Opinion Mining + Sentiment Classification :

For the Top 10 Indian Web Series(Horror Genre)

Getting The Data ¶

We have Web Scraped the user reviews from different OTT platforms(Amazon Prime,Netflix,ALT Balaji,ZEE5,Disney+Hotstar) for the top 10 Indian Web Series in Horror Genre, on which our further analysis are done.

In [1]:

```
import pandas as pd #for working with dataframes
```

In [2]:

```
#Reading the webscraped reviews of all the the top 10 webseries of HORROR genre.
r1_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\HORROR REVIE
r2_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\HORROR REVIE
r3_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\HORROR REVIE
r4_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\HORROR REVIE
r5_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\HORROR REVIE
```

In [3]:

```
#printing the dataframes to see the reviews
r1_df
```

Out[3]:

	Unnamed: 0
0	2/5\n\nGhoul is another offering by Netflix in...
1	Ghoul or Ghul is not a fascinating watch as a ...
2	It was a good horror miniseries, if not the be...
3	I came very late across this miniseries. And i...
4	Ghoul is a different kind of horror that makes...
...	...
514	Will shake you to the bone...
515	i will watch it
516	😶\n\nThat's my review ☝
517	Could have been better
518	More episodes when will be release \n.....

519 rows × 1 columns

In [4]:

```
r2_df
```

Out[4]:

	Unnamed: 0
0	first timer seeing indian original content - \...
1	After a very long time, I've watched a web ser...
2	Betaal is our "Made in India" project for supe...
3	Betaal is a really good series to watch on Net...
4	I thoroughly enjoyed watching it. God knows wh...
...	...
1877	No one can degrade the quality of web series t...
1878	All this series had done is tried to insult in...
1879	I loved to watch this web series
1880	Poor dialogue, no coherence, sub-par make up. ...
1881	Good story and horror scene i love it, but not...

1882 rows × 1 columns

In [5]:

```
r3_df
```

Out[5]:

	Unnamed: 0
0	Sujoy Ghosh directorial this 5 episodes series...
1	rating 2/10\n\nWASTE OF TIME.. \n\nLoosely The...
2	Stranger things ki sasti copy. (Cheap copy of...
3	Just watched Typewriter in a go. Really good p...
4	The guys at Just for Movie Freaks are the best...
...	...
344	Alright, the indian was kinda hot
345	Strainer things type adventures horror
346	Something different after long time
347	yoitttttttttttttt SUB TO PEWDS
348	Someting is special

349 rows × 1 columns

In [6]:

```
r4_df
```

Out[6]:

	Unnamed: 0
0	At first i was very excited to see 9*+ rating ...
1	Ok people who are surprised by this huge ratin...
2	I have stopped watching after par 4.....how ba...
3	Just read the top rated reviews carefully and ...
4	"Awesome , cinematography,direction" What kind...
...	...
599	You Guys Are Done Fantablous Job Keep It Up.....
600	The story of Boo -sabki phategi is fresh , in ...
601	Boo - sabki phategi is an extremely watchable ...
602	Boo -sabki phategi the story line is incredibl...
603	Boo - sabki phategi some very dark characters,...

604 rows × 1 columns

In [7]:

```
r5_df
```

Out[7]:

	Unnamed: 0
0	So finally bollywood has made its own short co...
1	Ghost stories is on another anthology of horro...
2	They should just stop making these until they ...
3	Indian anthology horror film consisting of fou...
4	A man (Sukant Goel) arrives in a small town, B...
...	...
895	Please someone explain Anurag kashyap story?
896	fuack this\n.....
897	#JusticeForSSR, #BoycottBollywood
898	Bakavass time vesting and illogical
899	Average.....

900 rows × 1 columns

In [8]:

```
#combining all the review dataframes into one dataframe
combined_df = pd.concat([r1_df, r2_df,r3_df,r4_df,r5_df], ignore_index = True)
```

In [9]:

```
combined_df
```

Out[9]:

	Unnamed: 0
0	2/5\n\nGhoul is another offering by Netflix in...
1	Ghoul or Ghul is not a fascinating watch as a ...
2	It was a good horror miniseries, if not the be...
3	I came very late across this miniseries. And i...
4	Ghoul is a different kind of horror that makes...
...	...
4249	Please someone explain Anurag kashyap story?
4250	fuack this\n.....
4251	#JusticeForSSR, #BoycottBollywood
4252	Bakavass time vesting and illogical
4253	Average.....

4254 rows × 1 columns

In [10]:

```
#naming the columns
combined_df.columns=['transcript']
```

In [11]:

```
# Let's take a look at the updated df
combined_df
```

Out[11]:

	transcript
0	2/5\n\nGhoul is another offering by Netflix in...
1	Ghoul or Ghul is not a fascinating watch as a ...
2	It was a good horror miniseries, if not the be...
3	I came very late across this miniseries. And i...
4	Ghoul is a different kind of horror that makes...
...	...
4249	Please someone explain Anurag kashyap story?
4250	fuack this\n.....
4251	#JusticeForSSR, #BoycottBollywood
4252	Bakavass time vesting and illogical
4253	Average.....

4254 rows × 1 columns

In [12]:

```
combined_df.sample(10)
```

Out[12]:

	transcript
2644	Not much justice has been done to the story an...
4172	Boooooooring!!! Really really bad.
1802	Mind blowing it was🔥🔥🔥🔥🔥🔥\nIt was very scared ...
1321	A chunk of elegant occultist claptrap! If you ...
2861	All this nonsense makes you want to watch what...
1616	if u have watched hollywood zombie and horror ...
3020	The story is good and the chemistry between th...
2511	The first episode was really dull, I could bar...
1568	Not bad not good, just lockdown to, as far as ...
149	Don't expect too much out of it! It was okay! ...

Cleaning The Data

When dealing with numerical data, data cleaning often involves removing null values and duplicate data, dealing with outliers, etc. With text data, there are some common data cleaning techniques, which are also known as text pre-processing techniques.

With text data, this cleaning process can go on forever. There's always an exception to every cleaning step. So, we're going to follow the MVP (minimum viable product) approach - start simple and iterate. Here are a bunch of things you can do to clean your data. We're going to execute just the common cleaning steps here and the rest can be done at a later point to improve our results.

Common data cleaning steps on all text:

- Make text all lower case
- Remove punctuation
- Remove numerical values
- Remove common non-sensical text (\n-new lines,\t-whitespaces etc)
- Tokenize text
- Remove stop words

More data cleaning steps after tokenization:

- Stemming / lemmatization
- Parts of speech tagging
- Create bi-grams or tri-grams
- Deal with typos
- And more...

In [13]:

```
# Applying a first round of text cleaning techniques
import re
import string

def clean_text_round1(text):
    '''Make text lowercase, remove text in square brackets, remove punctuation and remove w

    text = str(text)
    text = text.lower()

    text = re.sub('\[.*?\]', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text
```

In [14]:

```
# Let's take a look at the updated text
data_clean_df = pd.DataFrame(combined_df.transcript.apply(clean_text_round1))
data_clean_df
```

Out[14]:

	transcript
0	\n\nghoul is another offering by netflix in it...
1	ghoul or ghul is not a fascinating watch as a ...
2	it was a good horror miniseries if not the bes...
3	i came very late across this miniseries and i ...
4	ghoul is a different kind of horror that makes...
...	...
4249	please someone explain anurag kashyap story
4250	fuack this\n
4251	justiceforssr boycottbollywood
4252	bakavass time vesting and illogical
4253	average

4254 rows × 1 columns

In [15]:

```
# Apply a second round of cleaning
def clean_text_round2(text):
    '''Get rid of some additional punctuation and non-sensical text that was missed the fir
    text = str(text)
    text = re.sub('[‘’“”…]', '', text)
    text = re.sub('\n', '', text)
    return text
```

In [16]:

```
# Let's take a look at the updated text
data_clean_df = pd.DataFrame(data_clean_df.transcript.apply(clean_text_round2))
data_clean_df
```

Out[16]:

	transcript
0	ghoul is another offering by netflix in its at...
1	ghoul or ghul is not a fascinating watch as a ...
2	it was a good horror miniseries if not the bes...
3	i came very late across this miniseries and i ...
4	ghoul is a different kind of horror that makes...
...	...
4249	please someone explain anurag kashyap story
4250	fuack this
4251	justiceforssr boycottbollywood
4252	bakavass time vesting and illogical
4253	average

4254 rows × 1 columns

In [17]:

```
randomcheck=data_clean_df.loc[2694]
randomcheck
# emoji still present.
```

Out[17]:

```
transcript    good acting and relaxing stories 👍
Name: 2694, dtype: object
```

In [18]:

```
# Applying a third round of cleaning

import re
import string

text_translator = str.maketrans({ord(c): " " for c in string.punctuation})
def clean_text_round3(text, remove_punctuation_all=False):
    if not text:
        return ''
    try:
        text = text.replace(chr(160), " ")
        text = ''.join([i if ord(i) < 128 else ' ' for i in text])
    except Exception as e:
        try:
            text = text.encode('utf-8')
            text = text.decode('utf-8')
        except Exception as e:
            return ""
    try:
        text = text.encode('ascii', 'ignore').decode("utf-8")
        text = text.translate(text_translator)
    except Exception as e:
        return ""
    while '  ' in text:
        text = text.replace('  ', ' ')
    text = text.strip()
    return text
```

In [19]:

```
# Let's take a look at the updated text
data_clean_df= pd.DataFrame(data_clean_df.transcript.apply(clean_text_round3))
```

In [20]:

```
#Updated dataframe after three rounds of data cleaning
data_clean_df
```

Out[20]:

	transcript
0	ghoul is another offering by netflix in its at...
1	ghoul or ghul is not a fascinating watch as a ...
2	it was a good horror miniseries if not the bes...
3	i came very late across this miniseries and i ...
4	ghoul is a different kind of horror that makes...
...	...
4249	please someone explain anurag kashyap story
4250	fuack this
4251	justiceforssr boycottbollywood
4252	bakavass time vesting and illogical
4253	average

4254 rows × 1 columns

In [21]:

```
data_clean_df.sample(6)
```

Out[21]:

	transcript
3076	the narration way of the story from the first ...
1703	best show story is great best part is there is...
2191	its funny not scary and total time waste
1824	is this something kind of joke paly by sharukh...
2936	the cast are playing the roles very well and t...
2430	decentthe plot is gripping and there is a soli...

In [22]:

```
randomcheck=data_clean_df.loc[2694]
randomcheck
#we see that the emoji has been removed too , along with other text cleaning.
```

Out[22]:

```
transcript    good acting and relaxing stories
Name: 2694, dtype: object
```

NOTE:

This data cleaning aka text pre-processing step could go on for a while, but we are going to stop for now. After going through some analysis techniques, if you see that the results don't make sense or could be improved, you can come back and make more edits such as:

- Mark 'cheering' and 'cheer' as the same word (stemming / lemmatization)
- Combine 'thank you' into one term (bi-grams)
- And a lot more...

Exploratory Data Analysis

Introduction

After the data cleaning step where we put our data into a few standard formats, the next step is to take a look at the data and see if what we're looking at makes sense. Before applying any fancy algorithms, it's always important to explore the data first.

When working with numerical data, some of the exploratory data analysis (EDA) techniques we can use include finding the average of the data set, the distribution of the data, the most common values, etc. The idea is the same when working with text data. We are going to find some more obvious patterns with EDA before identifying the hidden patterns with machines learning (ML) techniques. Let's look at the

- Most common words - find these and create word clouds

Organizing The Data

The output of this notebook will be clean, organized data which can be done in two standard text formats:

1. Corpus - a collection of text
2. Document-Term Matrix - word counts in matrix format

Corpus

The definition of a corpus is a collection of texts, and they are all put together.

In [23]:

```
# Python program to generate WordCloud

# importing all necessary modules
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd


comment_words = ''
stopwords = set(STOPWORDS)

# iterate through the combined dataframe file
for val in data_clean_df.transcript:

    # typecaste each val to string
    val = str(val)


      # split the value
    tokens = val.split()
    comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (10,10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)

plt.show()
```

Stopwords are the English words which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence. For example, the words like the, he, have etc.

NOTE:

At this point, we could go on and continue with this word clouds. However, by looking at these top words, you can see that some of them have very little meaning and could be added to a stop words list, so let's do just that.

In [24]:

```
#present dictionary of stop words
print(stopwords)
```

```
{"shan't", 'through', 'about', 'why', "you'd", 'by', 'they', "why's", 'her
s', 'yourself', 'his', 'no', "shouldn't", 'whom', "don't", 'your', 'which',
'those', "hasn't", 'above', 'else', "hadn't", "weren't", 'but', 'himself',
'or', 'over', "we're", 'before', 'between', 'few', 'having', 'down', 'when',
"you'll", 'both', 'ever', 'ourselves', 'some', 'therefore', 'these', 'only',
'and', 'while', 'were', 'just', 'otherwise', 'who', 'be', "doesn't", "sh
e's", 'its', "i'm", 'shall', 'had', 'a', "i'd", 'r', "mustn't", 'any', "sh
e'll", "where's", 'into', 'her', "i'll", 'you', 'cannot', "what's", 'off',
'get', 'at', 'so', 'in', 'to', 'www', 'herself', 'for', 'ours', 'below', "wo
n't", 'after', 'because', 'very', "you've", 'being', 'then', "isn't", 'bee
n', 'nor', 'most', "they've", "we'll", "wouldn't", "they'll", 'themselves',
'once', "wasn't", "that's", 'there', "you're", 'here', 'itself', 'where', 'o
f', 'this', 'out', 'theirs', "they'd", 'http', "he'll", 'would', 'an', "ca
n't", 'up', "here's", "it's", 'them', "we've", "didn't", 'hence', 'again',
'against', "there's", "who's", 'other', 'as', 'me', 'their', 'yours', "sh
e'd", 'him', 'not', 'yourselves', 'what', 'same', "he'd", "couldn't", 'how',
'the', 'was', 'with', "i've", 'he', 'doing', 'myself', 'com', 'however', "ho
w's", 'during', 'my', 'also', 'if', "we'd", 'since', 'we', "let's", 'such',
"aren't", 'did', 'it', 'more', 'that', 'under', 'each', 'than', 'are', 'do',
'too', 'ought', 'am', 'she', 'until', 'from', "haven't", 'is', 'on', 'i', 'c
ould', "he's", 'has', 'k', 'should', 'does', 'our', 'can', 'further', "the
y're", 'like', 'have', 'own', 'all', "when's"}
```

In [25]:

```
#corpus of our reviews
comment_words
```

Out[25]:

```
'ghoul is another offering by netflix in its attempt to grow deeper roots
in the indian market it had created quite the hype after a trailer was dro
pped by netflixs youtube channel some months ago naturally like others i w
as immensely excited to see it being a miniseries ghoul was riddled with a
weak script from the get go with the requirement for it to be cut into epi
sodes there was little scope for characters to build up as the number of e
pisodes was limited to only three as s ghoul or ghul is not a fascinating
watch as a horrow series when a horror series tries to pack in as much sub
liminal messages into such a small run it ends up not only confusing its p
rimary goal it also ends up making a mince meat of the other goals the pac
e of the series is nice the actors are doing a fine job lets take some tim
e to appreciate ratnabali bhattacharjee mahesh balraj manav kaul and radhi
ka apte in this exact order also lets be happy that we now know a new horr
or word it was a good horror miniseries if not the best although some may
find it difficult to concieve the idea the director has struck the chords
perfectlypatrick graham has perfectly used the cgi except for some places
the cinematography and location were just mindblowing bringing in a feel o
f isolation similar to that produced by martin scorcese in his film the sh
```

In [26]:

```
# Python program to find the most frequent words from data set
from collections import Counter


# split() returns list of all the words in the string
split_it = comment_words.split(" ")


# Pass the split_it list to instance of Counter class.
Counter = Counter(split_it)

# most_common() produces k frequently encountered
# input values and their respective counts.
most_occur = Counter.most_common()

print(most_occur)
```

```
[('the', 5027), ('and', 3292), ('of', 2667), ('a', 2652), ('is', 2441),
('to', 2301), ('it', 2142), ('i', 1920), ('this', 1659), ('series', 1634),
('in', 1391), ('was', 1214), ('story', 1095), ('for', 1061), ('but', 104
4), ('good', 1027), ('horror', 968), ('not', 951), ('watch', 889), ('its',
880), ('are', 835), ('you', 814), ('with', 766), ('that', 760), ('have', 6
81), ('show', 660), ('very', 645), ('time', 633), ('like', 631), ('all', 6
23), ('one', 604), ('as', 539), ('so', 534), ('movie', 516), ('on', 514),
('indian', 507), ('be', 501), ('just', 490), ('by', 462), ('they', 453),
('really', 452), ('from', 447), ('dont', 423), ('stories', 399), ('actin
g', 388), ('will', 379), ('at', 376), ('more', 375), ('great', 373), ('som
e', 367), ('has', 361), ('best', 355), ('waste', 340), ('watching', 337),
('an', 333), ('netflix', 324), ('if', 324), ('no', 321), ('season', 316),
('there', 311), ('episode', 308), ('web', 306), ('zombie', 305), ('which',
301), ('first', 301), ('what', 300), ('were', 294), ('better', 288), ('o
r', 285), ('can', 268), ('amazing', 265), ('only', 264), ('much', 258),
('well', 258), ('betaal', 256), ('my', 255), ('after', 249), ('bad', 246),
('make', 245), ('must', 241), ('worst', 239), ('such', 232), ('even', 23
1), ('scary', 230), ('episodes', 227), ('than', 226), ('would', 225), ('ou
t', 224), ('india', 221), ('up', 216), ('zombies', 212), ('me', 210), ('mo
```

In [27]:

```
# Excluding few words from the list
# Look at the most common top words --> add them to the stop word list

add_stop_words = [word for word, count in Counter.most_common() if count > 1040]
add_stop_words
```

Out[27]:

```
['the',
 'and',
 'of',
 'a',
 'is',
 'to',
 'it',
 'i',
 'this',
 'series',
 'in',
 'was',
 'story',
 'for',
 'but']
```

In [28]:

```
#adding more stopwords for better analysis
from sklearn.feature_extraction import text
additional_stop_words = text.ENGLISH_STOP_WORDS
print (additional_stop_words)
```

```
frozenset({'why', 'whom', 'which', 'those', 'but', 'or', 'between', 'sixty',
'when', 'yet', 'detail', 'therefore', 'some', 'and', 'who', 'must', 'botto
m', 'anyone', 'her', 'hereupon', 'besides', 'becoming', 'latterly', 'someho
w', 'often', 'nor', 'therein', 'fill', 'next', 'done', 'name', 'there', 'bef
orehand', 'where', 'elsewhere', 'of', 'made', 'take', 'whose', 'up', 'well',
'still', 'hasnt', 'amongst', 'though', 'nine', 'how', 'top', 'was', 'eight',
'among', 'mine', 'ltd', 'onto', 'also', 'since', 'we', 'that', 'anywhere',
'do', 'noone', 'she', 'is', 'seeming', 'hereafter', 'has', 'something', 'som
etimes', 'nowhere', 'further', 'although', 'have', 'all', 'upon', 'through',
'no', 'might', 'five', 'system', 'mill', 'else', 'toward', 'none', 'hereby',
'few', 'down', 'behind', 'nothing', 'thence', 'ever', 'these', 'only', 'amou
ngst', 'third', 'already', 'otherwise', 'be', 'its', 'will', 'less', 'almos
t', 'due', 'sometime', 'without', 'many', 'twenty', 'cannot', 'herself', 'ne
ver', 'twelve', 'very', 'being', 'then', 'been', 'nevertheless', 'serious',
'here', 'whereafter', 'itself', 'fifteen', 're', 'nobody', 'meanwhile', 'thr
oughout', 'show', 'an', 'co', 'again', 'see', 'other', 'their', 'yours', 'hi
m', 'whence', 'whither', 'interest', 'what', 'same', 'the', 'find', 'con',
'may', 'it', 'more', 'put', 'under', 'each', 'thereafter', 'seems', 'acros
s', 'three', 'please', 'per', 'became', 'ten', 'could', 'i', 'another', 'emp
ty', 'can', 'within', 'four', 'own', 'thick', 'about', 'hers', 'yourself',
'his', 'us', 'himself', 'become', 'over', 'before', 'both', 'ourselves', 'wh
ile', 'formerly', 'together', 'around', 'except', 'either', 'had', 'anythin
g', 'back', 'you', 'now', 'former', 'move', 'get', 'off', 'in', 'to', 'six',
'whereupon', 'anyway', 'latter', 'first', 'amount', 'everywhere', 'once', 's
eemed', 'full', 'sincere', 'out', 'whatever', 'them', 'hence', 'keep', 'as',
'eleven', 'inc', 'not', 'two', 'yourselves', 'becomes', 'everything', 'wit
h', 'indeed', 'he', 'thereupon', 'cant', 'myself', 'fifty', 'wherever', 'i
f', 'somewhere', 'such', 'fire', 'than', 'are', 'thus', 'until', 'via', 'per
haps', 'front', 'mostly', 'others', 'etc', 'by', 'they', 'afterwards', 'bil
l', 'un', 'your', 'whenever', 'much', 'above', 'couldnt', 'least', 'part',
'anyhow', 'even', 'herein', 'describe', 'whereby', 'were', 'several', 'thi
n', 'towards', 'along', 'a', 'always', 'de', 'call', 'any', 'into', 'seem',
'everyone', 'alone', 'at', 'so', 'thereby', 'for', 'forty', 'cry', 'ours',
'below', 'after', 'because', 'side', 'found', 'most', 'someone', 'whole', 'b
eside', 'themselves', 'give', 'thru', 'eg', 'moreover', 'this', 'go', 'whoev
er', 'hundred', 'whether', 'would', 'neither', 'one', 'against', 'me', 'ever
y', 'whereas', 'however', 'my', 'during', 'beyond', 'ie', 'too', 'last', 'a
m', 'from', 'on', 'wherein', 'should', 'our', 'enough', 'rather', 'namely'})
```

In [29]:

```
# Python program to generate WordCloud

# importing all necessary modules
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd


comment_words = ''

# Add new stop words

selected_stop_words=['show','season','one','season','watch','story','bajpayee','webserie','
stopwords = list(additional_stop_words) + selected_stop_words + add_stop_words + list(STOPW

# iterate through the file
for val in data_clean_df.transcript:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()
    comment_words += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words)

# plot the WordCloud image
plt.figure(figsize = (10,10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.title('Sentiment WorldCloud\n',fontsize=35)
plt.show()
```

Sentiment WorldCloud

In [30]:

```
#all the stopwords that were used
print (stopwords)
```

```
['why', 'whom', 'which', 'those', 'but', 'or', 'between', 'sixty', 'when',
'yet', 'detail', 'therefore', 'some', 'and', 'who', 'must', 'bottom', 'anyon
e', 'her', 'hereupon', 'besides', 'becoming', 'latterly', 'somehow', 'ofte
n', 'nor', 'therein', 'fill', 'next', 'done', 'name', 'there', 'beforehand',
'where', 'elsewhere', 'of', 'made', 'take', 'whose', 'up', 'well', 'still',
'hasnt', 'amongst', 'though', 'nine', 'how', 'top', 'was', 'eight', 'among',
'mine', 'ltd', 'onto', 'also', 'since', 'we', 'that', 'anywhere', 'do', 'noo
ne', 'she', 'is', 'seeming', 'hereafter', 'has', 'something', 'sometimes',
'nowhere', 'further', 'although', 'have', 'all', 'upon', 'through', 'no', 'm
ight', 'five', 'system', 'mill', 'else', 'toward', 'none', 'hereby', 'few',
'down', 'behind', 'nothing', 'thence', 'ever', 'these', 'only', 'amoungst',
'third', 'already', 'otherwise', 'be', 'its', 'will', 'less', 'almost', 'du
e', 'sometime', 'without', 'many', 'twenty', 'cannot', 'herself', 'never',
'twelve', 'very', 'being', 'then', 'been', 'nevertheless', 'serious', 'her
e', 'whereafter', 'itself', 'fifteen', 're', 'nobody', 'meanwhile', 'through
out', 'show', 'an', 'co', 'again', 'see', 'other', 'their', 'yours', 'him',
'whence', 'whither', 'interest', 'what', 'same', 'the', 'find', 'con', 'ma
y', 'it', 'more', 'put', 'under', 'each', 'thereafter', 'seems', 'across',
'three', 'please', 'per', 'became', 'ten', 'could', 'i', 'another', 'empty',
'can', 'within', 'four', 'own', 'thick', 'about', 'hers', 'yourself', 'his',
'us', 'himself', 'become', 'over', 'before', 'both', 'ourselves', 'while',
'formerly', 'together', 'around', 'except', 'either', 'had', 'anything', 'ba
ck', 'you', 'now', 'former', 'move', 'get', 'off', 'in', 'to', 'six', 'where
upon', 'anyway', 'latter', 'first', 'amount', 'everywhere', 'once', 'seeme
d', 'full', 'sincere', 'out', 'whatever', 'them', 'hence', 'keep', 'as', 'el
even', 'inc', 'not', 'two', 'yourselves', 'becomes', 'everything', 'with',
'indeed', 'he', 'thereupon', 'cant', 'myself', 'fifty', 'wherever', 'if', 's
omewhere', 'such', 'fire', 'than', 'are', 'thus', 'until', 'via', 'perhaps',
'front', 'mostly', 'others', 'etc', 'by', 'they', 'afterwards', 'bill', 'u
n', 'your', 'whenever', 'much', 'above', 'couldnt', 'least', 'part', 'anyho
w', 'even', 'herein', 'describe', 'whereby', 'were', 'several', 'thin', 'tow
ards', 'along', 'a', 'always', 'de', 'call', 'any', 'into', 'seem', 'everyon
e', 'alone', 'at', 'so', 'thereby', 'for', 'forty', 'cry', 'ours', 'below',
'after', 'because', 'side', 'found', 'most', 'someone', 'whole', 'beside',
'themselves', 'give', 'thru', 'eg', 'moreover', 'this', 'go', 'whoever', 'hu
ndred', 'whether', 'would', 'neither', 'one', 'against', 'me', 'every', 'whe
reas', 'however', 'my', 'during', 'beyond', 'ie', 'too', 'last', 'am', 'fro
m', 'on', 'wherein', 'should', 'our', 'enough', 'rather', 'namely', 'show',
'season', 'one', 'season', 'watch', 'story', 'bajpayee', 'webserie', 'webser
ies', 'bajpai', 'manoj', 'episode', 'review', 'actor', 'actors', 'the', 'an
d', 'of', 'a', 'is', 'to', 'it', 'i', 'this', 'series', 'in', 'was', 'stor
y', 'for', 'but', "shan't", 'through', 'about', 'why', "you'd", 'by', 'the
y', "why's", 'hers', 'yourself', 'his', 'no', "shouldn't", 'whom', "don't",
'your', 'which', 'those', "hasn't", 'above', 'else', "hadn't", "weren't", 'b
ut', 'himself', 'or', 'over', "we're", 'before', 'between', 'few', 'having',
'down', 'when', "you'll", 'both', 'ever', 'ourselves', 'some', 'therefore',
'these', 'only', 'and', 'while', 'were', 'just', 'otherwise', 'who', 'be',
"doesn't", "she's", 'its', "i'm", 'shall', 'had', 'a', "i'd", 'r', "must
n't", 'any', "she'll", "where's", 'into', 'her', "i'll", 'you', 'cannot', "w
hat's", 'off', 'get', 'at', 'so', 'in', 'to', 'www', 'herself', 'for', 'our
s', 'below', "won't", 'after', 'because', 'very', "you've", 'being', 'then',
"isn't", 'been', 'nor', 'most', "they've", "we'll", "wouldn't", "they'll",
'themselves', 'once', "wasn't", "that's", 'there', "you're", 'here', 'itsel
f', 'where', 'of', 'this', 'out', 'theirs', "they'd", 'http', "he'll", 'woul
d', 'an', "can't", 'up', "here's", "it's", 'them', "we've", "didn't", 'henc
e', 'again', 'against', "there's", "who's", 'other', 'as', 'me', 'their', 'y
```

```
ours', "she'd", 'him', 'not', 'yourselves', 'what', 'same', "he'd", "could
n't", 'how', 'the', 'was', 'with', "i've", 'he', 'doing', 'myself', 'com',
'however', "how's", 'during', 'my', 'also', 'if', "we'd", 'since', 'we', "le
t's", 'such', "aren't", 'did', 'it', 'more', 'that', 'under', 'each', 'tha
n', 'are', 'do', 'too', 'ought', 'am', 'she', 'until', 'from', "haven't", 'i
s', 'on', 'i', 'could', "he's", 'has', 'k', 'should', 'does', 'our', 'can',
'further', "they're", 'like', 'have', 'own', 'all', "when's"]
```

Findings

We can clearly see that the word cloud has major chunk of positve reviews(roughly 60%) , some negative reviews (roughly 25%), with some neutral reviews(15%).

Let's dig into that and continue our analysis to back it up with statistical data.

Side Note

What was our goal for the EDA portion? To be able to take an initial look at our data and see if the results of some basic analysis made sense.

Guess what? Yes,now it does, for a first pass. There are definitely some things that could be better cleaned up, such as adding more stop words or including bi-grams. But we can save that for another day. The results, especially to our objective make general sense, so we're going to move on.

As a reminder, the data science process is an interative one. It's better to see some non-perfect but acceptable results to help you quickly decide whether your project is inoperative or not.

Sentiment Analysis

Introduction

So far, all of the analysis we've done has been pretty generic - looking at counts, creating wordcloud plots, etc. These techniques could be applied to numeric data as well.

When it comes to text data, there are a few popular techniques that we may go through, starting with sentiment analysis. A few key points to remember with sentiment analysis.

1. **TextBlob Module:** Linguistic researchers have labeled the sentiment of words based on their domain expertise. Sentiment of words can vary based on where it is in a sentence. The TextBlob module allows us to take advantage of these labels.
2. **Sentiment Labels:** Each word in a corpus is labeled in terms of polarity and subjectivity (there are more labels as well, but we're going to ignore them for now). A corpus' sentiment is the average of these.

- **Polarity:** How positive or negative a word is. -1 is very negative. +1 is very positive.
- **Subjectivity:** How subjective, or opinionated a word is. 0 is fact. +1 is very much an opinion.

For more info on how TextBlob coded up its sentiment function.(https://planspace.org/20150607-textblob_sentiment/ (https://planspace.org/20150607-textblob_sentiment/))

Let's take a look at the sentiment of the various transcripts.

In [31]:

```
# Create quick lambda functions to find the polarity and subjectivity of each routine

from textblob import TextBlob

pol = lambda x: TextBlob(str(x)).sentiment.polarity
sub = lambda x: TextBlob(str(x)).sentiment.subjectivity

# Another way of writing the code , instead of using lambda parameter above.
'''
def get_Subjectivity(text):
  return TextBlob(text).sentiment.subjectivity
def get_Polarity(text):
  return TextBlob(text).sentiment.polarity

  '''

data_clean_df['polarity'] = data_clean_df['transcript'].apply(pol)
data_clean_df['subjectivity'] = data_clean_df['transcript'].apply(sub)
data_clean_df
```

Out[31]:

	transcript	polarity	subjectivity
0	ghoul is another offering by netflix in its at...	-0.026488	0.569643
1	ghoul or ghul is not a fascinating watch as a ...	0.119859	0.559253
2	it was a good horror miniseries if not the bes...	0.338889	0.577778
3	i came very late across this miniseries and i ...	-0.086250	0.382917
4	ghoul is a different kind of horror that makes...	-0.096825	0.600317
...	...	...	...
4249	please someone explain anurag kashyap story	0.000000	0.000000
4250	fuack this	0.000000	0.000000
4251	justiceforssr boycottbollywood	0.000000	0.000000
4252	bakavass time vesting and illogical	0.000000	0.000000
4253	average	-0.150000	0.400000

4254 rows × 3 columns

In [32]:

```
data_clean_df.sample(5)
```

Out[32]:

	transcript	polarity	subjectivity
206	nice tv series with nice screen play little bi...	0.268750	0.816667
2821	i thought the acting was outstanding the writi...	0.362500	0.568750
926	outpost betaal some plots were changed to add ...	-0.006944	0.311111
1289	just watched episode to be honest after all ne...	0.262500	0.666667
3992	reviews says must be at least charactersmovie ...	-0.300000	0.400000

In [33]:

```
#classifying sentiments based on the reviews'score
def get_analysis(score):
  if score > 0:
    return "positive"
  elif score < 0:
      return "negative"
  else:
      return 'neutral'
data_clean_df["Analysis"] = data_clean_df.polarity.apply(get_analysis)
data_clean_df
```

Out[33]:

	transcript	polarity	subjectivity	Analysis
0	ghoul is another offering by netflix in its at...	-0.026488	0.569643	negative
1	ghoul or ghul is not a fascinating watch as a ...	0.119859	0.559253	positive
2	it was a good horror miniseries if not the bes...	0.338889	0.577778	positive
3	i came very late across this miniseries and i ...	-0.086250	0.382917	negative
4	ghoul is a different kind of horror that makes...	-0.096825	0.600317	negative
...	...	...	...	...
4249	please someone explain anurag kashyap story	0.000000	0.000000	neutral
4250	fuack this	0.000000	0.000000	neutral
4251	justiceforssr boycottbollywood	0.000000	0.000000	neutral
4252	bakavass time vesting and illogical	0.000000	0.000000	neutral
4253	average	-0.150000	0.400000	negative

4254 rows × 4 columns

In [34]:

```
data_clean_df.sample(10)
```

Out[34]:

	transcript	polarity	subjectivity	Analysis
3328	it is a great episode and everything has been ...	0.700000	0.825000	positive
2900	the entire cast is full of humour and entertai...	0.150000	0.525000	positive
1563	best web series till date horror thrilling adv...	0.683333	0.683333	positive
2376	superb web series scary music thrilled amazing	0.425000	0.900000	positive
1350	the story has come up with a new content which...	0.512121	0.551515	positive
3550	it shouldnt be called ghost stories the storie...	-0.492857	0.557143	negative
1333	i felt it was all over the place needed better...	0.375000	0.375000	positive
2491	forced everything is forced nothing naturalset...	-0.153030	0.265152	negative
486	i like this movie	0.000000	0.000000	neutral
974	definitely im not critic but this horror serie...	0.083333	0.677778	positive

In [35]:

```
j=0
k=0
for i in range(0,data_clean_df.shape[0]):
    if data_clean_df.Analysis[i]=='negative':

        j= j+1
    elif data_clean_df.Analysis[i]=='positive':
#The folloswing code can be undocumented , if you're interested in reading that sentiments'
        #         print (k,data_clean_df.transcript[i])
        k+=1
neu= data_clean_df.shape[0]- (j+k)
print ('So,The following is our "Sentiment Analysis" for the Top 10 Indian Web Series(Horro
print ('\nNo of Negative Reviews from our Total DataSet(around 5k) ->',j)
print ('No of Positive Reviews from our Total DataSet(around 5k) ->',k)
print ('No of  Neutral Reviews from our Total DataSet(around 5k) ->',neu)

neg_per= (j/data_clean_df.shape[0])*100
pos_per=(k/data_clean_df.shape[0])*100
neu_per=(neu/data_clean_df.shape[0])*100

print('\nPercentage of Negative Reviews -> '+ str(neg_per) + " %")
print('Percentage of Positive Reviews -> '+ str(pos_per) + ' %')
print('Percentage of Neutral  Reviews -> '+ str(neu_per) + "  %" )
```

```
So,The following is our "Sentiment Analysis" for the Top 10 Indian Web Serie
s(Horror Genre) :

No of Negative Reviews from our Total DataSet(around 5k) -> 1133
No of Positive Reviews from our Total DataSet(around 5k) -> 2837
No of  Neutral Reviews from our Total DataSet(around 5k) -> 284

Percentage of Negative Reviews -> 26.633756464503993 %
Percentage of Positive Reviews -> 66.69017395392571 %
Percentage of Neutral  Reviews -> 6.676069581570286  %
```

Sentiment Findings:

So,The following is our "Sentiment Analysis" for the Top 10 Indian Web Series(Horror Genre) :

```
No of Negative Reviews from our Total DataSet(around 5k) -> 1133
No of Positive Reviews from our Total DataSet(around 5k) -> 2837
No of  Neutral Reviews from our Total DataSet(around 5k) -> 284

Percentage of Negative Reviews -> 26.633756464503993 %
Percentage of Positive Reviews -> 66.69017395392571 %
Percentage of Neutral  Reviews -> 6.676069581570286  %
```

```
This also confirms our vague analysis that we did using just the wordcloud sentime
nts.
```

Data Visualizations

Data Visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

The advantages and benefits of good data visualization

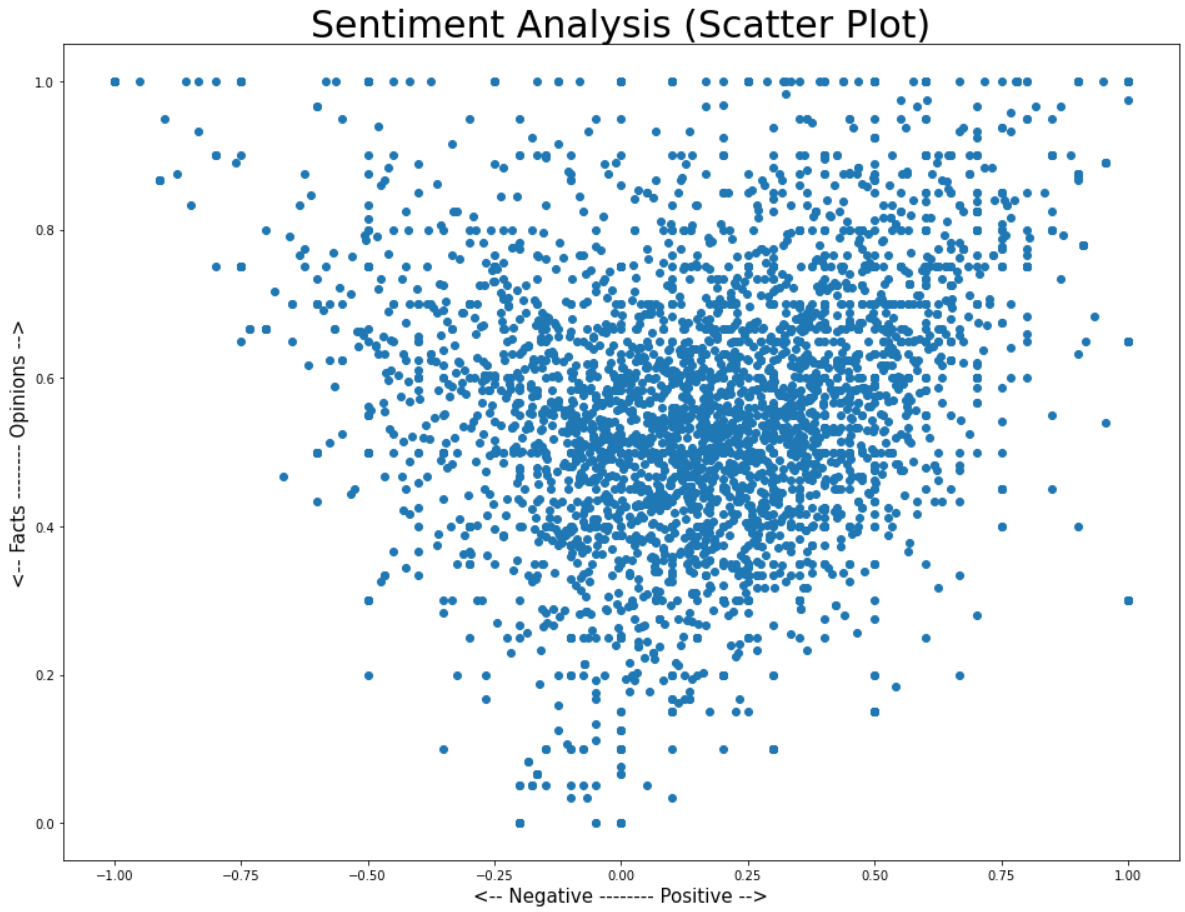
Our eyes are drawn to colors and patterns. We can quickly identify red from blue, square from circle. Our culture is visual, including everything from art and advertisements to TV and movies. Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message. When we see a chart, we quickly see trends and outliers. If we can see something, we internalize it quickly. It’s basically storytelling with a purpose.

Other benefits of data visualization include the following:

- **Confirms our results derived from numeric data analysis.**
- The ability to absorb information quickly, improve insights and make faster decisions;
- An increased understanding of the next steps that must be taken to improve the organization;
- An improved ability to maintain the audience's interest with information they can understand;
- An easy distribution of information that increases the opportunity to share insights with everyone involved;
- Eliminate the need for data scientists since data is more accessible and understandable; and
- An increased ability to act on findings quickly and, therefore, achieve success with greater speed and less mistakes.

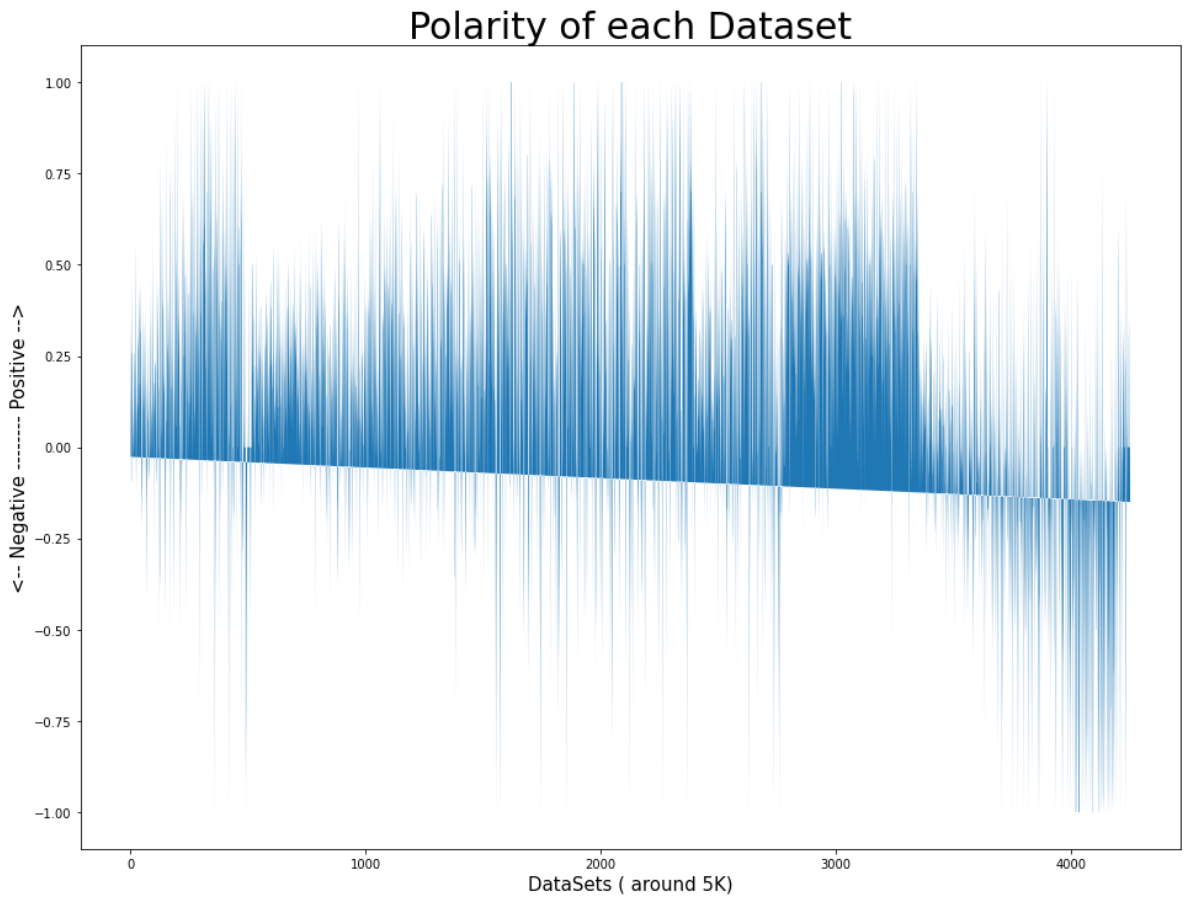
In [36]:

```
# Let's plot the results
import matplotlib.pyplot as plt

plt.rcParams['figure.figsize'] = [16, 12]

plt.scatter(data_clean_df['polarity'],data_clean_df['subjectivity'])

plt.title('Sentiment Analysis (Scatter Plot)', fontsize=30)
plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()
```



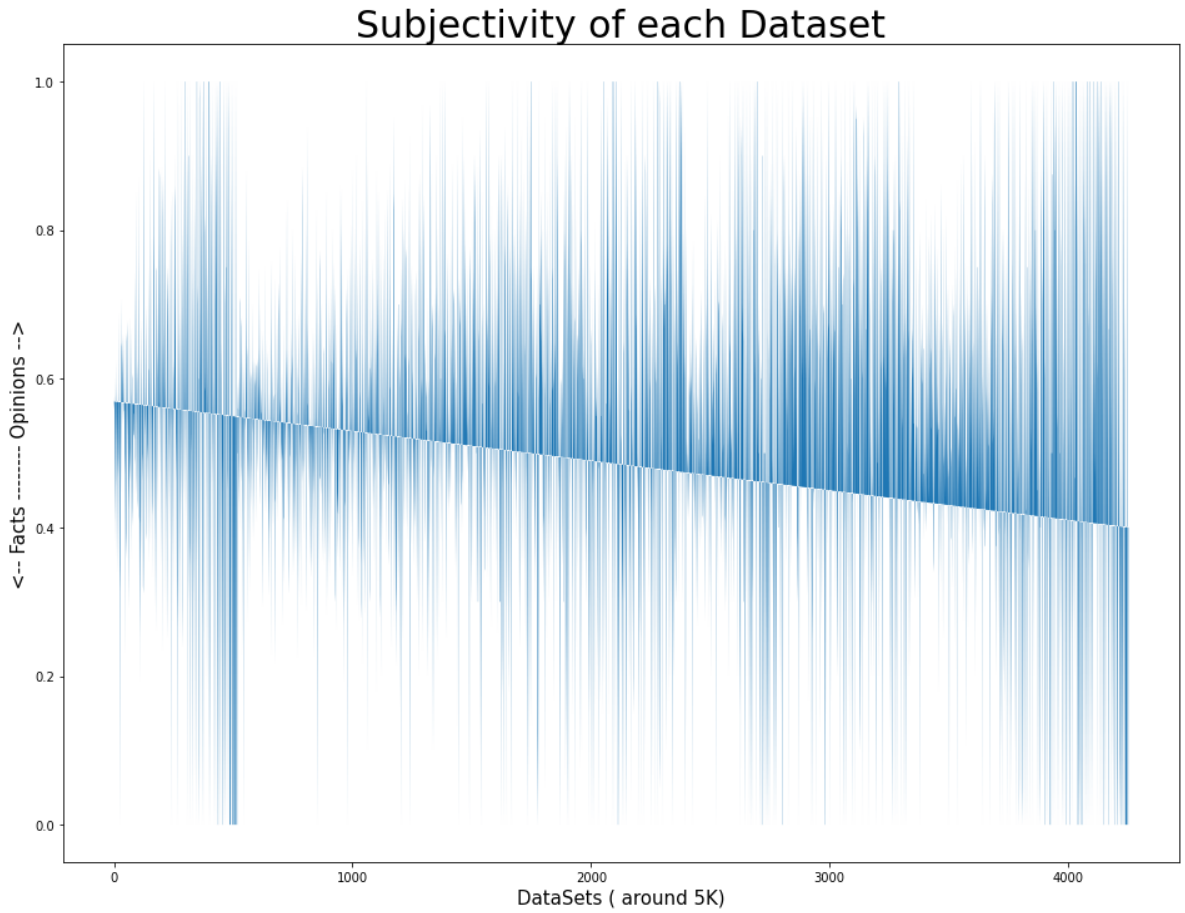
In [37]:

```
plt.fill(data_clean_df['polarity'])
plt.title('Polarity of each Dataset', fontsize=30)
plt.xlabel('DataSets ( around 5K)', fontsize=15)
plt.ylabel('<-- Negative -------- Positive -->', fontsize=15)

plt.show()
```



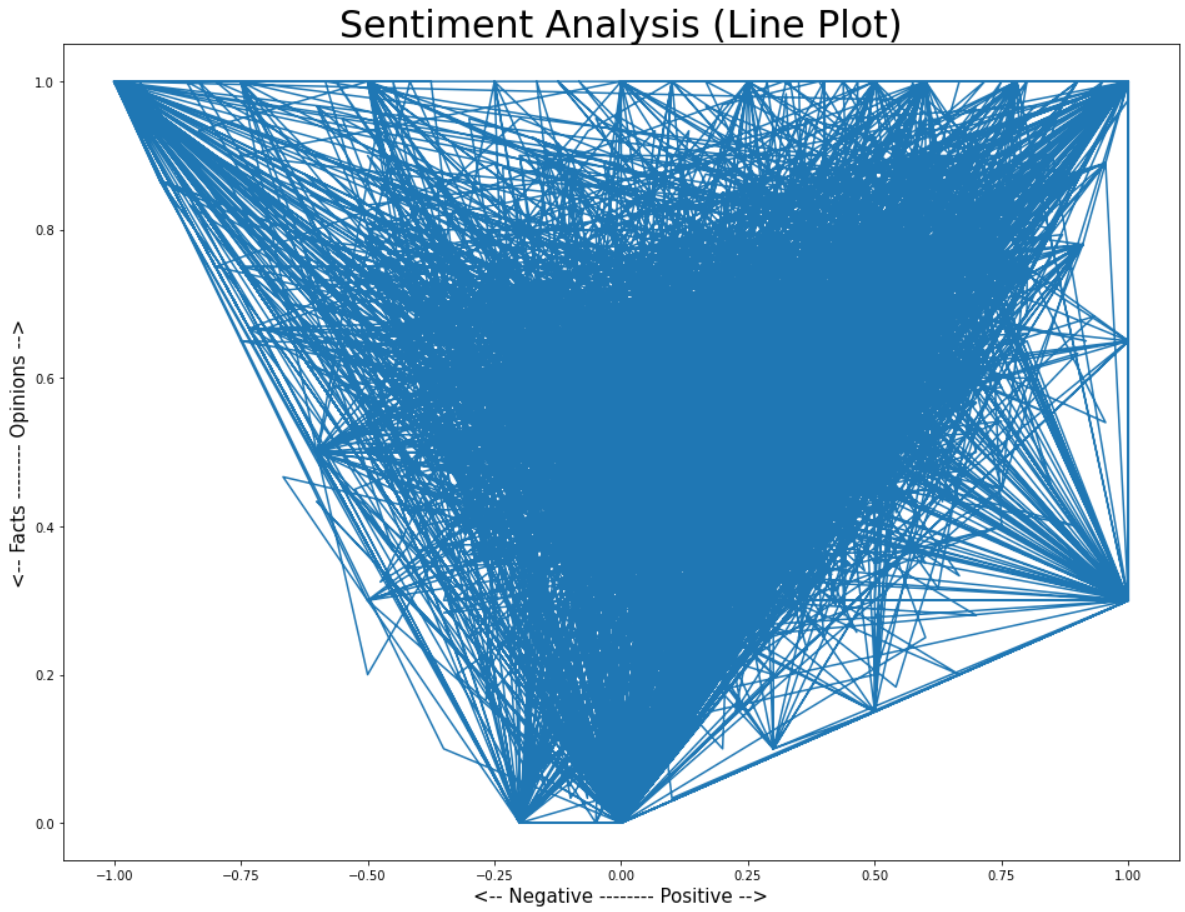
In [38]:

```
plt.fill(data_clean_df['subjectivity'])
plt.title('Subjectivity of each Dataset', fontsize=30)
plt.xlabel('DataSets ( around 5K)', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()
```



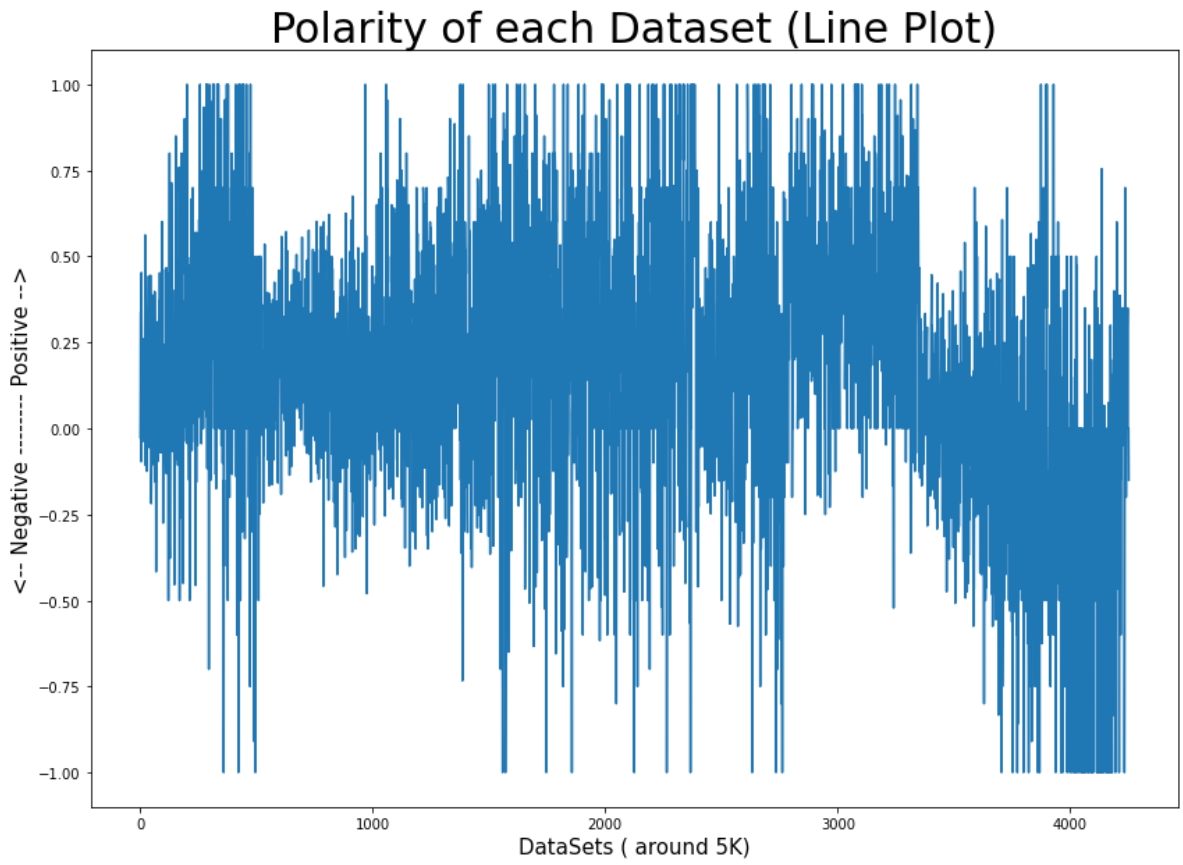
In [39]:

```
plt.plot(data_clean_df['polarity'],data_clean_df['subjectivity'])
plt.rcParams['figure.figsize'] = [14, 10]
plt.title('Sentiment Analysis (Line Plot)', fontsize=30)
plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()
```



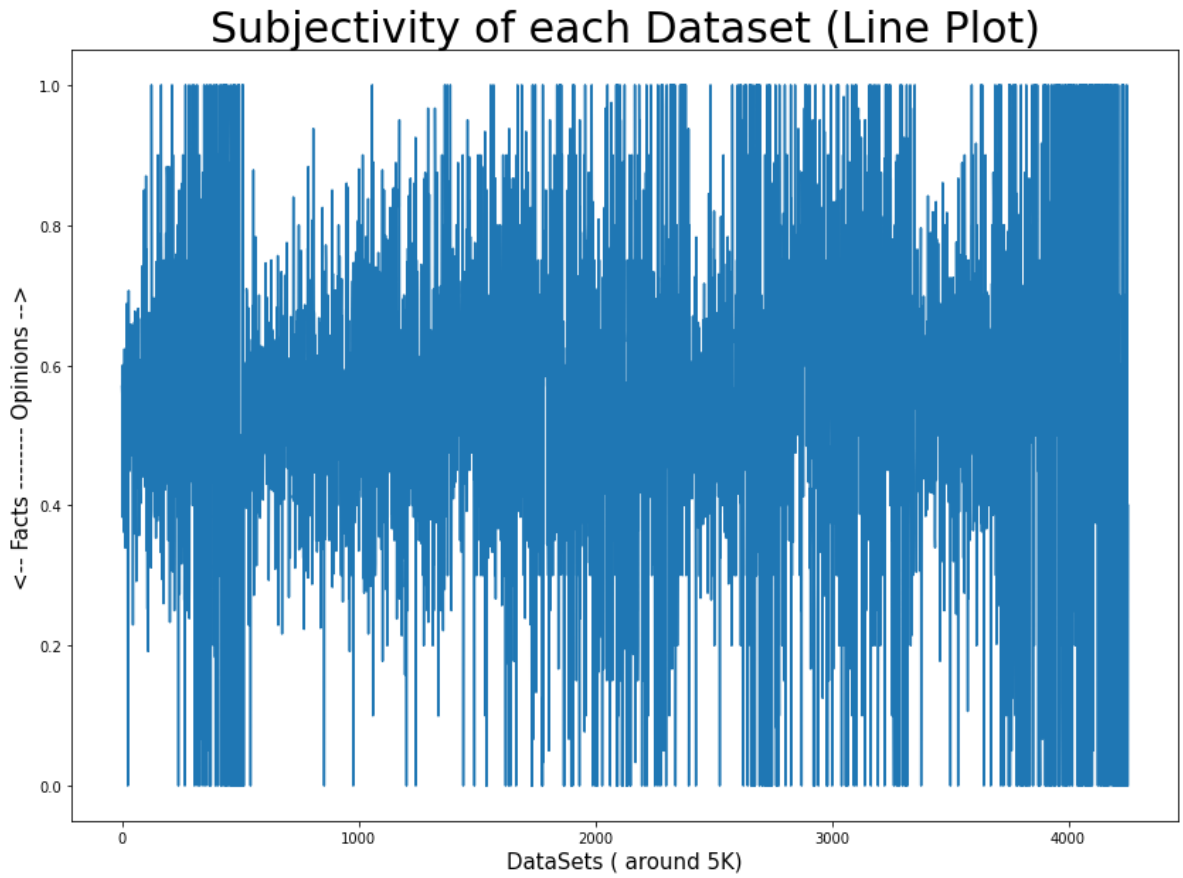
In [40]:

```
plt.plot(data_clean_df['polarity'])
plt.title('Polarity of each Dataset (Line Plot)', fontsize=30)
plt.xlabel('DataSets ( around 5K)', fontsize=15)
plt.ylabel('<-- Negative -------- Positive -->', fontsize=15)

plt.show()
```



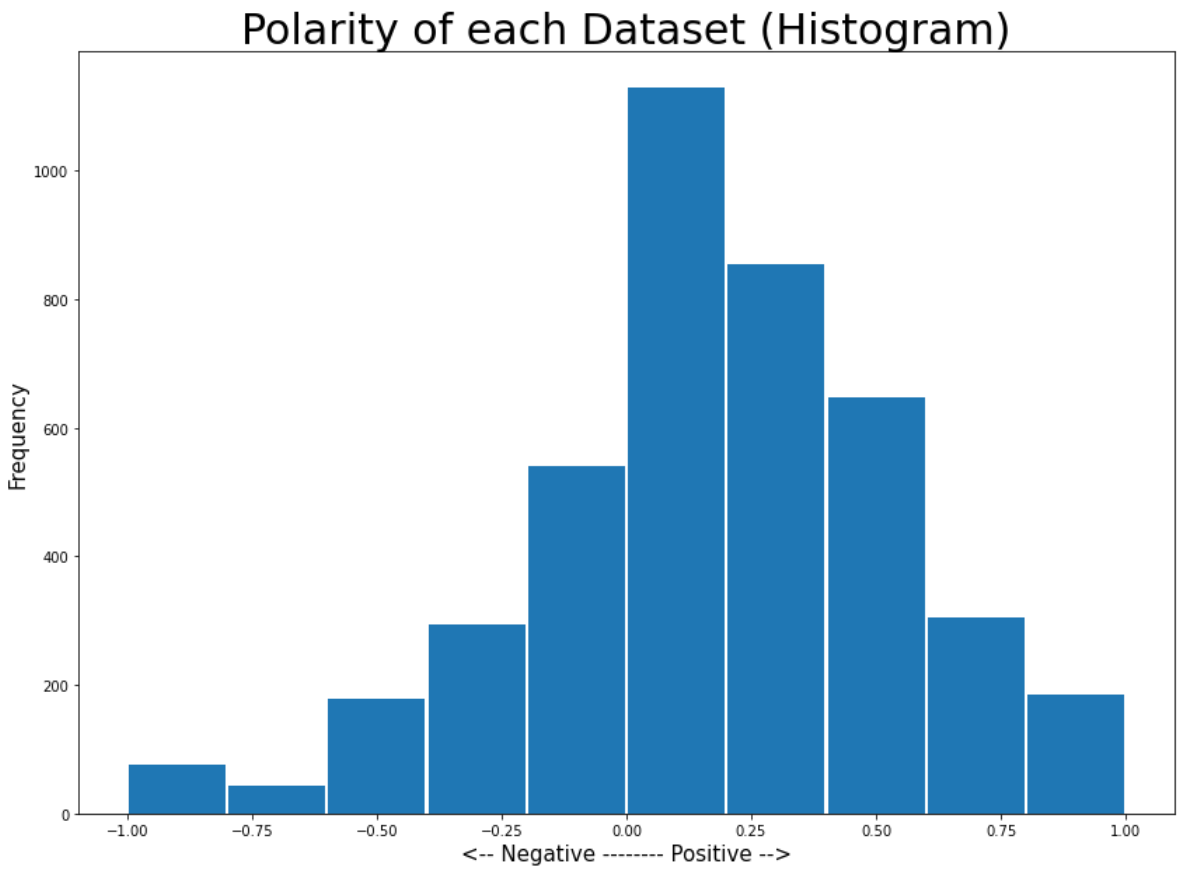
In [41]:

```
plt.plot(data_clean_df['subjectivity'])
plt.title('Subjectivity of each Dataset (Line Plot)', fontsize=30)
plt.xlabel('DataSets ( around 5K)', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)


plt.show()
```



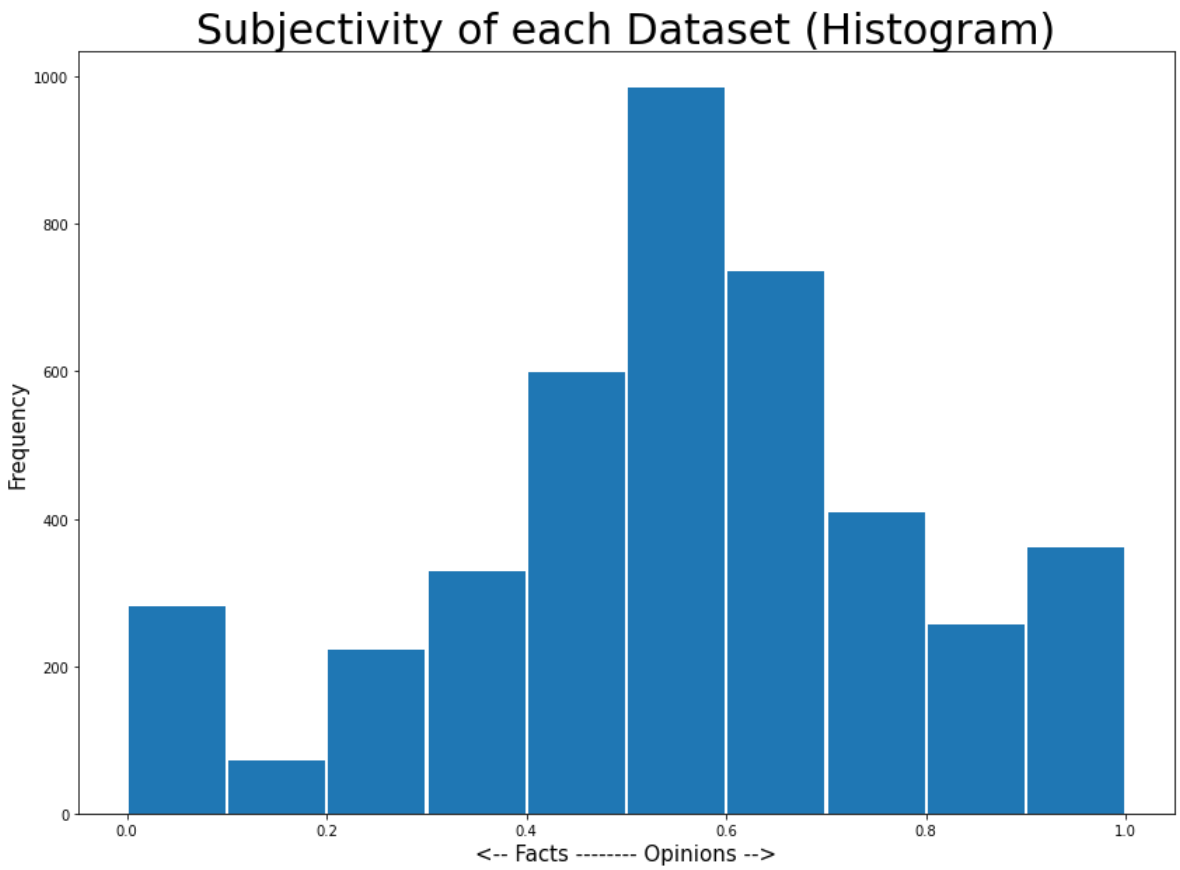
In [42]:

```
plt.hist(data_clean_df['polarity'], rwidth=.969)
plt.title('Polarity of each Dataset (Histogram)', fontsize=30)
plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)
plt.ylabel('Frequency', fontsize=15)

plt.show()
```



In [43]:

```
plt.hist(data_clean_df['subjectivity'], rwidth=.969)
plt.title('Subjectivity of each Dataset (Histogram)', fontsize=30)
plt.xlabel('<-- Facts -------- Opinions -->', fontsize=15)
plt.ylabel('Frequency', fontsize=15)

plt.show()
```



In [44]:

```
data_clean_df
```

Out[44]:

	transcript	polarity	subjectivity	Analysis
0	ghoul is another offering by netflix in its at...	-0.026488	0.569643	negative
1	ghoul or ghul is not a fascinating watch as a ...	0.119859	0.559253	positive
2	it was a good horror miniseries if not the bes...	0.338889	0.577778	positive
3	i came very late across this miniseries and i ...	-0.086250	0.382917	negative
4	ghoul is a different kind of horror that makes...	-0.096825	0.600317	negative
...	...	...	...	...
4249	please someone explain anurag kashyap story	0.000000	0.000000	neutral
4250	fuack this	0.000000	0.000000	neutral
4251	justiceforssr boycottbollywood	0.000000	0.000000	neutral
4252	bakavass time vesting and illogical	0.000000	0.000000	neutral
4253	average	-0.150000	0.400000	negative

4254 rows × 4 columns

In [45]:

```
#Creating a new DataFrame with only Positve Reviews.
#We will later use this df to create a wordcloud having only positive sentiments.
positive_df=data_clean_df[data_clean_df['Analysis']=='positive']
```

In [46]:

```
positive_df
```

Out[46]:

	transcript	polarity	subjectivity	Analysis
1	ghoul or ghul is not a fascinating watch as a ...	0.119859	0.559253	positive
2	it was a good horror miniseries if not the bes...	0.338889	0.577778	positive
5	lets discuss it in pointslets begin with the b...	0.453333	0.491667	positive
6	great script greater acting of course keeping ...	0.254615	0.406923	positive
7	ghoul is nothing short of a masterpiece of the...	0.261310	0.559921	positive
...	...	...	...	...
4237	anurag kashyaps was quite good avoid rest	0.700000	0.600000	positive
4238	not scary overratedtoo long	0.100000	0.700000	positive
4239	what a joke wasted my new year day main bahut ...	0.034343	0.262626	positive
4247	not scary at all although very funny	0.287500	1.000000	positive
4248	its not bad its bakwasssss	0.350000	0.666667	positive

2837 rows × 4 columns

In [51]:

```
# Python program to generate WordCloud for POSITVE SENTIMENTS

# importing all necessary modules
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd


comment_words_pos = ''

# Add new stop words

selected_stop_words=['show','season','one','season','watch','story','web','netflix','movie'
stopwords = list(additional_stop_words) + selected_stop_words + add_stop_words + list(STOPW

# iterate through the file
for val in positive_df.transcript:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()
    comment_words_pos += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words_pos)

# plot the WordCloud image
plt.figure(figsize = (10,10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.title('WordCloud for POSITVE SENTIMENTS\n',fontsize=30)

plt.show()
```

WordCloud for POSITVE SENTIMENTS

In [52]:

```
#Creating a new DataFrame with only Negatve Reviews.
#We will later use this df to create a wordcloud having only negative sentiments.
negative_df=data_clean_df[data_clean_df['Analysis']=='negative']
```

In [53]:

```
negative_df
```

Out[53]:

	transcript	polarity	subjectivity	Analysis
0	ghoul is another offering by netflix in its at...	-0.026488	0.569643	negative
3	i came very late across this miniseries and i ...	-0.086250	0.382917	negative
4	ghoul is a different kind of horror that makes...	-0.096825	0.600317	negative
10	off late it has become really fashionable in i...	-0.021429	0.514286	negative
13	who says that there are no jumpscares probably...	-0.036905	0.490476	negative
...	...	...	...	...
4226	unexplained plotsmade no sense	-0.050000	0.000000	negative
4234	worst everworst everworst ever	-1.000000	1.000000	negative
4235	boring over hyped confusing	-0.650000	0.700000	negative
4241	nonsense waste of time	-0.200000	0.000000	negative
4253	average	-0.150000	0.400000	negative

1133 rows × 4 columns

In [54]:

```
# Python program to generate WordCloud for NEGATVE SENTIMENTS

# importing all necessary modules
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd


comment_words_neg = ''

# Add new stop words

selected_stop_words=['show','season','one','good','season','watch','story','bajpayee','movi
stopwords = list(additional_stop_words) + selected_stop_words + add_stop_words + list(STOPW

# iterate through the file
for val in negative_df.transcript:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()
    comment_words_neg += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words_neg)

# plot the WordCloud image
plt.figure(figsize = (10,10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.title('WordCloud for NEGATVE SENTIMENTS\n',fontsize=30)

plt.show()
```

In [55]:

```
#Creating a new DataFrame with only Neutral Reviews.
#We will later use this df to create a wordcloud having only neutral sentiments.
neutral_df=data_clean_df[data_clean_df['Analysis']=='neutral']
```

In [60]:

```
# Python program to generate WordCloud for NEUTRAL SENTIMENTS

# importing all necessary modules
from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
import pandas as pd


comment_words_neu = ''

# Add new stop words

selected_stop_words=['show','season','one','good','thriller','season','shame','movie','watc
stopwords = list(additional_stop_words) + selected_stop_words + add_stop_words + list(STOPW

# iterate through the file
for val in neutral_df.transcript:

    # typecaste each val to string
    val = str(val)

    # split the value
    tokens = val.split()
    comment_words_neu += " ".join(tokens)+" "

wordcloud = WordCloud(width = 800, height = 800,
                background_color ='white',
                stopwords = stopwords,
                min_font_size = 10).generate(comment_words_neu)

# plot the WordCloud image
plt.figure(figsize = (10,10), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.title('WordCloud for NEUTRAL SENTIMENTS\n',fontsize=30)

plt.show()
```

Additonal Information

The most frequent words from POSITIVE , NEGATIVE and NEUTRAL REVIEWS' data set.

In [57]:

```
# Python program to find the most frequent words from POSITIVE REVIEWS' data set
from collections import Counter


# split() returns list of all the words in the string
split_it = comment_words_pos.split(" ")


# Pass the split_it list to instance of Counter class.
Counter = Counter(split_it)

# most_common() produces k frequently encountered
# input values and their respective counts.
most_occur = Counter.most_common()

print(most_occur)
```

```
[('the', 3593), ('and', 2353), ('a', 1881), ('is', 1782), ('of', 1718),
('it', 1569), ('to', 1495), ('i', 1359), ('series', 1291), ('this', 1142),
('in', 986), ('good', 926), ('was', 827), ('story', 782), ('for', 768),
('but', 748), ('horror', 730), ('its', 657), ('watch', 656), ('are', 596),
('you', 589), ('not', 541), ('with', 538), ('that', 534), ('show', 504),
('have', 475), ('very', 473), ('one', 436), ('like', 417), ('all', 414),
('indian', 390), ('really', 366), ('so', 362), ('on', 361), ('as', 353),
('be', 351), ('great', 350), ('best', 337), ('by', 322), ('more', 322),
('time', 314), ('just', 313), ('movie', 303), ('will', 302), ('they', 29
6), ('from', 291), ('acting', 280), ('has', 278), ('season', 268), ('amazi
ng', 261), ('some', 258), ('an', 257), ('web', 250), ('dont', 243), ('zomb
ie', 235), ('better', 234), ('episode', 233), ('first', 230), ('at', 229),
('which', 225), ('if', 220), ('watching', 218), ('must', 205), ('much', 19
9), ('betaal', 199), ('there', 198), ('well', 198), ('netflix', 195), ('ca
n', 190), ('awesome', 190), ('loved', 189), ('india', 185), ('or', 178),
('were', 178), ('after', 177), ('what', 173), ('episodes', 173), ('storie
s', 164), ('would', 161), ('love', 160), ('only', 158), ('than', 155), ('z
ombies', 155), ('new', 154), ('out', 154), ('concept', 152), ('been', 15
2), ('up', 150), ('thriller', 149), ('my', 145), ('ghoul', 144), ('we', 14
```

In [58]:

```
# Python program to find the most frequent words from NEGATIVE REVIEWS' data set
from collections import Counter


# split() returns list of all the words in the string
split_it = comment_words_neg.split(" ")


# Pass the split_it list to instance of Counter class.
Counter = Counter(split_it)

# most_common() produces k frequently encountered
# input values and their respective counts.
most_occur = Counter.most_common()

print(most_occur)
```

```
[('the', 1301), ('of', 869), ('and', 852), ('to', 706), ('a', 683), ('is',
593), ('it', 523), ('i', 502), ('this', 453), ('not', 389), ('was', 373),
('in', 362), ('time', 303), ('series', 287), ('story', 286), ('but', 276),
('waste', 253), ('for', 247), ('worst', 222), ('are', 221), ('stories', 21
3), ('that', 210), ('horror', 206), ('its', 205), ('with', 204), ('you', 1
98), ('like', 192), ('have', 185), ('movie', 184), ('watch', 182), ('all',
179), ('as', 174), ('very', 172), ('no', 167), ('so', 161), ('bad', 159),
('dont', 154), ('just', 154), ('one', 150), ('they', 140), ('at', 138),
('from', 134), ('be', 132), ('by', 131), ('on', 130), ('show', 123), ('sca
ry', 122), ('netflix', 119), ('were', 110), ('what', 107), ('even', 107),
('watching', 105), ('there', 102), ('acting', 102), ('indian', 100), ('goo
d', 100), ('only', 98), ('if', 97), ('boring', 97), ('some', 96), ('your',
95), ('my', 94), ('or', 93), ('make', 92), ('ghost', 91), ('poor', 90),
('ever', 87), ('really', 85), ('such', 84), ('any', 76), ('has', 75), ('pa
thetic', 73), ('which', 72), ('first', 71), ('after', 70), ('an', 69), ('t
han', 69), ('episode', 68), ('will', 68), ('people', 67), ('plot', 67),
('nothing', 67), ('can', 66), ('out', 64), ('disappointed', 64), ('zombi
e', 63), ('up', 62), ('would', 61), ('do', 60), ('made', 58), ('movies', 5
7), ('much', 57), ('star', 57), ('sense', 56), ('me', 55), ('give', 54),
```

In [59]:

```
# Python program to find the most frequent words from NEGUTRAL REVIEWS' data set
from collections import Counter


# split() returns list of all the words in the string
split_it = comment_words_neu.split(" ")


# Pass the split_it list to instance of Counter class.
Counter = Counter(split_it)

# most_common() produces k frequently encountered
# input values and their respective counts.
most_occur = Counter.most_common()

print(most_occur)
```

```
[('the', 133), ('to', 100), ('a', 88), ('and', 87), ('of', 80), ('is', 6
6), ('this', 64), ('i', 59), ('series', 56), ('watch', 51), ('it', 50),
('for', 46), ('in', 43), ('show', 33), ('horror', 32), ('all', 30), ('movi
e', 29), ('you', 27), ('story', 27), ('dont', 26), ('with', 24), ('just',
23), ('on', 23), ('from', 22), ('like', 22), ('stories', 22), ('not', 21),
('have', 21), ('but', 20), ('what', 20), ('please', 20), ('make', 18), ('i
ts', 18), ('one', 18), ('be', 18), ('are', 18), ('must', 18), ('they', 1
7), ('indian', 17), ('that', 16), ('my', 16), ('time', 16), ('well', 14),
('was', 14), ('or', 14), ('no', 14), ('season', 14), ('watching', 14), ('s
ome', 13), ('web', 13), ('star', 13), ('as', 12), ('see', 12), ('can', 1
2), ('ghost', 12), ('any', 11), ('every', 11), ('there', 11), ('me', 11),
('so', 11), ('even', 10), ('netflix', 10), ('ever', 10), ('ghoul', 10),
('at', 9), ('by', 9), ('will', 9), ('made', 9), ('should', 9), ('watched',
8), ('next', 8), ('betaal', 8), ('has', 8), ('your', 8), ('only', 8), ('su
ch', 7), ('know', 7), ('india', 7), ('want', 7), ('zombie', 7), ('conten
t', 7), ('wanted', 7), ('waiting', 7), ('their', 7), ('everything', 7),
('if', 7), ('an', 7), ('who', 7), ('episode', 7), ('people', 7), ('class',
7), ('sense', 7), ('apte', 6), ('say', 6), ('why', 6), ('about', 6), ('arm
y', 6), ('till', 6), ('nothing', 6), ('masterpiece', 6), ('genre', 6), ('d
```

THANK YOU

- By Harsh Kumar (Delhi Technological University,DTU (formerly Delhi College of Engineering,DCE))

- Intern under Prof. Sasadhar Bera, Ph.D. (Indian Institute of Management ,Ranchi)