Sentiment Analysis of People's opinion on High Speed Train

Abstract

Sentiment analysis is a popular theme in the natural language processing (NLP) domain. In this modern era online social media provide a platform for the people to share their experiences in the form of text, images and videos. YouTube is one of the well known social media platform across the world. In this study, we are using the comments done by the people on the YouTube Videos in respect to their opinion on High Speed Trains. This data is used to analyze the hidden sentiments using machine learning techniques such as Naive Bayes Classifier, Random forest (RF) and K Nearest Neighbors (KNN). Till now, the results show that Naive Bayes Classifier provides high accuracy with more training on the data. The results achieved from KNN and RF was also satisfactory but couldn't outperform Naive Bayes Classifier.

Contents

Introduction

Sentiments or opinions can be defined as a mental situation or feeling of a person in certain circumstances and conditions. These feelings may be a reflection of joy, sadness, discomfort or nervousness. Present world is full of technology and smart devices. Moreover, as stated above people love to share their views and feelings based on a particular topic on various internet platforms.

In this documentary we will try to draw out the Sentiments of people about what they feel regarding the functioning of high speed rails in India through the comments they had done on YouTube.

Materials and Methods

Data Gathering

The data has been gathered by scrapping the YouTube links related to the High Speed Rails.
The code for the same is attached in the Appendix.

The raw scrapped data is a csv file with around 25000 records in it.

	A	B	C	D	E
1	cid	text	time	author	channel
2	UgzSJRVOCZA0C4VXtYt4AaABAg	First you ask yourself, what is the benefit of 4g over 2g. then you will understand how bullet train is going t	1 year ago	Nitheesh Kumar	UCjh9s_65o5c7Z6SBZHEg5wQ
3	UgzBwbvfyZOSEhUpeRd4AaABAg	other progress and improvements of Indian railways are already catching momentum. They have not stop	3 years ago	Gurmeet Anand	UCLt7tLHhfDj_VXyTLAVgLpKQ
4	UgzWQXS9NYwa7Bdbi-54AaABAg	The bullet train is just the start. India needs to make more Highways and Freeways so goods can be distrib	4 years ago	Ã˜Ã¿Ã«Ã¯ Ã†ÃŸÃ±	UCryrYEvV0pxKOwQgK4wp8XQ
5	Ugy44dbTkj8rVNyVtAd4AaABAg	The same type of Gyan was distributed on the time of Delhi Metro. . . . . I still remember the way Mr Ma	3 years ago	Anil Kumar	UCqzd9KoBeC12qRaTlvvA9yw
6	UgxQkIE7R23I5_6ltZp4AaABAg	There is always two ways in life .. 1. Renovation....	4 years ago	Shyamali Setua	UCr3B7VfVZFl3O8WerMw5sCQ
7	UgzTTxuKBCKDlVSwhOt4AaABAg	Taught of slaves, Think like Business man, Why Bullet train came hit in other countries? Without knowing t	3 years ago	saravana prabu	UCjVox_Hvseu26qCMX7ggV-g
8	Ugy_oIXL3ORNGBkL72d4AaABAg	Bullet train se hamara growth aur hoga hum indian ko iska sath dena chahiye iska wirodh Nahi karna chahi	3 years ago	social worker	UCtzkZkQSLJNw4TsOqJPR74w
9	UgxHuVlaqLq7DHYXSrh4AaABAg	its not the fare which will pay off but actually the jobs directly or indirectly created alongwith the compani	4 years ago	BestIndiaDeals Fin	UCh6T15D5ByfIOZbxVJkceZg
10	Ugw94TrYMumZcbtVIpR4AaABAg	I think government has already considered all these points or i should say beyond these points. So i think tl	4 years ago	varun mathur	UCAVr1pE-Gq9DWLBGbN3sQ8A
11	UgyULHYvP-kTpHtplVB4AaABAg	à¤¦à¥‡à¤¶à¤•à¤« à¤¬à¤¹à¥ à¤¤ à¤¬à¤¢à¤¿à¤¯à¤¾ à¤¸à¥‹à¤š à¤¹à¥ˆ à¤¦à¥‡à¤¶	3 years ago	Sanjay Kamble	UCfnh2-9kOYosTNHA8ONNcRg
12	UggbDxE5of7BRSeJlZ4AaABAg	This is they way technology transfer from one country to another.This decision is very worthful in a long r	3 years ago	Prabhash Mishra	UCgwfZgSstoQF6KmxR7KvvMQ
13	Ugz-cKj17pGNOQR2pYF4AaABAg	Bullet train is good decision	1 year ago	PAPER CRAFT	UC1lkP7i7B73J4crgk7urEcg
14	UgwP-Sq-t0ZLjfaKdil4AaABAg	In lift if don't move upword U can't reach your Room like that if we don't implay new technology we can't	3 years ago	Indubhusan Behera	UCyeN2qeQYnL2cYuUuqAYf_Q
15	UgyRHdqQpL9crJIId3V4AaABAg	Big calculations you produced dear. I think you are also a student from IIM Ahmedabad to which you said	4 years ago	Kishor Kumar	UCIGaPtKVdUBDm30RluEqMQQ
16	UgyONzjPze49zdVAq8N4AaABAg	It's better to connect smaller cities through hydrogen fuel cell powered planes.	3 months ago	Atanu Chaudhury	UCAoefbwsHfPXx0Is6k0_5QQ
17	Ugz_rXaPXuoGhM2Z2PF4AaABAg	ALL RAIL SHOULD BE HIGH SPEED IF WE WANT TO BE SUPER POWER UPTO 2050	8 months ago	Pramod Mehta	UCSZ6E0w7VHV2yBmV_clKfRg
18	Ugytc2VF3TikQhVO0jt4AaABAg	Government is also spending crores of rupees on other railway projects in India (eastern freight corridors,	3 years ago	sanjay agarwal	UC9t1OKViylnkIx-4uwQDKpg
19	Ugx-KV0JSav1DzZsYdB4AaABAg	If you talk sense then you will get credibility, otherwise, you are talking nonsense.	3 years ago	Bharat Dalal	UCZALAlSB9iYGyeqcX58J9zQ
20	Ugzdjpwmp Pi0-Hlgf9d4AaABAg	2g=1.84 lakh crore ,koyla=12 lakh crore,common wealth=1.74 lakh crore ....itna paisa congress ne ghotala	4 years ago	Rahul Sah	UC1GWX7aPBNur2xCSfNmLNBw
21	UgzsxYu5Y_67FdN_l3B4AaABAg	In my opinion it might be a bad decision for now seeing the circumstances but in the long run as Japan ben	4 years ago	Armaan Goyal	UCYmwub1tIxWMKwllMSXll6w
22	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8g_iWrItP1K	There are many mega projects are under way for the same.	3 years ago	Naastik	UC5xCqgDGCjuiAM-sooWFRgQ
23	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8grnxQhTf8z	Ã˜Ã¿Ã«Ã¯ Ã†ÃŸÃ± Bullet trains will not carry goods. It will carry only elites as only they can afford it, not th	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
24	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hLMw0D2Ilg	Hyperloop	3 years ago	Yash Singh	UCwJddgeNWq6tRkFrM5VRcJQ
25	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hMBHYNb_tb	Yash Singh when all agricultural lands will be acquired for highways, freeways, airports, bullet trains etc wh	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
26	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hMpHfTfeWd	Rajeshwarprasad Shukla lol....u think so actually agricultural fields are also increasing day by day....	3 years ago	Yash Singh	UCwJddgeNWq6tRkFrM5VRcJQ
27	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hMpXR0sPkm	Rajeshwarprasad Shukla what do u mean by minded people lol....these trains are actually important in citi	3 years ago	Yash Singh	UCwJddgeNWq6tRkFrM5VRcJQ
28	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hMyZK18M40	Yash Singh but the bullet project is not limited to only Mumbai or Ahmadabad city. Can you tell me how m	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
29	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hMz0Tm0FQ2	Yash Singh Blinded development will lead to make us beggars, begging for food and water in front of forei	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
30	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hUJqxCPt1f	Yash Singh à¤ªà¤¿à¤›à¤¡à¤¼à¤¾ à¤¹à¥à¤† à¤¹à¥ˆ, à¤¸à¤¬à¤•à¥‹, à¤•à¤¿à¤¸à¥€ à¤¨à¥‡ à¤¸à¥‹à¤šà¤¾	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
31	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hUKHAA_C08	à¤¯à¤• à¤•à¤¿à¤¸à¤¾à¤¨ à¤¹à¥‹à¤•à¤° à¤¯à¤¹ à¤¸à¥‹à¤š à¤°à¤¹à¤¾ à¤¹à¥ˆ à¤•à¤¿ à¤•à¤¾à¤¶ à¤¯à¤¹	3 years ago	Rajeshwarprasad Balik	UC8cHLkI9rtzwg7Ac6PMJmYg
32	UgzWQXS9NYwa7Bdbi-54AaABAg.8cnbi8sLowU8hbaofqtUudA	True good idea	3 years ago	Arshvan Khan	UCdiQ1_BVDv16S2hpCQSw2HA

Data Cleaning

Data Cleaning was one of the most important part of this project. As the data was scrapped from the YouTube Comments, it was containing a lot of irregularities such as use of mixed languages, emojis and lots of special characters in it, thus it

was necessary for us to remove it otherwise it would result in deprecating the accuracy of the model.

This removal however did not affect the results whatsoever since the word preprocessing and tokenization we implemented through Scikit-learn (CountVectorizer) only considers alphanumeric characters.

Also, Regex and character replacing were used to make all datasets adhere to the same format.

In [7]:
```
for i in range(len(df)):
    df.loc[[i],['votes']]=int(df.loc[i]['votes'])#converting votes to int
    df.loc[[i],['text']]=df.loc[i]['text'].lower()#converting text to lower case
    df.loc[[i],['text']]=re.sub('[^a-zA-Z 0-9]','',df.text[i])#removing all the characters except a-z,A-Z,0-9 and space
df
```

Word Tokenization

In [8]:
```
#Word Tokenization and removing the words that are not the part of English Language

e = enchant.Dict("en_US")
a=[]
for i in range(len(df)):
    b=[]
    j=nltk.word_tokenize(df.text[i])
    for k in j:
        if(e.check(k)==True):
            b.append(k)
    a.append(b)
print(a)
```

Lemmatization and Stopwords Removal

In [9]:
```
# lemmatizing and stopwords removal

from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords

lemmatizer=WordNetLemmatizer()

d=[]
for i in a:
    b=''
    for j in range(len(i)):
        if(i[j] not in stopwords.words('english') and len(i[j])>3):
            b=b+lemmatizer.lemmatize(i[j])+' '
    d.append(b)
d
```

Finally after the data cleaning our dataset looks something like this:

	processed_comments	votes
0	first benefit understand bullet train going be...	9
1	progress improvement railway already catching ...	3
2	bullet train start need make highway freeway g...	50
3	type distributed time metro still remember opp...	26
4	always way life repair create traditional foot...	0
...	...	...
18634	know percent population poverty line want gove...	2
18635	double line	0
18636	project approve survey completion work start n...	3
18637	state contribution direct central	0
18638	stupidity limit project bring people poverty t...	2

18639 rows × 2 columns

Labeling the unlabeled Data:

Now since we had scrapped the data manually from the YouTube it do not contains label suggesting the polarity of the sentences.
Thus in order to deal with this python provides an amazing library known as **SentimentIntensityAnalyzer** which provides us with the negative, positive, neutral and compound scores of the particular record.

According to the Industry Standards:

- If the compound score >= 0.05, then the text is categorized as Positive.
- If the compound score <= -0.05, then the text is categorized as Negative.
- If the compound score falls in the range (-0.05 - 0.05) then the text is categorized as Neutral.

Here, the compound score is the sum of positive, negative & neutral scores which is then normalized between -1 (most extreme negative) and +1 (most extreme positive).

Labelling the Unlabelled Data

In [13]:
```
sentiments = SentimentIntensityAnalyzer()
final_df["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in final_df["processed_comments"]]
final_df["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in final_df["processed_comments"]]
final_df["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in final_df["processed_comments"]]
final_df['Compound'] = [sentiments.polarity_scores(i)["compound"] for i in final_df["processed_comments"]]
final_df.head()
```

Out[13]:

	processed_comments	votes	Positive	Negative	Neutral	Compound
0	first benefit understand bullet train going be...	9	0.500	0.000	0.500	0.7184
1	progress improvement railway already catching ...	3	0.496	0.162	0.342	0.5994
2	bullet train start need make highway freeway g...	50	0.358	0.000	0.642	0.6908
3	type distributed time metro still remember opp...	26	0.000	0.000	1.000	0.0000
4	always way life repair create traditional foot...	0	0.130	0.000	0.870	0.2732

In [15]:
```
final_df.replace('Positive',1,inplace=True)
final_df.replace('Negative',-1,inplace=True)
final_df.replace('Neutral',0,inplace=True)
final_df
```

Out[15]:

	processed_comments	votes	Positive	Negative	Neutral	Compound	Sentiment
0	first benefit understand bullet train going be...	9	0.500	0.000	0.500	0.7184	1
1	progress improvement railway already catching ...	3	0.496	0.162	0.342	0.5994	1
2	bullet train start need make highway freeway g...	50	0.358	0.000	0.642	0.6908	1
3	type distributed time metro still remember opp...	26	0.000	0.000	1.000	0.0000	0
4	always way life repair create traditional foot...	0	0.130	0.000	0.870	0.2732	1
...	...	...	...	...	...	...	...
18634	know percent population poverty line want gove...	2	0.236	0.400	0.364	-0.5719	-1
18635	double line	0	0.000	0.000	1.000	0.0000	0
18636	project approve survey completion work start n...	3	0.000	0.000	1.000	0.0000	0
18637	state contribution direct central	0	0.000	0.000	1.000	0.0000	0
18638	stupidity limit project bring people poverty t...	2	0.099	0.325	0.576	-0.6499	-1

18639 rows × 7 columns

The insight which we could draw after labeling the data is that our data contains around 10k neutral values, around 6k values which are positively labeled and around 2.3k values which are negatively labeled.

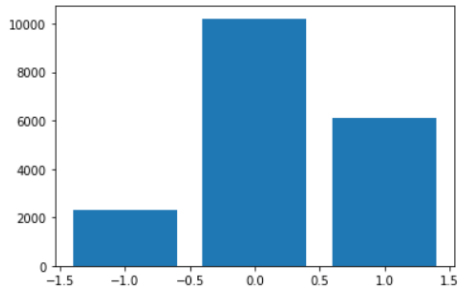
Insights after Labelling

```
In [16]: final_df.Sentiment.value_counts()
```

```
Out[16]:  0    10213
          1     6126
         -1     2300
         Name: Sentiment, dtype: int64
```

```
In [17]: plt.bar([0,1,-1],final_df.Sentiment.value_counts())
```

```
Out[17]: <BarContainer object of 3 artists>
```



Data Preprocessing

Now in order to make the machine understand the textual data so that it could be further used for the classification purpose we need to make use of Tf-Idf Vector which converts the textual data into the mathematical vectors.

Converting Text Data to Numeric using TfidfVectorizer

```
In [19]: from sklearn.feature_extraction.text import TfidfVectorizer
```

```
In [20]: final_df.processed_comments.values
```

```
Out[20]: array(['first benefit understand bullet train going benefit country ',
                'progress improvement railway already catching momentum stopped ',
                'bullet train start need make highway freeway good distributed throughout faster thus allowing economic growth ',
                ...,
                'project approve survey completion work start news paper news land purchase ',
                'state contribution direct central ',
                'stupidity limit project bring people poverty thousand people sleep without food stop eating feed '],
               dtype=object)
```

```
In [21]: vectorizer = TfidfVectorizer(max_features=5000)
         features = vectorizer.fit_transform(final_df.processed_comments.values)
         tdf=pd.DataFrame(features.toarray())
         tdf['Sentiment']=final_df['Sentiment']
         tdf
```

Out[21]:

	0	1	2	3	4	5	6	7	8	9	...	4991	4992	4993	4994	4995	4996	4997	4998	4999	Sentiment
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
18634	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	-1
18635	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0

Training

All the three classifiers were trained on the training datasets with a test train split of 80/20 percent. This enabled us to see the accuracy of the classifiers on the training datasets. The same random state was used between the classifiers to make sure that the training is reproducible between the classifiers.

Performing Train_Test_Split on our Converted Data

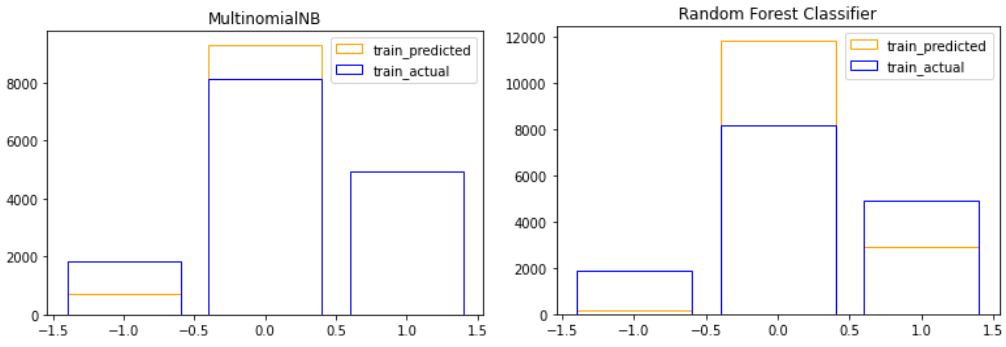
```
In [23]: from sklearn.model_selection import train_test_split
         train_set,test_set=train_test_split(tdf,test_size=0.2,random_state=42)
         x_train=train_set.drop(['Sentiment'],axis=1)
         y_train=train_set.Sentiment
         x_test=test_set.drop(['Sentiment'],axis=1)
         y_test=test_set.Sentiment
```

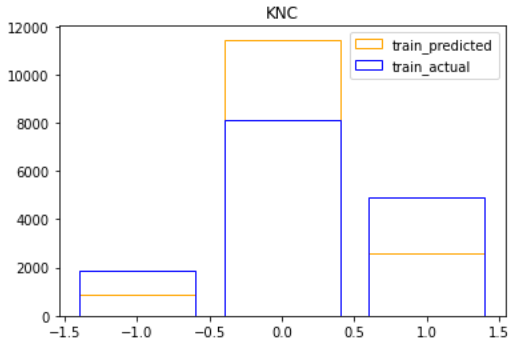
Classifiers

All the classifiers used were used with the standard parameters in Scikit-learn except for Random Forest Classifier where the max_features parameter was taken to be 500 and max_depth was setup to 10. This was done in order to prevent the model from overfitting and to reduce the computational cost.

Classifiers were selected based on what is suitable for text and social media sentiment analysis and what has been used in previous work in the field. Naive Bayes classifiers such as multinomial and complement naive Bayes are common for use in text classification due to being fast and simple to implement.

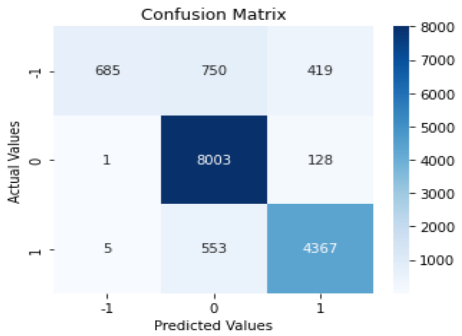
Comparative Analysis of all the three classifiers:





We can clearly notice from the above figures that the difference between the count of predicted values and actual values is minimum in case of Multinomial NB Classifier.
Thus from here we can estimate a rough idea that since count of predicted and actual values are almost same may be this model is perfectly fitted.
Still we are not sure as we haven't measured the accuracy yet.

Classification Reports

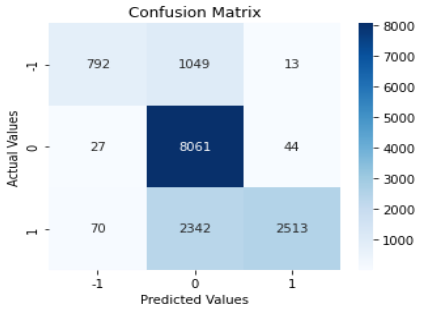



```
Training classification_report
              precision    recall  f1-score   support

          -1       0.99      0.37      0.54      1854
           0       0.86      0.98      0.92      8132
           1       0.89      0.89      0.89      4925

    accuracy                           0.88     14911
   macro avg       0.91      0.75      0.78     14911
weighted avg       0.89      0.88      0.86     14911
```

Classification Report of MultinomialNB

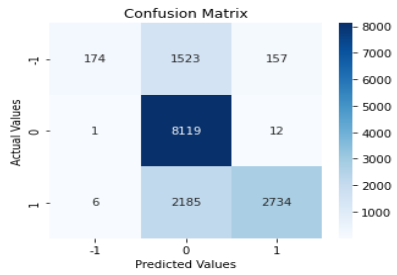



```
Training classification_report
              precision    recall  f1-score   support

          -1       0.89      0.43      0.58      1854
           0       0.70      0.99      0.82      8132
           1       0.98      0.51      0.67      4925

    accuracy                           0.76     14911
   macro avg       0.86      0.64      0.69     14911
weighted avg       0.82      0.76      0.74     14911
```

Classification Report of KNN




```
Training classification_report
              precision    recall  f1-score   support

          -1       0.96      0.09      0.17      1854
           0       0.69      1.00      0.81      8132
           1       0.94      0.56      0.70      4925

    accuracy                           0.74     14911
   macro avg       0.86      0.55      0.56     14911
weighted avg       0.80      0.74      0.70     14911
```

Classification Report of RFC

From the above Classification Reports it is very clear that Naive Bayes Algorithm has outperformed the other two algorithms in almost all the parameters be it in accuracy, recall, precision or f1_score.

Results:

Let us have a final accuracy check over both train and test dataset:

```
In [29]: print('train accuracy_score MNB',accuracy_score(pred,y_train))
         print('test accuracy_score MNB',accuracy_score(model.predict(x_test),y_test))

         train accuracy_score MNB 0.8755281335926497
         test accuracy_score MNB 0.8361051502145923
```

```
In [35]: print('train accuracy_score KNC',accuracy_score(pred,y_train))
         print('test accuracy_score KNC',accuracy_score(model.predict(x_test),y_test))

         train accuracy_score KNC 0.7622560525786333
         test accuracy_score KNC 0.7223712446351931
```

```
In [41]: print('train accuracy_score Random Forest Classifier',accuracy_score(pred,y_train))
         print('test accuracy_score Random Forest Classifier',accuracy_score(model.predict(x_test),y_test))

         train accuracy_score Random Forest Classifier 0.7395211588759976
         test accuracy_score Random Forest Classifier 0.744098712446352
```

It is clearly visible that the Multinomial NB has outperformed the rest of the two models with a train set accuracy of around 87% and test set accuracy of around 83%.

Conclusion

- In the study, while determining the labels for the dataset we realized that the sentiments of the people were ranging from neutral to mildly positive. There were very few comments (around 12%) which account for the negative sentiments regarding the High Speed Rails. On the contrary there was huge number of comments which were having the neutral sentiments (54%) followed by positive ones (34%).

- Results showed that Multinomial NB provides better results in comparison with KNN and RFC. Further, different categories of comments with more positive, neutral or negative tweets have been discussed. The study can be extended with gathering more data and applying deep learning technique to seek if there are any further improvements or not.

Future Work

- Accuracy of the predictions could possibly be improved by using more advanced classifiers such as neural networks or by tuning the Hyper Parameters using techniques like Grid Search CV and Randomized Search CV.
- Emojis are common in YouTube comments. It would be interesting to include both the text and emojis in the sentiment analysis since emojis can be indicative of sentiment in the comment.
- Additions of features like up vote on comments and likes and dislikes on the video can be another indicative for analyzing the sentiments of the comment.

Bibliography

- http://www.diva-portal.org/smash/get/diva2:1593439/FULLTEXT01.pdf
- file:///C:/Users/Hp/Downloads/RoughPowerSetTreeforFeatureSelection.pdf
- https://www.researchgate.net/figure/Classification-results-Accuracy-Mean-absolute-errorMAE-Root-mean-squared-error_tbl2_256390097
- https://scholar.google.co.in/scholar?hl=en&as_sdt=0,5&q=sentiment+analysis+high+speed+rail+India
- https://www.researchgate.net/profile/Vinodhini-G-2/publication/265163299_Sentiment_Analysis_and_Opinion_Mining_A_Survey/links/54018f330cf2bba34c1af133/Sentiment-Analysis-and-Opinion-Mining-A-Survey.pdf
- https://analyticsindiamag.com/sentiment-analysis-made-easy-using-vader/

Web Scrapping YouTube Links

https://youtu.be/9e_0SLlAX0Q
https://youtu.be/v0zipdSnI7g
https://youtu.be/k-sEsNVcBKE
https://youtu.be/LQ-xm-S7U18
https://youtu.be/ZA4qYgHXXWg
https://youtu.be/huGc2e9Sg6I
https://youtu.be/4Un0n0BXqZo
https://youtu.be/Nkj9-ITMnDQ
https://youtu.be/0kJLCAaMNlw
https://youtu.be/HCZQn4X19m4

https://youtu.be/hR17sHsLu50
https://youtu.be/CDBDSAfuxbQ
https://youtu.be/2Qdl6ZtLcds
https://youtu.be/KGPiGbOv5Vw
https://youtu.be/ThyElzggtII
https://youtu.be/S7xNwdKpW_Q
https://youtu.be/dMJoI0fJ5sU
https://youtu.be/r5R9zMEFBos
https://youtu.be/mppkoMK51f8
https://youtu.be/h4bzpaUuLvM
https://youtu.be/JdOVu9epmS8
https://youtu.be/DZIY1I_LjpY
https://youtu.be/bO-WSgIloeM
https://youtu.be/pjXVvW781Ng
https://youtu.be/929DhGoMqBI
https://youtu.be/11LRXsBswGk
https://youtu.be/_FmvDFTjMGM
https://youtu.be/ld2WFtWjuJw
https://youtu.be/gwlawHSnN3Q
https://youtu.be/gvDLFHJ3PDU
https://youtu.be/gvDLFHJ3PDU
https://youtu.be/KJcRvVGGnhQ
https://youtu.be/GzBR9SW0k4k
https://youtu.be/HBF5Q5ljzXM
https://youtu.be/mQXTicrUdQI

https://youtu.be/cI_IhRGZuks
https://youtu.be/hSdtjyf7188
https://youtu.be/1GE-o5DS6lI
https://youtu.be/HmQ9r7NQENU
https://youtu.be/fkkkVt_m59Q
https://youtu.be/_jxRiZsS0e0
https://youtu.be/HYgkXBIgY1A
https://youtu.be/1lmxGMYre9A
https://youtu.be/4BvKiRLrHyo

Appendix: 1

Source Code

```
import nltk
import pandas as pd
import numpy as np
import re #here re stands for regular expression library
import enchant #spell checker and word finder library
from wordcloud import WordCloud
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from nltk.sentiment.vader import SentimentIntensityAnalyzer
nltk.download("vader_lexicon")
df=pd.read_csv('ytb_comments.csv')
df.head()
df.drop(['cid','time','author','channel','photo','heart'],axis=1,inplace=True)
df.index=[i for i in range(len(df))]
df.index
for i in range(len(df)):
    if(str(df.loc[i]['votes']).find('2.5K')!=-1):
        print(i)
df.votes[11949]=2500
for i in range(len(df)):
    df.loc[[i],['votes']]=int(df.loc[i]['votes'])#converting votes to int
    df.loc[[i],['text']]=df.loc[i]['text'].lower()#converting text to lower case
    df.loc[[i],['text']]=re.sub('[^a-zA-Z 0-9]','',df.text[i])#removing all the characters
except a-z,A-Z,0-9 and space
df
#Word Tokenization and removing the words that are not the part of English Language

e = enchant.Dict("en_US")
a=[]
for i in range(len(df)):
    b=[]
    j=nltk.word_tokenize(df.text[i])
    for k in j:
        if(e.check(k)==True):
           b.append(k)
    a.append(b)
print(a)
# lemmatizing and stopwords removal

from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords

lemmatizer=WordNetLemmatizer()

d=[]
for i in a:
    b=''
    for j in range(len(i)):
        if(i[j] not in stopwords.words('english') and len(i[j])>3):
            b=b+lemmatizer.lemmatize(i[j])+' '
    d.append(b)
d
final_df=pd.concat([pd.DataFrame(d,columns=['processed_comments']),df['votes']],axis=1)
final_df_1=final_df.copy()
final_df
final_df.processed_comments.value_counts()
to_drop=final_df[final_df.processed_comments==''].index
final_df.drop(to_drop,axis=0,inplace=True)
final_df.index=[i for i in range(len(final_df))]
```

```
final_df
sentiments = SentimentIntensityAnalyzer()
final_df["Positive"] = [sentiments.polarity_scores(i)["pos"] for i in
final_df["processed_comments"]]
final_df["Negative"] = [sentiments.polarity_scores(i)["neg"] for i in
final_df["processed_comments"]]
final_df["Neutral"] = [sentiments.polarity_scores(i)["neu"] for i in
final_df["processed_comments"]]
final_df['Compound'] = [sentiments.polarity_scores(i)["compound"] for i in
final_df["processed_comments"]]
final_df.head()
score = final_df["Compound"].values
sentiment = []
for i in score:
    if i >= 0.05 :
        sentiment.append('Positive')
    elif i <= -0.05 :
        sentiment.append('Negative')
    else:
        sentiment.append('Neutral')
final_df["Sentiment"] = sentiment
final_df.head()
final_df.replace('Positive',1,inplace=True)
final_df.replace('Negative',-1,inplace=True)
final_df.replace('Neutral',0,inplace=True)
final_df
final_df.Sentiment.value_counts()
plt.bar([0,1,-1],final_df.Sentiment.value_counts())
wordcloud = WordCloud(width = 1200, height = 600,
                background_color ='white',
                min_font_size =
7,random_state=42).generate(str(final_df.processed_comments))

# plot the WordCloud image
plt.figure(figsize = (6,6), facecolor = None)
plt.imshow(wordcloud)
plt.axis("off")
plt.tight_layout(pad = 0)
plt.show()
from sklearn.feature_extraction.text import TfidfVectorizer
final_df.processed_comments.values
vectorizer = TfidfVectorizer(max_features=5000)
features = vectorizer.fit_transform(final_df.processed_comments.values)
tdf=pd.DataFrame(features.toarray())
tdf['Sentiment']=final_df['Sentiment']
tdf
features.shape
from sklearn.model_selection import train_test_split
train_set,test_set=train_test_split(tdf,test_size=0.2,random_state=42)
x_train=train_set.drop(['Sentiment'],axis=1)
y_train=train_set.Sentiment
x_test=test_set.drop(['Sentiment'],axis=1)
y_test=test_set.Sentiment
from sklearn.naive_bayes import MultinomialNB
model=MultinomialNB()
model.fit(x_train,y_train)
print('train_predicted')
pred=pd.DataFrame(model.predict(x_train))
print(pred.value_counts())
print("\n")
print('train_actual')
print(y_train.value_counts())
plt.bar([0,1,-1],pred.value_counts(),edgecolor='orange',fill=None,label='train_predicted')
plt.bar([0,1,-1],y_train.value_counts(),edgecolor='blue',fill=None,label='train_actual')
```

```
plt.title('MultinomialNB')
plt.legend()
plt.show()
cm=confusion_matrix(y_train,pred)
cm_df = pd.DataFrame(cm,index = [-1,0,1],columns = [-1,0,1])

#Plotting the confusion matrix
plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,cmap="Blues",fmt='g')
plt.title('Confusion Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
print('Training classification_report')
print(classification_report(y_train,pred))
print('train accuracy_score MNB',accuracy_score(pred,y_train))
print('test accuracy_score MNB',accuracy_score(model.predict(x_test),y_test))
from sklearn.neighbors import KNeighborsClassifier as KNC
model=KNC(n_neighbors=3)
model.fit(x_train,y_train)
print('train_predicted')
pred=pd.DataFrame(model.predict(x_train))
print(pred.value_counts())
print("\n")
print('train_actual')
print(y_train.value_counts())
plt.bar([0,1,-1],pred.value_counts(),edgecolor='orange',fill=None,label='train_predicted')
plt.bar([0,1,-1],y_train.value_counts(),edgecolor='blue',fill=None,label='train_actual')
plt.title('KNC')
plt.legend()
plt.show()
cm=confusion_matrix(y_train,pred)
cm_df = pd.DataFrame(cm,index = [-1,0,1],columns = [-1,0,1])

#Plotting the confusion matrix
plt.figure(figsize=(5,4))
sns.heatmap(cm_df, annot=True,cmap="Blues",fmt='g')
plt.title('Confusion Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
print('Training classification_report')
print(classification_report(y_train,pred))
print('train accuracy_score KNC',accuracy_score(pred,y_train))
print('test accuracy_score KNC',accuracy_score(model.predict(x_test),y_test))
from sklearn.ensemble import RandomForestClassifier as RFC
model=RFC(max_features=500,max_depth=10)
model.fit(x_train,y_train)
print('train_predicted')
pred=pd.DataFrame(model.predict(x_train))
print(pred.value_counts())
print("\n")
print('train_actual')
print(y_train.value_counts())
plt.bar([0,1,-1],pred.value_counts(),edgecolor='orange',fill=None,label='train_predicted')
plt.bar([0,1,-1],y_train.value_counts(),edgecolor='blue',fill=None,label='train_actual')
plt.title('Random Forest Classifier')
plt.legend()
plt.show()
cm=confusion_matrix(y_train,pred)
cm_df = pd.DataFrame(cm,index = [-1,0,1],columns = [-1,0,1])

#Plotting the confusion matrix
plt.figure(figsize=(5,4))
```

```
sns.heatmap(cm_df, annot=True,cmap="Blues",fmt='g')
plt.title('Confusion Matrix')
plt.ylabel('Actual Values')
plt.xlabel('Predicted Values')
plt.show()
print('Training classification_report')
print(classification_report(y_train,pred))
print('train accuracy_score Random Forest Classifier',accuracy_score(pred,y_train))
print('test accuracy_score Random Forest
Classifier',accuracy_score(model.predict(x_test),y_test))
```

Appendix: 2

Math behind the Classification Report

How to calculate FN, FP, TN, TP?

FN: The False-negative value for a class will be the sum of values of corresponding rows except for the TP value.
FP: The False-positive value for a class will be the sum of values of the corresponding column except for the TP value.
TN: The True Negative value for a class will be the sum of values of all columns and rows except the values of that class that we are calculating the values for.
TP: The True positive value is where the actual value and predicted value are the same.

Three class classification

Let us calculate the TP, TN, FP, FN values for the class -1 using the above tricks:

1. TP: The actual value and predicted value should be the same = (cell 1) = 127
2. FN: The sum of values of corresponding rows except the TP value = (cell 2 + cell3) =(186+133)
3. FP : The sum of values of corresponding column except the TP value = (cell 4 + cell 7) = (0 + 3) = 3
4. TN: The sum of values of all columns and row except the values of that class that we are calculating the values for = (cell 5 + cell 6 + cell 8 + cell 9) = 1983 + 98 + 191 + 1007 = 3279

Similarly, for 0 the values/ metrics are calculated as below:

1. TP : 1983 (cell 5)
2. FN : 0 + 98 = 98 (cell 4 +cell 6)
3. FP : 186 + 191 = 377 (cell 2 + cell 8)
4. TN : 127 + 133 + 3 + 1007 = 1270 (cell 1 + cell 3 + cell 7 + cell 9).

Similarly, for 1 the values/ metrics are calculated as below:

1. TP : 1007 (cell 9)
2. FN : 3 + 191 = 194 (cell 7 +cell 8)
3. FP : 133 + 98 = 231 (cell 3 + cell 6)
4. TN : 127 + 186 + 0 + 1983 = 2296 (cell 1 + cell 2 + cell 4 + cell 5).

Precision = TP/Predicted Yes
Recall = TP/Actual Yes

Github

To excess the Jupyter notebooks for source code and comment extraction notebook, Please follow the link:

https://github.com/sarthakkedia123/sentimental-analysis

Owing to Dr.Mudgal

I hereby thank Dr. Abhishek Mudgal Sir for giving me the opportunity to use my skills on such an innovative project. Your support and advice helped me a lot.
I promise to honor your belief on me by giving my absolute 100%.

Sarthak Kedia
19065109
sarthak.kedia.cd.civ19@itbhu.ac.in
Civil Engineering Part-III