1.Social Media Secrets: Understanding User Behavior Step-by-Step Implementation: Problem Understanding:

We want to analyze a social media dataset that includes user engagement metrics such as likes, shares, comments, etc. We will make predictions about user engagement patterns and identify posts that are likely to go viral. Dataset Creation:

Simulate a synthetic dataset that mimics real-world social media interactions. Key variables: likes, shares, comments, post type, user age, user location, and engagement time. Data Exploration:

Explore the dataset to understand engagement patterns, times, and post types. Prediction:

Use simple criteria (e.g., engagement thresholds) to predict which posts are likely to go viral. Create a method to predict viral posts based on the total engagement (likes + shares + comments). Visualization:

Use matplotlib and seaborn to visualize trends and patterns in the data.

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

In [ ]:
```
#Step 2: Simulating Social Media Data
# Simulating a social media dataset
np.random.seed(42)  # For reproducibility

# Number of sample posts
n_posts = 1000

# Generate random data for social media posts
data = {
    'post_id': np.arange(1, n_posts + 1),
    'likes': np.random.randint(0, 1000, n_posts),
    'shares': np.random.randint(0, 500, n_posts),
    'comments': np.random.randint(0, 300, n_posts),
    'post_type': np.random.choice(['Image', 'Video', 'Text', 'Link'], n_posts),
    'user_age': np.random.randint(18, 65, n_posts),
    'user_location': np.random.choice(['USA', 'India', 'UK', 'Canada'], n_posts)
    'engagement_time': np.random.choice(['Morning', 'Afternoon', 'Evening'], n_p
}

# Create DataFrame
df = pd.DataFrame(data)

# Display the first few rows of the dataset
print("Dataset Sample:\n", df.head())
```

In [ ]:
```
#Step 3: Exploring User Engagement Patterns
#When do users engage the most?
```

```
# Explore the engagement times (Morning, Afternoon, Evening)
engagement_counts = df['engagement_time'].value_counts()
print("\nEngagement Time Distribution:\n", engagement_counts)

# Plotting the engagement times
plt.figure(figsize=(8, 6))
sns.countplot(x='engagement_time', data=df, palette='coolwarm')
plt.title('User Engagement by Time of Day')
plt.xlabel('Engagement Time')
plt.ylabel('Frequency')
plt.show()
```

In [ ]:

```
#Which post types generate the most engagement?
# Create a new column for total engagement (likes + shares + comments)
df['total_engagement'] = df['likes'] + df['shares'] + df['comments']

# Aggregating total engagement by post type
engagement_by_type = df.groupby('post_type')['total_engagement'].mean().sort_val
print("\nAverage Engagement by Post Type:\n", engagement_by_type)

# Plotting average engagement by post type
plt.figure(figsize=(8, 6))
engagement_by_type.plot(kind='bar', color='skyblue')
plt.title('Average Engagement by Post Type')
plt.xlabel('Post Type')
plt.ylabel('Average Engagement')
plt.xticks(rotation=45)
plt.show()
```

In [ ]:

```
#Step 4: Predicting Viral Posts
# Calculate the 80th percentile of total engagement
engagement_threshold = np.percentile(df['total_engagement'], 80)

# Create a new column to identify viral posts (engagement above the 80th percent
df['viral_post'] = df['total_engagement'] > engagement_threshold

# Viewing viral posts
viral_posts = df[df['viral_post'] == True]
print("\nViral Posts Sample:\n", viral_posts[['post_id', 'total_engagement', 'pc

# Visualization of viral posts engagement by type
plt.figure(figsize=(8, 6))
sns.boxplot(x='post_type', y='total_engagement', data=df, palette='Set2')
plt.title('Viral Post Engagement by Type')
plt.xlabel('Post Type')
plt.ylabel('Engagement')
plt.show()

# Counting viral posts by post type
most_likely_to_go_viral = viral_posts.groupby('post_type')['post_id'].count()
print("\nMost Likely to Go Viral (Post Types):\n", most_likely_to_go_viral)
```

In [ ]: