Explorative Data Analysis (EDA) Project

September 26, 2025

[1]:
```
# Impport packages for data manipulation and data visualization
import pandas as pd
import numpy as np
import datetime as dt
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
# load the data into data frame
df=pd.read_csv('tiktok_dataset.csv')
```

[3]:
```
# Examine the first few raws of the dataframe
df.head()
```

[3]:

	# claim_status	video_id	video_duration_sec	video_transcription_text	verified_status	author_ban_status	video_view_count	video_like_count	video_share_count	video_download_count	video_comment_count
0	1	claim	7017666017	59	someone shared with me that drone deliveries a…	not verified	under review	343296.0	19425.0	241.0	1.0	0.0
1	2	claim	4014381136	32	someone shared with me that there are more mic…	not verified	active	140877.0	77355.0	19034.0	1161.0	684.0
2	3	claim	9859838091	31	someone shared with me that american industria…	not verified	active	902185.0	97690.0	2858.0		
3	4	claim	1866847991	25	someone shared with me that the metro of st. p…	not verified	active	437506.0	239954.0	34812.0		
4	5	claim	7105231098	19	someone shared with me that the number of busi…	not verified	active	56167.0	34987.0	4110.0		

```
2                833.0               329.0
3               1234.0               584.0
4                547.0               152.0
```

[4]:
```
# Examine the first few raws of the dataframe
df.shape
```

[4]:
```
(19382, 12)
```

[5]:
```
# Get basic information about the data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19382 entries, 0 to 19381
Data columns (total 12 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   #                         19382 non-null  int64
 1   claim_status              19084 non-null  object
 2   video_id                  19382 non-null  int64
 3   video_duration_sec        19382 non-null  int64
 4   video_transcription_text  19084 non-null  object
 5   verified_status           19382 non-null  object
 6   author_ban_status         19382 non-null  object
 7   video_view_count          19084 non-null  float64
 8   video_like_count          19084 non-null  float64
 9   video_share_count         19084 non-null  float64
 10  video_download_count      19084 non-null  float64
 11  video_comment_count       19084 non-null  float64
dtypes: float64(5), int64(3), object(4)
memory usage: 1.8+ MB
```

[6]:
```
# Get descriptive statistics of the data
df.describe()
```

[6]:
```
                 #      video_id  video_duration_sec  video_view_count  \
count  19382.000000  1.938200e+04        19382.000000      19084.000000
mean    9691.500000  5.627454e+09           32.421732     254708.558688
std     5595.245794  2.536440e+09           16.229967     322893.280814
min        1.000000  1.234959e+09            5.000000         20.000000
25%     4846.250000  3.430417e+09           18.000000       4942.500000
50%     9691.500000  5.618664e+09           32.000000       9954.500000
75%    14536.750000  7.843960e+09           47.000000     504327.000000
max    19382.000000  9.999873e+09           60.000000     999817.000000

       video_like_count  video_share_count  video_download_count  \
count      19084.000000       19084.000000          19084.000000
mean       84304.636030       16735.248323           1049.429627
```

```
std       133420.546814       32036.174350           2004.299894
min            0.000000           0.000000              0.000000
25%          810.750000         115.000000              7.000000
50%         3403.500000         717.000000             46.000000
75%       125020.000000       18222.000000           1156.250000
max       657830.000000      256130.000000          14994.000000

       video_comment_count
count         19084.000000
mean            349.312146
std             799.638865
min               0.000000
25%               1.000000
50%               9.000000
75%             292.000000
max            9599.000000
```

0.0.1 Data cleaning

- Handle issing value
- Handle duplicates

[7]:
```
# Get missing value in the dataframe
df.isnull().sum()
```

[7]:
```
#                           0
claim_status              298
video_id                    0
video_duration_sec          0
video_transcription_text  298
verified_status             0
author_ban_status           0
video_view_count          298
video_like_count          298
video_share_count         298
video_download_count      298
video_comment_count       298
dtype: int64
```

[8]:
```
df= df.dropna(axis= 0)
```

[9]:
```
df.isnull().sum()
```

[9]:
```
#                         0
claim_status              0
video_id                  0
video_duration_sec        0
video_transcription_text  0
```

```
verified_status             0
author_ban_status           0
video_view_count            0
video_like_count            0
video_share_count           0
video_download_count        0
video_comment_count         0
dtype: int64
```

[10]:
```
df= df.drop_duplicates()
```

[11]:
```
df.columns
```

[11]:
```
Index(['#', 'claim_status', 'video_id', 'video_duration_sec',
       'video_transcription_text', 'verified_status', 'author_ban_status',
       'video_view_count', 'video_like_count', 'video_share_count',
       'video_download_count', 'video_comment_count'],
      dtype='object')
```

[12]:
```
df['claim_status'].unique()
```

[12]:
```
array(['claim', 'opinion'], dtype=object)
```

[13]:
```
df['claim_status'].value_counts()
```

[13]:
```
claim_status
claim      9608
opinion    9476
Name: count, dtype: int64
```

[16]:
```
df['author_ban_status'].unique()
```

[16]:
```
array(['under review', 'active', 'banned'], dtype=object)
```

[17]:
```
df['author_ban_status'].value_counts()
```

[17]:
```
author_ban_status
active          15383
under review     2066
banned           1635
Name: count, dtype: int64
```

[20]:
```
#  # compile the information to determin count,mean and median of␣
 ↪vidio_view_count, video_like_count, video_share_count for each claim_status
df.groupby(['claim_status']).agg(
{'video_view_count': ['mean', 'median'],
'video_like_count': ['mean', 'median'],
'video_share_count': ['mean', 'median'],
```

```
'video_download_count':['mean', 'median'],
'video_comment_count':['mean', 'median']})
```

[20]:

	video_view_count		video_like_count		video_share_count		video_download_count		video_comment_count	
	mean	median	mean	median	mean	median	mean	median	mean	median
claim_status										
claim	501029.452748	501555.0	166373.331182	123649.0	33026.416216	17997.5	2070.952227	1139.5	691.164863	286.0
opinion	4956.432250	4953.0	1092.729844	823.0	217.145631	121.0	13.677290	7.0	2.697446	1.0

[21]:
```
#  # compile the information to determin count,mean and median of␣
 ↪vidio_view_count, video_like_count, video_share_count for each author ban␣
 ↪status
df.groupby(['author_ban_status']).agg(
{'video_view_count': ['mean', 'median'],
'video_like_count': ['mean', 'median'],
'video_share_count': ['mean', 'median'],
'video_download_count':['mean', 'median'],
'video_comment_count':['mean', 'median']})
```

[21]:

	video_view_count		video_like_count		video_share_count		video_download_count		video_comment_count	
	mean	median	mean	median	mean	median	mean	median	mean	median
author_ban_status										
active	215927.039524	8616.0	71036.533836	2222.0	14111.466164	437.0	882.276344	28.0		
banned	445845.439144	448201.0	153017.236697	105573.0	29998.942508	14468.0	1886.296024	892.0		
under review	392204.836399	365245.5	128718.050339	71204.5	25774.696999	9444.0	1631.734753	610.5		

```
author_ban_status
active                     295.134499    5.0
banned                     614.956575  209.0
under review               542.480639  136.5
```

[22]:
```
# Create a likes_per_view column
df['likes_per_view'] = df['video_like_count'] / df['video_view_count']
# Create a comments_per_view column
df['comments_per_view'] = df['video_comment_count'] / df['video_view_count']
# Create a shares_per_view column
df['shares_per_view'] = df['video_share_count'] / df['video_view_count']
# Create a download_per_view
df['download_per_view'] = df['video_download_count']/df['video_view_count']
df.head()
```

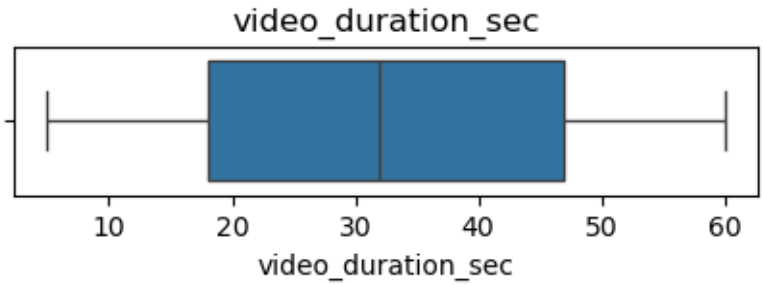
[22]:
```
   # claim_status     video_id  video_duration_sec  \
0  1        claim  7017666017                  59
1  2        claim  4014381136                  32
2  3        claim  9859838091                  31
3  4        claim  1866847991                  25
4  5        claim  7105231098                  19

                          video_transcription_text verified_status  \
0  someone shared with me that drone deliveries a…    not verified
1  someone shared with me that there are more mic…    not verified
2  someone shared with me that american industria…    not verified
3  someone shared with me that the metro of st. p…    not verified
4  someone shared with me that the number of busi…    not verified

  author_ban_status  video_view_count  video_like_count  video_share_count  \
0      under review          343296.0           19425.0              241.0
1            active          140877.0           77355.0            19034.0
2            active          902185.0           97690.0             2858.0
3            active          437506.0          239954.0            34812.0
4            active           56167.0           34987.0             4110.0

   video_download_count  video_comment_count  likes_per_view  \
0                   1.0                  0.0        0.056584
1                1161.0                684.0        0.549096
2                 833.0                329.0        0.108282
3                1234.0                584.0        0.548459
4                 547.0                152.0        0.622910

   comments_per_view  shares_per_view  download_per_view
0           0.000000         0.000702           0.000003
1           0.004855         0.135111           0.008241
2           0.000365         0.003168           0.000923
```

3	0.001335	0.079569	0.002821
4	0.002706	0.073175	0.009739

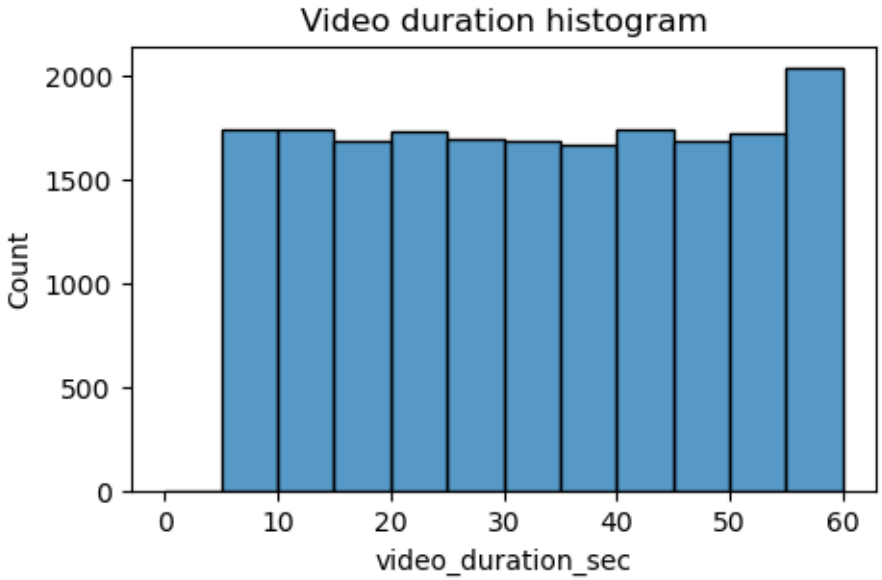
0.0.2 Data visvalization

[31]:
```
# Create a boxplot to visualize distribution of `video_duration_sec`

plt.figure(figsize=(5,1))
sns.boxplot(x= df['video_duration_sec'])
plt.title('video_duration_sec')
```

[31]:
```
Text(0.5, 1.0, 'video_duration_sec')
```

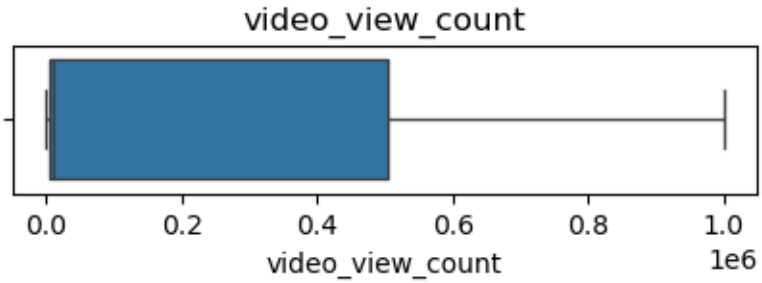


[29]:
```
 # Create a histogram to visualize distribution of `video_duration_sec`
plt.figure(figsize=(5,3))
sns.histplot(df['video_duration_sec'], bins=range(0,61,5))
plt.title('Video duration histogram');
```

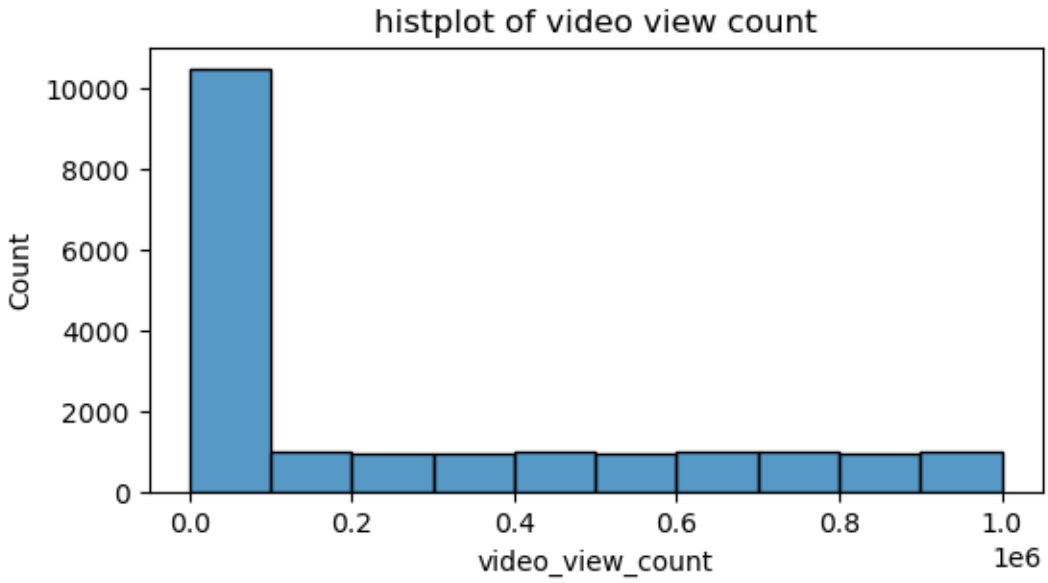


[32]:
```
#  # Create a boxplot to visualize distribution of `video_view_count`
plt.figure(figsize=(5,1))
sns.boxplot(x=df['video_view_count'])
plt.title('video_view_count')
```

[32]:
```
Text(0.5, 1.0, 'video_view_count')
```

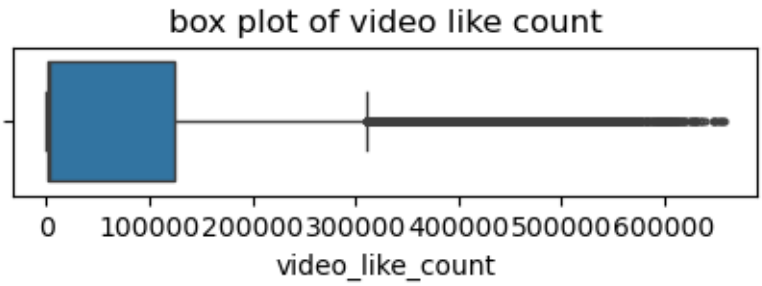


[40]:
```
#  Create a histogram to visualize distribution of `video_view_count`
plt.figure(figsize =(6,3))
sns.histplot(df['video_view_count'], bins=range(0,(10**6+1),10**5))
plt.title('histplot of video view count');
```



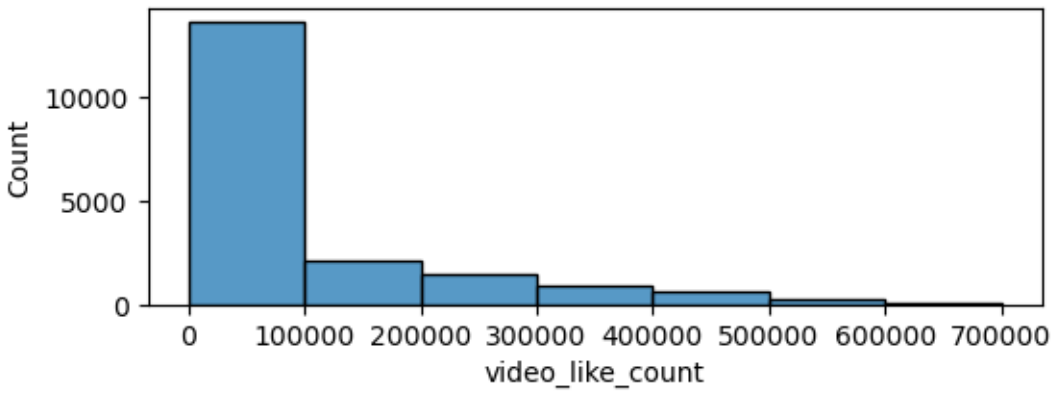
[43]:
```
# Create a boxplot to visualize distribution of video_like_count
plt.figure(figsize= (5,1))
sns.boxplot(x=df['video_like_count'], fliersize = 2)
plt.title('box plot of video like count')
```

[43]: Text(0.5, 1.0, 'box plot of video like count')

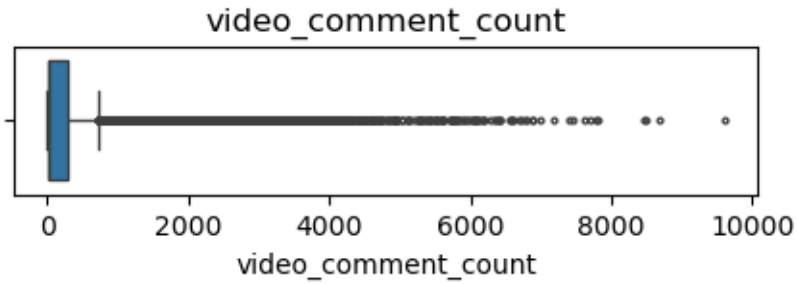


[46]:
```
# Create a histogram to visualize distribution of 'video_like_count'
plt.figure(figsize = (6,2))
sns.histplot(df['video_like_count'], bins=range(0,(7*10**5+1),10**5))
tick_labels = [0] + [f"{i}k" for i in range(100, 701, 100)]
ax.set_xticks(range(0, 7 * 10**5 + 1, 10**5))
ax.set_xticklabels(tick_labels)
plt.title('Video like count histogram');
```

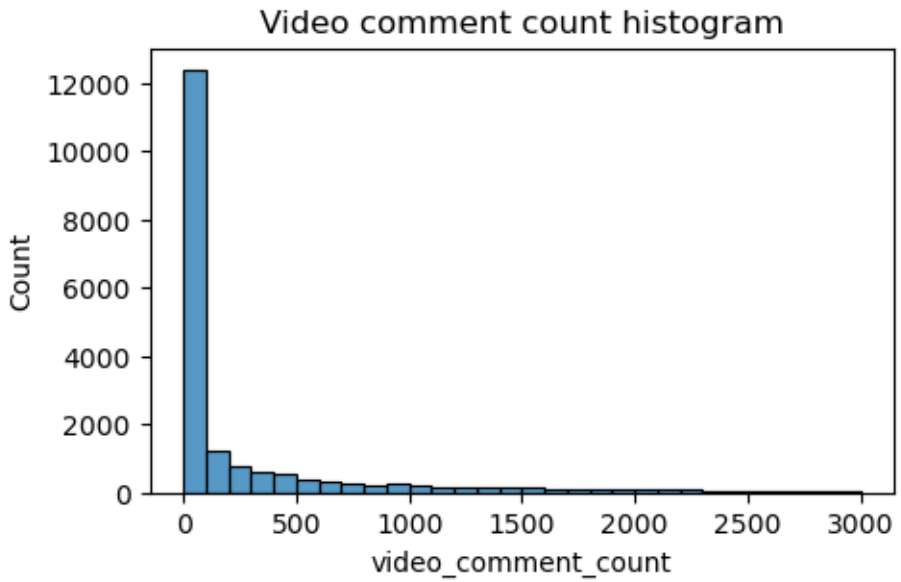
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[46], line 5
      3 sns.histplot(df['video_like_count'], bins=range(0,(7*10**5+1),10**5))
      4 tick_labels = [0] + [f"{i}k" for i in range(100, 701, 100)]
----> 5 ax.set_xticks(range(0, 7 * 10**5 + 1, 10**5))
      6 ax.set_xticklabels(tick_labels)
      7 plt.title('Video like count histogram');

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



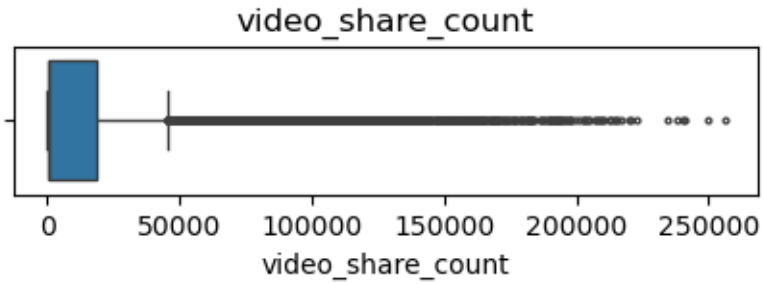
[47]:
```
 # Create a boxplot to visualize distribution of `video_comment_count`
plt.figure(figsize=(5,1))
plt.title('video_comment_count')
sns.boxplot(x=df['video_comment_count'],fliersize = 2);
```



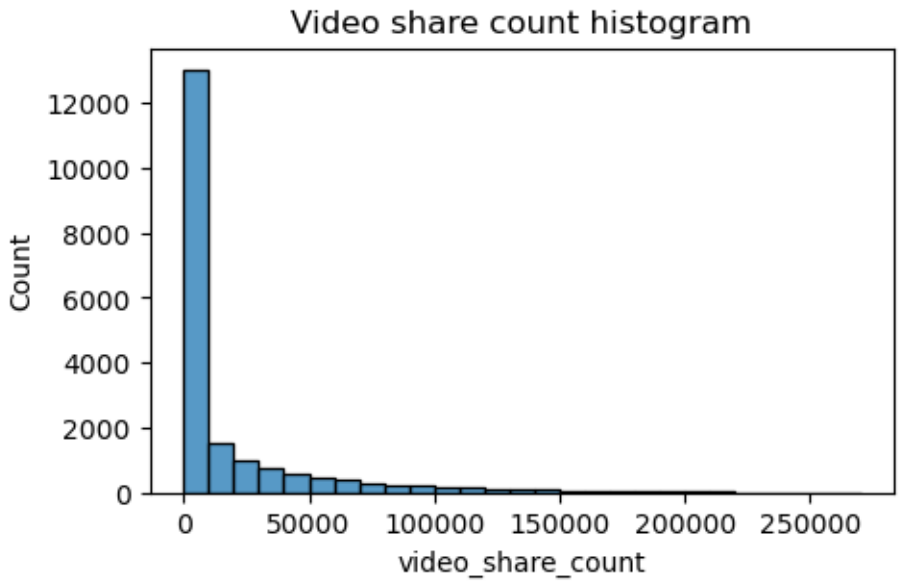
[48]:
```
 # Create a histogram to visualize distribution of `video_comment_count`
plt.figure(figsize=(5,3))
sns.histplot(df['video_comment_count'], bins=range(0,(3001),100))
plt.title('Video comment count histogram');
```



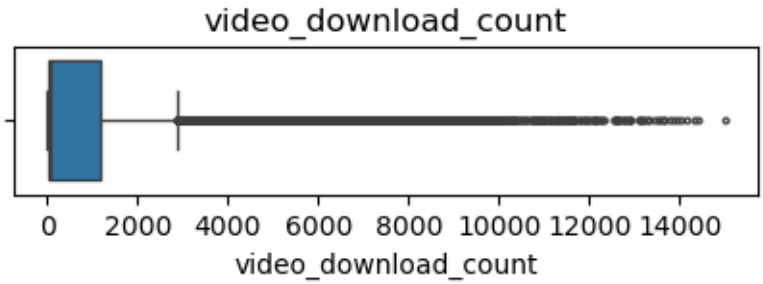
[49]:
```
 # Create a boxplot to visualize distribution of `video_share_count`
plt.figure(figsize=(5,1))
plt.title('video_share_count')
sns.boxplot(x=df['video_share_count'], fliersize= 2);
```



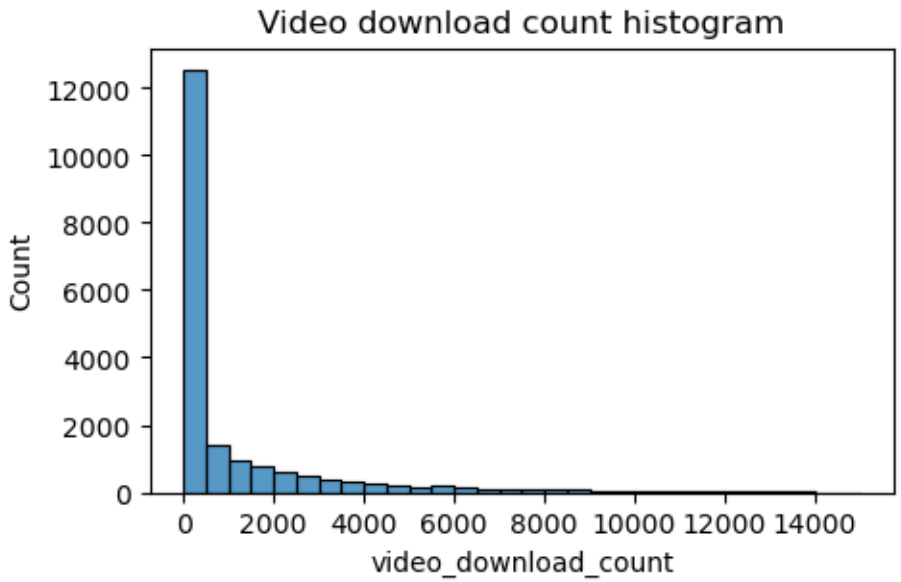
[50]:
```
# Create a histogram to visualize distribution of `video_share_count`
plt.figure(figsize=(5,3))
sns.histplot(df['video_share_count'], bins=range(0,(270001),10000))
plt.title('Video share count histogram');
```



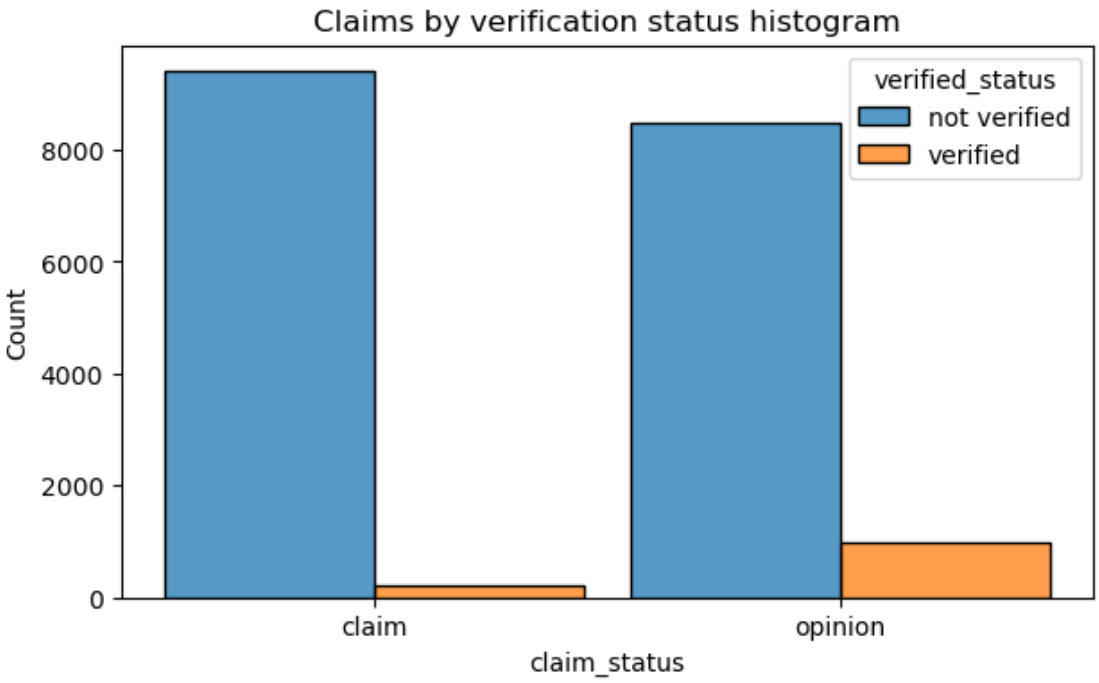
[51]:
```
# Create a boxplot to visualize distribution of `video_download_count`
plt.figure(figsize=(5,1))
plt.title('video_download_count')
sns.boxplot(x=df['video_download_count'], fliersize=2);
```



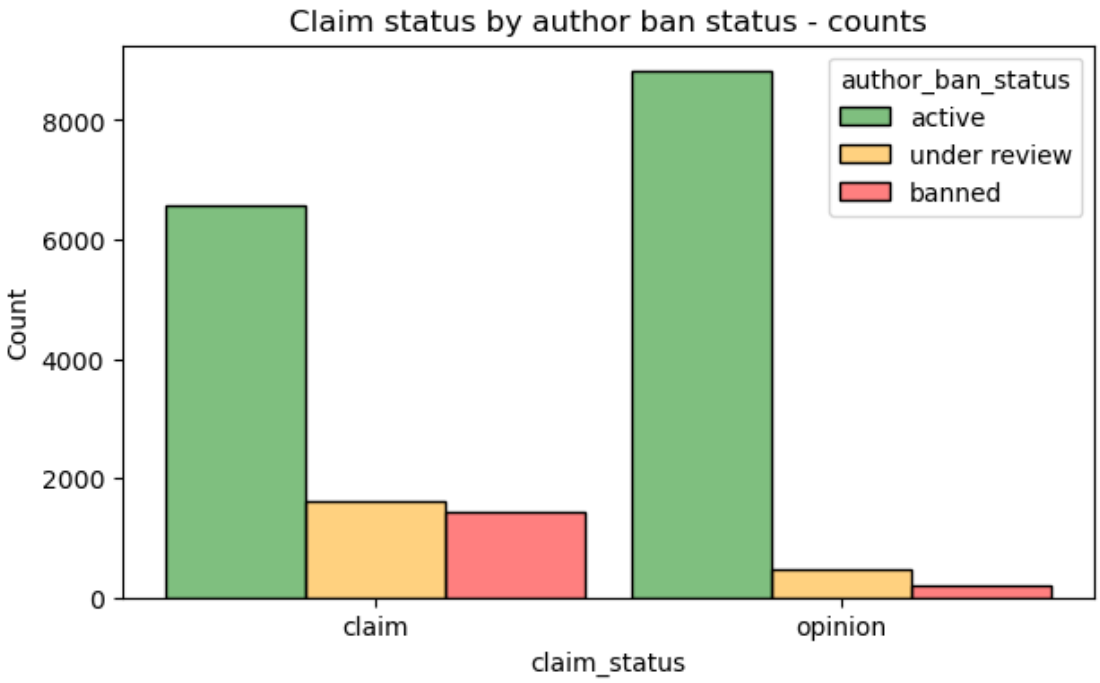
[52]:
```
 # Create a histogram to visualize distribution of `video_download_count`
plt.figure(figsize=(5,3))
sns.histplot(df['video_download_count'], bins=range(0,(15001),500))
plt.title('Video download count histogram');
```



[53]:
```
# Create a histogram with four bars: one for each combination of claim status␣
  ↪and verification status.
plt.figure(figsize=(7,4))
sns.histplot(data=df,
x='claim_status',
hue='verified_status',
multiple='dodge',
shrink=0.9)
plt.title('Claims by verification status histogram');
```



[54]:
```
# using histogram examine the count of each claim status for each author ban␣
↪status
fig = plt.figure(figsize=(7,4))
sns.histplot(df, x='claim_status', hue='author_ban_status',
multiple='dodge',
hue_order=['active', 'under review', 'banned'],
shrink=0.9,
palette={'active':'green', 'under review':'orange', 'banned':'red'},
alpha=0.5)
plt.title('Claim status by author ban status - counts');
```

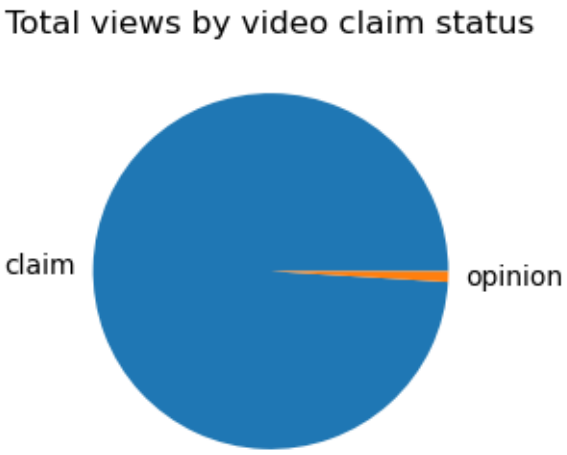


Create a bar plot with three bars: one for each author ban status. The height of each bar should correspond with the median number of views for all videos with that author ban status

[55]:
```
ban_status_counts = df.groupby(['author_ban_status']).
↪median(numeric_only=True).reset_index()
```

Create a pie graph that depicts the proportions of total views for claim videos and total views for opinion videos.

[59]:
```
fig = plt.figure(figsize=(3,3))
plt.pie(df.groupby('claim_status')['video_view_count'].
↪sum(),labels=['claim','opinion'])
plt.title('Total views by video claim status');
```

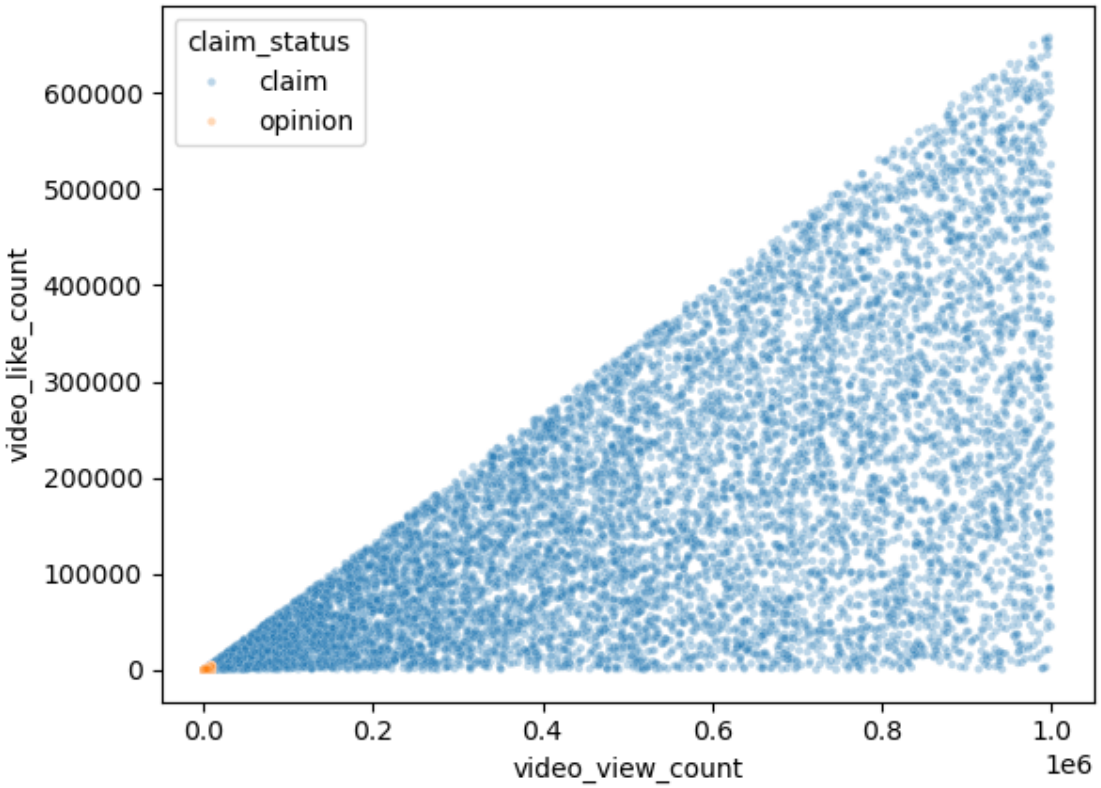


Determine outliers

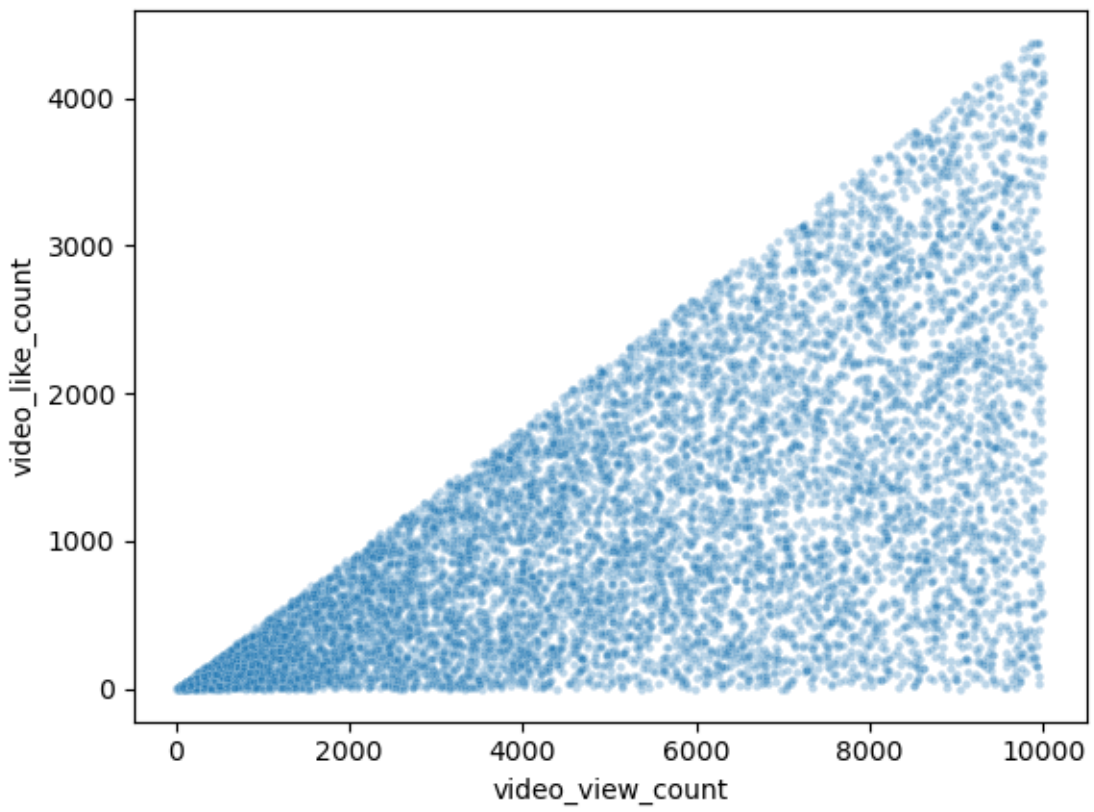
[60]:
```
count_cols = ['video_view_count',
              'video_like_count',
              'video_share_count',
              'video_download_count',
              'video_comment_count',
             ]
for column in count_cols:
    q1 = df[column].quantile(0.25)
    q3 = df[column].quantile(0.75)
    iqr = q3 - q1
    median = df[column].median()
    outlier_threshold = median + 1.5*iqr
    outlier_count = (df[column] > outlier_threshold).sum()
    print(f'Number of outliers, {column}:', outlier_count)
```

```
Number of outliers, video_view_count: 2343
Number of outliers, video_like_count: 3468
Number of outliers, video_share_count: 3732
Number of outliers, video_download_count: 3733
Number of outliers, video_comment_count: 3882
```

[61]:
```
 # Create a scatterplot of 'video_view_count' versus␣
 ↪'video_like_count'according to 'claim_status'
sns.scatterplot(x=df["video_view_count"], y=df["video_like_count"],
hue=df["claim_status"], s=10, alpha=.3)
plt.show()
```



[62]:
```
# Create a scatterplot of 'video_view_count' versus 'video_like_count' for␣
↪opinions only
opinion = df[df['claim_status']=='opinion']
sns.scatterplot(x=opinion["video_view_count"], y=opinion["video_like_count"],
s=10, alpha=.3)
plt.show()
```


video_like_count
4000
3000
2000
1000
0
0
2000
4000
6000
8000
10000
video_view_count

[ ]: