 columns

In [5]:
```
#Counting corresponding values of our annotation columns
df['annotation'].value_counts()
```

Out[5]:
```
নিশ্চিত নেতিবাচক    3928
কিছুটা নেতিবাচক     3198
নিরপেক্ষ             2951
নিশ্চিত ইতিবাচক     2280
কিছুটা ইতিবাচক      1445
Name: annotation, dtype: int64
```

Here, we can see that we have 5 different annotated sentiments written in bengali. We would translate them to English for our ease.

In [6]:
```
#Translating annotation column values to English
df['annotation'] = df['annotation'].map({'নিশ্চিত নেতিবাচক': 'Negative','কিছুটা

#Overview of our data
df
```

Out[6]:

	comment	annotation
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Slightly Negative
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Slightly Negative
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative
3	চুরি নয় লুটপাট।	Negative
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral
...	...	...
13797	ভালভাবে নির্বাচন দেন।	Neutral
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	Neutral
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	Neutral
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	Neutral

13802 rows × 2 columns

In [7]:
```
#Counting corresponding values of our annotation columns
df['annotation'].value_counts()
```

Out[7]:
```
Negative            3928
Slightly Negative   3198
Neutral             2951
Positive            2280
Slightly Positive   1445
Name: annotation, dtype: int64
```

Here our dataset is dealing with 5 different kinds of sentiments, for simplicity of analysis and for better performance we would be using 3 sentiments, Positive, Negative and Neutral. We would be converting Slightly Negative and Slightly Positive to Negative and Positive sentiments repectively

In [8]:
```
#Reducing 5 sentiments to 3 sentiments
df['annotation'] = df['annotation'].map({'Slightly Negative': 'Negative','Nega

#Overview of our data
df
```

Out[8]:

	comment	annotation
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative
3	চুরি নয় লুটপাট।	Negative
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral
...	...	...
13797	ভালভাবে নির্বাচন দেন।	Neutral
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	Neutral
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	Neutral
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	Neutral

13802 rows × 2 columns

In [9]:
```
#Counting corresponding values of our annotation columns
df['annotation'].value_counts()
```

Out[9]:
```
Negative    7126
Positive    3725
Neutral     2951
Name: annotation, dtype: int64
```

Dealing with missing values

In [10]:
```
#Checking number of missing values
df.isnull().sum()
```

Out[10]:
```
comment       0
annotation    0
dtype: int64
```

We can observe that out dataset does not have any missing values

Removing puntuations, numbers, special characters etc

Few codes and preprocessing steps are inspired from the following referred website: [https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/ (https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/)](https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/)

Below code ref from data/sentence cleaning part: [https://github.com/AkashBhuiyan/sentiment-analysis-bangla-language/blob/master/Sentiment%20Analysis%20For%20Bangla%20Language.ipynb (https://github.com/AkashBhuiyan/sentiment-analysis-bangla-language/blob/master/Sentiment%20Analysis%20For%20Bangla%20Language.ipynb)](https://github.com/AkashBhuiyan/sentiment-analysis-bangla-language/blob/master/Sentiment%20Analysis%20For%20Bangla%20Language.ipynb)

In [11]:
```
#Defining funtion to remove special characters
def clean(text):

    text = re.sub('[?.`*^()!°¢܌ⱶ੍ঀ~×Ꮶꝏ:¥ᱹÍ|ǀ;!,&%\'@#$><A-Za-z0+-9=./''""_০-৯]
    text = re.sub(r'(\W)(?=\1)', '', text)
    text = re.sub(r'https?:\/\/.*[\r\n]*', '', text, flags=re.MULTILINE)
    text = re.sub(r'\<a href', ' ', text)
    text = re.sub(r'&amp;', '', text)
    text = re.sub(r'<br />', ' ', text)
    text = re.sub(r'\'', ' ', text)
    text = re.sub(r'ᱹⱶ੍ঀᱹ', '', text)
    text = re.sub(r'ɯɰɯɰɯɟ', '', text)

    text = text.strip()
    return text

#Applying the function
for i, text  in enumerate(df['comment'].tolist()):
    df.loc[i,'clean_sentence'] = clean(text)

#Some part of the above code are taken from data/sentence cleaning part of the
#https://github.com/AkashBhuiyan/sentiment-analysis-bangla-language/blob/maste
```

In [12]:
```
#Overview of our data
df
```

Out[12]:

	comment	annotation	clean_sentence
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative	এটা কেন হচ্ছে সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে ন...
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...
3	চুরি নয় লুটপাট।	Negative	চুরি নয় লুটপাট
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...
...	...	...	...
13797	ভালভাবে নির্বাচন দেন।	Neutral	ভালভাবে নির্বাচন দেন
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral	বঙ্গবন্ধুর খুনীদের পারবেন না এই মূহূর্তে অবশ্য...

Stopword removal, tokenizing, stemming

Tokenization

In [13]:
```
df['tokenized_text'] = df['clean_sentence'].apply(word_tokenize)
```

Stopword removal

ref: https://stackoverflow.com/questions/65902816/removing-stop-words-from-a-pandas-column (https://stackoverflow.com/questions/65902816/removing-stop-words-from-a-pandas-column)

In [14]:
```
#Initialize NLTK features
nltk.download('punkt')
nltk.download('stopwords')

#Bengali stop words
stop = set(stopwords.words('bengali'))

#Create new column which is tokenized and does not have stopwords
df['tokenized_text_no_stop'] = df['clean_sentence'].apply(word_tokenize)
df['tokenized_text_no_stop'] = df['tokenized_text_no_stop'].apply(lambda words

#ref: https://stackoverflow.com/questions/65902816/removing-stop-words-from-a-
```

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

In [15]:
```
#Overview of data
df
```

Out[15]:

	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative	এটা কেন হচ্ছে সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে ন...	[এটা, কেন, হচ্ছে, সংশ্লিষ্ট, সকলের, ডিপ্রেশনের...	[সংশ্লিষ্ট, সকলের, ডিপ্রেশনের, সরকার, মনোনিত, ...
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	[আমাদের, দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবা...	[দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবাহকে, বাধ...
3	চুরি নয় লুটপাট।	Negative	চুরি নয় লুটপাট	[চুরি, নয়, লুটপাট]	[চুরি, লুটপাট]
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, দেখে, মনে,...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, শাসক, জামা...
...	...	...	...	...	...
13797	ভালভাবে নির্বাচন দেন।	Neutral	ভালভাবে নির্বাচন দেন	[ভালভাবে, নির্বাচন, দেন]	[ভালভাবে, নির্বাচন]
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral	বঙ্গবন্ধুর খুনীদের পারবেন না এই মূহূর্তে অবশ্য...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, না, এই, মূহূর্তে...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, মূহূর্তে, গুরুত্...
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	Neutral	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক	[আইনকে, তার, নিজস্ব, গতিতে, চলতে, দেওয়া, হোক]	[আইনকে, নিজস্ব, গতিতে, চলতে]
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	Neutral	দেশের প্রশাসন নিরপেক্ষ না এমতাবস্থায় তারেক জিয়...	[দেশের, প্রশাসন, নিরপেক্ষ, না, এমতাবস্থায়, তার...	[দেশের, প্রশাসন, নিরপেক্ষ, এমতাবস্থায়, তারেক, ...
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	Neutral	সেই আগস্টের কারিগর বিএনপির রা আজ আমাদের গনতন্ত...	[সেই, আগস্টের, কারিগর, বিএনপির, রা, আজ, আমাদের...	[আগস্টের, কারিগর, বিএনপির, রা, গনতন্ত্রের, ছবক...

13802 rows × 5 columns

Stemmer

Install bangla stemmer

In [16]:
```
pip install bangla-stemmer
```

```
Requirement already satisfied: bangla-stemmer in g:\anaconda2\lib\site-packag
es (1.0)
Note: you may need to restart the kernel to use updated packages.

WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
```

bangla-stemmer documentation: https://libraries.io/pypi/bangla-stemmer (https://libraries.io/pypi/bangla-stemmer)

In [17]:
```
#Import Stemmer
from bangla_stemmer.stemmer import stemmer
```

In [18]:
```
#Creating new column which uses stemmer on tokenised words having stopwords re
stmr = stemmer.BanglaStemmer()
df['tokenized_text_no_stop_stem'] = df['tokenized_text_no_stop'].apply(lambda
```

```
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied first rules..
applied fourth rules..
applied second rules..
applied second rules..
applied first rules..
applied fourth rules..
applied first rules..
applied fourth rules..
```

In [19]:
```
#Creating another new column which uses stemmer on tokenised words having stop
stmr = stemmer.BanglaStemmer()
df['tokenized_text_stem'] = df['tokenized_text'].apply(lambda x: [stmr.stem(y)
```

```
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied first rules..
applied fourth rules..
applied second rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied fourth rules..
applied first rules..
applied fourth rules..
applied first rules..
```

In [20]:
```
#Overview of our data
df
```

Out[20]:

	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	tokenized
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]	[লিখ, স
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative	এটা কেন হচ্ছে সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে ন...	[এটা, কেন, হচ্ছে, সংশ্লিষ্ট, সকলের, ডিপ্রেশনের...	[সংশ্লিষ্ট, সকলের, ডিপ্রেশনের, সরকার, মনোনিত, ...	[সংশ্লিষ্ট,
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	[আমাদের, দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবা...	[দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবাহকে, বাধ...	[দেশের
3	চুরি নয় লুটপাট।	Negative	চুরি নয় লুটপাট	[চুরি, নয়, লুটপাট]	[চুরি, লুটপাট]	
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, দেখে, মনে,...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, শাসক, জামা...	[ইসলামী,
...	...	...	...	...	...	
13797	ভালভাবে নির্বাচন দেন।	Neutral	ভালভাবে নির্বাচন দেন	[ভালভাবে, নির্বাচন, দেন]	[ভালভাবে, নির্বাচন]	
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral	বঙ্গবন্ধুর খুনীদের পারবেন না এই মূহূর্তে অবশ্য...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, না, এই, মূহূর্তে...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, মূহূর্তে, গুরুত্...	[ব
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	Neutral	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক	[আইনকে, তার, নিজস্ব, গতিতে, চলতে, দেওয়া, হোক]	[আইনকে, নিজস্ব, গতিতে, চলতে]	[আই
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	Neutral	দেশের প্রশাসন নিরপেক্ষ না এমতাবস্থায় তারেক জিয়...	[দেশের, প্রশাসন, নিরপেক্ষ, না, এমতাবস্থায়, তার...	[দেশের, প্রশাসন, নিরপেক্ষ, এমতাবস্থায়, তারেক, ...	[দে

	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	tokenized
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	Neutral	সেই আগস্টের কারিগর বিএনপির রা আজ আমাদের গনতন্ত...	[সেই, আগস্টের, কারিগর, বিএনপির, রা, আজ, আমাদের...	[আগস্টের, কারিগর, বিএনপির, রা, গনতন্ত্রের, ছবক...	[আগস্ট,

13802 rows × 7 columns

Dataset Analysis and Basic Dataset Statistics

Summary of our data

In [21]:
```
#Describing data
df.describe().T
```

Out[21]:

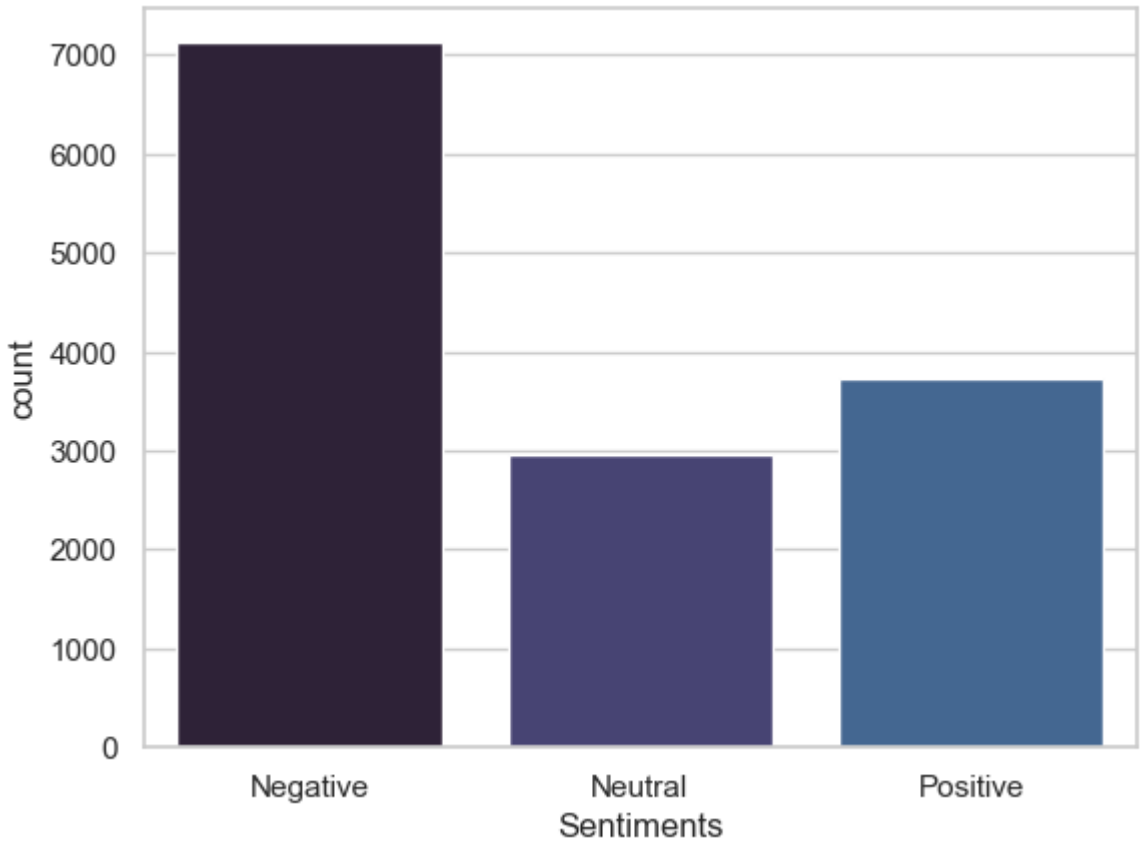
	count	unique	top	freq
comment	13802	13542	@ আন্দালীব- তৃতীয় নাম প্রকাশে অনিচ্ছুক ব্যক্তি...	2
annotation	13802	3	Negative	7126
clean_sentence	13802	13510	পাকিস্তান আজ এমনি রাষ্ট্র যার তুলনা সে নিজেই ক...	2
tokenized_text	13802	13510	[পাকিস্তান, আজ, এমনি, রাষ্ট্র, যার, তুলনা, সে,...	2
tokenized_text_no_stop	13802	13471	[]	17
tokenized_text_no_stop_stem	13802	13467	[]	17
tokenized_text_stem	13802	13510	[পাকিস্তান, আজ, এমনি, রাষ্ট্র, যার, তুলনা, সে,...	2

In [22]:
```
#Dataframe information
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 13802 entries, 0 to 13801
Data columns (total 7 columns):
 #   Column                      Non-Null Count  Dtype 
---  ------                      --------------  ----- 
 0   comment                     13802 non-null  object
 1   annotation                  13802 non-null  object
 2   clean_sentence              13802 non-null  object
 3   tokenized_text              13802 non-null  object
 4   tokenized_text_no_stop      13802 non-null  object
 5   tokenized_text_no_stop_stem 13802 non-null  object
 6   tokenized_text_stem         13802 non-null  object
dtypes: object(7)
memory usage: 754.9+ KB
```

Visualizing annotation column

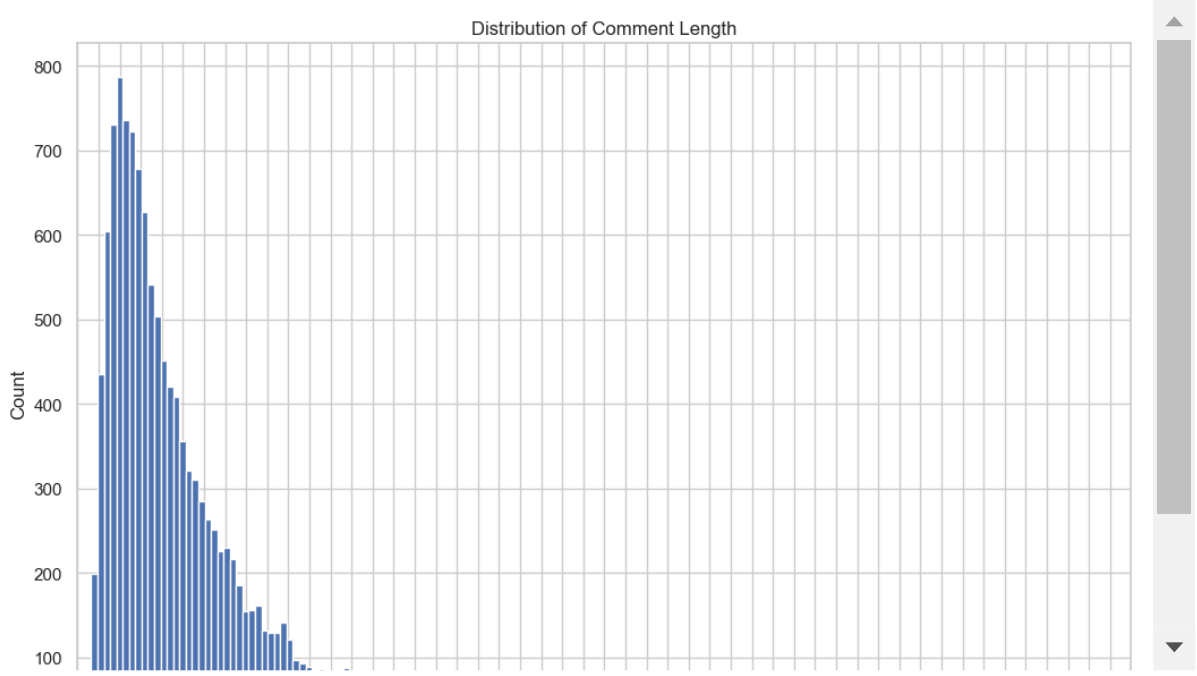
In [23]:
```
plt.figure()
#Setting similar style and color palettes throughout the analysis
sns.set(style="whitegrid", color_codes = True)
pal = sns.color_palette("mako")
#Countplot
sns.countplot(x = 'annotation', data = df, palette = pal)
plt.xlabel('Sentiments')
plt.show()
```



Finding comment lengths

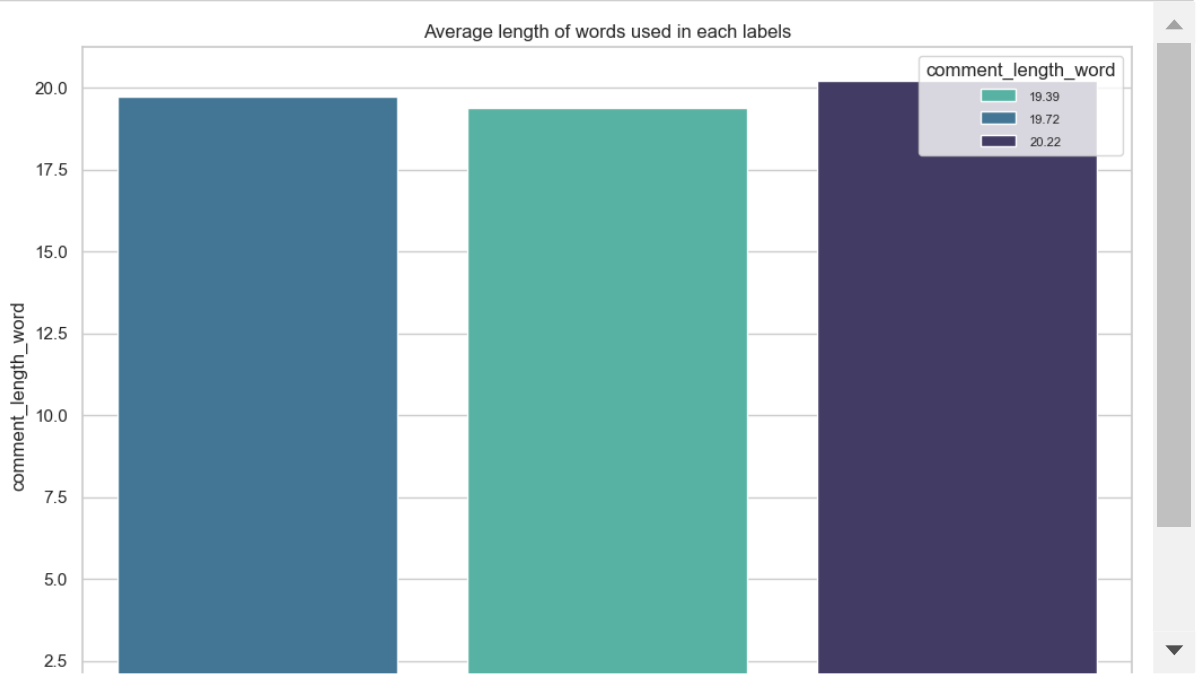
Visualizing character distribution

In [24]:
```
#Creating a new columns based on words and characters count
df["comment_length_word"] = df["comment"].apply(lambda text: len(text.split()))
df["comment_length_char"] = df["comment"].map(lambda x: len(x))


#Itilializing variable of "comment_length_char" for visualization
comment_length_char = df["comment"].map(lambda x: len(x))
plt.figure(figsize=(12,8))

#Visualizing first 1000 comment distribution
comment_length_char.loc[comment_length_char < 1000].hist(bins = 100)
plt.xlim([0, 1000])
plt.xticks(np.arange(0, 1000, 20),rotation=90)
plt.title("Distribution of Comment Length")
plt.xlabel('Comment length (Number of character)')
plt.ylabel('Count')
plt.show()
```



Visualizing average number of words used for each category

In [25]:
```
#Finding avg word lengths used for each sentiments
avg_word_len = df.groupby('annotation').aggregate({'comment_length_word':'mean
plt.figure(figsize=(12,8))

#Rounding the value
avg_word_len = avg_word_len.round({'comment_length_word': 2})

#Defining graph by colors and sorting it
pal = sns.color_palette("mako_r", len(avg_word_len.groupby("annotation").size(

#Sorting
rank = avg_word_len.groupby('comment_length_word').size().argsort().argsort()

#Ploting
ax = sns.barplot(x = avg_word_len.groupby("annotation").size().index, y = avg_
plt.title("Average length of words used in each labels")
plt.setp(ax.get_legend().get_texts(), fontsize = '8')
plt.show()
```



Dataset Basic Statistics

In [26]:
```
#Describing data
df.describe().T
```

Out[26]:

	count	mean	std	min	25%	50%	75%	max
comment_length_word	13802.0	19.786190	16.903015	1.0	8.0	14.0	25.0	125.0
comment_length_char	13802.0	116.888929	101.642777	1.0	47.0	82.0	149.0	599.0

In [27]:
```
#Count unique words
uni_word = set()
for x in df['comment']:
    word = x.split()
    for y in word:
        uni_word.add(y)
unique_word_count = len(uni_word)

#Count unique characters
uni_char = set()
for x in df['comment']:
    for char in x:
        uni_char.add(char)
unique_char_count = len(uni_char)

#Print Basic Statistics
#For Words
print('Total number of words', df['comment_length_word'].sum())
print('Total number of unique words', unique_word_count)
print('Maximum number of words used in each comment', df['comment_length_word'
print('Average number of words used in each comment', df['comment_length_word'
print('Standard deviation words in comment', df['comment_length_word'].std())

#For Characters
print('Total number of characters', df['comment_length_char'].sum())
print('Total number of unique characters', unique_char_count)
print('Maximum number of characters used in each comment', df['comment_length_
print('Average number of characters used in each comment', df['comment_length_
print('Standard deviation characters in comment', df['comment_length_char'].st
```

```
Total number of words 273089
Total number of unique words 44501
Maximum number of words used in each comment 125
Average number of words used in each comment 19.786190407187362
Standard deviation words in comment 16.903014772654615
Total number of characters 1613301
Total number of unique characters 138
Maximum number of characters used in each comment 599
Average number of characters used in each comment 116.88892914070425
Standard deviation characters in comment 101.64277723870828
```

Spliting Dataset (80% Training and 20% Testing)

In [28]:
```
#Creating a feature column by converting the lists of tokenizeest having stop
df['feature'] = df['tokenized_text_no_stop_stem']
df['feature'] = df['feature'].astype(str)
```

In [29]:
```
#Creating a feature column by converting the lists of tokenizeest having stop
df['feature_with_stop'] = df['tokenized_text_stem']
df['feature_with_stop'] = df['feature_with_stop'].astype(str)
```

In [ ]:
```

```

In [30]:
```
#Dataset overview
df
```

Out[30]:

	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	tokenized
0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]	[লিখার, সময়, পারলে, সত্য, লিখার, অভ্যাস, শিখুন]	[লিখ, স
1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative	এটা কেন হচ্ছে সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে ন...	[এটা, কেন, হচ্ছে, সংশ্লিষ্ট, সকলের, ডিপ্রেশনের...	[সংশ্লিষ্ট, সকলের, ডিপ্রেশনের, সরকার, মনোনিত, ...	[সংশ্লিষ্ট,
2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	Negative	আমাদের দেশের স্বাভাবিক অর্থনৈতিক গতিপ্রবাহকে ব...	[আমাদের, দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবা...	[দেশের, স্বাভাবিক, অর্থনৈতিক, গতিপ্রবাহকে, বাধ...	[দেশের
3	চুরি নয় লুটপাট।	Negative	চুরি নয় লুটপাট	[চুরি, নয়, লুটপাট]	[চুরি, লুটপাট]	
4	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	Neutral	ইসলামী ব্যাংকের বর্তমান অবস্থা দেখে মনে হয় শাস...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, দেখে, মনে,...	[ইসলামী, ব্যাংকের, বর্তমান, অবস্থা, শাসক, জামা...	[ইসলামী,
...	...	...	...	...	...	
13797	ভালভাবে নির্বাচন দেন।	Neutral	ভালভাবে নির্বাচন দেন	[ভালভাবে, নির্বাচন, দেন]	[ভালভাবে, নির্বাচন]	
13798	বঙ্গবন্ধুর খুনীদের পারবেন না? এই মূহূর্তে অবশ্...	Neutral	বঙ্গবন্ধুর খুনীদের পারবেন না এই মূহূর্তে অবশ্য...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, না, এই, মূহূর্তে...	[বঙ্গবন্ধুর, খুনীদের, পারবেন, মূহূর্তে, গুরুত্...	[ব
13799	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক।	Neutral	আইনকে তার নিজস্ব গতিতে চলতে দেওয়া হোক	[আইনকে, তার, নিজস্ব, গতিতে, চলতে, দেওয়া, হোক]	[আইনকে, নিজস্ব, গতিতে, চলতে]	[আই
13800	দেশের প্রশাসন নিরপেক্ষ না। এমতাবস্থায় তারেক জি...	Neutral	দেশের প্রশাসন নিরপেক্ষ না এমতাবস্থায় তারেক জিয়...	[দেশের, প্রশাসন, নিরপেক্ষ, না, এমতাবস্থায়, তার...	[দেশের, প্রশাসন, নিরপেক্ষ, এমতাবস্থায়, তারেক, ...	[দে

	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	tokenized
13801	সেই ২১ আগস্টের কারিগর বিএনপির রা আজ আমাদের গনত...	Neutral	সেই আগস্টের কারিগর বিএনপির রা আজ আমাদের গনতন্ত...	[সেই, আগস্টের, কারিগর, বিএনপির, রা, আজ, আমাদের...	[আগস্টের, কারিগর, বিএনপির, রা, গনতন্ত্রের, ছবক...	[আগস্ট,

13802 rows × 11 columns

Word/Feature extraction using TF-IDF and Word2vec

In [31]:
```
#Spliting feature and target
feature = df['feature']
target = df['annotation']
```

In [32]:
```
#Spliting feature and target
feature_with_stop = df['feature_with_stop']
target = df['annotation']
```

TF-IDF

In [33]:
```
tf_idf = TfidfVectorizer() #Initialize
X_tf_idf = tf_idf.fit_transform(feature) #Fit and transform
x_train_tf_idf, x_test_tf_idf, y_train_tf, y_test_tf = train_test_split(X_tf_i
```

In [34]:
```
tf_idf_with_stop = TfidfVectorizer() #Initialize
X_tf_idf_stop = tf_idf_with_stop.fit_transform(feature_with_stop) #Fit and tra
x_train_tf_idf_stop, x_test_tf_idf_stop, y_train_tf_stop, y_test_tf_stop = tra
```

Word2vec

Reference code of Word2Vec section of the given code:
https://github.com/BigWheel92/sentiment-analysis-using-word2vec/blob/main/code.ipynb
(https://github.com/BigWheel92/sentiment-analysis-using-word2vec/blob/main/code.ipynb)

In [35]:
```
#W2V model creation
model = Word2Vec(feature, size = 128, workers = 4, min_count = 1)
#Due to computational limitations we did not increase the number of workers. B

def perform_model(dataset):
  singleDataItemEmbedding=np.zeros(128)
  vectors = []
  for dataItem in dataset:
    wordCount = 0
    for word in dataItem:
      if word in model.wv.vocab:
        singleDataItemEmbedding = singleDataItemEmbedding+model.wv[word]
        wordCount = wordCount + 1

    singleDataItemEmbedding = singleDataItemEmbedding/wordCount
    vectors.append(singleDataItemEmbedding)
  return vectors

X_w2v = perform_model(feature)
```

In [36]:
```
X_w2v_stop = perform_model(feature_with_stop)
```

In [37]:
```
#Split into train and test
x_train_w2v, x_test_w2v, y_train_w2v, y_test_w2v = train_test_split(X_w2v, tar
```

In [38]:
```
#Split into train and test
x_train_w2v_stop, x_test_w2v_stop, y_train_w2v_stop, y_test_w2v_stop = train_t
```

Exporting train and test data using pickle

In [39]:
```
#For TF-IDF
with open('Data/x_train_tf_idf.pickle', 'wb') as out:
    pickle.dump(x_train_tf_idf, out)
with open('Data/x_test_tf_idf.pickle', 'wb') as out:
    pickle.dump(x_test_tf_idf, out)
with open('Data/y_train_tf.pickle', 'wb') as out:
    pickle.dump(y_train_tf, out)
with open('Data/y_test_tf.pickle', 'wb') as out:
    pickle.dump(y_test_tf, out)
```

In [40]:
```
#For TF-IDF with stop
with open('Data/x_train_tf_idf_stop.pickle', 'wb') as out:
    pickle.dump(x_train_tf_idf_stop, out)
with open('Data/x_test_tf_idf_stop.pickle', 'wb') as out:
    pickle.dump(x_test_tf_idf_stop, out)
with open('Data/y_train_tf_stop.pickle', 'wb') as out:
    pickle.dump(y_train_tf_stop, out)
with open('Data/y_test_tf_stop.pickle', 'wb') as out:
    pickle.dump(y_test_tf_stop, out)
```

In [41]:
```
#For W2V
with open('Data/x_train_w2v.pickle', 'wb') as out:
    pickle.dump(x_train_w2v, out)
with open('Data/x_test_w2v.pickle', 'wb') as out:
    pickle.dump(x_test_w2v, out)
with open('Data/y_train_w2v.pickle', 'wb') as out:
    pickle.dump(y_train_w2v, out)
with open('Data/y_test_w2v.pickle', 'wb') as out:
    pickle.dump(y_test_w2v, out)
```

In [42]:
```
#For W2V
with open('Data/x_train_w2v_stop.pickle', 'wb') as out:
    pickle.dump(x_train_w2v_stop, out)
with open('Data/x_test_w2v_stop.pickle', 'wb') as out:
    pickle.dump(x_test_w2v_stop, out)
with open('Data/y_train_w2v_stop.pickle', 'wb') as out:
    pickle.dump(y_train_w2v_stop, out)
with open('Data/y_test_w2v_stop.pickle', 'wb') as out:
    pickle.dump(y_test_w2v_stop, out)
```

In [47]:
```
#Export dataset
df.to_csv('Data/my_data.csv')
```

In [48]:
```
!pip freeze > requirements.txt
```

```
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
```

In [ ]:
```
```

In [1]:
```
#Importing necessary libraries
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.ticker as mtick
import matplotlib.pyplot as plt
import seaborn as sns
import torch
import torch.nn as nn
from sklearn.neural_network import MLPRegressor
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import Pipeline
import time
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
from sklearn.model_selection import train_test_split, StratifiedKFold
from joblib import dump
import pickle
```

Importing data

In [2]:
```
#TF-IDF
x_train_tf_idf = "Data/x_train_tf_idf.pickle"
with open(x_train_tf_idf, 'rb') as x:
    x_train_tf_idf = pickle.load(x)

x_test_tf_idf = "Data/x_test_tf_idf.pickle"
with open(x_test_tf_idf, 'rb') as x:
    x_test_tf_idf = pickle.load(x)

y_train_tf = "Data/y_train_tf.pickle"
with open(y_train_tf, 'rb') as x:
    y_train_tf = pickle.load(x)

y_test_tf = "Data/y_test_tf.pickle"
with open(y_test_tf, 'rb') as x:
    y_test_tf = pickle.load(x)
```

In [3]:
```
#TF-IDF with stop
x_train_tf_idf_stop = "Data/x_train_tf_idf_stop.pickle"
with open(x_train_tf_idf_stop, 'rb') as x:
    x_train_tf_idf_stop = pickle.load(x)

x_test_tf_idf_stop = "Data/x_test_tf_idf_stop.pickle"
with open(x_test_tf_idf_stop, 'rb') as x:
    x_test_tf_idf_stop = pickle.load(x)

y_train_tf_stop = "Data/y_train_tf_stop.pickle"
with open(y_train_tf_stop, 'rb') as x:
    y_train_tf_stop = pickle.load(x)

y_test_tf_stop = "Data/y_test_tf_stop.pickle"
with open(y_test_tf_stop, 'rb') as x:
    y_test_tf_stop = pickle.load(x)
```

In [4]:
```
#W2V
x_train_w2v = "Data/x_train_w2v.pickle"
with open(x_train_w2v, 'rb') as x:
    x_train_w2v = pickle.load(x)

x_test_w2v = "Data/x_test_w2v.pickle"
with open(x_test_w2v, 'rb') as x:
    x_test_w2v = pickle.load(x)

y_train_w2v = "Data/y_train_w2v.pickle"
with open(y_train_w2v, 'rb') as x:
    y_train_w2v = pickle.load(x)

y_test_w2v = "Data/y_test_w2v.pickle"
with open(y_test_w2v, 'rb') as x:
    y_test_w2v = pickle.load(x)
```

In [5]:
```
#W2V with stop
x_train_w2v_stop = "Data/x_train_w2v_stop.pickle"
with open(x_train_w2v_stop, 'rb') as x:
    x_train_w2v_stop = pickle.load(x)

x_test_w2v_stop = "Data/x_test_w2v_stop.pickle"
with open(x_test_w2v_stop, 'rb') as x:
    x_test_w2v_stop = pickle.load(x)

y_train_w2v_stop = "Data/y_train_w2v_stop.pickle"
with open(y_train_w2v_stop, 'rb') as x:
    y_train_w2v_stop = pickle.load(x)

y_test_w2v_stop = "Data/y_test_w2v_stop.pickle"
with open(y_test_w2v_stop, 'rb') as x:
    y_test_w2v_stop = pickle.load(x)
```

Baseline Model (Naive Bayes + TF-IDF)

In [6]:
```
#naive bayes without stopwords
model_naive = MultinomialNB()

start = time.time()
model_naive.fit(x_train_tf_idf, y_train_tf)
end = time.time()

print('Fitting time for Naive Bayes ', (end-start))
```

```
Fitting time for Naive Bayes  0.023221254348754883
```

In [7]:
```
#Exporting Naive bayes without stopword model
dump(model_naive, '01_NB.joblib')
```

Out[7]:
```
['01_NB.joblib']
```

In [8]:
```
#naive bayes with stopwords
model_naive_with_stop = MultinomialNB()

start = time.time()
model_naive_with_stop.fit(x_train_tf_idf_stop, y_train_tf_stop)
end = time.time()

print('Fitting time for Naive Bayes having stopwords', (end-start))
```

```
Fitting time for Naive Bayes having stopwords 0.023058414459228516
```

In [9]:
```
#Exporting Naive bayes with stopword model
dump(model_naive_with_stop, '01_NB_stop.joblib')
```

Out[9]:
```
['01_NB_stop.joblib']
```

MLP

Grid Search for MLP

In [10]:
```
#Defining 5-fold stratified cv
strat1 = StratifiedKFold(n_splits = 5, random_state = 40, shuffle = True)

#Defining sets of predefined hyperparameters
param = {'hidden_layer_sizes': [(5,5),(5,5,5)], "activation": ['logistic', "re

#Grid search
MLP = GridSearchCV(MLPClassifier(max_iter = 1000, random_state = 40), param, c
MLP.fit(x_train_w2v, y_train_w2v)

#Print best score and hyperparameters
print("Best accuracy score for base dataset: ", MLP.best_score_)
print("Hyperparameters used for best accuracy for base dataset: ", MLP.best_pa
```

```
Fitting 5 folds for each of 8 candidates, totalling 40 fits
Best accuracy score for base dataset:  0.5145369978546264
Hyperparameters used for best accuracy for base dataset:  {'activation': 'log
istic', 'hidden_layer_sizes': (5, 5, 5), 'learning_rate_init': 0.003}
```

Hyperparameter table

In [11]:
```
#Initializing pandas table
table = pd.concat([pd.DataFrame(MLP.cv_results_["params"]),pd.DataFrame(MLP.cv
                                                                        colu
#Sort table based on validation accuracy
table = table.sort_values(by = ['val_acc'], ascending = False)

#Clean the table and display it
table.reset_index(inplace = True)
table = table.drop(columns = ['index'])
table
```

Out[11]:

	activation	hidden_layer_sizes	learning_rate_init	val_acc
0	logistic	(5, 5, 5)	0.003	0.514537
1	logistic	(5, 5)	0.003	0.514084
2	logistic	(5, 5, 5)	0.030	0.513993
3	relu	(5, 5)	0.030	0.513993
4	relu	(5, 5, 5)	0.003	0.513993
5	relu	(5, 5, 5)	0.030	0.513993
6	logistic	(5, 5)	0.030	0.513087
7	relu	(5, 5)	0.003	0.511729

Train MLP + W2V

In [12]:
```
#Define model
model = MLPClassifier(max_iter= 1000, random_state= 42, activation= "logistic"
                      learning_rate_init = 0.030)

start_time = time.time() #Measure time
model.fit(x_train_w2v, y_train_w2v) #Fitting model
training_time = time.time() - start_time #Training time

#Exporting Optimised MLP model
dump(model, '02_MLP_Optimised_w2v.joblib')

#Print training time
print("Training time: ", training_time)
```

```
Training time:  3.68182110786438
```

Train MLP + TF-IDF

In [13]:
```
#Define model
model = MLPClassifier(max_iter= 1000, random_state= 42, activation= "logistic"
                      learning_rate_init = 0.030)

start_time = time.time() #Measure time
model.fit(x_train_tf_idf, y_train_tf) #Fitting model
training_time = time.time() - start_time #Training time

#Exporting Optimised MLP model
dump(model, '02_MLP_Optimised_tf_idf.joblib')

#Print training time
print("Training time: ", training_time)
```

```
Training time:  13.107040405273438
```

Decision Tree

Grid Search for Decision Tree

In [14]:
```
#Defining 5-fold stratified cv
strat2 = StratifiedKFold(n_splits = 5, random_state = 40, shuffle = True)

#Defining sets of predefined hyperparameters
param2 = {'max_depth': [2, 4, 8],
              'min_samples_leaf': [2, 8, 16]}

#Grid search
DT= DecisionTreeClassifier()
gri = GridSearchCV(DT, param2, cv = strat2, verbose = 1)
gri.fit(x_train_w2v, y_train_w2v)
#Print best score and hyperparameters
print("Best accuracy score: ", gri.best_score_)
print("Hyperparameters used for best accuracy: ", gri.best_params_)
```

```
Fitting 5 folds for each of 9 candidates, totalling 45 fits
Best accuracy score:  0.5133592565985003
Hyperparameters used for best accuracy:  {'max_depth': 2, 'min_samples_leaf':
2}
```

Hyperparameter table

In [15]:
```
#Initializing pandas table
table = pd.concat([pd.DataFrame(gri.cv_results_["params"]),pd.DataFrame(gri.cv
                                                                        colu
#Sort table based on validation accuracy
table = table.sort_values(by = ['val_acc'], ascending = False)

#Clean the table and display it
table.reset_index(inplace = True)
table = table.drop(columns = ['index'])
table
```

Out[15]:

	max_depth	min_samples_leaf	val_acc
0	2	2	0.513359
1	2	8	0.513359
2	2	16	0.513359
3	4	8	0.512635
4	4	16	0.512635
5	4	2	0.512363
6	8	2	0.483018
7	8	8	0.481117
8	8	16	0.480844

Train Decision Tree + W2V

In [16]:
```
#Define model
model = DecisionTreeClassifier(max_depth = 2, min_samples_leaf = 2)

start_time = time.time() #Measure time
model.fit(x_train_w2v, y_train_w2v) #Fitting model
training_time = time.time() - start_time #Training time

#Exporting Optimised MLP model
dump(model, '02_DT_Optimised_w2v.joblib')

#Print training time
print("Training time: ", training_time)
```

```
Training time:  0.37710070610046387
```

Train Decision Tree + TF-IDF

In [17]:
```
#Define model
model = DecisionTreeClassifier(max_depth = 2, min_samples_leaf = 2)

start_time = time.time() #Measure time
model.fit(x_train_tf_idf, y_train_tf) #Fitting model
training_time = time.time() - start_time #Training time

#Exporting Optimised MLP model
dump(model, '02_DT_Optimised_tf_idf.joblib')

#Print training time
print("Training time: ", training_time)
```

```
Training time:  0.11645627021789551
```

In [ ]:

In [ ]:

In [71]:
```
#Importing necessary libraries
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.ticker as mtick
import matplotlib.pyplot as plt
import seaborn as sns
import torch
from sklearn.metrics import classification_report
import torch.nn as nn
from sklearn.neural_network import MLPRegressor
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import GridSearchCV
from sklearn.pipeline import Pipeline
import time
from sklearn.tree import DecisionTreeClassifier
from sklearn.neural_network import MLPClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics
from sklearn.model_selection import train_test_split, StratifiedKFold
from joblib import dump
import pickle
from sklearn.metrics import accuracy_score
from joblib import load
from sklearn.metrics import confusion_matrix
```

Read Test Data

In [72]:
```
#TF-IDF
x_test_tf_idf = "Data/x_test_tf_idf.pickle"
with open(x_test_tf_idf, 'rb') as x:
    x_test_tf_idf = pickle.load(x)

y_test_tf = "Data/y_test_tf.pickle"
with open(y_test_tf, 'rb') as x:
    y_test_tf = pickle.load(x)

#TF-IDF with stop
x_test_tf_idf_stop = "Data/x_test_tf_idf_stop.pickle"
with open(x_test_tf_idf_stop, 'rb') as x:
    x_test_tf_idf_stop = pickle.load(x)

y_test_tf_stop = "Data/y_test_tf_stop.pickle"
with open(y_test_tf_stop, 'rb') as x:
    y_test_tf_stop = pickle.load(x)

#W2V
x_test_w2v = "Data/x_test_w2v.pickle"
with open(x_test_w2v, 'rb') as x:
    x_test_w2v = pickle.load(x)

y_test_w2v = "Data/y_test_w2v.pickle"
with open(y_test_w2v, 'rb') as x:
    y_test_w2v = pickle.load(x)

#W2V with stop
x_test_w2v_stop = "Data/x_test_w2v_stop.pickle"
with open(x_test_w2v_stop, 'rb') as x:
    x_test_w2v_stop = pickle.load(x)

y_test_w2v_stop = "Data/y_test_w2v_stop.pickle"
with open(y_test_w2v_stop, 'rb') as x:
    y_test_w2v_stop = pickle.load(x)
```

Testing on Baseline model (Naive Bayes + TF-IDF)

In [73]:
```
#Load model
nb = load('01_NB.joblib')
nb_stop = load('01_NB_stop.joblib')
```

NB + TF-IDF without stopwords

In [74]:
```
start_time = time.time() #Measure time
pred = nb.predict(x_test_tf_idf) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for Naive Bayes without stopwords: ", accuracy_score(y
#Print testing time
print("Testing time for Naive Bayes without stopwords:: ", testing_time)
```

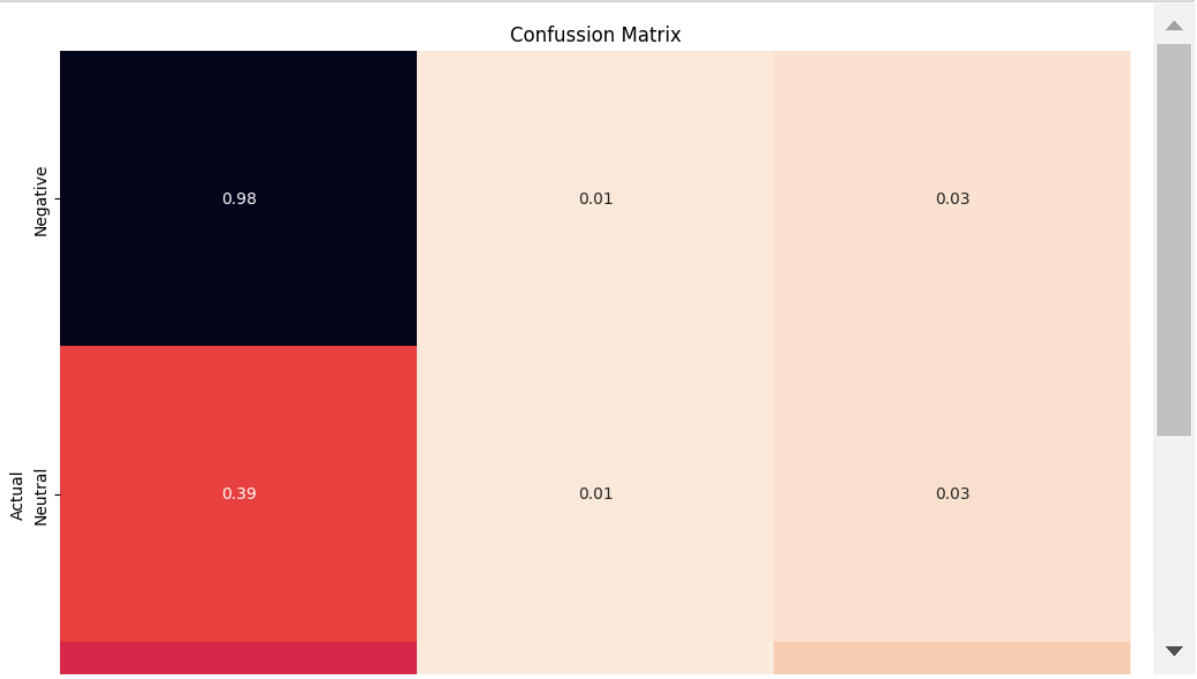
```
Testing accuracy for Naive Bayes without stopwords:  54.0021731256791
Testing time for Naive Bayes without stopwords::  0.005984306335449219
```

Confusion Matrix

In [75]:
```
print(classification_report(y_test_tf, pred))
```

```
              precision    recall  f1-score   support

    Negative       0.54      0.98      0.70      1451
     Neutral       0.50      0.01      0.02       597
    Positive       0.59      0.08      0.14       713

    accuracy                           0.54      2761
   macro avg       0.54      0.36      0.29      2761
weighted avg       0.54      0.54      0.41      2761
```

In [78]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred) #Define confussion matrix with test da
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



In [ ]:
```
```

NB + TF-IDF with stopwords

In [79]:
```
start_time = time.time() #Measure time
pred = nb_stop.predict(x_test_tf_idf_stop) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for Naive Bayes without stopwords: ", accuracy_score(y
#Print testing time
print("Testing time for Naive Bayes without stopwords:: ", testing_time)
```

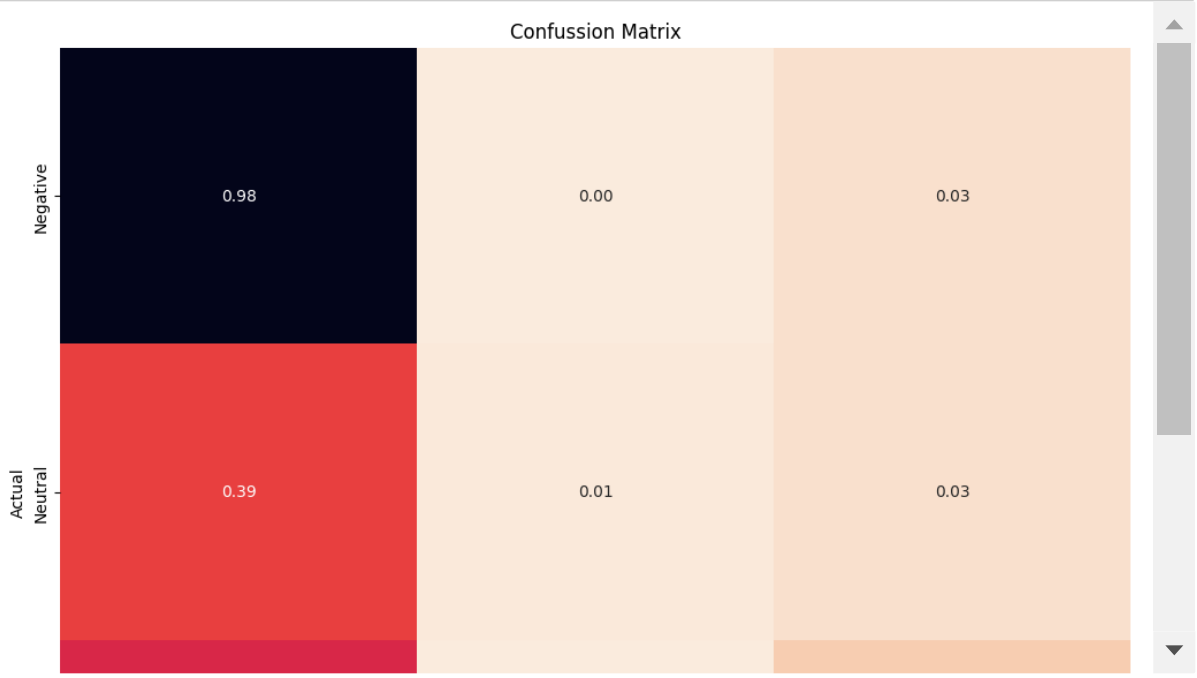
```
Testing accuracy for Naive Bayes without stopwords:  53.965954364360734
Testing time for Naive Bayes without stopwords::  0.0019638538360595703
```

Confusion Matrix

In [80]:
```
print(classification_report(y_test_tf, pred))
```

```
              precision    recall  f1-score   support

    Negative       0.54      0.98      0.70      1451
     Neutral       0.62      0.01      0.02       597
    Positive       0.57      0.08      0.14       713

    accuracy                           0.54      2761
   macro avg       0.58      0.36      0.28      2761
weighted avg       0.57      0.54      0.41      2761
```

In [81]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred) #Define confussion matrix with test da
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



Testing on MLP

MLP+W2V

In [82]:
```
#Load model
ml = load('02_MLP_Optimised_w2v.joblib')
```

In [83]:
```
start_time = time.time() #Measure time
pred3 = ml.predict(x_test_w2v) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for MLP+W2V: ", accuracy_score(y_test_w2v, pred3)*100)
#Print testing time
print("Testing time for MLP+W2V: ", testing_time)
```

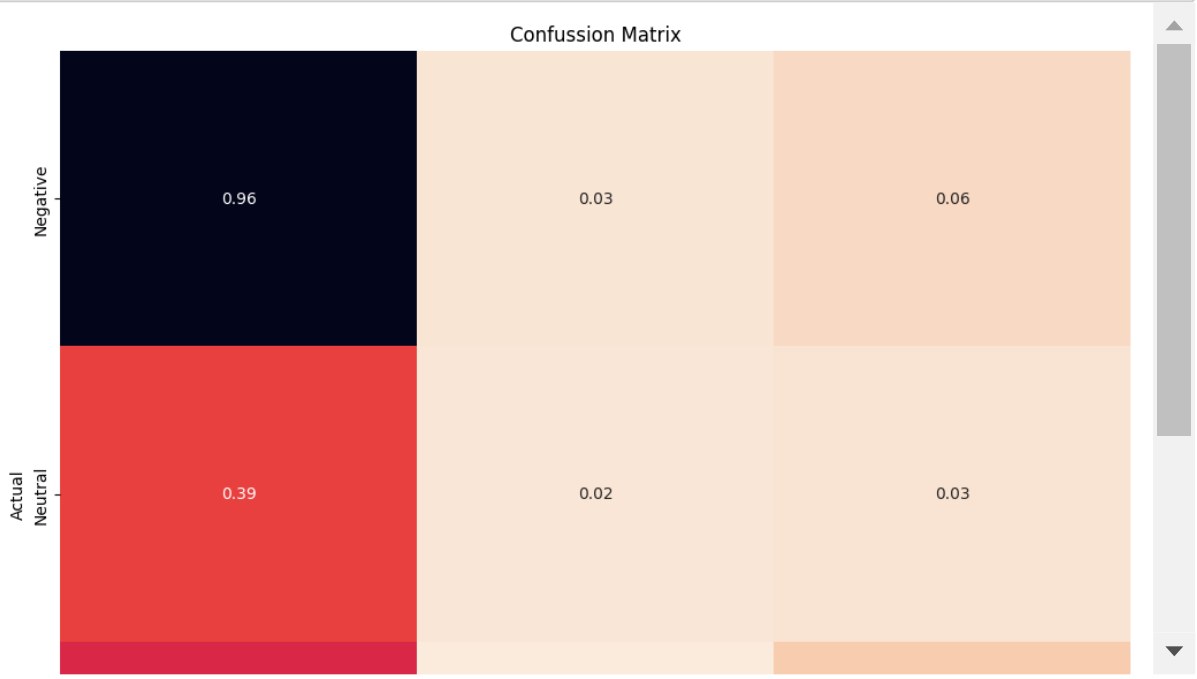
```
Testing accuracy for MLP+W2V:  53.38645418326693
Testing time for MLP+W2V:  0.012926101684570312
```

Confusion Matrix

In [84]:
```
print(classification_report(y_test_w2v, pred3))
```

```
              precision    recall  f1-score   support

    Negative       0.54      0.96      0.69      1451
     Neutral       0.40      0.02      0.04       597
    Positive       0.51      0.09      0.15       713

    accuracy                           0.53      2761
   macro avg       0.48      0.36      0.30      2761
weighted avg       0.50      0.53      0.41      2761
```

In [85]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred3) #Define confussion matrix with test a
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



MLP+TF-IDF

In [86]:
```
#Load model
ml2 = load('02_MLP_Optimised_tf_idf.joblib')
```

In [87]:
```
start_time = time.time() #Measure time
pred4 = ml2.predict(x_test_tf_idf) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for MLP+TF-IDF: ", accuracy_score(y_test_tf, pred4)*10
#Print testing time
print("Testing time for MLP+TF-IDF: ", testing_time)
```

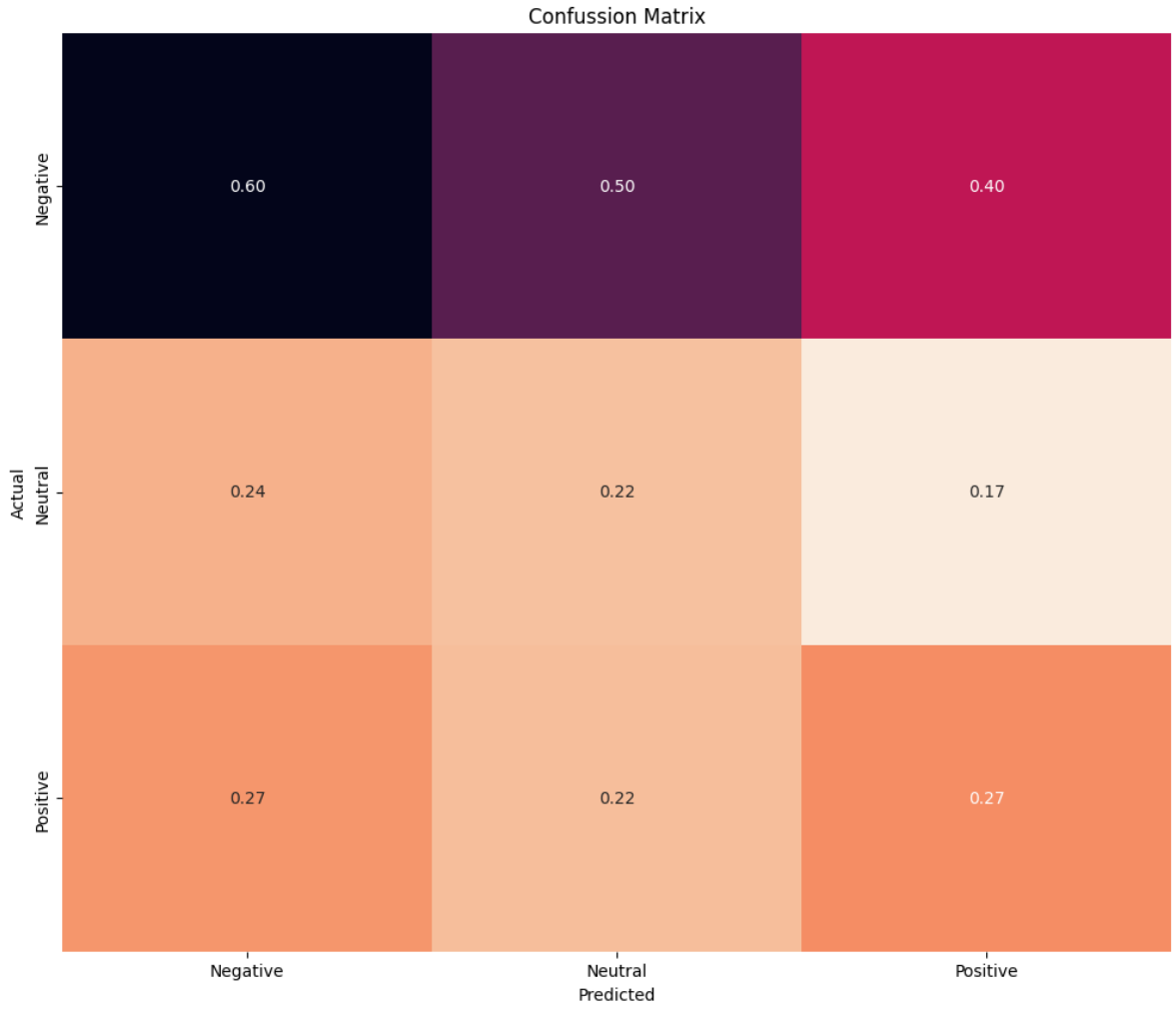
```
Testing accuracy for MLP+TF-IDF:  43.245201014125314
Testing time for MLP+TF-IDF:  0.0029752254486083984
```

Confusion Matrix

In [88]:
```
print(classification_report(y_test_tf, pred4))
```

```
              precision    recall  f1-score   support

    Negative       0.54      0.60      0.57      1451
     Neutral       0.23      0.22      0.23       597
    Positive       0.32      0.27      0.30       713

    accuracy                           0.43      2761
   macro avg       0.37      0.36      0.36      2761
weighted avg       0.42      0.43      0.42      2761
```

In [89]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred4) #Define confussion matrix with test a
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



Misclassification (MLP+TF-IDF)

In [126]:
```
y_test_tf_r = y_test_tf.reset_index(drop=True)
misclassified = []
for i in range(len(y_test_tf_r)):
    if y_test_tf_r[i] != pred4[i]:
        misclassified.append((i, pred4[i]))
```

In [127]:
```
y_test_tf_r = y_test_tf.reset_index(drop=True)
mis_lis = []
for i in range(len(y_test_tf_r)):
    if y_test_tf_r[i] != pred4[i]:
        mis_lis.append((i))
```

In [123]:
```
mis = df = pd.read_csv('Data/my_data.csv')
```

In [124]:
```
mis
```

Out[124]:

	Unnamed: 0	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_
0	0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন	['লিখার', 'সময়', 'পারলে', 'সত্য', 'লিখার', 'অভ...	['লিখার', 'সময়', 'পা 'সত্য', 'লিখার', 'ে
1	1	এটা কেন হচ্ছে? সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে?...	Negative	এটা কেন হচ্ছে সংশ্লিষ্ট সকলের ডিপ্রেশনের ফলে ন...	['এটা', 'কেন', 'হচ্ছে', 'সংশ্লিষ্ট', 'সকলের', ...	['সংশ্লিষ্ট', 'সক 'ডিপ্রেশনের', 'সরকা
2	2	আমাদের দেশের স্বাভাবিক অর্থনৈতিক	Negative	আমাদের দেশের স্বাভাবিক অর্থনৈতিক	['আমাদের', 'দেশের', 'স্বাভাবিক',	['দেশের', 'স্বাভা 'অর্থনৈতিক', 'গতিপ্র

In [128]:
```
mis = mis.loc[mis_lis]
```

In [130]:
```
misclassified
```

Out[130]:
```
[(0, 'Neutral'),
 (6, 'Neutral'),
 (9, 'Neutral'),
 (10, 'Neutral'),
 (12, 'Negative'),
 (13, 'Negative'),
 (14, 'Positive'),
 (15, 'Negative'),
 (18, 'Neutral'),
 (19, 'Negative'),
 (20, 'Neutral'),
 (23, 'Neutral'),
 (24, 'Positive'),
 (25, 'Neutral'),
 (27, 'Neutral'),
 (30, 'Neutral'),
 (33, 'Negative'),
 (34, 'Negative'),
 (35, 'Negative'),
```

In [131]:
```
mis.head(10)
```

Out[131]:

	Unnamed: 0	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	t
0	0	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন।	Negative	লিখার সময় পারলে সত্য লিখার অভ্যাস শিখুন	['লিখার', 'সময়', 'পারলে', 'সত্য', 'লিখার', 'অভ...	['লিখার', 'সময়', 'পারলে', 'সত্য', 'লিখার', 'অভ...	
6	6	সরকার যাদের এই ব্যাংকে নিয়গ দিয়েছে তারা ব্যাংক...	Negative	সরকার যাদের এই ব্যাংকে নিয়গ দিয়েছে তারা ব্যাংক...	['সরকার', 'যাদের', 'এই', 'ব্যাংকে', 'নিয়গ', 'দ...	['সরকার', 'ব্যাংকে', 'নিয়গ', 'দিয়েছে', 'ব্যাংক...	
9	9	ইসলামি ব্যাংক প্রারম্ভ থেকেই গ্রাহকদের পছন্দের...	Negative	ইসলামি ব্যাংক প্রারম্ভ থেকেই গ্রাহকদের পছন্দের...	['ইসলামি', 'ব্যাংক', 'প্রারম্ভ', 'থেকেই', 'গ্র...	['ইসলামি', 'ব্যাংক', 'প্রারম্ভ', 'গ্রাহকদের', ...	
10	10	এরা যেখানেই যাবে সেখানেই চুরি হবে।	Negative	এরা যেখানেই যাবে সেখানেই চুরি হবে	['এরা', 'যেখানেই', 'যাবে', 'সেখানেই', 'চুরি', ...	['যেখানেই', 'সেখানেই', 'চুরি']	
12	12	শেয়ার প্রতি আয় কমেছে। শূন্য হয় নি এখনও। সরকারী...	Negative	শেয়ার প্রতি আয় কমেছে শূন্য হয় নি এখনও সরকারী [...	['শেয়ার', 'প্রতি', 'আয়', 'কমেছে', 'শূন্য', 'হয়...	['শেয়ার', 'আয়', 'কমেছে', 'শূন্য', 'নি', 'সরকার...	
13	13	পুরোপুরি জামাতমুক্ত করা গেলেই লাভের মুখ দেখবে ...	Positive	পুরোপুরি জামাতমুক্ত করা গেলেই লাভের মুখ দেখবে ...	['পুরোপুরি', 'জামাতমুক্ত', 'করা', 'গেলেই', 'লা...	['পুরোপুরি', 'জামাতমুক্ত', 'গেলেই', 'লাভের', '...	
14	14	বিগত কয়েক বছরের অভিজ্ঞতা বলে ব্যাংকসহ শ্যেনদৃষ...	Negative	বিগত কয়েক বছরের অভিজ্ঞতা বলে ব্যাংকসহ শ্যেনদৃষ...	['বিগত', 'কয়েক', 'বছরের', 'অভিজ্ঞতা', 'বলে', '...	['বিগত', 'বছরের', 'অভিজ্ঞতা', 'ব্যাংকসহ', 'শ্য...	
15	15	দ্রুত জামাত প্রভাব মুক্ত করা হউক, গ্রাহকরা উপক...	Neutral	দ্রুত জামাত প্রভাব মুক্ত করা হউক গ্রাহকরা উপকৃ...	['দ্রুত', 'জামাত', 'প্রভাব', 'মুক্ত', 'করা', '...	['দ্রুত', 'জামাত', 'প্রভাব', 'মুক্ত', 'হউক', '...	
18	18	প্রবাসী বাংলাদেশিরা চলতি অর্থবছরের (২০১৮-১৯) প...	Negative	প্রবাসী বাংলাদেশিরা চলতি অর্থবছরের প্রথম মাসে ...	['প্রবাসী', 'বাংলাদেশিরা', 'চলতি', 'অর্থবছরের'...	['প্রবাসী', 'বাংলাদেশিরা', 'চলতি', 'অর্থবছরের'...	

	Unnamed: 0	comment	annotation	clean_sentence	tokenized_text	tokenized_text_no_stop	t
19	19	গরীবদের বেশী টানটান হতে নাই।	Negative	গরীবদের বেশী টানটান হতে নাই	['গরীবদের', 'বেশী', 'টানটান', 'হতে', 'নাই']	['গরীবদের', 'বেশী', 'টানটান']	

In [ ]:

Testing on Decision Tree

Decision Tree+W2V

In [90]:
```
#Load model
dt = load('02_DT_Optimised_w2v.joblib')
```

In [91]:
```
start_time = time.time() #Measure time
pred5 = dt.predict(x_test_w2v) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for DT+W2V: ", accuracy_score(y_test_w2v, pred5)*100)
#Print testing time
print("Testing time for DT+W2V: ", testing_time)
```

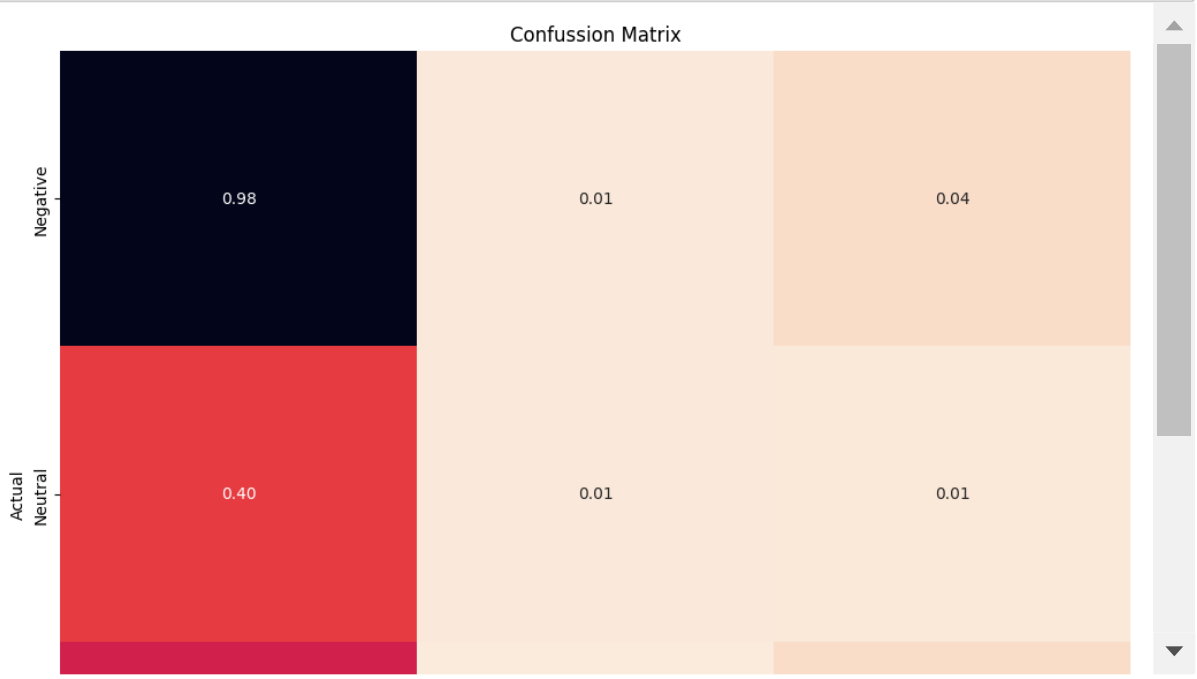
```
Testing accuracy for DT+W2V:  52.589641434262944
Testing time for DT+W2V:  0.00394439697265625
```

Confusion Matrix

In [92]:
```
print(classification_report(y_test_w2v, pred5))
```

```
              precision    recall  f1-score   support

    Negative       0.53      0.98      0.69      1451
     Neutral       0.46      0.01      0.02       597
    Positive       0.42      0.04      0.07       713

    accuracy                           0.53      2761
   macro avg       0.47      0.34      0.26      2761
weighted avg       0.49      0.53      0.38      2761
```

In [93]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred5) #Define confussion matrix with test a
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



Decision Tree+TF-IDF

In [94]:
```
#Load model
dt2 = load('02_DT_Optimised_tf_idf.joblib')
```

In [95]:
```
start_time = time.time() #Measure time
pred6 = dt2.predict(x_test_tf_idf) #Test model
testing_time = time.time() - start_time #Testing time

#Print testing accuracy
print("Testing accuracy for DT+W2V: ", accuracy_score(y_test_tf, pred6)*100)
#Print testing time
print("Testing time for DT+W2V: ", testing_time)
```

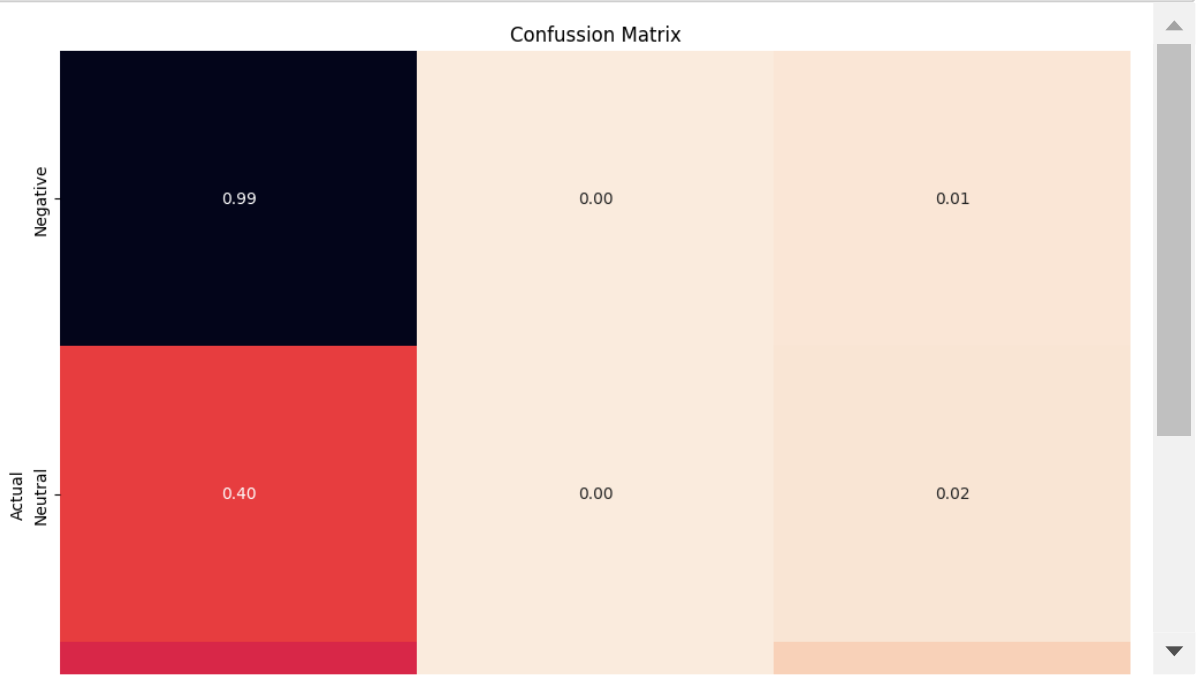
```
Testing accuracy for DT+W2V:  54.0021731256791
Testing time for DT+W2V:  0.0020236968994140625
```

Confusion Matrix

In [96]:
```
print(classification_report(y_test_tf, pred6))
```

```
              precision    recall  f1-score   support

    Negative       0.54      0.99      0.70      1451
     Neutral       0.00      0.00      0.00       597
    Positive       0.69      0.07      0.12       713

    accuracy                           0.54      2761
   macro avg       0.41      0.35      0.27      2761
weighted avg       0.46      0.54      0.40      2761


G:\Anaconda2\lib\site-packages\sklearn\metrics\_classification.py:1344: Undef
inedMetricWarning: Precision and F-score are ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
G:\Anaconda2\lib\site-packages\sklearn\metrics\_classification.py:1344: Undef
inedMetricWarning: Precision and F-score are ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
G:\Anaconda2\lib\site-packages\sklearn\metrics\_classification.py:1344: Undef
inedMetricWarning: Precision and F-score are ill-defined and being set to 0.0
in labels with no predicted samples. Use `zero_division` parameter to control
this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [97]:
```
fig = plt.figure(figsize = (12,10)) #Define figure size
mat = confusion_matrix(y_test_tf, pred6) #Define confussion matrix with test a
mat = mat/np.sum(mat, axis = 1) #Normalizing
name = ['Negative', 'Neutral', 'Positive'] #Define names of the class accordin

#Heatmap
sns.heatmap(mat, annot = True, cmap = 'rocket_r', fmt = '.2f',xticklabels=name
plt.title('Confussion Matrix')
plt.ylabel('Actual')
plt.xlabel('Predicted')

plt.show()
```



In [ ]:

In [132]:
```
!pip freeze > requirements.txt
```

```
WARNING: Ignoring invalid distribution -cipy (g:\anaconda2\lib\site-packages)
```

In [ ]: