1. """

A Program to analyse, Explore and Process Date so that prediction model can be created

1. """

## Step 1: Import all the required libraries and read the entire data sheet.

3. **import** numpy as np
4. **import** scipy as sp
5. **import** pandas as pd
6. **import** nltk
7. **import** math
8. **import** xlrd
9. **import** string
10. **from** sklearn.preprocessing **import** LabelEncoder
11. **import** random
12. **from** nltk.corpus **import** stopwords
13. **from** sklearn.feature\_extraction.text **import** TfidfVectorizer
14. **from** sklearn.ensemble **import** RandomForestClassifier
15. **from** sklearn.ensemble **import** RandomForestRegressor
16. **from** sklearn.ensemble **import** GradientBoostingClassifier
17. **from** sklearn.metrics **import** mean\_absolute\_error, mean\_squared\_error
18. **from** sklearn.model\_selection **import** train\_test\_split

21. >>> inputtweet=pd.read\_excel('tweets.xlsx')

## to get the a good perspective of the data we need to see the summary of the dataset and check how is the behaviour of the data.

## Check all the Columns of dataset

2. >>> inputtweet.column
4. Index(['TweetPostedTime', 'TweetID', 'TweetBody', 'TweetRetweetFlag',
5. 'TweetSource', 'TweetInReplyToStatusID', 'TweetInReplyToUserID',
6. 'TweetInReplyToScreenName', 'TweetRetweetCount', 'TweetFavoritesCount',
7. 'TweetHashtags', 'TweetPlaceID', 'TweetPlaceName', 'TweetPlaceFullName',
8. 'TweetCountry', 'TweetPlaceBoundingBox', 'TweetPlaceAttributes',
9. 'TweetPlaceContainedWithin', 'UserID', 'UserName', 'UserScreenName',
10. 'UserLocation', 'UserDescription', 'UserLink', 'UserExpandedLink',
11. 'UserFollowersCount', 'UserFriendsCount', 'UserListedCount',
12. 'UserSignupDate', 'UserTweetCount', 'MacroIterationNumber',
13. 'tweet.place'],
14. dtype='object')
16. ## Check first two records of teh dataset

19. >>> inputtweet.head(2)
21. >>> inputtweet.describe()
23. TweetID  TweetInReplyToStatusID  TweetInReplyToUserID
24. count  4.236800e+04            1.010000e+02          1.890000e+02
25. mean   8.110767e+17            8.107005e+17          6.489163e+16
26. std    4.408380e+13            2.581401e+15          2.140714e+17
27. min    8.110045e+17            7.860072e+17          3.389391e+06
28. 25%    8.110410e+17            8.110171e+17          7.271373e+07
29. 50%    8.110762e+17            8.110663e+17          3.664521e+08
30. 75%    8.111137e+17            8.111160e+17          2.615368e+09
31. max    8.111635e+17            8.111578e+17          8.016277e+17
33. TweetRetweetCount  TweetFavoritesCount  TweetPlaceAttributes
34. count       42368.000000         42368.000000                   0.0
35. mean         1081.095402             0.805561                   NaN
36. std          1432.473561            37.214059                   NaN
37. min             0.000000             0.000000                   NaN
38. 25%             0.000000             0.000000                   NaN
39. 50%            14.000000             0.000000                   NaN
40. 75%          3069.000000             0.000000                   NaN
41. max          4056.000000          2813.000000                   NaN
43. TweetPlaceContainedWithin        UserID  UserFollowersCount
44. count                        0.0  4.236800e+04        4.236800e+04
45. mean                         NaN  3.398540e+17        6.196995e+03
46. std                          NaN  3.799438e+17        5.620006e+04
47. min                          NaN  1.968000e+03        0.000000e+00
48. 25%                          NaN  5.862261e+08        4.000000e+01
49. 50%                          NaN  4.610624e+09        1.490000e+02
50. 75%                          NaN  7.662723e+17        1.819250e+03
51. max                          NaN  8.111532e+17        4.572706e+06
53. UserFriendsCount  UserListedCount  UserTweetCount  MacroIterationNumber
55. count      42368.000000     42368.000000    4.236800e+04          42368.000000
57. mean        3859.351846       312.092546    3.777249e+04            226.111688
59. std        19345.980461       850.623581    9.001777e+04            128.720271
61. min            0.000000         0.000000    1.000000e+00              0.000000
63. 25%           40.000000        17.000000    2.793500e+03            115.000000
65. 50%          304.000000        41.000000    1.989700e+04            227.000000
67. 75%         1780.000000       217.000000    3.327400e+04            337.000000
69. max       689302.000000     26577.000000    2.064424e+06            449.000000

72. ## identify the categorical and Numerical variables
74. >>> allcolumns= inputtweet.columns
75. >>> numericcol= inputtweet.\_get\_numeric\_data().columns
76. >>> numericcol
78. Index(['TweetID', 'TweetRetweetFlag', 'TweetInReplyToStatusID',
79. 'TweetInReplyToUserID', 'TweetRetweetCount', 'TweetFavoritesCount',
80. 'TweetPlaceAttributes', 'TweetPlaceContainedWithin', 'UserID',
81. 'UserFollowersCount', 'UserFriendsCount', 'UserListedCount',
82. 'UserTweetCount', 'MacroIterationNumber'],
83. dtype='object')
85. >>> categoricalcol= list(set(allcolumns)-set(numericcol))
86. >>> categoricalcol
88. ['TweetSource', 'TweetPostedTime', 'UserExpandedLink', 'TweetPlaceBoundingBx','TweetInReplyToScreenName', 'UserScreenName', 'TweetBody', 'UserSignupDate', 'Us
89. erName', 'TweetPlaceName', 'UserDescription', 'UserLink', 'TweetCountry', 'Tweet
90. PlaceID', 'UserLocation', 'TweetPlaceFullName', 'TweetHashtags', 'tweet.place']

## Now have divided our data into numerical and categorical columns, moving ahead we need to check the columns with null values.

2. >>> inputtweet.isnull().any()
4. TweetPostedTime              False
5. TweetID                      False
6. TweetBody                    False
7. TweetRetweetFlag             False
8. TweetSource                  False
9. TweetInReplyToStatusID        True
10. TweetInReplyToUserID          True
11. TweetInReplyToScreenName      True
12. TweetRetweetCount            False
13. TweetFavoritesCount          False
14. TweetHashtags                 True
15. TweetPlaceID                  True
16. TweetPlaceName                True
17. TweetPlaceFullName            True
18. TweetCountry                  True
19. TweetPlaceBoundingBox         True
20. TweetPlaceAttributes          True
21. TweetPlaceContainedWithin     True
22. UserID                       False
23. UserName                     False
24. UserScreenName               False
25. UserLocation                  True
26. UserDescription               True
27. UserLink                      True
28. UserExpandedLink              True
29. UserFollowersCount           False
30. UserFriendsCount             False
31. UserListedCount              False
32. UserSignupDate               False
33. UserTweetCount               False
34. MacroIterationNumber         False
35. tweet.place                   True
36. dtype: bool
38. ## This result tells us the coluns with missing values. If the output is true then 'missng values are there in that column' and false 'No missing values'.
39. ## As we can see here that dataset has 16 columns with missing values in it.
40. ## As the number of columns with missing values is hight we need to romove some of it which are not ceating any significance.
42. ## we will create a new variable which will give us the percentage values of null in every column and then
44. >>> colnullper=inputtweet.isnull().sum()/inputtweet.shape[0]
45. >>> colnullper
47. TweetPostedTime              0.000000
48. TweetID                      0.000000
49. TweetBody                    0.000000
50. TweetRetweetFlag             0.000000
51. TweetSource                  0.000000
52. TweetInReplyToStatusID       0.997616
53. TweetInReplyToUserID         0.995539
54. TweetInReplyToScreenName     0.995539
55. TweetRetweetCount            0.000000
56. TweetFavoritesCount          0.000000
57. TweetHashtags                0.002360
58. TweetPlaceID                 0.976397
59. TweetPlaceName               0.976397
60. TweetPlaceFullName           0.976397
61. TweetCountry                 0.976421
62. TweetPlaceBoundingBox        0.976397
63. TweetPlaceAttributes         1.000000
64. TweetPlaceContainedWithin    1.000000
65. UserID                       0.000000
66. UserName                     0.000000
67. UserScreenName               0.000000
68. UserLocation                 0.378257
69. UserDescription              0.103002
70. UserLink                     0.608218
71. UserExpandedLink             0.609092
72. UserFollowersCount           0.000000
73. UserFriendsCount             0.000000
74. UserListedCount              0.000000
75. UserSignupDate               0.000000
76. UserTweetCount               0.000000
77. MacroIterationNumber         0.000000
78. tweet.place                  0.976397
79. dtype: float64

## We will have to assume that null percentage with more than 60% is not going to be in the calculation as they

##don’t make any significant changes in the dataset.

## tweeting and retweeting can have a pattern if hashtags have been analysed. So seeingthe hastag percentage it is 24%, still manageable to conduct the analysis.

# We will remove these columns.

2. >>> colnullperc=pd.DataFrame(colnullper).reset\_index() ## Converting Series to Dataframe
3. >>> colnullperc.columns=['ColName','Value']                ## giving dataframe column names
4. >>> dropcol=colnullperc.loc[colnullperc['Value']>0.8000]
5. >>> dropcolfinal=dropcol['ColName']  ## created a final list of columns to be dropped

8. >>> inputtweet\_mo= inputtweet.drop(dropcolfinal, axis=1).dropna() ## we have created new datafrane without all teh columns where null values percentage> 80%

## Now wherever we have missing values in the dataset we will imput the mean value for Numerical variable and for the categorical variables we will impute the value with -9999

1. >>> numericcol\_mo= list(set(numericcol)-set(dropcol['ColName']))  ## Removed the dropeed column
3. >>> inputtweet\_mo[numericcol\_mo] = inputtweet\_mo[numericcol\_mo].fillna(inputtweet\_mo[numericcol\_mo].mean(),inplace=True)
5. >>> Categorical\_mo= list(set(categoricalcol)-set(dropcol['ColName']))  ## Removed the dropeed column
7. >>> inputtweet\_mo[Categorical\_mo] = inputtweet\_mo[Categorical\_mo].fillna(value = -9999)

## Taking a look at the datat it can be seen that these tweets are of teh same date so the analysis should be at time level. we will sepearte date and time and append new columns

2. >>> inputtweet\_mo.TweetPostedTime= pd.to\_datetime(inputtweet\_mo.TweetPostedTime)
4. >>> inputtweet\_mo.UserSignupDate= pd.to\_datetime(inputtweet\_mo.UserSignupDate)
6. inputtweet\_mo['thour'] = inputtweet\_mo.TweetPostedTime.apply(**lambda** x: x.hour)
7. inputtweet\_mo['tminute'] = inputtweet\_mo.TweetPostedTime.apply(**lambda** x: x.minute)
8. inputtweet\_mo['tseconds'] = inputtweet\_mo.TweetPostedTime.apply(**lambda** x: x.second)

## For a single day we dont need posteddate now, so we will remove it. Also Username and userscreen is redundant, as we are doing the analysis for the retweet so

## username does not make much sense. We still have userid with us.

## Tweethashtag is very important as to check the count of tweet for a particular tag. We can have a segmentation to hastags

3. >>> remcol= ['UserName', 'TweetSource', 'UserScreenName','TweetPostedTime']
4. >>> inputtweet\_mo.drop(remcol, axis=1, inplace=True)           ## Remove the columns which are
6. ## Calculate the number of unique Tweets, check all teh tweets with TweetBody not having 'RT' at start
8. >>> nooforiginaltweet=[element **for** element **in** inputtweet\_mo['TweetBody'].values
9. **if** **not** element.startswith('RT')]
10. >>>  len(ooforiginaltweet)
12. ## calculate Number of Retweets
14. >>> noorretweet=[element **for** element **in** inputtweet\_mo['TweetBody'].values **if** ele
15. ment.startswith('RT')]
16. >>> len(noorretweet)

## Now looking at the body of the tweet. We need to normalize  this column.

## To do that first step is to Remove the "RT" from the start whereever its occurring.

## After Removal of RT we need to clean String.

2. **def** remove\_punctuation(s):
3. s = ''.join([i **for** i **in** s **if** i **not** **in** frozenset(string.punctuation)])
4. **return** s

7. **def** remove\_stopwords(s):
8. s = ''.join([i **for** i **in** s **if** i **not** **in** stopwords.words('english')])
9. **return** s
11. **def** remove\_rt(s):
12. s = ''.join([i **for** i **in** s **if** i **not** **in** 'RT'])
13. **return** s
15. ## We have created the Method to remove punctuations and also Stopwords.
17. >>> inputtweet\_mo['TweetBody']=inputtweet\_mo['TweetBody'].apply(remove\_punctuati
18. on).apply(remove\_rt).apply(remove\_stopwords)
20. >>> inputtweet\_mo['TweetHashtags']=inputtweet\_mo['TweetHashtags'].astype(str).ap
21. ply(remove\_punctuation).apply(remove\_rt)
23. >>> inputtweet\_mo['UserLocation']=inputtweet\_mo['UserLocation'].astype(str).ap
24. ply(remove\_punctuation).apply(remove\_rt)

## As all these variables are in text format. To check some simlilarity we need to create a vector form of it.

## we can use tfidf (term frequency- inverse dicument frequency) vectorization for the creation of the vectors for TweetBody, TweetHashtags and UserLocation

1. >>> v = TfidfVectorizer()
2. >>> x2 = v.transform(inputtweet['TweetBody'].values.astype('U'))
3. >>> inputtweet\_mo['TweetBody1']=list(x2.toarray())
5. >>> y2 = v.transform(inputtweet\_mo['TweetHashtags'].values.astype('U'))
6. >>> inputtweet\_mo['TweetHashtags2']=list(y2.toarray())
8. >>> z2 = v.transform(inputtweet\_mo['UserLocation'].values.astype('U'))
9. >>> inputtweet\_mo['UserLocation1']=list(y2.toarray())

## We will remove the original columns as we dont need them now, as we have created the vectorized form of those columns.

2. >>> rcol=['TweetBody','TweetHashtags','UserLocation']
3. >>> inputtweet\_mo.drop(rcol, axis=1, inplace=True)

## all the other categorical variables except UserLocation1,TweetHashtags1,TweetBody, We will create label encoders for these variables

2. Categorical\_mo = Categorical\_mo-['TweetBody','TweetHashtags','UserLocation']
4. >>> **for** var **in** Categorical\_mo
5. >>> number = LabelEncoder()
6. >>> Inputtweet\_mo[var] = number.fit\_transform(Inputtweet\_mo[var].astype('str'))

## Deciding the features for the model, these features will be used to predict the modeeling.

2. >>> features= list(set(inputtweet\_mo)-set(['TweetRetweetCount']))
4. ## dividing teh data into Test and Train
6. >>> X=inputtweet\_mo[features].values
7. >>> Y=inputtweet\_mo['TweetRetweetCount'].values
9. >>> X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.30, random\_state=350)

## No we are going to fit the data in to a model. I am going to use Random forest modelling approach for this.

1. >>> rf = RandomForestClassifier(n\_estimators=1000)
2. >>> rf.fit(X\_train,Y\_train)
3. RandomForestClassifier(bootstrap=True, class\_weight=None, criterion='gini',
4. max\_depth=None, max\_features='auto', max\_leaf\_nodes=None,
5. min\_impurity\_split=1e-07, min\_samples\_leaf=1,
6. min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,
7. n\_estimators=1000, n\_jobs=1, oob\_score=False,

## Now we have fitted the model into classifier, we will fit the same into repressor also.

1. random\_state=None, verbose=0, warm\_start=False)
2. >>> rfg = RandomForestRegressor(n\_estimators=1000)
3. >>> rfg.fit(X\_train,Y\_train)
4. RandomForestRegressor(bootstrap=True, criterion='mse', max\_depth=None,
5. max\_features='auto', max\_leaf\_nodes=None,
6. min\_impurity\_split=1e-07, min\_samples\_leaf=1,
7. min\_samples\_split=2, min\_weight\_fraction\_leaf=0.0,
8. n\_estimators=1000, n\_jobs=1, oob\_score=False, random\_state=None,
9. verbose=0, warm\_start=False)