In [1]:
```
#This model is building a logistic regression model to predict if the content
 was shared or not.
#Installing necessary libraries


import pandas as pd
import numpy as np
import seaborn as sns
%matplotlib inline
```

In [2]:
```
pwd
```

Out[2]:
```
'C:\\Users\\rehnu'
```

In [3]:
```
cd C:\Users\rehnu\OneDrive\Desktop\Python!
```

```
C:\Users\rehnu\OneDrive\Desktop\Python!
```

In [5]:
```
#Loading the dataset

Popularity = pd.read_csv("NewsPopularity_Logistics_BB2.csv")
```

In [6]:
```
#Showing first few rows of the dataset

Popularity.head()
```

Out[6]:

	length_title	length_content	images	videos	keywords	onWeekend	Relevancy	positive_word
0	12	219	1	0	5	No	0.500331	0.76923
1	9	255	1	0	4	No	0.799756	0.73333
2	9	211	1	0	6	No	0.217792	0.85714
3	9	531	1	0	7	No	0.028573	0.66666
4	13	1072	20	0	7	No	0.028633	0.86021

In [7]:
```
#Showing last few rows of the dataset

Popularity.tail()
```

Out[7]:

	length_title	length_content	images	videos	keywords	onWeekend	Relevancy	positive_
39639	11	346	1	1	8	No	0.025038	0.7
39640	12	328	3	48	7	No	0.029349	0.8
39641	10	442	12	1	8	No	0.159004	0.5
39642	6	682	1	0	5	No	0.040004	0.4
39643	10	157	0	2	4	No	0.050001	0.8

In [8]:
```
#Observing column type of the dataset

Popularity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39644 entries, 0 to 39643
Data columns (total 12 columns):
length_title            39644 non-null int64
 length_content         39644 non-null int64
images                  39644 non-null int64
videos                  39644 non-null int64
 keywords               39644 non-null int64
onWeekend               39644 non-null object
Relevancy               39644 non-null float64
positive_words          39644 non-null float64
negative_words          39644 non-null float64
 title_subjectivity     39644 non-null float64
 title_sentiments       39644 non-null float64
wasShared               39644 non-null object
dtypes: float64(5), int64(5), object(2)
memory usage: 3.6+ MB
```

In [9]:
```
#Showing number of rows and columns of the dataset

Popularity.shape
```

Out[9]: (39644, 12)

In [10]:
```
#Observing if there is any missing data on the file. No missing value found.

Popularity.isnull().sum()
```

Out[10]:
```
length_title           0
 length_content        0
images                 0
videos                 0
 keywords              0
onWeekend              0
Relevancy              0
positive_words         0
negative_words         0
 title_subjectivity    0
 title_sentiments      0
wasShared              0
dtype: int64
```

In [11]:
```
#There were two categorical columns ('onWeekend' and 'wasShared'). Making them
binary.


Popularity['onWeekend'] = Popularity['onWeekend'].map({'Yes' : 1, 'No' : 0})
Popularity['wasShared'] = Popularity['wasShared'].map({'Yes' : 1, 'NO' : 0})
```

In [12]:
```
#Again showing the first few rows of the file to see if those categorical colu
mns are now integer or not.

Popularity.head()
```

Out[12]:

	length_title	length_content	images	videos	keywords	onWeekend	Relevancy	positive_word
0	12	219	1	0	5	0	0.500331	0.76923
1	9	255	1	0	4	0	0.799756	0.73333
2	9	211	1	0	6	0	0.217792	0.85714
3	9	531	1	0	7	0	0.028573	0.66666
4	13	1072	20	0	7	0	0.028633	0.86021

In [13]:
```
#Observing the column type of the data file.

Popularity.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 39644 entries, 0 to 39643
Data columns (total 12 columns):
length_title           39644 non-null int64
 length_content        39644 non-null int64
images                 39644 non-null int64
videos                 39644 non-null int64
 keywords              39644 non-null int64
onWeekend              39644 non-null int64
Relevancy              39644 non-null float64
positive_words         39644 non-null float64
negative_words         39644 non-null float64
 title_subjectivity    39644 non-null float64
 title_sentiments      39644 non-null float64
wasShared              39644 non-null int64
dtypes: float64(5), int64(7)
memory usage: 3.6 MB
```

In [14]:
```
#showing the percentage of time the content was shared.

wasShared = (sum(Popularity['wasShared'])/len(Popularity['wasShared'].index))*
100
```

In [15]:
```
wasShared
```

Out[15]:
```
68.66108364443548
```

In [16]:
```
#Importing sklearn library and train_test_split for spliting the data file int
o test and train

from sklearn.model_selection import train_test_split
```

In [17]:
```
#Declaring depending and independing variables

x = Popularity.drop(['wasShared'], axis = 1)

y = Popularity['wasShared']
```

In [18]:
```
#Showing all independent variables

x.head()
```

Out[18]:

	length_title	length_content	images	videos	keywords	onWeekend	Relevancy	positive_word
0	12	219	1	0	5	0	0.500331	0.76923
1	9	255	1	0	4	0	0.799756	0.73333
2	9	211	1	0	6	0	0.217792	0.85714
3	9	531	1	0	7	0	0.028573	0.66666
4	13	1072	20	0	7	0	0.028633	0.86021

In [19]:
```
#Showing dependent variable

y.head()
```

Out[19]:
```
0    0
1    0
2    1
3    1
4    0
Name: wasShared, dtype: int64
```

In [20]:
```
#Spliting dependent and independent variables into 70% and 30% train and test
 categories.

x_train, x_test, y_train, y_test = train_test_split(x,y, train_size = 0.7, tes
t_size = 0.3, random_state = 100)
```

In [21]:
```
import statsmodels.api as sm
```

In [22]:
```
#Buidling and Showing Linear Regression model

logml = sm.GLM(y_train,(sm.add_constant(x_train)), family = sm.families.Binomial())
logml.fit().summary()
```

```
C:\Users\rehnu\Anaconda3\lib\site-packages\numpy\core\fromnumeric.py:2389: FutureWarning: Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.
  return ptp(axis=axis, out=out, **kwargs)
```

Out[22]:

Generalized Linear Model Regression Results

Dep. Variable:	wasShared	**No. Observations:**	27750
Model:	GLM	**Df Residuals:**	27738
Model Family:	Binomial	**Df Model:**	11
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-16653.
Date:	Sun, 24 May 2020	**Deviance:**	33306.
Time:	01:34:52	**Pearson chi2:**	2.77e+04
No. Iterations:	5		
Covariance Type:	nonrobust		

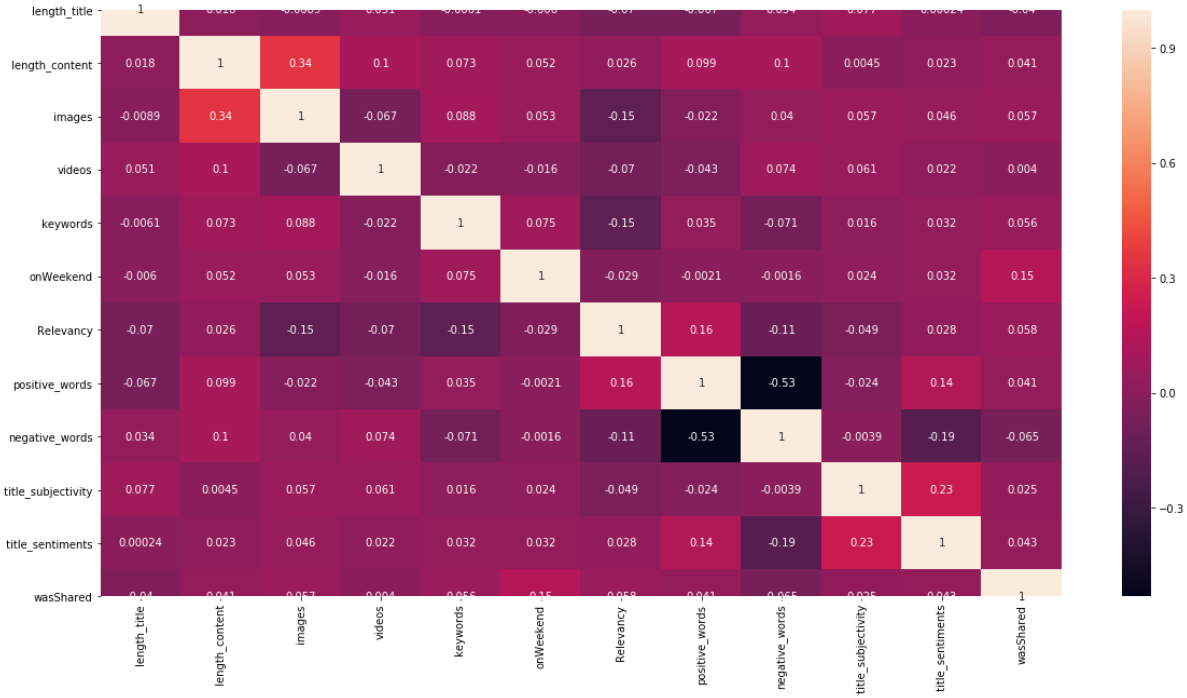
	coef	std err	z	P>|z|	[0.025	0.975]
const	0.6499	0.118	5.492	0.000	0.418	0.882
length_title	-0.0395	0.006	-6.264	0.000	-0.052	-0.027
length_content	8.163e-05	3.26e-05	2.508	0.012	1.78e-05	0.000
images	0.0157	0.002	8.097	0.000	0.012	0.019
videos	0.0100	0.003	2.929	0.003	0.003	0.017
keywords	0.0594	0.007	8.356	0.000	0.045	0.073
onWeekend	1.1737	0.051	23.146	0.000	1.074	1.273
Relevancy	0.6285	0.055	11.408	0.000	0.521	0.737
positive_words	-0.0497	0.087	-0.574	0.566	-0.219	0.120
negative_words	-0.8252	0.104	-7.937	0.000	-1.029	-0.621
title_subjectivity	0.1023	0.042	2.423	0.015	0.020	0.185
title_sentiments	0.1819	0.053	3.443	0.001	0.078	0.285

In [23]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

In [24]:
```
#Creating a Heatmap of the data file.

plt.figure(figsize = (20,10))
sns.heatmap(Popularity.corr(), annot= True)
```

Out[24]: <matplotlib.axes._subplots.AxesSubplot at 0x2952660b108>



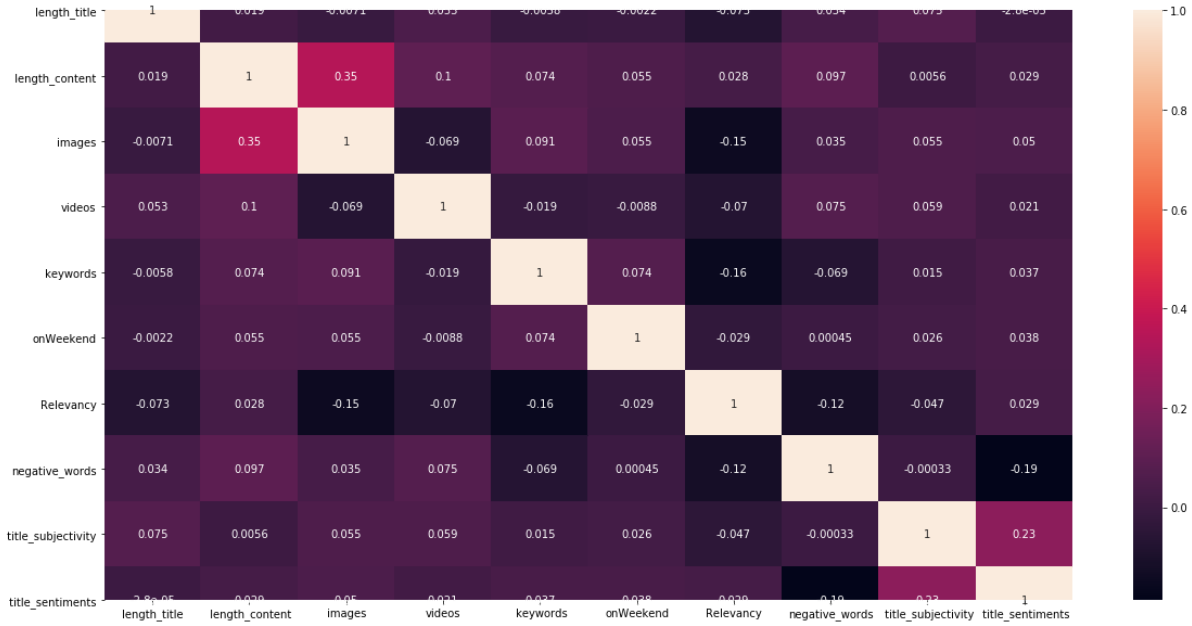
In [25]:
```
#Removing 'positive_words' as it has high p-value and less impact on a content being shared.

x_test2 = x_test.drop(['positive_words'],axis =1)
x_train2 = x_train.drop(['positive_words'], axis = 1)
```

In [26]:
```
#After deleting 'Positive_words' creating the Heatmap again.

plt.figure(figsize = (20,10))
sns.heatmap(x_train2.corr(), annot= True)
```

Out[26]: <matplotlib.axes._subplots.AxesSubplot at 0x29527006808>



In [27]:
```
#After deleting 'Positive_words' building and showing the result of Linear Regression model.

logm2 = sm.GLM(y_train,(sm.add_constant(x_train2)), family = sm.families.Binomial())
logm2.fit().summary()
```

Out[27]:

Generalized Linear Model Regression Results

Dep. Variable:	wasShared	**No. Observations:**	27750
Model:	GLM	**Df Residuals:**	27739
Model Family:	Binomial	**Df Model:**	10
Link Function:	logit	**Scale:**	1.0000
Method:	IRLS	**Log-Likelihood:**	-16653.
Date:	Sun, 24 May 2020	**Deviance:**	33306.
Time:	01:36:16	**Pearson chi2:**	2.77e+04
No. Iterations:	5		
Covariance Type:	nonrobust		

	coef	std err	z	P>|z|	[0.025	0.975]
const	0.6067	0.091	6.648	0.000	0.428	0.786
length_title	-0.0393	0.006	-6.244	0.000	-0.052	-0.027
length_content	7.817e-05	3.2e-05	2.446	0.014	1.55e-05	0.000
images	0.0157	0.002	8.132	0.000	0.012	0.020
videos	0.0100	0.003	2.942	0.003	0.003	0.017
keywords	0.0594	0.007	8.357	0.000	0.045	0.073
onWeekend	1.1738	0.051	23.151	0.000	1.074	1.273
Relevancy	0.6256	0.055	11.404	0.000	0.518	0.733
negative_words	-0.7926	0.087	-9.123	0.000	-0.963	-0.622
title_subjectivity	0.1032	0.042	2.445	0.015	0.020	0.186
title_sentiments	0.1803	0.053	3.418	0.001	0.077	0.284

In [28]:
```
#Building Logistic Regression model with train dataset of dependent and indepe
ndent variables.

from sklearn.linear_model import LogisticRegression
from sklearn import metrics
logsk = LogisticRegression()
logsk.fit(x_train,y_train)
```

```
C:\Users\rehnu\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:4
32: FutureWarning: Default solver will be changed to 'lbfgs' in 0.22. Specify
a solver to silence this warning.
  FutureWarning)
```

Out[28]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [29]:
```
logsk
```

Out[29]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='warn', n_jobs=None, penalty='l2',
                   random_state=None, solver='warn', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [30]:
```
#Creating a vaiable to predict values for the test data set of independent var
iables.

y_pred = logsk.predict_proba(x_test)
```

In [32]:
```
#Declaring a dataframe of the predicted variable data

y_pred_df = pd.DataFrame(y_pred)
```

In [33]:
```
#Showing only the shared content column

y_pred_1 = y_pred_df.iloc[:,[1]]
```

In [34]:
```
y_pred_1.head()
```

Out[34]:

	1
0	0.737322
1	0.710362
2	0.768704
3	0.763017
4	0.600920

In [35]:
```
#Another dataframe with the dependent varible's test dataset.

y_test_df = pd.DataFrame(y_test)
```

In [36]:
```
#Joining two dataframes together and removing their index number column so tha
t they appear side by side.

y_pred_1.reset_index(drop= True, inplace = True)
y_test_df.reset_index(drop= True, inplace = True)
y_pred_1 = y_pred_1.rename(columns= {1 : 'WasShared_Predicted'})
y_pred_final = pd.concat([y_test_df, y_pred_1],axis = 1)
```

In [37]:
```
#Showing dependent variable's actual and predicted results.

y_pred_final.head(10)
```

Out[37]:

	wasShared	WasShared_Predicted
0	1	0.737322
1	0	0.710362
2	1	0.768704
3	0	0.763017
4	1	0.600920
5	1	0.568105
6	0	0.681206
7	1	0.713121
8	0	0.756291
9	1	0.758318

In [38]:
```
#Creating a new column with a condition to be matched with actual result.

y_pred_final['Predicted'] = y_pred_final.WasShared_Predicted.map(lambda x: 1 i
f x >0.5 else 0 )
```

In [39]:
```
y_pred_final.head(10)
```

Out[39]:

	wasShared	WasShared_Predicted	Predicted
0	1	0.737322	1
1	0	0.710362	1
2	1	0.768704	1
3	0	0.763017	1
4	1	0.600920	1
5	1	0.568105	1
6	0	0.681206	1
7	1	0.713121	1
8	0	0.756291	1
9	1	0.758318	1

In [40]:
```
from sklearn import metrics
```

In [41]:
```
#Creating confusion matrix.

cm = metrics.confusion_matrix(y_pred_final.wasShared, y_pred_final.Predicted)
```

In [42]:
```
cm
```

Out[42]:
```
array([[  31, 3704],
       [  28, 8131]], dtype=int64)
```

In [43]:
```
#showing the overall accuracy of the model.

metrics.accuracy_score(y_pred_final.wasShared, y_pred_final.Predicted)
```

Out[43]:
```
0.6862283504287876
```

In [44]:
```
#A model was built using Logistic Regression to predict if the content was sha
red or not.
#All the independent variables were affecting if the content will be shared or
not except 'Positive_words' column as it's
#p-value was higher.
#From the logistic regression model I can say that the number of words in the
 content (length_content), the number of images in the content (images), the n
umber of videos in the content (videos), the number of keywords (keywords), if
the content was read on a weekend (onWeekend), a measure of the relevancy of t
he content (relevancy), a measure of subjectivity in the title (title_subjecti
vity), a measure of sentiments in the title (title_sentiments) are the factors
that can determine whether a new content will be shared or not.
#For one-unit change in these factors(the log odds ratio of wasShared (measure
if the content was shared) will be increased.
#The Accuracy is 0.686(68.6%) which means the model is capable of classifying
 instances correctly in 68.6% cases.
```

In [ ]:

In [ ]:

In [ ]: