Main

April 21, 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.

---

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from openpyxl import load_workbook
np.set_printoptions(suppress=True)
pd.set_option('display.float_format', lambda x: '%.2f' % x)
```

[2]:
```
xls = pd.ExcelFile('data/main dataset.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.

[3]:
```
print('Advertising Posts first 5 rows:')
ad_post.head()
```

```
Advertising Posts first 5 rows:
```

[3]:
```
   ad_post_no             name  follower          field   view    cost  \
0           1          3kanstv   1000000          video   9435  450000
1           2  bazigaran.iraani   1700000          video   7926  450000
2           3         bedaanim   1700000           fact  19433  404607
```

```
3            4         bekhaanim    224000  art & culture    8424  175393
4            5        beyond.mag   1000000           fact    8212  350000

   threshold
0         30
1         30
2         30
3         30
4         30
```

[4]:
```
print('Advertising Stories first 5 rows:')
ad_story.head()
```

```
Advertising Stories first 5 rows:
```

[4]:
```
   ad_story_no           name   field    view  threshold  follower  action  \
0            0   4rahesalamat  health    6260          8    686000      82
1            1  90tv.official    news   58990          8    877000     234
2            2  ancientworld1    fact  101631          8   2600000     273
3            3      ayamidooni    fact   97671          8   2300000     365
4            4  banooye_khone   women   21887          8   2400000     239

   interaction  impression    cost
0            7        6374  190578
1           90       58568  444000
2          218       94682  556000
3          488       92023  650000
4           38       74414  430000
```

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

```
Minor Influencers first 5 rows:
```

[5]:
```
   story_no         influ_name  gender      field  l_threshold  h_threshold  \
0         0  ali_bakhtiarvandi  family  lifestyle           20           60
1         1  ali_bakhtiarvandi  family  lifestyle           20           60
2         2  ali_bakhtiarvandi  family  lifestyle           20           60
3         3  ali_bakhtiarvandi  family  lifestyle           20           60
4         4  ali_bakhtiarvandi  family  lifestyle           20           60

   follower  view  action  impression  cta  interaction    cost
0    141000  3996      14        4186    0            0  360000
1    141000  3279      30        3473    1           28  360000
2    141000  3636       5        3867    0            0  360000
3    141000  3145      16        3317    1           11  360000
4    141000  3113      30        3286    1           22  360000
```

[6]:
```python
print('Main influencers stories first 5 rows:')
leaders_story.head()
```

Main influencers stories first 5 rows:

[6]:

```
   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
```

[7]:
```python
print('Main influencers posts first 5 rows:')
leaders_post.head()
```

Main influencers posts first 5 rows:

[7]:

```
   post_no                name  gender  l_threshold  h_threshold  follower  \
0        0     aidapooryanasab  female          200          400    692000
1        1       alimona.trips  family          200          400     73400
2        2     amirparsaneshat    male          200          400    146000
3        3  ghonche.ostovarnia  female          200          400    122000
4        4            maandani    male          200          400    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  30000000
1   5000000
2  15200000
3   6000000
4   5200000
```

[8]:
```python
print('Campain stories first 5 rows:')
story.head()
```

Campain stories first 5 rows:

[8]:
```
   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
```

[9]:
```python
print('Campaign posts first 5 rows:')
post.head()
```

```
Campaign posts first 5 rows:
```

[9]:
```
   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.

[10]:
```python
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.

[11]:
```
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.

[12]:
```
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: 18
	Number of Loss media: 9
	Overall Cost: 14,485,000
	Actual Cost per View: 15,108,510
	Benefit Amount: 623,510

In Advertising Stories:
	Number of Benefit media: 19
	Number of Loss media: 8
	Overall Cost: 10,564,000
	Actual Cost per View: 11,966,600
	Benefit Amount: 1,402,600

In Influencers:
	Number of Benefit media: 35
	Number of Neutral media: 46
	Number of Loss media: 21
	Overall Cost: 56,199,993
	Lowest Anticipated Overall Cost: 56,378,000
	Highest Anticipated Overall Cost: 139,699,000
	Average Anticipated Overall Cost: 98,038,500.0

In Main Influencers Posts:
	Number of Benefit media: 2
	Number of Neutral media: 6
	Number of Loss media: 1
	Overall Cost: 122,400,000
	Lowest Anticipated Overall Cost: 101,448,400
	Highest Anticipated Overall Cost: 202,896,800
	Average Anticipated Overall Cost: 152,172,600.0
```

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

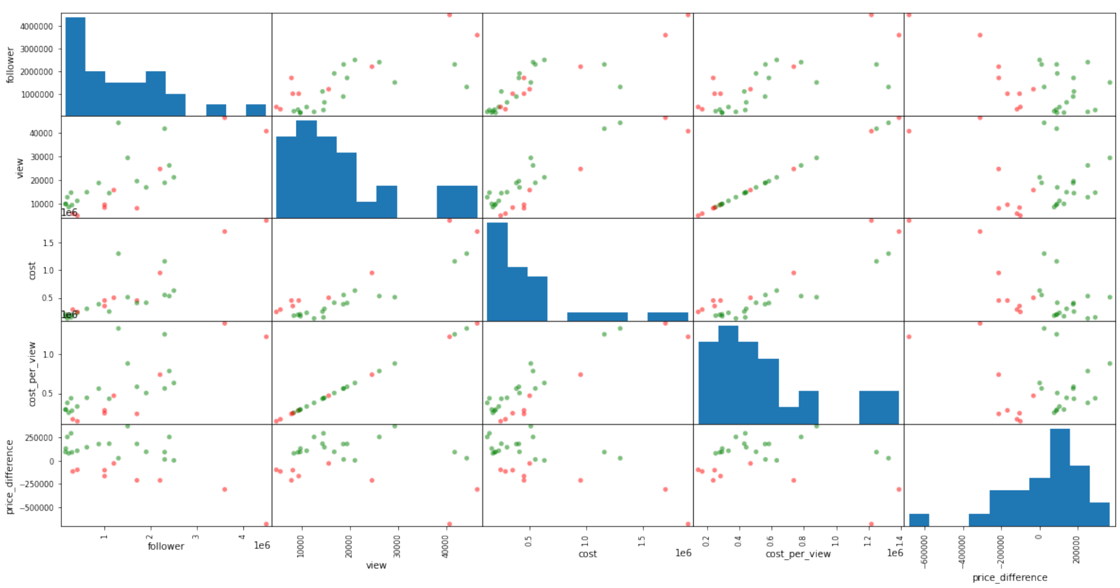
In this part we are going to check the descriptive analysis of datasets and their scatter matrix among every single feature.

Advertising Posts:

[13]:
```
ad_post.describe()
```

[13]:
```
       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  536481.48      30.00      559574.44
std          7.94  1112368.09 12119.83  467384.04       0.00      363594.93
min          1.00   153000.00  4808.00  125352.00      30.00      144240.00
25%          7.50   359500.00  9581.00  235507.00      30.00      287430.00
50%         14.00  1100000.00 14744.00  404607.00      30.00      442320.00
75%         20.50  2050000.00 22814.50  540000.00      30.00      684435.00
max         27.00  4500000.00 46340.00 1900000.00      30.00     1390200.00

       price_difference  benefit
count             27.00    27.00
mean           23092.96     0.67
std           217612.53     0.48
min          -681100.00     0.00
25%          -101567.00     0.00
50%            87921.00     1.00
75%           159502.00     1.00
max           368707.00     1.00
```

[14]:
```python
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:

[15]:
```python
ad_story.describe()
```

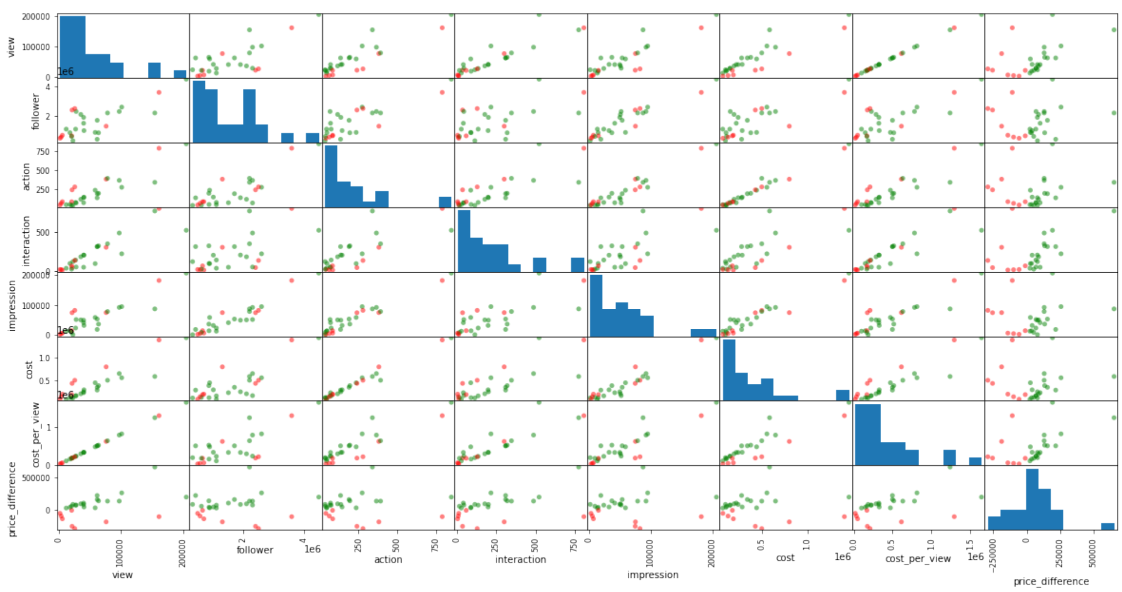
[15]:

	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	8.00	1577074.07	209.96	215.48
std	7.94	51021.85	0.00	1031146.56	210.22	219.63
min	0.00	2514.00	8.00	311000.00	33.00	7.00
25%	6.50	21549.50	8.00	769000.00	66.50	44.50
50%	13.00	40400.00	8.00	1300000.00	135.00	137.00
75%	19.50	70493.50	8.00	2250000.00	275.50	311.00
max	26.00	205375.00	8.00	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	391259.26	443207.41	51948.15	0.70
std	50542.85	355078.14	408174.77	181642.78	0.47
min	1141.00	75000.00	20112.00	-290192.00	0.00
25%	14319.50	154275.50	172396.00	-33363.50	0.00
50%	49339.00	280410.00	323200.00	68801.00	1.00
75%	76583.50	500000.00	563948.00	131180.00	1.00
max	206633.00	1450000.00	1643000.00	653552.00	1.00

[16]:

```python
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()
```

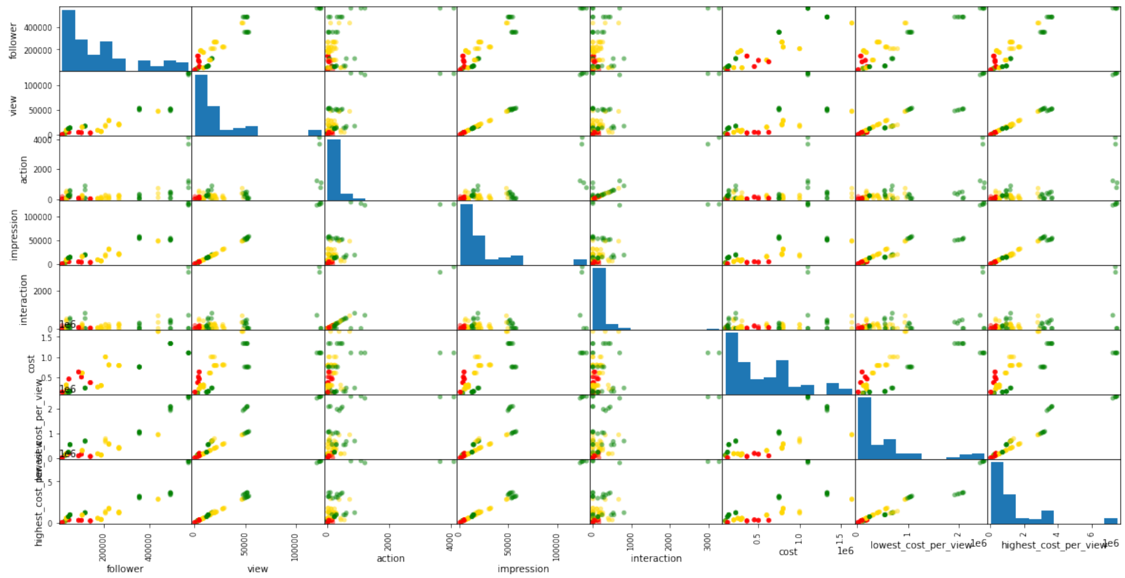


Minor Influencers:

[17]:
```
influencer.describe()
```

[17]:
```
       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        28.24        64.12  183950.00   21864.71  254.63
std       29.59         9.89         4.95  159314.56   28090.10  565.16
min        0.00        20.00        60.00   18000.00     396.00    3.00
25%       25.25        20.00        60.00   47000.00    4628.25   30.00
50%       50.50        20.00        60.00  141000.00   12000.00  103.50
75%       75.75        40.00        70.00  266000.00   21874.25  229.00
max      101.00        40.00        70.00  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  550980.32             552725.49
std      28774.04   0.49       454.80  403312.45             653557.61
min        823.00   0.00         0.00  100000.00              15840.00
25%       4967.00   0.00         0.00  225000.00             142640.00
50%      12876.00   1.00        42.50  450000.00             240000.00
75%      22286.75   1.00       184.25  785714.00             684960.00
max     129903.00   1.00      3284.00 1633333.00            2507420.00

       highest_cost_per_view  benefit
count                 102.00   102.00
mean              1369598.04     0.14
std               1711763.68     0.73
min                 27720.00    -1.00
25%                317677.50     0.00
50%                720000.00     0.00
75%               1370010.00     1.00
max               7522260.00     1.00
```

[18]:
```
pd.plotting.scatter_matrix(influencer.drop(['story_no', 'l_threshold',␣
↪'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()
```



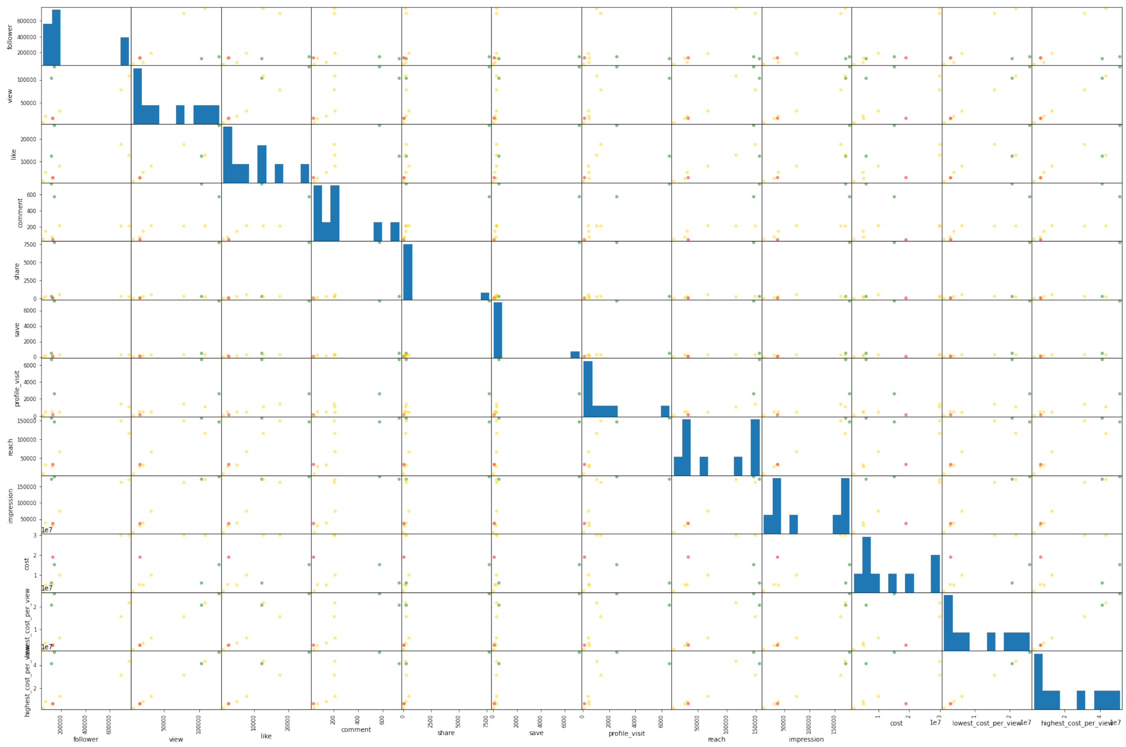
Major Influencers Advertising Posts:

[19]:
```
leaders_post.describe()
```

[19]:
```
       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       200.00       400.00 254933.33   56360.22   9759.44
std       2.74         0.00         0.00 269557.86   47940.99   8203.85
min       0.00       200.00       400.00  54000.00    6191.00   1201.00
25%       2.00       200.00       400.00 122000.00   15701.00   2766.00
50%       4.00       200.00       400.00 133000.00   31714.00   7890.00
75%       6.00       200.00       400.00 189000.00  103347.00  12731.00
max       8.00       200.00       400.00 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  13600000.00           11272044.44            22544088.89     0.11
std   10720541.03            9588197.17            19176394.35     0.60
```

min	2000000.00	1238200.00	2476400.00	-1.00
25%	5200000.00	3140200.00	6280400.00	0.00
50%	10000000.00	6342800.00	12685600.00	0.00
75%	19000000.00	20669400.00	41338800.00	0.00
max	30000000.00	25675600.00	51351200.00	1.00

[20]:
```
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:

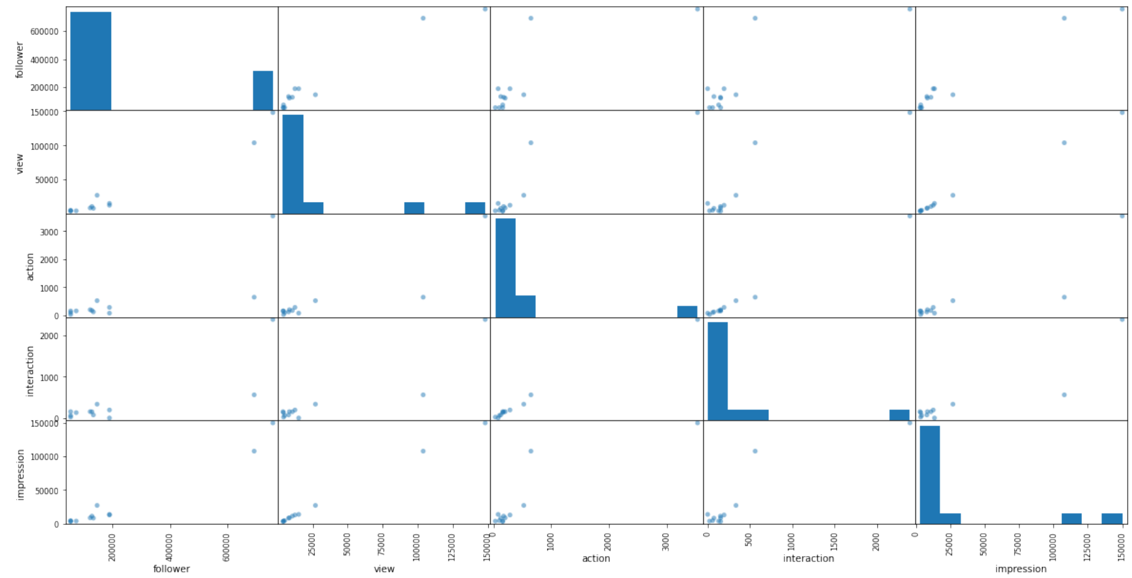
[21]:
```
leaders_story.describe()
```

[21]:

	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
```

[22]:
```
pd.plotting.scatter_matrix(leaders_story.drop(['story_no', 'cost'], axis=1),␣
 ↪figsize=(20,10), s=100)
plt.show()
```



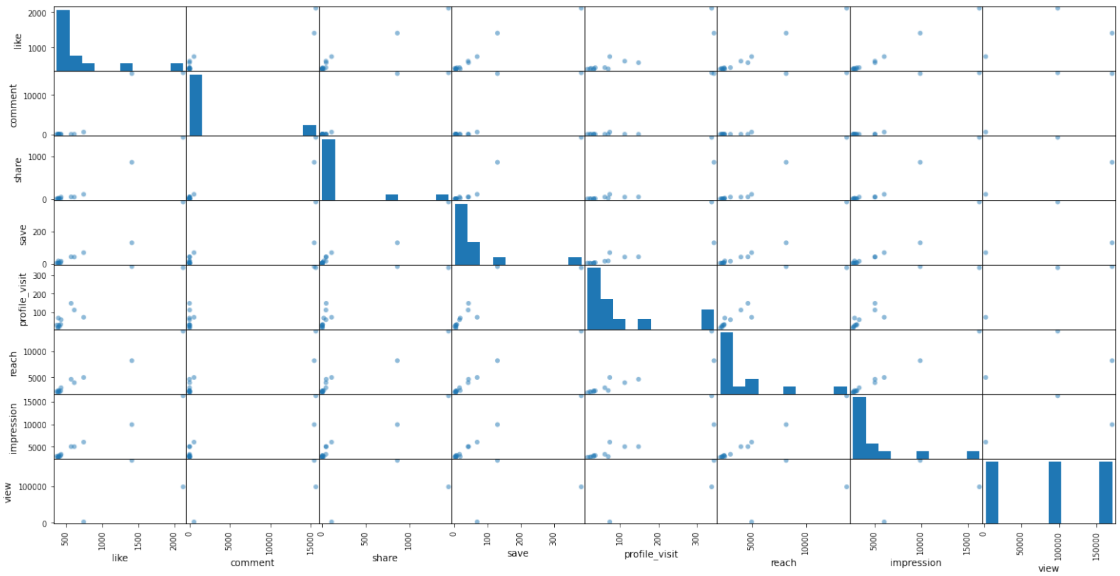
Campaign Posts:

[23]:
```
post.describe()
```

[23]:
```
         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
```

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



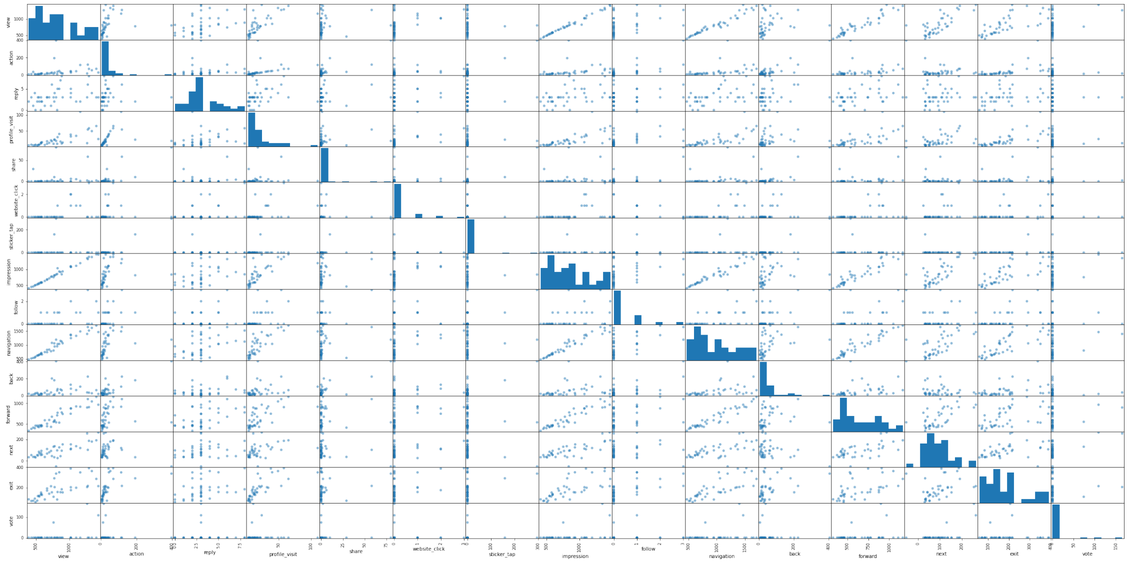
Campaign Story:

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

```
[25]:        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
```

[26]:
```
pd.plotting.scatter_matrix(story.drop(['story_no'], axis=1), figsize=(40,20),␣
↪s=100)
plt.show()
```

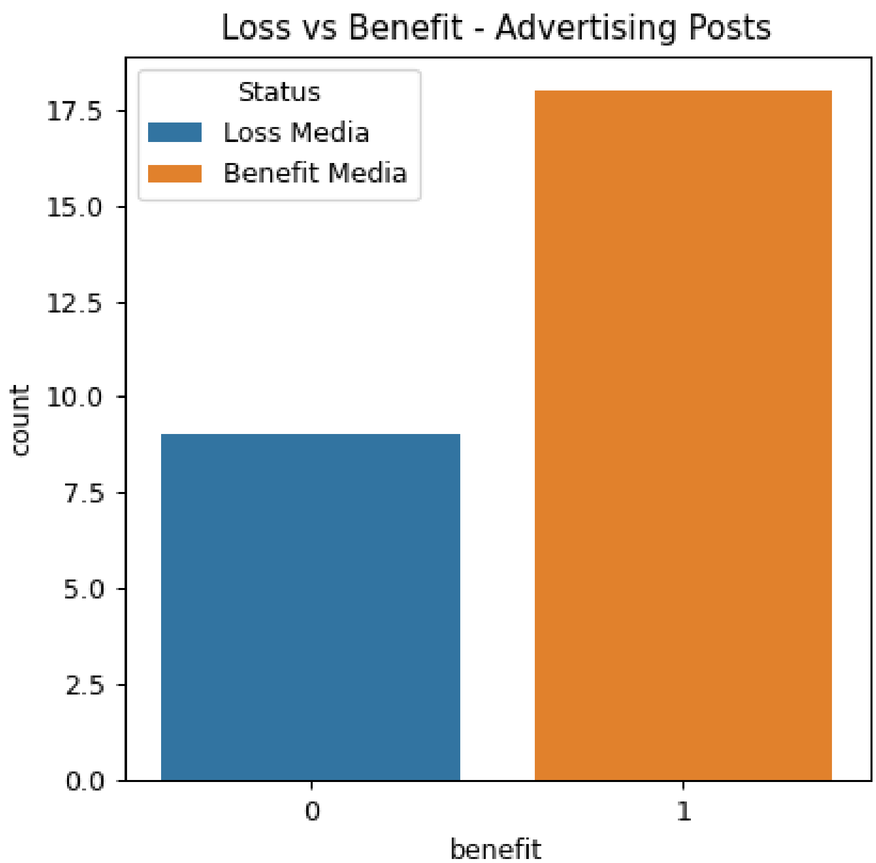


1.1.3 Data Exploration:

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

[27]:
```
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()
```

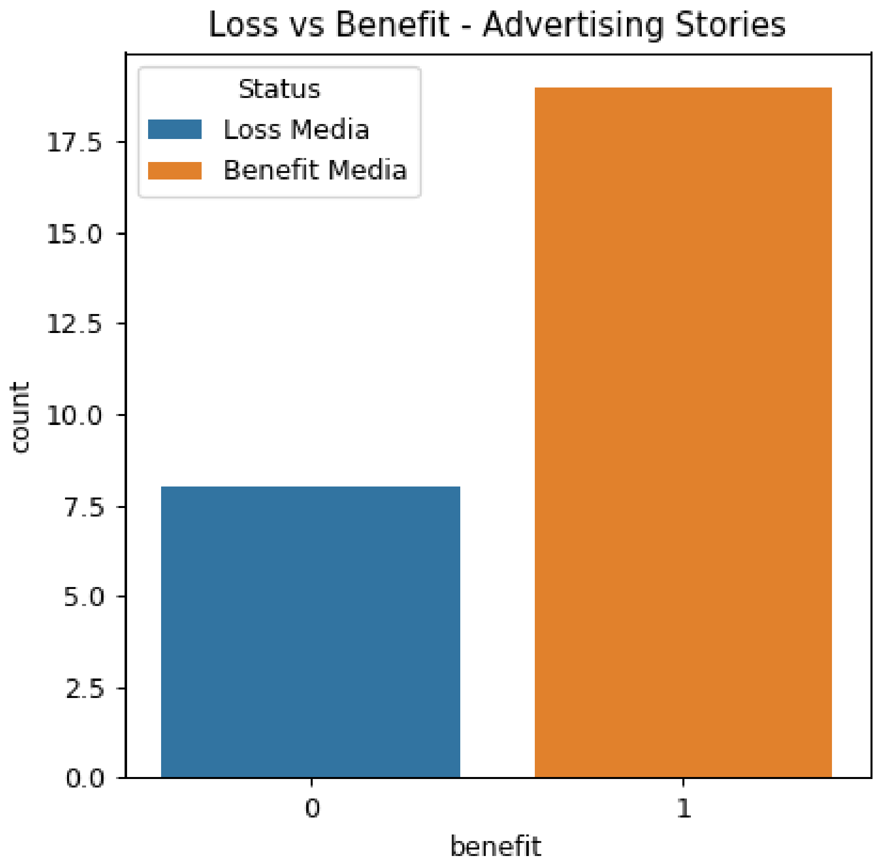
```python
count_benefit = len(ad_post[ad_post['benefit'] == 1])
count_loss = len(ad_post[ad_post['benefit'] == 0])
print(f'The number of benefit media are: {count_benefit}.')
print(f'The number of loss media are: {count_loss}.')
```




```
The number of benefit media are: 18.
The number of loss media are: 9.
```

[28]:
```python
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}.')
```

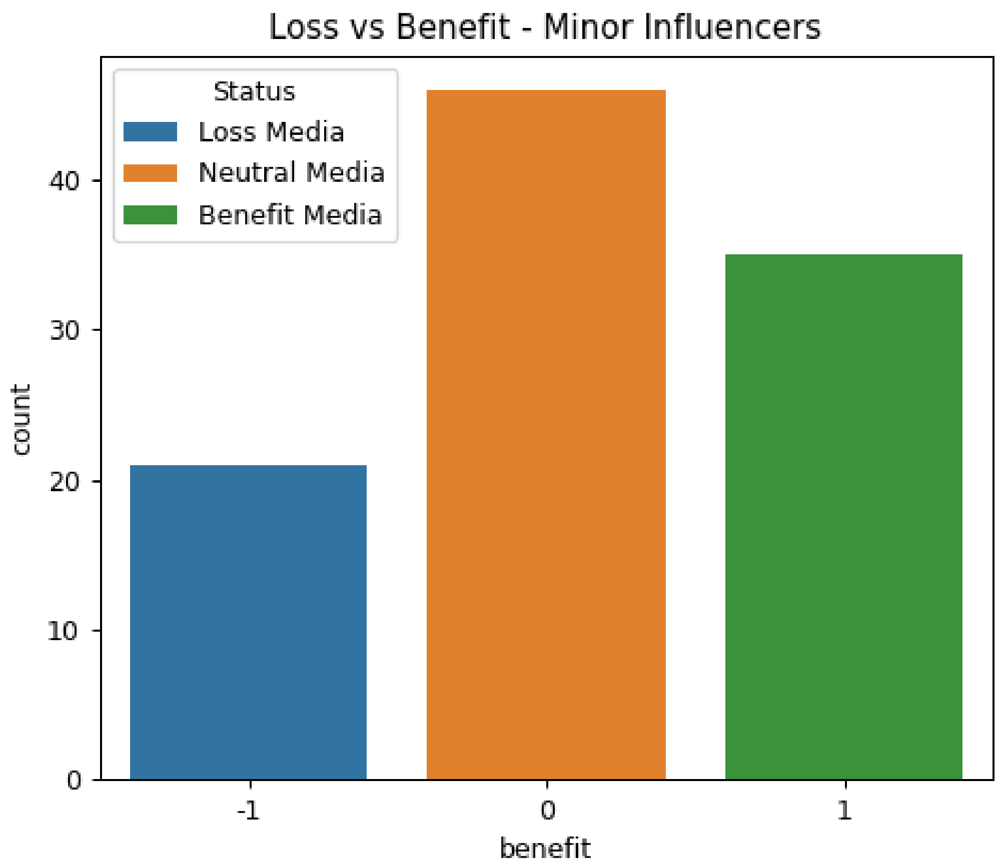



```
The number of benefit media are: 19.
The number of loss media are: 8.
```

[29]:

```
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}.')
```

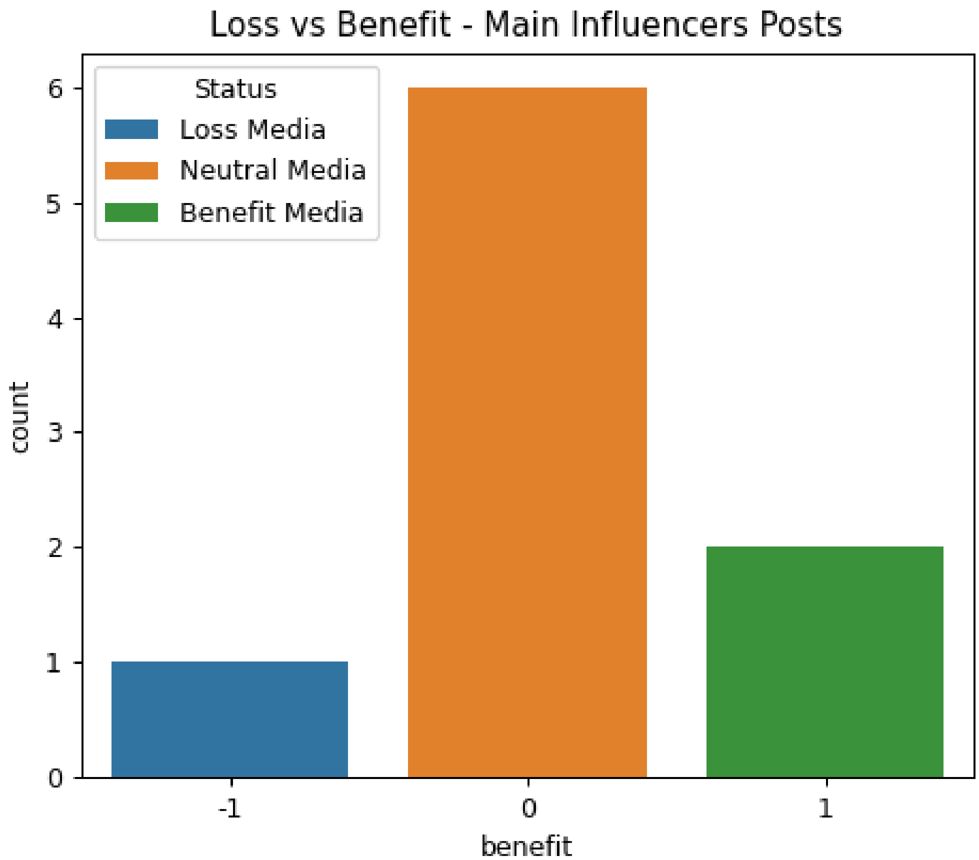



```
The number of benefit media are: 35.
The number of loss media are: 21.
The number of neutral media are: 46.
```

[30]:

```
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}.')
```

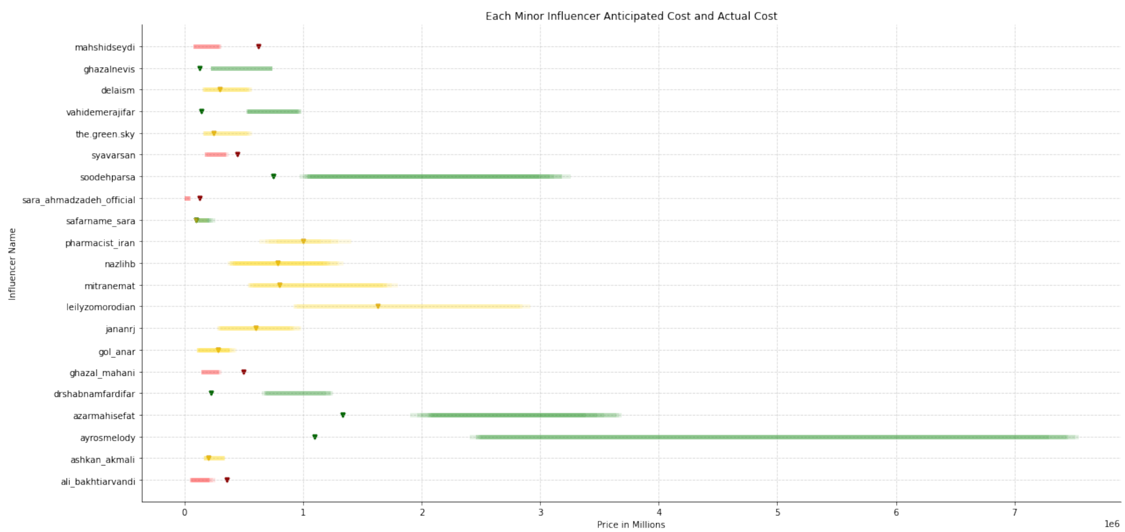
```
print(f'The number of loss media are: {count_loss}.')
print(f'The number of neutral media are: {count_neutral}.')
```




```
The number of benefit media are: 2.
The number of loss media are: 1.
The number of neutral media are: 6.
```

[31]:
```
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)
```

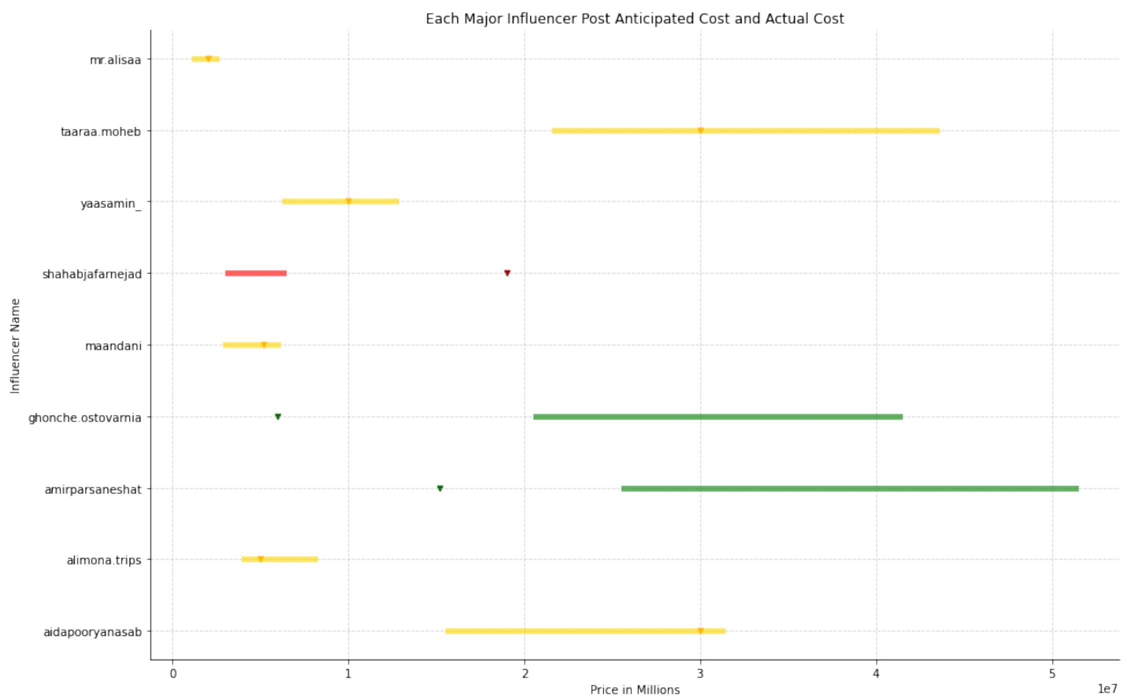
```
    ax.scatter(c_, y_, s=20, marker='v', c="darkred" if z_ == -1 else␣
 ↪"darkgreen" if z_ == 1 else 'goldenrod')

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”.

[32]:
```
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.

[33]:
```
ad_post.drop(columns = ['ad_post_no', 'threshold']).groupby('benefit').mean()
```

```
[33]:          follower     view      cost  cost_per_view  price_difference
benefit
0        1769111.11 18134.11 758888.89      544023.33        -214865.56
1        1122166.67 18911.67 425277.78      567350.00         142072.22
```

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.

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

```
[34]:            view   follower  action  interaction  impression      cost  \
benefit
0        40084.38 1521000.00  243.75       171.38    55173.88 463750.00
1        61850.00 1600684.21  195.74       234.05    53145.21 360736.84

         cost_per_view  price_difference
benefit
0            320675.00        -143075.00
1            494800.00         134063.16
```

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.

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

```
[35]:         l_threshold  h_threshold  follower     view  action  impression  cta  \
benefit
-1             31.43        65.71  82428.57  3445.05   63.90     3697.24 0.67
 0             24.35        62.17 174750.00 15745.87  144.52    16470.07 0.63
 1             31.43        65.71 256954.29 40958.40  513.77    42579.37 0.54

         interaction      cost  lowest_cost_per_view  highest_cost_per_view
benefit
-1             50.05 409523.81             103733.33              224119.05
 0            119.91 586956.41             356943.91              965765.43
 1            347.66 588571.37            1079433.71             2587636.86
```

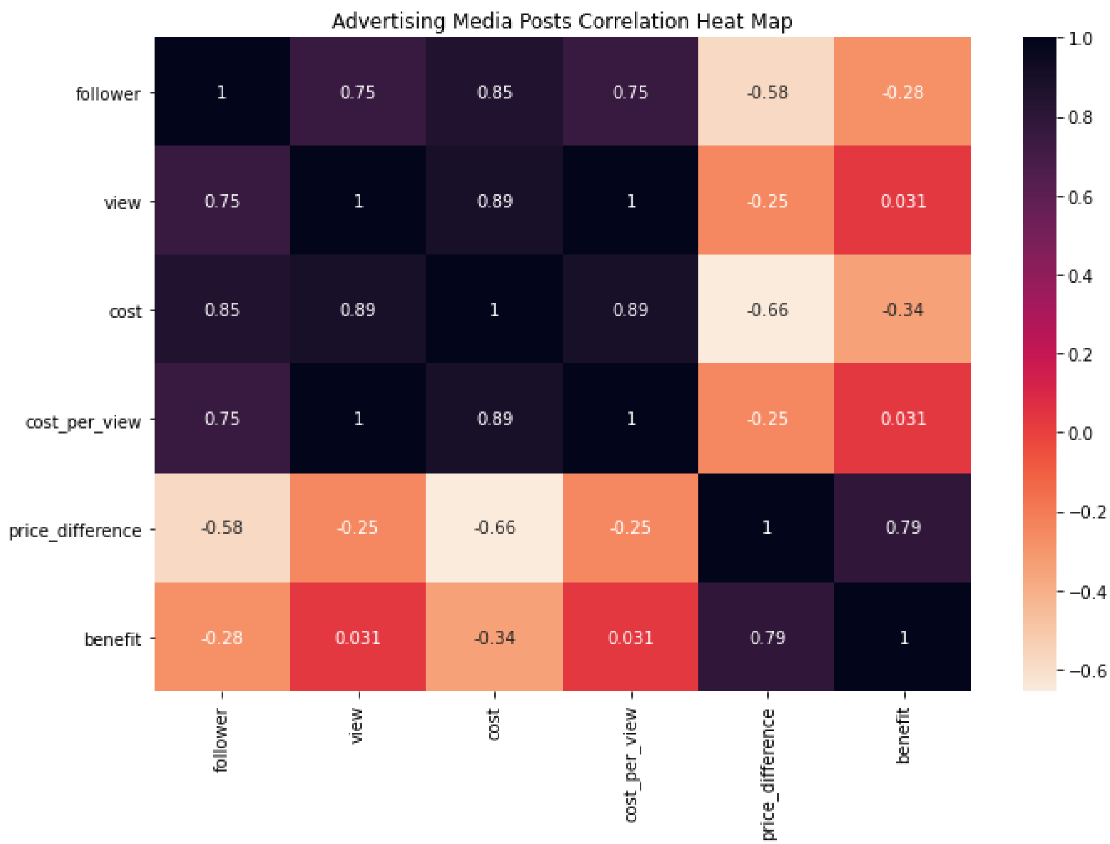
In the cell above you can see the minor influencers grouped by their benefit status, based on that information we can deduce that: - more followers in minor influencers means the higher chance of being benefitted. this fact can be interpreted as selected influencers had almost no fake followers and their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

[36]:
```
leaders_post.drop(columns = ['post_no', 'l_threshold', 'h_threshold']).
↪groupby('benefit').mean()
```

[36]:
```
         follower       view      like  comment    share    save  profile_visit  \
benefit
-1      133000.00   15701.00   2766.00    35.00    46.00   73.00         125.00
 0      315566.67   43302.67   7804.83   145.50   222.00  202.67         645.00
 1      134000.00  115862.50  19120.00   653.00  3996.50 3839.00        4602.00

            reach  impression        cost  lowest_cost_per_view  \
benefit
-1       33338.00    36830.00 19000000.00            3140200.00
 0       66549.33    81156.33 13700000.00            8660533.33
 1      151312.50   176229.00 10600000.00           23172500.00

         highest_cost_per_view
benefit
-1                  6280400.00
 0                 17321066.67
 1                 46345000.00
```

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 corrolated with view are important. we can vaguly 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".

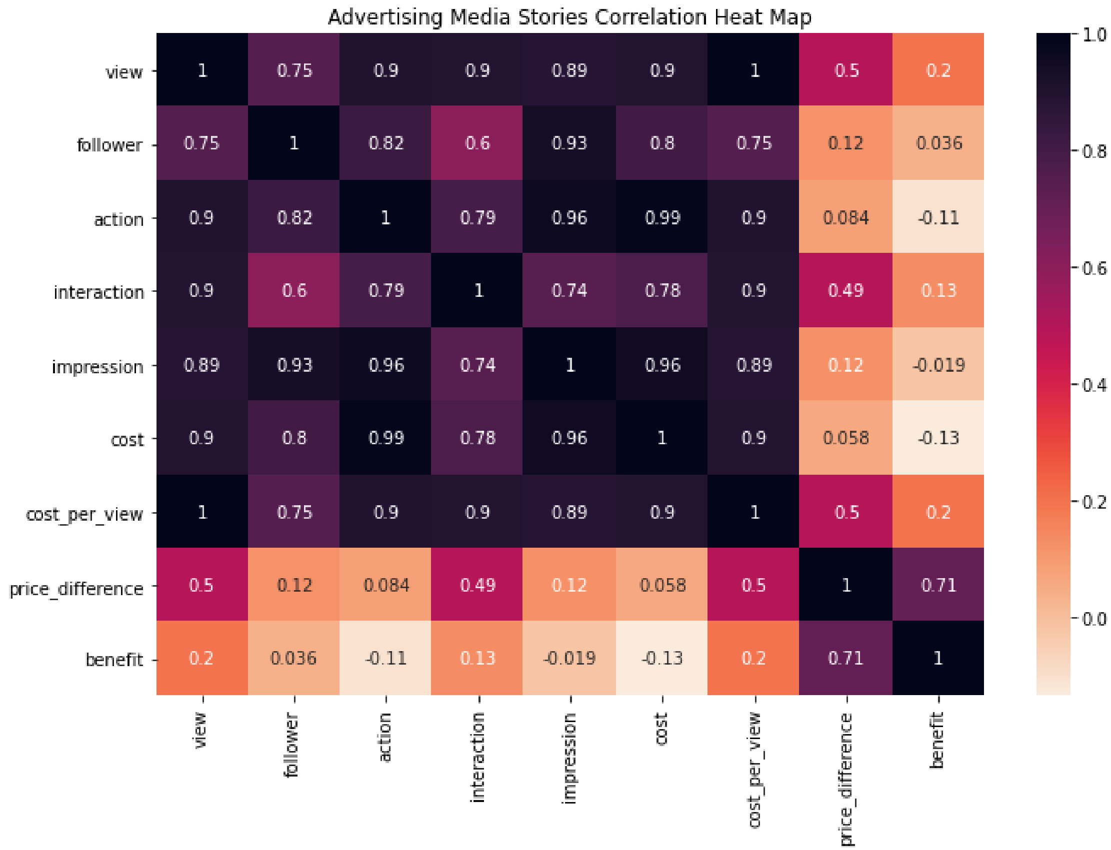
[37]:
```
intercor = ad_post.drop(columns = ['ad_post_no', 'threshold']).corr()
plt.figure(figsize=(10,7))
sns.heatmap(intercor,annot=True, cmap = 'rocket_r')
plt.tight_layout()
plt.title('Advertising Media Posts Correlation Heat Map')
plt.show()
```



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.

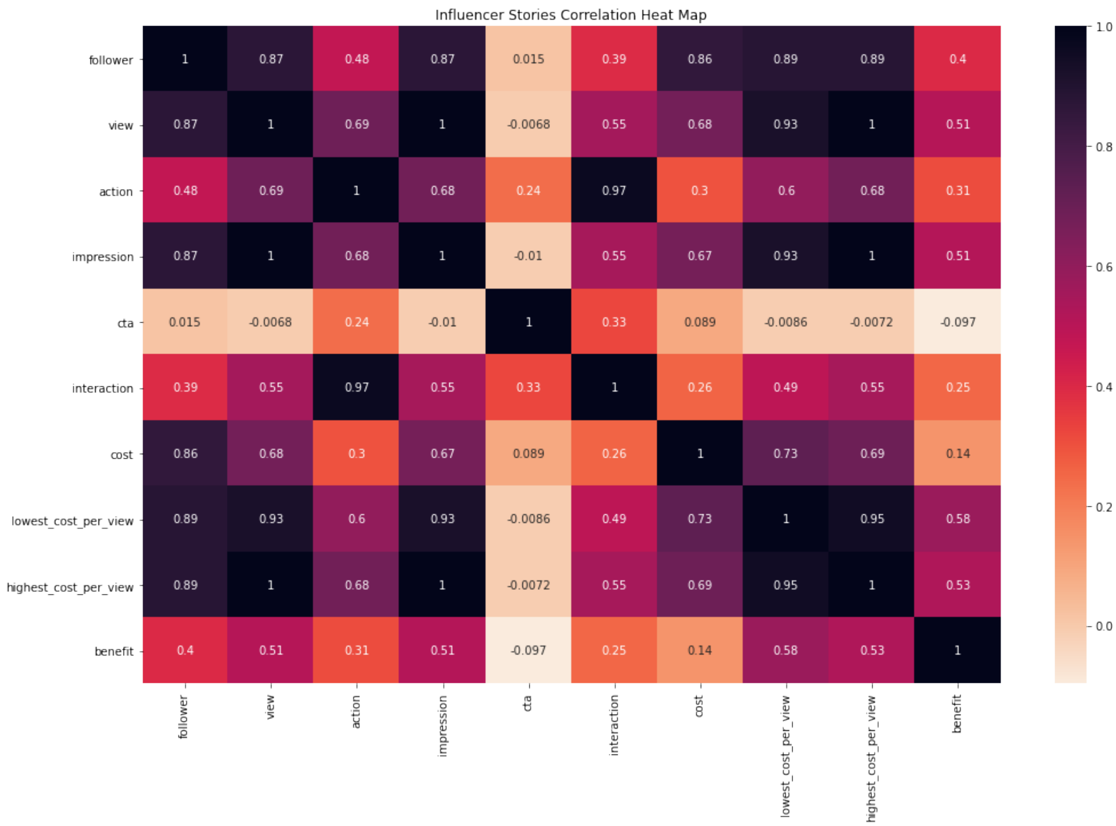
[38]:

```
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()
```



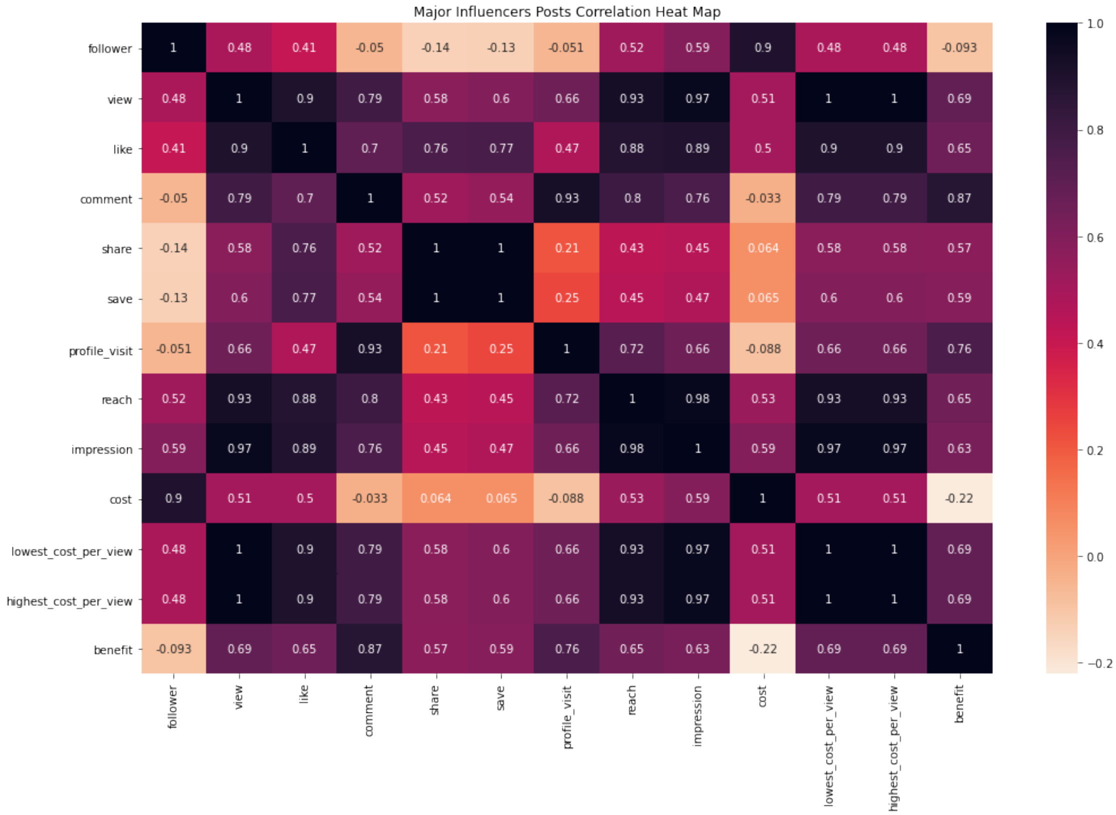
In the graph above you can see the feature correlation heatmap for advertising media stories, some interesting insights: - view is strongly correlated with action, interaction, impression and improtantly, cost and cost per view. - although view and follower are correlated positevly, their relationship 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 benefical. - 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.

[39]:
```
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()
```



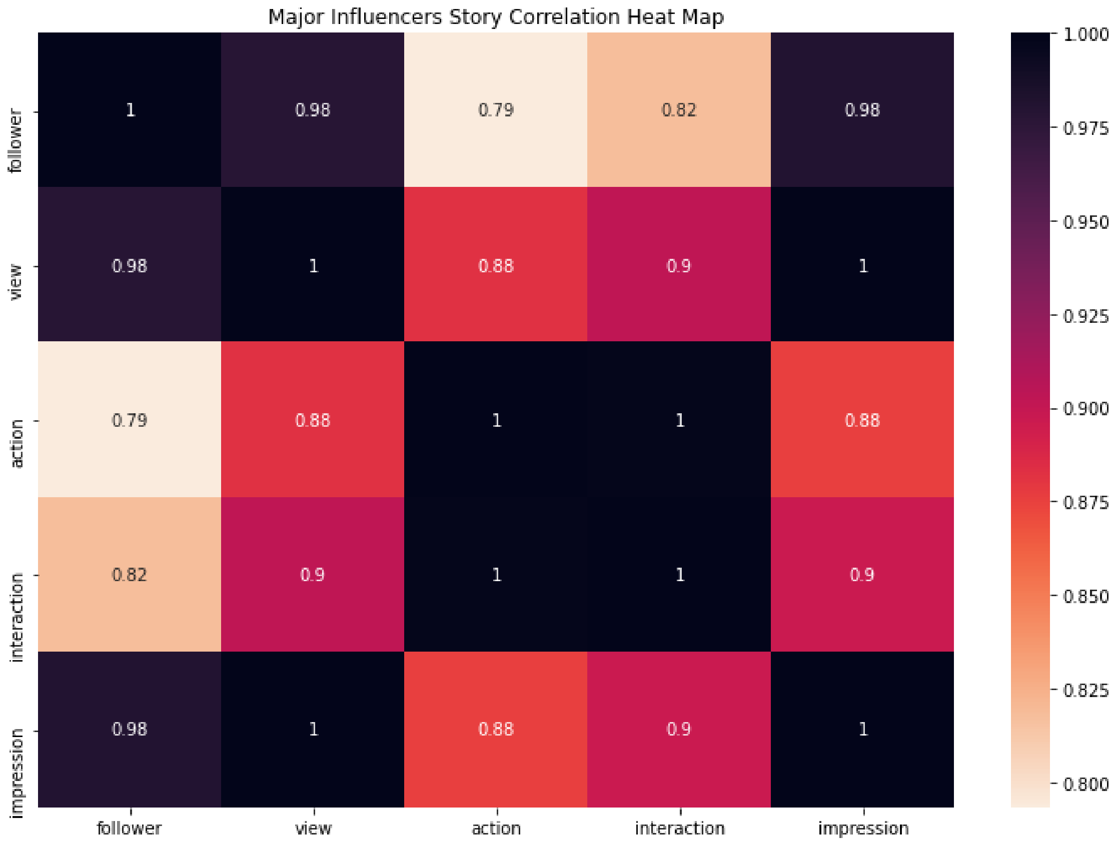
In the cell above you can see the features correlation heatmap of minor influencers. almost the same insight as advertising stories can be deduced from this heatmap.

[40]:
```
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()
```



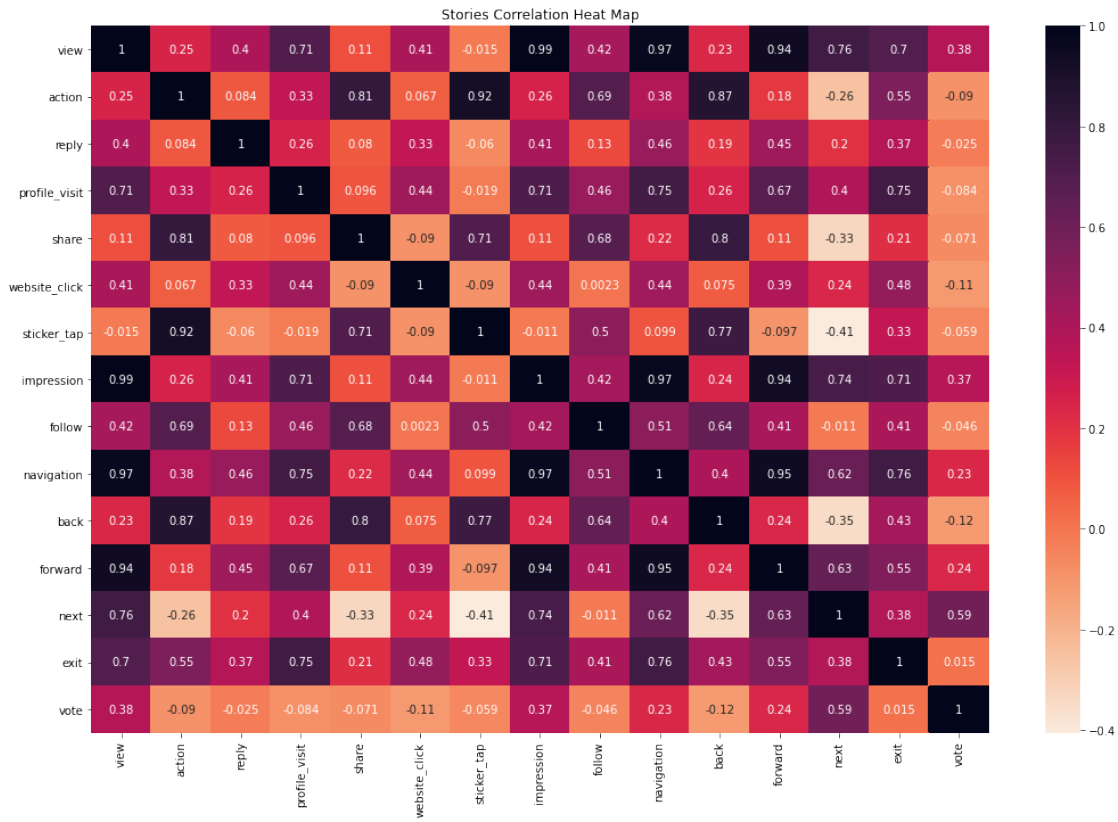
In the cell above you can see the feature correlation heatmap for Major influencers advertising posts. some interesting insights from this graph are: - there is strong positive between share and save. this could be interpreted as almost everyone who shared their post, also saved their post too. - major 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.

[41]:
```
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()
```



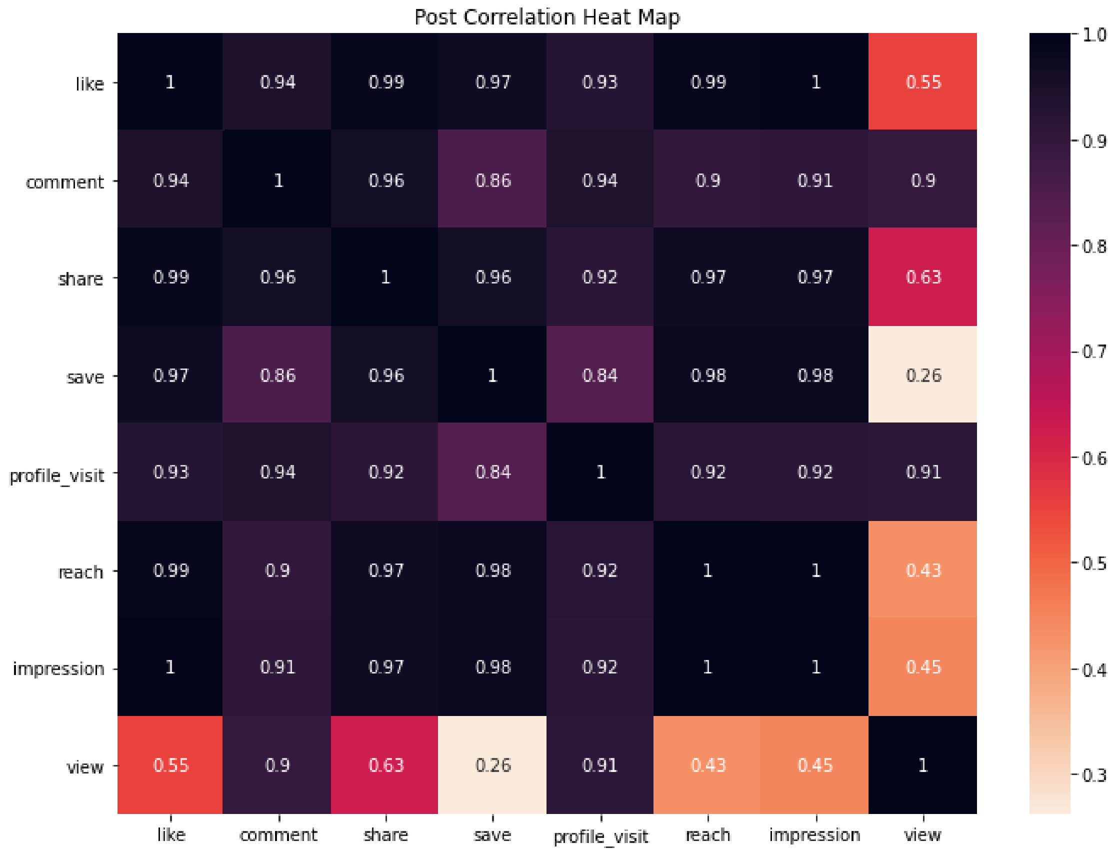
In the cell above you can see the correlation heatmap for Major influencers advertising story contents, some interesting insights from this data are: - view and follower are strongly correlated, this means that almost the good amount of major influencers followers watch their stories. also this fact 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.

[42]:
```
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.

[43]:
```
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 we worthy to have glimpse at the result.

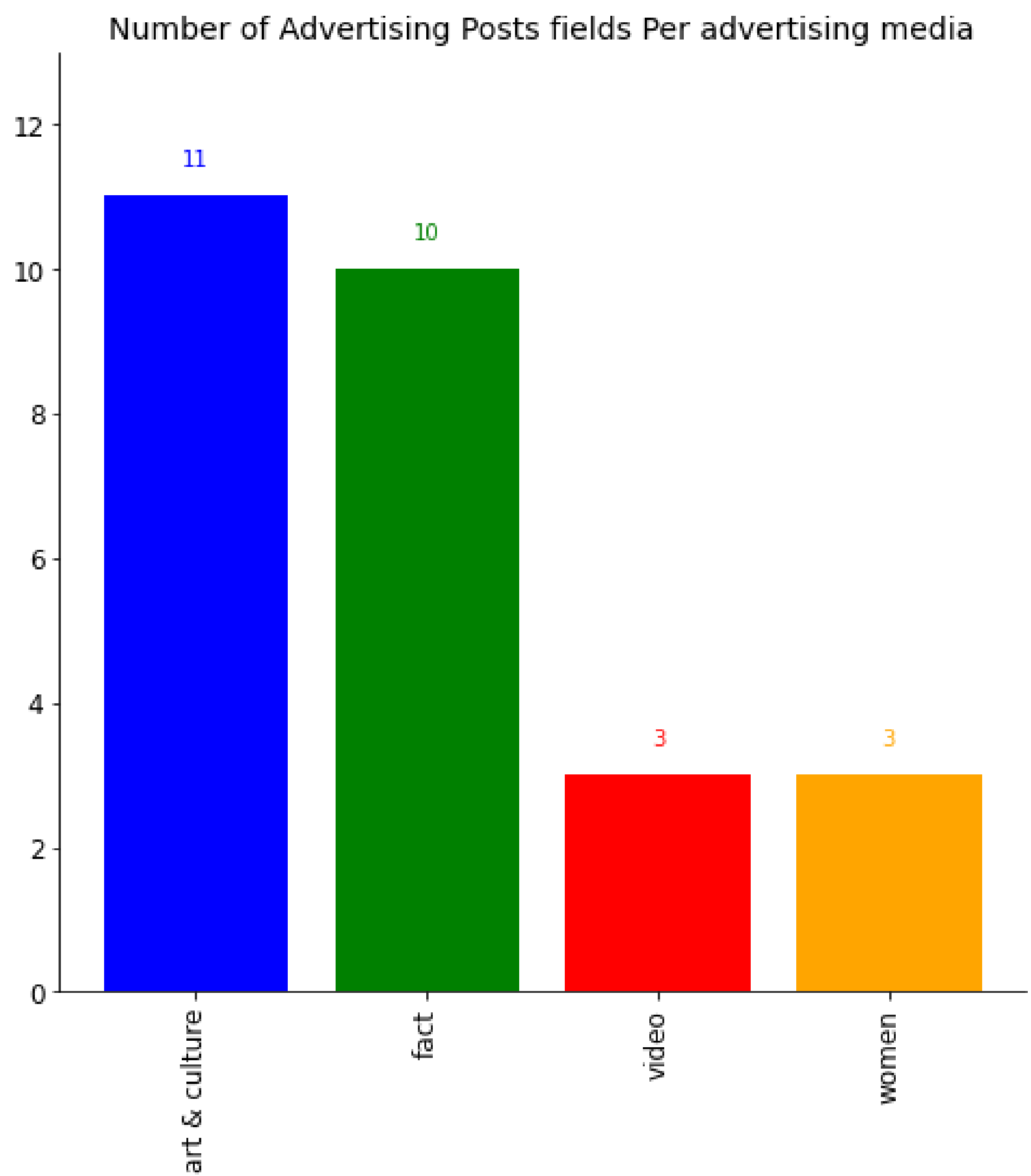
[44]:
```python
ad_post.drop(columns = ['ad_post_no', 'threshold']).groupby('field').mean()
```

[44]:
```
                follower      view       cost  cost_per_view  price_difference  \
field
art & culture  418636.36  13519.45  312763.00      405583.64          92820.64
fact          2210000.00  26001.70  818460.70      780051.00         -38409.70
video         1300000.00  10997.33  466666.67      329920.00        -136746.67
women         1838666.67  20631.33  486666.67      618940.00         132273.33

               benefit
field
art & culture     0.82
fact              0.60
video             0.00
women             1.00
```

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.

[102]:

```
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. "video": 11.11111111111111 %
```

[45]:
```
ad_story.drop(columns = ['ad_story_no', 'threshold']).groupby('field').mean()
```

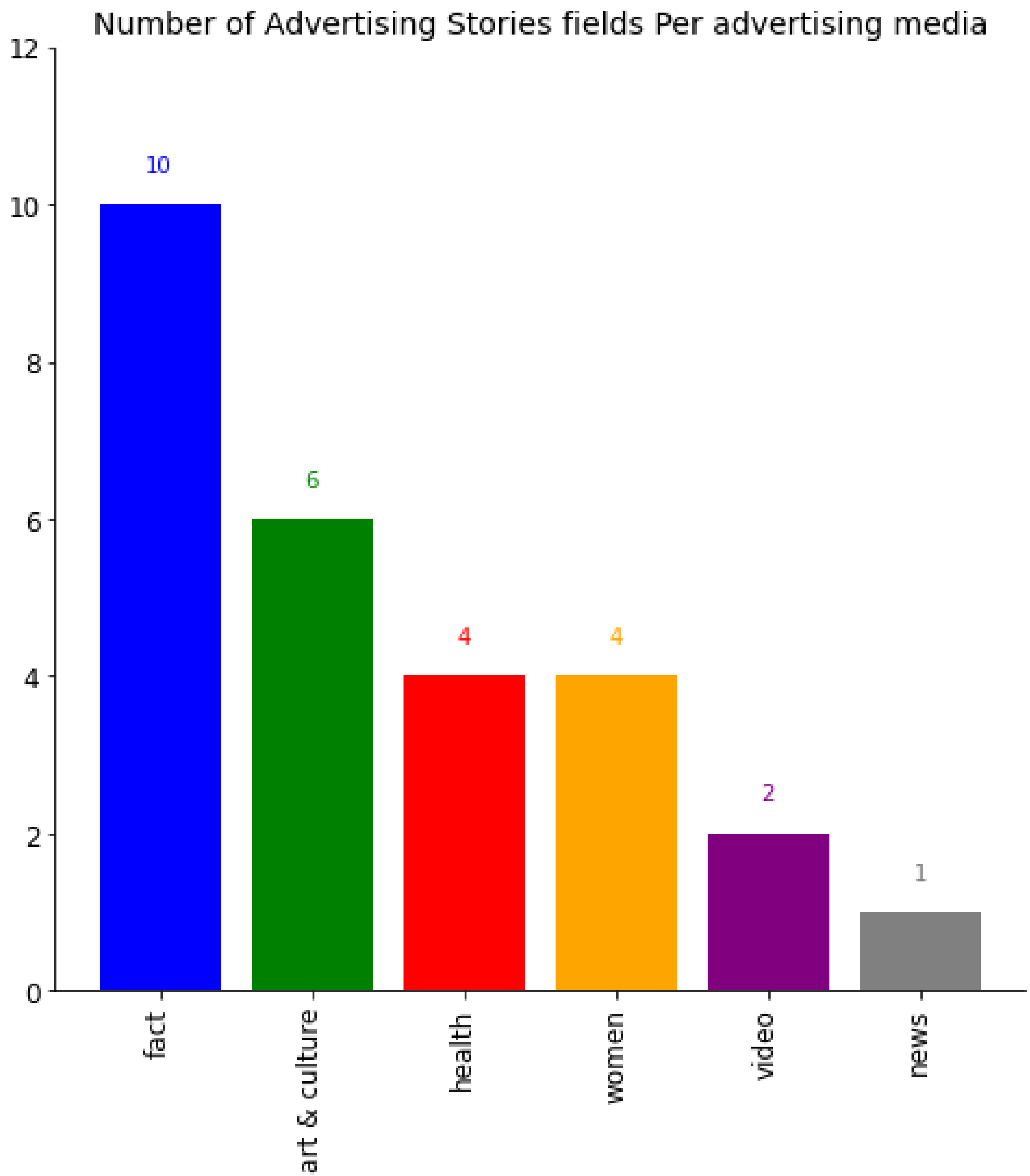
[45]:
```
                    view   follower  action  interaction  impression      cost  \
field
```

```
art & culture 52953.50 1029000.00  195.67        238.00     36994.50 356666.67
fact          87894.60 2260500.00  299.30        325.60     83649.20 553100.00
health        16256.25 1050750.00   75.25         72.25     18878.00 158116.25
news          58990.00  877000.00  234.00         90.00     58568.00 444000.00
video         51652.00 1350000.00  165.50        255.50     42506.00 315000.00
women         17959.75 1505500.00  159.00         61.00     43399.75 296633.75

               cost_per_view  price_difference  benefit
field
art & culture      423628.00          66961.33     0.83
fact               703156.80         150056.80     0.90
health             130050.00         -28066.25     0.50
news               471920.00          27920.00     1.00
video              413216.00          98216.00     1.00
women              143678.00        -152955.75     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.

[101]:
```
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. "health": 14.814814814814813 %

[48]:
```
influencer.drop(columns = ['story_no', 'l_threshold', 'h_threshold']).
↪groupby('field').mean()
```

[48]:

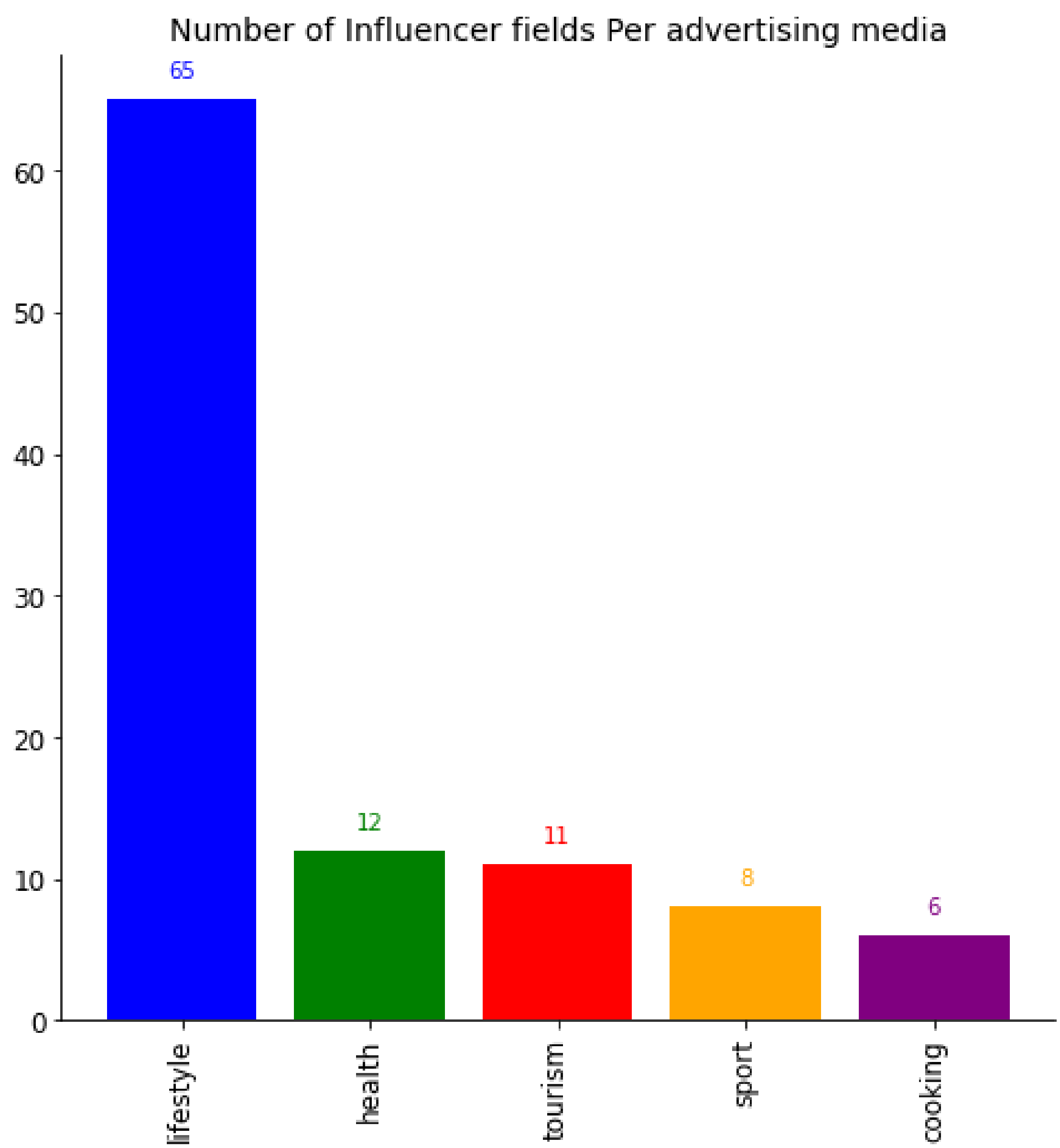
field	follower	view	action	impression	cta	interaction	cost
cooking	491000.00	50626.50	293.33	51752.50	0.67	244.00	1333333.00
health	123366.67	15702.00	262.08	16774.50	0.58	178.42	450000.00
lifestyle	206315.38	25158.05	286.02	26221.20	0.66	206.89	599999.92
sport	60000.00	2184.38	59.00	2524.38	0.38	37.88	312500.00
tourism	40545.45	7751.73	182.18	8105.55	0.45	125.36	118181.82

field	lowest_cost_per_view	highest_cost_per_view	benefit
cooking	2025060.00	3543855.00	1.00
health	548080.00	1059140.00	0.67
lifestyle	516014.15	1515909.38	-0.03
sport	87375.00	152906.25	-1.00
tourism	310069.09	542620.91	0.91

The table above is the mean of features grouped by field in minor influencers advertising, intersting 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.

[100]:
```
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, 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, 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 %
```

[50]:
```
influencer.drop(columns = ['story_no', 'l_threshold', 'h_threshold']).
 ↪groupby('gender').mean()
```

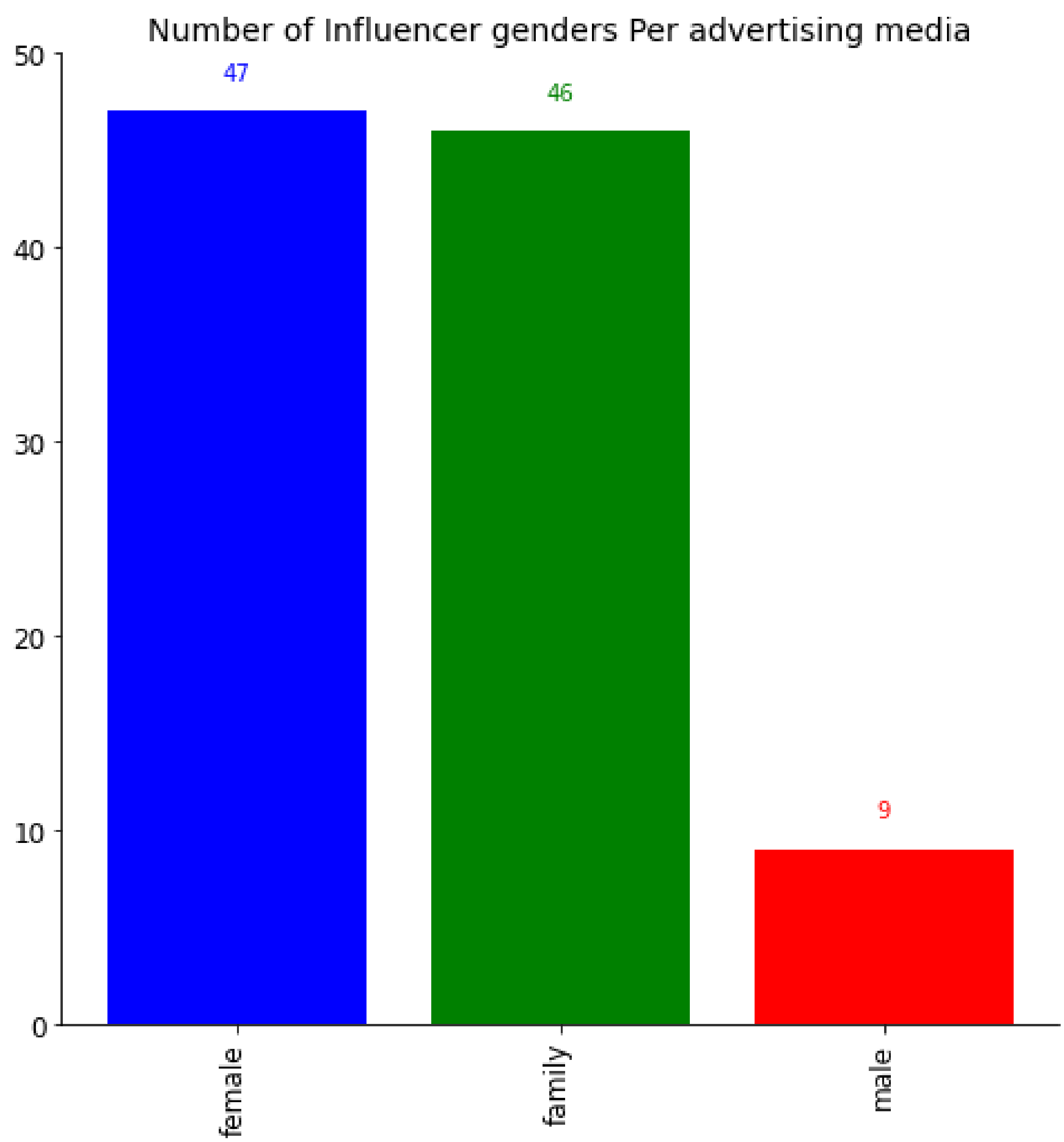
[50]:

gender	follower	view	action	impression	cta	interaction	cost	lowest_cost_per_view	highest_cost_per_view	benefit
family	268130.43	32021.26	334.52	33309.26	0.63	231.57	689130.33	640425.22	1921275.65	0.04
female	128848.94	15222.32	190.83	15877.64	0.55	140.60	461702.09	537181.28	1029706.60	0.34
male	41444.44	4641.44	179.44	5229.78	0.78	163.89	311111.11	185657.78	324901.11	-0.44

The table above is the mean of features grouped by gender in minor influencers advertising, intersting 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 benefitual but the female and family category was generally benefitual at this campaign.

[99]:
```
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 %
```

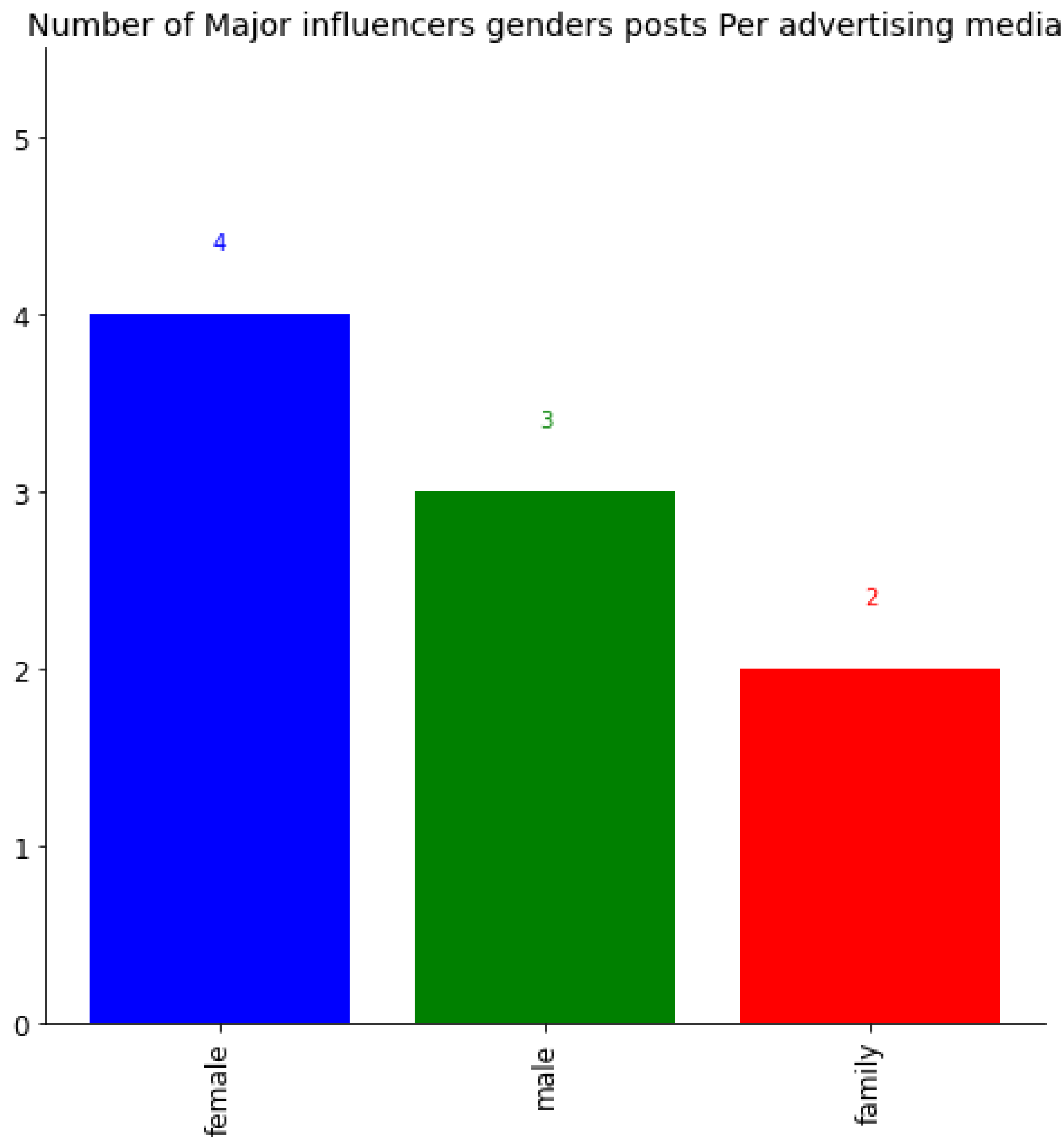
[53]:
```python
leaders_post.drop(columns = ['post_no', 'l_threshold', 'h_threshold']).
↪groupby('gender').mean()
```

[53]:
```
         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  3500000.00            2641100.00
female  122032.50   145265.50 19000000.00           16087500.00
male     69058.67    82379.33 13133333.33           10605400.00

        highest_cost_per_view  benefit
gender
family             5282200.00     0.00
female            32175000.00     0.25
male              21210800.00     0.00
```

The table above is the mean of features groued 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.

[103]:
```python
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, 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 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} %')
```




```
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 %
```

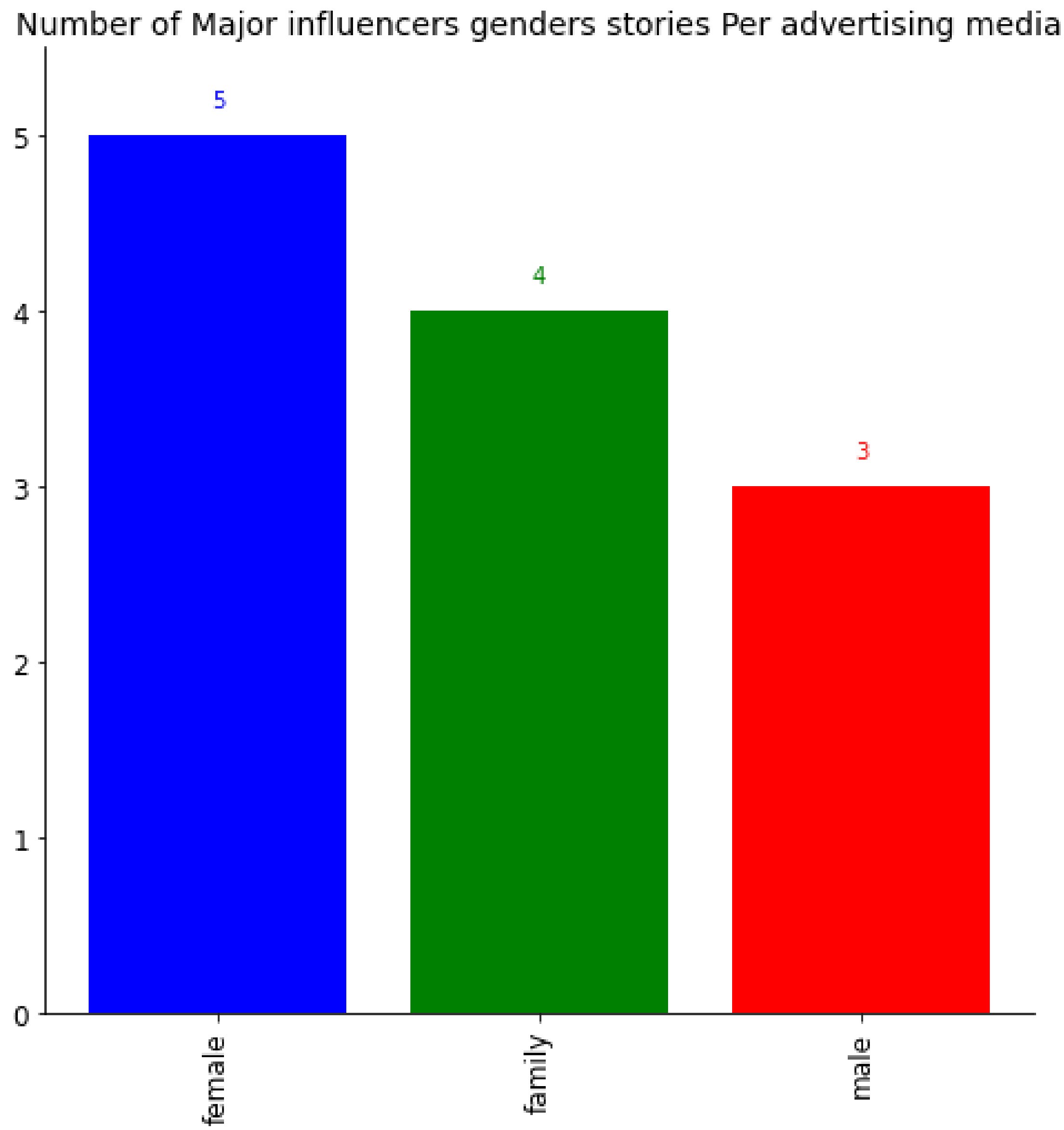
[55]:
```python
leaders_story.drop(columns = ['story_no', 'cost']).groupby('gender').mean()
```

[55]:

	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
```

[105]:
```
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} %')
```




```
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 infuencers 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.

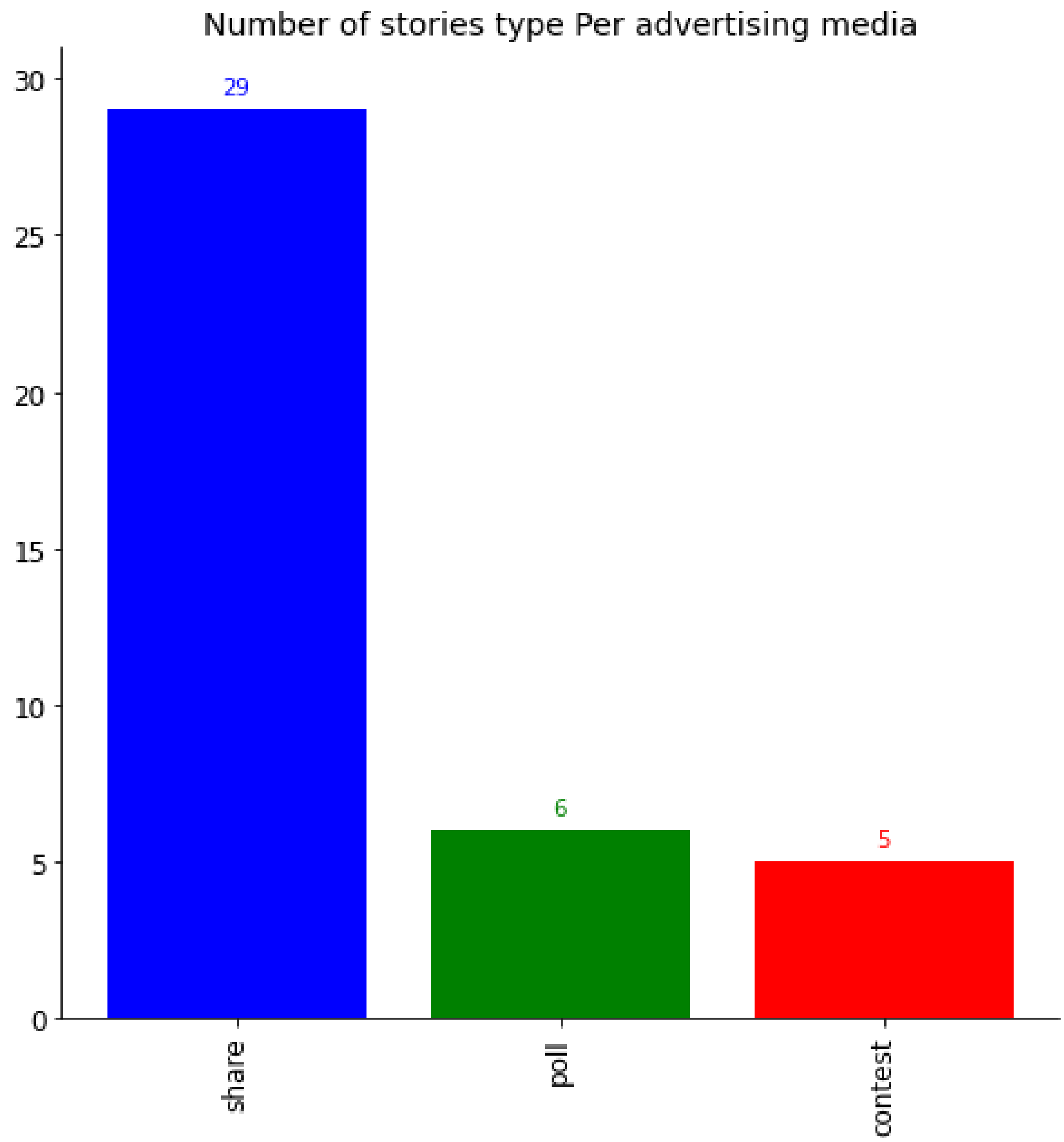
[57]:
```
story.drop(columns = ['story_no']).groupby('type').mean()
```

[57]:
```
           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.

[109]:
```
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} %')
```



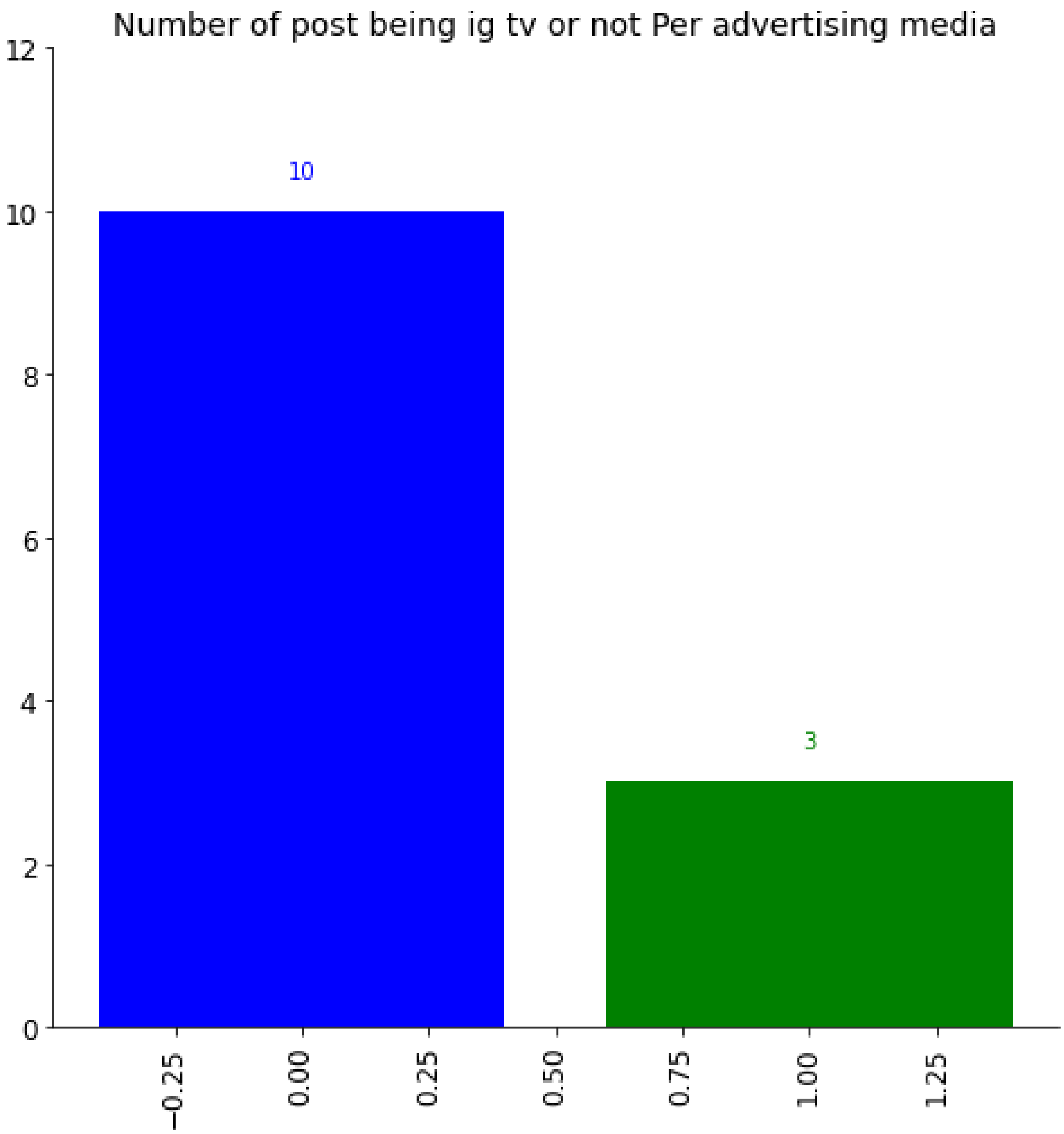

```
the top 3 types in campaign published stories and their percentages are:
1. "share": 72.5 %
2. "poll": 15.0 %
3. "contest": 12.5 %
```

[60]:
```python
post.drop(columns = ['post_no']).groupby('ig_tv').mean()
```

[60]:
```
          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 bein ig_tv or not in campaign published posts. as it's obvious ig_tv posts got much more love from followers.

[114]:
```python
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(story['type'].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": 25.0 %
2. "1": 7.5 %
```

[ ]: