Project name:Instagram Influencer Data Analysis using Python

Internship Project – Unifield Mentor

By: Madhuri Sawant

Tools: Python, Pandas, NumPy, Matplotlib, Seaborn

In [ ]:
```
# Step 1:import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [16]:
```
# LOADING DATASET
df =pd.read_csv("C:\\Users\\SHREE\\Desktop\\Unified Mentor project\\instagram
```

In [18]:
```
# Exploring the data
print(df.head()) # its used give first 10 reords of dataset
```

```
   rank channel_info  influence_score  posts followers avg_likes  \
0     1    cristiano               92   3.3k    475.8m      8.7m
1     2  kyliejenner               91   6.9k    366.2m      8.3m
2     3     leomessi               90  0.89k    357.3m      6.8m
3     4  selenagomez               93   1.8k    342.7m      6.2m
4     5      therock               91   6.8k    334.1m      1.9m

  60_day_eng_rate new_post_avg_like total_likes        country
0           1.39%              6.5m       29.0b          Spain
1           1.62%              5.9m       57.4b  United States
2           1.24%              4.4m        6.0b            NaN
3           0.97%              3.3m       11.5b  United States
4           0.20%            665.3k       12.5b  United States
```

In [19]:
```
print(df.tail(5)) #display the last few rows of dataframe
```

```
     rank channel_info  influence_score  posts followers avg_likes  \
195   196    iambeckyg               71   2.3k     33.2m    623.8k
196   197   nancyajram               81   3.8k     33.2m    390.4k
197   198  luansantana               79  0.77k     33.2m    193.3k
198   199    nickjonas               78   2.3k     33.0m    719.6k
199   200    raisa6690               80   4.2k     32.8m    232.2k

    60_day_eng_rate new_post_avg_like total_likes        country
195           1.40%            464.7k        1.4b  United States
196           0.64%            208.0k        1.5b         France
197           0.26%             82.6k      149.2m         Brazil
198           1.42%            467.7k        1.7b  United States
199           0.30%             97.4k      969.1m      Indonesia
```

In [ ]:
```
print(df.shape) #display numberof columns and rows of a dataframe
print(df.info()) # display summary of dataframe
```

```
(200, 10)
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 10 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   rank               200 non-null    int64
 1   channel_info       200 non-null    object
 2   influence_score    200 non-null    int64
 3   posts              200 non-null    object
 4   followers          200 non-null    object
 5   avg_likes          200 non-null    object
 6   60_day_eng_rate    200 non-null    object
 7   new_post_avg_like  200 non-null    object
 8   total_likes        200 non-null    object
 9   country            138 non-null    object
dtypes: int64(2), object(8)
memory usage: 15.8+ KB
None
```

In [11]:
```
print(df.columns) # display all columns in dataframe
print(df.describe()) # display summary of statistics of dataframe
```

```
Index(['rank', 'channel_info', 'influence_score', 'posts', 'followers',
       'avg_likes', '60_day_eng_rate', 'new_post_avg_like', 'total_likes',
       'country'],
      dtype='object')
             rank  influence_score
count  200.000000       200.000000
mean   100.500000        81.820000
std     57.879185         8.878159
min      1.000000        22.000000
25%     50.750000        80.000000
50%    100.500000        84.000000
75%    150.250000        86.000000
max    200.000000        93.000000
```

In [6]:
```
print(df.value_counts()) # its used to count unique values
```

```
rank  channel_info   influence_score  posts  followers  avg_likes  60_day_eng_r
ate  new_post_avg_like  total_likes  country
1     cristiano      92               3.3k   475.8m     8.7m       1.39%
6.5m               29.0b        Spain           1
2     kyliejenner    91               6.9k   366.2m     8.3m       1.62%
5.9m               57.4b        United States    1
4     selenagomez    93               1.8k   342.7m     6.2m       0.97%
3.3m               11.5b        United States    1
5     therock        91               6.8k   334.1m     1.9m       0.20%
665.3k             12.5b        United States    1
6     kimkardashian  91               5.6k   329.2m     3.5m       0.88%
2.9m               19.9b        United States    1

..
196   iambeckyg      71               2.3k   33.2m      623.8k     1.40%
464.7k             1.4b         United States    1
197   nancyajram     81               3.8k   33.2m      390.4k     0.64%
208.0k             1.5b         France           1
198   luansantana    79               0.77k  33.2m      193.3k     0.26%
82.6k              149.2m       Brazil           1
199   nickjonas      78               2.3k   33.0m      719.6k     1.42%
467.7k             1.7b         United States    1
200   raisa6690      80               4.2k   32.8m      232.2k     0.30%
97.4k              969.1m       Indonesia        1
Name: count, Length: 138, dtype: int64
```

In [21]:
```
df.iloc() # its used select by numeric index position
print(df.iloc[0]) # select first row
print(df.iloc[0:2]) # first two rows
```

```
rank                         1
channel_info         cristiano
influence_score             92
posts                     3.3k
followers                475.8m
avg_likes                 8.7m
60_day_eng_rate          1.39%
new_post_avg_like         6.5m
total_likes              29.0b
country                  Spain
Name: 0, dtype: object
   rank channel_info  influence_score posts followers avg_likes  \
0     1    cristiano               92  3.3k    475.8m      8.7m
1     2  kyliejenner               91  6.9k    366.2m      8.3m

  60_day_eng_rate new_post_avg_like total_likes        country
0           1.39%              6.5m       29.0b          Spain
1           1.62%              5.9m       57.4b  United States
```

In [19]:
```
df.size # it is used return total number of elements in dataframe
```

Out[19]: 2000

In [22]:
```
print(df.isnull()) # display the missing value in dataframe
```

```
# if there is no missing values its gives false(not null)
# if there is missing values value its gives true (null)
```

```
      rank  channel_info  influence_score  posts  followers  avg_likes  \
0    False         False            False  False      False      False
1    False         False            False  False      False      False
2    False         False            False  False      False      False
3    False         False            False  False      False      False
4    False         False            False  False      False      False
..     ...           ...              ...    ...        ...        ...
195  False         False            False  False      False      False
196  False         False            False  False      False      False
197  False         False            False  False      False      False
198  False         False            False  False      False      False
199  False         False            False  False      False      False

     60_day_eng_rate  new_post_avg_like  total_likes  country
0              False              False        False    False
1              False              False        False    False
2              False              False        False     True
3              False              False        False    False
4              False              False        False    False
..               ...                ...          ...      ...
195            False              False        False    False
196            False              False        False    False
197            False              False        False    False
198            False              False        False    False
199            False              False        False    False

[200 rows x 10 columns]
```

In [23]:
```
df.isnull().sum() #to check how many empty valuesare each column
```

Out[23]:
```
rank                 0
channel_info         0
influence_score      0
posts                0
followers            0
avg_likes            0
60_day_eng_rate      0
new_post_avg_like    0
total_likes          0
country             62
dtype: int64
```

In [25]:
```
print(df.dropna()) # helpes in cleaning data by removing missing values in row
```

```
     rank   channel_info  influence_score  posts followers avg_likes  \
0       1      cristiano               92   3.3k    475.8m      8.7m
1       2    kyliejenner               91   6.9k    366.2m      8.3m
3       4    selenagomez               93   1.8k    342.7m      6.2m
4       5        therock               91   6.8k    334.1m      1.9m
5       6  kimkardashian               91   5.6k    329.2m      3.5m
..    ...            ...              ...    ...       ...       ...
195   196      iambeckyg               71   2.3k     33.2m    623.8k
196   197     nancyajram               81   3.8k     33.2m    390.4k
197   198    luansantana               79  0.77k     33.2m    193.3k
198   199      nickjonas               78   2.3k     33.0m    719.6k
199   200      raisa6690               80   4.2k     32.8m    232.2k

    60_day_eng_rate new_post_avg_like total_likes        country
0             1.39%              6.5m       29.0b          Spain
1             1.62%              5.9m       57.4b  United States
3             0.97%              3.3m       11.5b  United States
4             0.20%            665.3k       12.5b  United States
5             0.88%              2.9m       19.9b  United States
..              ...               ...         ...            ...
195           1.40%            464.7k        1.4b  United States
196           0.64%            208.0k        1.5b         France
197           0.26%             82.6k      149.2m         Brazil
198           1.42%            467.7k        1.7b  United States
199           0.30%             97.4k      969.1m      Indonesia

[138 rows x 10 columns]
```

Syntax of handling missing value

- new_variable = df["column_name"].mean() # for numeric
- new_variable = df["column_name"].median() # for numeric
- new_variable = df["column_name"].mode()[0] # for categorical

When cleaning or analyzing data:

- Categorical columns → Use .mode() to fill missing values (most common category)
- Numerical columns → Use .mean() or .median() to fill missing values

In [ ]:
```
# handling missing data
var =df["country"].mode()[0] #Finds most frequent value incountry column [0] g
print(var)
```

```
United States
```

there is two way to filled missing values in columns

- 1. creating new column
- syntax
- declared_variable ["new_column_name"] =

declared_variable["column_name"] .fillna(new_variable)
Ex:df["country_filled"]=df["country"].fillna(var)

- 2. modify existing columns
  - syntax
  - DataFrame["column_name"].fillna(DataFrame["column_name"].method()[0 or None], inplace=True) Ex: df["country"].fillna(df["country"].mode()[0],inplace=true)
  - .fillna() replaces NaN values.
  - .mode()[0] gets the most common value.
  - inplace=True updates the same column (no new one created).

In [40]:
```
# filled the missing value using .fillna
df["country_filled"]=df["country"].fillna(var)
```

In [ ]:
```
df.select_dtypes(include='object').columns #his shows all columns that store t
```

Out[ ]:
```
Index(['channel_info', 'posts', 'followers', 'avg_likes', '60_day_eng_rate',
       'new_post_avg_like', 'total_likes', 'country', 'country_filled'],
      dtype='object')
```

In [ ]:
```
df.select_dtypes(include=['number']).columns #This will show all the column na
```

Out[ ]:
```
Index(['rank', 'influence_score'], dtype='object')
```

In [ ]:
```
df["country_filled"].isnull().sum() # recheck there is any missing values
```

Out[ ]:
```
np.int64(0)
```

In [ ]:
```
df.isnull().sum()  # recheck there is any missing values
```

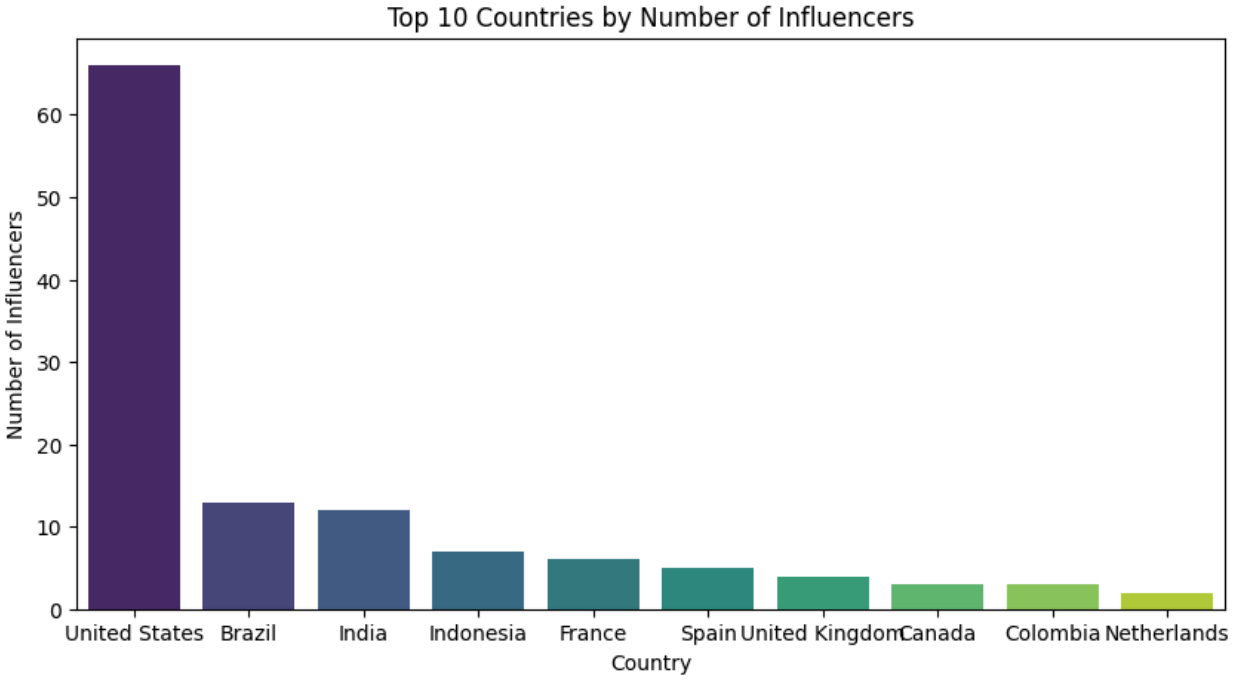
Out[ ]:
```
rank                0
channel_info        0
influence_score     0
posts               0
followers           0
avg_likes           0
60_day_eng_rate     0
new_post_avg_like   0
total_likes         0
country             0
country_filled      0
dtype: int64
```

In [ ]:
```
# basic analysis
print("Total influencers:", len(df))
print("\nTop 5 countries by influencers:\n", df['country'].value_counts().head
```

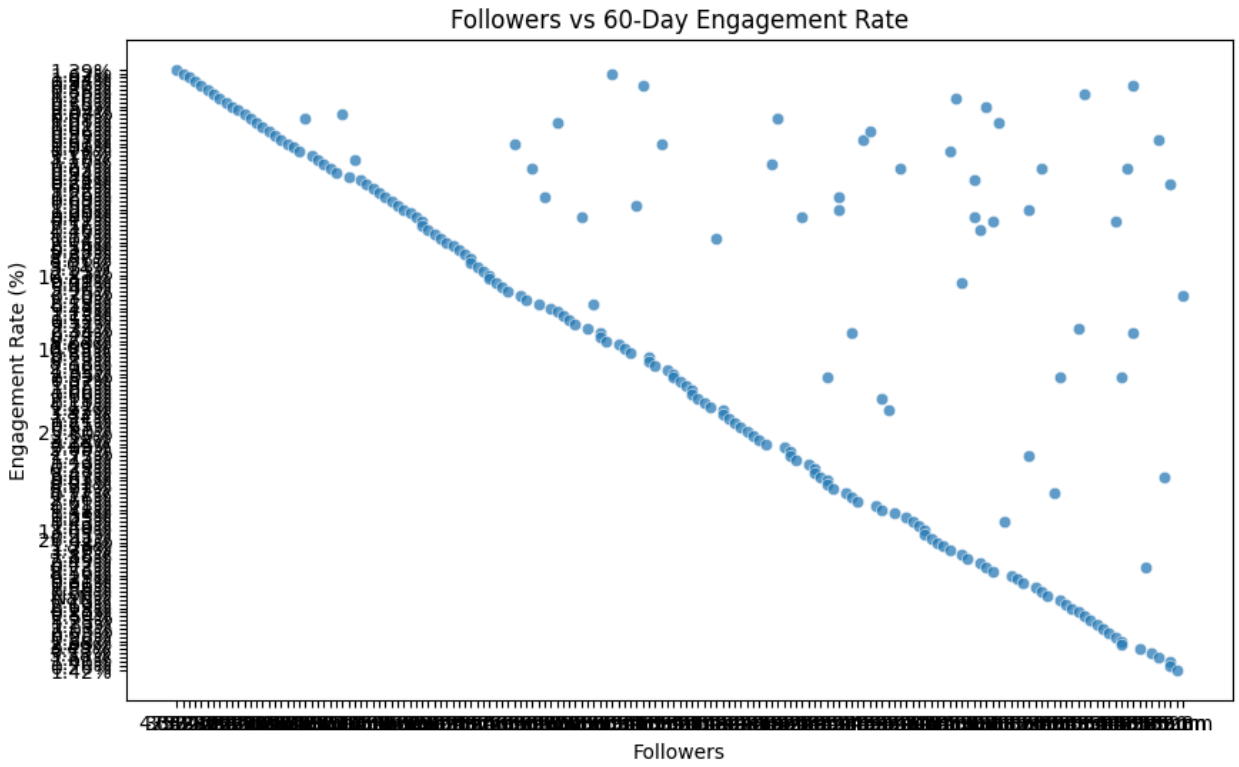
```
Total influencers: 200

Top 5 countries by influencers:
 country
United States    128
Brazil            13
India             12
Indonesia          7
France             6
Name: count, dtype: int64
```

In [28]:
```python
import warnings
warnings.filterwarnings('ignore')
```

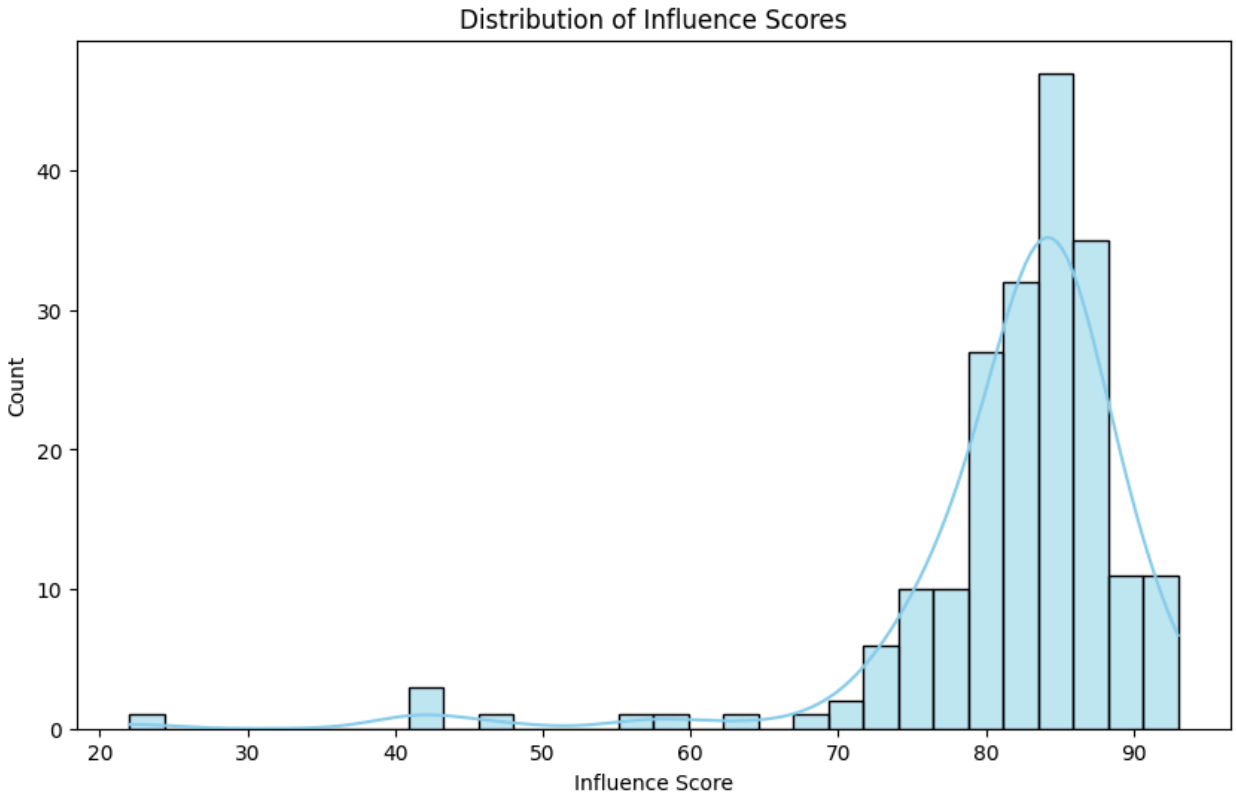
In [ ]:
```python
top_countries = df['country'].value_counts().head(10)
plt.figure(figsize=(10,5))
sns.barplot(x=top_countries.index, y=top_countries.values, palette="viridis")
plt.title('Top 10 Countries by Number of Influencers')
plt.xlabel('Country')
plt.ylabel('Number of Influencers')
plt.show()
```



In [58]:
```python
# Followers vs Engagement Rate
plt.figure(figsize=(10,6))
sns.scatterplot(x='followers',y='60_day_eng_rate', data=df, alpha=0.7)
plt.title('Followers vs 60-Day Engagement Rate')
plt.xlabel('Followers')
plt.ylabel('Engagement Rate (%)')
plt.show()
```



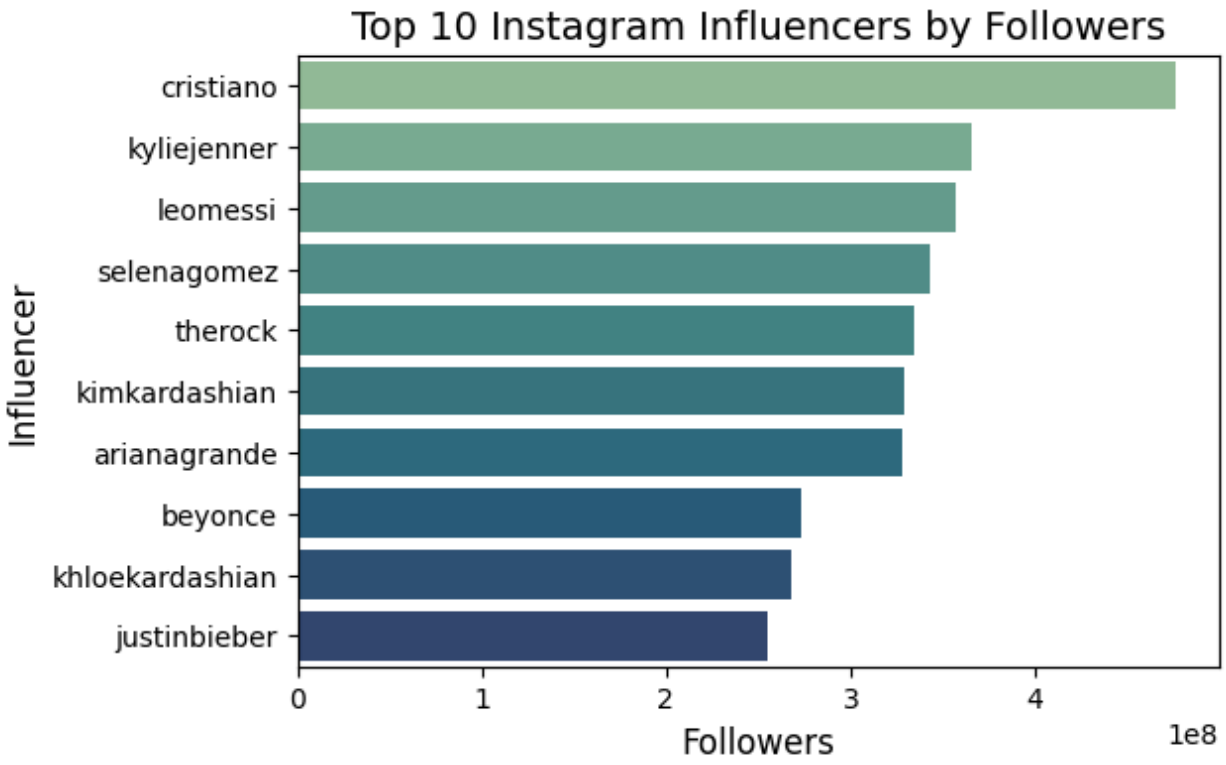
In [62]:
```
# Distribution of Influence Scores
plt.figure(figsize=(10,6))
sns.histplot(df['influence_score'], kde=True, bins=30, color='skyblue')
plt.title('Distribution of Influence Scores')
plt.xlabel('Influence Score')
plt.ylabel('Count')
plt.show()
```



In [35]:
```
#Clean and convert the columns
replace_dict = {'b': 'e9', 'm': 'e6', 'k': 'e3', '%': ''}

cols_to_convert = ['followers', 'avg_likes', 'total_likes', 'posts', 'new_post

for col in cols_to_convert:
    df[col] = df[col].replace(replace_dict, regex=True).astype(float)
```

In [36]:
```
# Select top 10 influencers by number of followers
top10 = df.nlargest(10, 'followers')[['channel_info', 'followers']]
```

In [38]:
```
# Top Influencers by Followers
plt.figure(figsize=(6,4))
sns.barplot(x='followers',
y='channel_info',
data=top10,
palette='crest')
plt.title('Top 10 Instagram Influencers by Followers', fontsize=14)
plt.xlabel('Followers', fontsize=12)
plt.ylabel('Influencer', fontsize=12)
plt.show()
```



In [39]:
```
# Average likes-to-followers ratio
df['like_to_followers_ratio'] = df['avg_likes'] / df['followers']
top_ratio = df.nlargest(10, 'like_to_followers_ratio')[['channel_info', 'like_
print("\nTop 10 influencers with highest like-to-follower ratio:\n")
print(top_ratio)
```

```
Top 10 influencers with highest like-to-follower ratio:

        channel_info  like_to_followers_ratio        country
140              j.m                 0.338902            NaN
102              thv                 0.312373            NaN
167            rkive                 0.294595            NaN
147  jenniferaniston                 0.113022            NaN
155         mahi7781                 0.104859            NaN
118             zayn                 0.101075  United States
114      harrystyles                 0.100213  United States
97             adele                 0.092702  United States
186      blakelively                 0.089595  United States
138       badbunnypr                 0.087886            NaN
```

Instagram Influencer Data Analysis

I explored data from **200 top Instagram influencers** using Python. This project reveals which countries dominate the influencer world , and how engagement rates differ by audience size.

Libraries Used

- pandas
- matplotlib
- seaborn

Key Insights

- Cristiano leads with the highest followers.
- Engagement rate and followers are not directly proportional.
- The USA has the largest share of top influencers.

📊 Summary of Insights

- The United States has the highest number of top influencers.
- Engagement rate and followers are not perfectly correlated — smaller accounts can have higher engagement.
- The average engagement rate across the dataset is around **1–2%**.
- High-follower accounts (>300M) tend to have **lower engagement rates**.