

# Main

May 15, 2021

## 1 Data Exploration

In this chapter we are going to explore the data and extract useful insights in order to increase business understanding and problem knowledge to perform better modeling.

---

```
[10]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import warnings
warnings.filterwarnings("ignore")
import seaborn as sns
from openpyxl import load_workbook
np.set_printoptions(suppress=True)
pd.set_option('display.float_format', lambda x: '%.2f' % x)
```

```
[25]: xls = pd.ExcelFile('data/Main Dataset V3.0.xlsx')
ad_post = pd.read_excel(xls, 'Ad-Post')
ad_story = pd.read_excel(xls, 'Ad-Story')
influencer = pd.read_excel(xls, 'Influencer')
leaders_post = pd.read_excel(xls, 'Leaders-Post')
leaders_story = pd.read_excel(xls, 'Leaders-Story')
post = pd.read_excel(xls, 'Post')
story = pd.read_excel(xls, 'Story')
print('Datasets Loaded Completely.')
```

Datasets Loaded Completely.

In the below cells you can the top 5 row and features of prepared datasets. Please have in mind that we already tackle the problem of missing data with imputations which you can see implementation in separate file.

```
[26]: print('Advertising Posts first 5 rows:')
ad_post.head()
```

Advertising Posts first 5 rows:

```
[26]:
```

	ad_post_no	name	follower	field	view	cost \
0	1	3kanstv	1000000	video	9435	1242664.83
1	2	bazigaran.iraani	1700000	video	7926	1242664.83
2	3	bedaanim	1700000	fact	19433	1117313.08
3	4	bekhaanim	224000	art & culture	8424	484343.80
4	5	beyond.mag	1000000	fact	8212	966517.09

	threshold
0	80
1	80
2	80
3	80
4	80

```
[27]: print('Advertising Stories first 5 rows:')
ad_story.head()
```

Advertising Stories first 5 rows:

```
[27]:
```

	ad_story_no	name	field	view	threshold	follower	action \
0	0	4rahesalamat	health	6260	20	686000	82
1	1	90tv.official	news	58990	20	877000	234
2	2	ancientworld1	fact	101631	20	2600000	273
3	3	ayamidooni	fact	97671	20	2300000	365
4	4	banoooye_khone	women	21887	20	2400000	239

	interaction	impression	cost
0	7	6374	559250.09
1	90	58568	1302915.56
2	218	94682	1631578.95
3	488	92023	1907421.43
4	38	74414	1261832.64

```
[28]: print('Minor Influencers first 5 rows:')
influencer.head()
```

Minor Influencers first 5 rows:

```
[28]:
```

	story_no	influ_name	gender	field	l_threshold	h_threshold \
0	0	ali_bakhtiarvandi	family	lifestyle	40	80
1	1	ali_bakhtiarvandi	family	lifestyle	40	80
2	2	ali_bakhtiarvandi	family	lifestyle	40	80
3	3	ali_bakhtiarvandi	family	lifestyle	40	80
4	4	ali_bakhtiarvandi	family	lifestyle	40	80

	follower	view	action	impression	cta	interaction	cost
0	141000	3996	14	4186	0	0	502206.47
1	141000	3279	30	3473	1	28	502206.47

2	141000	3636	5	3867	0	0	502206.47
3	141000	3145	16	3317	1	11	502206.47
4	141000	3113	30	3286	1	22	502206.47

```
[29]: print('Main influencers stories first 5 rows:')
      leaders_story.head()
```

Main influencers stories first 5 rows:

```
[29]: story_no      name gender cost follower view action \
0      0  aidapooryanasab female    0   692000  103909   651
1      1    alimona.trips  family    0    73400   4169   162
2      2  amirparsaneshat   male    0   146000  26972   527
3      3 ghonche.ostovarnia female    0   122000   8381   205
4      4      maandani    male    0   128000  10493   178

      interaction impression
0             562      107902
1             130       3548
2             335      26925
3             154       8381
4             151      10952
```

```
[30]: print('Main influencers posts first 5 rows:')
      leaders_post.head()
```

Main influencers posts first 5 rows:

```
[30]: post_no      name gender l_threshold h_threshold follower \
0      0  aidapooryanasab female       250         500   692000
1      1    alimona.trips  family       250         500    73400
2      2  amirparsaneshat   male       250         500   146000
3      3 ghonche.ostovarnia female       250         500   122000
4      4      maandani    male       250         500   128000

      view  like  comment  share  save  profile_visit  reach  impression \
0   78137 17500     205    275   272         1374  149048   162532
1   20220  5099     140    238   138          463   31642    38437
2  128378 25940     573   7732  7207         2593  146276   180104
3  103347 12300     733    261   471         6611  156349   172354
4   15002  2408      68     98   232          482   27562    30204

      cost
0  41813725.49
1   6968954.25
2  21185620.92
3   8362745.10
4   7247712.42
```

```
[32]: print('Campaing stories first 5 rows:')
      story.head()
```

Campaing stories first 5 rows:

```
[32]:  story_no  type  view  action  reply  profile_visit  share  website_click  \
0         0  share  1337    53      4           49      0           0
1         1  share  1164   114      2          110      1           1
2         2  share   727    21      1           20      0           0
3         3  share   850    45      5           40      0           0
4         4  share  1294    69      8           58      0           3

      sticker_tap  impression  follow  navigation  back  forward  next  exit  \
0         0      1380      0      1618      28    1048    179    363
1         0      1190      1      1490     106     919    119    350
2         0       765      0       772      38     428     92    214
3         0       930      1      1038      31     531    125    351
4         0      1384      0      1522      35     909    186    392

      vote
0         0
1         0
2         0
3         0
4         0
```

```
[33]: print('Campaign posts first 5 rows:')
      post.head()
```

Campaign posts first 5 rows:

```
[33]:  post_no  like  comment  share  save  profile_visit  reach  impression  \
0         1  2118   15636   1448   381           339  13760    16292
1         2   611     26    44    41           112   3968     5018
2         3  1408   15438   862   129           345   8157     9908
3         4   741    566   109    68            73   4957     6024
4         5   567     6    47    42           148   4616     5024

      ig_tv      view
0         1  98313.00
1         0         nan
2         1 170437.00
3         1   2762.00
4         0         nan
```

first thing first, we must check how many records and features are there in our datasets.

```
[34]: print(f'There are {ad_story.shape[0]} Records and {ad_story.shape[1]} Features_
      ↪in Advertising Stories Dataset.')
```

```

print(f'There are {ad_post.shape[0]} Records and {ad_post.shape[1]} Features in_
↳Advertising Posts Dataset.')
print(f'There are {influencer.shape[0]} Records and {influencer.shape[1]}_
↳Features in Minor Influencers Dataset.')
print(f'There are {leaders_story.shape[0]} Records and {leaders_story.shape[1]}_
↳Features in Main Influencers Stories Dataset.')
print(f'There are {leaders_post.shape[0]} Records and {leaders_post.shape[1]}_
↳Features in Main Influencers Posts Dataset.')
print(f'There are {story.shape[0]} Records and {story.shape[1]} Features in_
↳Campaign Stories Dataset.')
print(f'There are {post.shape[0]} Records and {post.shape[1]} Features in Posts_
↳Dataset.')

```

There are 27 Records and 10 Features in Advertising Stories Dataset.

There are 27 Records and 7 Features in Advertising Posts Dataset.

There are 102 Records and 13 Features in Minor Influencers Dataset.

There are 12 Records and 9 Features in Main Influencers Stories Dataset.

There are 9 Records and 15 Features in Main Influencers Posts Dataset.

There are 40 Records and 17 Features in Campaign Stories Dataset.

There are 13 Records and 10 Features in Posts Dataset.

## 1.1 Media Effectiveness Indicator

In this step we are going to implement new feature based on threshold and paid price for the media. For datasets that have a range threshold we are going to implement multi class feature for them.

It's important to have this facts in mind: - Negative value in price difference means that specific medium charged us more than it should and positive value means that we benefitted from that medium more than we paid based on main deciding factor, which is view. - 'Benefit' feature is a binary class which shows that we are benefitting or not.

```

[35]: ad_post['cost_per_view'] = ad_post['view'] * ad_post['threshold']
ad_post['price_difference'] = ad_post['cost_per_view'] - ad_post['cost']
ad_post['benefit'] = (ad_post['price_difference'] >= 0).astype(int)

ad_story['cost_per_view'] = ad_story['view'] * ad_story['threshold']
ad_story['price_difference'] = ad_story['cost_per_view'] - ad_story['cost']
ad_story['benefit'] = (ad_story['price_difference'] >= 0).astype(int)

influencer['lowest_cost_per_view'] = influencer['l_threshold'] *_
↳influencer['view']
influencer['highest_cost_per_view'] = influencer['h_threshold'] *_
↳influencer['view']
influencer['benefit'] = np.where(influencer['cost'] <_
↳influencer['lowest_cost_per_view'], 1, np.where(influencer['cost'] >_
↳influencer['highest_cost_per_view'], -1, 0))

```

```

leaders_post['lowest_cost_per_view'] = leaders_post['l_threshold'] *
↳ leaders_post['view']
leaders_post['highest_cost_per_view'] = leaders_post['h_threshold'] *
↳ leaders_post['view']
leaders_post['benefit'] = np.where(leaders_post['cost'] <
↳ leaders_post['lowest_cost_per_view'], 1, np.where(leaders_post['cost'] >
↳ leaders_post['highest_cost_per_view'], -1, 0))

```

### 1.1.1 Overall Cost Status for Paid Media

In this section we are going to review overall status of paid media for different approaches used for this campaign.

```

[36]: print('In Advertising Posts:')
print(f'\tNumber of Benefit media: {ad_post["benefit"].value_counts()[1]}')
print(f'\tNumber of Loss media: {ad_post["benefit"].value_counts()[0]}')
print(f'\tOverall Cost: {ad_post["cost"].sum():,}')
print(f'\tActual Cost per View: {ad_post["cost_per_view"].sum():,}')
print(f'\tBenefit Amount: {ad_post["price_difference"].sum():,}')

print('\nIn Advertising Stories:')
print(f'\tNumber of Benefit media: {ad_story["benefit"].value_counts()[1]}')
print(f'\tNumber of Loss media: {ad_story["benefit"].value_counts()[0]}')
print(f'\tOverall Cost: {ad_story["cost"].sum():,}')
print(f'\tActual Cost per View: {ad_story["cost_per_view"].sum():,}')
print(f'\tBenefit Amount: {ad_story["price_difference"].sum():,}')

print('\nIn Influencers:')
print(f'\tNumber of Benefit media: {influencer["benefit"].value_counts()[1]}')
print(f'\tNumber of Neutral media: {influencer["benefit"].value_counts()[0]}')
print(f'\tNumber of Loss media: {influencer["benefit"].value_counts()[-1]}')
print(f'\tOverall Cost: {influencer["cost"].sum():,}')
print(f'\tLowest Anticipated Overall Cost: {influencer["lowest_cost_per_view"].
↳ sum():,}')
print(f'\tHighest Anticipated Overall Cost:
↳ {influencer["highest_cost_per_view"].sum():,}')
print(f'\tAverage Anticipated Overall Cost:
↳ {((influencer["highest_cost_per_view"].sum() +
↳ influencer["lowest_cost_per_view"].sum()) / 2):,}')

print('\nIn Main Influencers Posts:')
print(f'\tNumber of Benefit media: {leaders_post["benefit"].value_counts()[1]}')
print(f'\tNumber of Neutral media: {leaders_post["benefit"].value_counts()[0]}')
print(f'\tNumber of Loss media: {leaders_post["benefit"].value_counts()[-1]}')
print(f'\tOverall Cost: {leaders_post["cost"].sum():,}')

```

```

print(f'\tLowest Anticipated Overall Cost:␣
↳{leaders_post["lowest_cost_per_view"].sum():,}')
print(f'\tHighest Anticipated Overall Cost:␣
↳{leaders_post["highest_cost_per_view"].sum():,}')
print(f'\tAverage Anticipated Overall Cost:␣
↳{((leaders_post["highest_cost_per_view"].sum() +␣
↳leaders_post["lowest_cost_per_view"].sum()) / 2):,}')

```

#### In Advertising Posts:

Number of Benefit media: 15  
 Number of Loss media: 12  
 Overall Cost: 39,999,999.99999991  
 Actual Cost per View: 40,289,360  
 Benefit Amount: 289,360.00000008615

#### In Advertising Stories:

Number of Benefit media: 16  
 Number of Loss media: 11  
 Overall Cost: 30,999,999.99999996  
 Actual Cost per View: 29,916,500  
 Benefit Amount: -1,083,499.999999961

#### In Influencers:

Number of Benefit media: 36  
 Number of Neutral media: 43  
 Number of Loss media: 23  
 Overall Cost: 78,399,999.99999984  
 Lowest Anticipated Overall Cost: 89,208,000  
 Highest Anticipated Overall Cost: 178,416,000  
 Average Anticipated Overall Cost: 133,812,000.0

#### In Main Influencers Posts:

Number of Benefit media: 2  
 Number of Neutral media: 5  
 Number of Loss media: 2  
 Overall Cost: 170,599,999.9999997  
 Lowest Anticipated Overall Cost: 126,810,500  
 Highest Anticipated Overall Cost: 253,621,000  
 Average Anticipated Overall Cost: 190,215,750.0

A very interesting insight that can be get from this exploration is that the threshold for influencers are not set correctly. Agency charged customer less than the anticipated price for this number of views.

### 1.1.2 Descriptive Analysis of Datasets

In this part we are going to check the descriptive analysis of datasets and their scatter matrix among every single feature.

#### Advertising Posts:

```
[37]: ad_post.describe()
```

```
[37]:
```

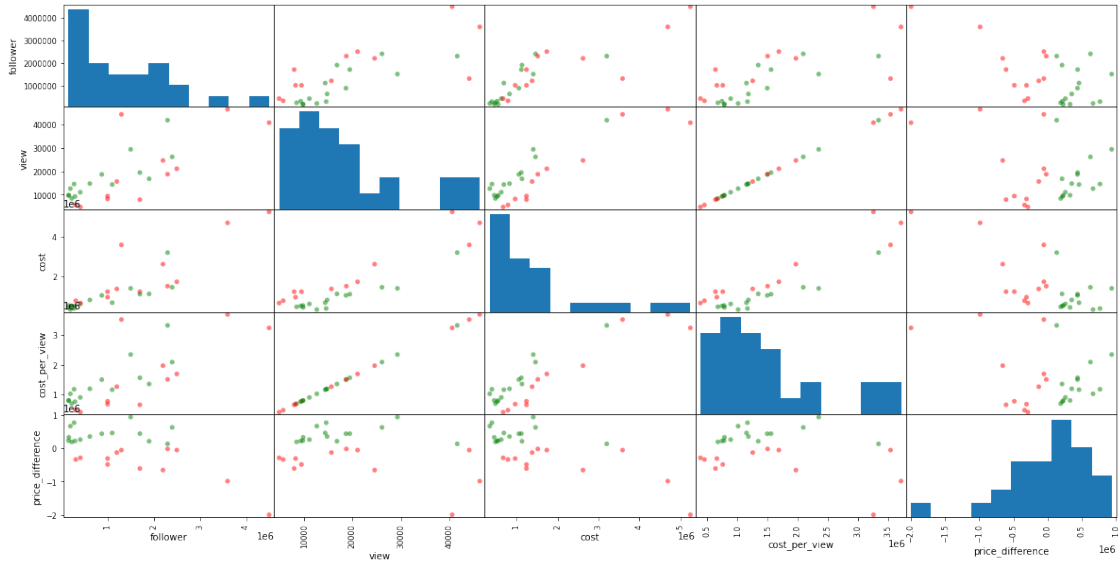
	ad_post_no	follower	view	cost	threshold	cost_per_view \
count	27.00	27.00	27.00	27.00	27.00	27.00
mean	14.00	1337814.81	18652.48	1481481.48	80.00	1492198.52
std	7.94	1112368.09	12119.83	1290670.47	0.00	969586.48
min	1.00	153000.00	4808.00	346156.71	80.00	384640.00
25%	7.50	359500.00	9581.00	650347.26	80.00	766480.00
50%	14.00	1100000.00	14744.00	1117313.08	80.00	1179520.00
75%	20.50	2050000.00	22814.50	1491197.79	80.00	1825160.00
max	27.00	4500000.00	46340.00	5246807.04	80.00	3707200.00

	price_difference	benefit
count	27.00	27.00
mean	10717.04	0.56
std	610319.73	0.51
min	-1996407.04	0.00
25%	-298991.51	0.00
50%	189576.20	1.00
75%	392976.64	1.00
max	934848.80	1.00

```
[38]: pd.plotting.scatter_matrix(ad_post.drop(['ad_post_no', 'threshold', 'benefit'],  
→axis=1), figsize=(20,10), s=100,  
c = np.where(ad_post['benefit'] == 1, 'green', 'red'))  
plt.show()
```





## Advertising Stories:

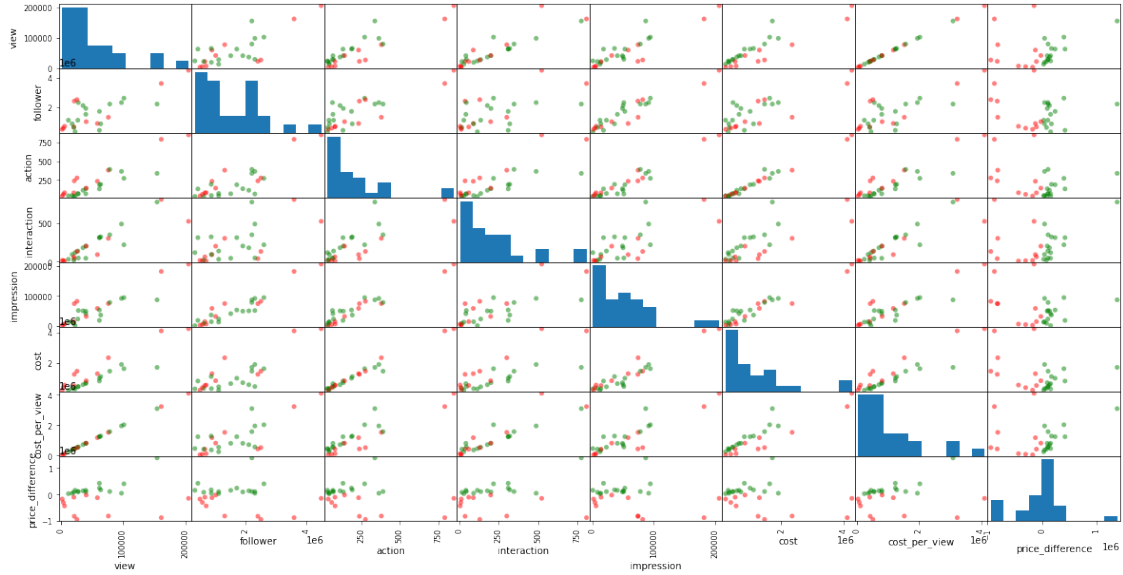
[39]: `ad_story.describe()`

```
[39]:
```

	ad_story_no	view	threshold	follower	action	interaction	\
count	27.00	27.00	27.00	27.00	27.00	27.00	
mean	13.00	55400.93	20.00	1577074.07	209.96	215.48	
std	7.94	51021.85	0.00	1031146.56	210.22	219.63	
min	0.00	2514.00	20.00	311000.00	33.00	7.00	
25%	6.50	21549.50	20.00	769000.00	66.50	44.50	
50%	13.00	40400.00	20.00	1300000.00	135.00	137.00	
75%	19.50	70493.50	20.00	2250000.00	275.50	311.00	
max	26.00	205375.00	20.00	4500000.00	850.00	807.00	

	impression	cost	cost_per_view	price_difference	benefit
count	27.00	27.00	27.00	27.00	27.00
mean	53746.30	1148148.15	1108018.52	-40129.63	0.59
std	50542.85	1041974.85	1020436.93	470992.24	0.50
min	1141.00	220087.09	50280.00	-942727.25	0.00
25%	14319.50	452720.61	430990.00	-160914.29	0.00
50%	49339.00	822861.61	808000.00	45998.57	1.00
75%	76583.50	1467247.25	1409870.00	146356.75	1.00
max	206633.00	4255017.04	4107500.00	1379700.71	1.00

```
[40]: pd.plotting.scatter_matrix(ad_story.drop(['ad_story_no', 'threshold',
↪ 'benefit'], axis=1), figsize=(20,10), s=100,
                                             c = np.where(ad_story['benefit'] == 1, 'green',
↪ 'red'))
plt.show()
```



### Minor Influencers:

```
[41]: influencer.describe()
```

```
[41]:
```

	story_no	l_threshold	h_threshold	follower	view	action	\
count	102.00	102.00	102.00	102.00	102.00	102.00	
mean	50.50	40.00	80.00	183950.00	21864.71	254.63	
std	29.59	0.00	0.00	159314.56	28090.10	565.16	
min	0.00	40.00	80.00	18000.00	396.00	3.00	
25%	25.25	40.00	80.00	47000.00	4628.25	30.00	
50%	50.50	40.00	80.00	141000.00	12000.00	103.50	
75%	75.75	40.00	80.00	266000.00	21874.25	229.00	
max	101.00	40.00	80.00	570000.00	125371.00	4108.00	

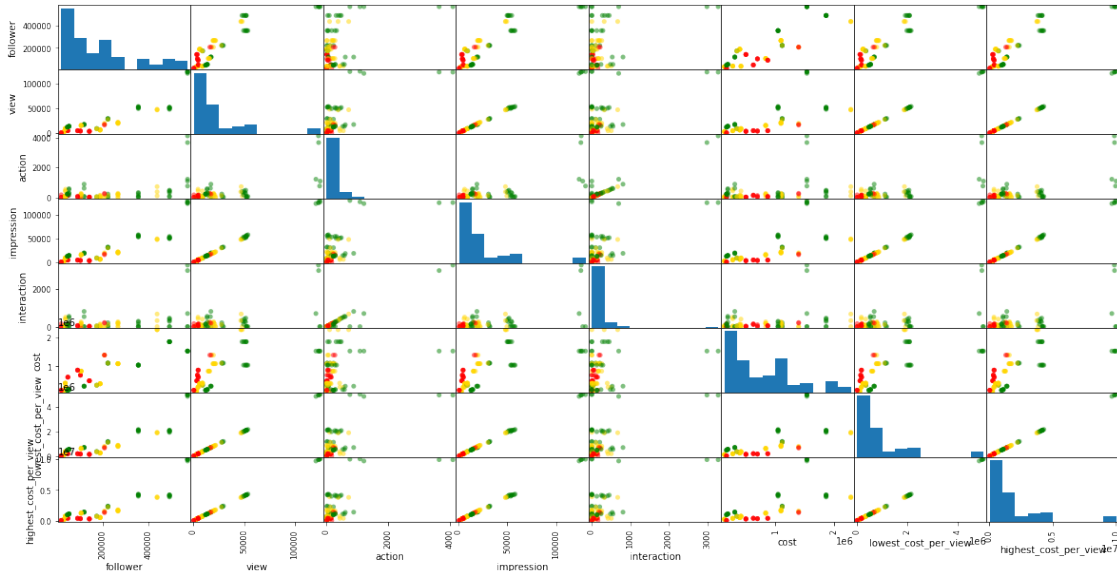
	impression	cta	interaction	cost	lowest_cost_per_view	\
count	102.00	102.00	102.00	102.00	102.00	
mean	22799.44	0.61	183.68	768627.45	874588.24	
std	28774.04	0.49	454.80	562628.11	1123603.97	
min	823.00	0.00	0.00	139501.80	15840.00	
25%	4967.00	0.00	0.00	313879.04	185130.00	
50%	12876.00	1.00	42.50	627758.09	480000.00	
75%	22286.75	1.00	184.25	1096085.15	874970.00	
max	129903.00	1.00	3284.00	2278528.88	5014840.00	

	highest_cost_per_view	benefit
count	102.00	102.00
mean	1749176.47	0.13
std	2247207.93	0.75
min	31680.00	-1.00

25%	370260.00	0.00
50%	960000.00	0.00
75%	1749940.00	1.00
max	10029680.00	1.00

```
[42]: pd.plotting.scatter_matrix(influencer.drop(['story_no', 'l_threshold', 'h_threshold', 'cta'], axis=1), figsize=(20,10), s=100,
    ↪ 'benefit', 'h_threshold', 'cta'], axis=1), figsize=(20,10), s=100,
    c=np.where(influencer['benefit'] == 1, 'green', np.
    ↪ where(influencer['benefit'] == -1, 'red', 'gold')))
plt.show()
```



### Major Influencers Advertising Posts:

```
[43]: leaders_post.describe()
```

```
[43]:
```

	post_no	l_threshold	h_threshold	follower	view	like	\
count	9.00	9.00	9.00	9.00	9.00	9.00	
mean	4.00	250.00	500.00	254933.33	56360.22	9759.44	
std	2.74	0.00	0.00	269557.86	47940.99	8203.85	
min	0.00	250.00	500.00	54000.00	6191.00	1201.00	
25%	2.00	250.00	500.00	122000.00	15701.00	2766.00	
50%	4.00	250.00	500.00	133000.00	31714.00	7890.00	
75%	6.00	250.00	500.00	189000.00	103347.00	12731.00	
max	8.00	250.00	500.00	757000.00	128378.00	25940.00	

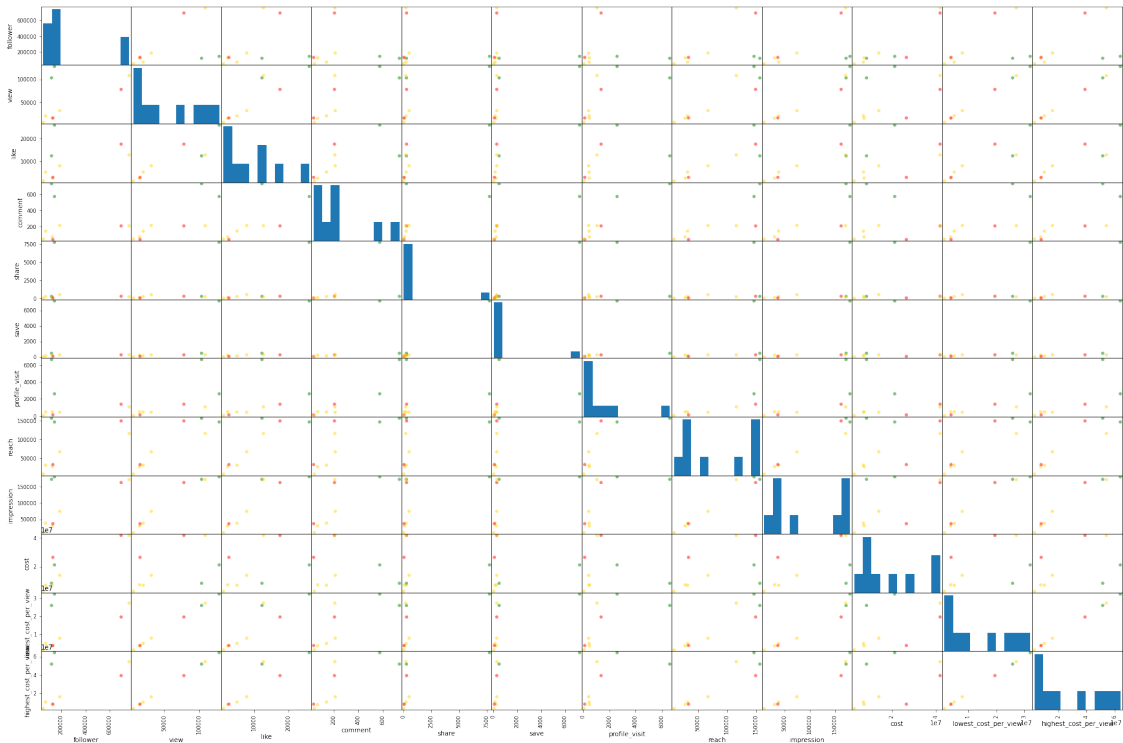
  

	comment	share	save	profile_visit	reach	impression	\
count	9.00	9.00	9.00	9.00	9.00	9.00	
mean	246.00	1041.22	996.33	1466.56	81695.44	97358.44	

std	244.42	2513.23	2332.72	2083.63	60007.57	72536.55
min	35.00	15.00	24.00	64.00	8311.00	9589.00
25%	68.00	98.00	138.00	427.00	31642.00	36830.00
50%	205.00	238.00	272.00	482.00	67071.00	74606.00
75%	211.00	275.00	278.00	1374.00	146276.00	171570.00
max	733.00	7732.00	7207.00	6611.00	156349.00	180104.00

	cost	lowest_cost_per_view	highest_cost_per_view	benefit
count	9.00	9.00	9.00	9.00
mean	18955555.56	14090055.56	28180111.11	0.00
std	14942191.99	11985246.47	23970492.93	0.71
min	2787581.70	1547750.00	3095500.00	-1.00
25%	7247712.42	3925250.00	7850500.00	0.00
50%	13937908.50	7928500.00	15857000.00	0.00
75%	26482026.14	25836750.00	51673500.00	0.00
max	41813725.49	32094500.00	64189000.00	1.00

```
[44]: pd.plotting.scatter_matrix(leaders_post.drop(['post_no', 'l_threshold',
↳ 'benefit', 'h_threshold'], axis=1), figsize=(30,20), s=100,
      c=np.where(leaders_post['benefit'] == 1, 'green', np.
↳ where(leaders_post['benefit'] == -1, 'red', 'gold')))
plt.show()
```



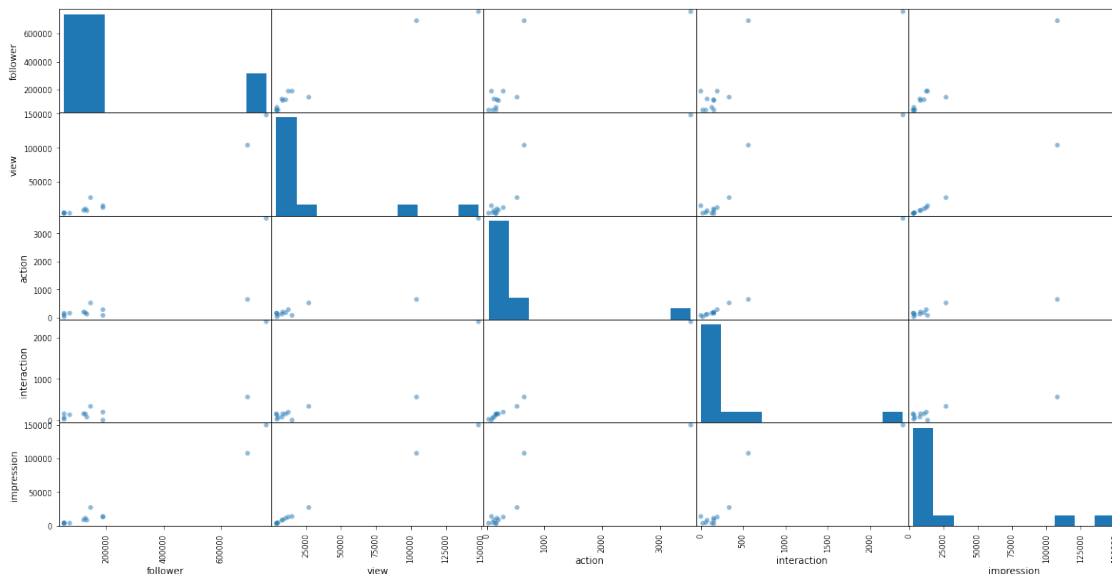
## Major Influencers Advertising Stories:

```
[45]: leaders_story.describe()
```

```
[45]:
```

	story_no	cost	follower	view	action	interaction	impression
count	12.00	12.00	12.00	12.00	12.00	12.00	12.00
mean	5.50	0.00	215950.00	29196.67	505.00	352.25	29381.58
std	3.61	0.00	242740.18	46663.27	972.49	660.23	47822.71
min	0.00	0.00	54000.00	3803.00	34.00	0.00	3002.00
25%	2.75	0.00	68550.00	4823.50	122.00	70.25	4058.75
50%	5.50	0.00	130500.00	9437.00	170.00	152.50	9666.50
75%	8.25	0.00	189000.00	18008.75	347.75	229.25	16875.00
max	11.00	0.00	757000.00	148197.00	3538.00	2392.00	150001.00

```
[46]: pd.plotting.scatter_matrix(leaders_story.drop(['story_no', 'cost'], axis=1),
    ↳ figsize=(20,10), s=100)
plt.show()
```



## Campaign Posts:

```
[47]: post.describe()
```

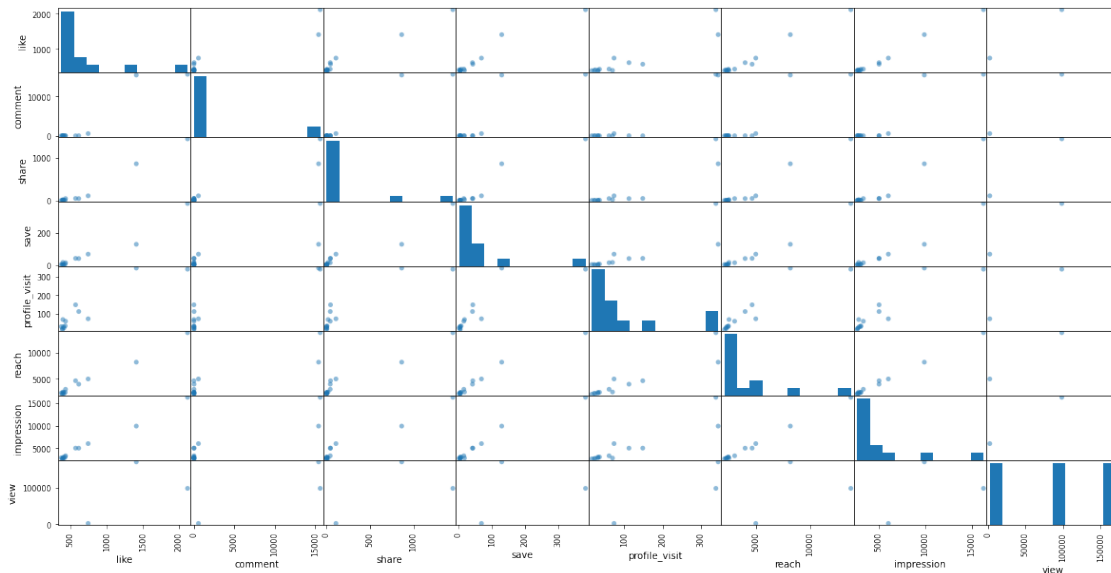
```
[47]:
```

	post_no	like	comment	share	save	profile_visit	reach	\
count	13.00	13.00	13.00	13.00	13.00	13.00	13.00	
mean	7.00	664.15	2443.15	199.38	55.15	100.23	4192.23	
std	3.89	520.58	5813.36	441.72	104.46	113.97	3353.03	
min	1.00	368.00	0.00	1.00	1.00	15.00	2057.00	
25%	4.00	391.00	11.00	3.00	4.00	31.00	2282.00	
50%	7.00	424.00	13.00	19.00	15.00	60.00	2474.00	
75%	10.00	611.00	26.00	47.00	42.00	112.00	4616.00	

max	13.00	2118.00	15636.00	1448.00	381.00	345.00	13760.00
-----	-------	---------	----------	---------	--------	--------	----------

	impression	ig_tv	view
count	13.00	13.00	3.00
mean	5038.62	0.23	90504.00
std	3959.25	0.44	84109.82
min	2655.00	0.00	2762.00
25%	2823.00	0.00	50537.50
50%	3061.00	0.00	98313.00
75%	5024.00	0.00	134375.00
max	16292.00	1.00	170437.00

```
[48]: pd.plotting.scatter_matrix(post.drop(['post_no', 'ig_tv'], axis=1),
    ↪    figsize=(20,10), s=100)
plt.show()
```



### Campaign Story:

```
[49]: story.describe()
```

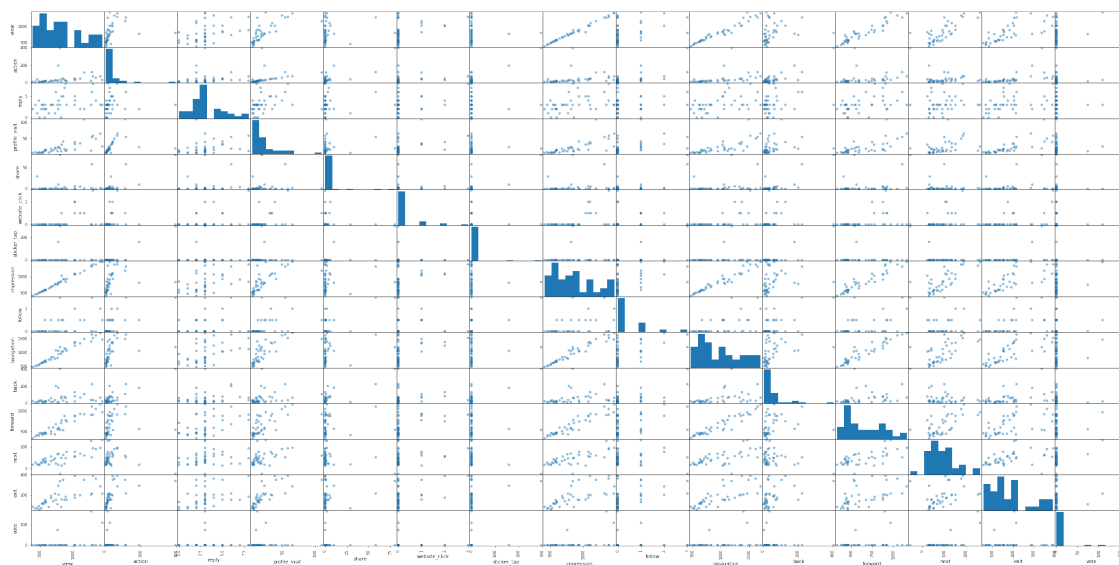
[49]:	story_no	view	action	reply	profile_visit	share	website_click	\
count	40.00	40.00	40.00	40.00	40.00	40.00	40.00	
mean	19.50	812.62	41.15	3.17	21.27	5.05	0.28	
std	11.69	290.68	68.53	1.93	21.46	15.84	0.68	
min	0.00	393.00	5.00	0.00	3.00	0.00	0.00	
25%	9.75	577.25	11.25	2.00	8.25	0.00	0.00	
50%	19.50	770.50	19.50	3.00	13.50	0.00	0.00	
75%	29.25	1021.50	39.00	4.00	25.50	1.25	0.00	

max	39.00	1434.00	397.00	8.00	110.00	80.00	3.00
-----	-------	---------	--------	------	--------	-------	------

	sticker_tap	impression	follow	navigation	back	forward	next \
count	40.00	40.00	40.00	40.00	40.00	40.00	40.00
mean	11.38	843.15	0.45	983.60	58.45	647.33	93.38
std	52.56	308.03	0.81	360.90	76.89	228.79	63.63
min	0.00	410.00	0.00	472.00	6.00	332.00	-53.00
25%	0.00	586.00	0.00	664.75	17.75	455.25	47.75
50%	0.00	792.00	0.00	946.50	30.00	575.50	88.00
75%	0.00	1072.50	1.00	1267.75	65.25	876.00	126.50
max	296.00	1465.00	3.00	1702.00	405.00	1160.00	264.00

	exit	vote
count	40.00	40.00
mean	183.05	8.70
std	94.68	32.64
min	58.00	0.00
25%	116.00	0.00
50%	156.00	0.00
75%	214.25	0.00
max	392.00	165.00

```
[50]: pd.plotting.scatter_matrix(story.drop(['story_no'], axis=1), figsize=(40,20),
    ↪s=100)
plt.show()
```

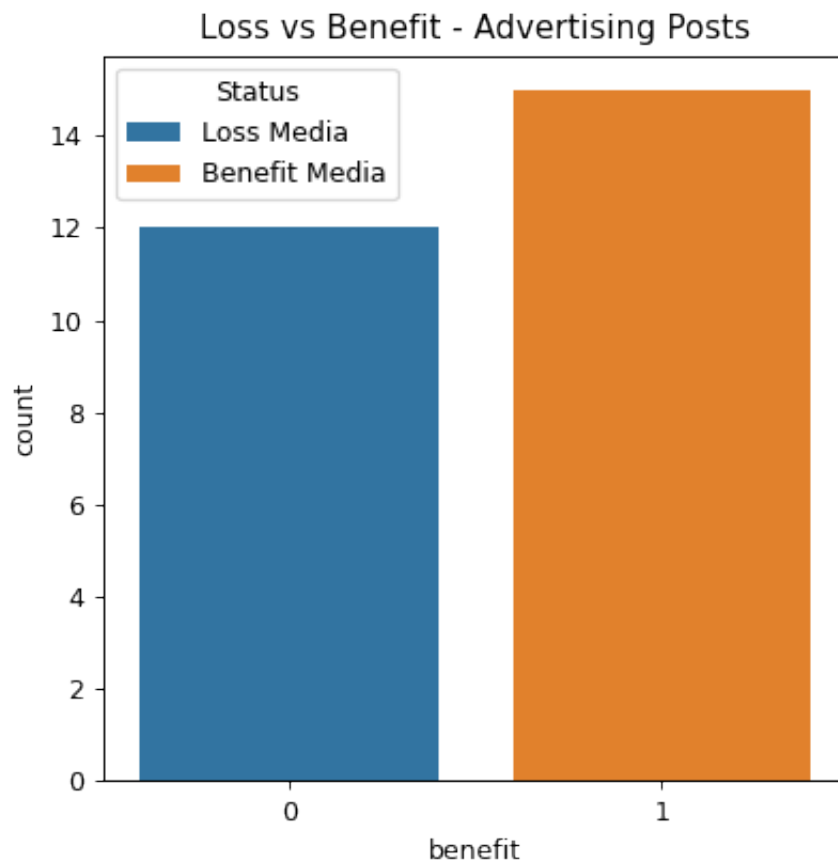


### 1.1.3 Data Exploration:

In this step we are going to explore the data and extract some insights from it.

```
[51]: plt.figure(figsize=(5,5), dpi=90)
g = sns.countplot(x="benefit", data = ad_post, dodge = False, hue='benefit')
h,l = g.get_legend_handles_labels()
labels=['Loss Media','Benefit Media']
g.legend(h,labels,title="Status", loc="upper left")
plt.title('Loss vs Benefit - Advertising Posts')
plt.show()

count_benefit = len(ad_post[ad_post['benefit'] == 1])
count_loss = len(ad_post[ad_post['benefit'] == 0])
print(f'The number of benefit media are: {count_benefit}.')
print(f'The number of loss media are: {count_loss}.')
```



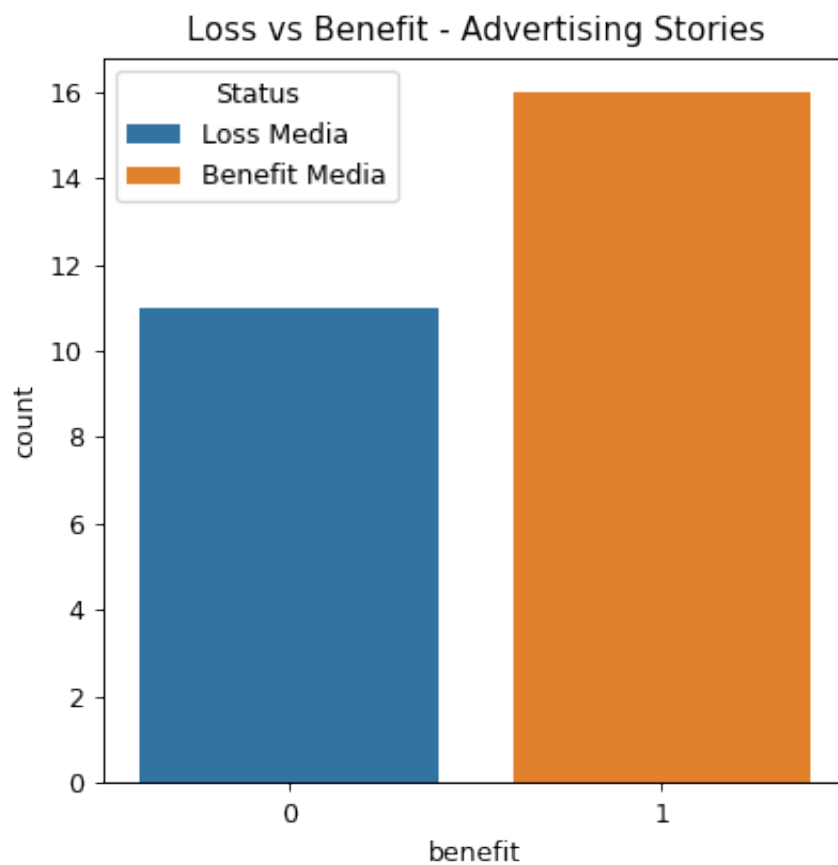
The number of benefit media are: 15.

The number of loss media are: 12.



```
[52]: plt.figure(figsize=(5,5), dpi=90)
g = sns.countplot(x="benefit", data = ad_story, dodge = False, hue='benefit')
h,l = g.get_legend_handles_labels()
labels=['Loss Media', 'Benefit Media']
g.legend(h,labels,title="Status", loc="upper left")
plt.title('Loss vs Benefit - Advertising Stories')
plt.show()

count_benefit = len(ad_story[ad_story['benefit'] == 1])
count_loss = len(ad_story[ad_story['benefit'] == 0])
print(f'The number of benefit media are: {count_benefit}.')
print(f'The number of loss media are: {count_loss}.')
```



The number of benefit media are: 16.  
The number of loss media are: 11.

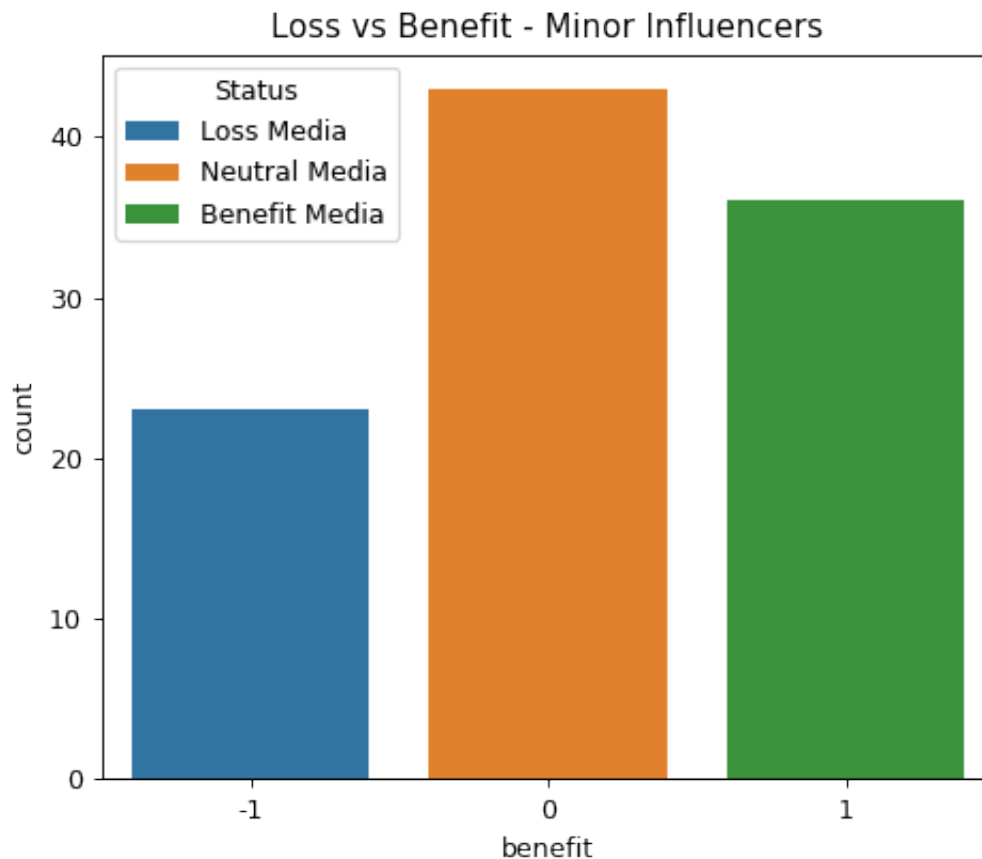
```
[53]: plt.figure(figsize=(6,5), dpi=90)
g = sns.countplot(x="benefit", data = influencer, dodge = False, hue='benefit')
h,l = g.get_legend_handles_labels()
labels=['Loss Media', 'Neutral Media', 'Benefit Media']
```

```

g.legend(h,labels,title="Status", loc="upper left")
plt.title('Loss vs Benefit - Minor Influencers')
plt.show()

count_benefit = len(influencer[influencer['benefit'] == 1])
count_loss = len(influencer[influencer['benefit'] == -1])
count_neutral = len(influencer[influencer['benefit'] == 0])
print(f'The number of benefit media are: {count_benefit}.')
print(f'The number of loss media are: {count_loss}.')
print(f'The number of neutral media are: {count_neutral}.')

```



The number of benefit media are: 36.  
 The number of loss media are: 23.  
 The number of neutral media are: 43.

```

[54]: plt.figure(figsize=(6,5), dpi=90)
      g = sns.countplot(x="benefit", data = leaders_post, dodge = False,
      ↪hue='benefit')
      h,l = g.get_legend_handles_labels()

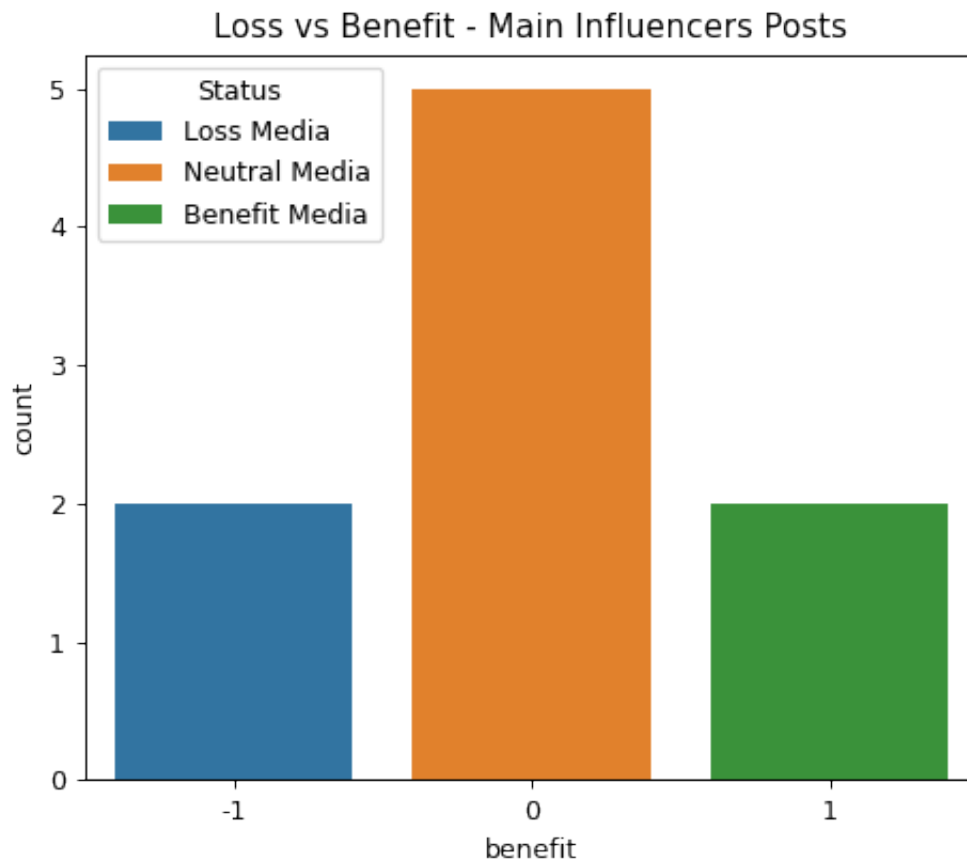
```

```

labels=['Loss Media','Neutral Media', 'Benefit Media']
g.legend(h,labels,title="Status", loc="upper left")
plt.title('Loss vs Benefit - Main Influencers Posts')
plt.show()

count_benefit = len(leaders_post[leaders_post['benefit'] == 1])
count_loss = len(leaders_post[leaders_post['benefit'] == -1])
count_neutral = len(leaders_post[leaders_post['benefit'] == 0])
print(f'The number of benefit media are: {count_benefit}.')
print(f'The number of loss media are: {count_loss}.')
print(f'The number of neutral media are: {count_neutral}.')

```

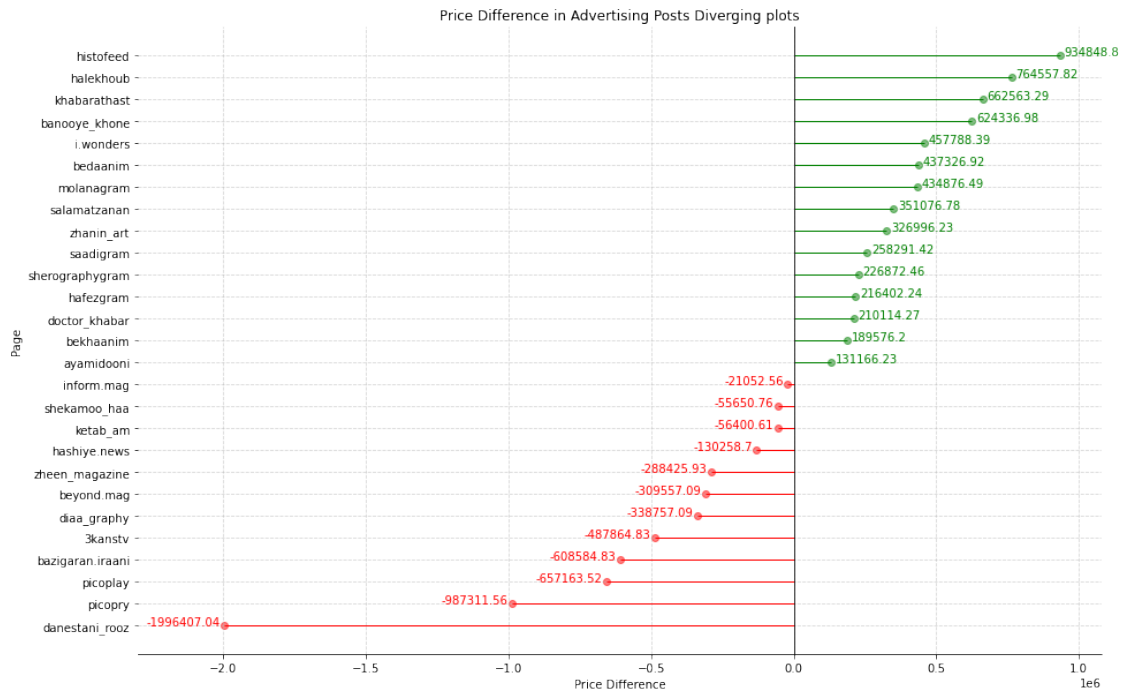


The number of benefit media are: 2.  
 The number of loss media are: 2.  
 The number of neutral media are: 5.

#### 1.1.4 Price Difference among advertising approaches diverging plot and Anticipated Cost vs Actual cost in Minor and Major Influencers

```
[59]: temp_df = ad_post.sort_values('price_difference')
temp_df.reset_index(inplace = True)
colors = []
for x in temp_df['price_difference']:
    if x < 0:
        colors.append('red')
    elif x > 0:
        colors.append('green')
    else:
        colors.append('goldenrod')

fig = plt.figure(figsize = (15, 10))
ax = fig.add_subplot()
ax.hlines(y = temp_df.index, xmin = 0 , color = colors,  xmax =
    ↳temp_df['price_difference'], linewidth = 1)
for x, y in zip(temp_df['price_difference'], temp_df['name']):
    c = None
    if x < 0:
        c = 'red'
    elif x > 0:
        c = 'green'
    else:
        c = 'goldenrod'
    ax.text(x - 15000 if x < 0 else x + 15000,
            y,
            round(x, 2),
            color = c,
            horizontalalignment='right' if x < 0 else 'left',
            size = 10)
    ax.scatter(x,
               y,
               color = c,
               alpha = 0.5)
ax.set_title("Price Difference in Advertising Posts Diverging plots")
ax.set_xlim(-2_300_000)
ax.set_xlabel("Price Difference")
ax.set_ylabel("Page")
ax.grid(linestyle='--', alpha=0.5)
ax.set_yticks(temp_df.index)
ax.spines["top"].set_color("None")
ax.spines["left"].set_color("None")
ax.spines['right'].set_position(('data',0))
ax.spines['right'].set_color('black')
plt.show()
```



```
[63]: temp_df = ad_story.sort_values('price_difference')
temp_df.reset_index(inplace = True)
colors = []
for x in temp_df['price_difference']:
    if x < 0:
        colors.append('red')
    elif x > 0:
        colors.append('green')
    else:
        colors.append('goldenrod')

fig = plt.figure(figsize = (15, 10))
ax = fig.add_subplot()
ax.hlines(y = temp_df.index, xmin = 0 , color = colors, xmax =
    ↪temp_df['price_difference'], linewidth = 1)
for x, y in zip(temp_df['price_difference'], temp_df['name']):
    c = None
    if x < 0:
        c = 'red'
    elif x > 0:
        c = 'green'
    else:
        c = 'goldenrod'
    ax.text(x - 10000 if x < 0 else x + 10000,
```

```

        y,
        round(x, 2),
        color = c,
        horizontalalignment='right' if x < 0 else 'left',
        size = 10)
ax.scatter(x,
           y,
           color = c,
           alpha = 0.5)
ax.set_title("Price Difference in Advertising Stories Diverging plots")
ax.set_xlim(-1_300_000)
ax.set_xlabel("Price Difference")
ax.set_ylabel("Page")
ax.grid(linestyle='--', alpha=0.5)
ax.set_yticks(temp_df.index)
ax.spines["top"].set_color("None")
ax.spines["left"].set_color("None")
ax.spines['right'].set_position(('data',0))
ax.spines['right'].set_color('black')
plt.show()

```



```

[64]: x1 = influencer['lowest_cost_per_view']
      x2 = influencer['highest_cost_per_view']
      y = influencer['influ_name']
      z = influencer['benefit']

```

```

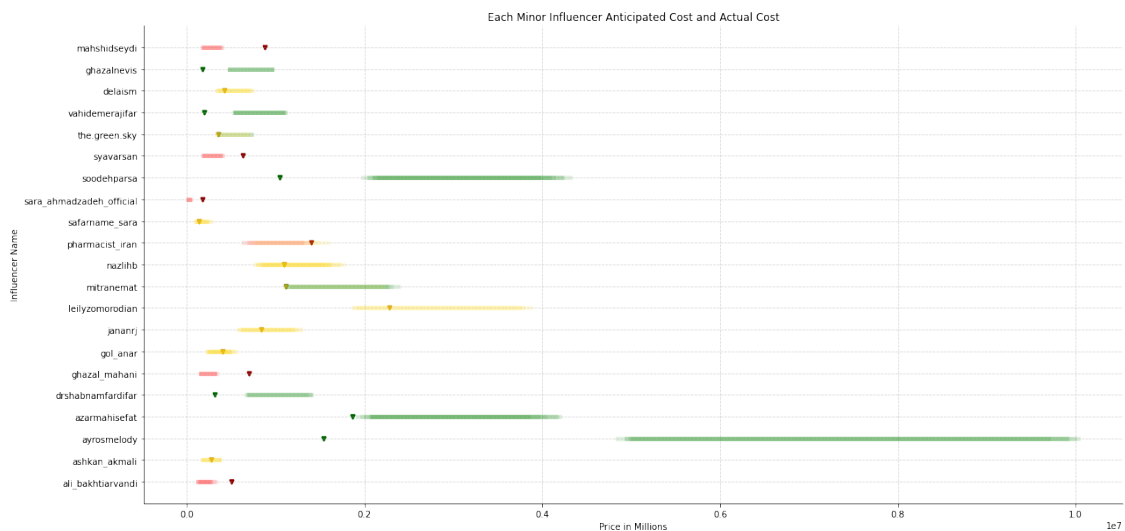
c = influencer['cost']

fig = plt.figure(figsize = (20, 10))
ax = fig.add_subplot()

for x1_, x2_, y_, z_, c_ in zip(x1, x2, y, z, c):
    ax.plot([int(x1_), int(x2_)], [y_, y_], color = "red" if z_ == -1 else
    ↪ "green" if z_ == 1 else 'gold', linewidth=5, alpha=.11)
    ax.scatter(c_, y_, s=20, marker='v', c="darkred" if z_ == -1 else
    ↪ "darkgreen" if z_ == 1 else 'goldenrod')

ax.grid(linestyle='--', alpha=0.5)
ax.set_title('Each Minor Influencer Anticipated Cost and Actual Cost')
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylabel('Influencer Name')
ax.set_xlabel('Price in Millions')
plt.show()

```



In the Graph above you can see each minor influencer lowest and highest anticipated cost as a bar and their actual cost as triangle. with quick glimpse we can deduce that: - The distance between highest anticipated cost and actual cost for not benefitted influencers are not very far, the most over paid influencer is “mahshidseydi”. - the distance between lowest anticipated cost and actual cost for benefitted influencers are far and thats good sign, the most under paid influencers are “ayrosmelody” and in second place is “azarmahisefat”.

```

[65]: x1 = leaders_post['lowest_cost_per_view']
      x2 = leaders_post['highest_cost_per_view']
      y = leaders_post['name']

```

```

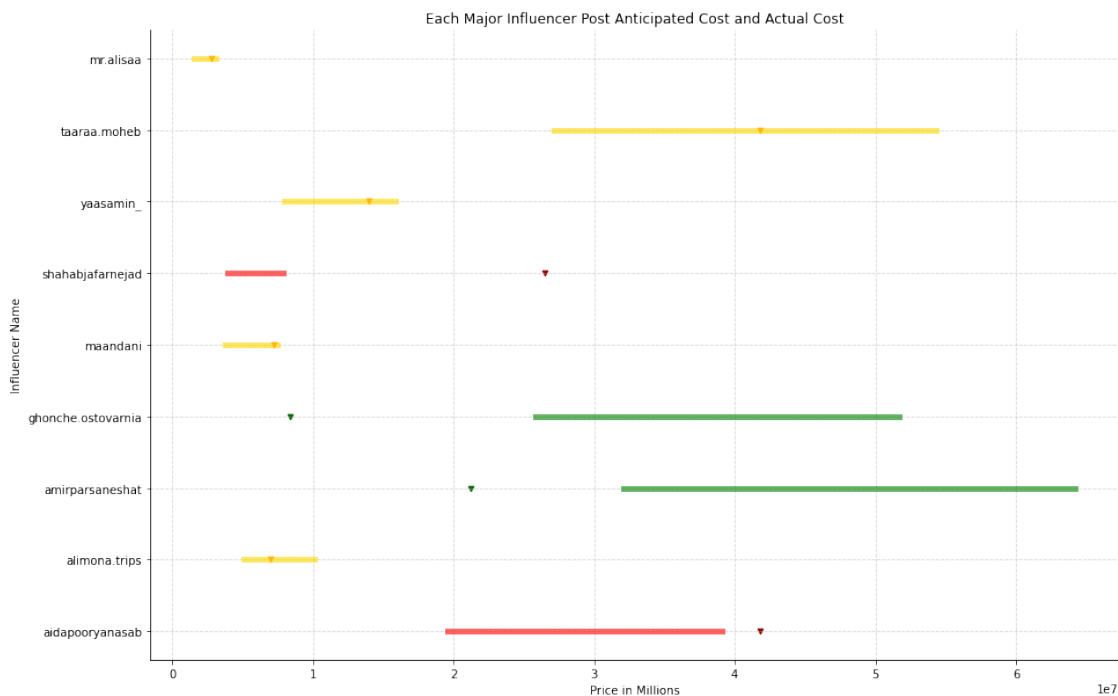
z = leaders_post['benefit']
c = leaders_post['cost']

fig = plt.figure(figsize = (15, 10))
ax = fig.add_subplot()

for x1_, x2_, y_, z_, c_ in zip(x1, x2, y, z, c):
    ax.plot([int(x1_), int(x2_)], [y_, y_], color = "red" if z_ == -1 else
    ↪ "green" if z_ == 1 else 'gold', linewidth=5, alpha=.6)
    ax.scatter(c_, y_, s=20, marker='v', c="darkred" if z_ == -1 else
    ↪ "darkgreen" if z_ == 1 else 'darkorange')

ax.grid(linestyle='--', alpha=0.5)
ax.set_title('Each Major Influencer Post Anticipated Cost and Actual Cost')
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylabel('Influencer Name')
ax.set_xlabel('Price in Millions')
plt.show()

```



In the Graph above you can see each major influencer lowest and highest anticipated cost as a bar and their actual cost as triangle. with quick glimpse we can deduce that: - There are only 1 influencer which was overpaid, the distance between its cost and highest anticipated value are high. It's advised to review the price and further project with "shahabjafarnejad". - There are 2 influencer



which was underpaid and the distance between their actual cost and lowest anticipated value are far and that's a good sign. These influencers are "amirparsaneshat" and "ghonche.ostovarnia". - The 2 underpaid influencer are the main reason that this approach was benefitted for the agency.

```
[66]: ad_post.drop(columns = ['ad_post_no', 'threshold']).groupby('benefit').mean()
```

```
[66]:
```

	follower	view	cost	cost_per_view	price_difference
benefit					
0	1835166.67	20595.75	2142446.21	1647660.00	-494786.21
1	939933.33	17097.87	952709.70	1367829.33	415119.63

In the cell above you can see the advertising post media grouped by their benefit status, based on that information we can deduce that: - benefit media had less followers but they actually brought more views in contrast of non-benefit media. - price difference between benefit and non-benefit media are significant. - high follower media tend to charge more but their view amounts are not correlated with their followers and thats a sign of fake followers.

```
[67]: ad_story.drop(columns = ['ad_story_no', 'threshold']).groupby('benefit').mean()
```

```
[67]:
```

	view	follower	action	interaction	impression	cost \
benefit						
0	56913.09	1685909.09	288.82	198.55	67125.64	1571555.20
1	54361.31	1502250.00	155.75	227.12	44548.00	857055.80

	cost_per_view	price_difference
benefit		
0	1138261.82	-433293.38
1	1087226.25	230170.45

In the cell above you can see the advertising story media grouped by their benefit status, based on that information we can deduce that: - the difference between the mean value of benefit and non-benefit media followers are 100k. - although the non-benefit media got more impressions that benefit ones, benefit media got more views, almost 33% more. - the difference between the prices are not very significant.

```
[68]: influencer.drop(columns = ['story_no']).groupby('benefit').mean()
```

```
[68]:
```

	l_threshold	h_threshold	follower	view	action	impression	cta \
benefit							
-1	40.00	80.00	93260.87	4602.00	80.04	4840.30	0.70
0	40.00	80.00	152220.93	13321.33	134.98	13821.35	0.58
1	40.00	80.00	279788.89	43098.25	509.08	44997.17	0.58

	interaction	cost	lowest_cost_per_view	highest_cost_per_view
benefit				
-1	63.26	642921.32	184080.00	368160.00
0	112.37	707241.50	532853.02	1065706.05
1	345.78	922261.80	1723930.00	3447860.00

In the cell above you can see the minor influencers grouped by their benefit status, based on that information we can deduce that: - more followers in minor influencers means the higher chance of being benefitted. this fact can be interpreted as selected influencers had almost no fake followers and their view counts are organic. - high follower influencers got more action percentage regarding their story than low followers influencers. This means that followers of high follower influencers engage more with their story. this fact should be in mind when proposing action-based campaign to customers.

```
[69]: leaders_post.drop(columns = ['post_no', 'l_threshold', 'h_threshold']).
      ↳groupby('benefit').mean()
```

```
[69]:
```

	follower	view	like	comment	share	save	profile_visit	\
benefit								
-1	412500.00	46919.00	10133.00	120.00	160.50	172.50	749.50	
0	240280.00	36335.80	5865.80	133.60	211.40	188.80	499.20	
1	134000.00	115862.50	19120.00	653.00	3996.50	3839.00	4602.00	

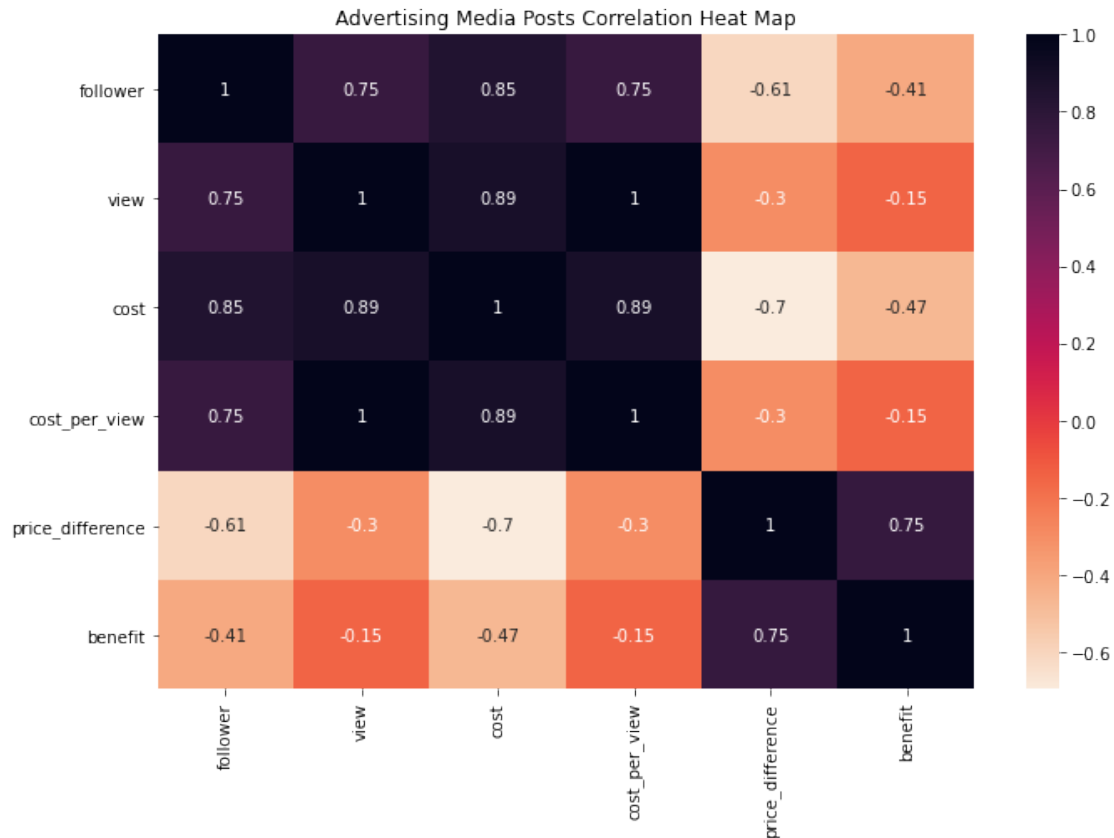
	reach	impression	cost	lowest_cost_per_view	\
benefit					
-1	91193.00	99681.00	34147875.82	11729750.00	
0	50049.60	64881.20	14551176.47	9083950.00	
1	151312.50	176229.00	14774183.01	28965625.00	

	highest_cost_per_view
benefit	
-1	23459500.00
0	18167900.00
1	57931250.00

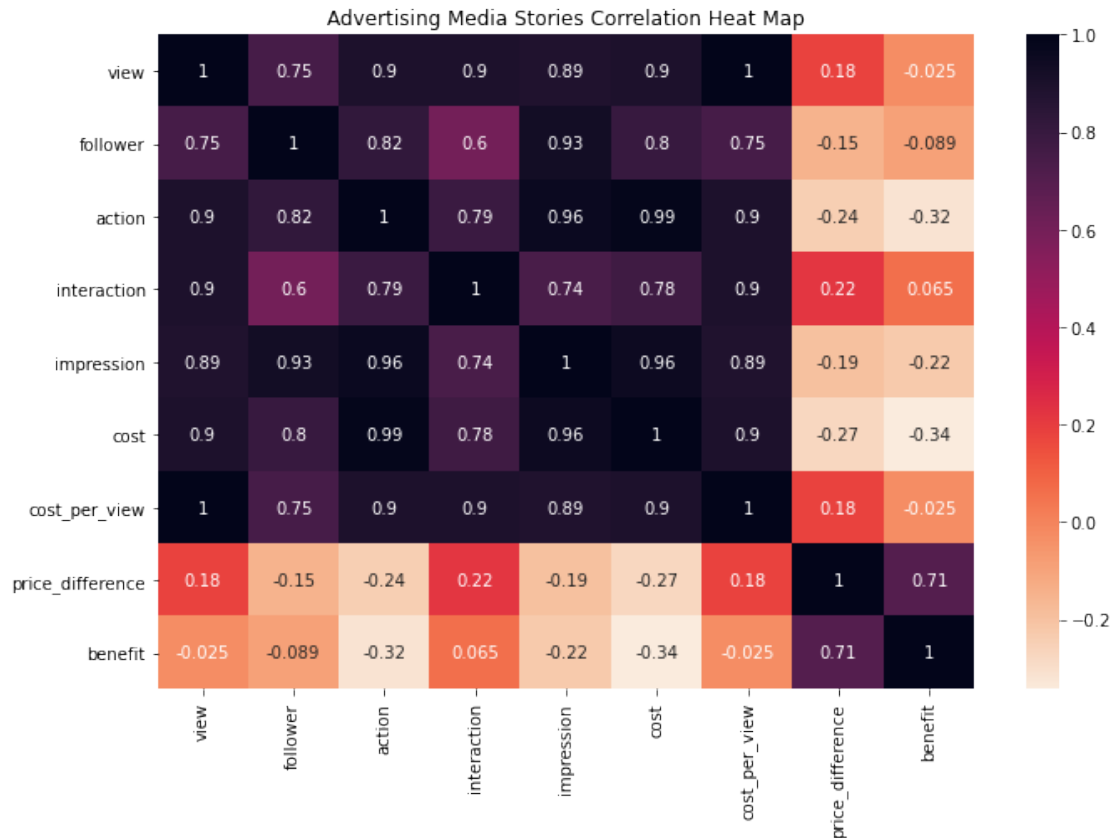
In the cell above you can see the major influencers grouped by their benefit status, based on that information we can deduce that: - the deciding factor regarding the benefit are view, thus performance metric which are correlated with view are important. we can vaguely see this effect in benefit and neutral media. - as you can see benefit and neutral media are rich in performance metrics. - as we said earlier, the only major influencer which was not benefit and overpaid is “shahabjafarnejad”.

```
[70]: intercor = ad_post.drop(columns = ['ad_post_no', 'threshold']).corr()
      plt.figure(figsize=(10,7))
      sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
      plt.tight_layout()
      plt.title('Advertising Media Posts Correlation Heat Map')
      plt.show()
```



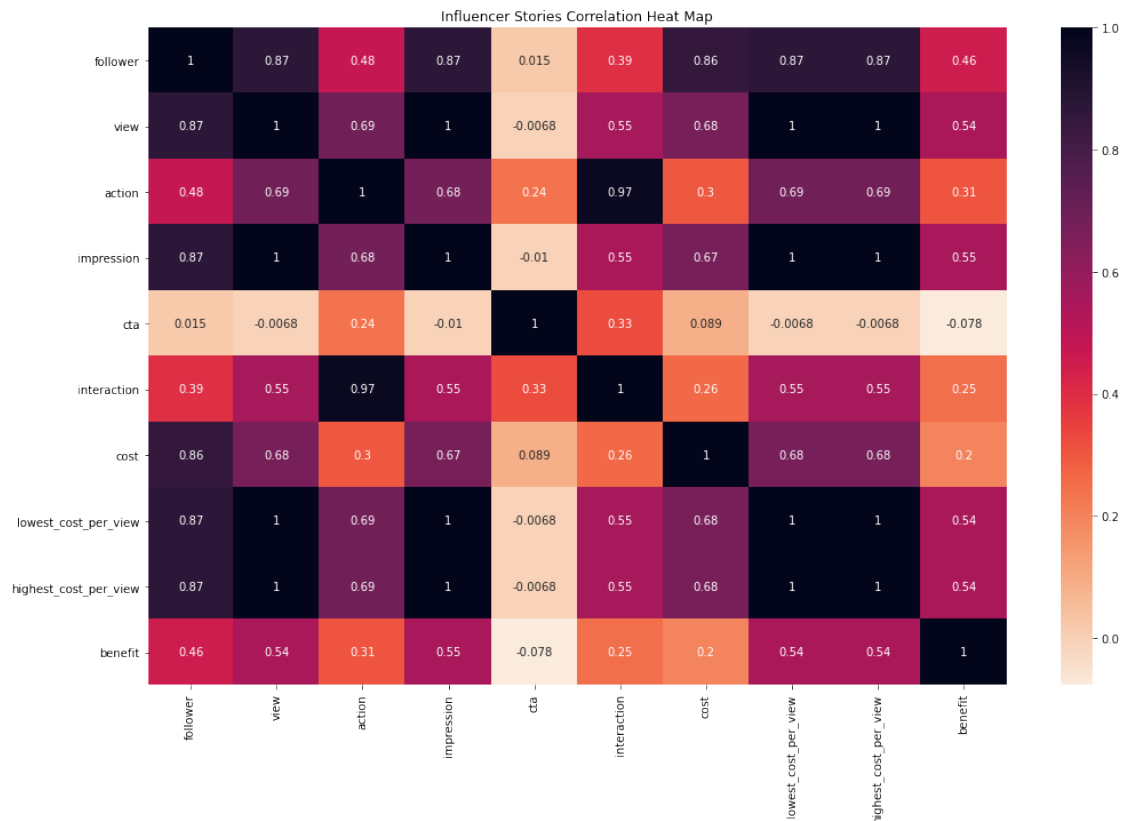
In the Graph above you can see the feature correlation heatmap for advertising media posts, based on that there are some worth mentioning insights: - the strongest correlation is between “cost per view” and “view”, it’s obvious since cost per view is calculated by view. - second strongest correlation are for “cost per view” and “cost” and “cost” and “view”. since view is our main performance metric and cost is a important feature we are trying to optimize. - the correlation between cost and follower are more than view and follower. this means that in order to make our media optimized cost-wise we must focus on view more than follower.

```
[71]: intercor = ad_story.drop(columns = ['ad_story_no', 'threshold']).corr()
plt.figure(figsize=(10,7))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Advertising Media Stories Correlation Heat Map')
plt.show()
```



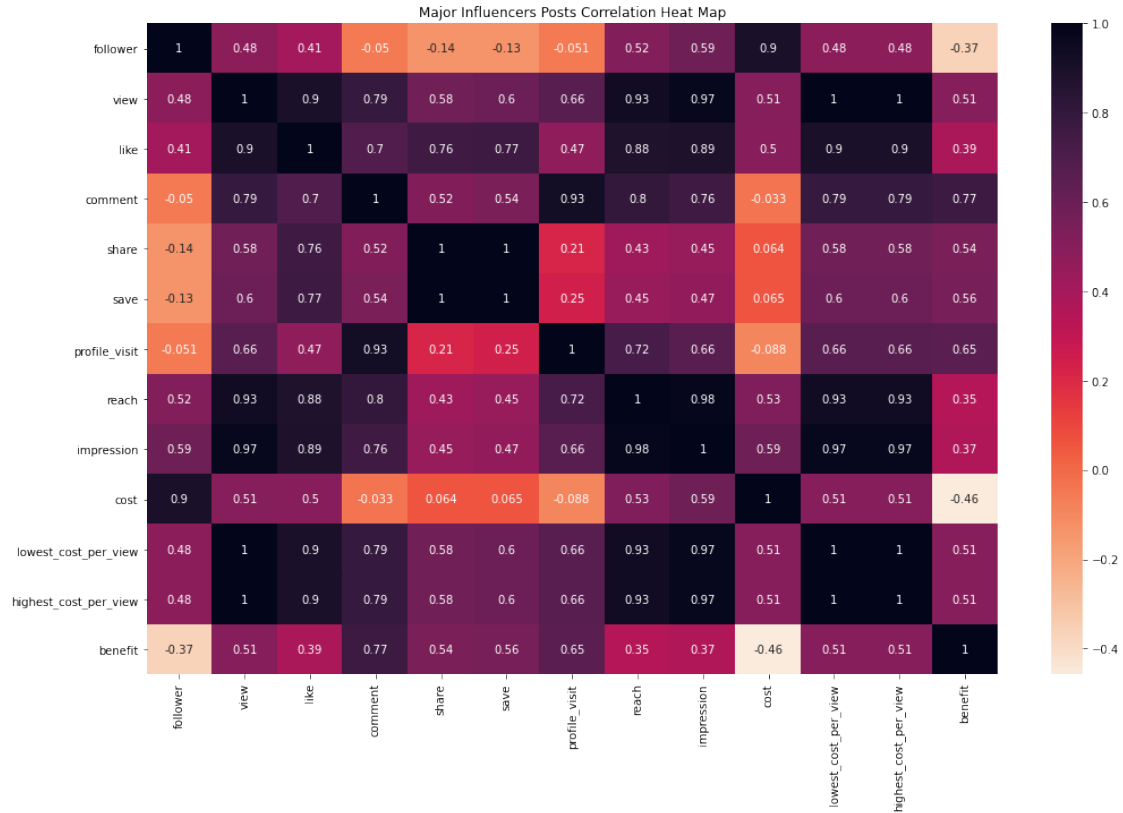
In the graph above you can see the feature correlation heatmap for advertising media stories, some interesting insights: - view is strongly correlated with action, interaction, impression and importantly, cost and cost per view. - although view and follower are correlated positively, their relationship strength is less than forementioned features. - follower and impression are very strongly correlated in stories. - although action and impression are strongly correlated with cost, interaction are less correlated. this means that other actions except sticker tap are far more important for a story to be estimated costly beneficial. - follower and interaction are not correlated very strongly. this suggest that the increase of followers are not linearly affect interaction quantity, so if we are performing a interaction-based campaign, it's wise to consider medium and low media since their followers are interacting partially more.

```
[72]: intercor = influencer.drop(columns = ['story_no', 'l_threshold', 'h_threshold']).corr()
plt.figure(figsize=(15,10))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Influencer Stories Correlation Heat Map')
plt.show()
```



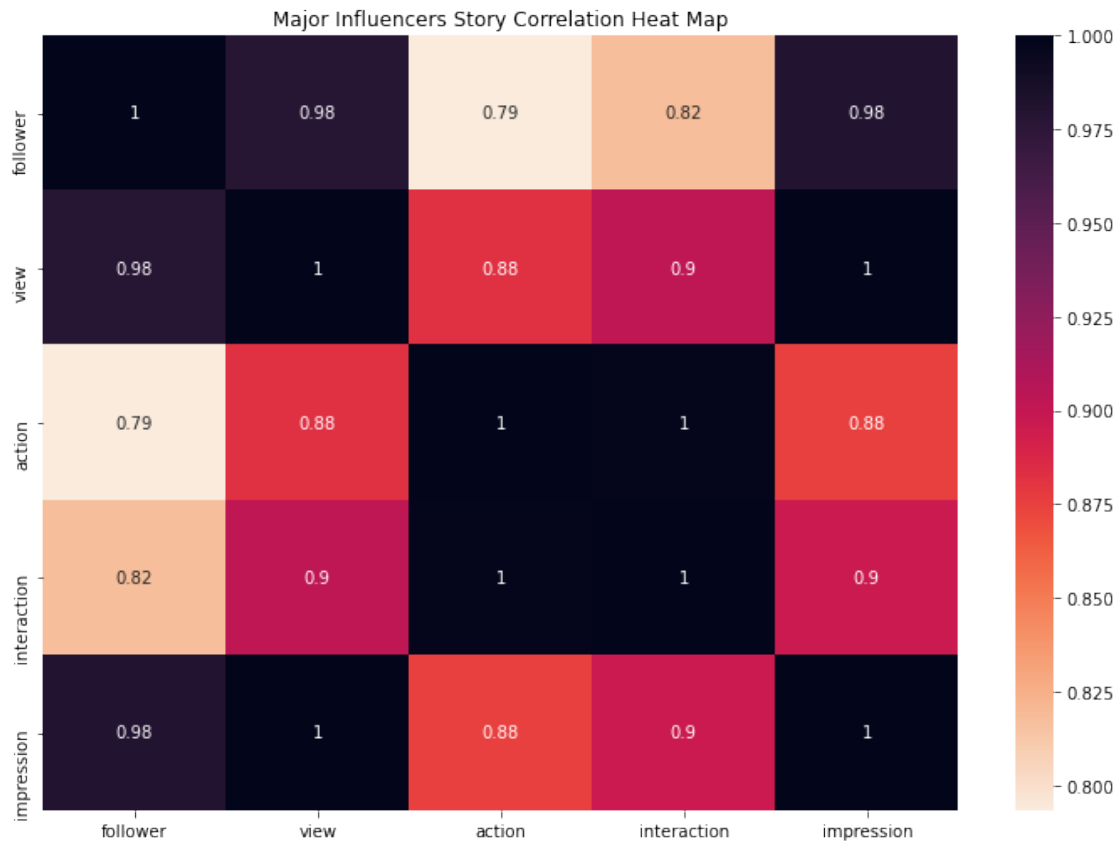
In the cell above you can see the features correlation heatmap of minor influencers. almost the same insight as advertising stories can be deduced from this heatmap.

```
[73]: intercor = leaders_post.drop(columns = ['post_no', 'l_threshold',
      ↳ 'h_threshold']).corr()
plt.figure(figsize=(15,10))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Major Influencers Posts Correlation Heat Map')
plt.show()
```



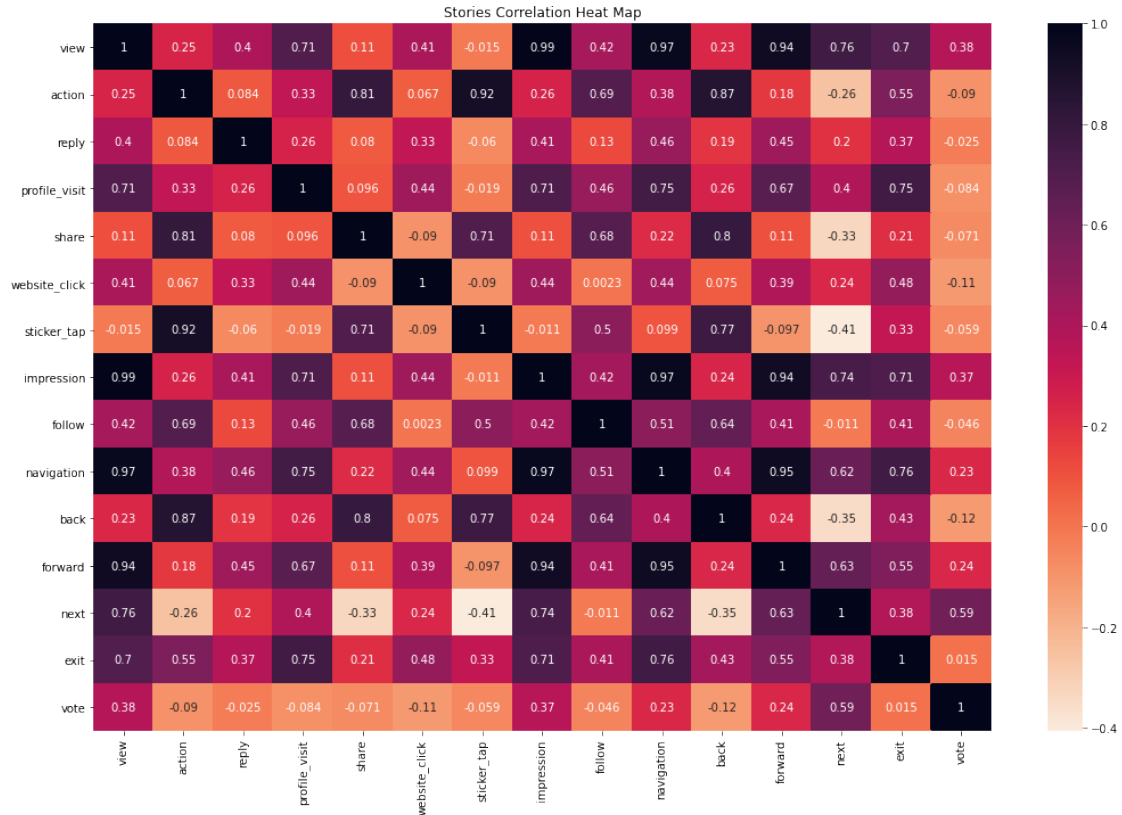
In the cell above you can see the feature correlation heatmap for Major influencers advertising posts. some interesting insights from this graph are: - there is strong positive between share and save. this could be interpreted as almost everyone who shared their post, also saved their post too. - major influencers cost are strongly correlated to their quantity of followers and far less dependent to their view. this means that we should be looking precisely to their view count when we are selecting influencers, not their followers. - in video type contents, there are strong correlation between view and like. this can be interpreted as whoever watches a video in influencers page, like that video too. - follower correlation with comment, share, save and profile visit are negative. this can be interpreted as the more follower an influencer get, the less engagement he/she will get from their follower. also this can be a sign of a passive/fake followers for influencers.

```
[74]: intercor = leaders_story.drop(columns = ['story_no', 'cost']).corr()
plt.figure(figsize=(10,7))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Major Influencers Story Correlation Heat Map')
plt.show()
```



In the cell above you can see the correlation heatmap for Major influencers advertising story contents, some interesting insights from this data are: - view and follower are strongly correlated, this means that almost the good amount of major influencers followers watch their stories. also this fact should be taken in mind when the goal of a campaign is awareness. - action and follower are mediocore strength-wise. this means that followers engage with influencers content type, but when proposing action-based campaign should be take in mind that it's probably need a lot of influencers.

```
[75]: intercor = story.drop(columns = ['story_no']).corr()
plt.figure(figsize=(15,10))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Stories Correlation Heat Map')
plt.show()
```

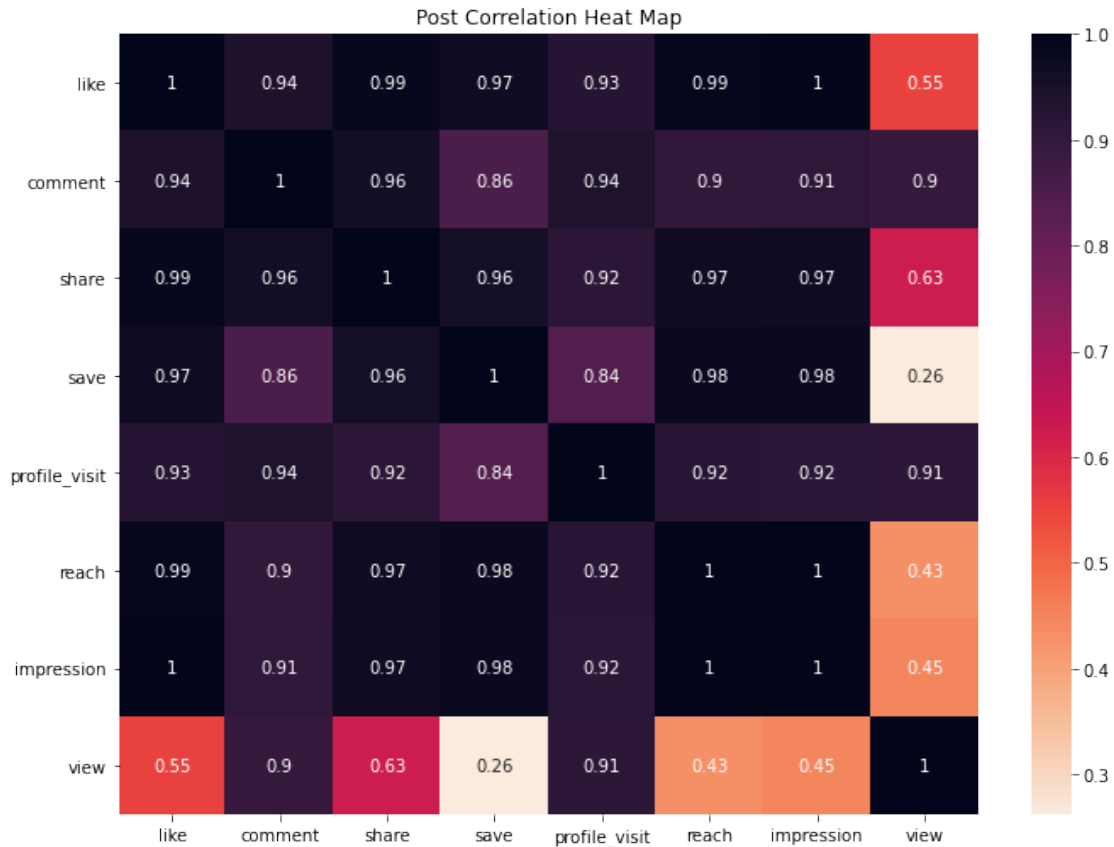


In the cell above you can see the feature correlation heatmap for stories, some interesting insights are:

- majority of actions in instagram stories are sticker taps. this means that putting tappable stickers in stories always attract the majority of actions for a story.
- on the other hand, correlation between action and view are at 0.25. this indicates that people are not very interacting with stories if we are using this approach.
- as you can see influencers' followers have more action with influencers' stories than campaign page stories. we must take follower quantity in mind but generally when we are proposing action-based campaigns, it's better to invest in influencers.
- majority quantity of navigation comes from forward and in the second plance, exit.
- people who vote in instagram stories are more likely to push to next story than just wait for story time to finish.

```
[76]: intercor = post.drop(columns = ['post_no', 'ig_tv']).corr()
plt.figure(figsize=(10,7))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Post Correlation Heat Map')
plt.show()
```





In the cell above you can see the correlation heatmap for posts, since we have just 12 posts is not very accurate, but it will be worthy to have a glimpse at the result.

```
[77]: ad_post.drop(columns = ['ad_post_no', 'threshold']).groupby('field').mean()
```

```
[77]:
```

	follower	view	cost	cost_per_view	price_difference \
field					
art & culture	418636.36	13519.45	863687.95	1081556.36	217868.41
fact	2210000.00	26001.70	2260160.72	2080136.00	-180024.72
video	1300000.00	10997.33	1288689.45	879786.67	-408902.78
women	1838666.67	20631.33	1343919.00	1650506.67	306587.67

```
benefit
```

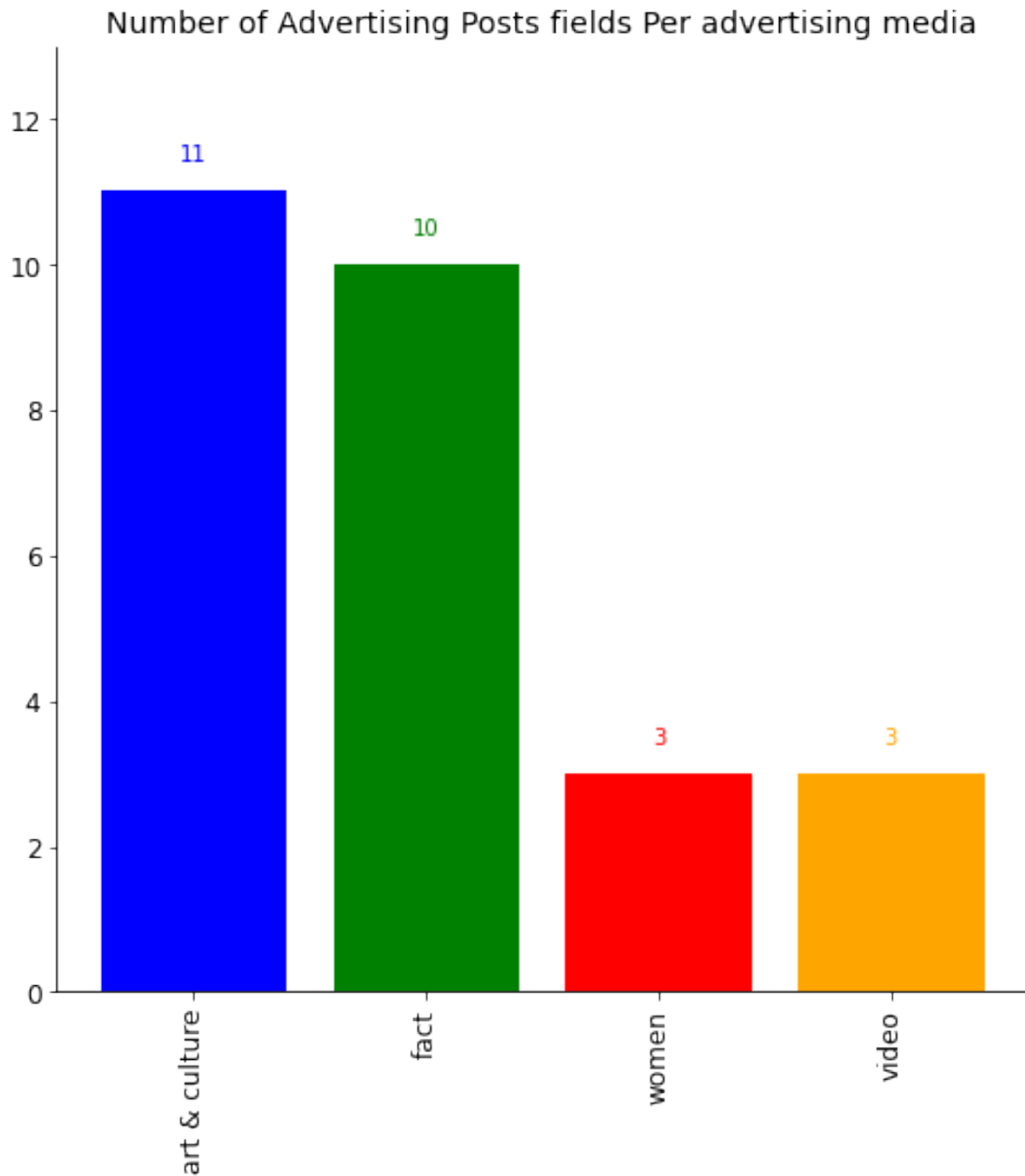
field	benefit
art & culture	0.73
fact	0.50
video	0.00
women	0.67

The table above is the mean of features grouped by field in advertising posts. we can deduce from that information: - fact media has most followers and in the second place women field. - although

fact media cost twice as much women field, their view difference are not significant. - although video field has significant followers, but their view are fairly low, this could be interpreted as fake/passive followers.

```
[78]: d = ad_post['field'].value_counts().to_dict()
colors = ['blue', 'green', 'red', 'orange']
fig = plt.figure(figsize = (8, 8))
ax = fig.add_subplot()
ax.bar(d.keys(), d.values(), color = colors)
for i, (k, v) in enumerate(d.items()):
    ax.text(k,
            v + .5,
            v,
            color = colors[i],
            fontsize = 10,
            horizontalalignment = 'center',
            verticalalignment = 'center')
ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
ax.tick_params(axis = 'y', labelsize = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 13)
ax.set_title("Number of Advertising Posts fields Per advertising media",
            ↪ fontsize = 14);
plt.show()

total = sum(ad_post['field'].value_counts())
print('the top 3 field in advertising posts and their percentages are:')
print(f'1. "{list(d.keys())[0]}": {(ad_post["field"].value_counts()[0]) / total ↪
    ↪ * 100} %')
print(f'2. "{list(d.keys())[1]}": {(ad_post["field"].value_counts()[1]) / total ↪
    ↪ * 100} %')
print(f'3. "{list(d.keys())[2]}": {(ad_post["field"].value_counts()[2]) / total ↪
    ↪ * 100} %')
```



the top 3 field in advertising posts and their percentages are:

1. "art & culture": 40.74074074074074 %
2. "fact": 37.03703703703704 %
3. "women": 11.11111111111111 %

```
[79]: ad_story.drop(columns = ['ad_story_no', 'threshold']).groupby('field').mean()
```

```
[79]:
```

field	view	follower	action	interaction	impression	cost	\
-------	------	----------	--------	-------------	------------	------	---

art & culture	52953.50	1029000.00	195.67	238.00	36994.50	1046636.38
fact	87894.60	2260500.00	299.30	325.60	83649.20	1623068.91
health	16256.25	1050750.00	75.25	72.25	18878.00	463991.27
news	58990.00	877000.00	234.00	90.00	58568.00	1302915.56
video	51652.00	1350000.00	165.50	255.50	42506.00	924365.77
women	17959.75	1505500.00	159.00	61.00	43399.75	870470.11

	cost_per_view	price_difference	benefit
field			
art & culture	1059070.00	12433.62	0.83
fact	1757892.00	134823.09	0.80
health	325125.00	-138866.27	0.50
news	1179800.00	-123115.56	0.00
video	1033040.00	108674.23	0.50
women	359195.00	-511275.11	0.00

The table above is the mean of features grouped by field in advertising stories. some interesting facts from this table is: - fact category got more followers and view than other categories. - although video category is not top 3 in view, but it got significant amount of interactions. this means that this type of medium is good for action-based campaigns. - news category despite being with the least follower among other categories, it got more view than other type of media except fact.

```
[80]: d = ad_story['field'].value_counts().to_dict()
colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
fig = plt.figure(figsize = (8, 8))
ax = fig.add_subplot()
ax.bar(d.keys(), d.values(), color = colors)
for i, (k, v) in enumerate(d.items()):
    ax.text(k,
            v + .5,
            v,
            color = colors[i],
            fontsize = 10,
            horizontalalignment = 'center',
            verticalalignment = 'center')
ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
ax.tick_params(axis = 'y', labelsize = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 12)
ax.set_title("Number of Advertising Stories fields Per advertising media",
             ↪ fontsize = 14);
plt.show()

total = sum(ad_story['field'].value_counts())
print('the top 3 fields in advertising stories and their percentages are:')
```

```

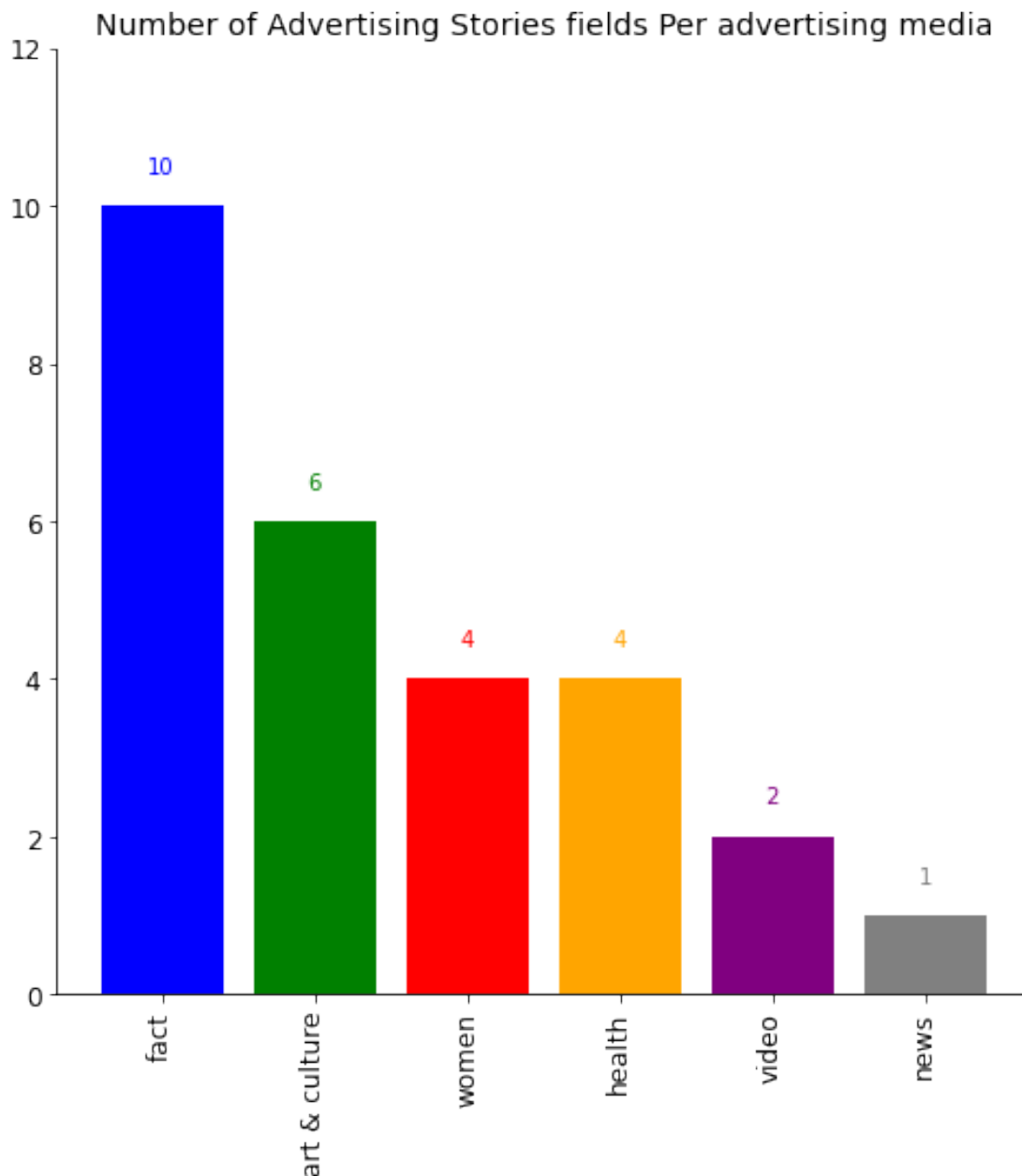
print(f'1. "{list(d.keys())[0]}": {(ad_story["field"].value_counts()[0]) /  

↳total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(ad_story["field"].value_counts()[1]) /  

↳total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(ad_story["field"].value_counts()[2]) /  

↳total * 100} %')

```



the top 3 fields in advertising stories and their percentages are:

1. "fact": 37.03703703703704 %

2. "art & culture": 22.22222222222222 %
3. "women": 14.814814814814813 %

```
[81]: influencer.drop(columns = ['story_no', 'l_threshold', 'h_threshold']).
      ↪groupby('field').mean()
```

```
[81]:
```

	follower	view	action	impression	cta	interaction	cost \
field							
cooking	491000.00	50626.50	293.33	51752.50	0.67	244.00	1860023.49
health	123366.67	15702.00	262.08	16774.50	0.58	178.42	627758.09
lifestyle	206315.38	25158.05	286.02	26221.20	0.66	206.89	837010.67
sport	60000.00	2184.38	59.00	2524.38	0.38	37.88	435943.11
tourism	40545.45	7751.73	182.18	8105.55	0.45	125.36	164865.76

	lowest_cost_per_view	highest_cost_per_view	benefit
field			
cooking	2025060.00	4050120.00	1.00
health	628080.00	1256160.00	0.50
lifestyle	1006321.85	2012643.69	0.06
sport	87375.00	174750.00	-1.00
tourism	310069.09	620138.18	0.45

The table above is the mean of features grouped by field in minor influencers advertising, interesting insights are listed as below: - lifestyle category despite having less than half o cooking category followers, it go almost the same amount of action and interaction. important thing to remember when designing action-based campaigns. - the best performing category is for cooking. please have in mind that we only had 1 influencer in this category. - sport category despite having not the least amount of follower, but this category performed worst. please have in mind that we only had 1 influencer in this category.

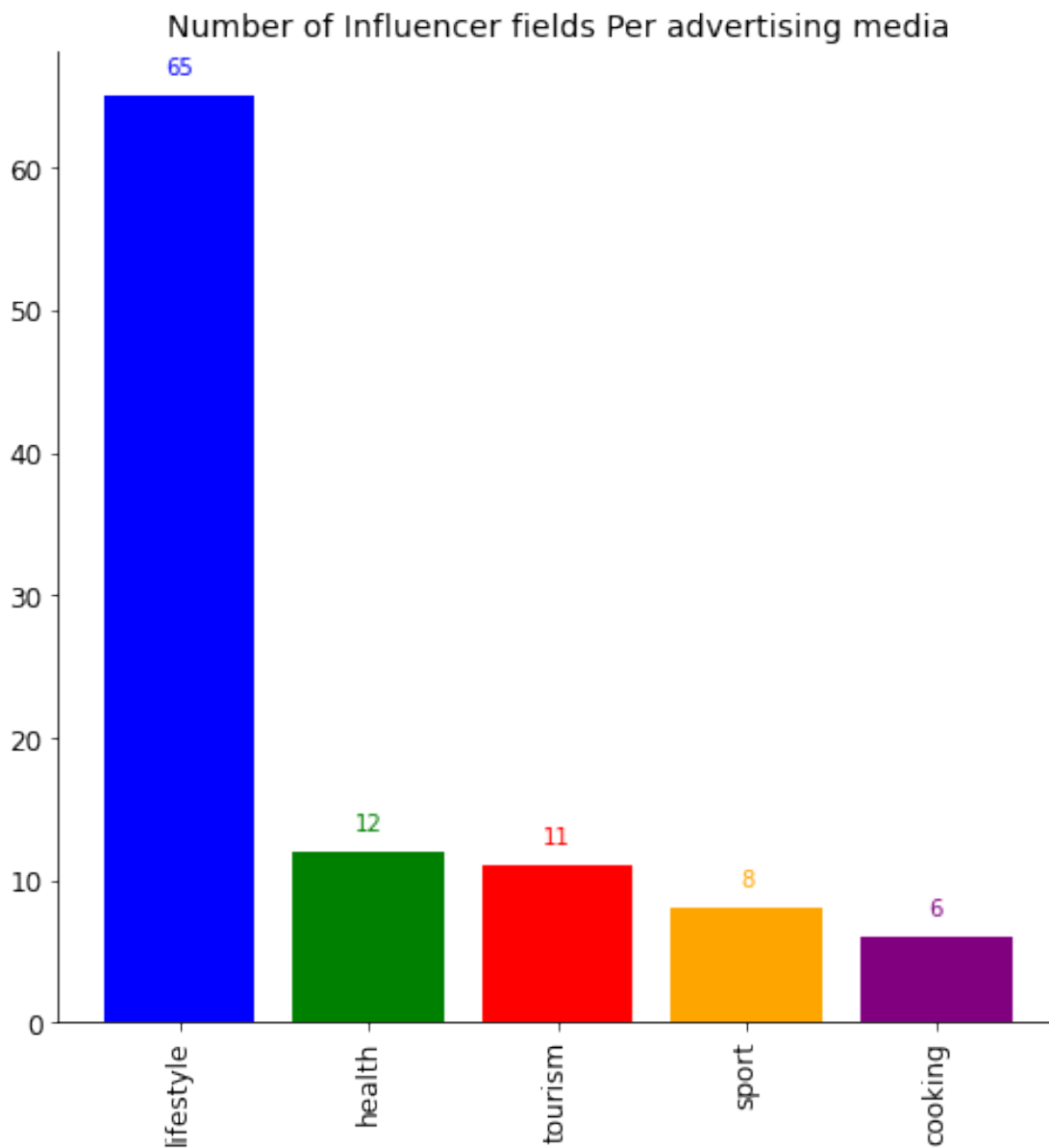
```
[82]: d = influencer['field'].value_counts().to_dict()
      colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
      fig = plt.figure(figsize = (8, 8))
      ax = fig.add_subplot()
      ax.bar(d.keys(), d.values(), color = colors)
      for i, (k, v) in enumerate(d.items()):
          ax.text(k,
                  v + 2,
                  v,
                  color = colors[i],
                  fontsize = 10,
                  horizontalalignment = 'center',
                  verticalalignment = 'center')
      ax.tick_params(axis = 'x', labelrotation = 90, labelsz = 12)
      ax.tick_params(axis = 'y', labelsz = 12)
      ax.spines["top"].set_color("None")
      ax.spines["right"].set_color("None")
      # ax.set_ylim(0, 70)
```

```

ax.set_title("Number of Influencer fields Per advertising media", fontsize = 14);
plt.show()

total = sum(influencer['field'].value_counts())
print('the top 3 fields in minor influencers and their percentages are:')
print(f'1. "{list(d.keys())[0]}": {(influencer["field"].value_counts()[0]) / total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(influencer["field"].value_counts()[1]) / total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(influencer["field"].value_counts()[2]) / total * 100} %')

```



the top 3 fields in minor influencers and their percentages are:

1. "lifestyle": 63.725490196078425 %
2. "health": 11.76470588235294 %
3. "tourism": 10.784313725490197 %

```
[83]: influencer.drop(columns = ['story_no', 'l_threshold', 'h_threshold']).
      ↪groupby('gender').mean()
```

```
[83]:
```

	follower	view	action	impression	cta	interaction	cost \
gender							
family	268130.43	32021.26	334.52	33309.26	0.63	231.57	961349.19
female	128848.94	15222.32	190.83	15877.64	0.55	140.60	644082.70
male	41444.44	4641.44	179.44	5229.78	0.78	163.89	434005.59

	lowest_cost_per_view	highest_cost_per_view	benefit
gender			
family	1280850.43	2561700.87	0.17
female	608892.77	1217785.53	0.19
male	185657.78	371315.56	-0.44

The table above is the mean of features grouped by gender in minor influencers advertising, interesting insights are listed as below: - best performing category is for family, in the second spot, females and in the last spot males. - male category despite of having less follower and views, got almost the same amount of action in contrast of female category, and more interaction than female category. - male category generally was not beneficial but the female and family category was generally beneficial at this campaign.

```
[84]: d = influencer['gender'].value_counts().to_dict()
      colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
      fig = plt.figure(figsize = (8, 8))
      ax = fig.add_subplot()
      ax.bar(d.keys(), d.values(), color = colors)
      for i, (k, v) in enumerate(d.items()):
          ax.text(k,
                  v + 2,
                  v,
                  color = colors[i],
                  fontsize = 10,
                  horizontalalignment = 'center',
                  verticalalignment = 'center')
      ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
      ax.tick_params(axis = 'y', labelsize = 12)
      ax.spines["top"].set_color("None")
      ax.spines["right"].set_color("None")
      ax.set_ylim(0, 50)
```

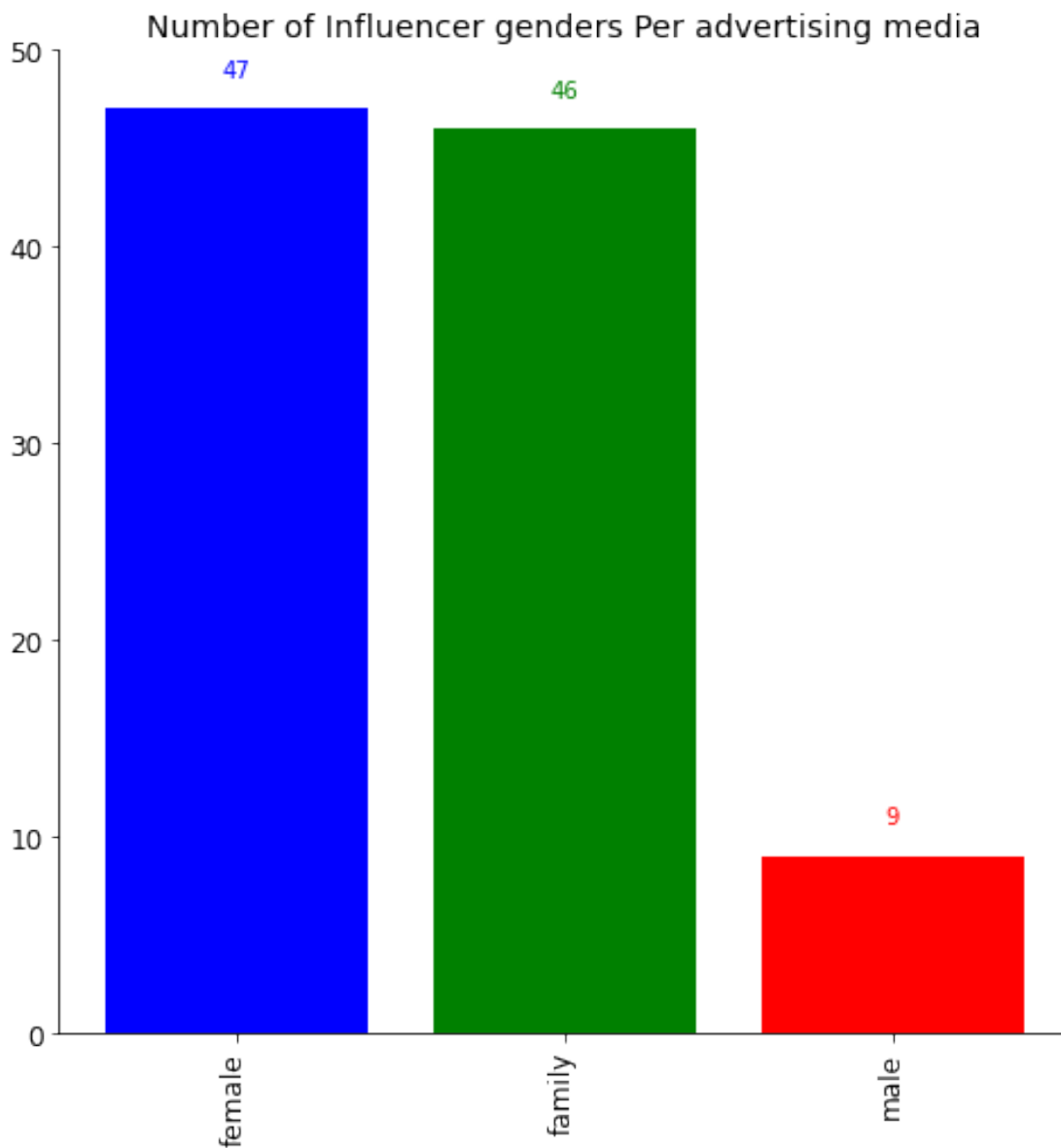


```

ax.set_title("Number of Influencer genders Per advertising media", fontsize = 14);
plt.show()

total = sum(influencer['gender'].value_counts())
print('the top 3 genders in minor influencers and their percentages are:')
print(f'1. "{list(d.keys())[0]}": {(influencer["gender"].value_counts()[0]) / total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(influencer["gender"].value_counts()[1]) / total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(influencer["gender"].value_counts()[2]) / total * 100} %')

```



the top 3 genders in minor influencers and their percentages are:

1. "female": 46.07843137254902 %
2. "family": 45.09803921568628 %
3. "male": 8.823529411764707 %

```
[85]: leaders_post.drop(columns = ['post_no', 'l_threshold', 'h_threshold']).
      ↳groupby('gender').mean()
```

```
[85]:
```

	follower	view	like	comment	share	save	profile_visit \
gender							
family	63700.00	13205.50	3150.00	90.50	126.50	81.00	263.50
female	440000.00	80437.50	12605.25	339.25	310.50	323.25	2368.00
male	135666.67	53027.00	10371.33	225.33	2625.33	2504.00	1066.67

	reach	impression	cost	lowest_cost_per_view \
gender				
family	19976.50	24013.00	4878267.97	3301375.00
female	122032.50	145265.50	26482026.14	20109375.00
male	69058.67	82379.33	18305119.83	13256750.00

	highest_cost_per_view	benefit
gender		
family	6602750.00	0
female	40218750.00	0
male	26513500.00	0

The table above is the mean of features grouped by gender in Major influencers advertising posts. interesting insights are listed below: - the best performing group of major influencers was female and in the second place male and in the last spot family. - male group despite being in second spot got a significant amount share and save in contrast of other categories. - the only group that the mean of their benefit status was positive, is female and the two other categories are neutral in benefit feature.

```
[86]: d = leaders_post['gender'].value_counts().to_dict()
      colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
      fig = plt.figure(figsize = (8, 8))
      ax = fig.add_subplot()
      ax.bar(d.keys(), d.values(), color = colors)
      for i, (k, v) in enumerate(d.items()):
          ax.text(k,
                  v + .4,
                  v,
                  color = colors[i],
                  fontsize = 10,
                  horizontalalignment = 'center',
                  verticalalignment = 'center')
```

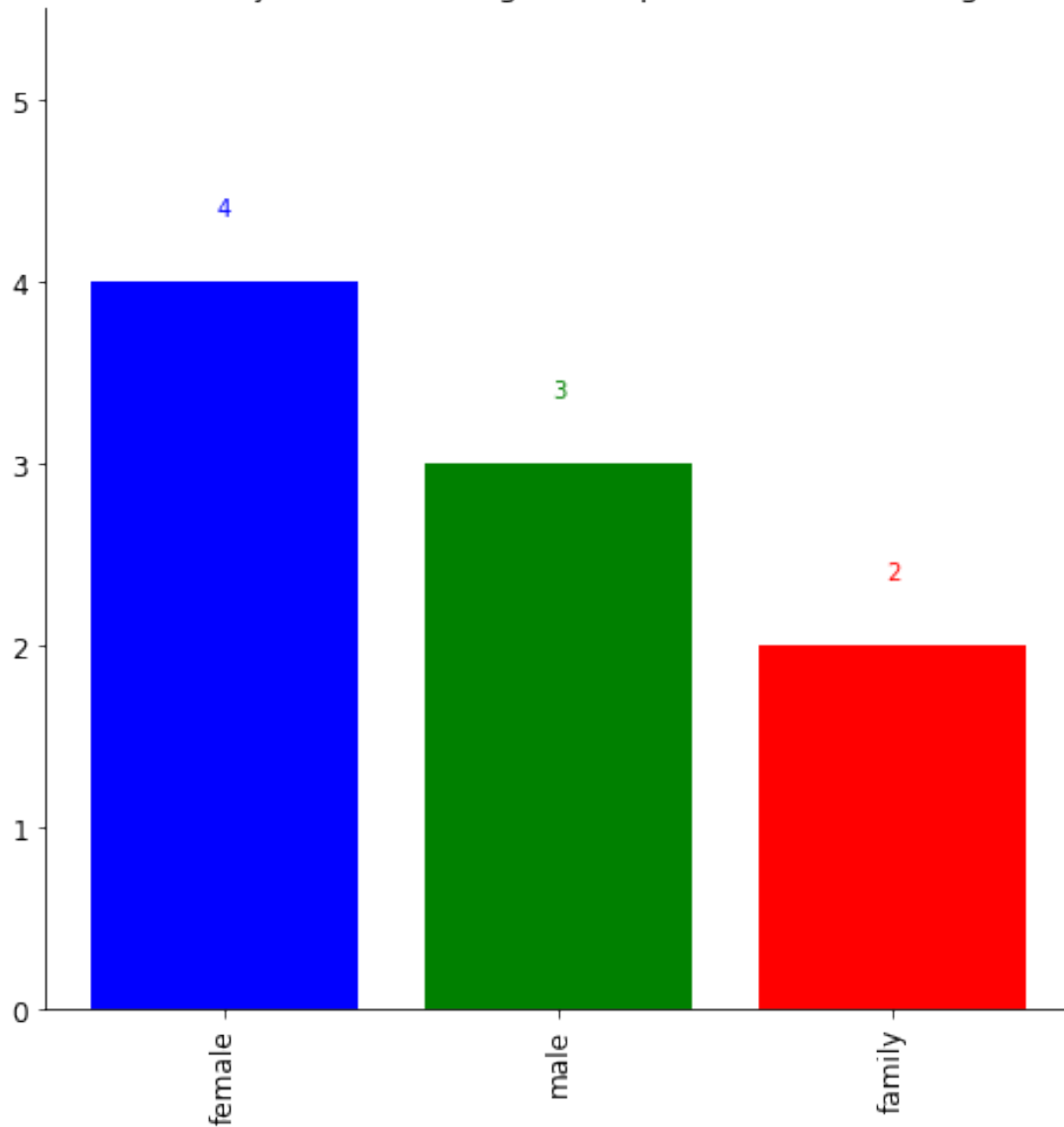
```

ax.tick_params(axis = 'x', labelrotation = 90, labelsizе = 12)
ax.tick_params(axis = 'y', labelsizе = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 5.5)
ax.set_title("Number of Major influencers genders posts Per advertising media",
    ↳fontsize = 14);
plt.show()

total = sum(leaders_post['gender'].value_counts())
print('the top 3 genders in major influencers advertising posts and their
    ↳percentages are:')
print(f'1. "{list(d.keys())[0]}": {(leaders_post["gender"].value_counts()[0]) /
    ↳total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(leaders_post["gender"].value_counts()[1]) /
    ↳total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(leaders_post["gender"].value_counts()[2]) /
    ↳total * 100} %')

```

Number of Major influencers genders posts Per advertising media



the top 3 genders in major influencers advertising posts and their percentages are:

1. "female": 44.44444444444444 %
2. "male": 33.33333333333333 %
3. "family": 22.22222222222222 %

```
[87]: leaders_story.drop(columns = ['story_no', 'cost']).groupby('gender').mean()
```

```
[87]:
```

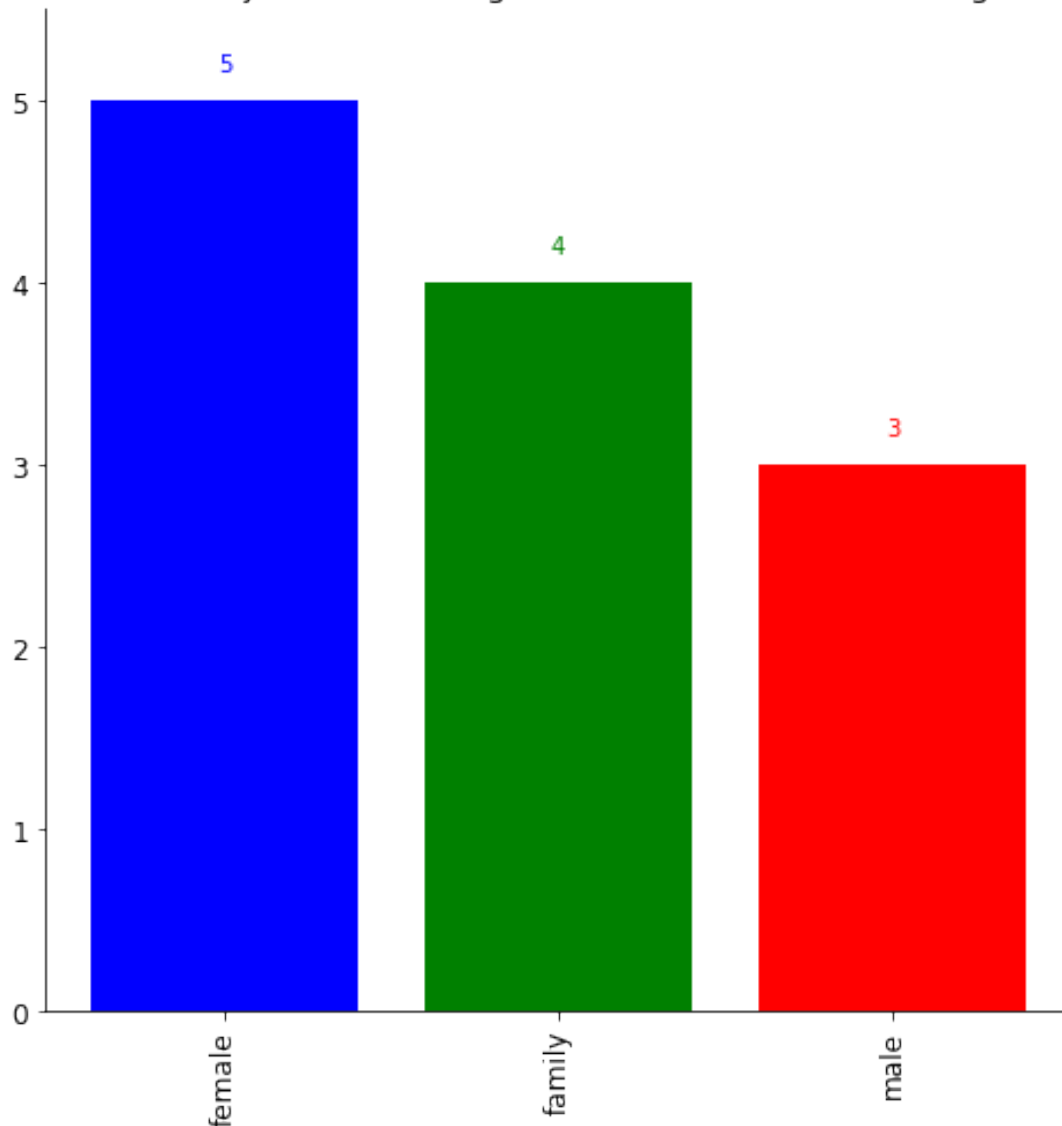
	follower	view	action	interaction	impression
gender					
family	58850.00	4306.50	115.50	91.25	3565.50

female	389800.00	57572.00	953.00	660.40	58489.80
male	135666.67	15091.33	277.67	186.67	15289.33

```
[88]: d = leaders_story['gender'].value_counts().to_dict()
colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
fig = plt.figure(figsize = (8, 8))
ax = fig.add_subplot()
ax.bar(d.keys(), d.values(), color = colors)
for i, (k, v) in enumerate(d.items()):
    ax.text(k,
            v + .2,
            v,
            color = colors[i],
            fontsize = 10,
            horizontalalignment = 'center',
            verticalalignment = 'center')
ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
ax.tick_params(axis = 'y', labelsize = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 5.5)
ax.set_title("Number of Major influencers genders stories Per advertising_
↳media", fontsize = 14);
plt.show()

total = sum(leaders_story['gender'].value_counts())
print('the top 3 genders in major influencers advertising posts and their_
↳percentages are:')
print(f'1. "{list(d.keys())[0]}": {(leaders_story["gender"].value_counts()[0]) /
↳ total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(leaders_story["gender"].value_counts()[1]) /
↳ total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(leaders_story["gender"].value_counts()[2]) /
↳ total * 100} %')
```

Number of Major influencers genders stories Per advertising media



the top 3 genders in major influencers advertising posts and their percentages are:

1. "female": 41.66666666666667 %
2. "family": 33.33333333333333 %
3. "male": 25.0 %

In the table above you can see the mean of features grouped by gender in major influencers advertising stories. interesting insights are listed below: - female category was the best performing category and in the second spot male and in the last spot family. - other performance metric features are fairly similar and anticipated.

```
[89]: story.drop(columns = ['story_no']).groupby('type').mean()
```

```
[89]:
```

	view	action	reply	profile_visit	share	website_click	\
type							
contest	807.80	128.20	2.80	15.80	18.60	0.00	
poll	1028.50	22.83	3.00	17.00	2.33	0.50	
share	768.79	29.93	3.28	23.10	3.28	0.28	

	sticker_tap	impression	follow	navigation	back	forward	next	\
type								
contest	91.00	842.80	0.80	1067.00	172.20	645.40	54.80	
poll	0.00	1052.00	0.33	1155.00	39.17	776.67	161.83	
share	0.00	800.00	0.41	933.76	42.83	620.90	85.86	

	exit	vote
type		
contest	204.00	0.00
poll	176.17	58.00
share	180.86	0.00

In the table above you can see the mean of features grouped by their type in campaign published stories. some interesting insights: - poll category got the most view and contest and share categories in the next spots. - contest type stories got much more action in contrast of other categories. - contest type stories shared much more than other type of stories.

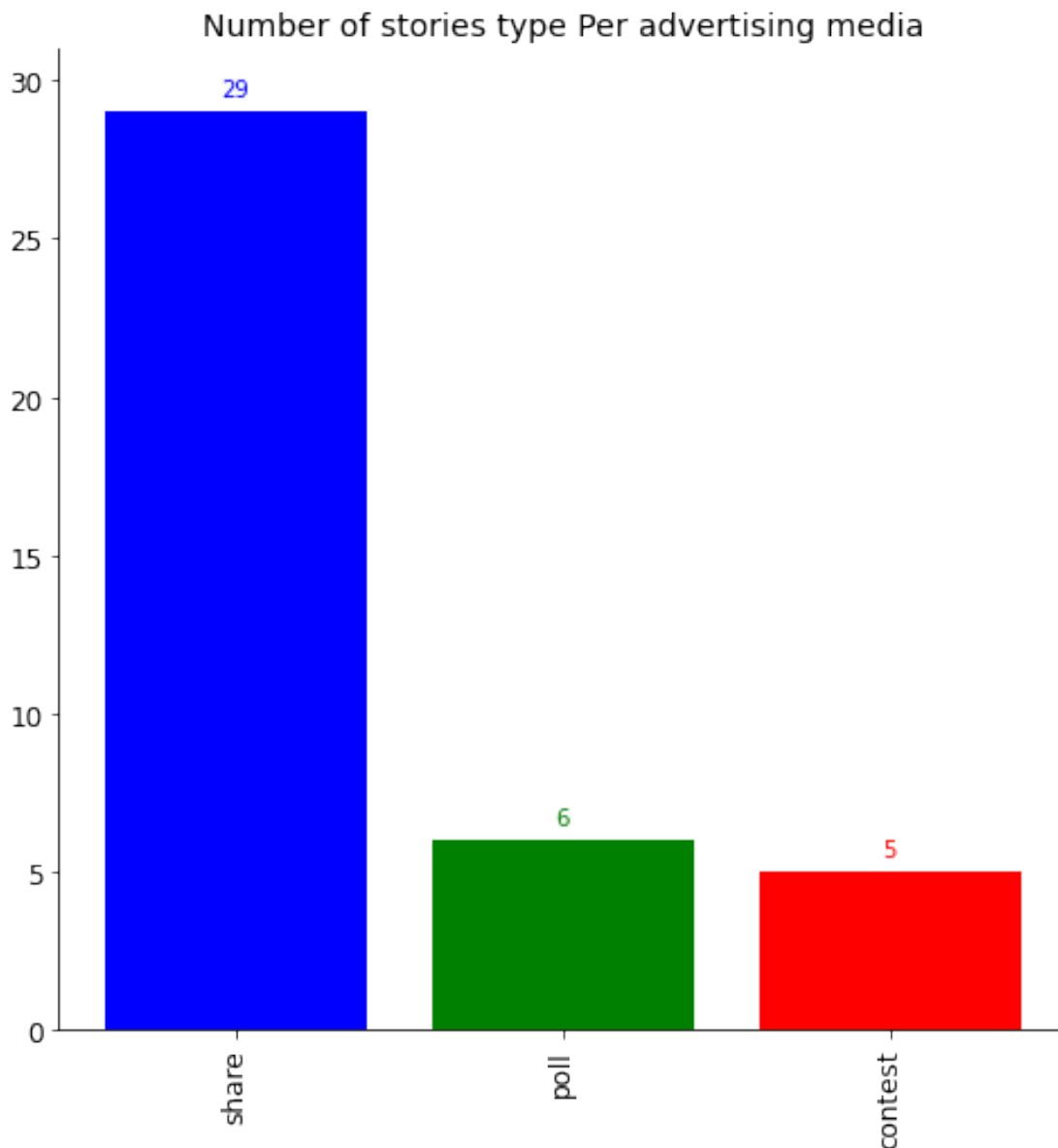
```
[90]: d = story['type'].value_counts().to_dict()
colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
fig = plt.figure(figsize = (8, 8))
ax = fig.add_subplot()
ax.bar(d.keys(), d.values(), color = colors)
for i, (k, v) in enumerate(d.items()):
    ax.text(k,
            v + .7,
            v,
            color = colors[i],
            fontsize = 10,
            horizontalalignment = 'center',
            verticalalignment = 'center')
ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
ax.tick_params(axis = 'y', labelsize = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 31)
ax.set_title("Number of stories type Per advertising media", fontsize = 14);
plt.show()

total = sum(story['type'].value_counts())
print('the top 3 types in campaign published stories and their percentages are:
↪')
```

```

print(f'1. "{list(d.keys())[0]}": {(story["type"].value_counts()[0]) / total * 100} %')
print(f'2. "{list(d.keys())[1]}": {(story["type"].value_counts()[1]) / total * 100} %')
print(f'3. "{list(d.keys())[2]}": {(story["type"].value_counts()[2]) / total * 100} %')

```



the top 3 types in campaign published stories and their percentages are:

1. "share": 72.5 %
2. "poll": 15.0 %
3. "contest": 12.5 %



```
[91]: post.drop(columns = ['post_no']).groupby('ig_tv').mean()
```

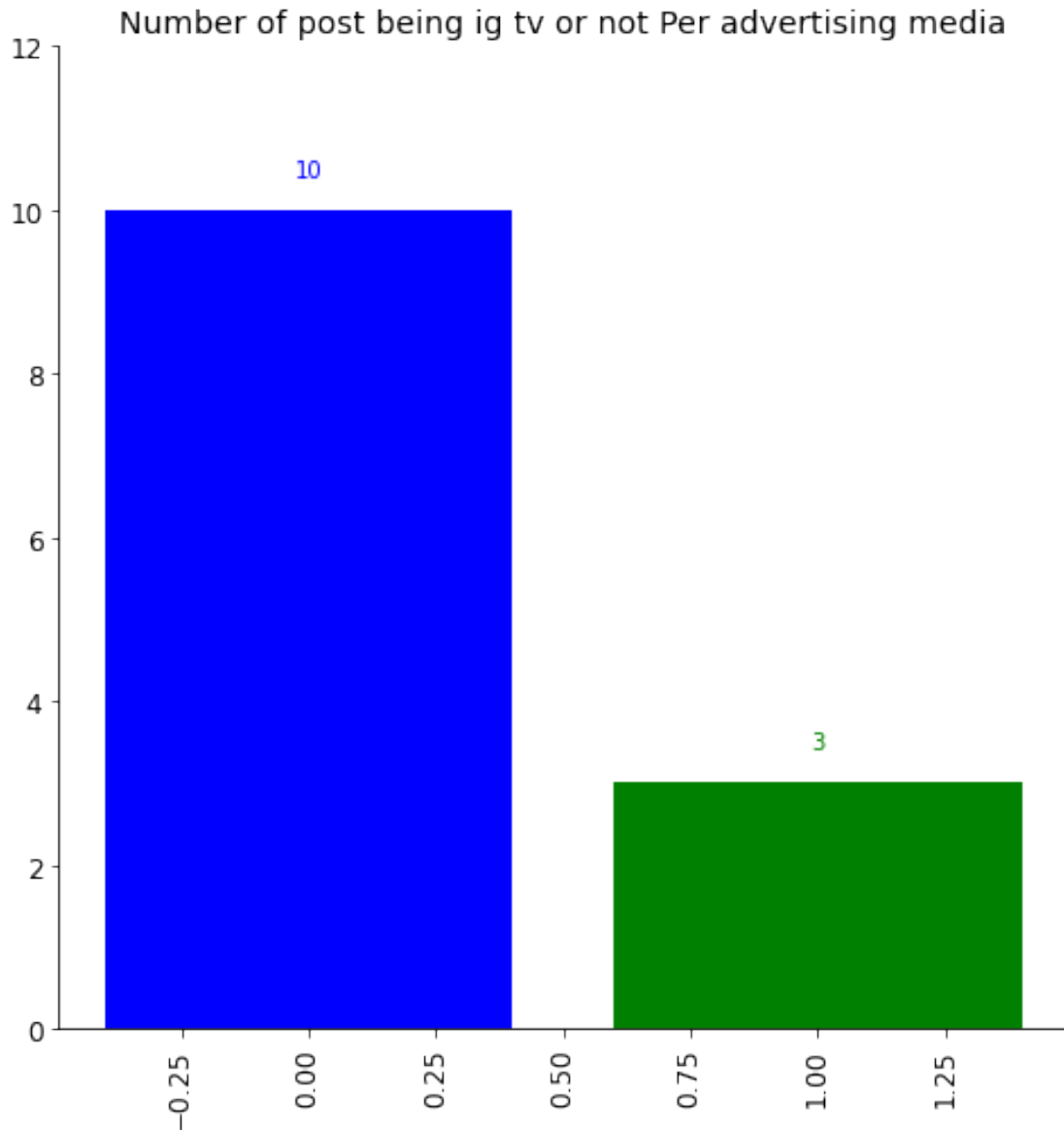
```
[91]:      like  comment  share  save  profile_visit  reach  impression  \
ig_tv
0      436.70    12.10  17.30  13.90           54.60  2762.50      3327.80
1     1422.33 10546.67  806.33 192.67           252.33  8958.00     10741.33

      view
ig_tv
0      nan
1    90504.00
```

In the table above you can see the mean of features grouped by their being ig\_tv or not in campaign published posts. as it's obvious ig\_tv posts got much more love from followers.

```
[92]: d = post['ig_tv'].value_counts().to_dict()
colors = ['blue', 'green', 'red', 'orange', 'purple', 'gray', 'brown']
fig = plt.figure(figsize = (8, 8))
ax = fig.add_subplot()
ax.bar(d.keys(), d.values(), color = colors)
for i, (k, v) in enumerate(d.items()):
    ax.text(k,
            v + .5,
            v,
            color = colors[i],
            fontsize = 10,
            horizontalalignment = 'center',
            verticalalignment = 'center')
ax.tick_params(axis = 'x', labelrotation = 90, labelsize = 12)
ax.tick_params(axis = 'y', labelsize = 12)
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
ax.set_ylim(0, 12)
ax.set_title("Number of post being ig tv or not Per advertising media",
             ↪ fontsize = 14);
plt.show()

total = sum(post['ig_tv'].value_counts())
print('the 2 post being ig tv or not in campaign published posts and their
↪ percentages are:')
print(f'1. "{list(d.keys())[0]}": {(post["ig_tv"].value_counts()[0]) / total *
↪ 100} %')
print(f'2. "{list(d.keys())[1]}": {(post["ig_tv"].value_counts()[1]) / total *
↪ 100} %')
```



the 2 post being ig tv or not in campaign published posts and their percentages are:

1. "0": 76.92307692307693 %
2. "1": 23.076923076923077 %

```
[101]: temp_df = ad_post[['name', 'view', 'benefit']].sort_values('view')
x = temp_df['name']
y = temp_df['view']
z = temp_df['benefit']

fig = plt.figure(figsize = (25, 10))
```

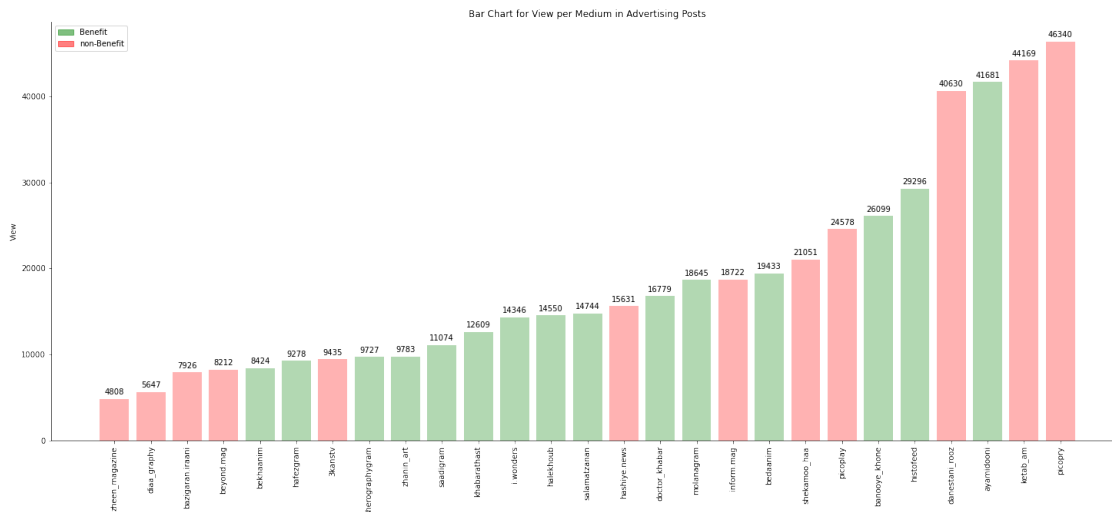
```

ax = fig.add_subplot()

for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == 0 else "green", alpha = .3)
    ax.text(x_, y_ + 500, round(y_, 1), horizontalalignment = 'center')

ax.set_xticklabels(x, rotation=90)
ax.set_ylabel("View")
ax.set_title("Bar Chart for View per Medium in Advertising Posts");
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
plt.legend(handles=[green_patch, red_patch])
plt.show()

```



as you can see in the graph above, the ability of getting huge amount of views are used by media owners to push their advertising price to a state that ad won't be beneficial cost-wise for agency.

```

[103]: temp_df = ad_post[['name', 'follower', 'benefit']].sort_values('follower')
x = temp_df['name']
y = temp_df['follower']
z = temp_df['benefit']

fig = plt.figure(figsize = (25, 10))
ax = fig.add_subplot()

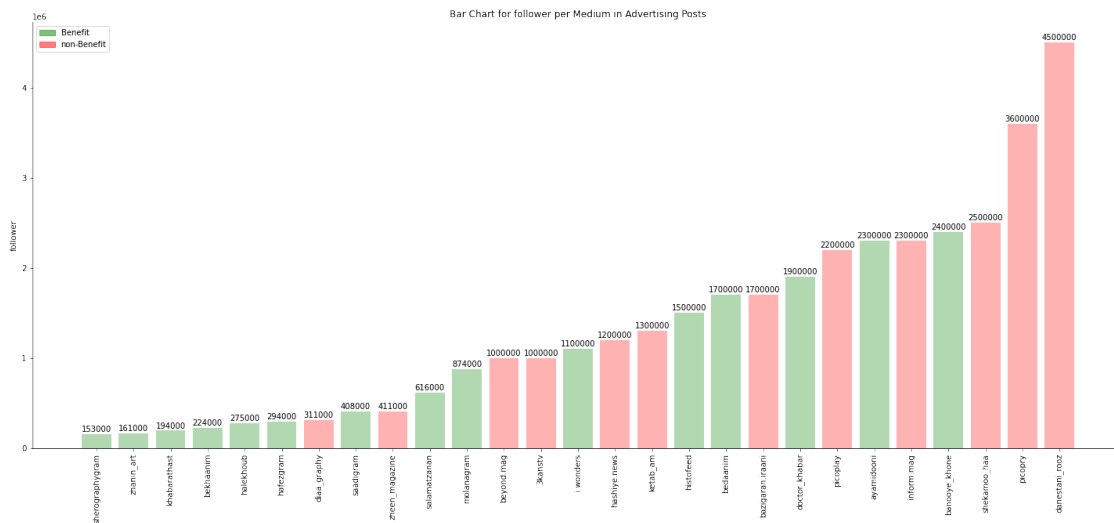
for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == 0 else "green", alpha = .3)
    ax.text(x_, y_ + 25_000, round(y_, 1), horizontalalignment = 'center')

```

```

ax.set_xticklabels(x, rotation=90)
ax.set_ylabel("follower")
ax.set_title("Bar Chart for follower per Medium in Advertising Posts");
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
plt.legend(handles=[green_patch, red_patch])
plt.show()

```



as you can see in the graph above, with proper pricing, both high follower and low followers media could be beneficial.

```

[104]: temp_df = ad_story[['name', 'view', 'benefit']].sort_values('view')
x = temp_df['name']
y = temp_df['view']
z = temp_df['benefit']

fig = plt.figure(figsize = (25, 10))
ax = fig.add_subplot()

for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == 0 else "green", alpha = .3)
    ax.text(x_, y_ + 1_500, round(y_, 1), horizontalalignment = 'center')

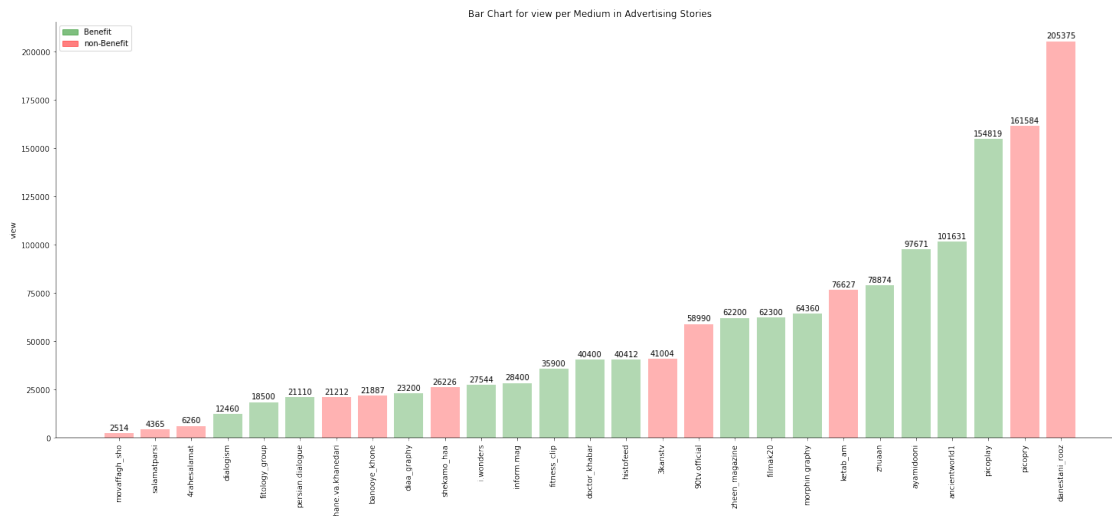
ax.set_xticklabels(x, rotation=90)
ax.set_ylabel("view")
ax.set_title("Bar Chart for view per Medium in Advertising Stories");
ax.spines["top"].set_color("None")

```

```

ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
plt.legend(handles=[green_patch, red_patch])
plt.show()

```



In the graph above you can see the view per medium in advertising stories. almost the same conclusion from advertising posts can be drawn from this graph too.

```

[105]: temp_df = ad_story[['name', 'interaction', 'benefit']].
        ↪sort_values('interaction')
x = temp_df['name']
y = temp_df['interaction']
z = temp_df['benefit']

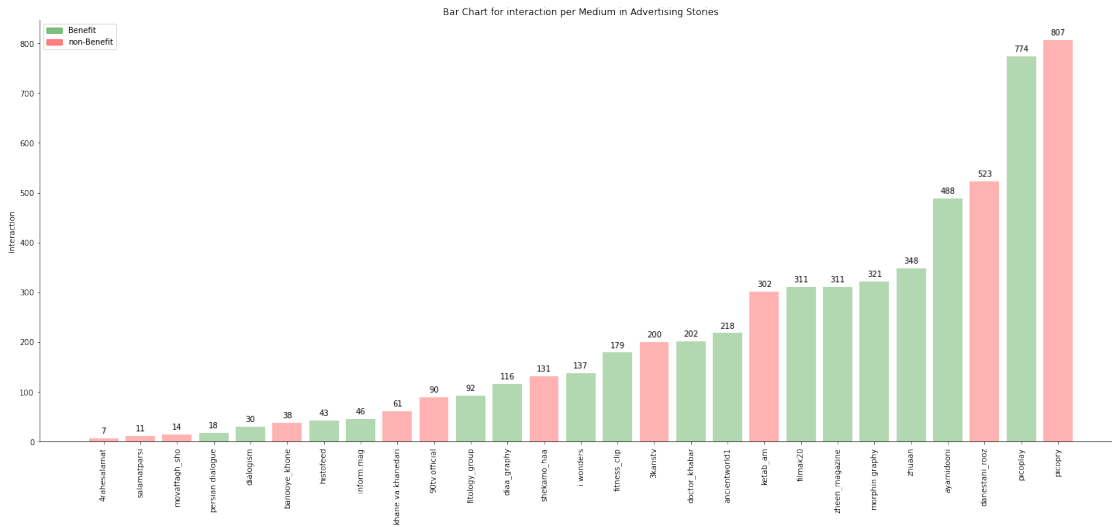
fig = plt.figure(figsize = (25, 10))
ax = fig.add_subplot()

for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == 0 else "green", alpha = .3)
    ax.text(x_, y_ + 10, round(y_, 1), horizontalalignment = 'center')

plt.xticks(rotation = 90)
ax.set_ylabel("Interaction")
ax.set_title("Bar Chart for interaction per Medium in Advertising Stories");
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
plt.legend(handles=[green_patch, red_patch])

```

```
plt.show()
```



In the graph above you can see the bar chart of interaction per medium for advertising stories. although danestani\_rooz medium got the most views but its interaction rate are fairly low in contrast of picoplay and picoplay.

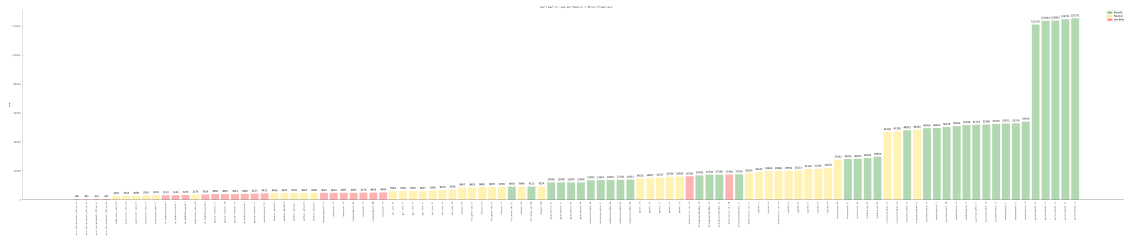
```
[106]: temp_df = influencer[['story_no', 'influ_name', 'view', 'benefit']].
        ↪sort_values('view')
x = temp_df['influ_name'] + ' ' + temp_df['story_no'].astype(str)
y = temp_df['view']
z = temp_df['benefit']

fig = plt.figure(figsize = (85, 15))
ax = fig.add_subplot()

for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
        ↪"gold", alpha = .3)
    ax.text(x_, y_ + 1_500, round(y_, 1), horizontalalignment = 'center')

plt.xticks(rotation = 90)
ax.set_ylabel("view")
ax.set_title("Bar Chart for view per Medium in Minor Influencers");
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
gold_patch = patches.Patch(color='gold', alpha = .5, label='Neutral')
plt.legend(handles=[green_patch, gold_patch, red_patch])
```

```
plt.show()
```



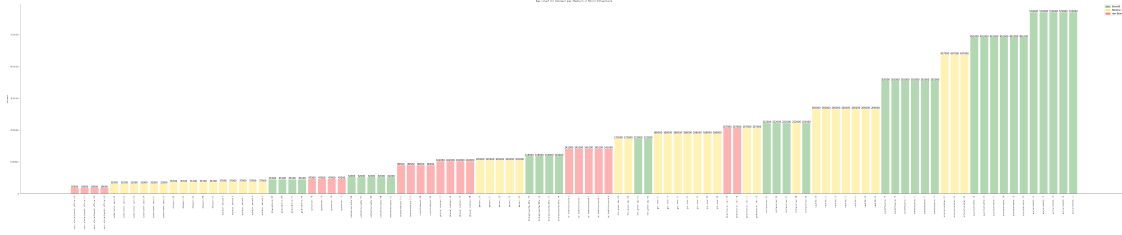
In the graph above you can see the bar chart of view per medium in minor influencers. it's worth to mentions: - this campaign influencer selection was very percise and you can obviously see the gradual growth of their view is smooth. - ayrosmelody performance was amazing regarding its cost, it's worthy to have her in mind for other campaign since her is very beneficial with current price.

```
[107]: temp_df = influencer[['story_no', 'influ_name', 'follower', 'benefit']].
        ↪sort_values('follower')
x = temp_df['influ_name'] + ' ' + temp_df['story_no'].astype(str)
y = temp_df['follower']
z = temp_df['benefit']

fig = plt.figure(figsize = (85, 15))
ax = fig.add_subplot()

for x_, y_, z_ in zip(x, y, z):
    ax.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else ↪
    ↪"gold", alpha = .3)
    ax.text(x_, y_ + 1_500, round(y_, 1), horizontalalignment = 'center')

plt.xticks(rotation = 90)
ax.set_ylabel("follower")
ax.set_title("Bar Chart for follower per Medium in Minor Influencers");
ax.spines["top"].set_color("None")
ax.spines["right"].set_color("None")
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
gold_patch = patches.Patch(color='gold', alpha = .5, label='Neutral')
plt.legend(handles=[green_patch, gold_patch, red_patch])
plt.show()
```



In the graph above you can see the bar chart of follower per medium in minor influencers. as we said earlier the selection of influencers was very good so we can see an even distribution of high-low to high-medium influencers was used in this campaign.

```
[108]: fig = plt.figure(figsize = (24, 22))
ax1 = fig.add_subplot(3,3,1)
ax2 = fig.add_subplot(3,3,2)
ax3 = fig.add_subplot(3,3,3)
ax4 = fig.add_subplot(3,3,4)
ax5 = fig.add_subplot(3,3,5)
ax6 = fig.add_subplot(3,3,6)
ax7 = fig.add_subplot(3,3,7)
ax8 = fig.add_subplot(3,3,8)
ax9 = fig.add_subplot(3,3,9)
fig.tight_layout(h_pad = 12, w_pad = 4)
red_patch = patches.Patch(color='red', alpha = .5, label='non-Benefit')
green_patch = patches.Patch(color='green', alpha = .5, label='Benefit')
gold_patch = patches.Patch(color='gold', alpha = .5, label='Neutral')

temp_df = leaders_post[['name', 'follower', 'benefit']].sort_values('follower')
x = temp_df['name']
y = temp_df['follower']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax1.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else "gold", alpha = .3)
    ax1.text(x_, y_ + 10_000, round(y_, 1), horizontalalignment = 'center')
ax1.set_ylabel("follower")
ax1.set_title("Bar Chart for follower per Medium in Major Influencers Posts")
ax1.set_xticklabels(x, rotation=90)
ax1.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'view', 'benefit']].sort_values('view')
x = temp_df['name']
y = temp_df['view']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
```



```

    ax2.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
↳"gold", alpha = .3)
    ax2.text(x_, y_ + 1000, round(y_, 1), horizontalalignment = 'center')
ax2.set_ylabel("view")
ax2.set_title("Bar Chart for view per Medium in Major Influencers Posts")
ax2.set_xticklabels(x, rotation=90)
ax2.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'like', 'benefit']].sort_values('like')
x = temp_df['name']
y = temp_df['like']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax3.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
↳"gold", alpha = .3)
    ax3.text(x_, y_ + 200, round(y_, 1), horizontalalignment = 'center')
ax3.set_ylabel("like")
ax3.set_title("Bar Chart for like per Medium in Major Influencers Posts")
ax3.set_xticklabels(x, rotation=90)
ax3.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'comment', 'benefit']].sort_values('comment')
x = temp_df['name']
y = temp_df['comment']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax4.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
↳"gold", alpha = .3)
    ax4.text(x_, y_ + 10, round(y_, 1), horizontalalignment = 'center')
ax4.set_ylabel("comment")
ax4.set_title("Bar Chart for comment per Medium in Major Influencers Posts")
ax4.set_xticklabels(x, rotation=90)
ax4.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'share', 'benefit']].sort_values('share')
x = temp_df['name']
y = temp_df['share']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax5.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
↳"gold", alpha = .3)
    ax5.text(x_, y_ + 100, round(y_, 1), horizontalalignment = 'center')
ax5.set_ylabel("share")
ax5.set_title("Bar Chart for share per Medium in Major Influencers Posts")
ax5.set_xticklabels(x, rotation=90)
ax5.legend(handles=[green_patch, gold_patch, red_patch])

```

```

temp_df = leaders_post[['name', 'save', 'benefit']].sort_values('save')
x = temp_df['name']
y = temp_df['save']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax6.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
    ↪ "gold", alpha = .3)
    ax6.text(x_, y_ + 100, round(y_, 1), horizontalalignment = 'center')
ax6.set_ylabel("save")
ax6.set_title("Bar Chart for save per Medium in Major Influencers Posts")
ax6.set_xticklabels(x, rotation=90)
ax6.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'profile_visit', 'benefit']].
    ↪ sort_values('profile_visit')
x = temp_df['name']
y = temp_df['profile_visit']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax7.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
    ↪ "gold", alpha = .3)
    ax7.text(x_, y_ + 100, round(y_, 1), horizontalalignment = 'center')
ax7.set_ylabel("profile visit")
ax7.set_title("Bar Chart for profile visit per Medium in Major Influencers
    ↪ Posts")
ax7.set_xticklabels(x, rotation=90)
ax7.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'reach', 'benefit']].sort_values('reach')
x = temp_df['name']
y = temp_df['reach']
z = temp_df['benefit']
for x_, y_, z_ in zip(x, y, z):
    ax8.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else
    ↪ "gold", alpha = .3)
    ax8.text(x_, y_ + 1000, round(y_, 1), horizontalalignment = 'center')
ax8.set_ylabel("reach")
ax8.set_title("Bar Chart for reach per Medium in Major Influencers Posts")
ax8.set_xticklabels(x, rotation=90)
ax8.legend(handles=[green_patch, gold_patch, red_patch])

temp_df = leaders_post[['name', 'impression', 'benefit']].
    ↪ sort_values('impression')
x = temp_df['name']
y = temp_df['impression']
z = temp_df['benefit']

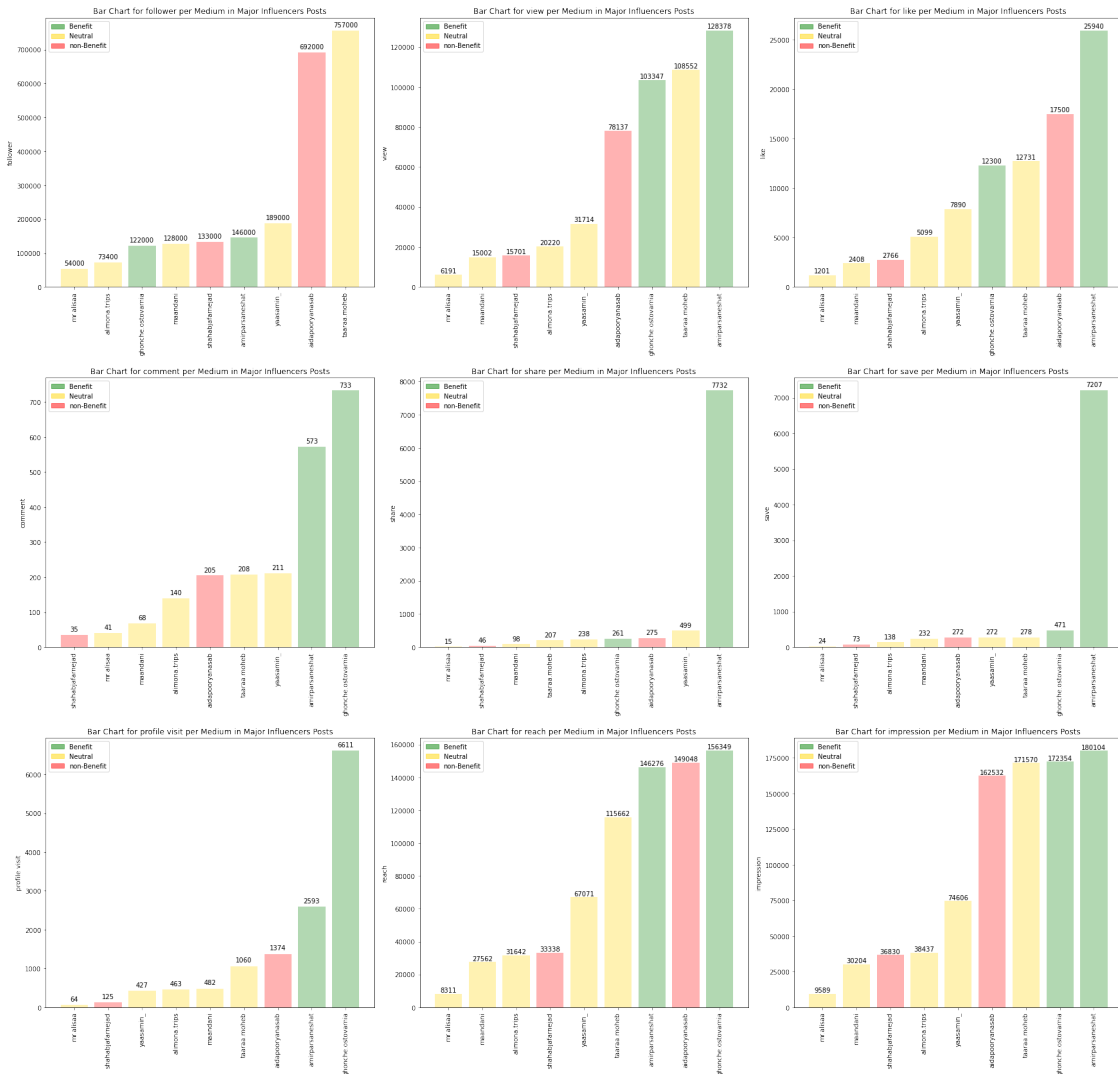
```

```

for x_, y_, z_ in zip(x, y, z):
    ax9.bar(x_, y_, color = "red" if z_ == -1 else "green" if z_ == 1 else "gold", alpha = .3)
    ax9.text(x_, y_ + 1000, round(y_, 1), horizontalalignment = 'center')
ax9.set_ylabel("impression")
ax9.set_title("Bar Chart for impression per Medium in Major Influencers Posts")
ax9.set_xticklabels(x, rotation=90)
ax9.legend(handles=[green_patch, gold_patch, red_patch])

plt.show()

```



In the graph above you can see the collection of every performance metric in major influencers post for each medium and benefit status marked with color, some interesting insights are: - although

amirparsaneshat and ghoncheostovarnia don't have highest amount of followers, they are performing very well in performance metrics. amirparsaneshat performance in like, share, save and comment are height defining. - amirparsaneshat and ghoncheostovarnia are only major influencers that we benefitted from them. - as we said earlier the only not beneficial major influencer is shahabjafanejad and you can see his performance are fairly low in contrast of other influencers.

```
[109]: from pandas import ExcelWriter
writer = ExcelWriter('data\Main Dataset V3.0.xlsx')
ad_post.to_excel(writer, 'Ad-Post')
ad_story.to_excel(writer, 'Ad-Story')
influencer.to_excel(writer, 'Influencer')
leaders_post.to_excel(writer, 'Leaders-Post')
leaders_story.to_excel(writer, 'Leaders_Story')
post.to_excel(writer, 'Post')
story.to_excel(writer, 'Story')
writer.save()
```

---

**2 Made By: Ramin Ferdos, @SimplyRamin**

```
[ ]:
```