import praw

import sqlite3

import pandas as pd

from datetime import datetime, timedelta

from textblob import TextBlob

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

from langchain\_community.utilities import SQLDatabase

from langchain\_experimental.sql import SQLDatabaseChain

from sqlalchemy import create\_engine

import streamlit as st

import matplotlib.pyplot as plt

from wordcloud import WordCloud

from langchain\_community.chat\_models import ChatOpenAI *# Fixed import*

*# Reddit API credentials*

CLIENT\_ID = "Fj0Sk2BLVJ0PCBA\_r41zEg"

CLIENT\_SECRET = "arhY7vtO1j-XiMsyyiwl4kKWI\_PLeQ"

USER\_AGENT = "python:RedditScraper:v1.0 (by u/Monk481)"

OPENAI\_API\_KEY = "sk-proj-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" *# Masked key for security*

*# Initialize Reddit API*

reddit = praw.Reddit(client\_id=CLIENT\_ID, client\_secret=CLIENT\_SECRET, user\_agent=USER\_AGENT)

*# Connect to SQLite*

conn = sqlite3.connect("reddit\_analysis.db", check\_same\_thread=False)

cursor = conn.cursor()

*# Create Database Table (if not exists)*

cursor.execute("""

CREATE TABLE IF NOT EXISTS reddit\_data (

id INTEGER PRIMARY KEY AUTOINCREMENT,

type TEXT,

subreddit TEXT,

score INTEGER,

content TEXT,

url TEXT,

created\_at TEXT,

sentiment TEXT

)

""")

conn.commit()

*# \*\*Function to Scrape Reddit Data\*\**

**def** store\_reddit\_data(years=2):

"""Fetch all posts & comments from the user 'opinionsareus' within the last X years."""

username = "opinionsareus" *# Fixed username*

user = reddit.redditor(username)

data = []

cutoff\_time = datetime.utcnow() - timedelta(days=years \* 365)

for submission in user.submissions.new(limit=None):

post\_time = datetime.utcfromtimestamp(submission.created\_utc)

if post\_time < cutoff\_time:

break

data.append(("Post", submission.subreddit.display\_name, submission.score,

submission.title, submission.url, post\_time, None))

for comment in user.comments.new(limit=None):

comment\_time = datetime.utcfromtimestamp(comment.created\_utc)

if comment\_time < cutoff\_time:

break

data.append(("Comment", comment.subreddit.display\_name, comment.score,

comment.body, **f**"https://www.reddit.com{comment.permalink}", comment\_time, None))

cursor.executemany("INSERT INTO reddit\_data (type, subreddit, score, content, url, created\_at, sentiment) VALUES (?, ?, ?, ?, ?, ?, ?)", data)

conn.commit()

return len(data)

*# \*\*Function to Perform Sentiment Analysis\*\**

**def** analyze\_sentiment():

"""Perform sentiment analysis on all stored posts and comments."""

vader = SentimentIntensityAnalyzer()

cursor.execute("SELECT id, content FROM reddit\_data WHERE sentiment IS NULL")

rows = cursor.fetchall()

for row\_id, text in rows:

sentiment\_score = vader.polarity\_scores(text)["compound"] if len(text.split()) < 5 else TextBlob(text).sentiment.polarity

sentiment = "Positive" if sentiment\_score > 0 else "Negative" if sentiment\_score < 0 else "Neutral"

cursor.execute("UPDATE reddit\_data SET sentiment = ? WHERE id = ?", (sentiment, row\_id))

conn.commit()

*# \*\*Function to Plot Sentiment Over Time\*\**

**def** plot\_sentiment\_over\_time():

"""Visualizes how the sentiment of user comments changed over time."""

df = pd.read\_sql("SELECT created\_at, sentiment FROM reddit\_data WHERE type='Comment'", conn)

df["created\_at"] = pd.to\_datetime(df["created\_at"])

df["Month"] = df["created\_at"].dt.to\_period("M")

sentiment\_counts = df.groupby(["Month", "sentiment"]).size().unstack(fill\_value=0)

fig, ax = plt.subplots(figsize=(10, 5))

sentiment\_counts.plot(kind="line", ax=ax, marker="o")

ax.set\_title("Sentiment of User Comments Over Time")

ax.set\_xlabel("Time (Months)")

ax.set\_ylabel("Number of Comments")

ax.legend(title="Sentiment")

ax.grid(True)

st.pyplot(fig)

*# \*\*Function to Generate Word Cloud\*\**

**def** generate\_wordcloud(start\_date, end\_date):

"""Generates a word cloud for comments within a selected date range."""

start\_date = pd.to\_datetime(start\_date)

end\_date = pd.to\_datetime(end\_date)

df\_comments = pd.read\_sql("SELECT content, created\_at FROM reddit\_data WHERE type='Comment'", conn)

df\_comments["created\_at"] = pd.to\_datetime(df\_comments["created\_at"])

df\_filtered = df\_comments[(df\_comments["created\_at"] >= start\_date) & (df\_comments["created\_at"] <= end\_date)]

if df\_filtered.empty:

st.write("No comments found for the selected time range.")

return

text = " ".join(df\_filtered["content"])

wordcloud = WordCloud(width=800, height=400, background\_color="white").generate(text)

fig, ax = plt.subplots()

ax.imshow(wordcloud, interpolation="bilinear")

ax.axis("off")

st.pyplot(fig)

*# \*\*Function for AI-Powered Queries\*\**

**def** query\_reddit\_data(natural\_language\_query):

"""Allows users to query Reddit data using natural language."""

engine = create\_engine("sqlite:///reddit\_analysis.db")

db = SQLDatabase(engine)

llm = ChatOpenAI(temperature=0, model="gpt-3.5-turbo", openai\_api\_key=OPENAI\_API\_KEY)

db\_chain = SQLDatabaseChain.from\_llm(llm, db, verbose=True)

response = db\_chain.run(natural\_language\_query)

return response

*# \*\*Streamlit Web App\*\**

st.title(" Reddit User Analysis Tool for 'opinionsareus'")

if st.button("Scrape & Analyze Data for 'opinionsareus'"):

st.write("Fetching data...")

num\_entries = store\_reddit\_data(years=2)

analyze\_sentiment()

st.success(**f**" Scraped & Analyzed {num\_entries} posts and comments!")

*# \*\*Sentiment Over Time\*\**

st.subheader(" Sentiment Over Time (Last 2 Years)")

plot\_sentiment\_over\_time()

*# \*\*Word Cloud with Date Selection\*\**

st.subheader(" Most Frequent Words in User's Comments (Select Time Period)")

col1, col2 = st.columns(2)

with col1:

start\_date = st.date\_input("Start Date", datetime.today().date() - timedelta(days=365\*2))

with col2:

end\_date = st.date\_input("End Date", datetime.today().date())

if st.button("Generate Word Cloud"):

generate\_wordcloud(start\_date, end\_date)

*# \*\*AI-Powered Querying\*\**

st.header(" AI-Powered Queries")

query = st.text\_input("Ask a question about the user's Reddit activity:", "Show me all negative comments.")

if st.button("Run Query"):

response = query\_reddit\_data(query)

st.write(" AI Response:", response)