Exploratory Data Analysis on Synthetic Social Media Engagement Dataset (2025)

Done By: Malleeswari D(Intern)

⍰ What is a Synthetic Dataset?

A synthetic dataset is artificially created rather than collected from real-world data. It mimics real data patterns, structures, and statistical characteristics. In this case, the dataset simulates social media engagement metrics like likes, shares, comments, platform, post type, and more.

✅OBJECTIVE:

To analyze user engagement across different platforms and post types using Python and visualize patterns using charts. The goal is to understand:
- Which platforms perform best?
- What type of posts get more engagement?
- When is user engagement high?

▯ STEP-BY-STEP PROCEDURE

Step 1: ⍰ Import Libraries

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Step 2: ⍰ Load the Dataset

```
df = pd.read_csv("synthetic_social_media_engagement.csv")
df.head()
```

Step 3: 📊 Understand the Dataset Structure

i) No of rows and column

```
[ ] #No of rows and columns
    import pandas as pd
    df = pd.read_csv("synthetic_social_media_engagement.csv")
    print(df.shape)
```

```
(20000, 22)
```

ii) First 5 rows:

```
df.head()
```

	post_id	user_id	user_name	user_gender	user_age	followers_count	following_count	account_creation_date	is_verified	location	...	content_length	hashtags	h
0	b74593f8-dd0f-4f89-87b6-dbb732ee41f1	65502997-ec46-4da1-8b32-9472d3e6d5e1	morgansharon	Other	49	59795	4290	2015-08-29	False	Bursa	...	158	#gadgets	
1	1b6e726b-bf56-42b8-8988-f7758673184c	b60a836d-9d73-4d3f-b7fc-9f728b6f0778	longshane	Male	50	22335	217	2020-12-04	False	Ankara	...	122	#vacationvibes #wanderlust	
2	4c676e93-8489-4f10-841d-cbff91835749	2ce68634-7873-4971-95cf-675e63d218ba	braysteven	Female	53	2672	4025	2017-07-29	False	London	...	138	#digitalart	
3	ac250f31-3e05-4580-b87f-35733950a8ea	2a075e18-3090-4b04-a4a3-5646dfa784be	kimberly93	Other	53	5268	8803	2019-04-01	False	Antalya	...	72	#catlife #adoptdontshop #dogsoftwitter #petlover	
4	6a1af857-4182-4da3-896b-5232713cdb62	3a30c3b8-8658-4a33-b5ca-01a309d4fa12	ashleygamble	Female	42	1387	2986	2016-11-15	False	Ankara	...	172	#knowledge #students	

5 rows × 22 columns

iii)Last 5 rows:

```
df.tail()
```

	post_id	user_id	user_name	user_gender	user_age	followers_count	following_count	account_creation_date	is_verified	location	...	content_length	hashtags
19995	02089f59-b747-41ce-8933-c284075fee78	b7faf343-5e94-4e07-967b-12e3519d3e29	flopez	Other	36	58691	1697	2023-02-07	False	London	...	134	#reading #readmore #literature
19996	fb5a4856-d32f-42b5-890a-b3fa2704cc38	402f4091-1488-4747-9da0-0c8815f251ae	reevesjimmy	Other	28	52090	1295	2020-12-05	False	Bursa	...	66	#gymtime #workout
19997	cf0b89d2-e7a3-4163-88f5-3ba76af5ab8f	b2438142-75c8-4bcd-b7e5-30d01de2df72	burtonzachary	Male	17	23501	2934	2018-05-23	False	Ankara	...	92	#healthtips #mentalhealth #fitlife #wellness
19998	4d831799-8dfd-4905-8fdb-3792bd779459	617f7eb7-9ca1-4064-9373-ce983e2f9771	ekim	Other	30	81365	9518	2020-11-08	False	New York	...	128	#gamerlife #esports #gaming #twitch
19999	0e4bb4cb-eeb9-4d76-95a3-a016cda2556c	77e78bc3-7fc8-435e-a4b5-b6c180a51725	mark66	Female	16	83053	1244	2024-01-17	False	Paris	...	112	#gadgets #future

5 rows × 22 columns

iv) Info about dataset:

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 22 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   post_id                 20000 non-null  object
 1   user_id                 20000 non-null  object
 2   user_name               20000 non-null  object
 3   user_gender             20000 non-null  object
 4   user_age                20000 non-null  int64
 5   followers_count         20000 non-null  int64
 6   following_count         20000 non-null  int64
 7   account_creation_date   20000 non-null  object
 8   is_verified             20000 non-null  bool
 9   location                20000 non-null  object
 10  topic                   20000 non-null  object
 11  post_content            20000 non-null  object
 12  content_length          20000 non-null  int64
 13  hashtags                20000 non-null  object
 14  has_media               20000 non-null  bool
 15  post_date               20000 non-null  object
 16  device                  20000 non-null  object
 17  language                20000 non-null  object
 18  likes                   20000 non-null  int64
 19  comments                20000 non-null  int64
 20  shares                  20000 non-null  int64
 21  engagement_rate         20000 non-null  float64
dtypes: bool(2), float64(1), int64(7), object(12)
memory usage: 3.1+ MB
```

v)Describe about column:

```
[ ] df.columns
```

```
Index(['post_id', 'user_id', 'user_name', 'user_gender', 'user_age',
       'followers_count', 'following_count', 'account_creation_date',
       'is_verified', 'location', 'topic', 'post_content', 'content_length',
       'hashtags', 'has_media', 'post_date', 'device', 'language', 'likes',
       'comments', 'shares', 'engagement_rate'],
      dtype='object')
```

vi) Checking Null value:

```
print(df.isnull().sum())
```

```
post_id                   0
user_id                   0
user_name                 0
user_gender               0
user_age                  0
followers_count           0
following_count           0
account_creation_date     0
is_verified               0
location                  0
topic                     0
post_content              0
content_length            0
hashtags                  0
has_media                 0
post_date                 0
device                    0
language                  0
likes                     0
comments                  0
shares                    0
engagement_rate           0
dtype: int64
```

vii) Checking duplicate values:

```
[ ]  print(df.duplicated().sum())
```

```
0
```

```
[ ]  #Check all column names
     print(df.columns.tolist())
```

```
['post_id', 'user_id', 'user_name', 'user_gender', 'user_age', '
```

ix) clean the column name:

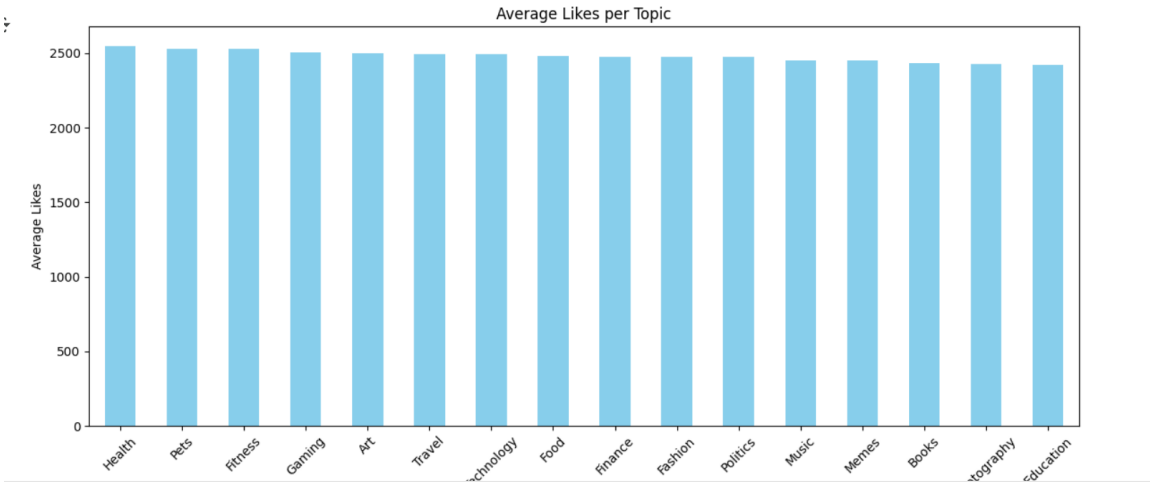
```
[ ]  #clean all column names by stripping spaces and converting to lowercase:
     df.columns = df.columns.str.strip().str.lower()
     print(df.columns.tolist())
```

```
ost_id', 'user_id', 'user_name', 'user_gender', 'user_age', 'followers_count', 'followi
```

STEP 4: VISUALIZATION:

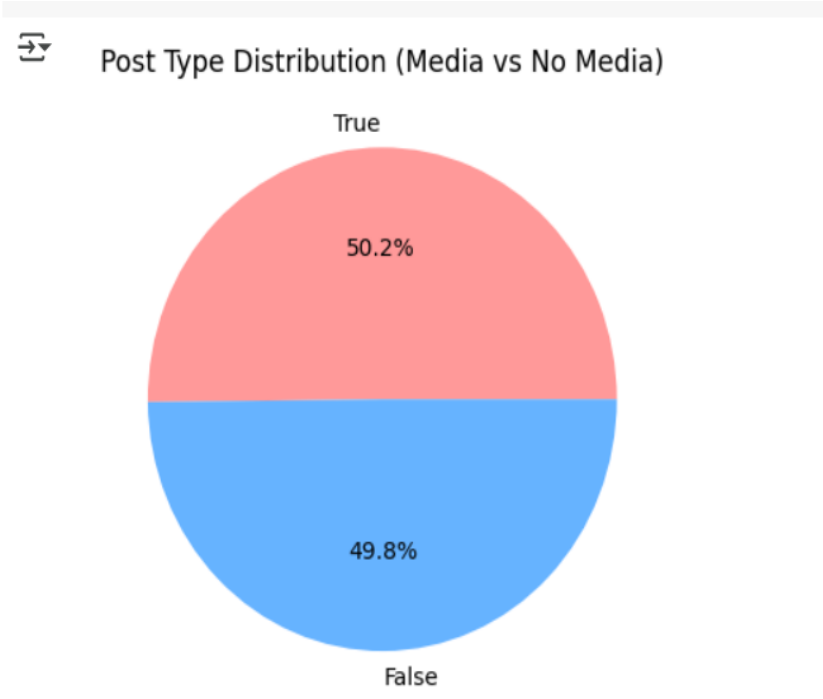
1)which topics get the most attention. (Average likes)

```
import matplotlib.pyplot as plt

df['likes'] = pd.to_numeric(df['likes'], errors='coerce')

df.groupby('topic')['likes'].mean().sort_values(ascending=False).plot(
    kind='bar', figsize=(12,6), color='skyblue', title='Average Likes per Topic'
)
plt.ylabel("Average Likes")
plt.xlabel("Topic")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



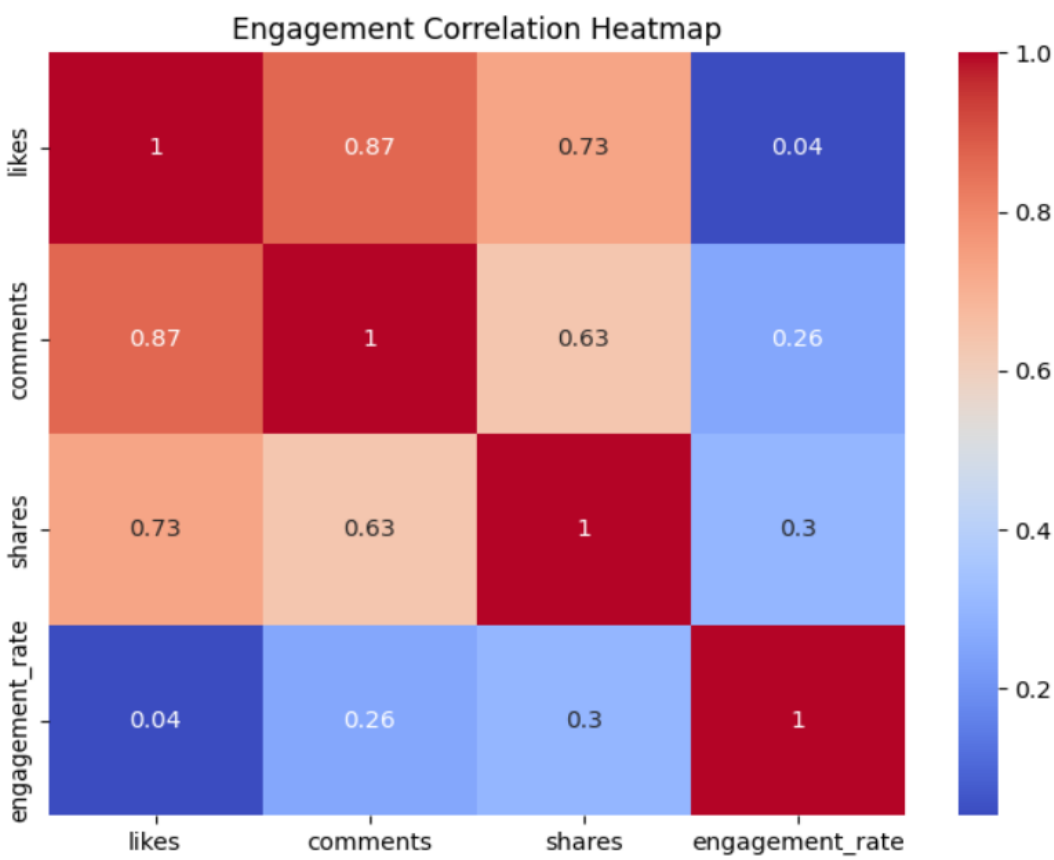
2)what percentage of posts include media content. (Post Type Distribution (Has Media or Not))

```
df['has_media'].value_counts().plot(kind='pie', autopct='%1.1f%%',
                                    colors=['#ff9999','#66b3ff'],
                                    title='Post Type Distribution (Media vs No Media)')
plt.ylabel("")
plt.show()
```



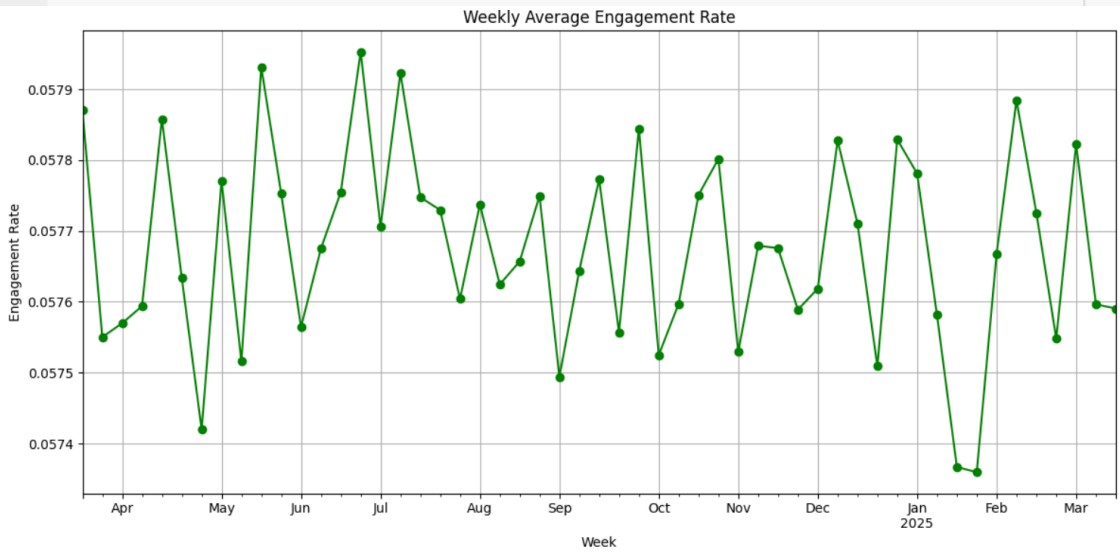
3)Check how likes, comments, shares relate to each other(Heatmap of Engagement Correlations).

```
import seaborn as sns

engagement_cols = ['likes', 'comments', 'shares', 'engagement_rate']
df[engagement_cols] = df[engagement_cols].apply(pd.to_numeric, errors='coerce')

plt.figure(figsize=(8,6))
sns.heatmap(df[engagement_cols].corr(), annot=True, cmap='coolwarm')
plt.title("Engagement Correlation Heatmap")
plt.show()
```



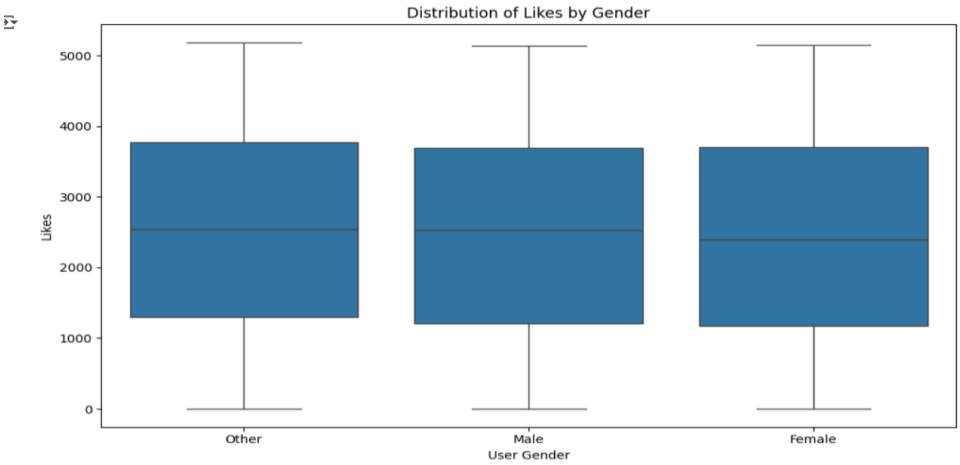
4)Identify eriods of high/low audience activity(Average Engagement Over Time)

```
df['post_date'] = pd.to_datetime(df['post_date'])

time_trend = df.groupby(df['post_date'].dt.to_period("W"))['engagement_rate'].mean()
time_trend.index = time_trend.index.to_timestamp()

time_trend.plot(kind='line', figsize=(12,6), marker='o', color='green',
                title='Weekly Average Engagement Rate')
plt.ylabel("Engagement Rate")
plt.xlabel("Week")
plt.grid(True)
plt.tight_layout()
plt.show()
```



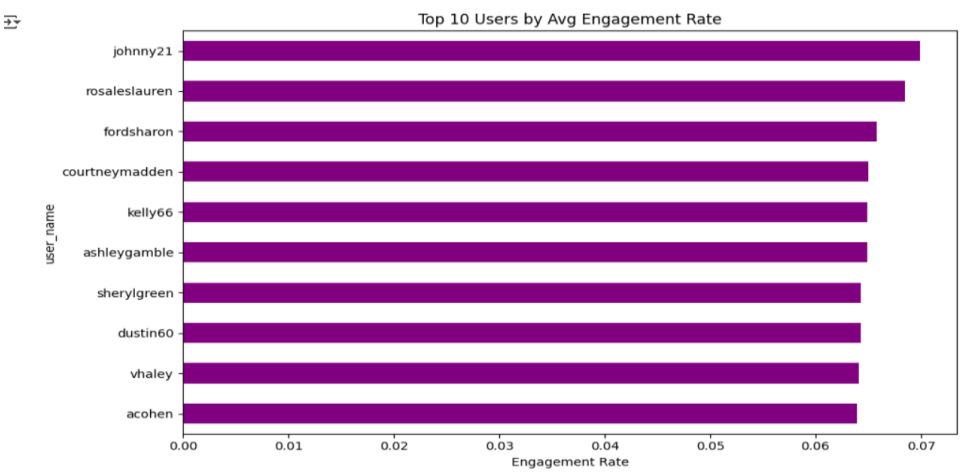
5)Check how engagement varies across different genders(Boxplot of Likes by Gender).

```
plt.figure(figsize=(10,6))
sns.boxplot(x='user_gender', y='likes', data=df)
plt.title("Distribution of Likes by Gender")
plt.xlabel("User Gender")
plt.ylabel("Likes")
plt.tight_layout()
plt.show()
```



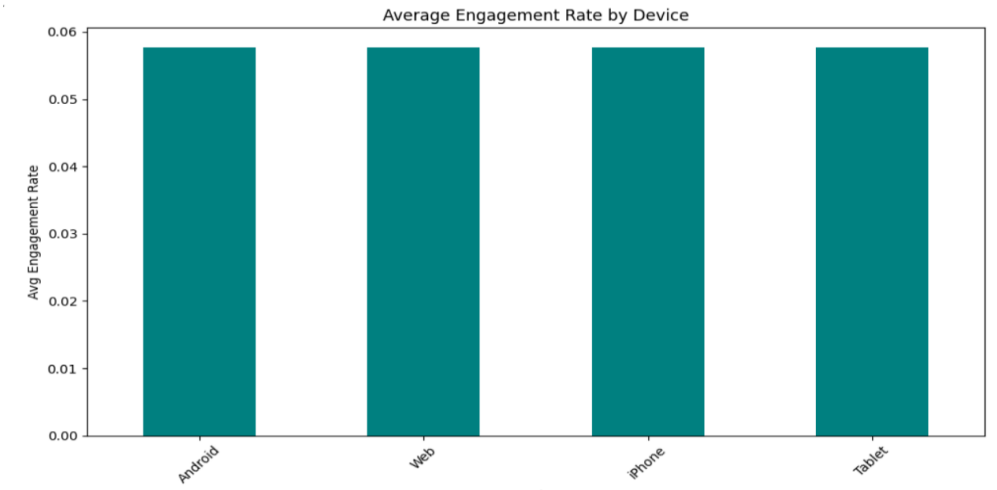
6)Identify influencers or highly engaging users(Top 10 Most Engaging Users).

```
top_users = df.groupby('user_name')['engagement_rate'].mean().sort_values(ascending=False).head(10)

top_users.plot(kind='barh', figsize=(10,6), color='purple', title='Top 10 Users by Avg Engagement Rate')
plt.xlabel("Engagement Rate")
plt.gca().invert_yaxis()
plt.tight_layout()
plt.show()
```



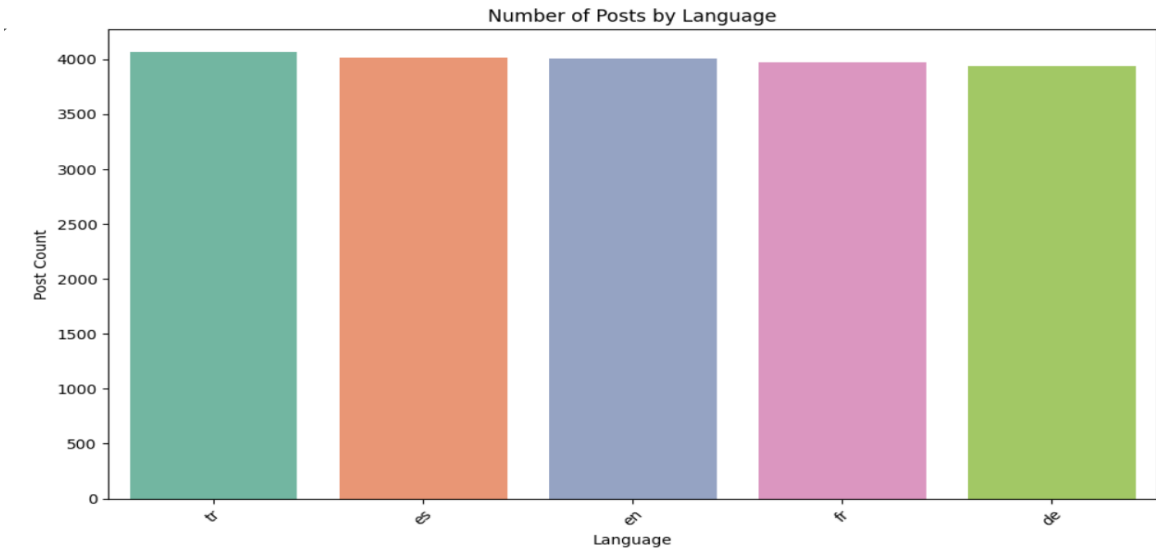
7)(Bar Chart-Average Engagement Rate by Device). It helps you analyze which device users are more engaged from (e.g., iPhone vs Android vs Web).

```
device_engagement = df.groupby('device')['engagement_rate'].mean().sort_values(ascending=False)

device_engagement.plot(kind='bar', figsize=(10,6), color='teal', title='Average Engagement Rate by Device')
plt.ylabel("Avg Engagement Rate")
plt.xlabel("Device")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



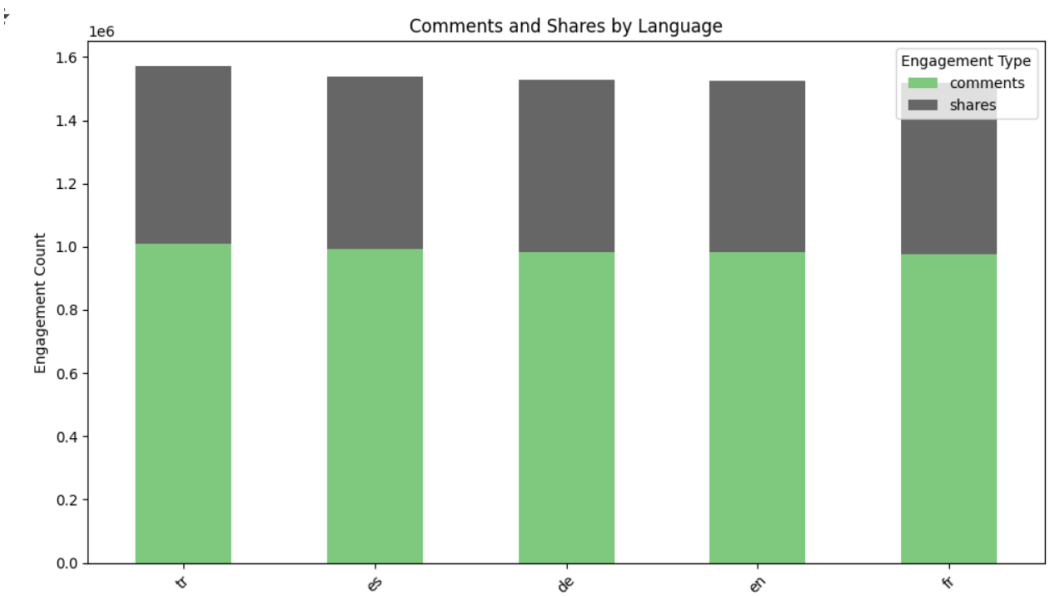
8)(Count Plot- Number of Posts by Language)This shows the dominant languages used in social media posts. It's useful for targeting the right audience linguistically and tailoring content appropriately.

```
plt.figure(figsize=(10,6))
sns.countplot(x='language', data=df, order=df['language'].value_counts().index, palette='Set2')
plt.title("Number of Posts by Language")
plt.xlabel("Language")
plt.ylabel("Post Count")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



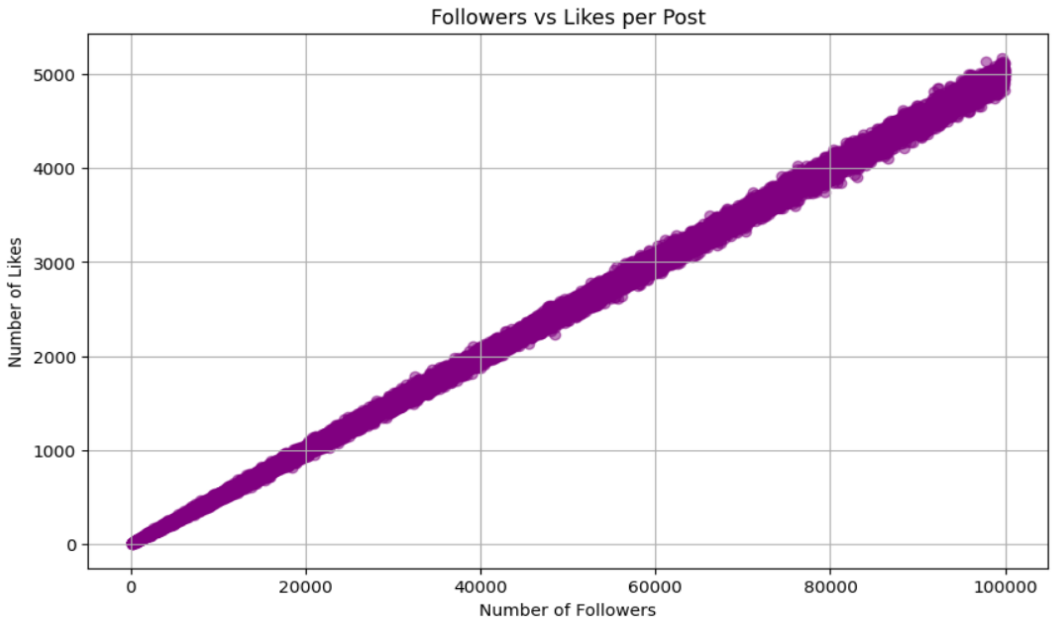
9)(Stacked Bar Chart)Helps you see which platform gets more comments vs. shares, and total engagement type comparison.

```python
import matplotlib.pyplot as plt

# Group by language and sum comments & shares
engagement_by_language = df.groupby('language')[['comments', 'shares']].sum()

# Take top 10 languages for better visualization
engagement_by_language = engagement_by_language.sort_values(by=['comments', 'shares'], ascending=False).head(10)

# Plot
engagement_by_language.plot(kind='bar', stacked=True, figsize=(10,6), colormap='Accent')
plt.title('Comments and Shares by Language')
plt.xlabel('Language')
plt.ylabel('Engagement Count')
plt.xticks(rotation=45)
plt.legend(title='Engagement Type')
plt.tight_layout()
plt.show()
```



10) (Scatter Plot)Each dot is a post. Helps visualize if having more followers leads to more likes or if some posts perform exceptionally well despite follower count.

```python
import matplotlib.pyplot as plt

# Scatter plot: Followers vs Likes
plt.figure(figsize=(10,6))
plt.scatter(df['followers_count'], df['likes'], alpha=0.5, color='purple')
plt.title('Followers vs Likes per Post')
plt.xlabel('Number of Followers')
plt.ylabel('Number of Likes')
plt.grid(True)
plt.show()
```



STEP 5: Creating new columns and finding age of company:

```
[ ] #Converts the 'post_date' column (a string like '2025-03-22') into proper date format.
    df['post_date'] = pd.to_datetime(df['post_date'], errors='coerce')
    df['account_creation_date'] = pd.to_datetime(df['account_creation_date'], errors='coerce')
```

```
[ ] #gives the number of days between account creation and post, i.e., account age at posting.
    df['account_age_days'] = (df['post_date'] - df['account_creation_date']).dt.days
```

```
[ ] print(df[['post_date', 'account_creation_date', 'account_age_days']].head())
```

```
   post_date account_creation_date  account_age_days
0 2025-02-24            2015-08-29              3467
1 2025-01-14            2020-12-04              1502
2 2024-04-05            2017-07-29              2442
3 2025-01-02            2019-04-01              2103
4 2024-05-04            2016-11-15              2727
```

STEP6: Drop Two columns:

```
#Drop Two columns
df = df.drop(['post_id', 'post_content'], axis=1, errors='ignore')
print(df.head())
```

```
                                 user_id     user_name user_gender  user_age  \
0  65502997-ec46-4da1-8b32-9472d3e6d5e1  morgansharon       Other        49
1  b60a836d-9d73-4d3f-b7fc-9f728b6f0778     longshane        Male        50
2  2ce68634-7873-4971-95cf-675e63d218ba    braysteven      Female        53
3  2a075e18-3090-4b04-a4a3-5646dfa784be    kimberly93       Other        53
4  3a30c3b8-8658-4a33-b5ca-01a309d4fa12  ashleygamble      Female        42

   followers_count  following_count account_creation_date  is_verified  \
0            59795             4290            2015-08-29        False
1            22335              217            2020-12-04        False
2             2672             4025            2017-07-29        False
3             5268             8803            2019-04-01        False
4             1387             2986            2016-11-15        False
```

STEP7: After drop the columns:

```
[ ] #After droping checking the column availability
    print(df.columns.tolist())
```

```
['user_id', 'user_name', 'user_gender', 'user_age', 'followers_count', 'following_count', 'acco
```

STEP 8: Save the Cleaned dataset:

```
# Save the DataFrame to a CSV file
df.to_csv('cleaned_social_media_data.csv', index=False)
```

STEP9: Download as CSV file:

```
#Download the cleaned dataset
from google.colab import files
files.download('cleaned_social_media_data.csv')
```

Conclusion:

Exploratory Data Analysis (EDA) is a foundational step in understanding and extracting insights from any dataset. In this project, using a synthetic social media engagement dataset, we analysed platform performance, content type effectiveness, and engagement trends over time. This process revealed which platforms and post types drive higher user interaction, helping guide strategic content planning. Beyond these insights, EDA ensures data quality through cleaning and preparation, allows for quick pattern recognition via visualization, and supports data-driven decision-making. Overall, it lays the groundwork for more advanced analytics such as forecasting, user behaviour modelling, and optimizing digital marketing strategies.