import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from prophet import Prophet

import geopandas as gpd

import plotly.express as px

# Load dataset

df = pd.read\_json("cleaned\_data.json")

### 📊 USER ENGAGEMENT ANALYSIS ###

df["engagement\_score"] = df["score"] + (df["num\_comments"] \* 2) # Weight comments more

plt.figure(figsize=(10, 5))

sns.histplot(df["engagement\_score"], bins=50, kde=True)

plt.title("User Engagement Score Distribution")

plt.xlabel("Engagement Score")

plt.ylabel("Frequency")

plt.show()

### 🌍 LOCATION ANALYSIS ###

df["location"] = df["location"].fillna("Unknown") # Handle missing locations

# Count posts by location

location\_counts = df["location"].value\_counts().reset\_index()

location\_counts.columns = ["Location", "Count"]

fig = px.bar(location\_counts, x="Location", y="Count", title="Post Count by Location")

fig.show()

# If you want to visualize location data on a map:

# Ensure you have a dataset that maps locations to lat/lon

# Example: Indian states map (Uncomment below if applicable)

# india\_map = gpd.read\_file("india\_states.geojson") # Load India map GeoJSON

# india\_map = india\_map.merge(location\_counts, left\_on="state\_name", right\_on="Location", how="left")

# india\_map.plot(column="Count", cmap="Blues", legend=True)

# plt.title("Engagement by State")

# plt.show()

### 📈 TREND ANALYSIS ###

df["created\_utc"] = pd.to\_datetime(df["created\_utc"]) # Convert to datetime

# Group by date and aggregate scores to analyze trends

time\_series = df.groupby(df["created\_utc"].dt.date)["score"].sum().reset\_index()

time\_series.columns = ["ds", "y"]

# Apply Prophet for trend forecasting

model = Prophet()

model.fit(time\_series)

future = model.make\_future\_dataframe(periods=30) # Predict for next 30 days

forecast = model.predict(future)

fig = model.plot(forecast)

plt.title("Trend Analysis of Post Scores Over Time")

plt.show()

### 🌍 CATEGORY VS LOCATION ANALYSIS ###

df["location"] = df["location"].fillna("Unknown") # Handle missing locations

df["category"] = df["category"].fillna("Uncategorized") # Handle missing categories

# Count posts by category and location

category\_location\_counts = df.groupby(["location", "category"]).size().reset\_index(name="Count")

# Plot category-wise distribution across locations

fig = px.bar(

category\_location\_counts,

x="location",

y="Count",

color="category",

title="Category Distribution by Location",

text="Count",

barmode="stack"

)

fig.update\_traces(textposition="outside")

fig.show()

### 📈 CATEGORY-WISE TREND ANALYSIS ###

df["created\_utc"] = pd.to\_datetime(df["created\_utc"]) # Convert to datetime

df["engagement\_score"] = df["score"] + (df["num\_comments"] \* 2) # Calculate engagement score

# Aggregate engagement scores by date and category

category\_trend = df.groupby([df["created\_utc"].dt.date, "category"])["engagement\_score"].sum().reset\_index()

category\_trend.columns = ["Date", "Category", "Engagement\_Score"]

# Plot category-wise engagement trends

fig = px.line(

category\_trend,

x="Date",

y="Engagement\_Score",

color="Category",

title="Category-Wise Engagement Trends Over Time"

)

fig.show()