Instagram Post Analysis for The Impact Engine - 2020

Importing all the packages needed for the analysis

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as py
import seaborn as sns
from scipy.stats import pearsonr
from warnings import filterwarnings
```

Loading datasets in variables

In [2]:

```
data_corr = pd.read_csv("../Dataset_csv/correlation_matrix1.csv")
data_clean = pd.read_csv("../Dataset_csv/instagram_cleandata_main.csv")
data_clean_blanks = pd.read_csv("../Dataset_csv/instagram_cleandata_withblanks.csv")
data_timings = pd.read_csv("../Dataset_csv/User_active_timings.csv")
```

1. Correlation matrix

Displaying contents of csv file

In [3]:

```
data_corr.head()
```

Out[3]:

	Dates	Post no.	Type of post	Likes	Comments	Shares	Save	Profile visits - Int	Website clicks - Int	Emails - Int	...	Total - Int	Follows	Reach	New Reach	New reach %
0	18-06-2020	1	Carousel	368	37	444	105	871	1	6	...	878	124	3284	2988	91%
1	19-06-2020	2	Carousel	188	9	52	29	114	0	0	...	114	16	1612	1145	71%
2	20-06-2020	3	Carousel	163	13	26	29	84	0	0	...	84	5	1212	764	63%
3	21-06-2020	4	Carousel	188	68	112	85	123	0	0	...	124	7	1760	1285	73%
4	22-06-2020	5	Carousel	290	38	288	73	719	0	3	...	722	92	2511	2009	80%

5 rows × 21 columns

In [4]:

```
data_corr.describe()
```

Out[4]:

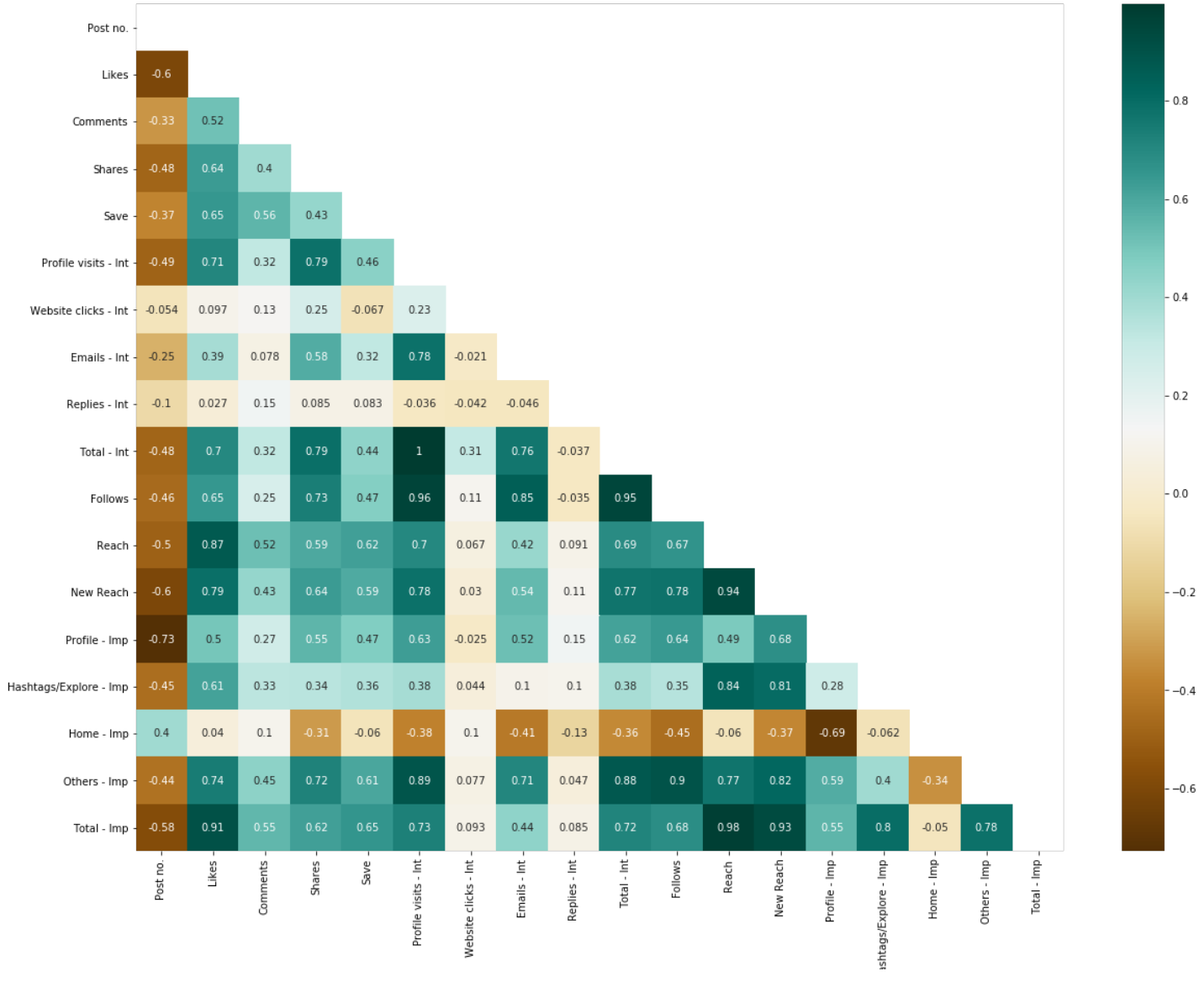
	Post no.	Likes	Comments	Shares	Save	Profile visits - Int	Website clicks - Int	Emails - Int	Replies - Int	Total - I
count	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.0000
mean	56.000000	179.954955	21.117117	66.333333	31.756757	87.531532	4.009009	0.171171	0.036036	91.7477
std	32.186954	63.514979	17.795167	83.047996	25.713179	132.259718	11.631849	0.724555	0.187225	135.9930
min	1.000000	75.000000	0.000000	0.000000	6.000000	2.000000	0.000000	0.000000	0.000000	2.0000
25%	28.500000	137.500000	8.500000	14.000000	17.000000	25.500000	0.000000	0.000000	0.000000	27.0000
50%	56.000000	169.000000	16.000000	43.000000	26.000000	50.000000	1.000000	0.000000	0.000000	53.0000
75%	83.500000	202.500000	28.500000	80.500000	36.000000	84.500000	2.500000	0.000000	0.000000	89.0000
max	111.000000	472.000000	101.000000	444.000000	195.000000	871.000000	92.000000	6.000000	1.000000	878.0000

Visualization

In [5]:

```
fig, ax = py.subplots(figsize=(20, 15))
mask = np.triu(np.ones_like(data_corr.corr(), dtype=bool))
sns.heatmap(data_corr.corr(), cmap="BrBG", annot=True, mask=mask, cbar_kws = {'shrink': 1
})

py.show()
```



2. Reach with respect to Time

Displaying contents of csv file

In [6]:

```
data_clean_blanks.head()
```

Out[6]:

	Sr no.	Date	Dates	Post no.	Type of post	Likes	Comments	Shares	Save	Profile visits	...	Total	Follows	Reach	New Reach	New reach %	Profile	H
0	1	1	01-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
1	2	2	02-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
2	3	3	03-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	4	4	04-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
4	5	5	05-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	

5 rows × 23 columns

In [7]:

```
data_clean_blanks.describe()
```

Out[7]:

	Sr no.	Date	Post no.	Likes	Comments	Shares	Save	Profile visits	Website clicks	Ema
count	250.000000	250.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.0000
mean	125.500000	15.560000	56.000000	179.954955	21.117117	66.333333	31.756757	87.531532	4.009009	0.1711
std	72.312977	8.955849	32.186954	63.514979	17.795167	83.047996	25.713179	132.259718	11.631849	0.7245
min	1.000000	1.000000	1.000000	75.000000	0.000000	0.000000	6.000000	2.000000	0.000000	0.0000
25%	63.250000	8.000000	28.500000	137.500000	8.500000	14.000000	17.000000	25.500000	0.000000	0.0000
50%	125.500000	15.500000	56.000000	169.000000	16.000000	43.000000	26.000000	50.000000	1.000000	0.0000
75%	187.750000	23.000000	83.500000	202.500000	28.500000	80.500000	36.000000	84.500000	2.500000	0.0000
max	250.000000	31.000000	111.000000	472.000000	101.000000	444.000000	195.000000	871.000000	92.000000	6.0000

8 rows × 21 columns

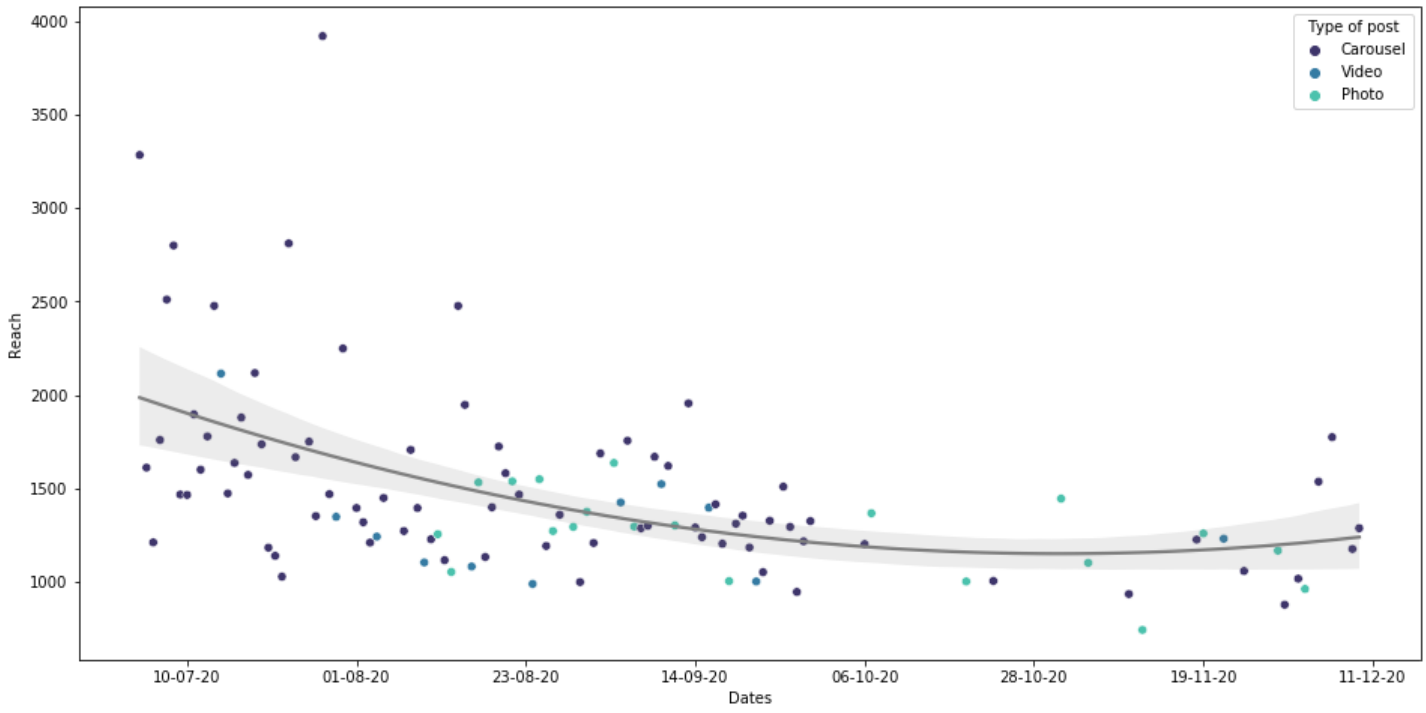
Visualization

In [8]:

```
x_axis_labels = ['18-06-20','19-06-20','20-06-20','21-06-20','22-06-20','23-06-20','24-06
```

```
-20','25-06-20','26-06-20','27-06-20','28-06-20','29-06-20','30-06-20','01-07-20','02-07-
0','03-07-20','04-07-20','05-07-20','06-07-20','07-07-20','08-07-20','09-07-20','10-07-20
,'11-07-20','12-07-20','13-07-20','14-07-20','15-07-20','16-07-20','17-07-20','18-07-20',
19-07-20','20-07-20','21-07-20','22-07-20','23-07-20','24-07-20','25-07-20','26-07-20','2
-07-20','28-07-20','29-07-20','30-07-20','31-07-20','01-08-20','02-08-20','03-08-20','04-
8-20','05-08-20','06-08-20','07-08-20','08-08-20','09-08-20','10-08-20','11-08-20','12-08
20','13-08-20','14-08-20','15-08-20','16-08-20','17-08-20','18-08-20','19-08-20','20-08-2
','21-08-20','22-08-20','23-08-20','24-08-20','25-08-20','26-08-20','27-08-20','28-08-20'
'29-08-20','30-08-20','31-08-20','01-09-20','02-09-20','03-09-20','04-09-20','05-09-20','
6-09-20','07-09-20','08-09-20','09-09-20','10-09-20','11-09-20','12-09-20','13-09-20','14
09-20','15-09-20','16-09-20','17-09-20','18-09-20','19-09-20','20-09-20','21-09-20','22-0
-20','23-09-20','24-09-20','25-09-20','26-09-20','27-09-20','28-09-20','29-09-20','30-09-
0','01-10-20','02-10-20','03-10-20','04-10-20','05-10-20','06-10-20','07-10-20','08-10-20
,'09-10-20','10-10-20','11-10-20','12-10-20','13-10-20','14-10-20','15-10-20','16-10-20',
17-10-20','18-10-20','19-10-20','20-10-20','21-10-20','22-10-20','23-10-20','24-10-20','2
-10-20','26-10-20','27-10-20','28-10-20','29-10-20','30-10-20','31-10-20','01-11-20','02-
1-20','03-11-20','04-11-20','05-11-20','06-11-20','07-11-20','08-11-20','09-11-20','10-11
20','11-11-20','12-11-20','13-11-20','14-11-20','15-11-20','16-11-20','17-11-20','18-11-2
','19-11-20','20-11-20','21-11-20','22-11-20','23-11-20','24-11-20','25-11-20','26-11-20'
'27-11-20','28-11-20','29-11-20','30-11-20','01-12-20','02-12-20','03-12-20','04-12-20','
5-12-20','06-12-20','07-12-20','08-12-20','09-12-20','10-12-20','11-12-20','12-12-20','13
12-20','14-12-20','15-12-20']
py.subplots(figsize=(16, 8))

fig = sns.scatterplot(x="Sr no.", y="Reach", palette="mako", hue="Type of post",data=dat
a_clean_blanks)
sns.regplot(x="Sr no.", y="Reach",data=data_clean_blanks, order=2, scatter=False, fit_re
g=True, color="gray")
#sns.rugplot(y="Reach", hue="Type of post",palette="mako",data=data_clean_blanks, height=
0.01, lw=1, alpha=0.5)
xaxis = []
for i in range(0,len(x_axis_labels),(int(len(x_axis_labels)/8))):
    xaxis = xaxis+[x_axis_labels[i]]
fig.set_xticklabels(xaxis)
py.xlabel('Dates')
py.show()
```



3. New reach with respect to Time

Displaying contents of csv file

In [9]:

```
data_clean_blanks.head()
```

Out[9]:

	Sr no.	Date	Dates	Post no.	Type of post	Likes	Comments	Shares	Save	Profile visits	...	Total	Follows	Reach	New Reach	New reach %	Profile	H
0	1	1	01-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
1	2	2	02-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
2	3	3	03-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
3	4	4	04-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	
4	5	5	05-06-2020	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	

5 rows × 23 columns

In [10]:

```
data_clean_blanks.describe()
```

Out[10]:

	Sr no.	Date	Post no.	Likes	Comments	Shares	Save	Profile visits	Website clicks	Ema
count	250.000000	250.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.0000
mean	125.500000	15.560000	56.000000	179.954955	21.117117	66.333333	31.756757	87.531532	4.009009	0.1711
std	72.312977	8.955849	32.186954	63.514979	17.795167	83.047996	25.713179	132.259718	11.631849	0.7245
min	1.000000	1.000000	1.000000	75.000000	0.000000	0.000000	6.000000	2.000000	0.000000	0.0000
25%	63.250000	8.000000	28.500000	137.500000	8.500000	14.000000	17.000000	25.500000	0.000000	0.0000
50%	125.500000	15.500000	56.000000	169.000000	16.000000	43.000000	26.000000	50.000000	1.000000	0.0000
75%	187.750000	23.000000	83.500000	202.500000	28.500000	80.500000	36.000000	84.500000	2.500000	0.0000
max	250.000000	31.000000	111.000000	472.000000	101.000000	444.000000	195.000000	871.000000	92.000000	6.0000

8 rows × 21 columns

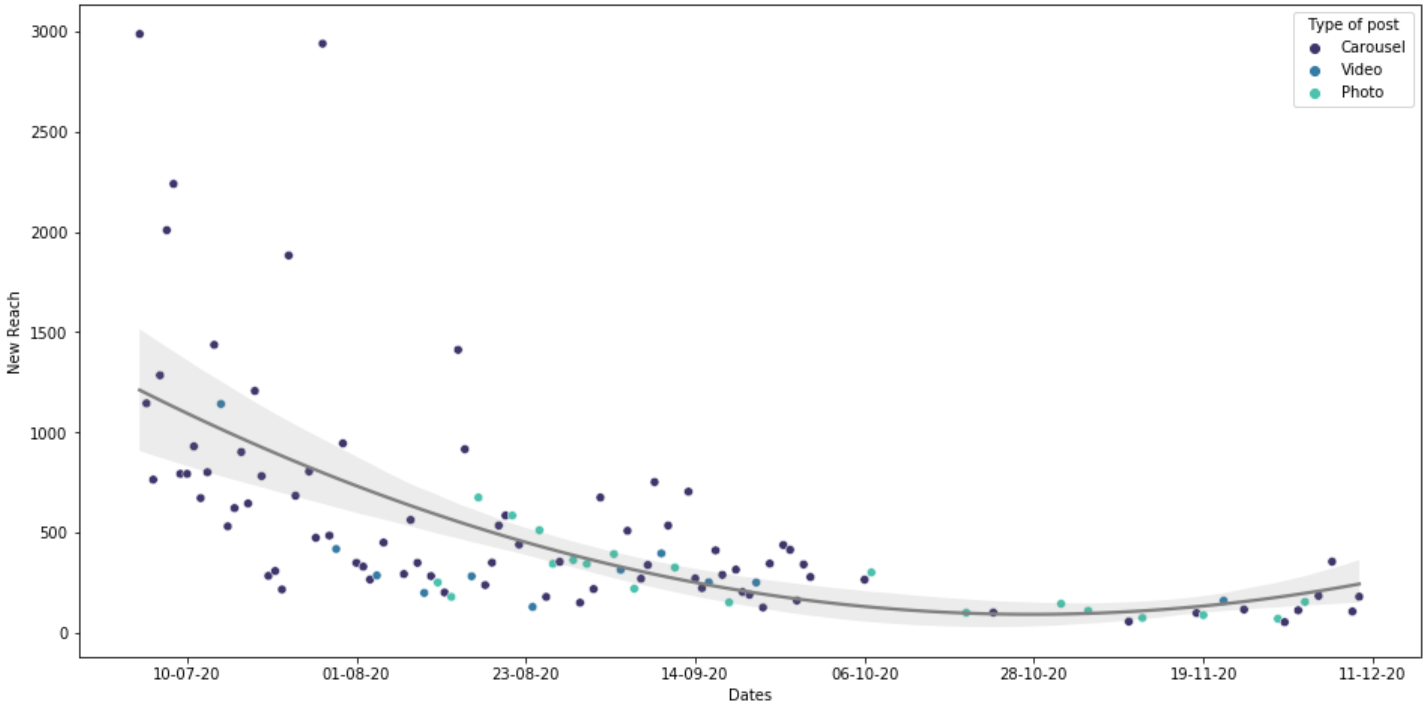
Visualization

In [11]:

```
x_axis_labels = ['18-06-20','19-06-20','20-06-20','21-06-20','22-06-20','23-06-20','24-06
-20','25-06-20','26-06-20','27-06-20','28-06-20','29-06-20','30-06-20','01-07-20','02-07-
0','03-07-20','04-07-20','05-07-20','06-07-20','07-07-20','08-07-20','09-07-20','10-07-20
,'11-07-20','12-07-20','13-07-20','14-07-20','15-07-20','16-07-20','17-07-20','18-07-20',
19-07-20','20-07-20','21-07-20','22-07-20','23-07-20','24-07-20','25-07-20','26-07-20','2
-07-20','28-07-20','29-07-20','30-07-20','31-07-20','01-08-20','02-08-20','03-08-20','04-
8-20','05-08-20','06-08-20','07-08-20','08-08-20','09-08-20','10-08-20','11-08-20','12-08
20','13-08-20','14-08-20','15-08-20','16-08-20','17-08-20','18-08-20','19-08-20','20-08-2
','21-08-20','22-08-20','23-08-20','24-08-20','25-08-20','26-08-20','27-08-20','28-08-20'
'29-08-20','30-08-20','31-08-20','01-09-20','02-09-20','03-09-20','04-09-20','05-09-20','
6-09-20','07-09-20','08-09-20','09-09-20','10-09-20','11-09-20','12-09-20','13-09-20','14
09-20','15-09-20','16-09-20','17-09-20','18-09-20','19-09-20','20-09-20','21-09-20','22-0
-20','23-09-20','24-09-20','25-09-20','26-09-20','27-09-20','28-09-20','29-09-20','30-09-
0','01-10-20','02-10-20','03-10-20','04-10-20','05-10-20','06-10-20','07-10-20','08-10-20
```

```
,'09-10-20','10-10-20','11-10-20','12-10-20','13-10-20','14-10-20','15-10-20','16-10-20',
17-10-20','18-10-20','19-10-20','20-10-20','21-10-20','22-10-20','23-10-20','24-10-20','2
-10-20','26-10-20','27-10-20','28-10-20','29-10-20','30-10-20','31-10-20','01-11-20','02-
1-20','03-11-20','04-11-20','05-11-20','06-11-20','07-11-20','08-11-20','09-11-20','10-11
20','11-11-20','12-11-20','13-11-20','14-11-20','15-11-20','16-11-20','17-11-20','18-11-2
','19-11-20','20-11-20','21-11-20','22-11-20','23-11-20','24-11-20','25-11-20','26-11-20'
'27-11-20','28-11-20','29-11-20','30-11-20','01-12-20','02-12-20','03-12-20','04-12-20','
5-12-20','06-12-20','07-12-20','08-12-20','09-12-20','10-12-20','11-12-20','12-12-20','13
12-20','14-12-20','15-12-20']
py.subplots(figsize=(16, 8))

fig = sns.scatterplot(x="Sr no.", y="New Reach", palette="mako", hue="Type of post",data
=data_clean_blanks)
sns.regplot(x="Sr no.", y="New Reach",data=data_clean_blanks, order=2, scatter=False, fi
t_reg=True, color="gray")
#sns.rugplot(y="Reach", hue="Type of post",palette="mako",data=data_clean_blanks, height=
0.01, lw=1, alpha=0.5)
xaxis = []
for i in range(0,len(x_axis_labels),(int(len(x_axis_labels)/8))):
    xaxis = xaxis+[x_axis_labels[i]]
fig.set_xticklabels(xaxis)
py.xlabel('Dates')
py.show()
```



4. User/Follower activity heatmap

Displaying contents of csv file

In [12]:

```
data_timings.head()
```

Out[12]:

	Time(standard format)	Time	Day	Active Users	Time(improvised)	Time final
0	12:30 AM	0	Monday	1066	1	15
1	12:30 AM	0	Tuesday	1024	1	15
2	12:30 AM	0	Wednesday	1046	1	15
3	12:30 AM	0	Thursday	1031	1	15
4	12:30 AM	0	Friday	993	1	15

In [13]:

```
data_timings.describe()
```

Out[13]:

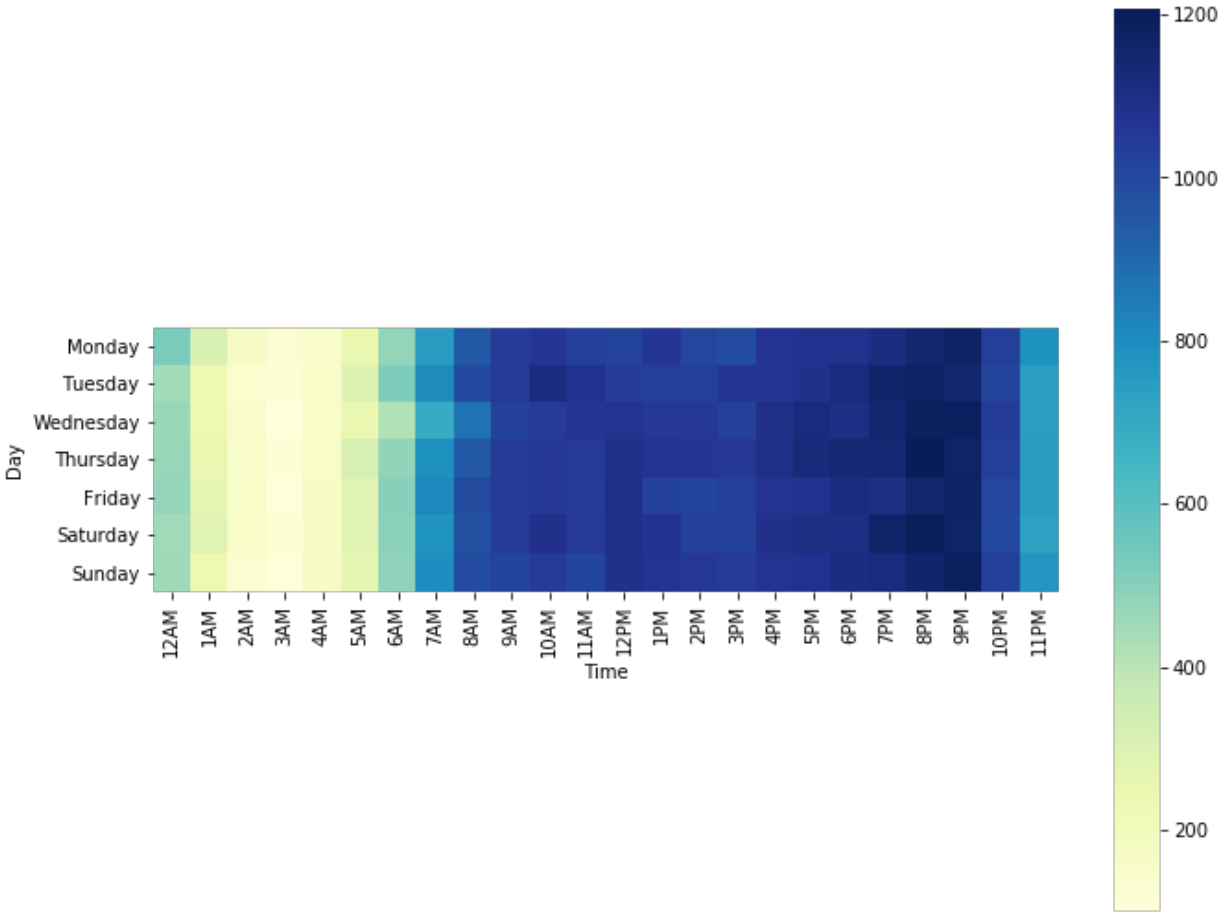
	Time	Active Users	Time(improvised)	Time final
count	168.000000	168.000000	168.000000	168.000000
mean	11.500000	815.886905	11.500000	11.500000
std	6.942881	369.080853	6.942881	6.942881
min	0.000000	101.000000	0.000000	0.000000
25%	5.750000	478.000000	5.750000	5.750000
50%	11.500000	1027.500000	11.500000	11.500000
75%	17.250000	1083.750000	17.250000	17.250000
max	23.000000	1208.000000	23.000000	23.000000

Visualization

In [14]:

```
data = data_timings.pivot("Day", "Time final", "Active Users")
y_axis_labels = ["Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sun
day"]
x_axis_labels = ["12AM", "1AM", "2AM", "3AM", "4AM", "5AM", "6AM", "7AM", "8AM", "9AM",
"10AM", "11AM", "12PM","1PM", "2PM", "3PM", "4PM", "5PM", "6PM", "7PM", "8PM", "9PM", "1
0PM", "11PM"]


fig, ax = py.subplots(figsize=(11, 9))
sns.heatmap(data, cmap="YlGnBu", yticklabels=y_axis_labels, xticklabels=x_axis_labels, sq
uare=True)
py.xlabel("Time")
py.show()
```



5. Engagement rating with respect to Type of posts

Displaying contents of csv file

In [15]:

```
data_clean1 = data_clean[["Dates", "Post no.", "Type of post","Likes", "Comments", "Shar
es", "Save", "Reach", "Engagement rate %"]]
data_clean1.head()
```

Out[15]:

	Dates	Post no.	Type of post	Likes	Comments	Shares	Save	Reach	Engagement rate %
0	18-06-2020	1	Carousel	368	37	444	105	3284	6.26
1	19-06-2020	2	Carousel	188	9	52	29	1612	3.08
2	20-06-2020	3	Carousel	163	13	26	29	1212	3.38
3	21-06-2020	4	Carousel	188	68	112	85	1760	5.70
4	22-06-2020	5	Carousel	290	38	288	73	2511	5.78

In [16]:

```
data_clean1.describe()
```

Out[16]:

	Post no.	Likes	Comments	Shares	Save	Reach	Engagement rate %
count	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000
mean	56.000000	179.954955	21.117117	66.333333	31.756757	1469.945946	3.635856
std	32.186954	63.514979	17.795167	83.047996	25.713179	481.838880	1.381729
min	1.000000	75.000000	0.000000	0.000000	6.000000	744.000000	1.580000
25%	28.500000	137.500000	8.500000	14.000000	17.000000	1204.000000	2.690000
50%	56.000000	169.000000	16.000000	43.000000	26.000000	1353.000000	3.540000
75%	83.500000	202.500000	28.500000	80.500000	36.000000	1616.500000	4.340000
max	111.000000	472.000000	101.000000	444.000000	195.000000	3920.000000	8.730000

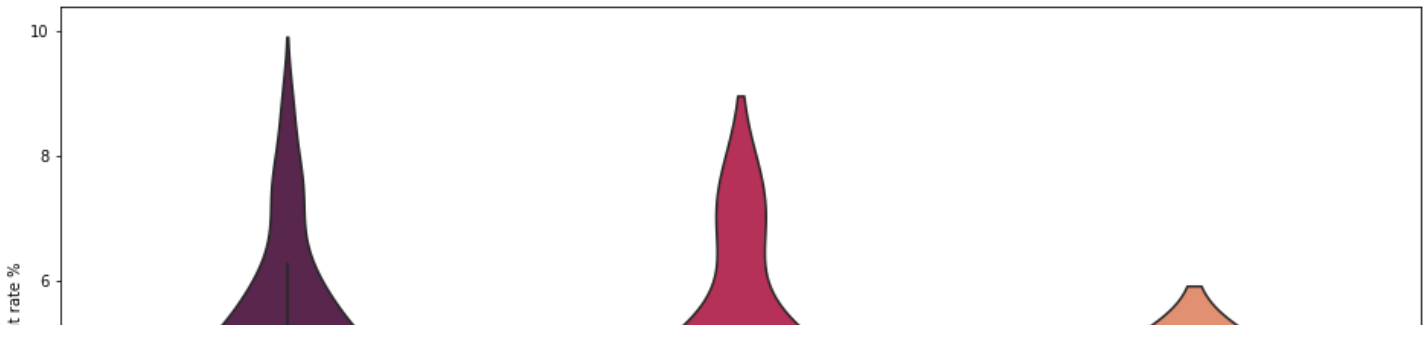
Visualization

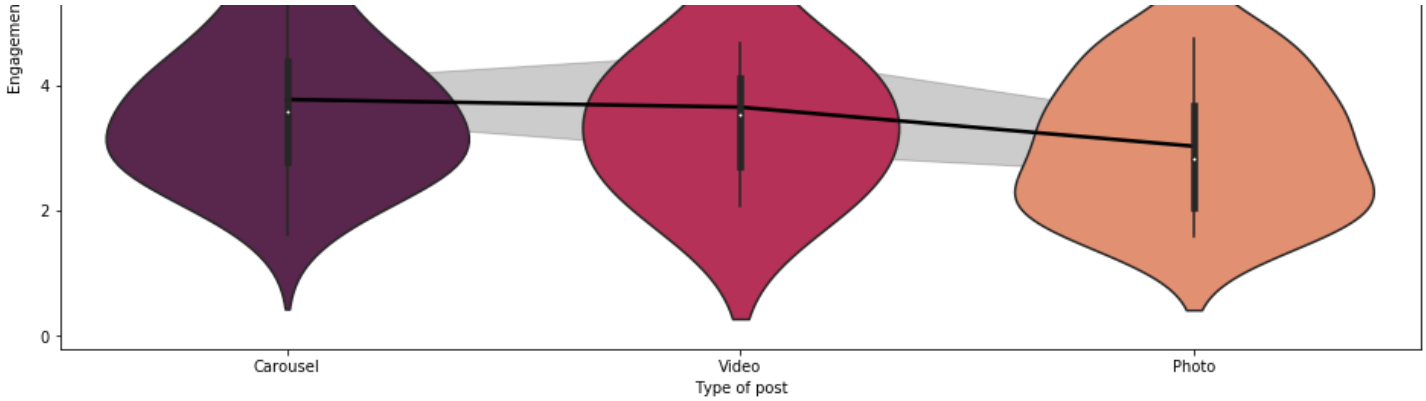
In [17]:

```
py.subplots(figsize=(16, 8))
sns.lineplot(x="Type of post", y="Engagement rate %", data=data_clean1, color="black", l
inewidth=2.5, markers= ["o"])
sns.violinplot(x="Type of post", y="Engagement rate %", data=data_clean1, palette="rocke
t")
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f54a732bd90>
```





6. Save with respect to reach

Displaying contents of csv file

In [18]:

```
data_clean2 = data_clean[["Dates", "Post no.", "Type of post", "Save", "Reach", "Save vs Reach"]]
data_clean2.head()
```

Out[18]:

	Dates	Post no.	Type of post	Save	Reach	Save vs Reach
0	18-06-2020	1	Carousel	105	3284	3.20
1	19-06-2020	2	Carousel	29	1612	1.80
2	20-06-2020	3	Carousel	29	1212	2.39
3	21-06-2020	4	Carousel	85	1760	4.83
4	22-06-2020	5	Carousel	73	2511	2.91

In [19]:

```
data_clean2.describe()
```

Out[19]:

	Post no.	Save	Reach	Save vs Reach
count	111.000000	111.000000	111.000000	111.000000
mean	56.000000	31.756757	1469.945946	2.052613
std	32.186954	25.713179	481.838880	1.176670
min	1.000000	6.000000	744.000000	0.600000
25%	28.500000	17.000000	1204.000000	1.275000
50%	56.000000	26.000000	1353.000000	1.830000
75%	83.500000	36.000000	1616.500000	2.510000
max	111.000000	195.000000	3920.000000	8.670000

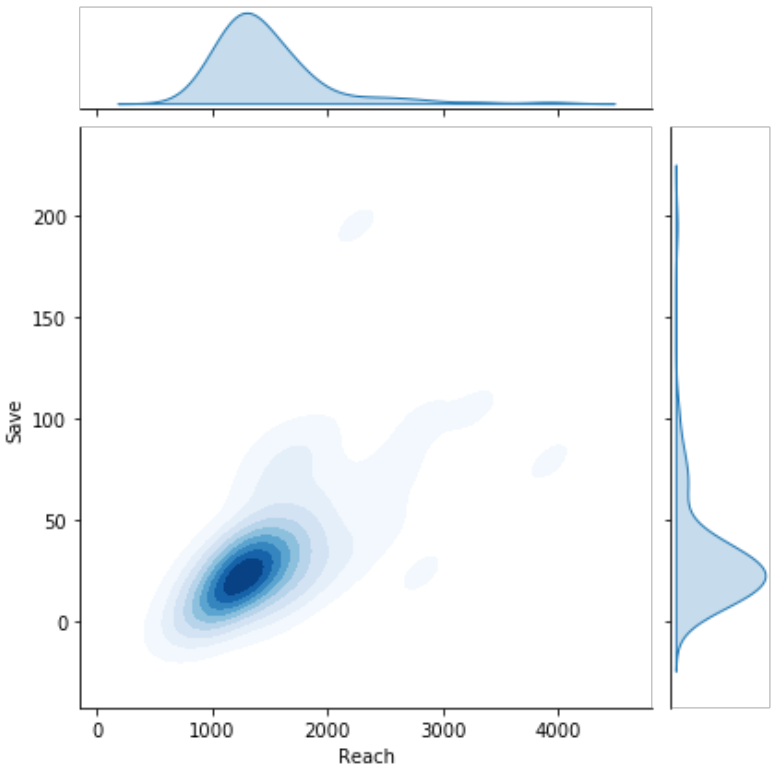
Visualization

In [20]:

```
sns.jointplot(x="Reach", y="Save", data=data_clean2, kind="kde", cmap="Blues", levels = 10, fill = True)
```

Out[20]:

```
<seaborn.axisgrid.JointGrid at 0x7f54a72581d0>
```



7. Effective Frequency with respect to Post no.

Displaying contents of csv file

In [21]:

```
data_clean3 = data_clean[["Dates", "Post no.", "Type of post", "Total - Impressions", "Reach", "Effective frequency"]]
data_clean3.head()
```

Out[21]:

	Dates	Post no.	Type of post	Total - Impressions	Reach	Effective frequency
0	18-06-2020	1	Carousel	4157	3284	1.265834
1	19-06-2020	2	Carousel	2163	1612	1.341811
2	20-06-2020	3	Carousel	1640	1212	1.353135
3	21-06-2020	4	Carousel	2297	1760	1.305114
4	22-06-2020	5	Carousel	3142	2511	1.251294

In [22]:

```
data_clean3.describe()
```

Out[22]:

	Post no.	Total - Impressions	Reach	Effective frequency
count	111.000000	111.000000	111.000000	111.000000
mean	56.000000	1873.738739	1469.945946	1.277737
std	32.186954	591.163317	481.838880	0.078430
min	1.000000	891.000000	744.000000	1.107021
25%	28.500000	1519.000000	1204.000000	1.229228
50%	56.000000	1707.000000	1353.000000	1.290347

	Post no.	Total - Impressions	Reach	Effective frequency
75%	83.500000	2114.000000	1616.500000	1.331822
max	111.000000	4552.000000	3920.000000	1.440299

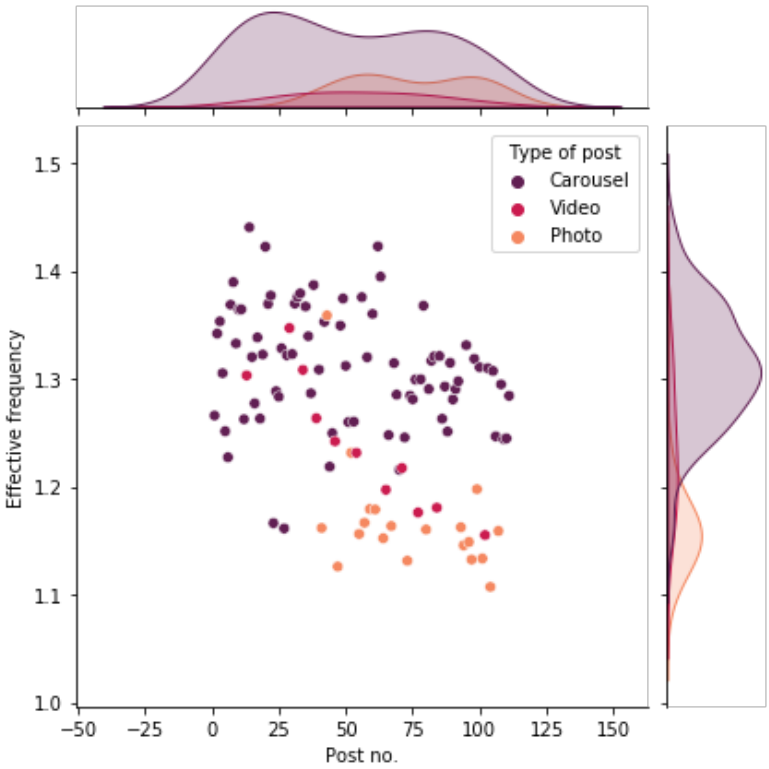
Visualization

In [23]:

```
sns.jointplot(x= "Post no.", y= "Effective frequency",hue="Type of post" ,data=data_clean
3, palette="rocket")
```

Out[23]:

```
<seaborn.axisgrid.JointGrid at 0x7f54a74735d0>
```



8.Profile visit with "Dates", "Post no.", "Type of post", "Total - Impressions", "Reach", "Effective frequency"respect to Time

Displaying contents of csv file

In [24]:

```
data_clean4 = data_clean[["Dates", "Post no.", "Type of post", "Profile visits - Interac
tions", "New Reach", "Profile visit rate%"]]
data_clean4.head()
```

Out[24]:

	Dates	Post no.	Type of post	Profile visits - Interactions	New Reach	Profile visit rate%
0	18-06-2020	1	Carousel	871	2988	29.15
1	19-06-2020	2	Carousel	114	1145	9.96
2	20-06-2020	3	Carousel	84	764	10.99
3	21-06-2020	4	Carousel	123	1285	9.57
4	22-06-2020	5	Carousel	719	2009	35.79

In [25]:

```
data_clean4.describe()
```

Out[25]:

	Post no.	Profile visits - Interactions	New Reach	Profile visit rate%
count	111.000000	111.000000	111.000000	111.000000
mean	56.000000	87.531532	510.441441	17.715676
std	32.186954	132.259718	521.574307	12.411734
min	1.000000	2.000000	53.000000	1.790000
25%	28.500000	25.500000	209.500000	9.240000
50%	56.000000	50.000000	344.000000	13.340000
75%	83.500000	84.500000	633.500000	21.680000
max	111.000000	871.000000	2988.000000	58.890000

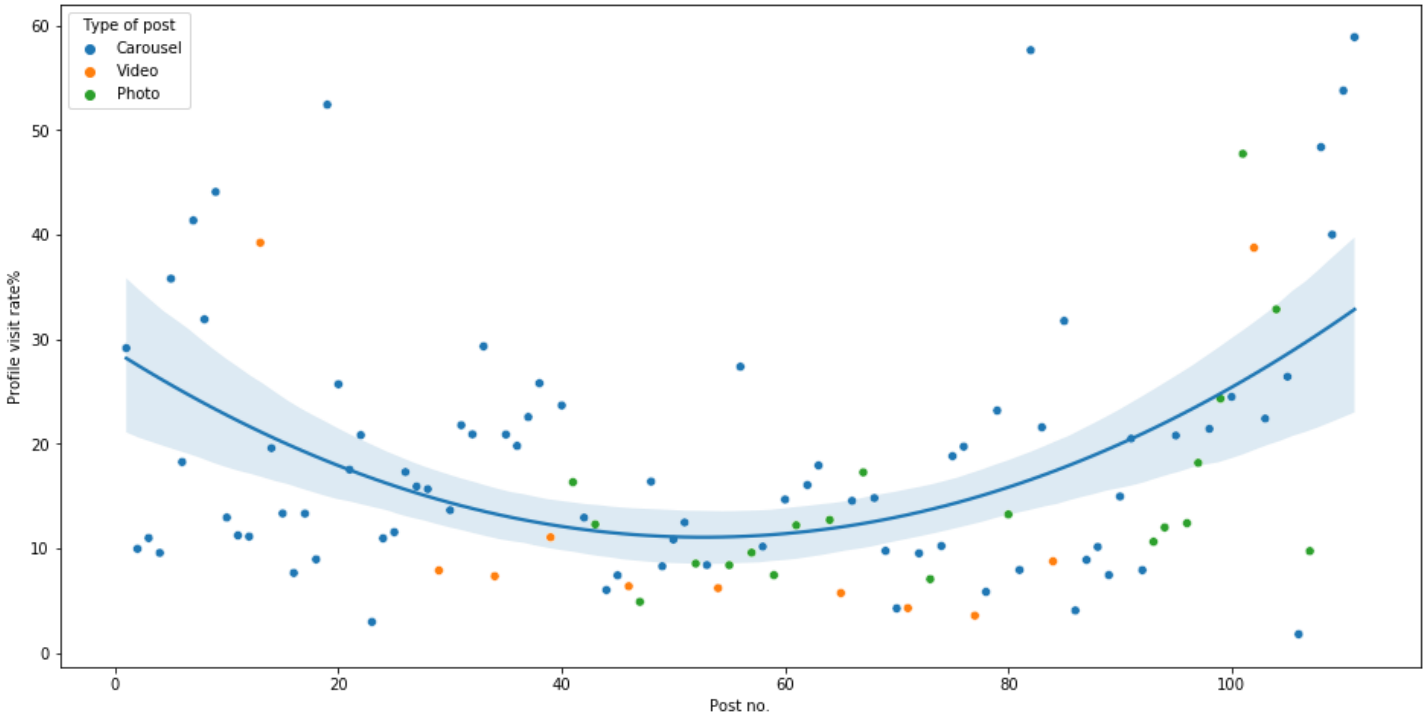
Visualization

In [26]:

```
py.subplots(figsize=(16, 8))
sns.regplot(x="Post no.", y="Profile visit rate%", data=data_clean4, order=2, scatter=False, fit_reg=True)
sns.scatterplot(x="Post no.", y="Profile visit rate%", hue="Type of post",data=data_clean4)
```

Out[26]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f54a7567dd0>
```



9.Follow rate with respect to Post type

Displaying contents of csv file

In [27]:

```
data_clean5 = data_clean[["Dates", "Post no.", "Type of post","Follows","Profile visits - Interactions", "New Reach", "Profile visit rate%","Follow rate/profile visit %"]]
data_clean5.head()
```

Out[27]:

	Dates	Post no.	Type of post	Follows	Profile visits - Interactions	New Reach	Profile visit rate%	Follow rate/profile visit %
0	18-06-2020	1	Carousel	124	871	2988	29.15	14.24
1	19-06-2020	2	Carousel	16	114	1145	9.96	14.04
2	20-06-2020	3	Carousel	5	84	764	10.99	5.95
3	21-06-2020	4	Carousel	7	123	1285	9.57	5.69
4	22-06-2020	5	Carousel	92	719	2009	35.79	12.80

In [28]:

```
data_clean5.describe()
```

Out[28]:

	Post no.	Follows	Profile visits - Interactions	New Reach	Profile visit rate%	Follow rate/profile visit %
count	111.000000	111.000000	111.000000	111.000000	111.000000	111.000000
mean	56.000000	7.369369	87.531532	510.441441	17.715676	5.352523
std	32.186954	17.393273	132.259718	521.574307	12.411734	4.809207
min	1.000000	0.000000	2.000000	53.000000	1.790000	0.000000
25%	28.500000	0.500000	25.500000	209.500000	9.240000	0.560000
50%	56.000000	2.000000	50.000000	344.000000	13.340000	4.550000
75%	83.500000	5.000000	84.500000	633.500000	21.680000	8.200000
max	111.000000	124.000000	871.000000	2988.000000	58.890000	21.430000

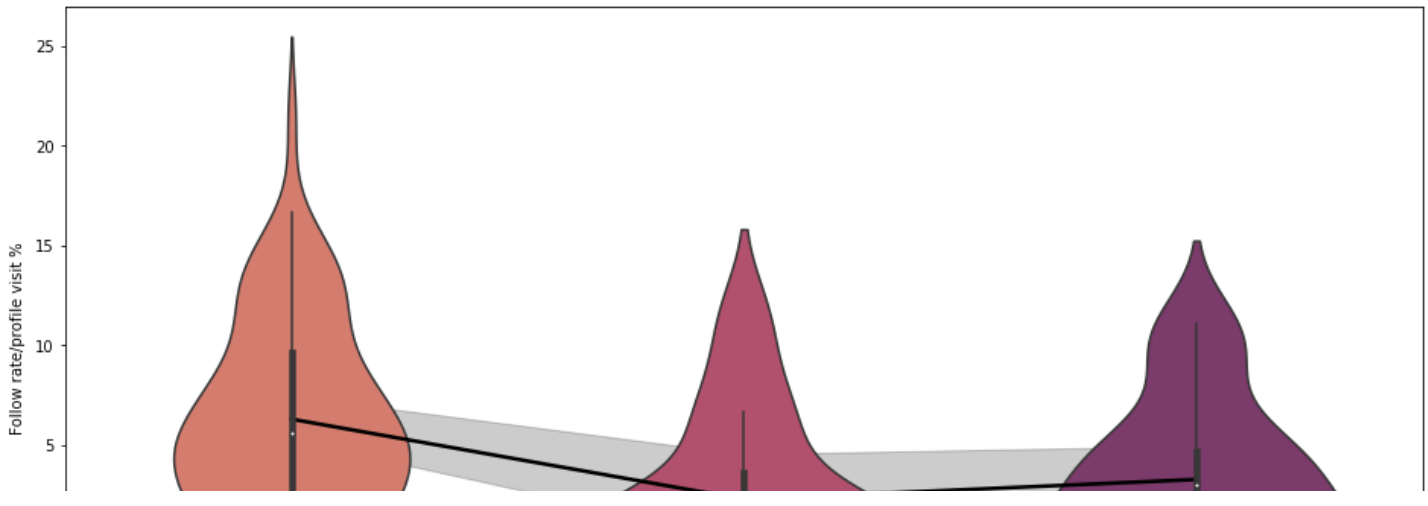
Visualization

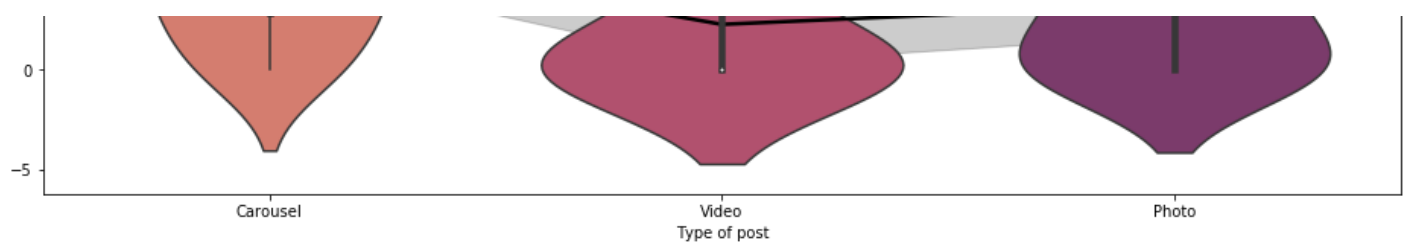
In [29]:

```
py.subplots(figsize=(16, 8))
sns.lineplot(x="Type of post", y="Follow rate/profile visit %", data=data_clean5, color="
black", linewidth=2.5, markers= ["o"])
sns.violinplot(x="Type of post", y="Follow rate/profile visit %", data=data_clean5, palet
te="flare")
```

Out[29]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f54a751c790>
```




0
−5
Carousel
Video
Photo
Type of post