import streamlit as st

import pandas as pd

import numpy as np

import yfinance as yf

from datetime import datetime, timedelta

# Streamlit app title

st.title("Stock RS Analysis Dashboard")

# Sidebar for user inputs

st.sidebar.header("Input Parameters")

ticker = st.sidebar.text\_input("Enter Stock Ticker", value="UBER")

start\_date = st.sidebar.date\_input("Start Date", value=datetime(2024, 7, 1))

end\_date = st.sidebar.date\_input("End Date", value=datetime(2025, 7, 25))

lookback = st.sidebar.slider("Lookback Period for Percent Rank", min\_value=50, max\_value=200, value=100)

# Function to calculate RS metrics

def calculate\_rs\_and\_rank(df, lookback=100):

df['ThreeMthRS'] = 0.4 \* (df['Close'] / df['Close'].shift(63))

df['SixMthRS'] = 0.2 \* (df['Close'] / df['Close'].shift(126))

df['NineMthRS'] = 0.2 \* (df['Close'] / df['Close'].shift(189))

df['TwelveMthRS'] = 0.2 \* (df['Close'] / df['Close'].shift(250))

df['RSraw'] = df[['ThreeMthRS', 'SixMthRS', 'NineMthRS', 'TwelveMthRS']].sum(axis=1)

df['PercentRank'] = df['Close'].rolling(window=lookback).apply(

lambda x: pd.Series(x).rank(pct=True).iloc[-1] \* 100, raw=True

)

df['Color\_HSB'] = df['PercentRank'].apply(lambda x: (x \* 64 / 100, 255, 255))

return df

# Fetch data and process when the user clicks a button

if st.button("Run Analysis"):

try:

# Fetch data from Yahoo Finance

df = yf.download(ticker, start=start\_date, end=end\_date, progress=False)

if df.empty:

st.error("No data found for the ticker or date range. Please try again.")

else:

df = df[['Close', 'Volume']].copy()

df.reset\_index(inplace=True)

df.set\_index('Date', inplace=True)

# Calculate RS metrics

result = calculate\_rs\_and\_rank(df, lookback=lookback)

output = result[['Close', 'Volume', 'RSraw', 'PercentRank']]

# Display results

st.subheader(f"Analysis for {ticker}")

st.write("Last 5 rows of the data:")

st.dataframe(output.tail(), use\_container\_width=True)

# Plot closing prices

st.subheader("Closing Price Trend")

st.line\_chart(output['Close'])

# Provide Excel download option

st.subheader("Download Results")

excel\_buffer = pd.ExcelWriter(f"{ticker}\_RS\_Analysis.xlsx", engine='openpyxl')

output.to\_excel(excel\_buffer, sheet\_name=f'{ticker} Data', index=True)

excel\_buffer.close()

with open(f"{ticker}\_RS\_Analysis.xlsx", "rb") as f:

st.download\_button(

label="Download Excel File",

data=f,

file\_name=f"{ticker}\_RS\_Analysis.xlsx",

mime="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet"

)

except Exception as e:

st.error(f"An error occurred: {e}")

# Note about trading days

st.sidebar.write(f"Note: There are approximately 270 trading days between {start\_date} and {end\_date}.")