In [81]:
```
import twint
import nest_asyncio
import asyncio
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
import re
from textblob import TextBlob
from transformers import AutoTokenizer, AutoModelForSequenceClassification
import torch
```

Classes 1 "Scraping Section"

In [82]:
```
#Class for Data Collection
class Tweets_Scraping:
    #Impelementation of the constructor
    def __init__(self,key_word="Tesla",lang="en",min_l=30,limit=100):
        self.conf=twint.Config()
        self.s=key_word
        self.lang=lang
        self.min=min_l
        self.lim=limit

    #For Collecting tweets and creating the DataFrame
    def scraping_tweets(self,key=None,lang=None,min_l=None,limit=None):
        if key is None:
            key=self.s

        if lang is None:
            lang=self.lang

        if min_l is None:
            min_l=self.min

        if limit is None:
            limit=self.lim
        nest_asyncio.apply()
        self.conf.Limit=limit
        self.conf.Lang = lang
        self.conf.Search =key
        self.conf.Hide_output=True
        self.conf.Min_likes =min_l
        self.conf.Pandas=True
        twint.run.Search(self.conf)
        df = twint.storage.panda.Tweets_df
        return df
```

Classes 2 "Pre-Processing Section"

In [83]:
```
#Class for Dealing with Data, preparing it and analysing it

class Data_preprocessing:
    #Function use for dealing with missing data
    def handle_missing_data(self,df,column,fill_with):
        df[column]=df[column].replace(np.nan,fill_with)

    # Function used to remove and clean tweets from special chracters
    def clean_tweets_content(self, tweet):
        return ' '.join(re.sub("(@[A-Za-z0-9]+)|([^0-9A-Za-z \t])|(\w+:\/\/\S+)|(RT)", " ", tweet).split())

    #This function is used to get the percentage of dataset column
    def get_col_percentage(self,col,df):

        total=df[col].value_counts()
        percentage=round(df[col].value_counts(dropna=False,normalize=True)*100,3)
        # or percentage=round((df[col]/df[col].sum())*100,2)
        res=pd.concat([total,percentage],axis=1,keys=["Total No.","Percentage"])
        #res['Percentage'] = res['Percentage'].astype(str) + '%'
        return res
```

Classes 3 "Sentiment Analysis Section"

In [85]:
```
class Sentiment_Analysis:

    def __init__(self,analyze_method):
        self.analyze_method=analyze_method

    #Function used for applying sentiment analysis based on the input method "Object"
    def sentiment_analysis_method(self,tweet_content):
        pre=Data_preprocessing()
        try:
            if self.analyze_method.lower() == "textblob":
                analysis = TextBlob(pre.clean_tweets_content(tweet_content))
                if analysis.sentiment.polarity > 0:
                    return 1
                elif analysis.sentiment.polarity == 0:
                    return 0
                else:
                    return -1

            elif self.analyze_method.lower() =="bert":
                tokenizer = AutoTokenizer.from_pretrained('nlptown/bert-base-multilingual-uncased-sentiment')
                model = AutoModelForSequenceClassification.from_pretrained('nlptown/bert-base-multilingual-uncase
                tokens = tokenizer.encode(pre.clean_tweets_content(tweet_content), return_tensors='pt')
                result = model(tokens)
                return int(torch.argmax(result.logits))+1

            else:
                return 7

        except Exception:
            print("Error!!!, Check Your Inputs Again, Please")
```

In [21]:
```
#t=Tweets_Scraping()
```

In [22]:
```
#df=t.scraping_tweets(key="Tesla",limit=1000,lang="en")
```

In [23]:
```
#df.language.value_counts()
```

In [24]:
```
#df.head()
```

In [12]:
```
#df.to_csv("h.csv")
```

In [68]:
```
df=pd.read_csv("h.csv")
```

In [69]:
```
#Exploring Data
df.head()
```

Out[69]:

	Unnamed: 0	id	conversation_id	created_at	date	timezone	place	tweet	language	hashtags	...	g
0	0	1510261577806721031	1510261577806721031	1.648909e+12	2022-04-02 16:23:14	200	NaN	The day I learned Tesla's had this feature	en	[]	...	N
1	1	1510259733223161858	1510259733223161858	1.648909e+12	2022-04-02 16:15:54	200	NaN	Ok….Newport to Austin - here we go!! #CyberRod...	en	['cyberrodeo', 'tesla']	...	N
2	2	1510258033879924739	1510258033879924739	1.648909e+12	2022-04-02 16:09:09	200	NaN	Epic Final 4, Tesla 1Q deliveries, then Master...	en	['tarheels']	...	N
3	3	1510251249408434179	1510251249408434179	1.648907e+12	2022-04-02 15:42:11	200	NaN	This @Tesla fandom stuff is getting out of con...	en	[]	...	N
4	4	1510249279247380482	1510249279247380482	1.648906e+12	2022-04-02 15:34:21	200	NaN	Amazing Drone Video Shows How Tesla Model Y Is...	en	[]	...	N

5 rows × 39 columns

In [70]:
```
df.columns
```

Out[70]:
```
Index(['Unnamed: 0', 'id', 'conversation_id', 'created_at', 'date', 'timezone',
       'place', 'tweet', 'language', 'hashtags', 'cashtags', 'user_id',
       'user_id_str', 'username', 'name', 'day', 'hour', 'link', 'urls',
       'photos', 'video', 'thumbnail', 'retweet', 'nlikes', 'nreplies',
       'nretweets', 'quote_url', 'search', 'near', 'geo', 'source',
       'user_rt_id', 'user_rt', 'retweet_id', 'reply_to', 'retweet_date',
       'translate', 'trans_src', 'trans_dest'],
      dtype='object')
```

In [71]:
```
#Dropping unuseful features
df.drop(columns=["id","place","Unnamed: 0"
                 ,"urls","photos","video",
                 "thumbnail","quote_url"
                  ,"conversation_id","created_at",
                  'geo', 'source', 'user_rt_id', 'user_rt',
                 'retweet_id', 'reply_to', 'retweet_date'
                  , 'translate', 'trans_src',
                 'trans_dest',"near"
                 ],axis=1,inplace=True, errors='ignore')
```

In [73]:
```
#Checking for Nulls in the remaining features
df.isna().sum()
```

Out[73]:
```
date            0
timezone        0
tweet           0
language        0
hashtags        0
cashtags        0
user_id         0
user_id_str     0
username        0
name            0
day             0
hour            0
link            0
retweet         0
nlikes          0
nreplies        0
nretweets       0
search          0
dtype: int64
```

In [74]:
```
d=Data_preprocessing()
```

In [75]:
```
#Cleaning "tweet" column using regex
df.tweet=df.tweet.apply(lambda x:d.clean_tweets_content(x))
```

In [87]:
```
senti=Sentiment_Analysis("TextBlob");
```

In [88]:
```
#sentiment Analysis process using "TextBlob"
df["sentiment_res"]=df.tweet.apply(lambda x:senti.sentiment_analysis_method(x))
```

In [91]:
```
d.get_col_percentage("sentiment_res",df)
```

Out[91]:

	Total No.	Percentage
1	510	51.0
0	317	31.7
-1	173	17.3

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js