Opinion Mining + Sentiment Classification :

For the Top 10 Indian Web Series(Romance 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 Romance 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 ROMANCE genre.
r1_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r2_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r3_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r4_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r5_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r6_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r7_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r8_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r9_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REVI
r10_df=pd.read_excel(r"C:\Users\Asus\Desktop\Intern Work-IIM Ranchi\ALL REVIEWS\ROMANCE REV
```

Loading [MathJax]/extensions/Safe.js

In [3]:

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

Out[3]:

```
                                          Unnamed: 0
  0    Even though I am no expert but I found the cam...
  1        I stumbled upon this series randomly while sur...
  2    I was prejudice towards watching The Married W...
  3           I am sure we will wait a long while till somet...
  4    10/10 without thinking and would recommend to ...
 ...                                                  ...
409     Drama thrill, suspense. Monica and Ridhi bang ...
410             The storyline of this series is different but ...
411          The series is so beautiful it was so pure just...
412            This time love is going to stay, irrespective ...
413    #Married Woman is the best television show. Ev...

414 rows × 1 columns
```

In [4]:

```
r2_df
```

Out[4]:

```
                                          Unnamed: 0
  0     Having watched the 1st two seasons of BBB, I w...
  1        It was one of my first OTT Romance Series. I w...
  2         With season 3 streaming in ALT Balaji, I ventu...
  3    Beautiful Webseries Which Shows Real Face of L...
  4           I don't have any words to describe the series ...
 ...                                                  ...
498                              Broken but feel beautiful bro...
499                                  End of lovely 2019......💘💘
500            It's stolen my heart ... sometime it's really ...
501    💘💘💘💘,\nit is the besttt series i ever seen.......
502                        It's awsmm specially vikrant acting ❤

503 rows × 1 columns
```

Loading [MathJax]/extensions/Safe.js

In [5]:

```
r3_df
```

Out[5]:

```
                                               Unnamed: 0
   0  CLASSICAL MUSIC MEETS FUSION\n\nFrom a Directo...
   1          Bandish Bandits is a masterclass in the music ...
   2         A web series that makes me feel proud of our v...
   3          Bandish Bandits, what a pleasant surprise! In ...
   4          Its a musical saga based on two opposite indiv...
 ...                                                      ...
2093            I never watched a series like this before..Its...
2094          Refreshing web series with good portray of ind...
2095            brilliant fascinating and heart touching \nsal...
2096                       Awesome web series. Must watch.
2097        Fabulous compositions and brilliant idea of re...

2098 rows × 1 columns
```

In [6]:

```
r4_df
```

Out[6]:

```
                                          Unnamed: 0
  0           10/10 It's s really Beautiful love story with ...
  1      10/10 \n\nWOW, Superb, Amazing\n\nThis is the ...
  2       100% perfect \n\nExellent, Superb, Amazing\n\n...
  3        What a show! It is very good show. As I have w...
  4          My Rating--5/5\n\nI am in love with this serie...
...                                                      ...
408                      Baarish ...♥ romantic ....classical...
409            Its an amazing web series but the ending .......
410    Season one is awesome and season two is waste ...
411                        Plz 2 season 12 to 20 part release
412                     It is too good ♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥...

413 rows × 1 columns
```

Loading [MathJax]/extensions/Safe.js

In [7]:

```
r5_df
```

Out[7]:

```
                                           Unnamed: 0
  0    So much emotions. Must watch. Everyone was awe...
  1       Such a mature content. We cannot hate any char...
  2             This is the proof that if Ekta Kapoor wants sh...
  3           The show is too good. It touches the hearts bu...
  4       This Web Series Did Excellent Performance. Spe...
...                                                  ...
346    A great example of an interesting FAMILY DRAMA...
347      The concept is not new..We've seen many such d...
348           All the episodes are really good...A jaw-dropp...
349        Mona Singh has once again left us awe-struck w...
350              I loved this show very much...It is a great fa...

351 rows × 1 columns
```

In [8]:

```
r6_df
```

Out[8]:

```
                                         Unnamed: 0
  0        Bebaakee Is Outstanding I can’t express it in ...
  1    Bebakee is an amazing show.I am seriously addi...
  2            I don't ever write a review but this one's an ...
  3     Bebakee is an outstanding series. ♡ to watche...
  4            I must say all the characters of this series h...
...                                                ...
789             Amazing series......... waiting for this kind ...
790        This series is really really awesome 😘 I love ...
791            Kushal , u tried best 2 play role of Sujal bt ...
792  Awsm one one of the best web series\nRomantic ...
793         People's say that ALT balaji didn't create goo...

794 rows × 1 columns
```

Loading [MathJax]/extensions/Safe.js

In [9]:

```
r7_df
```

Out[9]:

```
                                        Unnamed: 0
  0   Ram Kapoor and Sakshi Tanwar create magic on s...
  1           While I wasn't an ardent follower of Indian te...
  2            Great chemistry! Love the simplicity of their ...
  3         I awaited the season 3 for long as to see Ram ...
  4              Grear family story. A little bit complecated. ...
...                                                    ...
302            Wow the trailer is too good ...It will become ...
303       Gosh.. Cinematography and the plot is just awe...
304           Alt Balaji always have something interesting t...
305              Interesting show with thrilling twists and tur...
306        Enjoyed this season too much, now just waiting...

307 rows × 1 columns
```

In [10]:

```
r8_df
```

Out[10]:

```
                                        Unnamed: 0
  0   Season 1 was fairly decent, seemingly well wri...
  1         I'm not into giving detailed reviews but this ...
  2   Just now I completed both seasons of out of lo...
  3              First of all its not a thriller....let us rath...
  4   Rasika duggal acting was super... Amazing i lo...
...                                                    ...
367    A very well written, directed and performed sc...
368     Both seasons a 10+. Hats off to directors, act...
369      Only location is good;, acting was avg; story,...
370      "Out of Love" is the remade version of the cri...
371       You will really enjoy this movie but what I di...

372 rows × 1 columns
```

Loading [MathJax]/extensions/Safe.js

In [11]:
```
r9_df
```

Out[11]:
```
                                                  Unnamed: 0
  0          Directed by Mira Nair, 'A Suitable Boy' consis...
  1   BUNNY REVIEWED- A SUITABLE BOY ✨\n\nI've had t...
  2         A Suitable Boy is Surprisingly a very good com...
  3          Dazed after completing the watching of The Sui...
  4          After binge watching the thrilling Mirzapur S2...
...                                                        ...
416                           Review in one word\nPathetic...
417              British agenda for glorify them by BBC product...
418               Cinematography is nice but boring storyline......
419                     JUSTICE FOR SUSHANT SING RAJPUT
420     banned 🚫 #Boycott webseries\n# Anti Hindu seri...

421 rows × 1 columns
```

In [12]:
```
r10_df
```

Out[12]:
```
                                           Unnamed: 0
  0     I loved the series and highly recommend it to ...
  1   This show has my heart! One of the best rom-co...
  2   Mismatched is a Fun, Cute, Sweet and Entertain...
  3      It was sooo amazing series...why have this got...
  4     Honestly, I'm definitely happy with the show a...
...                                                 ...
367            The only good thing about series is Rohit
368                 It was wasted of time 😵‍💫😵‍💫🤮🤮
369                       Hated it. total waste of time.
370                  Goddd..such a waste of time 🤷
371             worst show ever!!!!!really disappointed.

372 rows × 1 columns
```

Loading [MathJax]/extensions/Safe.js

In [13]:

```
#combining all the review dataframes into one dataframe
combined_df = pd.concat([r1_df, r2_df,r3_df,r4_df,r5_df,r6_df,r7_df,r8_df,r9_df,r10_df], ig
```

In [14]:

```
combined_df
```

Out[14]:

	Unnamed: 0
0	Even though I am no expert but I found the cam...
1	I stumbled upon this series randomly while sur...
2	I was prejudice towards watching The Married W...
3	I am sure we will wait a long while till somet...
4	10/10 without thinking and would recommend to ...
...	...
6040	The only good thing about series is Rohit
6041	It was wasted of time 😵‍💫😵‍💫🤮🤮
6042	Hated it. total waste of time.
6043	Goddd..such a waste of time 🤷‍♀️
6044	worst show ever!!!!!really disappointed.

6045 rows × 1 columns

In [15]:

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

Loading [MathJax]/extensions/Safe.js

In [16]:

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

Out[16]:

	transcript
0	Even though I am no expert but I found the cam...
1	I stumbled upon this series randomly while sur...
2	I was prejudice towards watching The Married W...
3	I am sure we will wait a long while till somet...
4	10/10 without thinking and would recommend to ...
...	...
6040	The only good thing about series is Rohit
6041	It was wasted of time 😵‍💫😵‍💫🤮🤮
6042	Hated it. total waste of time.
6043	Goddd..such a waste of time 🤷‍♀️
6044	worst show ever!!!!!really disappointed.

6045 rows × 1 columns

In [23]:

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

Out[23]:

	transcript
1490	Oh my god it is just awesome and please come b...
5690	This story takes of great showing how there ar...
4093	I love imtize ❤️❤️❤️❤️❤️😭😭😭
1354	Nothing less than Brilliant. Their music bring...
3209	It's very much lovely web series I never seen ...
5022	Absolutely brilliant. You will see without ge...
4725	Terrific story-line and all the artists who ar...
2649	loved everything about musical series Bandish ...
494	This is just so so beautiful... ❤️ I haven't g...
2794	Superb Series ever watched.. I really love it.

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 [24]:

```
# 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
```

Loading [MathJax]/extensions/Safe.js

In [25]:

```
# 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[25]:

	transcript
0	even though i am no expert but i found the cam...
1	i stumbled upon this series randomly while sur...
2	i was prejudice towards watching the married w...
3	i am sure we will wait a long while till somet...
4	without thinking and would recommend to every...
...	...
6040	the only good thing about series is rohit
6041	it was wasted of time 😵😵🤮🤮
6042	hated it total waste of time
6043	godddsuch a waste of time 🤷‍♀️
6044	worst show everreally disappointed

6045 rows × 1 columns

In [26]:

```
# 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
```

Loading [MathJax]/extensions/Safe.js

In [27]:

```
# 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[27]:

	transcript
0	even though i am no expert but i found the cam...
1	i stumbled upon this series randomly while sur...
2	i was prejudice towards watching the married w...
3	i am sure we will wait a long while till somet...
4	without thinking and would recommend to every...
...	...
6040	the only good thing about series is rohit
6041	it was wasted of time 😵😵🤮🤮
6042	hated it total waste of time
6043	godddsuch a waste of time 🤷‍♀️
6044	worst show everreally disappointed

6045 rows × 1 columns

In [28]:

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

Out[28]:

```
transcript    very nice 👌 series and we can understand need ...
Name: 2694, dtype: object
```

Loading [MathJax]/extensions/Safe.js

In [29]:

```
# 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 [30]:

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

Loading [MathJax]/extensions/Safe.js

In [31]:

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

Out[31]:

	transcript
0	even though i am no expert but i found the cam...
1	i stumbled upon this series randomly while sur...
2	i was prejudice towards watching the married w...
3	i am sure we will wait a long while till somet...
4	without thinking and would recommend to every ...
...	...
6040	the only good thing about series is rohit
6041	it was wasted of time
6042	hated it total waste of time
6043	godddsuch a waste of time
6044	worst show everreally disappointed

6045 rows × 1 columns

In [32]:

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

Out[32]:

	transcript
4848	just give this show a chance you wont be disap...
4008	omg season of bebakee is incredible
5465	stilted actingdialogues and dialogue delivery ...
882	its just amazing
3419	super
1326	while i am awed by the performances of veteran...

In [33]:

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

Out[33]:

```
transcript    very nice series and we can understand need to...
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

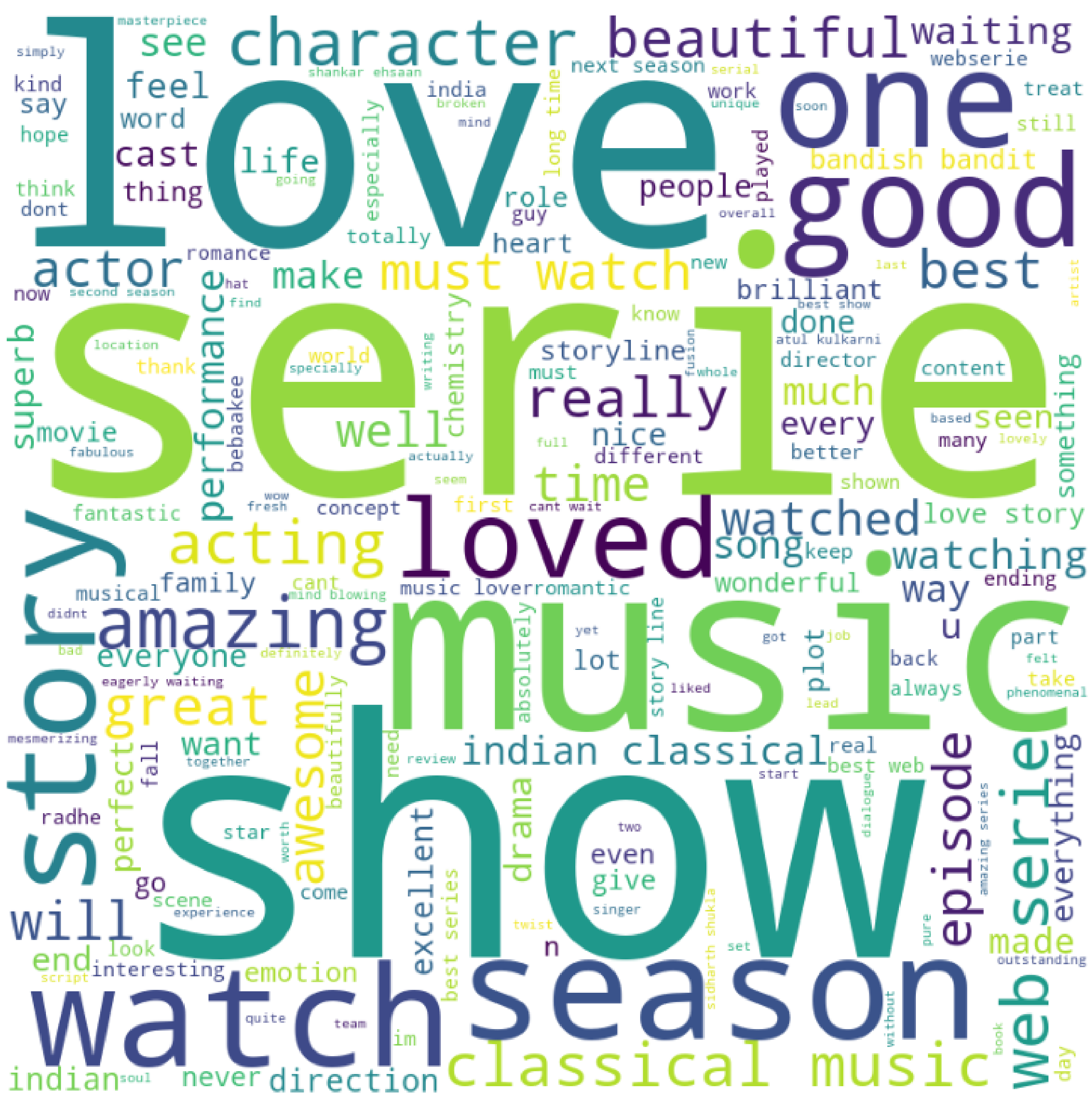
The definition of a corpus is a collection of texts, and they are all put together.

Loading [MathJax]/extensions/Safe.js

In [34]:

```
# 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()
```

Loading [MathJax]/extensions/Safe.js



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.

Loading [MathJax]/extensions/Safe.js

In [35]:

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

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

In [36]:

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

Out[36]:

```
'even though i am no expert but i found the camera work to be quite effect
ive all in all a great effort and kudos to the entire team i stumbled upon
this series randomly while surfing youtube the first thing i noticed was t
he smart and funny dialogue writing i was prejudice towards watching the m
arried woman because i never liked fantasy but i was hooked immediately i
am sure we will wait a long while till something shows up and competes i l
ove this series its magnificent i think it deserves without thinking and w
ould recommend to every thrill seeker that has yet to enjoy it for some re
ason i recommend this series script writing casting acting storyline music
score twists and never ends as you expected right to the end deserves this
show is one of the best shows i ever seen the acting was amazing the sound
track was fantastic its really good show love this show with all character
s this show means alot to melets say if you watch this you will never find
better i wouldnt say it is a must watch series but its certainly not one t
hat you must miss i watched it last night in one sitting it is interesting
and catchy the background score is great it was a total shocker and grippi
ng believable surprise i got hooked because of the underlying forces at wo
rk that shape the personalities characters are very well written and are p
```

Loading [MathJax]/extensions/Safe.js

In [37]:

```
# 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', 7602), ('and', 5242), ('of', 3853), ('a', 3822), ('is', 3372),
('to', 3218), ('i', 3200), ('it', 2756), ('this', 2703), ('series', 2643),
('in', 2148), ('for', 1808), ('music', 1760), ('love', 1487), ('show', 148
3), ('with', 1338), ('watch', 1256), ('you', 1175), ('its', 1129), ('was',
1121), ('story', 1107), ('all', 1062), ('season', 1028), ('but', 1023),
('just', 994), ('so', 976), ('that', 958), ('have', 958), ('are', 928),
('very', 853), ('classical', 826), ('good', 822), ('best', 815), ('one', 8
10), ('amazing', 800), ('by', 738), ('loved', 727), ('acting', 695), ('we
b', 680), ('on', 668), ('great', 659), ('not', 652), ('as', 639), ('has',
623), ('like', 622), ('indian', 604), ('really', 569), ('will', 551), ('mu
st', 543), ('be', 517), ('watching', 493), ('my', 486), ('beautiful', 47
7), ('an', 474), ('well', 470), ('more', 466), ('what', 464), ('awesome',
461), ('from', 439), ('time', 428), ('such', 408), ('watched', 407), ('the
y', 399), ('their', 392), ('waiting', 372), ('me', 370), ('ever', 365),
('characters', 359), ('after', 359), ('every', 350), ('about', 342), ('ca
n', 341), ('much', 329), ('actors', 328), ('at', 327), ('which', 325), ('c
haracter', 317), ('episode', 315), ('drama', 315), ('if', 308), ('cast', 3
01), ('see', 288), ('episodes', 287), ('who', 283), ('am', 277), ('too', 2
68), ('also', 263), ('we', 261), ('some', 260), ('musical', 257), ('dont',
```

Loading [MathJax]/extensions/Safe.js

In [38]:

```
# 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 > 1750]
add_stop_words
```

Out[38]:

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

Loading [MathJax]/extensions/Safe.js

In [39]:

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

```
frozenset({'am', 'find', 'somewhere', 'be', 'whither', 'wherein', 'now', 'th
an', 'amongst', 'six', 'other', 'describe', 'namely', 'well', 'do', 'but',
'ltd', 'nor', 'were', 'show', 'eg', 'also', 'at', 'if', 'its', 'move', 'some
time', 'never', 'can', 'wherever', 'three', 'herein', 'see', 'they', 'thus',
'something', 'become', 'not', 'it', 'then', 'anyhow', 'etc', 'many', 'otherw
ise', 'toward', 'upon', 'a', 'side', 'my', 'nine', 'ie', 'was', 'neither',
'please', 'yourselves', 'have', 'across', 'eleven', 'with', 'everything', 'n
ext', 'alone', 'by', 'serious', 'give', 'along', 'seem', 'i', 'former', 'tha
t', 'less', 'front', 'some', 'of', 'often', 'becoming', 'in', 'everyone', 't
hird', 'where', 'ourselves', 'hasnt', 'whereby', 'hereafter', 'could', 'whoe
ver', 'only', 'into', 'when', 'latterly', 'himself', 'out', 'among', 'the',
'two', 'who', 'therefore', 'beforehand', 'thin', 'most', 'go', 'their', 'sin
cere', 'get', 'whose', 'though', 'this', 'herself', 'sometimes', 'any', 'vi
a', 'your', 'rather', 'myself', 'eight', 'un', 'within', 'is', 'to', 'couldn
t', 'least', 'own', 'thereby', 'until', 'his', 'mostly', 'thru', 'becomes',
'me', 'co', 'without', 'throughout', 'ours', 'whom', 'whereas', 'amount', 'f
ifteen', 'had', 'around', 'hereby', 'up', 'ever', 're', 'full', 'which', 'do
wn', 'both', 'for', 'beyond', 'might', 'whenever', 'whence', 'even', 'yet',
'mine', 'interest', 'somehow', 'more', 'indeed', 'such', 'although', 'toward
s', 'hereupon', 'found', 'over', 'part', 'con', 'mill', 'fifty', 'latter',
'from', 'however', 'has', 'he', 'further', 'during', 'several', 'whether',
'always', 'formerly', 'hence', 'between', 'due', 'take', 'because', 'us', 'a
gainst', 'name', 'seemed', 'as', 'thereafter', 'anyway', 'sixty', 'besides',
'elsewhere', 'these', 'top', 'how', 'behind', 'beside', 'de', 'hundred', 'he
rs', 'same', 'anything', 'call', 'every', 'above', 'inc', 'will', 'back', 's
eeming', 'our', 'except', 'put', 'twelve', 'everywhere', 'been', 'she', 'som
eone', 'there', 'bill', 'one', 'anywhere', 'what', 'else', 'once', 'before',
'empty', 'her', 'few', 'him', 'below', 'nobody', 'on', 'are', 'cry', 'made',
'forty', 'cannot', 'no', 'perhaps', 'an', 'through', 'nevertheless', 'five',
'nothing', 'we', 'so', 'bottom', 'already', 'done', 'noone', 'thereupon', 'f
ill', 'all', 'almost', 'whereupon', 'therein', 'others', 'whatever', 'very',
'first', 'whole', 'per', 'again', 'ten', 'another', 'each', 'fire', 'why',
'would', 'enough', 'much', 'whereafter', 'while', 'together', 'here', 'becam
e', 'cant', 'them', 'none', 'you', 'itself', 'moreover', 'yourself', 'must',
'or', 'since', 'should', 'detail', 'nowhere', 'last', 'keep', 'off', 'thos
e', 'after', 'seems', 'afterwards', 'four', 'anyone', 'amoungst', 'about',
'either', 'may', 'system', 'yours', 'too', 'twenty', 'and', 'thence', 'meanw
hile', 'still', 'themselves', 'onto', 'being', 'under', 'thick'})
```

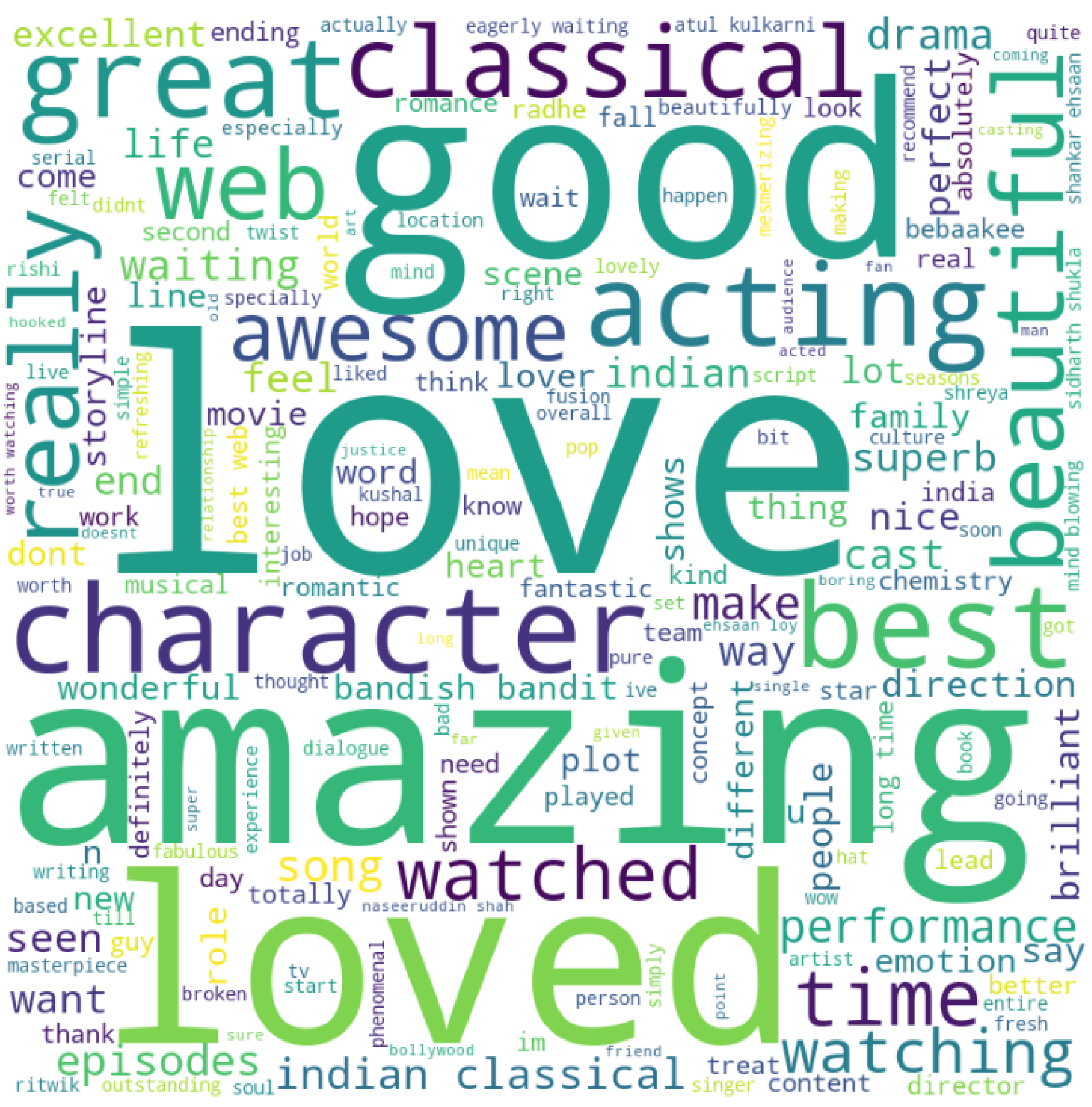
Loading [MathJax]/extensions/Safe.js

In [40]:

```
# 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()
```

Loading [MathJax]/extensions/Safe.js

Sentiment WorldCloud



Loading [MathJax]/extensions/Safe.js

In [41]:

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

```
['am', 'find', 'somewhere', 'be', 'whither', 'wherein', 'now', 'than', 'amon
gst', 'six', 'other', 'describe', 'namely', 'well', 'do', 'but', 'ltd', 'no
r', 'were', 'show', 'eg', 'also', 'at', 'if', 'its', 'move', 'sometime', 'ne
ver', 'can', 'wherever', 'three', 'herein', 'see', 'they', 'thus', 'somethin
g', 'become', 'not', 'it', 'then', 'anyhow', 'etc', 'many', 'otherwise', 'to
ward', 'upon', 'a', 'side', 'my', 'nine', 'ie', 'was', 'neither', 'please',
'yourselves', 'have', 'across', 'eleven', 'with', 'everything', 'next', 'alo
ne', 'by', 'serious', 'give', 'along', 'seem', 'i', 'former', 'that', 'les
s', 'front', 'some', 'of', 'often', 'becoming', 'in', 'everyone', 'third',
'where', 'ourselves', 'hasnt', 'whereby', 'hereafter', 'could', 'whoever',
'only', 'into', 'when', 'latterly', 'himself', 'out', 'among', 'the', 'two',
'who', 'therefore', 'beforehand', 'thin', 'most', 'go', 'their', 'sincere',
'get', 'whose', 'though', 'this', 'herself', 'sometimes', 'any', 'via', 'you
r', 'rather', 'myself', 'eight', 'un', 'within', 'is', 'to', 'couldnt', 'lea
st', 'own', 'thereby', 'until', 'his', 'mostly', 'thru', 'becomes', 'me', 'c
o', 'without', 'throughout', 'ours', 'whom', 'whereas', 'amount', 'fifteen',
'had', 'around', 'hereby', 'up', 'ever', 're', 'full', 'which', 'down', 'bot
h', 'for', 'beyond', 'might', 'whenever', 'whence', 'even', 'yet', 'mine',
'interest', 'somehow', 'more', 'indeed', 'such', 'although', 'towards', 'her
eupon', 'found', 'over', 'part', 'con', 'mill', 'fifty', 'latter', 'from',
'however', 'has', 'he', 'further', 'during', 'several', 'whether', 'always',
'formerly', 'hence', 'between', 'due', 'take', 'because', 'us', 'against',
'name', 'seemed', 'as', 'thereafter', 'anyway', 'sixty', 'besides', 'elsewhe
re', 'these', 'top', 'how', 'behind', 'beside', 'de', 'hundred', 'hers', 'sa
me', 'anything', 'call', 'every', 'above', 'inc', 'will', 'back', 'seeming',
'our', 'except', 'put', 'twelve', 'everywhere', 'been', 'she', 'someone', 't
here', 'bill', 'one', 'anywhere', 'what', 'else', 'once', 'before', 'empty',
'her', 'few', 'him', 'below', 'nobody', 'on', 'are', 'cry', 'made', 'forty',
'cannot', 'no', 'perhaps', 'an', 'through', 'nevertheless', 'five', 'nothin
g', 'we', 'so', 'bottom', 'already', 'done', 'noone', 'thereupon', 'fill',
'all', 'almost', 'whereupon', 'therein', 'others', 'whatever', 'very', 'firs
t', 'whole', 'per', 'again', 'ten', 'another', 'each', 'fire', 'why', 'woul
d', 'enough', 'much', 'whereafter', 'while', 'together', 'here', 'became',
'cant', 'them', 'none', 'you', 'itself', 'moreover', 'yourself', 'must', 'o
r', 'since', 'should', 'detail', 'nowhere', 'last', 'keep', 'off', 'those',
'after', 'seems', 'afterwards', 'four', 'anyone', 'amoungst', 'about', 'eith
er', 'may', 'system', 'yours', 'too', 'twenty', 'and', 'thence', 'meanwhil
e', 'still', 'themselves', 'onto', 'being', 'under', 'thick', 'show', 'seaso
n', 'one', 'season', 'watch', 'story', 'bajpayee', 'webserie', 'webseries',
'bajpai', 'manoj', 'episode', 'review', 'actor', 'actors', 'the', 'and', 'o
f', 'a', 'is', 'to', 'i', 'it', 'this', 'series', 'in', 'for', 'music', 'a
m', "he'd", 'be', 'some', "who's", "aren't", 'cannot', 'no', "they'll", 'mor
e', 'shall', 'does', 'of', "she'll", "wouldn't", 'such', 'an', 'than', 'in',
'through', 'we', 'so', 'over', "we'd", 'other', 'where', "we'll", "here's",
"haven't", 'ourselves', 'doing', "i'll", "you've", 'all', "weren't", 'from',
"you're", 'however', "they're", 'do', "they've", 'could', 'only', 'but', 'we
re', 'nor', 'very', "shan't", 'has', 'into', "wasn't", "shouldn't", 'again',
'he', 'when', 'each', 'himself', 'out', 'why', 'would', 'also', 'further',
'during', "hadn't", 'at', 'if', 'its', 'just', 'com', 'the', 'can', 'while',
"doesn't", 'hence', 'between', "isn't", "there's", "why's", 'because',
"i'm", "hasn't", 'against', 'here', 'therefore', 'who', 'theirs', 'as', 'di
d', "we're", 'they', 'r', "where's", 'most', "he'll", 'these', 'their', 'ge
t', 'how', "couldn't", 'them', "you'd", 'you', 'not', 'it', 'then', 'ought',
'this', 'itself', 'www', "you'll", 'herself', 'otherwise', 'hers', "he's",
'same', "can't", 'any', 'yourself', 'your', 'a', 'my', 'or', 'above', 'sinc
e', 'should', 'was', 'yourselves', 'myself', "they'd", 'our', "won't",
"i'd", 'have', "how's", 'off', "that's", 'is', 'to', "let's", "what's", 'htt
p', 'own', 'until', 'his', 'those', 'been', 'me', 'after', 'she', 'with', 't
here', 'ours', 'whom', "when's", 'what', 'about', "she'd", 'by', "didn't",
"she's", "i've", 'had', "it's", 'else', 'up', 'yours', 'too', 'once', 'eve
r', 'having', 'and', 'i', 'before', 'her', "don't", 'like', 'few', "we've",
'him', 'which', 'themselves', 'down', 'both', 'for', 'below', 'that', 'bein
g', 'under', 'on', "mustn't", 'k', 'are']
```

Findings

We can clearly see that the word cloud has major chunk of positve reviews(roughly 80%) , some negative reviews (roughly 10%), with some neutral reviews(10%).

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.

Loading [MathJax]/extensions/Safe.js

In [42]:

```
# 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[42]:

	transcript	polarity	subjectivity
0	even though i am no expert but i found the cam...	0.466667	0.725000
1	i stumbled upon this series randomly while sur...	0.053571	0.619048
2	i was prejudice towards watching the married w...	-0.025000	0.525000
3	i am sure we will wait a long while till somet...	0.487500	0.722222
4	without thinking and would recommend to every ...	0.400000	0.500000
...	...	...	...
6040	the only good thing about series is rohit	0.350000	0.800000
6041	it was wasted of time	-0.200000	0.000000
6042	hated it total waste of time	-0.366667	0.483333
6043	godddsuch a waste of time	-0.200000	0.000000
6044	worst show everreally disappointed	-0.875000	0.875000

6045 rows × 3 columns

Loading [MathJax]/extensions/Safe.js

In [43]:

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

Out[43]:

	transcript	polarity	subjectivity
3934	outstanding series no words for this series ku...	0.75	0.5875
4488	trust me go and watch this show as soon as pos...	0.00	1.0000
4079	kushal tandon just outstanding	0.50	0.8750
1942	i never really knew what classical music was n...	0.22	0.4000
5894	its a lovely film and wishing to watch more fi...	0.50	0.6250

In [44]:

```
#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[44]:

	transcript	polarity	subjectivity	Analysis
0	even though i am no expert but i found the cam...	0.466667	0.725000	positive
1	i stumbled upon this series randomly while sur...	0.053571	0.619048	positive
2	i was prejudice towards watching the married w...	-0.025000	0.525000	negative
3	i am sure we will wait a long while till somet...	0.487500	0.722222	positive
4	without thinking and would recommend to every ...	0.400000	0.500000	positive
...	...	...	...	...
6040	the only good thing about series is rohit	0.350000	0.800000	positive
6041	it was wasted of time	-0.200000	0.000000	negative
6042	hated it total waste of time	-0.366667	0.483333	negative
6043	godddsuch a waste of time	-0.200000	0.000000	negative
6044	worst show everreally disappointed	-0.875000	0.875000	negative

6045 rows × 4 columns

Loading [MathJax]/extensions/Safe.js

In [45]:

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

Out[45]:

	transcript	polarity	subjectivity	Analysis
145	i have given it star rating why because i coul...	0.187500	0.437500	positive
1216	it started pretty well with shankar ehsaan loy...	0.126768	0.487626	positive
5380	a delight to watch excellent casting new comer...	0.789394	0.825758	positive
800	you can go for itwatched this during the quara...	0.000000	0.000000	neutral
5383	in starting it was difficult to understand the...	0.224167	0.760833	positive
4367	while watching the pilot episode of bebaakee t...	0.500000	1.000000	positive
3150	what a show i admire ekta kapoors imagination ...	0.900000	1.000000	positive
1750	one of the best hindi web series ive ever watc...	0.500000	0.150000	positive
217	each everyone associated to it gave best to ma...	0.589286	0.517857	positive
2861	excellent music super acting by all actors	0.444444	0.555556	positive

Loading [MathJax]/extensions/Safe.js

In [47]:

```
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(Roman
print ('\nNo of Negative Reviews from our Total DataSet(around 10k) ->',j)
print ('No of Positive Reviews from our Total DataSet(around 10k) ->',k)
print ('No of  Neutral Reviews from our Total DataSet(around 10k) ->',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(Romance Genre) :

No of Negative Reviews from our Total DataSet(around 10k) -> 396
No of Positive Reviews from our Total DataSet(around 10k) -> 5252
No of  Neutral Reviews from our Total DataSet(around 10k) -> 397

Percentage of Negative Reviews -> 6.550868486352357 %
Percentage of Positive Reviews -> 86.88172043010752 %
Percentage of Neutral  Reviews -> 6.567411083540116  %
```

Sentiment Findings:

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

```
No of Negative Reviews from our Total DataSet(around 10k) -> 396
No of Positive Reviews from our Total DataSet(around 10k) -> 5252
No of  Neutral Reviews from our Total DataSet(around 10k) -> 397

Percentage of Negative Reviews -> 6.550868486352357 %
Percentage of Positive Reviews -> 86.88172043010752 %
Percentage of Neutral  Reviews -> 6.567411083540116  %

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

Loading [MathJax]/extensions/Safe.js

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

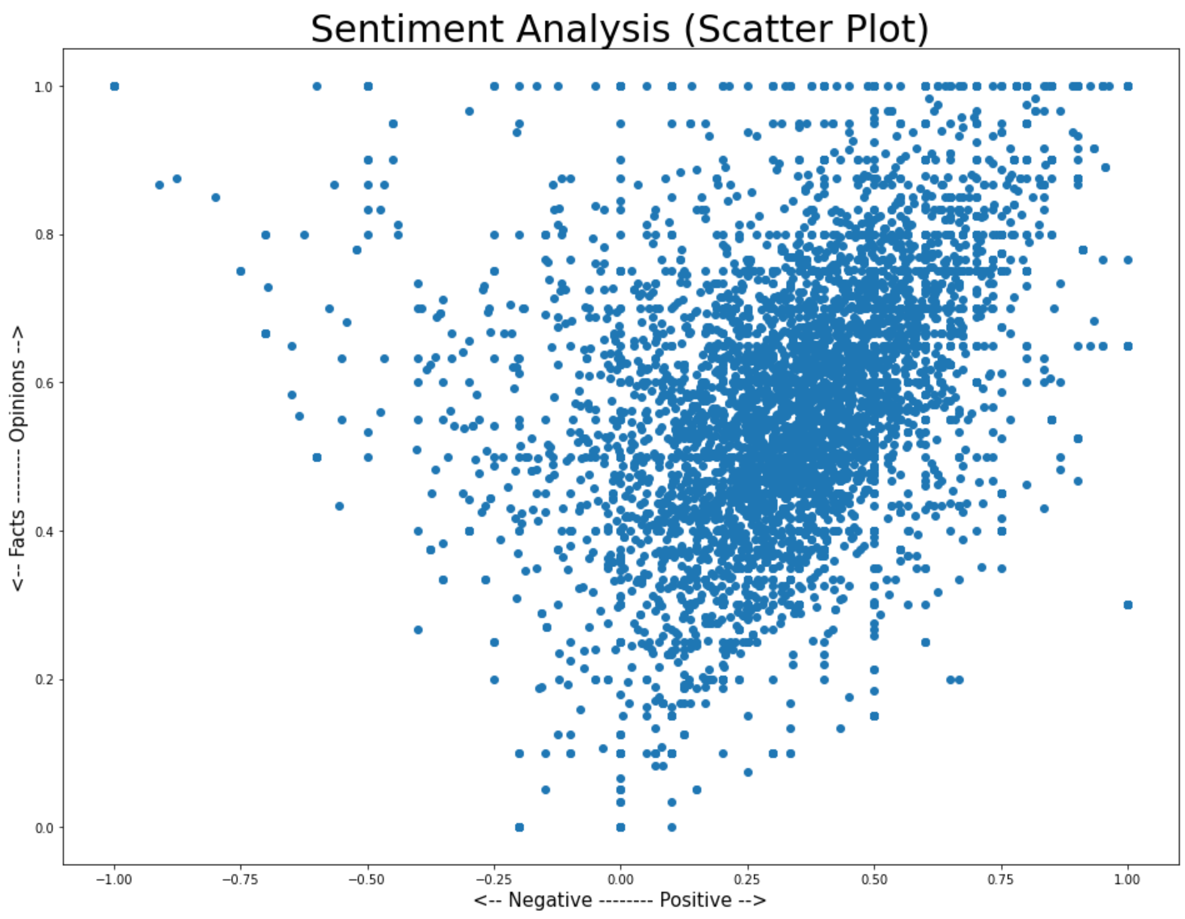
Our eyes are drawn to colors and patterns. We can quickly identify red from blue, square from circle. Our culture is visual, including everything from art and advertisements to TV and movies. Data visualization is another form of visual art that grabs our interest and keeps our eyes on the message. When 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 [48]:

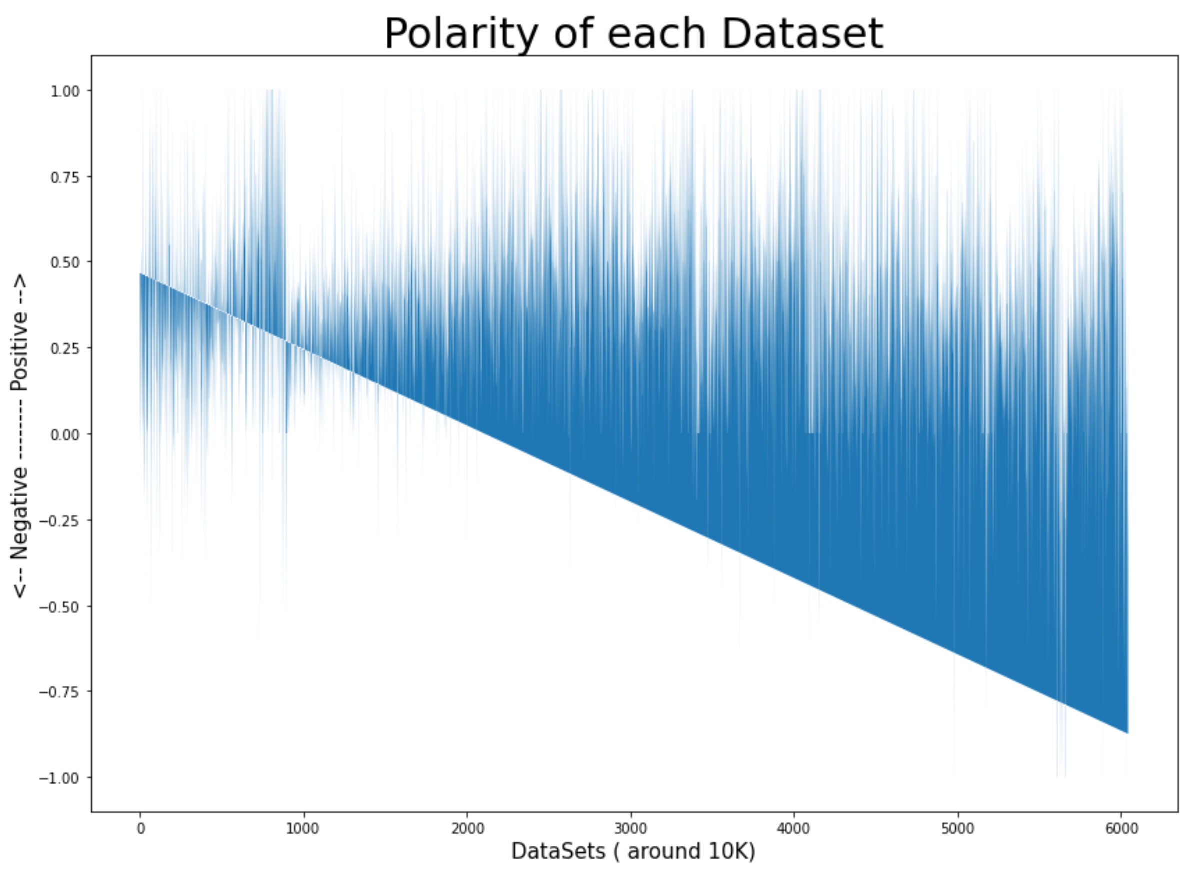
```
# Let's plot the results
import matplotlib.pyplot as plt

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()
```



Loading [MathJax]/extensions/Safe.js

In [56]:

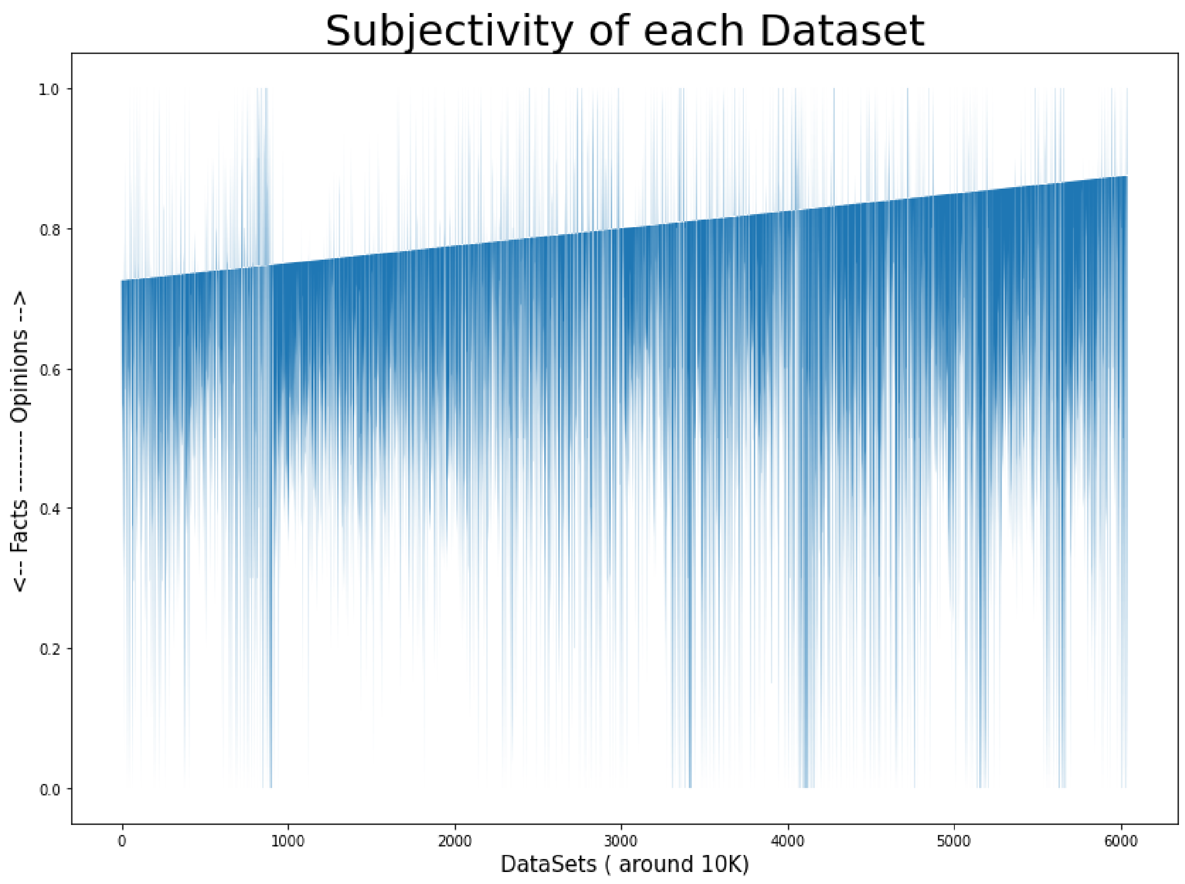
```
plt.fill(data_clean_df['polarity'])
plt.title('Polarity of each Dataset', fontsize=30)
plt.xlabel('DataSets ( around 10K)', fontsize=15)
plt.ylabel('<-- Negative -------- Positive -->', fontsize=15)

plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [57]:

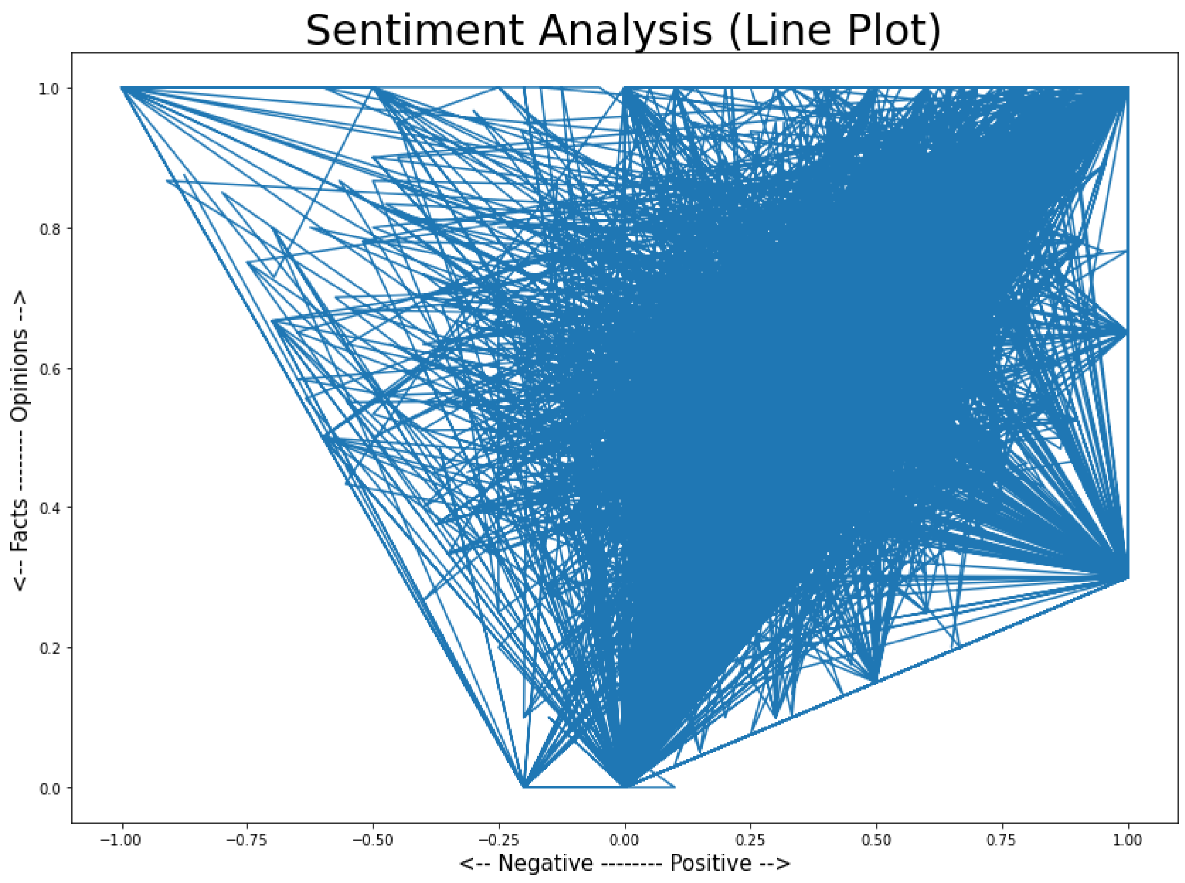
```
plt.fill(data_clean_df['subjectivity'])
plt.title('Subjectivity of each Dataset', fontsize=30)
plt.xlabel('DataSets ( around 10K)', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [58]:

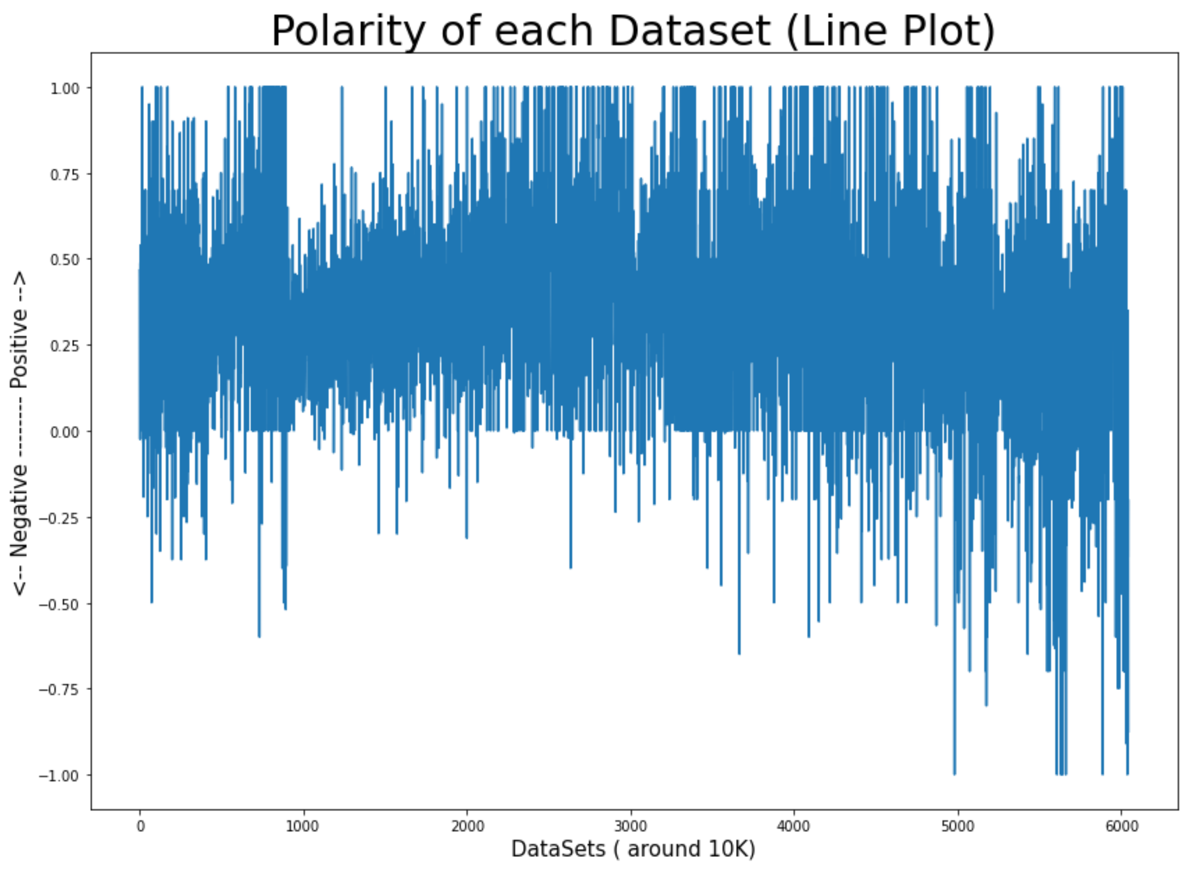
```
plt.plot(data_clean_df['polarity'],data_clean_df['subjectivity'])
plt.rcParams['figure.figsize'] = [14, 10]
plt.title('Sentiment Analysis (Line Plot)', fontsize=30)
plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)

plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [59]:

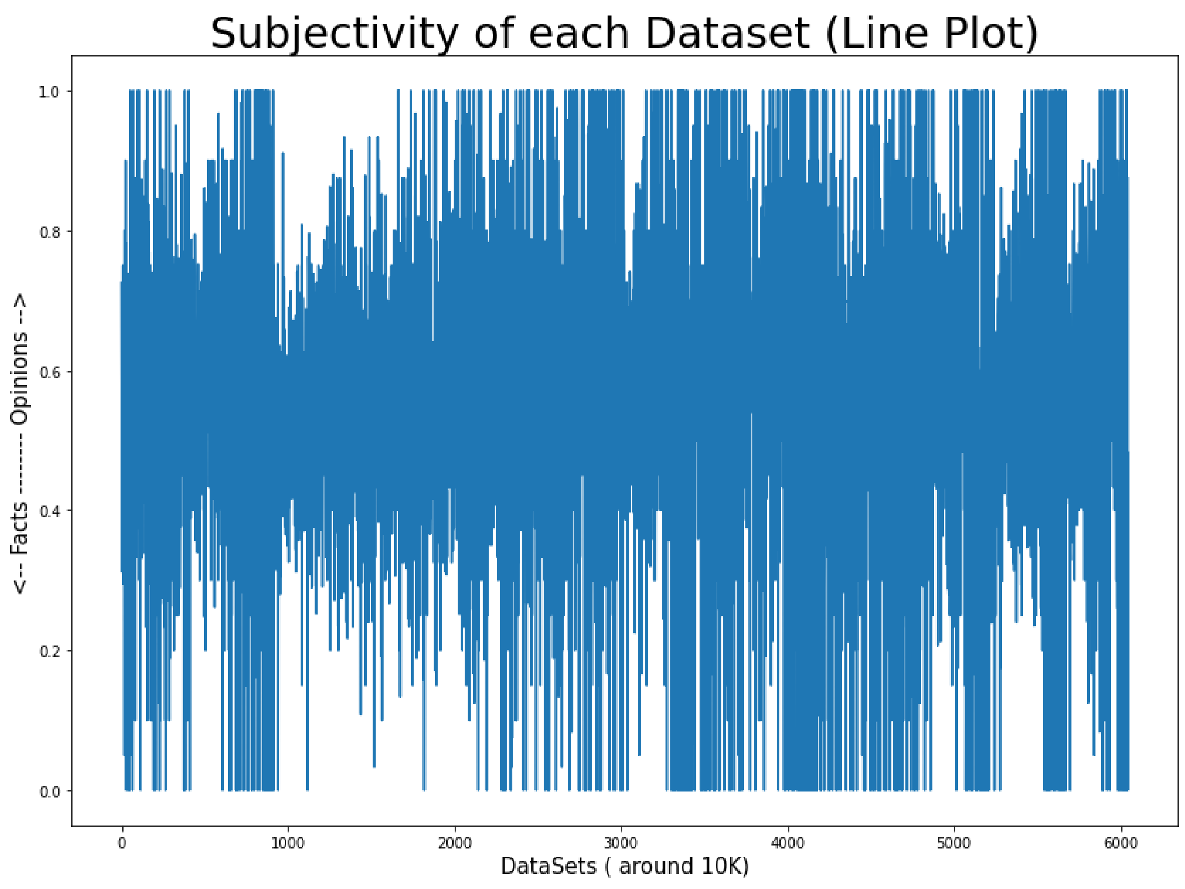
```
plt.plot(data_clean_df['polarity'])
plt.title('Polarity of each Dataset (Line Plot)', fontsize=30)
plt.xlabel('DataSets ( around 10K)', fontsize=15)
plt.ylabel('<-- Negative -------- Positive -->', fontsize=15)

plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [60]:

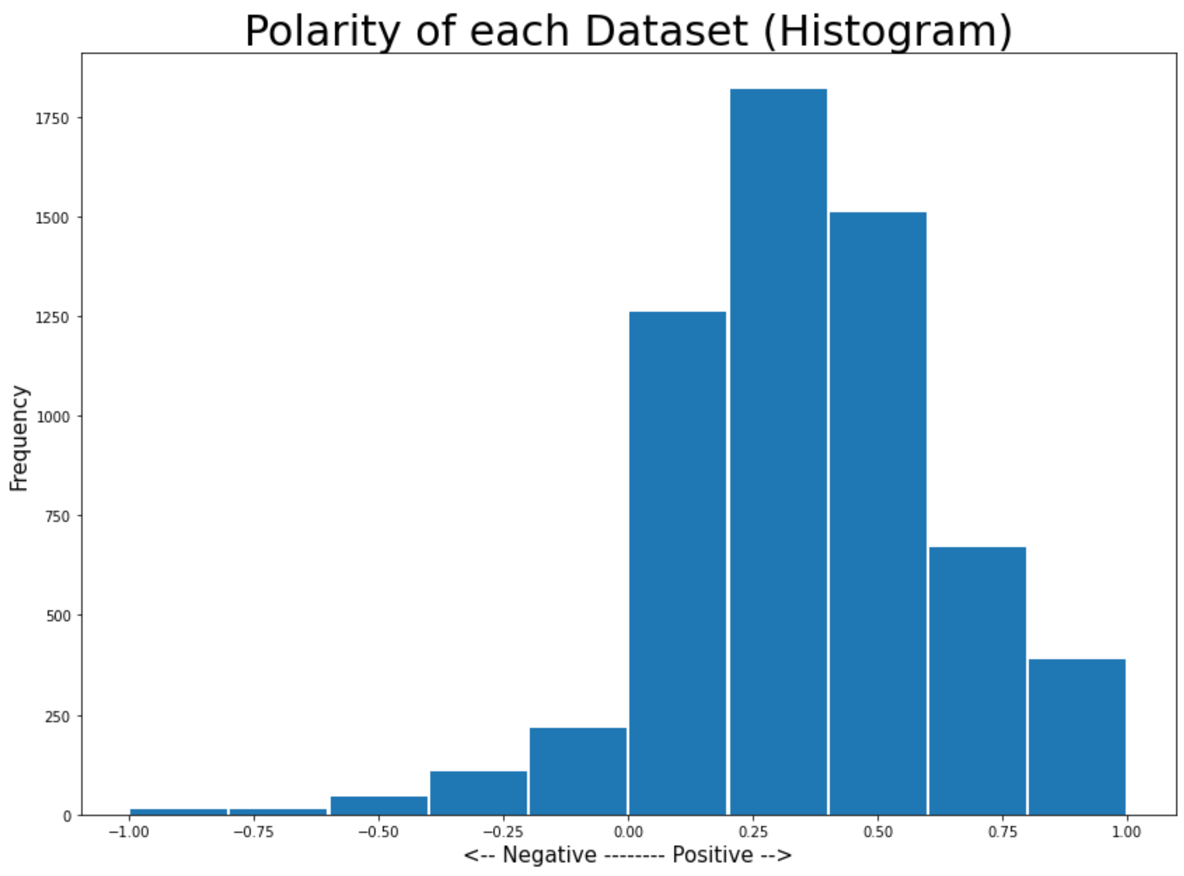
```
plt.plot(data_clean_df['subjectivity'])
plt.title('Subjectivity of each Dataset (Line Plot)', fontsize=30)
plt.xlabel('DataSets ( around 10K)', fontsize=15)
plt.ylabel('<-- Facts -------- Opinions -->', fontsize=15)


plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [61]:

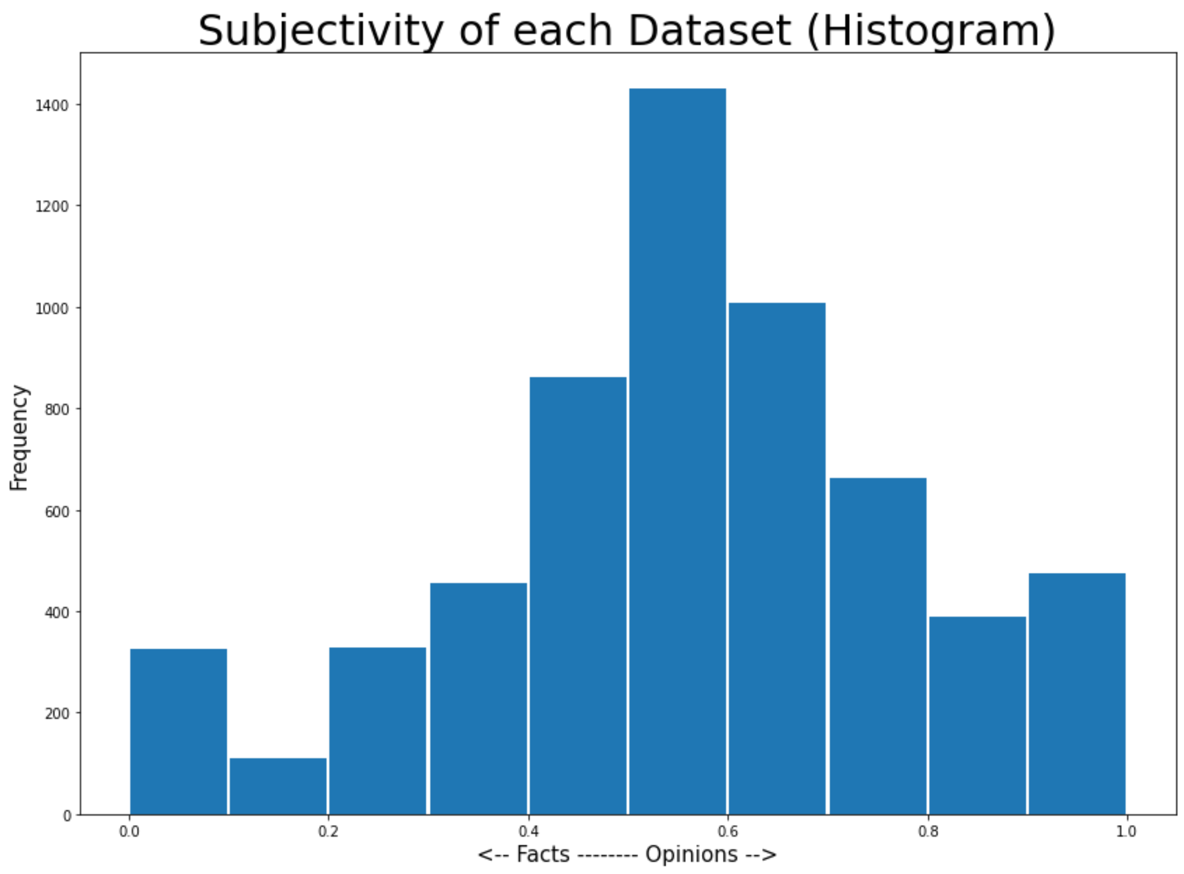
```
plt.hist(data_clean_df['polarity'], rwidth=.969)
plt.title('Polarity of each Dataset (Histogram)', fontsize=30)
plt.xlabel('<-- Negative -------- Positive -->', fontsize=15)
plt.ylabel('Frequency', fontsize=15)

plt.show()
```



Loading [MathJax]/extensions/Safe.js

In [62]:

```
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()
```



Loading [MathJax]/extensions/Safe.js

In [63]:

```
data_clean_df
```

Out[63]:

	transcript	polarity	subjectivity	Analysis
0	even though i am no expert but i found the cam...	0.466667	0.725000	positive
1	i stumbled upon this series randomly while sur...	0.053571	0.619048	positive
2	i was prejudice towards watching the married w...	-0.025000	0.525000	negative
3	i am sure we will wait a long while till somet...	0.487500	0.722222	positive
4	without thinking and would recommend to every ...	0.400000	0.500000	positive
...	...	...	...	...
6040	the only good thing about series is rohit	0.350000	0.800000	positive
6041	it was wasted of time	-0.200000	0.000000	negative
6042	hated it total waste of time	-0.366667	0.483333	negative
6043	godddsuch a waste of time	-0.200000	0.000000	negative
6044	worst show everreally disappointed	-0.875000	0.875000	negative

6045 rows × 4 columns

In [64]:

```
#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 [65]:

```
positive_df
```

Out[65]:

	transcript	polarity	subjectivity	Analysis
0	even though i am no expert but i found the cam...	0.466667	0.725000	positive
1	i stumbled upon this series randomly while sur...	0.053571	0.619048	positive
3	i am sure we will wait a long while till somet...	0.487500	0.722222	positive
4	without thinking and would recommend to every ...	0.400000	0.500000	positive
5	i recommend this series script writing casting...	0.061905	0.311905	positive
...	...	...	...	...
6029	loved it	0.700000	0.800000	positive
6031	back with season good luck	0.350000	0.300000	positive
6036	everything was going perfect bad ending	0.150000	0.833333	positive
6037	best story but tragic ending	0.125000	0.525000	positive
6040	the only good thing about series is rohit	0.350000	0.800000	positive

5252 rows × 4 columns

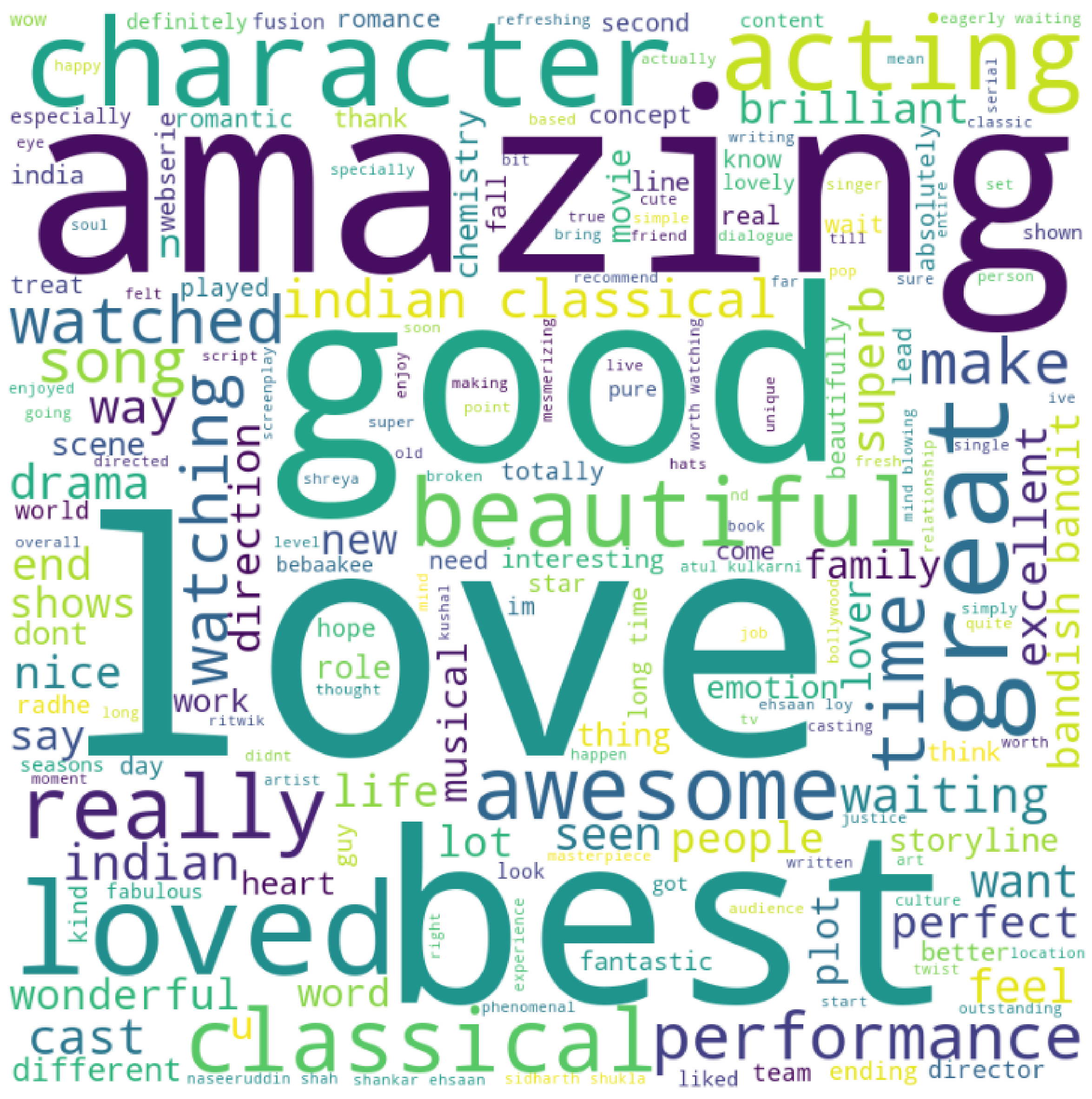
Loading [MathJax]/extensions/Safe.js

In [66]:

```
# 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','episodes','bajpa
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()
```

Loading [MathJax]/extensions/Safe.js

WordCloud for POSITVE SENTIMENTS



In [67]:

```
#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 [68]:

```
negative_df
```

Out[68]:

	transcript	polarity	subjectivity	Analysis
2	i was prejudice towards watching the married w...	-0.025000	0.525000	negative
20	his show has amazing actors amazing story ill ...	-0.034286	0.782857	negative
23	the series is just going round and round on on...	-0.065152	0.571212	negative
24	below average slow and boring at some timesome...	-0.192857	0.700000	negative
25	will not recommend to anyone main plot is ok b...	-0.022024	0.527679	negative
...	...	...	...	...
6039	a beautiful fairy tale show with the worst end...	-0.050000	1.000000	negative
6041	it was wasted of time	-0.200000	0.000000	negative
6042	hated it total waste of time	-0.366667	0.483333	negative
6043	godddsuch a waste of time	-0.200000	0.000000	negative
6044	worst show everreally disappointed	-0.875000	0.875000	negative

396 rows × 4 columns

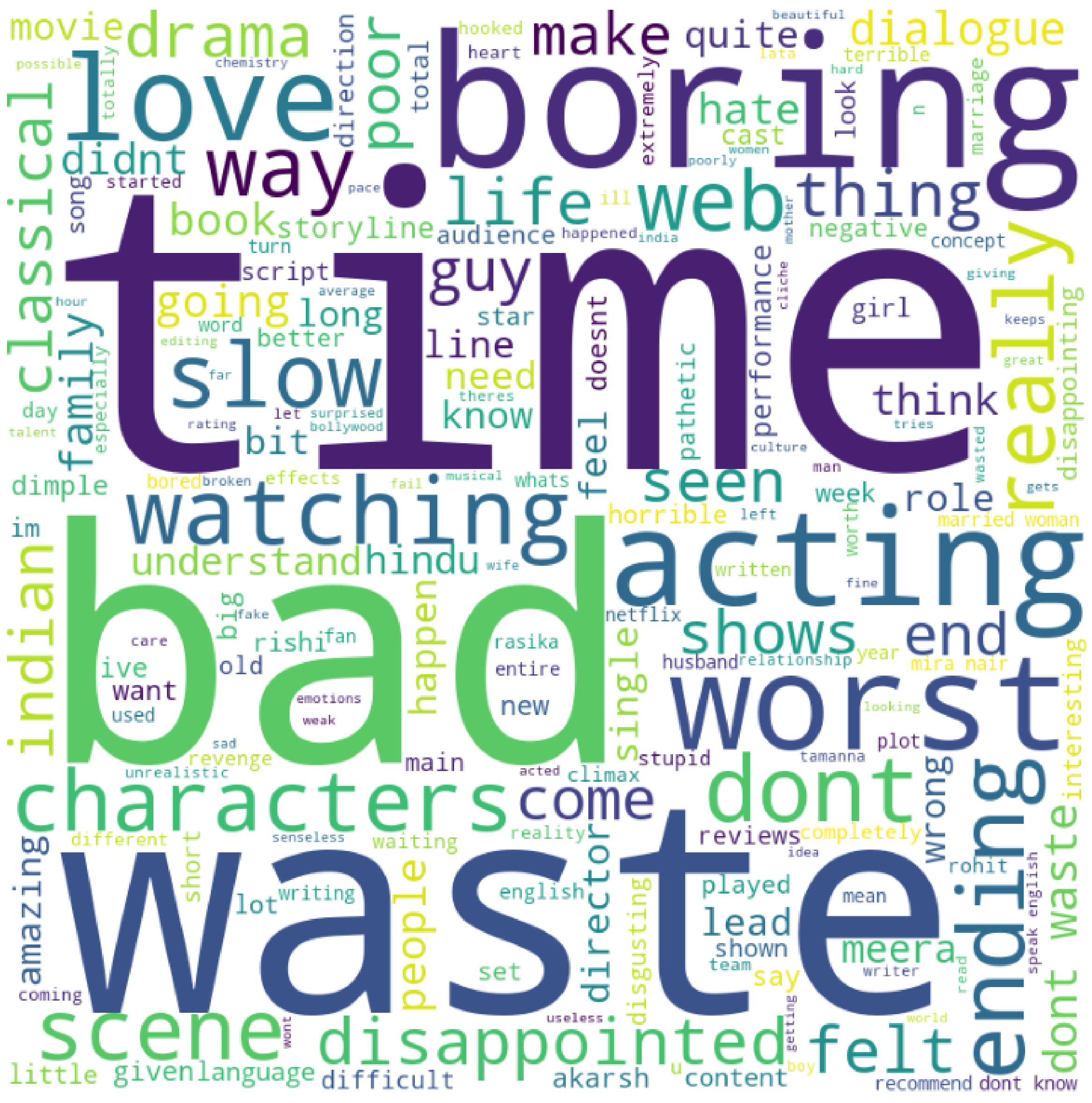
Loading [MathJax]/extensions/Safe.js

In [72]:

```
# 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','character','wat
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()
```

Loading [MathJax]/extensions/Safe.js

WordCloud for NEGATVE SENTIMENTS



In [73]:

```
#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']
```

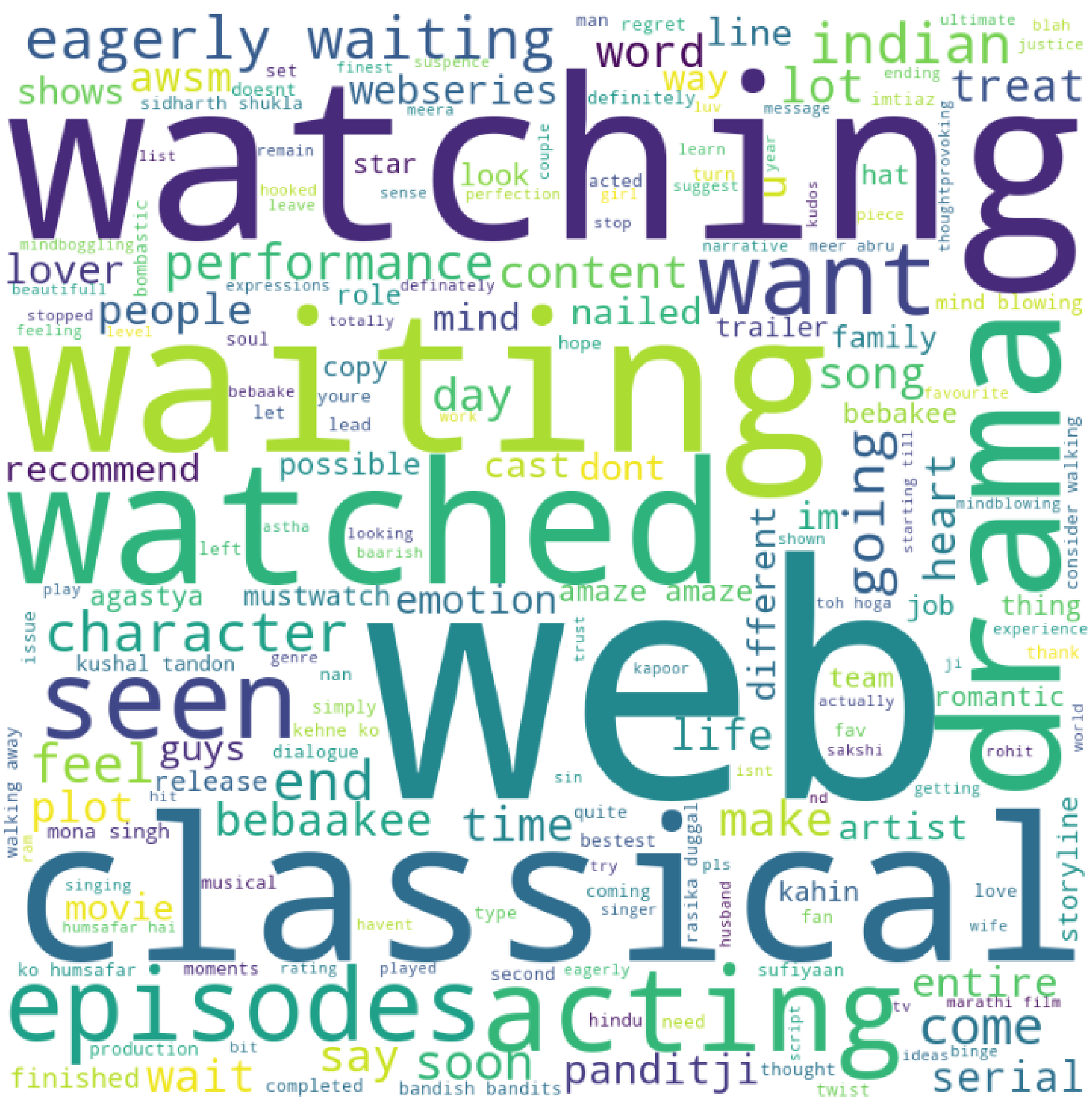
Loading [MathJax]/extensions/Safe.js

In [74]:

```
# 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','watch','stor
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()
```

Loading [MathJax]/extensions/Safe.js

WordCloud for NEUTRAL SENTIMENTS



Additonal Information

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

Loading [MathJax]/extensions/Safe.js

In [75]:

```
# 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', 6857), ('and', 4763), ('of', 3499), ('a', 3399), ('is', 2996),
('i', 2911), ('to', 2820), ('it', 2492), ('series', 2432), ('this', 2430),
('in', 1927), ('music', 1696), ('for', 1633), ('love', 1460), ('show', 132
4), ('with', 1212), ('watch', 1114), ('you', 1049), ('its', 1042), ('stor
y', 1024), ('was', 1020), ('all', 967), ('season', 905), ('but', 895), ('s
o', 891), ('just', 888), ('have', 841), ('that', 838), ('are', 836), ('bes
t', 813), ('very', 799), ('classical', 793), ('amazing', 789), ('good', 78
7), ('one', 741), ('loved', 726), ('by', 676), ('great', 654), ('acting',
645), ('web', 630), ('on', 588), ('indian', 578), ('has', 555), ('really',
548), ('like', 543), ('as', 540), ('not', 538), ('will', 504), ('must', 49
5), ('beautiful', 472), ('awesome', 461), ('more', 450), ('be', 449), ('wa
tching', 449), ('my', 439), ('well', 438), ('an', 432), ('what', 400), ('f
rom', 385), ('such', 370), ('watched', 365), ('their', 356), ('time', 35
2), ('me', 346), ('they', 342), ('ever', 341), ('waiting', 333), ('charact
ers', 332), ('after', 327), ('every', 322), ('much', 311), ('actors', 29
9), ('can', 297), ('which', 295), ('about', 294), ('cast', 288), ('drama',
286), ('character', 278), ('at', 277), ('episode', 270), ('if', 268), ('se
e', 261), ('am', 251), ('too', 248), ('episodes', 248), ('musical', 248),
('who', 247), ('also', 245), ('superb', 243), ('we', 233), ('how', 229),
```

Loading [MathJax]/extensions/Safe.js

In [76]:

```
# 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', 509), ('and', 339), ('a', 287), ('to', 274), ('is', 267), ('of',
223), ('i', 197), ('it', 179), ('this', 158), ('in', 151), ('series', 12
4), ('but', 107), ('with', 96), ('show', 95), ('for', 92), ('was', 89),
('not', 88), ('that', 86), ('have', 84), ('are', 73), ('you', 72), ('so',
71), ('just', 69), ('time', 68), ('watch', 64), ('its', 64), ('like', 62),
('story', 61), ('as', 61), ('season', 56), ('all', 55), ('on', 54), ('wast
e', 54), ('be', 51), ('very', 50), ('bad', 47), ('dont', 47), ('what', 4
5), ('no', 45), ('they', 45), ('by', 43), ('one', 42), ('about', 41), ('a
t', 40), ('has', 39), ('or', 38), ('from', 37), ('music', 37), ('boring',
36), ('acting', 36), ('worst', 36), ('good', 34), ('character', 33), ('epi
sode', 32), ('only', 32), ('some', 31), ('can', 31), ('why', 28), ('even',
28), ('episodes', 27), ('there', 27), ('their', 27), ('watched', 27), ('h
e', 27), ('watching', 26), ('such', 26), ('any', 26), ('love', 26), ('you
r', 26), ('her', 26), ('other', 26), ('which', 26), ('his', 25), ('will',
25), ('who', 25), ('slow', 24), ('after', 24), ('if', 24), ('when', 23),
('an', 23), ('out', 23), ('ending', 23), ('do', 23), ('my', 22), ('she', 2
2), ('characters', 22), ('we', 22), ('never', 21), ('actors', 21), ('web',
21), ('really', 21), ('could', 20), ('should', 20), ('disappointed', 20),
('well', 19), ('how', 19), ('felt', 19), ('know', 19), ('family', 18), ('w
```

Loading [MathJax]/extensions/Safe.js

In [77]:

```python
# 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', 236), ('and', 140), ('a', 136), ('of', 131), ('to', 124), ('thi
s', 115), ('is', 109), ('i', 92), ('series', 87), ('it', 85), ('for', 83),
('watch', 78), ('in', 70), ('season', 67), ('show', 64), ('you', 54), ('al
l', 40), ('as', 38), ('just', 37), ('must', 36), ('that', 34), ('have', 3
3), ('with', 30), ('has', 29), ('web', 29), ('waiting', 28), ('one', 27),
('music', 27), ('not', 26), ('on', 26), ('my', 25), ('its', 23), ('story',
22), ('will', 22), ('but', 21), ('an', 19), ('what', 19), ('by', 19), ('ar
e', 19), ('cant', 18), ('watching', 18), ('next', 18), ('be', 17), ('pleas
e', 17), ('from', 17), ('see', 17), ('like', 17), ('classical', 17), ('eve
ry', 16), ('if', 16), ('her', 15), ('eagerly', 15), ('watched', 15), ('nev
er', 14), ('drama', 14), ('so', 14), ('acting', 14), ('episode', 13), ('we
ll', 13), ('can', 13), ('go', 13), ('me', 13), ('they', 12), ('when', 12),
('seen', 12), ('such', 12), ('was', 12), ('episodes', 12), ('want', 12),
('again', 12), ('ever', 11), ('end', 11), ('your', 11), ('going', 11), ('w
ho', 11), ('made', 10), ('come', 10), ('at', 10), ('indian', 10), ('mind',
10), ('am', 10), ('back', 10), ('soon', 10), ('everyone', 9), ('there',
9), ('feel', 9), ('performance', 9), ('should', 9), ('only', 9), ('been',
9), ('their', 9), ('into', 9), ('out', 9), ('content', 8), ('she', 8), ('n
o', 8), ('would', 8), ('treat', 8), ('after', 8), ('wait', 8), ('get', 8),
```

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)

Loading [MathJax]/extensions/Safe.js