```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from statsmodels.graphics.gofplots import qqplot
from sklearn.preprocessing import StandardScaler


data = pd.read_csv('./dataset_Facebook.csv',sep=';')


data.head()
```

	Page total likes	Type	Category	Post Month	Post Weekday	Post Hour	Paid	Lifetime Post Total Reach	Lifetime Post Total Impressions	Lifetime Engaged Users
0	139441	Photo	2	12	4	3	0.0	2752	5091	178
1	139441	Status	2	12	3	10	0.0	10460	19057	1457
2	139441	Photo	3	12	3	3	0.0	2413	4373	177
3	139441	Photo	2	12	2	10	1.0	50128	87991	2211
4	139441	Photo	2	12	2	3	0.0	7244	13594	671

```
data.iloc[:5,:5]
```

	Page total likes	Type	Category	Post Month	Post Weekday
0	139441	Photo	2	12	4
1	139441	Status	2	12	3
2	139441	Photo	3	12	3
3	139441	Photo	2	12	2
4	139441	Photo	2	12	2

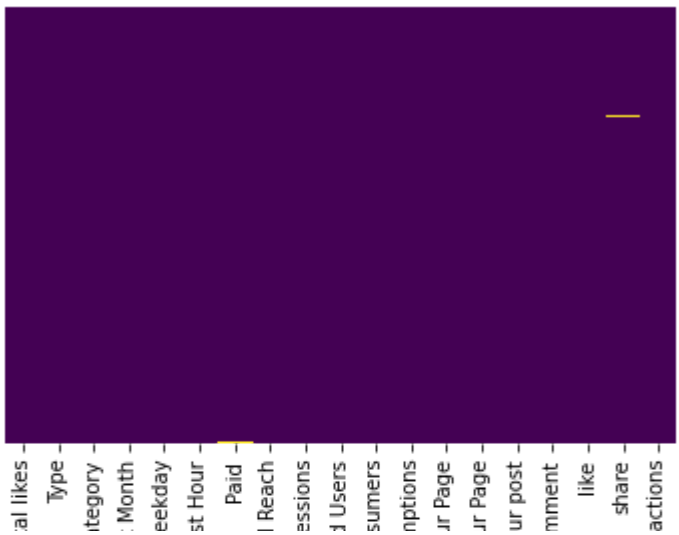
```
data.isnull()
```

	Page total likes	Type	Category	Post Month	Post Weekday	Post Hour	Paid	Lifetime Post Total Reach	Lifetime Post Total Impressions	Lifetime Engaged Users
0	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...
495	False	False	False	False	False	False	False	False	False	False
496	False	False	False	False	False	False	False	False	False	False
497	False	False	False	False	False	False	False	False	False	False
498	False	False	False	False	False	False	False	False	False	False
499	False	False	False	False	False	False	True	False	False	False

500 rows × 19 columns

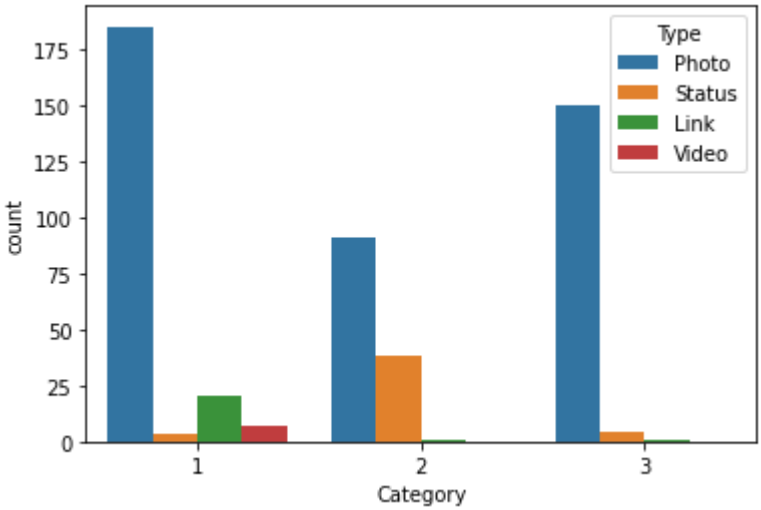
```
sns.heatmap(data.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

```
<Axes: >
```



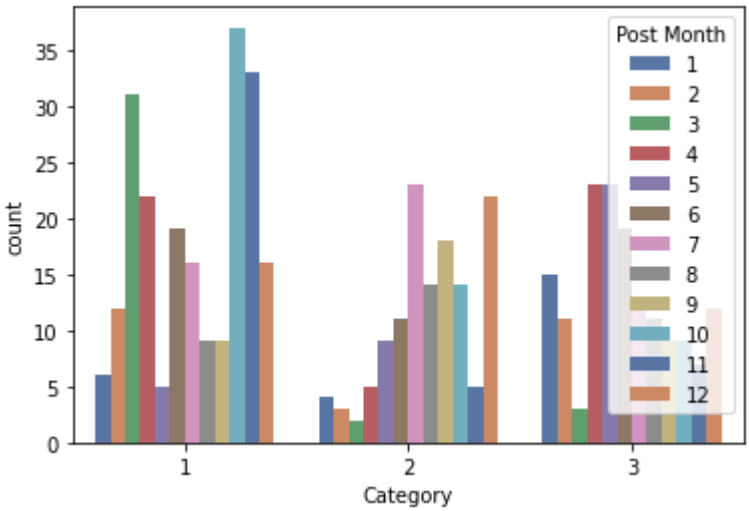

```
sns.countplot(x='Category',hue='Type',data=data,)
```

```
<Axes: xlabel='Category', ylabel='count'>
```



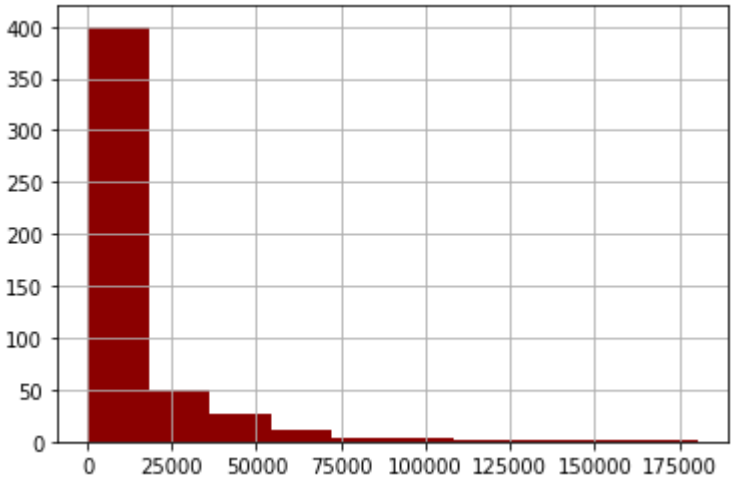

```
sns.countplot(x='Category',hue='Post Month',data=data,palette='deep')
```

```
<Axes: xlabel='Category', ylabel='count'>
```



```
data['Lifetime Post Total Reach'].hist(bins=10,color='darkred')
```

```
<Axes: >
```




```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 500 entries, 0 to 499
Data columns (total 19 columns):
 #   Column                                                    Non-Null Coun
---  ------                                                    -------------
 0   Page total likes                                          500 non-null
 1   Type                                                      500 non-null
 2   Category                                                  500 non-null
 3   Post Month                                                500 non-null
 4   Post Weekday                                              500 non-null
 5   Post Hour                                                 500 non-null
 6   Paid                                                      499 non-null
 7   Lifetime Post Total Reach                                 500 non-null
 8   Lifetime Post Total Impressions                           500 non-null
 9   Lifetime Engaged Users                                    500 non-null
 10  Lifetime Post Consumers                                   500 non-null
 11  Lifetime Post Consumptions                                500 non-null
 12  Lifetime Post Impressions by people who have liked your Page  500 non-null
 13  Lifetime Post reach by people who like your Page          500 non-null
 14  Lifetime People who have liked your Page and engaged with your post  500 non-null
 15  comment                                                   500 non-null
 16  like                                                      499 non-null
 17  share                                                     496 non-null
 18  Total Interactions                                        500 non-null
dtypes: float64(3), int64(15), object(1)
memory usage: 74.3+ KB
```

```
data['Category'].value_counts()
```

```
1    215
3    155
```

```
2    130
Name: Category, dtype: int64
```

```
data['Post Month'].value_counts()
```

```
10    60
7     52
12    50
4     50
6     49
11    45
5     37
9     36
3     36
8     34
2     26
1     25
Name: Post Month, dtype: int64
```

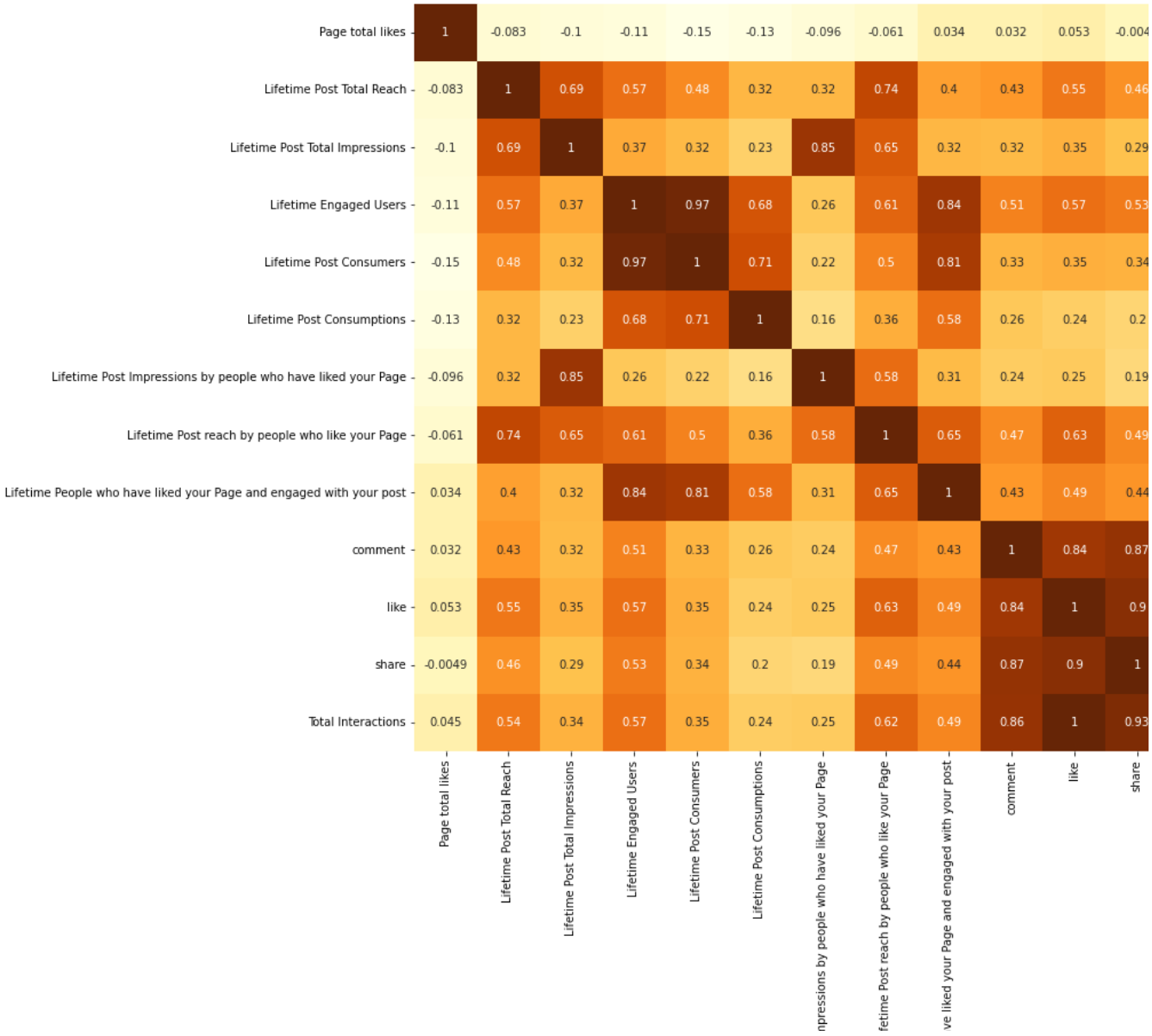
PCA

```
categorical_data = ['Type','Category','Post Month', 'Post Weekday', 'Post Hour','Paid']
```

```
data_pca = data.drop(columns=categorical_data)
```

```
plt.figure(figsize=(16,12))
sns.heatmap(data_pca.corr(),annot=True,cbar=True,cmap='YlOrBr')
```

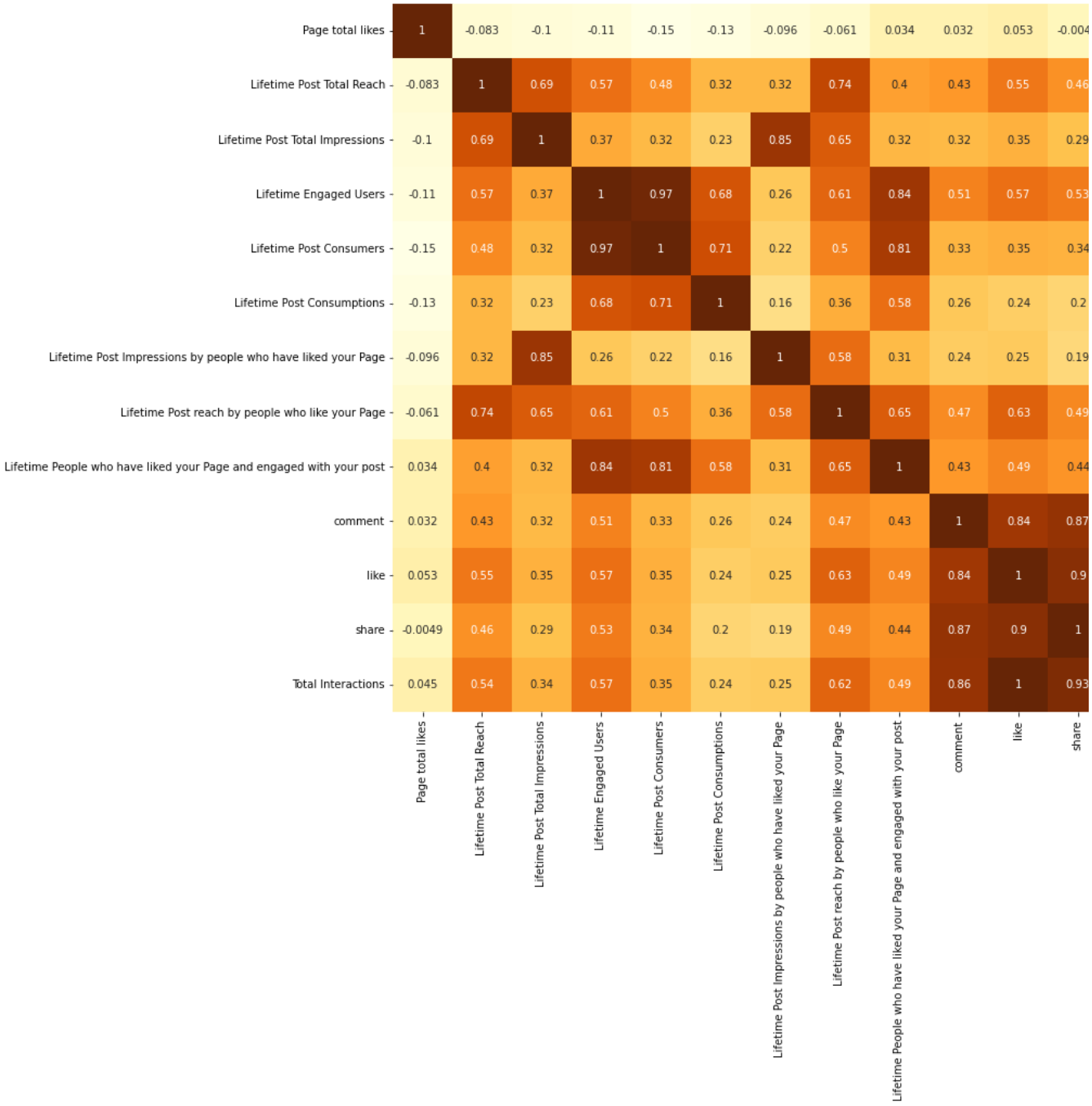
```
<Axes: >
```




```
scaler = StandardScaler()
scaler.fit(data_pca)
data_std = pd.DataFrame( scaler.transform(data_pca),columns=data_pca.columns)
```

```
plt.figure(figsize=(16,12))
sns.heatmap(data_std.corr(),annot=True,cbar=True,cmap='YlOrBr')
```

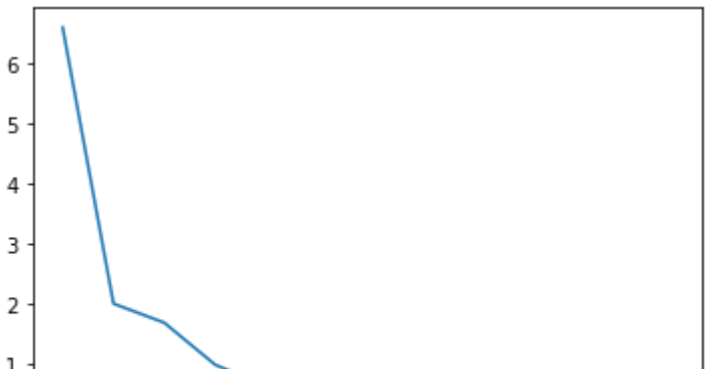
```
<Axes: >
```




```
e_values , e_vectors = np.linalg.eig(data_std.cov().values)
```

```
plt.plot(sorted(e_values,reverse=True))
```

```
[<matplotlib.lines.Line2D at 0x7f92e73ad2e0>]
```




```
np.sum(e_values[:7])/np.sum(e_values)
```

```
0.9717832602619733
```

PRE PROCESSING

```
data.dropna(inplace=True)
```

Regression

```
from sklearn import linear_model
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.feature_selection import SelectKBest
```

Preprocessing for Regression

```
outlier=np.percentile(data['Lifetime Post Total Reach'],90)
outlier
```

```
37260.8
```

```
df = data[data['Lifetime Post Total Reach']
```

```
  File "<ipython-input-139-122fcff627f1>", line 1
```

```
def Weekday(x):
    if x == 1:
        return 'Sunday'
    elif x== 2:
        return 'Monday'
    elif x == 3:
        return 'Tuesday'
    elif x == 4:
        return 'Wednesday'
    elif x == 5:
        return 'Thursday'
    elif x ==6:
        return 'Friday'
    elif x == 7:
        return "Saturday"

data['Weekday'] = data['Post Weekday'].apply(lambda x: Weekday(x))


df = pd.concat([df,pd.get_dummies(df['Weekday'])],axis=1)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-137-b485aa6dbb31> in <module>
----> 1 df = pd.concat([df,pd.get_dummies(df['Weekday'])],axis=1)

NameError: name 'df' is not defined
```

SEARCH STACK OVERFLOW

```
df = pd.concat([df,pd.get_dummies(df['Post Hour'],prefix='hour')],axis=1)
df = pd.concat([df,pd.get_dummies(df['Post Month'],prefix='Month')],axis=1)
df['Video'] = pd.get_dummies(df['Type'])['Video']
df['Status'] = pd.get_dummies(df['Type'])['Status']
df['Photo'] = pd.get_dummies(df['Type'])['Photo']
df['Category_1'] = pd.get_dummies(df['Category'])[1]
df['Category_2'] = pd.get_dummies(df['Category'])[2]
```

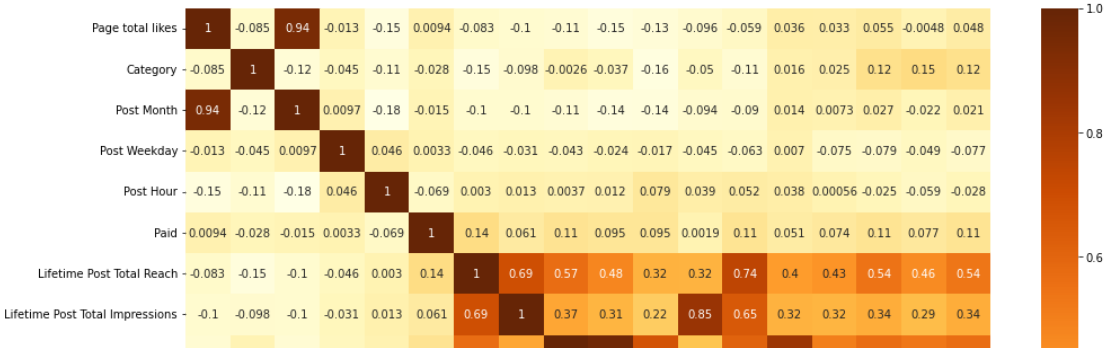
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-138-7b798b5ebbc0> in <module>
----> 1 df = pd.concat([df,pd.get_dummies(df['Post Hour'],prefix='hour')],axis=1)
      2 df = pd.concat([df,pd.get_dummies(df['Post Month'],prefix='Month')],axis=1)
      3 df['Video'] = pd.get_dummies(df['Type'])['Video']
      4 df['Status'] = pd.get_dummies(df['Type'])['Status']
      5 df['Photo'] = pd.get_dummies(df['Type'])['Photo']

NameError: name 'df' is not defined
```

SEARCH STACK OVERFLOW

EDA

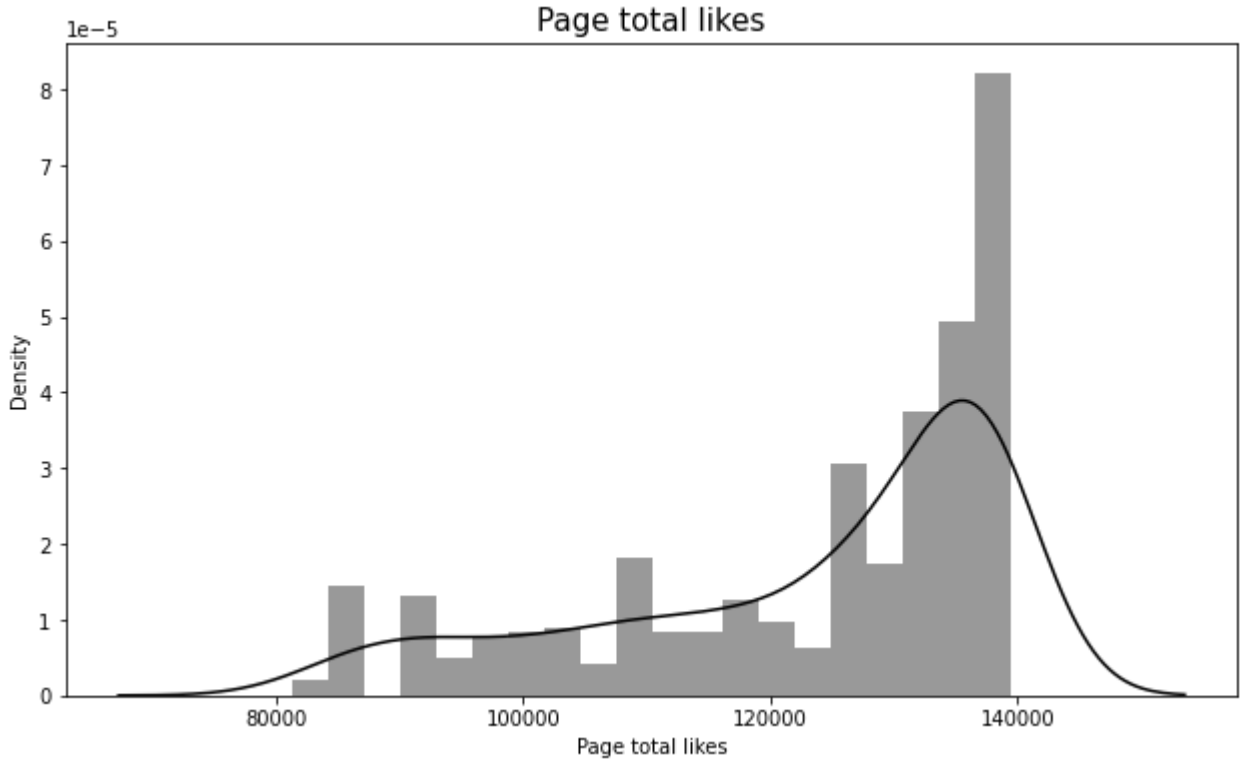
```
plt.figure(figsize=(16,12))
sns.heatmap(data.corr(),annot=True,cbar=True,cmap='YlOrBr')
```

```
<Axes: >
```



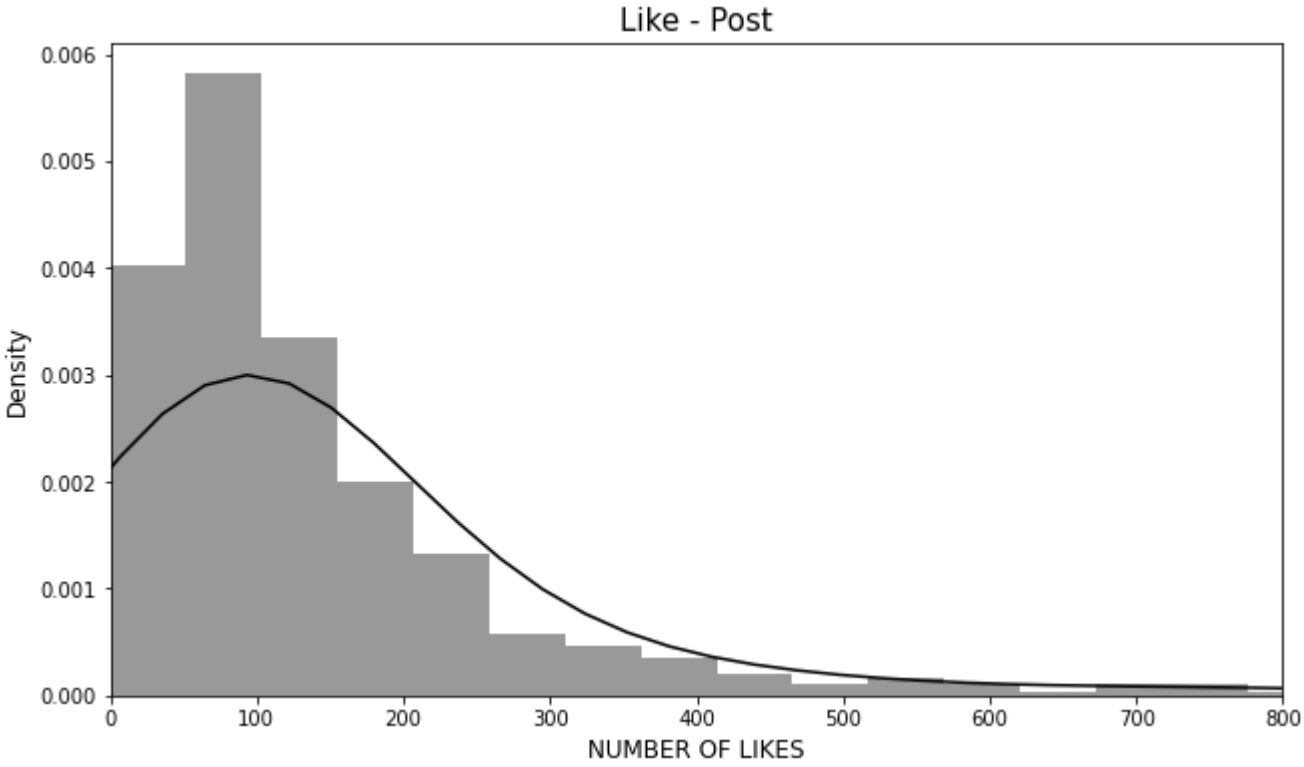

```
plt.figure(figsize=(10.5,6))
sns.distplot(data['Page total likes'],bins=20,kde=True,color="black")
# plt.xlim([6000,14000])
plt.title("Page total likes",fontsize=15)
```

```
<ipython-input-67-dd26b3035c1a>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Page total likes'],bins=20,kde=True,color="black")
Text(0.5, 1.0, 'Page total likes')
```



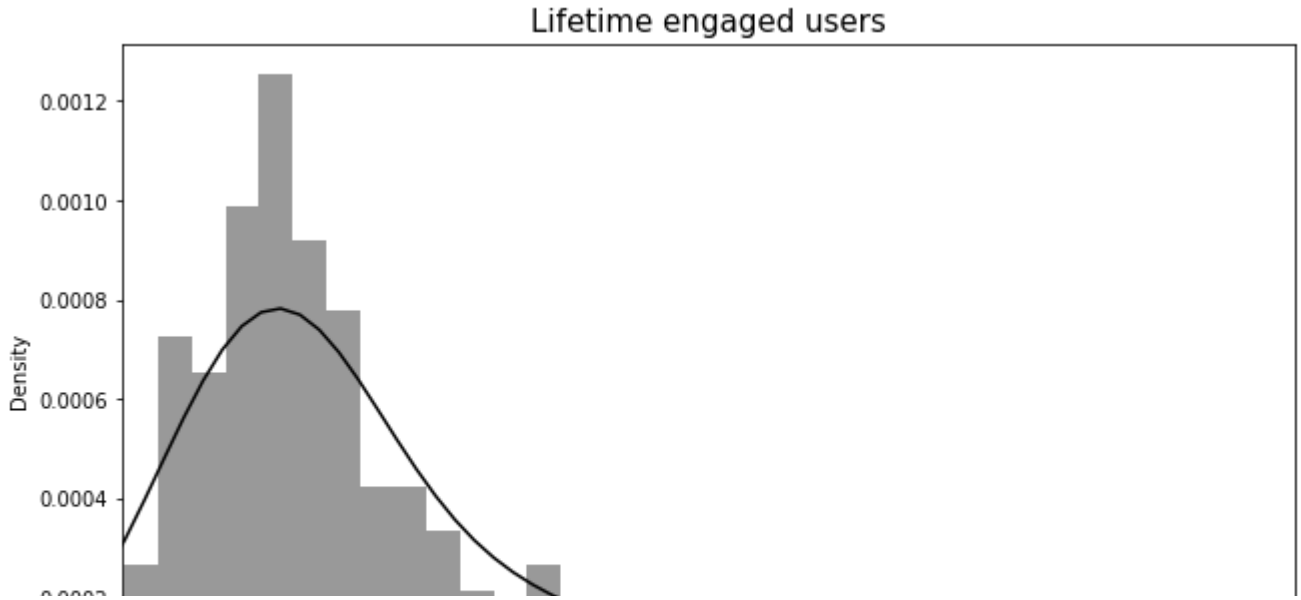
```
plt.figure(figsize=(10.5,6))
sns.distplot(data['like'],bins=100,color='black',kde=True)
plt.xlim(0,800)
plt.xlabel("NUMBER OF LIKES",fontsize=12)
plt.ylabel('Density',fontsize=12)
plt.title('Like - Post',fontsize=15)
```

```
<ipython-input-68-7383435df19e>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['like'],bins=100,color='black',kde=True)
Text(0.5, 1.0, 'Like - Post')
```




```
plt.figure(figsize=(10.5,6))
sns.distplot(data['Lifetime Engaged Users'],bins=100,color='black')
plt.xlim(0,4000)
plt.title('Lifetime engaged users',fontsize=15)
```

```
<ipython-input-69-c526363965f1>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Lifetime Engaged Users'],bins=100,color='black')
Text(0.5, 1.0, 'Lifetime engaged users')
```

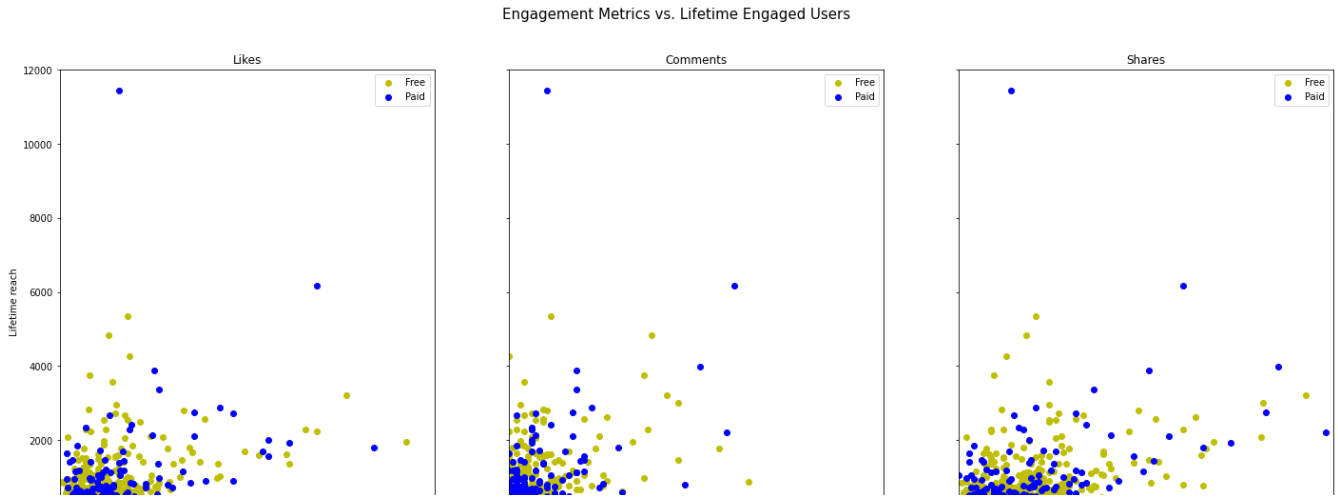



```
plt.figure(figsize=(10.5,6))
sns.distplot(data['Lifetime Post Total Reach'],bins=200,color='black')
plt.xlim(0,100000)
```

```
<ipython-input-70-614c7488a561>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Lifetime Post Total Reach'],bins=200,color='black')
(0.0, 100000.0)
```

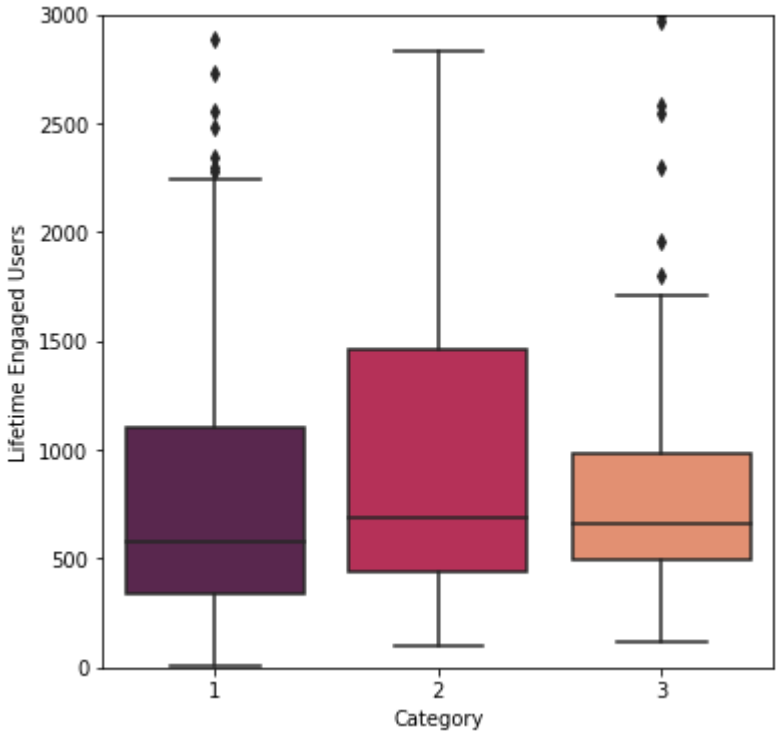
```
fig, ax = plt.subplots(ncols=3,nrows=1,sharey=True,figsize=(24,9))

paid = data[data['Paid']==1]
free = data[data['Paid']==0]

ax[0].scatter(free['like'],free['Lifetime Engaged Users'],color='y')
ax[0].scatter(paid['like'],paid['Lifetime Engaged Users'],color='b')
ax[0].set_title('Likes')
ax[0].set_xlim(0,1250)
ax[0].legend(labels=['Free','Paid'])

ax[1].scatter(free['comment'],free['Lifetime Engaged Users'],color='y')
ax[1].scatter(paid['comment'],paid['Lifetime Engaged Users'],color='b')
ax[1].set_title('Comments')
ax[1].set_xlim(0,100)
ax[1].legend(labels=['Free','Paid'])

ax[2].scatter(free['share'],free['Lifetime Engaged Users'],color='y')
ax[2].scatter(paid['share'],paid['Lifetime Engaged Users'],color='b')
ax[2].set_title('Shares')
ax[2].set_xlim(0,150)
ax[2].legend(labels=['Free','Paid'])

ax[0].set_ylabel("Lifetime reach")

fig.suptitle('Engagement Metrics vs. Lifetime Engaged Users',fontsize=15)
```

```
Text(0.5, 0.98, 'Engagement Metrics vs. Lifetime Engaged Users')
```



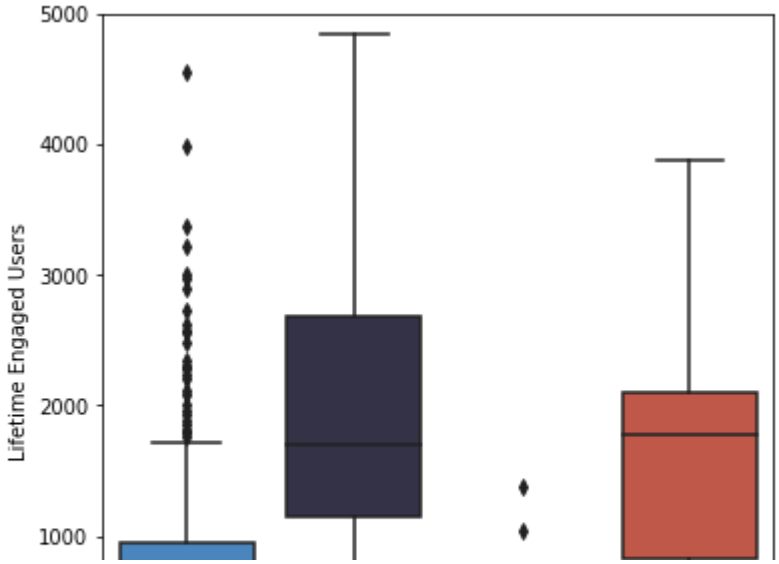

```
plt.figure(figsize=(6,6))
sns.boxplot(x=data['Category'],y=data['Lifetime Engaged Users'],palette='rocket')
plt.ylim(0,3000)
```

```
(0.0, 3000.0)
```



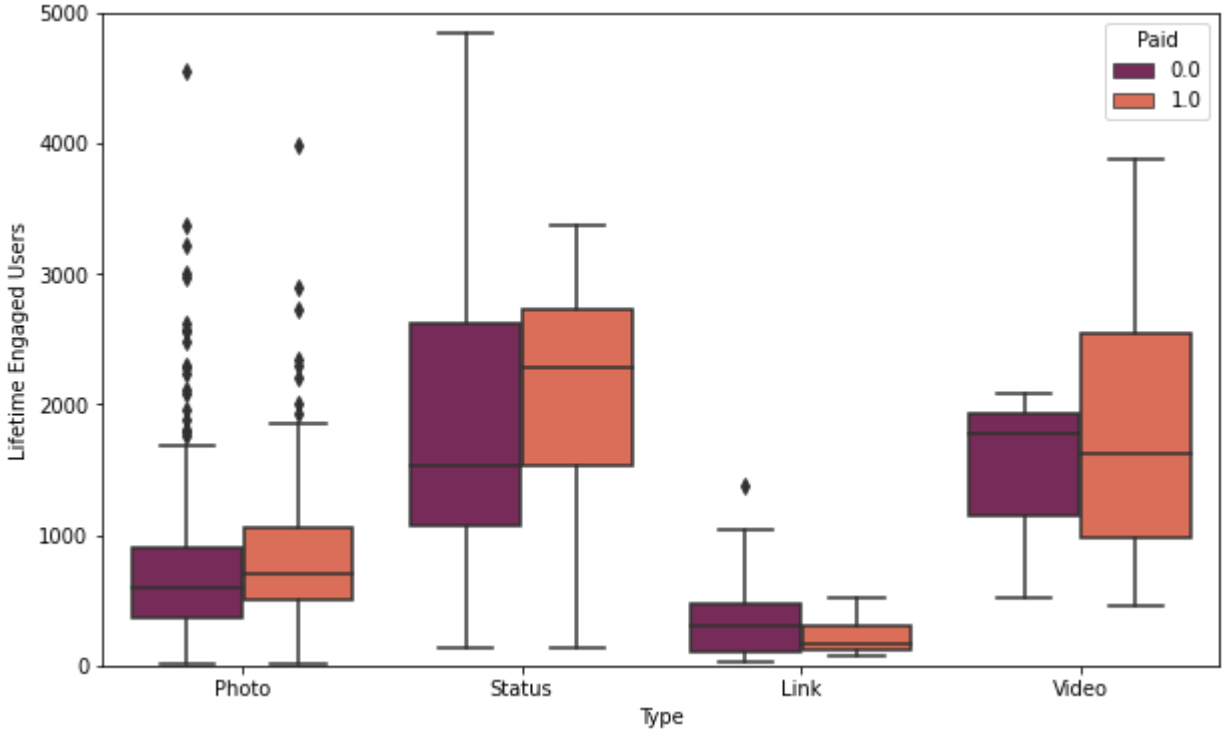

```
plt.figure(figsize=(6,6))
sns.boxplot(x=data['Type'],y=data['Lifetime Engaged Users'],palette='icefire')
plt.ylim(0,5000)
```

```
(0.0, 5000.0)
```



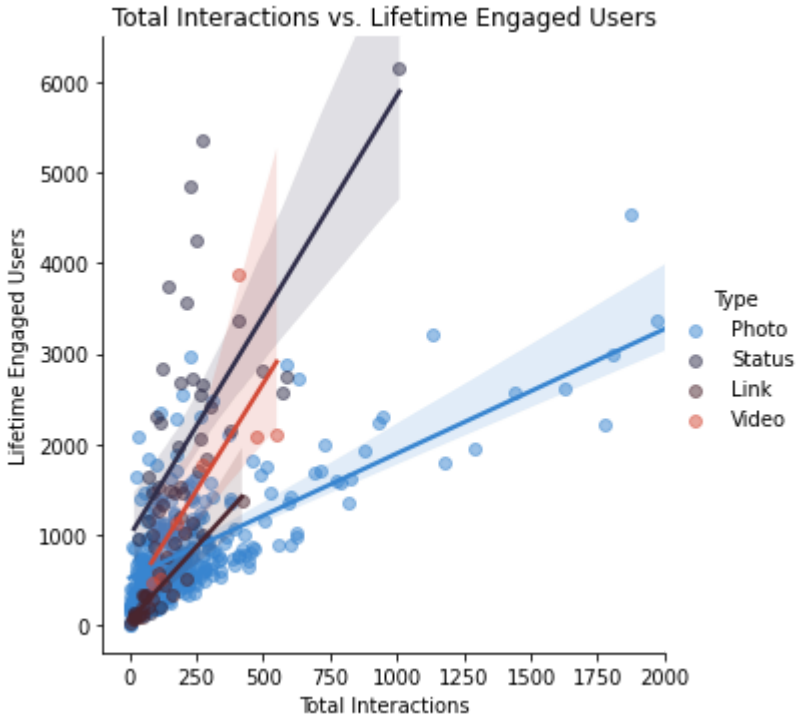

```
plt.figure(figsize=(10,6))
sns.boxplot(x=data['Type'],y=data['Lifetime Engaged Users'],hue=data['Paid'],palette='rocket'
plt.ylim(0,5000)
```

```
(0.0, 5000.0)
```



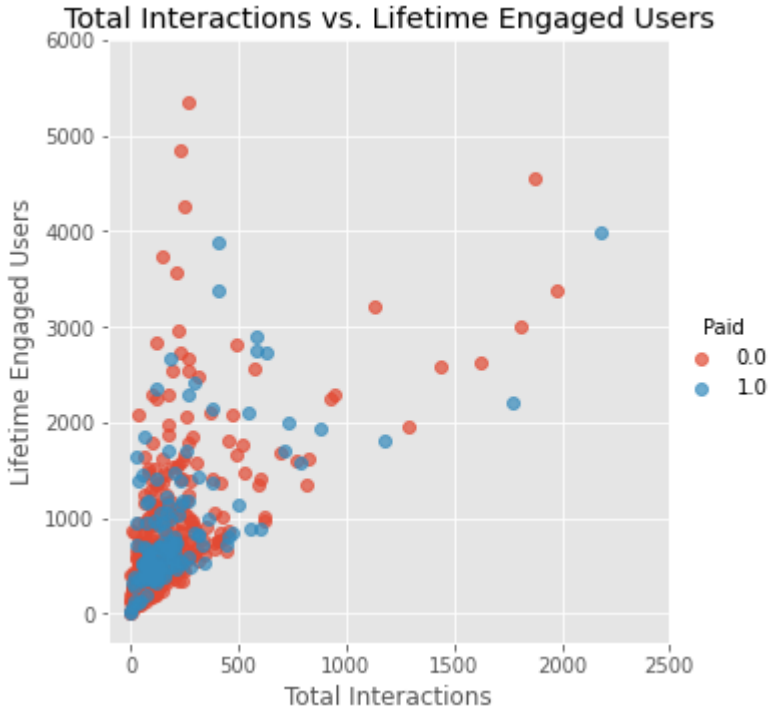

```
sns.lmplot(x='Total Interactions',y='Lifetime Engaged Users',
           hue='Type',data=data,scatter_kws= {'alpha':0.5},palette='icefire')
plt.title('Total Interactions vs. Lifetime Engaged Users')
plt.xlim(-100,2000)
plt.ylim(-300,6500)
```

```
(-300.0, 6500.0)
```



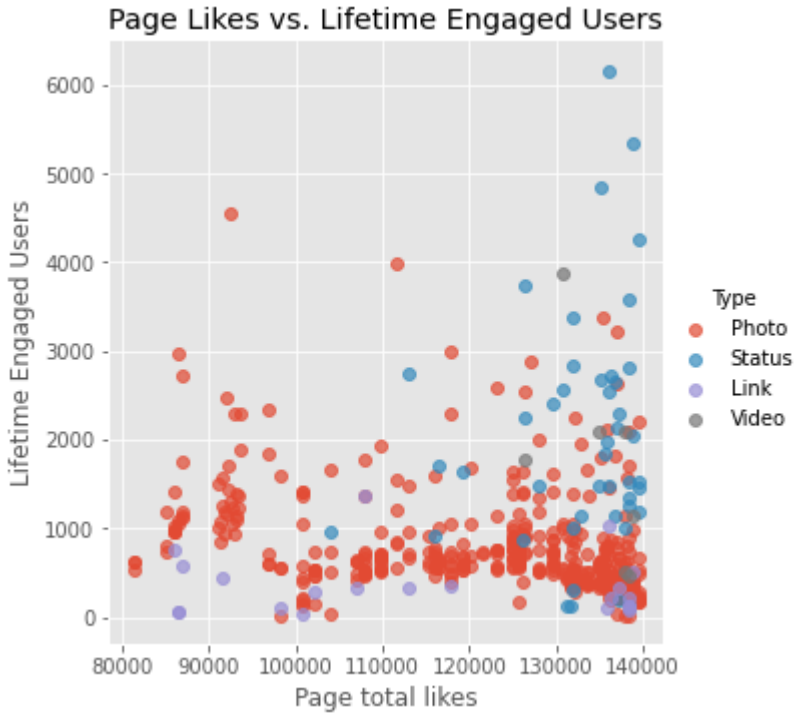

```
plt.style.use('ggplot')
sns.lmplot(x='Total Interactions',y='Lifetime Engaged Users',
           hue='Paid',data=data,fit_reg=False,scatter_kws= {'alpha':0.7})
plt.title('Total Interactions vs. Lifetime Engaged Users')
plt.xlim(-100,2500)
plt.ylim(-300,6000)
```

```
(-300.0, 6000.0)
```



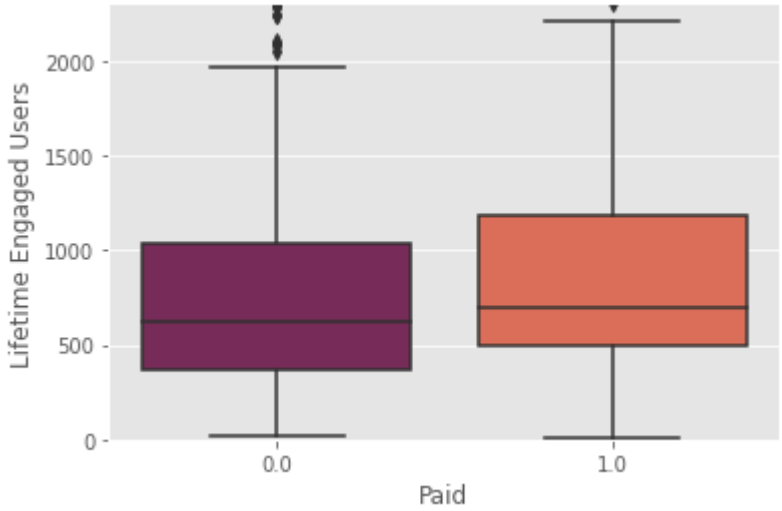
```
plt.style.use('ggplot')
sns.lmplot(x='Page total likes',y='Lifetime Engaged Users',
           hue='Type',data=data,fit_reg=False,scatter_kws= {'alpha':0.7},)
plt.title('Page Likes vs. Lifetime Engaged Users')
# plt.xlim(-100,140000)
plt.ylim(-300,6500)
```

```
(-300.0, 6500.0)
```




```
sns.boxplot(x=data['Paid'],y=data['Lifetime Engaged Users'],palette='rocket')
plt.ylim(0,2300)
```

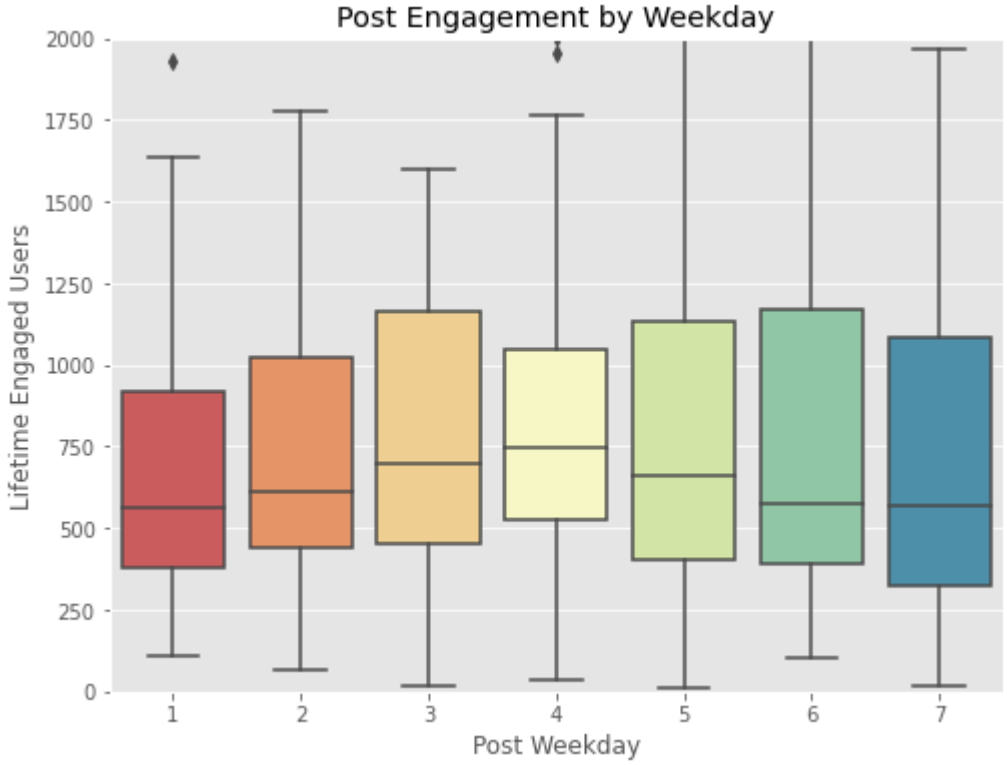
```
(0.0, 2300.0)
```




```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Post Weekday'],y=data['Lifetime Engaged Users'],palette='Spectral')
```

```
plt.ylim(0,2000)
plt.title("Post Engagement by Weekday")
```

```
Text(0.5, 1.0, 'Post Engagement by Weekday')
```

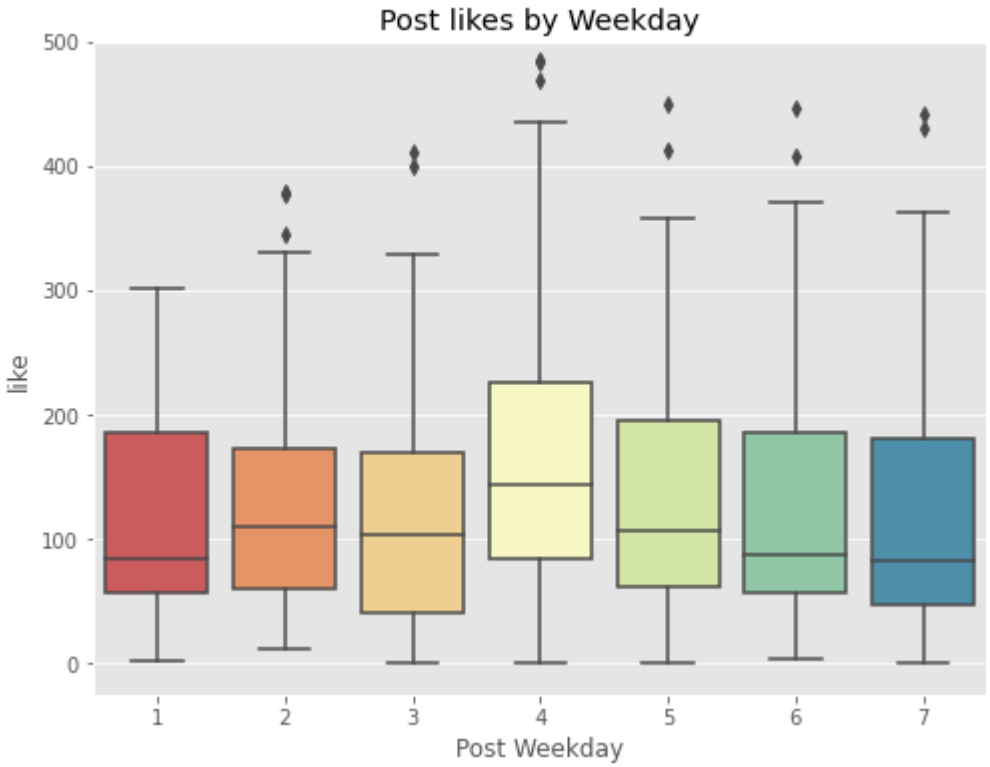



```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Post Weekday'],y=data['Lifetime Engaged Users'],hue=data['Paid'],palette=
plt.ylim(0,2000)
plt.title("Post Engagement by Weekday")
```

```
Text(0.5, 1.0, 'Post Engagement by Weekday')
```

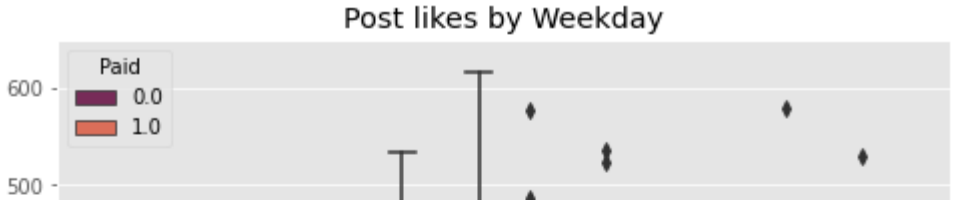

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Post Weekday'],y=data['like'],palette='Spectral')
plt.ylim(-25,500)
plt.title("Post likes by Weekday")
```

```
Text(0.5, 1.0, 'Post likes by Weekday')
```



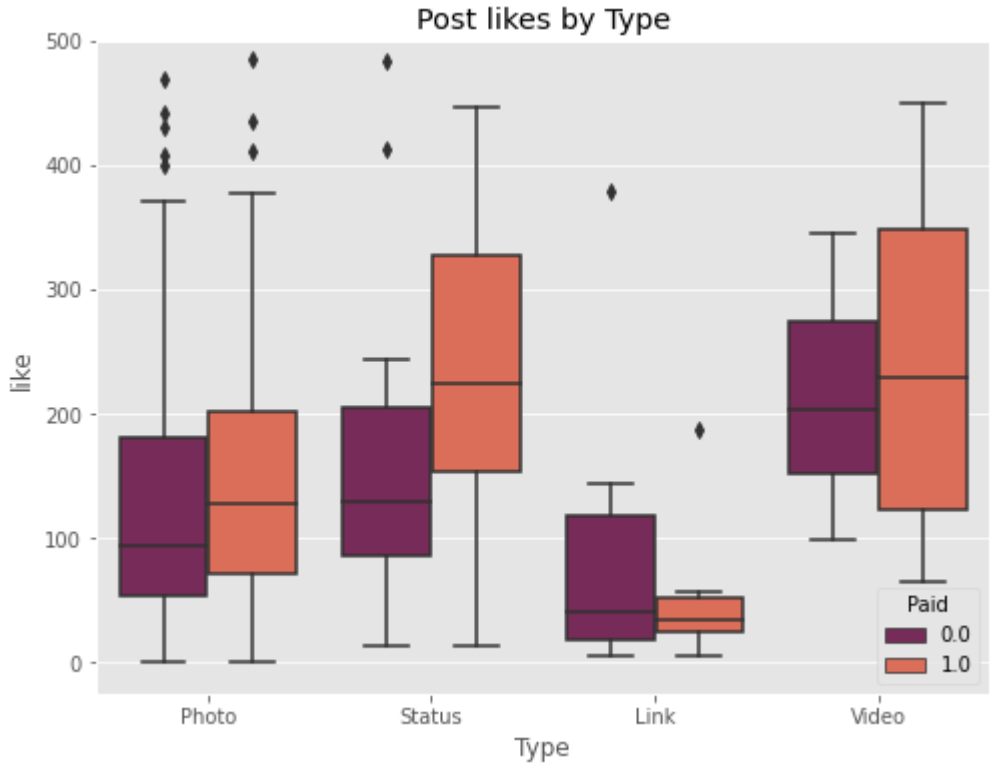

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Post Weekday'],y=data['like'],hue=data['Paid'],palette='rocket')
plt.ylim(-25,650)
plt.title("Post likes by Weekday")
```

```
Text(0.5, 1.0, 'Post likes by Weekday')
```



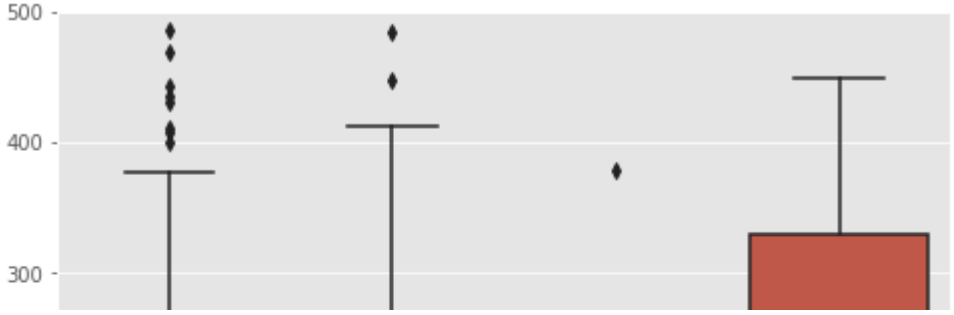

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Type'],y=data['like'],hue=data['Paid'],palette='rocket')
plt.ylim(-25,500)
plt.title("Post likes by Type")
```

```
Text(0.5, 1.0, 'Post likes by Type')
```



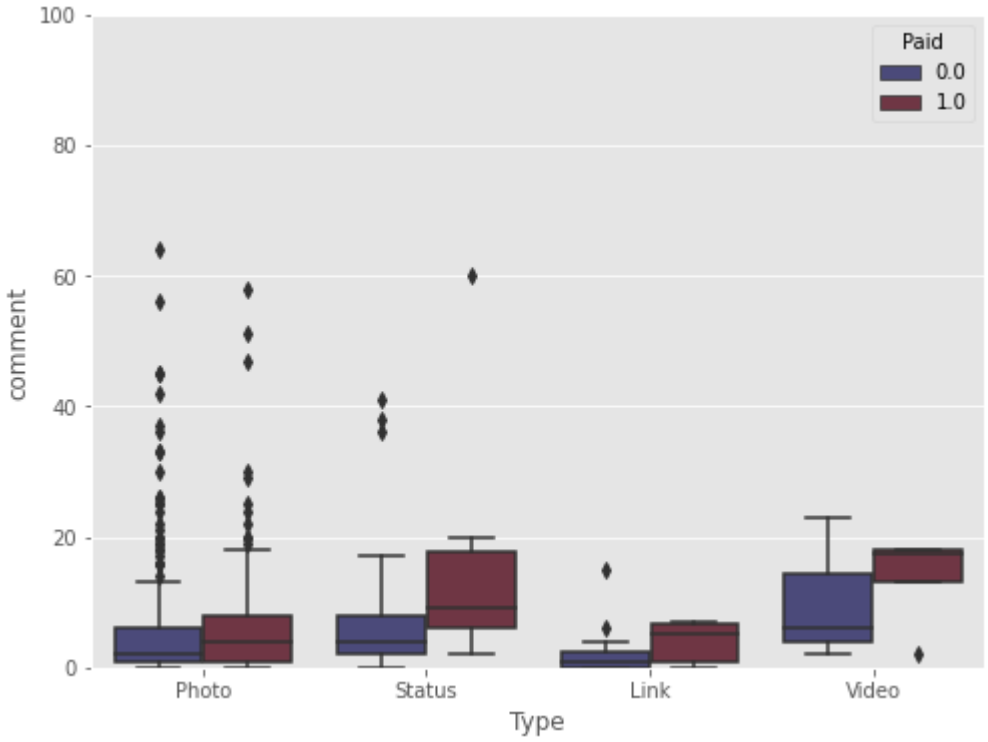

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Type'],y=data['like'],palette='icefire')
plt.ylim(0,500)
```

```
(0.0, 500.0)
```



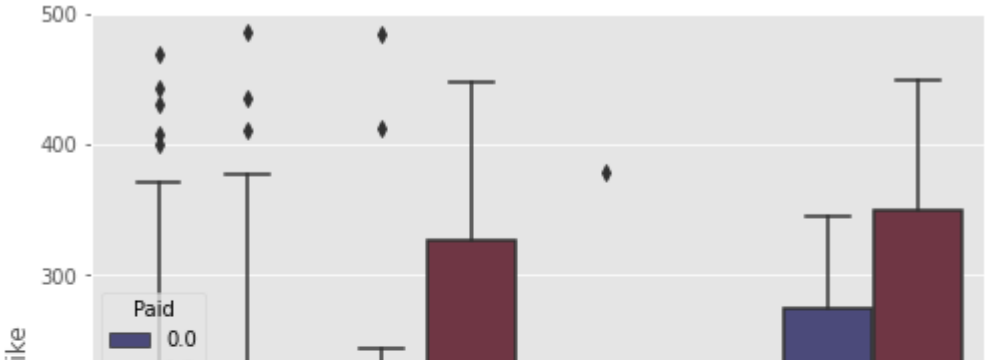

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Type'],y=data['comment'],hue=data['Paid'],palette='icefire')
plt.ylim(0,100)
```

```
(0.0, 100.0)
```



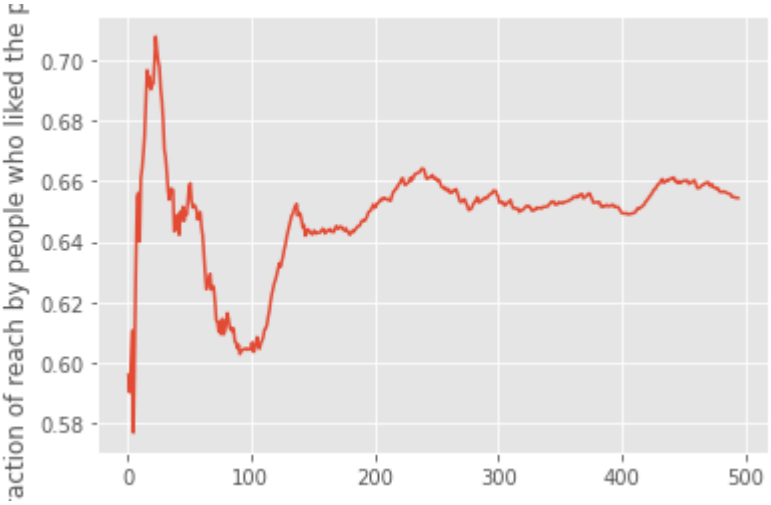

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Type'],y=data['like'],hue=data['Paid'],palette='icefire')
plt.ylim(0,500)
```

```
(0.0, 500.0)
```



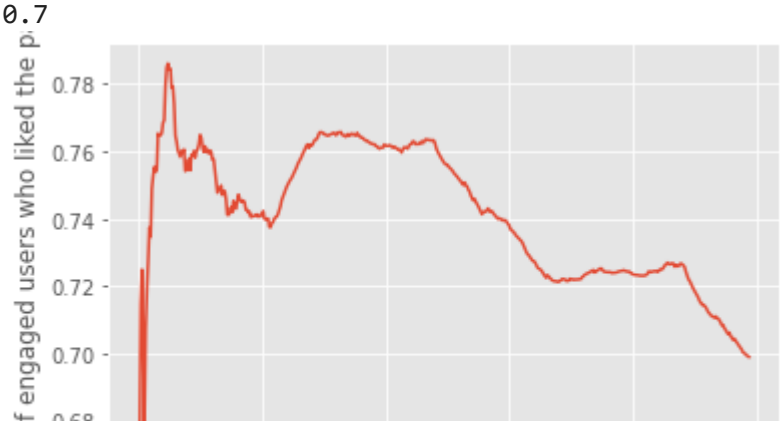

```
a = data['Lifetime Post reach by people who like your Page']/data['Lifetime Post Total Reach'
new = []
for i in range(len(a)):
  new.append(np.mean(a[:i]))
plt.ylabel('Fraction of reach by people who liked the page')
plt.plot(new)
print(f"{np.mean(a)}")
```

```
0.6543343852701147
```



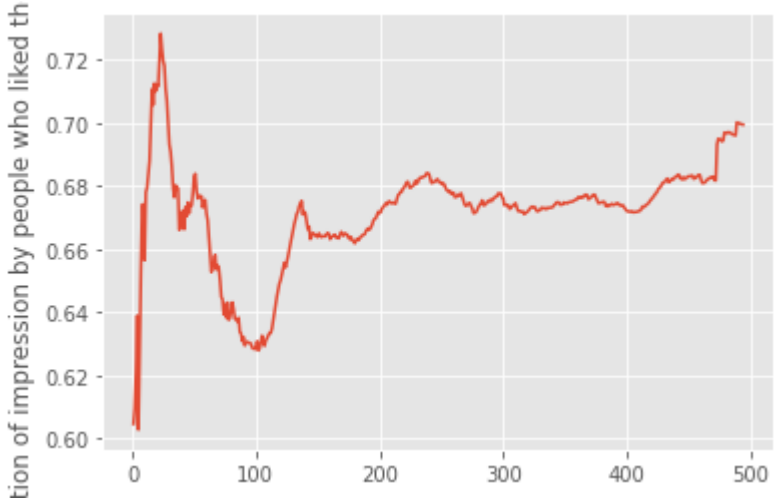

```
a = data['Lifetime People who have liked your Page and engaged with your post']/data['Lifetim

new = []
for i in range(len(a)):
  new.append(np.mean(a[:i]))

plt.ylabel('Fraction of engaged users who liked the page')

plt.plot(new)
print(f"{round(np.mean(a),2)}")
```

```
0.7
```



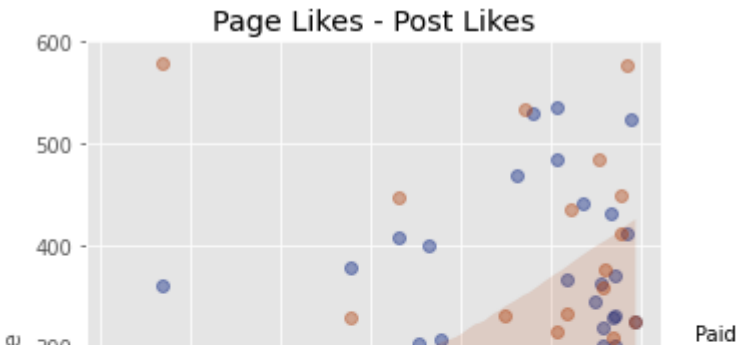

```
a = data['Lifetime Post Impressions by people who have liked your Page']/data['Lifetime Post
new = []
for i in range(len(a)):
  new.append(np.mean(a[:i]))
plt.ylabel('Fraction of impression by people who liked the page')
plt.plot(new)
print(f"{np.mean(a)}")
```

```
0.6992567097851073
```




```
sns.lmplot(x='Page total likes',y='like',
           hue='Paid',data=data,scatter_kws= {'alpha':0.4},palette='dark')
plt.title('Page Likes - Post Likes')
plt.ylim(0,600)
```

```
(0.0, 600.0)
```




```
data
```

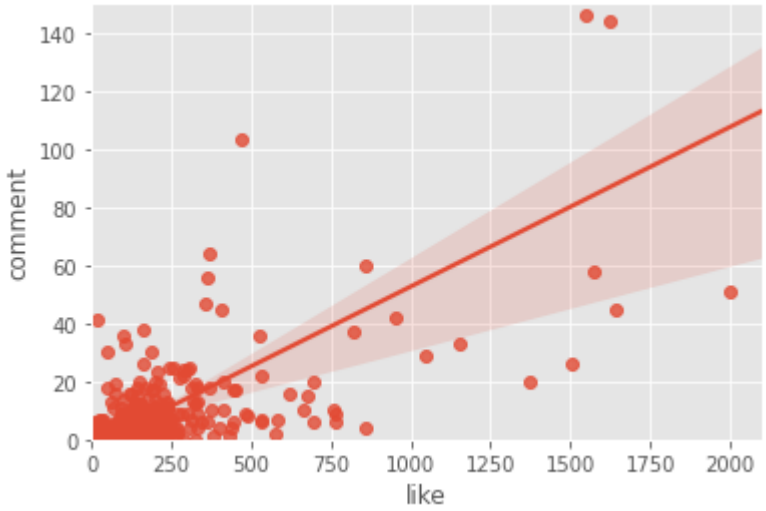
	Page total likes	Type	Category	Post Month	Post Weekday	Post Hour	Paid	Lifetime Post Total Reach	Lifetime Post Total Impressions	Lifetin Engage User
0	139441	Photo	2	12	4	3	0.0	2752	5091	17
1	139441	Status	2	12	3	10	0.0	10460	19057	145
2	139441	Photo	3	12	3	3	0.0	2413	4373	17
3	139441	Photo	2	12	2	10	1.0	50128	87991	221
4	139441	Photo	2	12	2	3	0.0	7244	13594	67
...	...	...	...	...	...	...	...	...	...	
494	85093	Photo	3	1	7	10	0.0	5400	9218	81
495	85093	Photo	3	1	7	2	0.0	4684	7536	73
496	81370	Photo	2	1	5	8	0.0	3480	6229	53
497	81370	Photo	1	1	5	2	0.0	3778	7216	62
498	81370	Photo	3	1	4	11	0.0	4156	7564	62

495 rows × 20 columns

Relation between Like Comment and Share

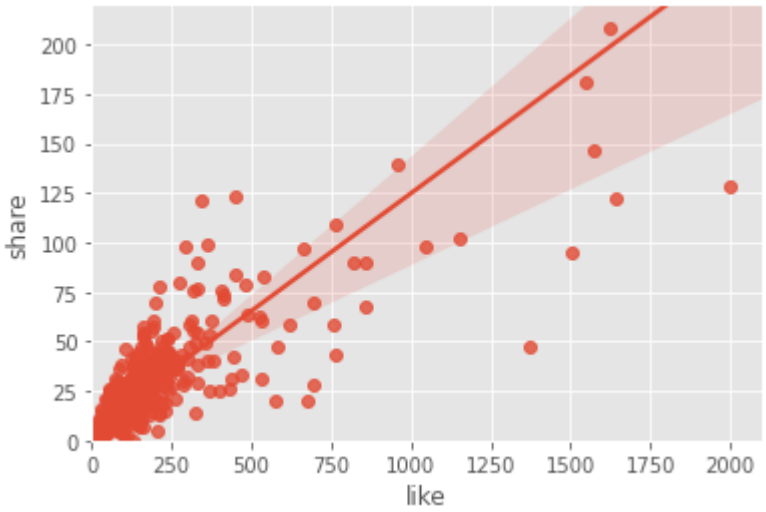
```
sns.regplot(x="like",y="comment",data=data);
plt.ylim(0,150)
plt.xlim(0,2100)
```

```
(0.0, 2100.0)
```



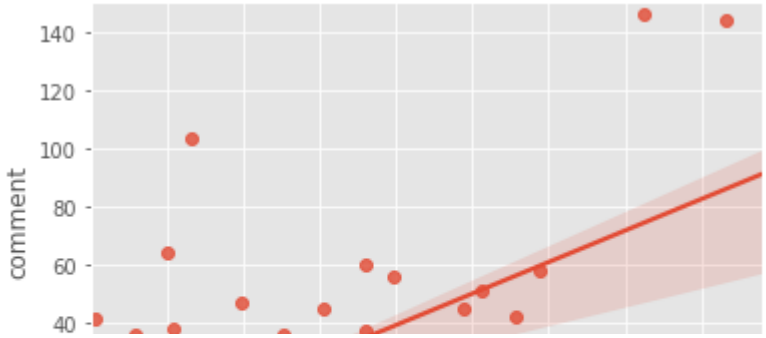

```
sns.regplot(x="like",y="share",data=data);
plt.xlim(0,2100)
plt.ylim(0,220)
```

```
(0.0, 220.0)
```



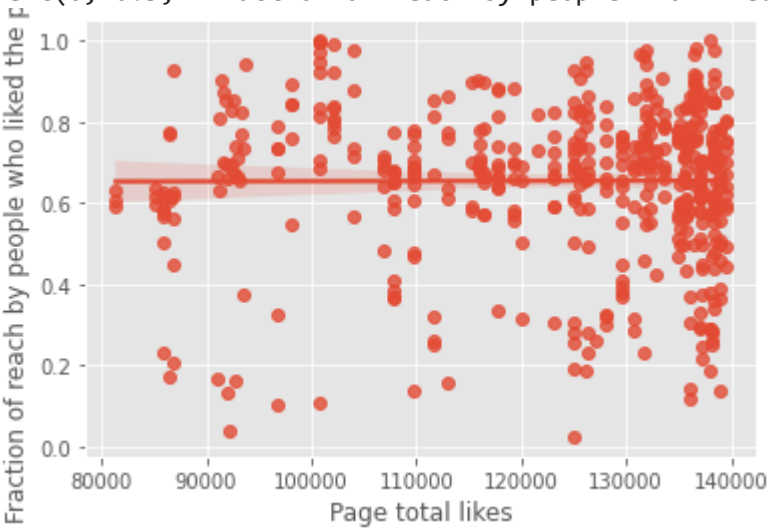

```
sns.regplot(x="share",y="comment",data=data);
plt.ylim(0,150)
plt.xlim(0,220)
```

```
(0.0, 220.0)
```



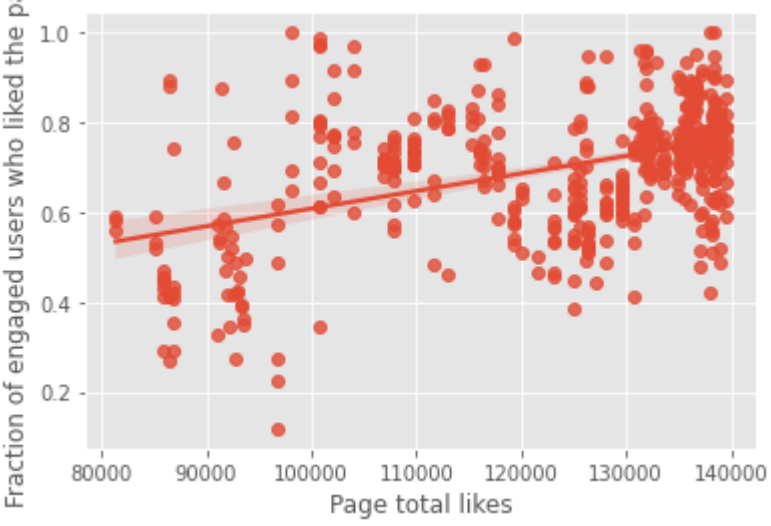

```
a = data['Lifetime Post reach by people who like your Page']/data['Lifetime Post Total Reach'
sns.regplot(x="Page total likes",y=a, data=data);
plt.ylabel('Fraction of reach by people who liked the page')
```

```
Text(0, 0.5, 'Fraction of reach by people who liked the page')
```



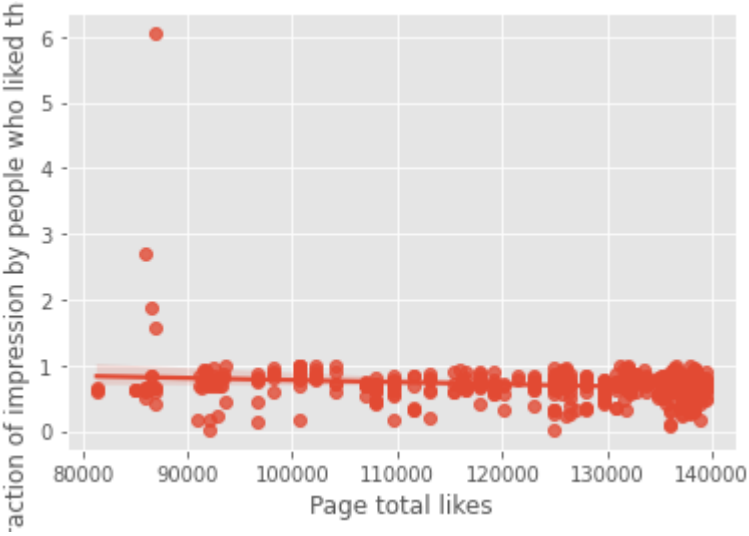

```
a = data['Lifetime People who have liked your Page and engaged with your post']/data['Lifetim
sns.regplot(x="Page total likes",y=a, data=data);
plt.ylabel('Fraction of engaged users who liked the page')
```

```
Text(0, 0.5, 'Fraction of engaged users who liked the page')
```



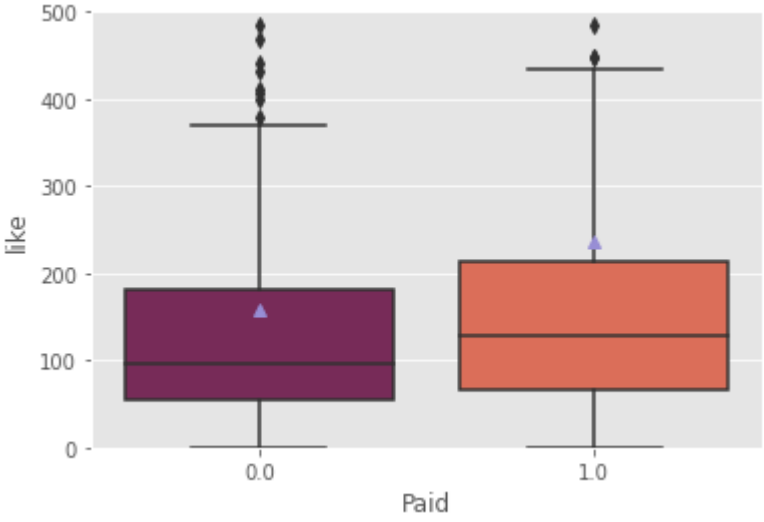
```
a = data['Lifetime Post Impressions by people who have liked your Page']/data['Lifetime Post
sns.regplot(x="Page total likes",y=a, data=data);
plt.ylabel('Fraction of impression by people who liked the page')
```

```
Text(0, 0.5, 'Fraction of impression by people who liked the page')
```



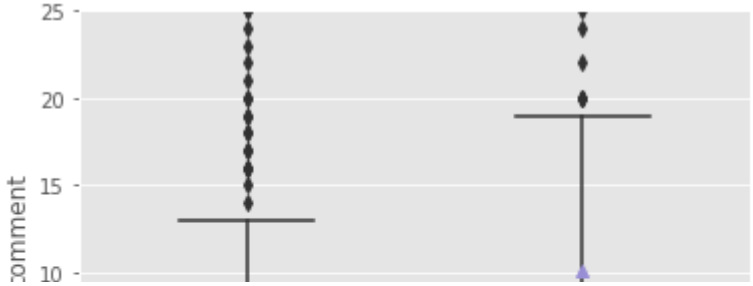

```
sns.boxplot(x=data['Paid'],y=data['like'],showmeans=True, palette='rocket')
plt.ylim(0,500)
```

```
(0.0, 500.0)
```



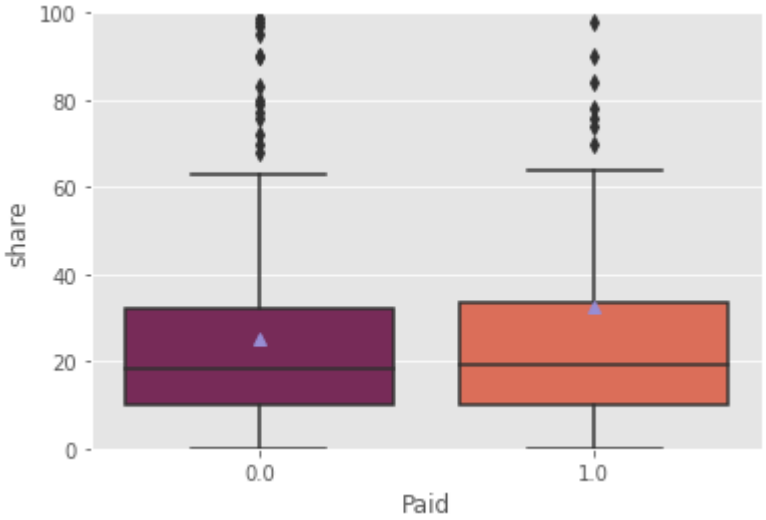

```
sns.boxplot(x=data['Paid'],y=data['comment'],showmeans=True,palette='rocket')
plt.ylim(0,25)
```

```
(0.0, 25.0)
```



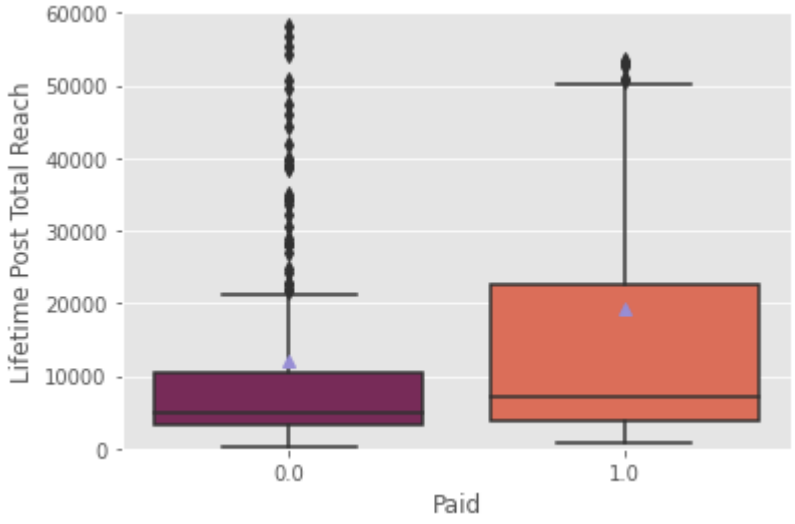

```
sns.boxplot(x=data['Paid'],y=data['share'],showmeans=True,palette='rocket')
plt.ylim(0,100)
```

```
(0.0, 100.0)
```



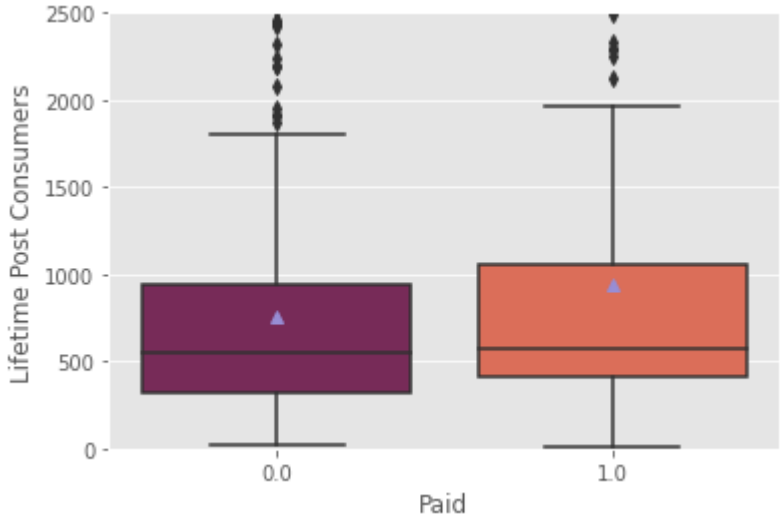

```
sns.boxplot(x=data['Paid'],y=data['Lifetime Post Total Reach'],showmeans=True,palette='rocket
plt.ylim(0,60000)
```

```
(0.0, 60000.0)
```



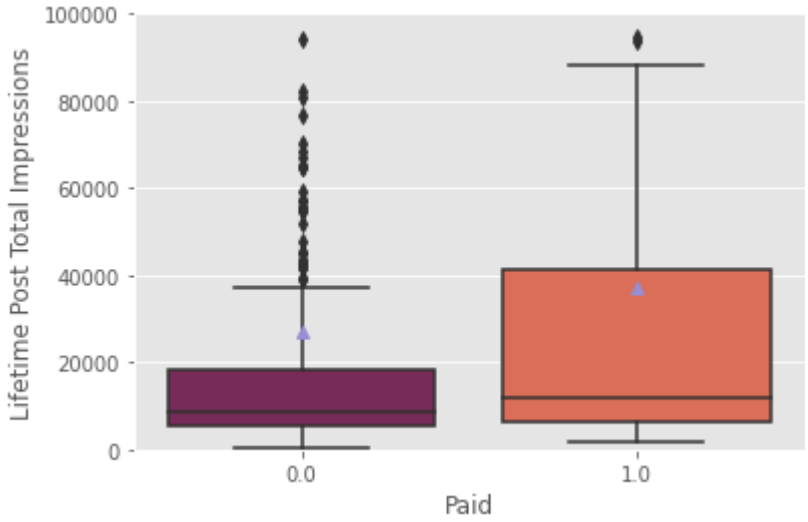

```
sns.boxplot(x=data['Paid'],y=data['Lifetime Post Consumers'],showmeans=True,palette='rocket'
plt.ylim(0,2500)
```

```
(0.0, 2500.0)
```

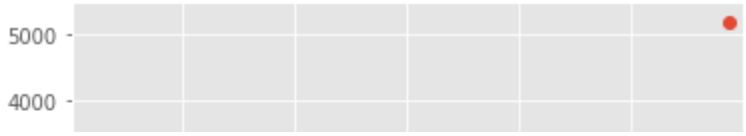



```
sns.boxplot(x=data['Paid'],y=data['Lifetime Post Total Impressions'],showmeans=True,palette='
plt.ylim(0,100000)
```

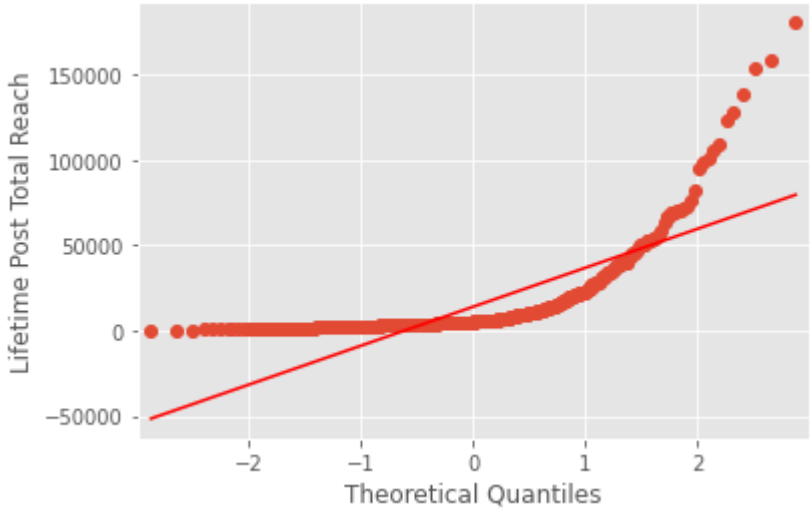
```
(0.0, 100000.0)
```



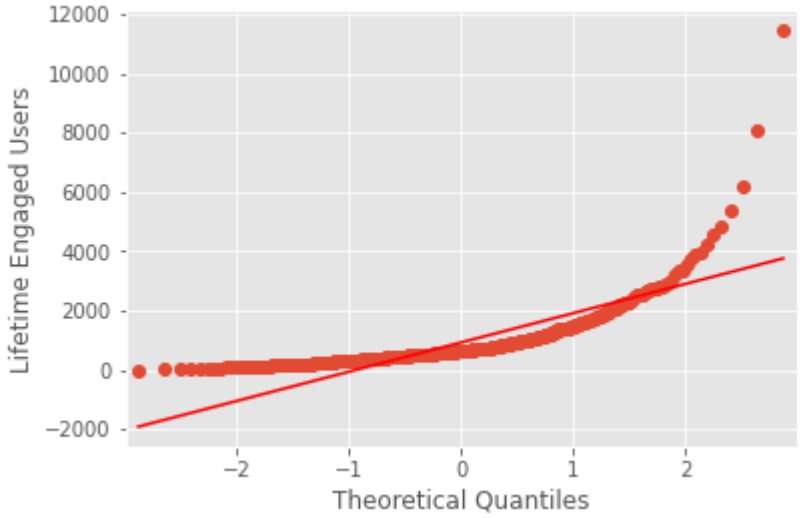

```
qqplot(data['like'], line='s')
plt.ylabel('like')
plt.show()
```



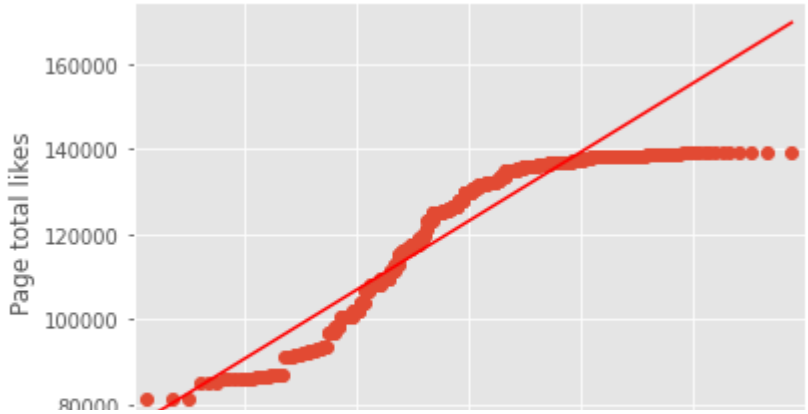

```
qqplot(data['Lifetime Post Total Reach'], line='s')
plt.ylabel('Lifetime Post Total Reach')
plt.show()
```



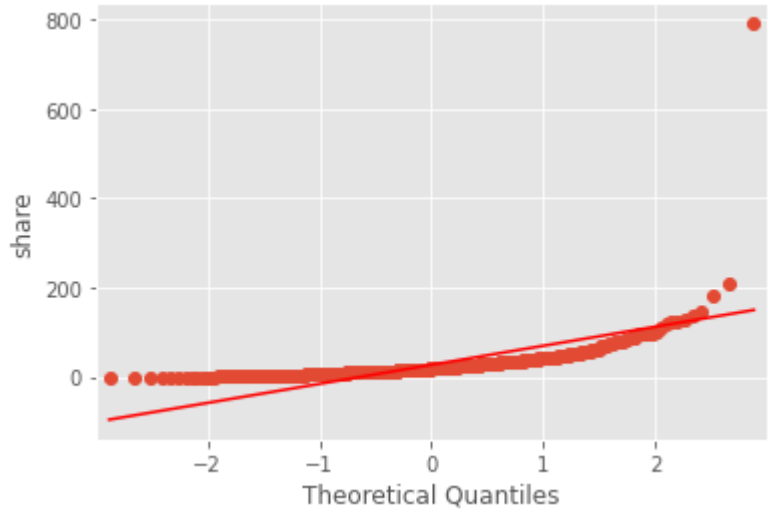

```
qqplot(data['Lifetime Engaged Users'], line='s')
plt.ylabel('Lifetime Engaged Users')
plt.show()
```



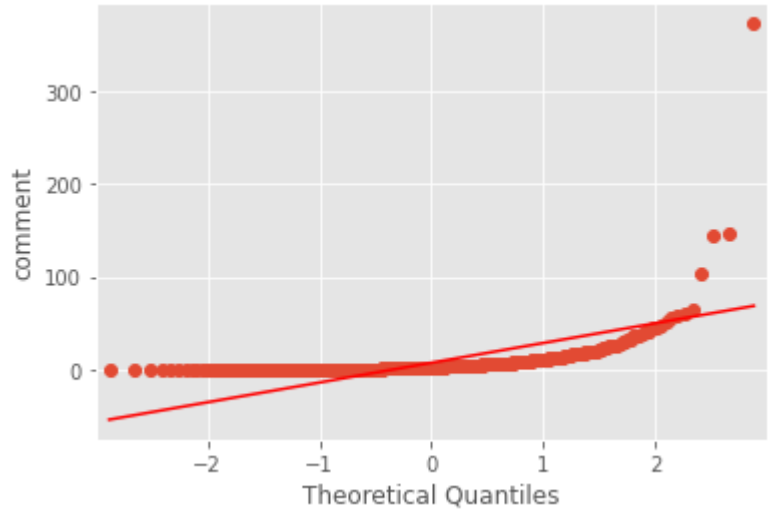

```
qqplot(data['Page total likes'], line='s')
plt.ylabel('Page total likes')
plt.show()
```




```
qqplot(data['share'], line='s')
plt.ylabel('share')
plt.show()
```

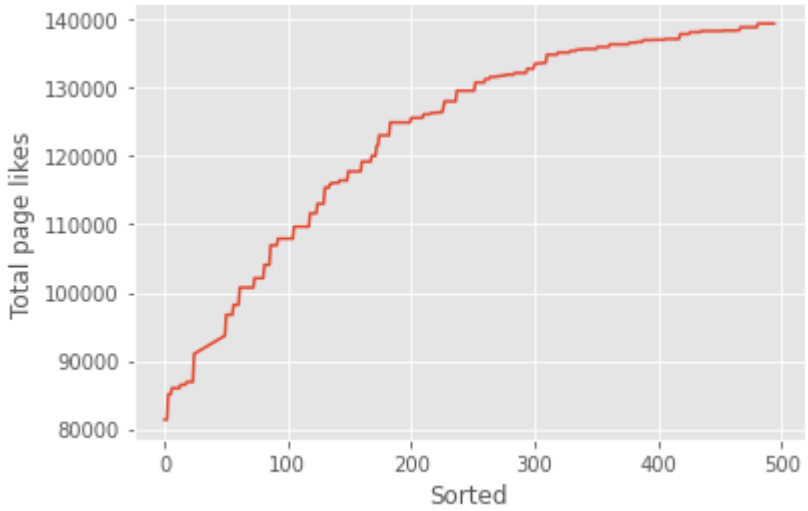



```
qqplot(data['comment'], line='s')
plt.ylabel('comment')
plt.show()
```



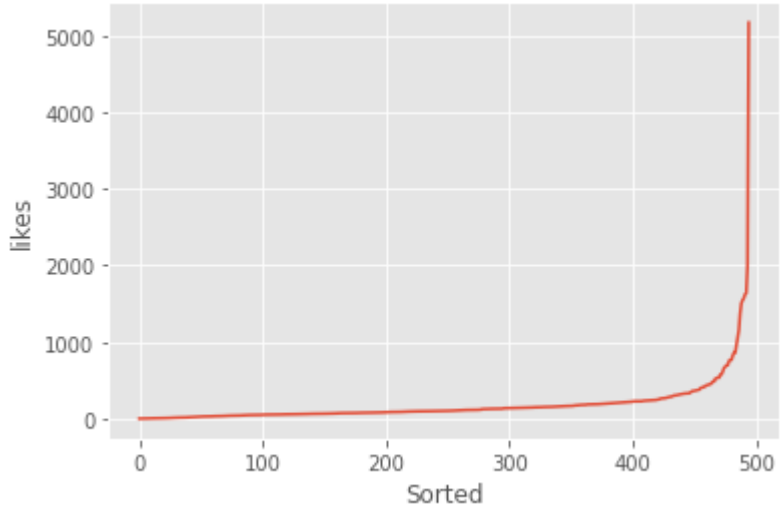
```
data1 = data.sort_values('Page total likes')
data1 = data1.reset_index(drop=True)
plt.ylabel('Total page likes')
plt.xlabel('Sorted')
plt.plot(data1['Page total likes'])
```

```
[<matplotlib.lines.Line2D at 0x7f92e057bd00>]
```




```
data1 = data.sort_values('like')
data1 = data1.reset_index(drop=True)
a = data1['like']
plt.ylabel('likes')
plt.xlabel('Sorted')
plt.plot(a)
```

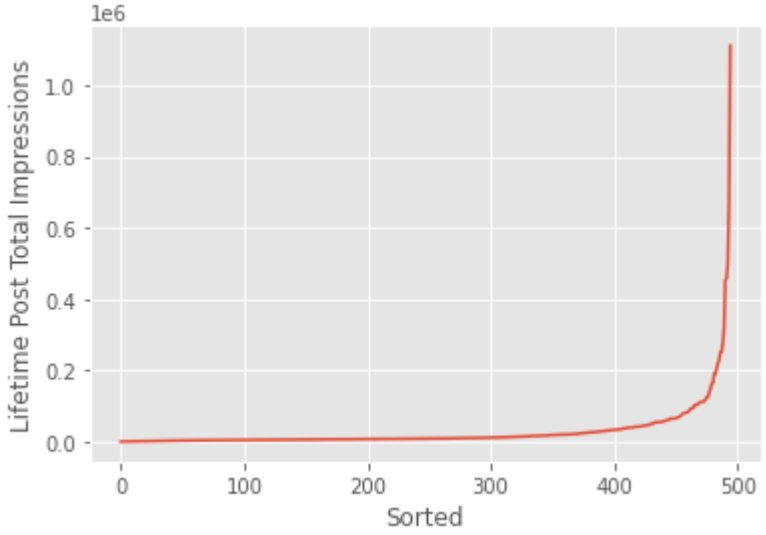
```
[<matplotlib.lines.Line2D at 0x7f92e03e2a60>]
```




```
data1 = data.sort_values('Lifetime Post Total Impressions')
data1 = data1.reset_index(drop=True)
a = data1['Lifetime Post Total Impressions']
plt.ylabel('Lifetime Post Total Impressions')
```

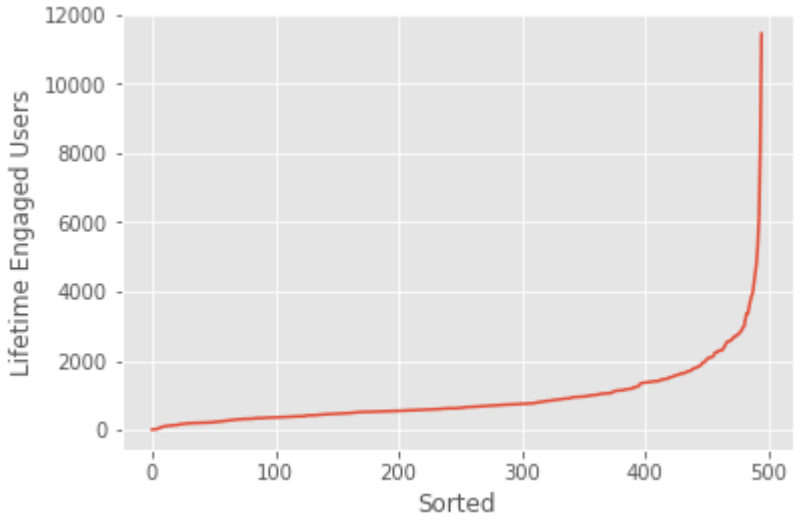
```
plt.xlabel('Sorted')
plt.plot(a)
```

```
[<matplotlib.lines.Line2D at 0x7f92e0319c10>]
```




```
data1 = data.sort_values('Lifetime Engaged Users')
data1 = data1.reset_index(drop=True)
a = data1['Lifetime Engaged Users']
plt.ylabel('Lifetime Engaged Users')
plt.xlabel('Sorted')
plt.plot(a)
```

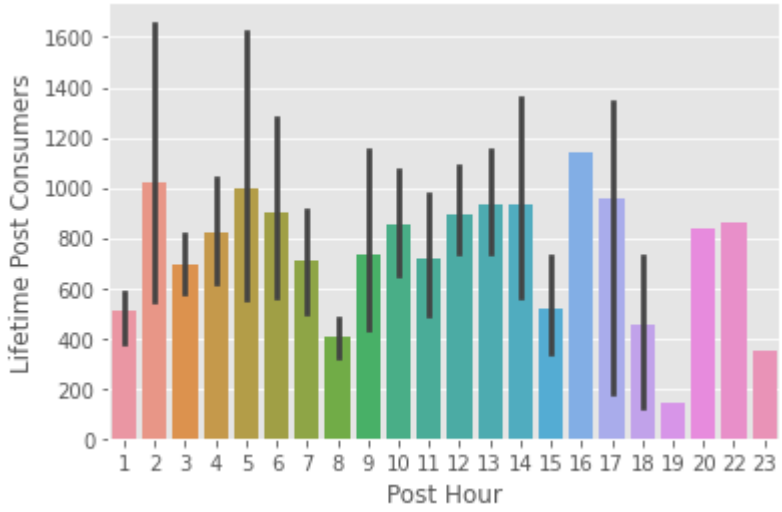
```
[<matplotlib.lines.Line2D at 0x7f92e0292d90>]
```




```
data.head()
```

	Page total likes	Type	Category	Post Month	Post Weekday	Post Hour	Paid	Lifetime Post Total Reach	Lifetime Post Total Impressions	Lifetime Engaged Users
0	139441	Photo	2	12	4	3	0.0	2752	5091	178
1	139441	Status	2	12	3	10	0.0	10460	19057	1457
2	139441	Photo	3	12	3	3	0.0	2413	4373	177
3	139441	Photo	2	12	2	10	1.0	50128	87991	2211
4	139441	Photo	2	12	2	3	0.0	7244	13594	671

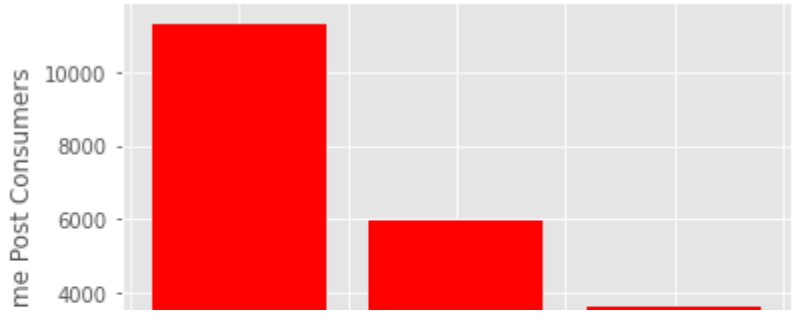
```
ax = sns.barplot(x="Post Hour", y="Lifetime Post Consumers", data=data)
```



Influence of hour on lifetime consumers

```
plt.bar("Category","Lifetime Post Consumers", color="Red",data=data)
plt.xlabel("Category")
plt.ylabel("Life Time Post Consumers")
```

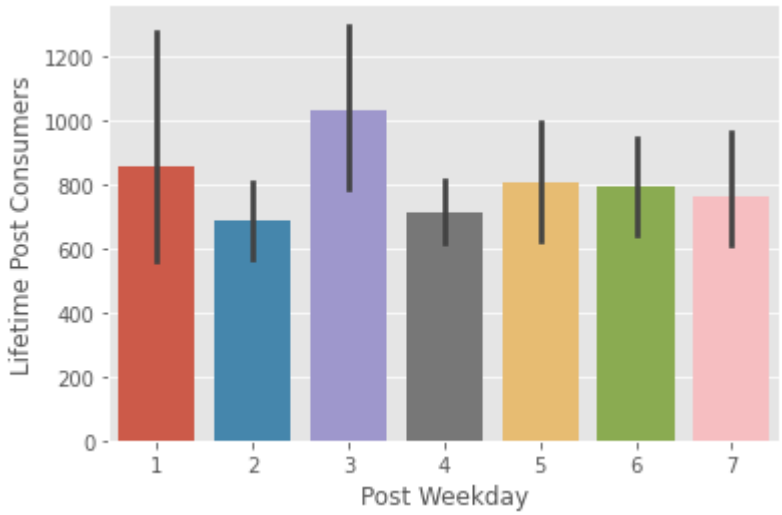
```
Text(0, 0.5, 'Life Time Post Consumers')
```



Influence of category on life time post consumers

```
0.5       1.0       1.5       2.0       2.5       3.0       3.5
```

```
ax = sns.barplot(x="Post Weekday", y="Lifetime Post Consumers", data=data)
```

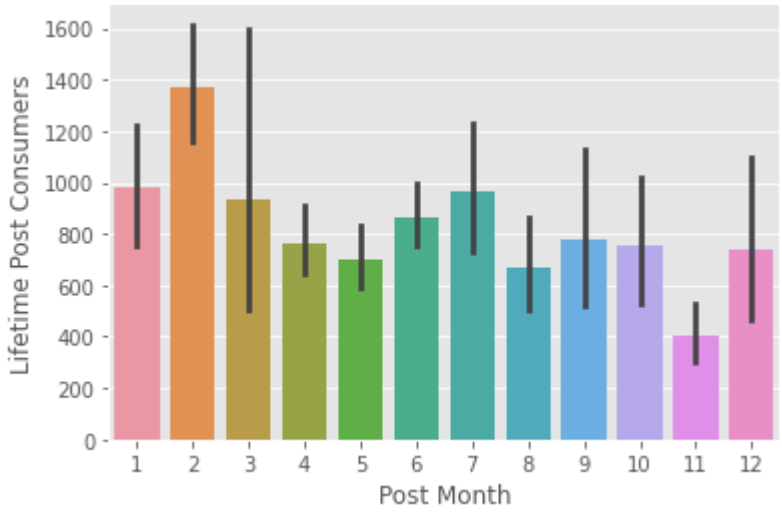


Influence of weekday on life time post consumers

```
ax = sns.barplot(x="Type", y="Lifetime Post Consumers", data=data)
```

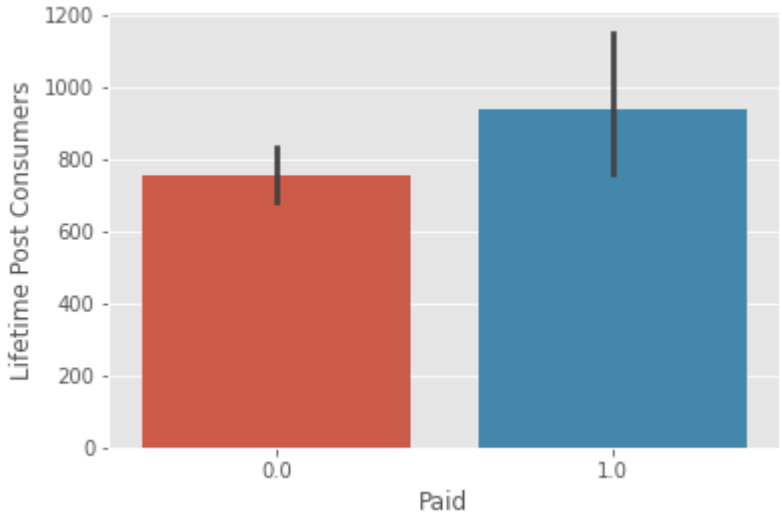
Influence of Type on life time post consumers

```
ax = sns.barplot(x="Post Month", y="Lifetime Post Consumers", data=data)
```



Influence of month on lifetime post consumers

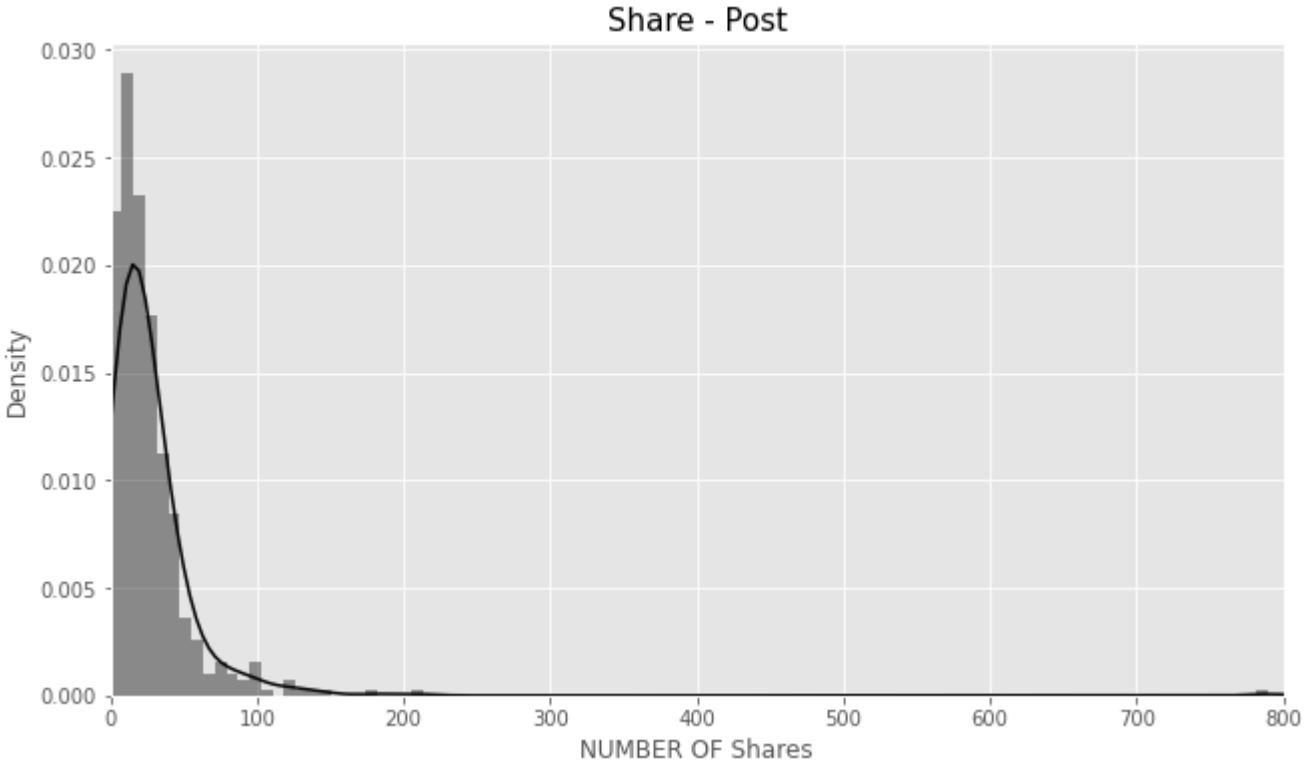
```
ax = sns.barplot(x="Paid", y="Lifetime Post Consumers", data=data)
```



Influence of paid on lifetime post time consumers

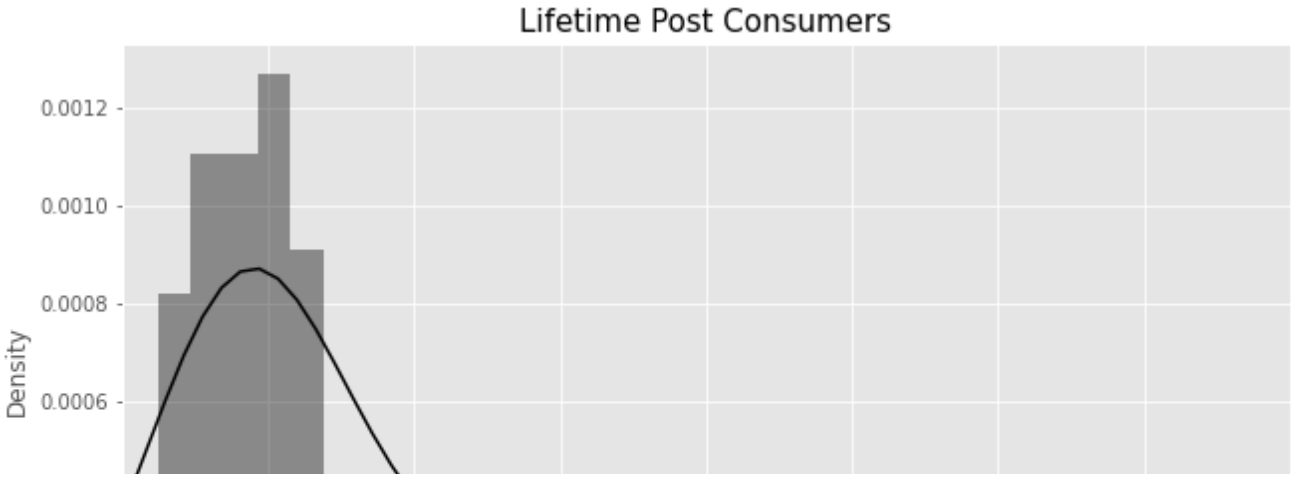
```
plt.figure(figsize=(10.5,6))
sns.distplot(data['share'],bins=100,color='black',kde=True)
plt.xlim(0,800)
plt.xlabel("NUMBER OF Shares",fontsize=12)
plt.ylabel('Density',fontsize=12)
plt.title('Share - Post',fontsize=15)
```

```
<ipython-input-123-744bdd963281>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['share'],bins=100,color='black',kde=True)
Text(0.5, 1.0, 'Share - Post')
```



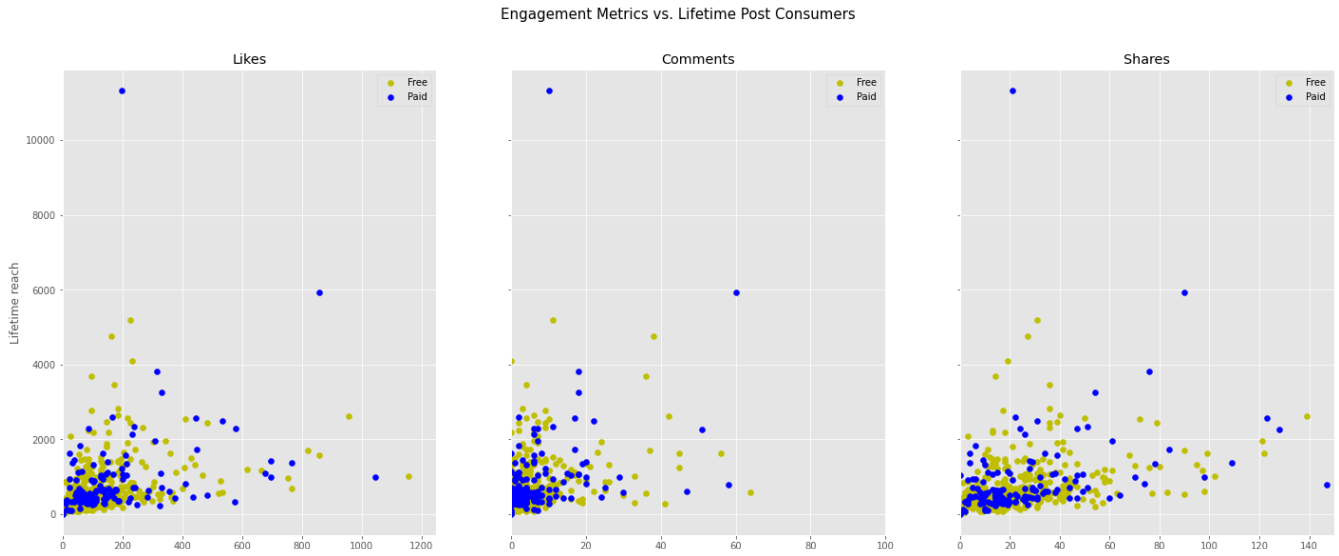

```
plt.figure(figsize=(10.5,6))
sns.distplot(data['Lifetime Post Consumers'],bins=100,color='black')
plt.xlim(0,4000)
plt.title('Lifetime Post Consumers',fontsize=15)
```

```
<ipython-input-124-11318ed92917>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['Lifetime Post Consumers'],bins=100,color='black')
Text(0.5, 1.0, 'Lifetime Post Consumers')
```



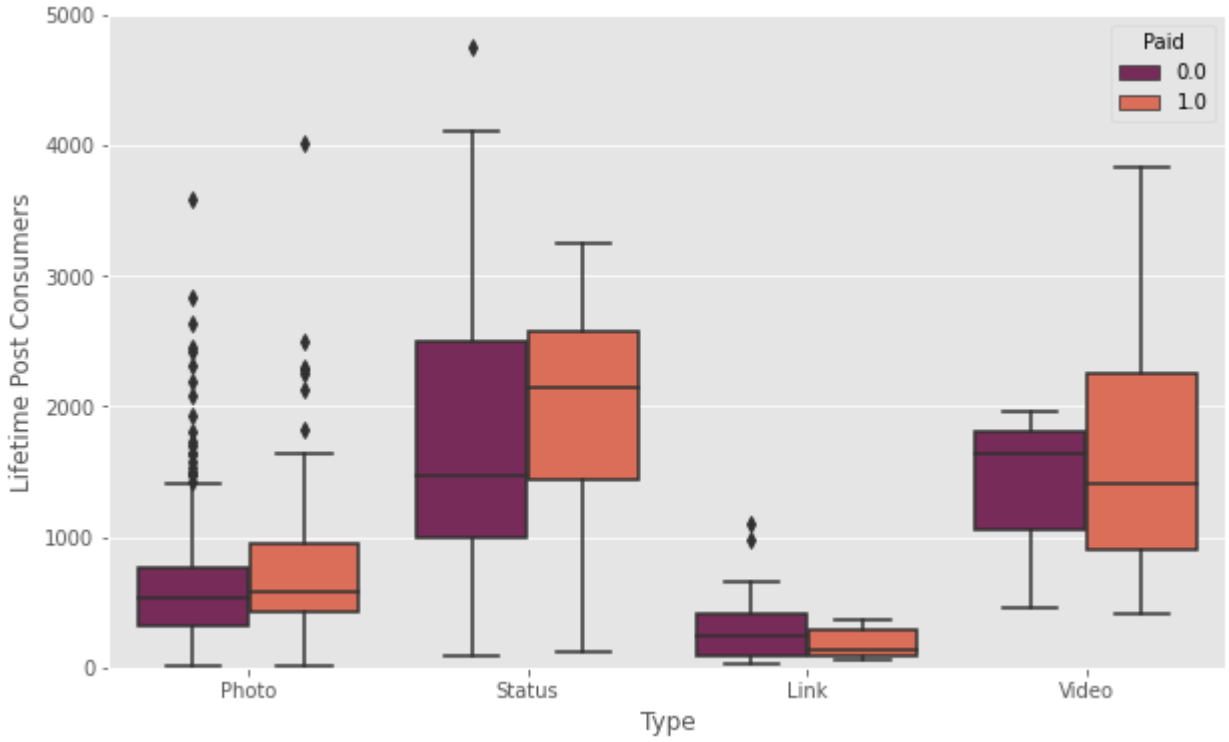

```
fig, ax = plt.subplots(ncols=3,nrows=1,sharey=True,figsize=(24,9))

paid = data[data['Paid']==1]
free = data[data['Paid']==0]

ax[0].scatter(free['like'],free['Lifetime Post Consumers'],color='y')
ax[0].scatter(paid['like'],paid['Lifetime Post Consumers'],color='b')
ax[0].set_title('Likes')
ax[0].set_xlim(0,1250)
ax[0].legend(labels=['Free','Paid'])

ax[1].scatter(free['comment'],free['Lifetime Post Consumers'],color='y')
ax[1].scatter(paid['comment'],paid['Lifetime Post Consumers'],color='b')
ax[1].set_title('Comments')
ax[1].set_xlim(0,100)
ax[1].legend(labels=['Free','Paid'])

ax[2].scatter(free['share'],free['Lifetime Post Consumers'],color='y')
ax[2].scatter(paid['share'],paid['Lifetime Post Consumers'],color='b')
ax[2].set_title('Shares')
ax[2].set_xlim(0,150)
ax[2].legend(labels=['Free','Paid'])

ax[0].set_ylabel("Lifetime reach")

fig.suptitle('Engagement Metrics vs. Lifetime Post Consumers',fontsize=15)
```

```
Text(0.5, 0.98, 'Engagement Metrics vs. Lifetime Post Consumers')
```




```
plt.figure(figsize=(10,6))
sns.boxplot(x=data['Type'],y=data['Lifetime Post Consumers'],hue=data['Paid'],palette='rocket
plt.ylim(0,5000)
```

```
(0.0, 5000.0)
```

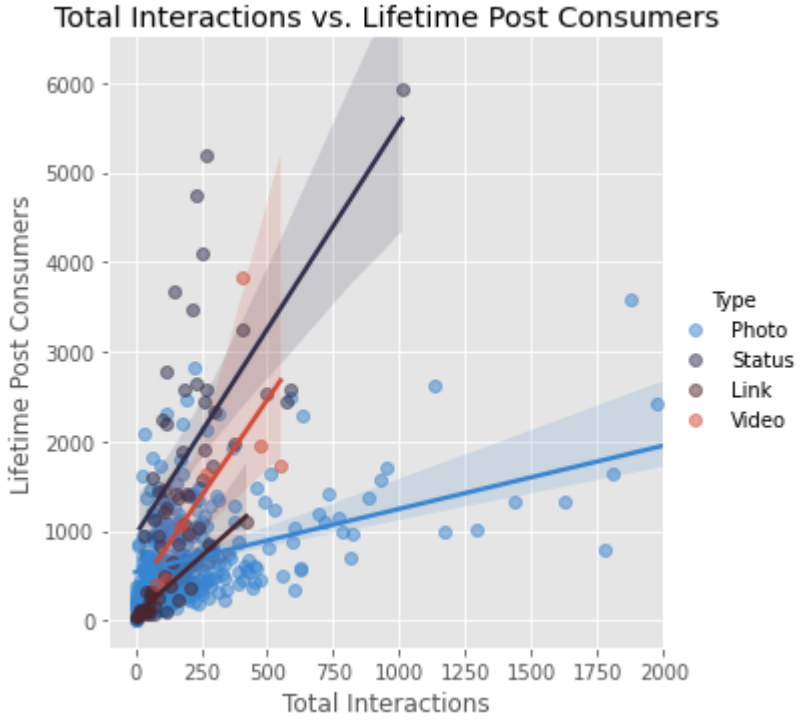


Engagement for Status > Video > photo > Link

```
sns.lmplot(x='Total Interactions',y='Lifetime Post Consumers',
           hue='Type',data=data,scatter_kws= {'alpha':0.5},palette='icefire')
plt.title('Total Interactions vs. Lifetime Post Consumers')
```

```
plt.xlim(-100,2000)
plt.ylim(-300,6500)
```

```
(-300.0, 6500.0)
```

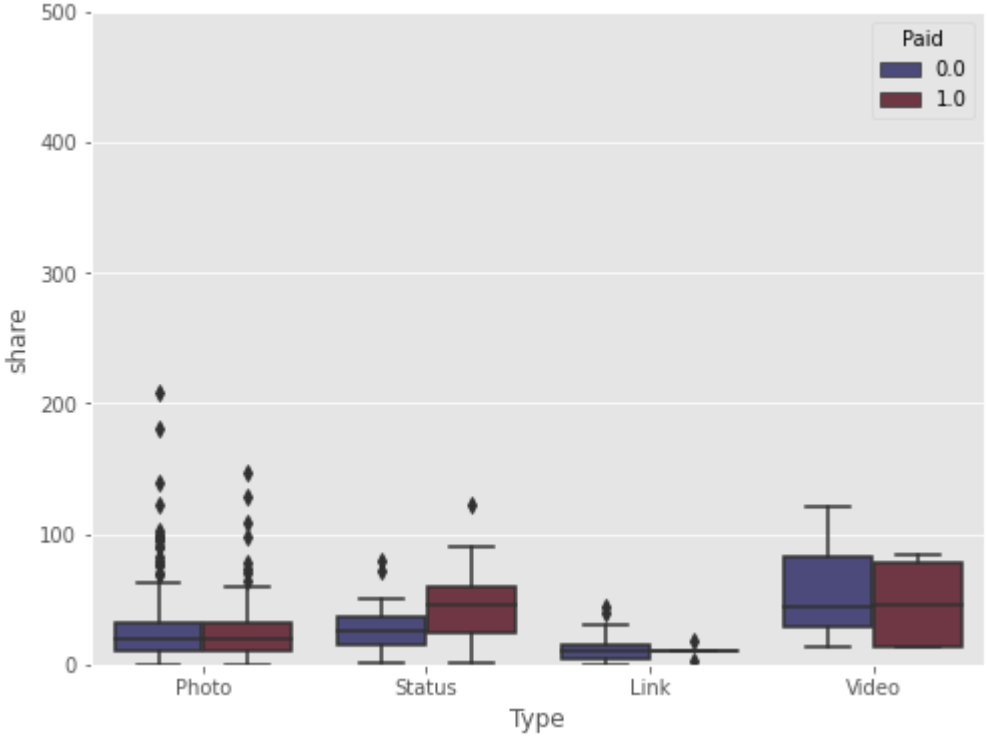



```
plt.style.use('ggplot')
sns.lmplot(x='Total Interactions',y='Lifetime Post Consumers',
           hue='Paid',data=data,fit_reg=False,scatter_kws= {'alpha':0.7})
plt.title('Total Interactions vs. Lifetime Post Consumers')
plt.xlim(-100,2500)
plt.ylim(-300,6000)
```

```
(-300.0, 6000.0)
```

```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Type'],y=data['share'],hue=data['Paid'],palette='icefire')
plt.ylim(0,500)
```

```
(0.0, 500.0)
```

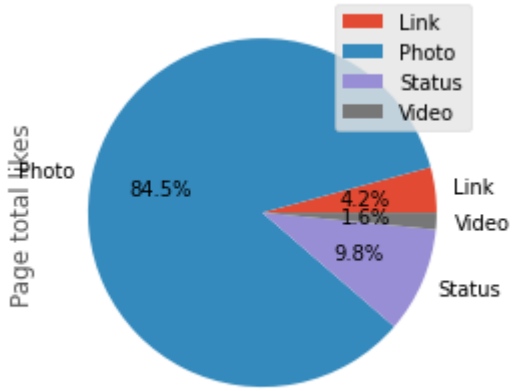



```
plt.figure(figsize=(8,6))
sns.boxplot(x=data['Post Weekday'],y=data['share'],hue=data['Paid'],palette='rocket')
plt.ylim(-25,650)
plt.title("Shares in Weekday")
```

```
Text(0.5, 1.0, 'Shares in Weekday')
```


```
data.groupby(['Type']).sum().plot(kind='pie', y='Page total likes', autopct='%1.1f%%')
```

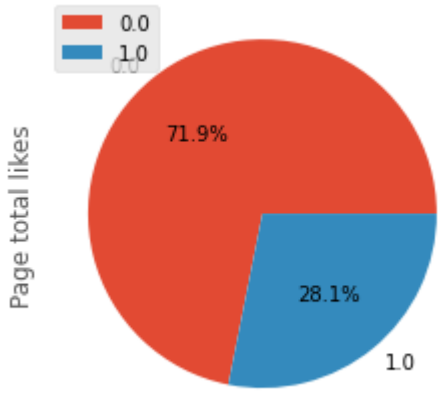
```
<Axes: ylabel='Page total likes'>
```



Post Weekday

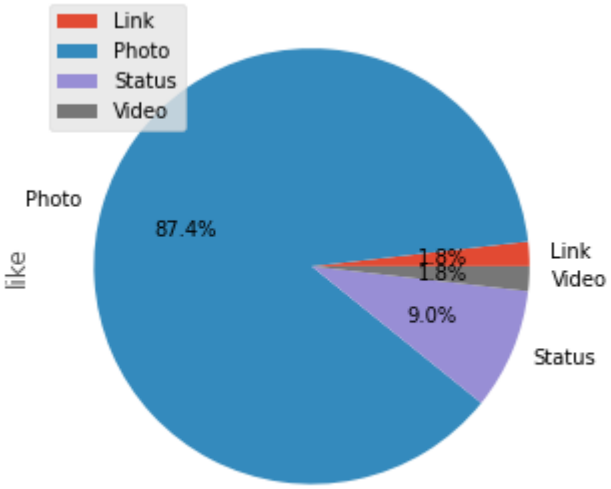
```
data.groupby(['Paid']).sum().plot(kind='pie', y='Page total likes', autopct='%1.1f%%')
```

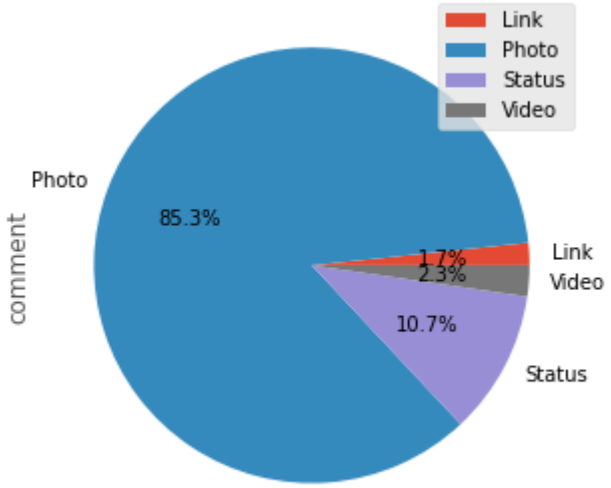
```
<Axes: ylabel='Page total likes'>
```

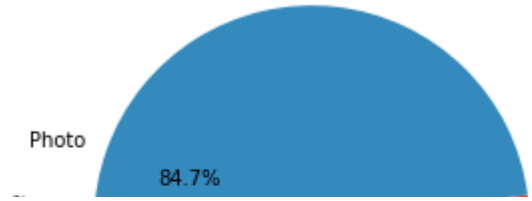



```
data.groupby(['Type']).sum().plot(kind='pie', y='like', autopct='%1.1f%%', figsize=(10, 5))
data.groupby(['Type']).sum().plot(kind='pie', y='comment', autopct='%1.1f%%', figsize=(10, 5)
data.groupby(['Type']).sum().plot(kind='pie', y='share', autopct='%1.1f%%', figsize=(10, 5))
```

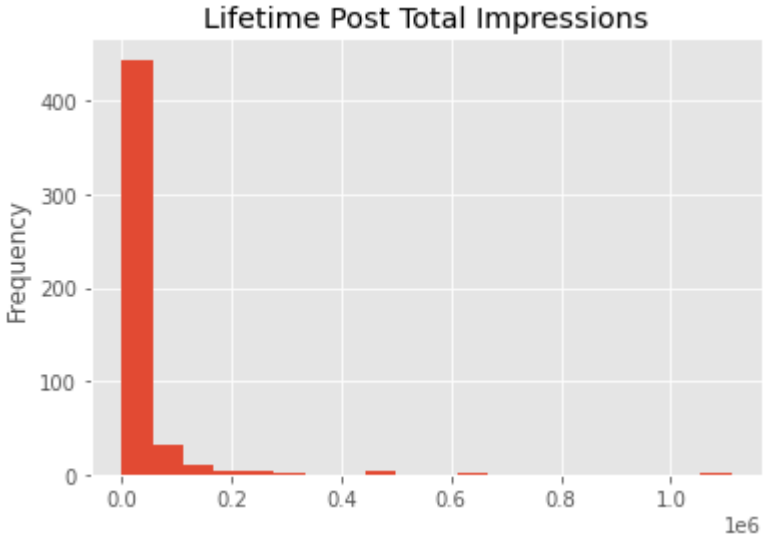
```
<Axes: ylabel='share'>
```

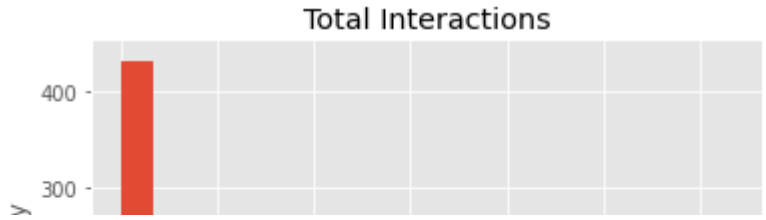







```
for name in ['Lifetime Post Total Impressions','Total Interactions']:
    data_pca[name].plot(kind='hist',title=name,bins=20)
    plt.show()
```






```
data_count = data.groupby("Post Weekday")["Lifetime Post Total Impressions"].sum().sort_value
data_count.plot(kind="barh",title='Total Impressions based on Weekdays')
```

```
<Axes: title={'center': 'Total Impressions based on Weekdays'}, ylabel='Post Weekday'>
```