Data_Retrieval

April 10, 2023

1 Data Retrieval

This code file completes the data collection and storage process. Headlines and financial data are gathered from respective sources, processed into final forms and stored in the PostgreSQL database.

*The database used in this work is locally hosted, usage of this file outside of the environment in which it was created will be unsuccessful

[ ]:
```
!pip install yfinance
!pip install pyspark
!pip install findspark
!pip install dateparser
!pip install vaderSentiment
import pandas as pd
import csv
import datetime
import yfinance as yf
import numpy as np
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
from bs4 import BeautifulSoup as bs
import requests
from pyspark.sql.functions import sum,max,min,mean,count
import datetime as dt
import pyspark
from pyspark.sql import SparkSession
import pyspark.pandas as ps
import findspark
import yaml
from yaml.loader import SafeLoader
from os.path import abspath

warehouse_location = abspath('spark-warehouse')
with open('cfg.yml') as f:
    config = yaml.load(f, Loader = SafeLoader)

findspark.init()
spark = SparkSession.builder \
    .master(config['spark']['spark_master'])\
```

```
    .appName('gather')\
    .enableHiveSupport()\
    .config('spark.sql.warehouse.dir', warehouse_location)\
    .config(config['spark']['spark_jars'], config['spark']['spark_jars_path'])\
    .config('spark.cores.max', '2')\
    .config('spark.executor.cores', '2')\
    .getOrCreate()
spark.sparkContext.setLogLevel("WARN")
spark
```

[2]:
```
url = config['postgres']['url']
props = {
    'user': config['postgres']['user'],
    'password' : config['postgres']['user'],
    'url': url,
    'driver': config['postgres']['driver']

}
```

2 Retrieve Headlines for Sentiment Analysis

[4]:
```
#retrieve headlines from financial post
headers = {'User-Agent':
        'Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:108.0) Gecko/20100101␣
 ↪Firefox/108.0'}
def gather_headlines(company_name, ticker):
    headlines = []
    dates = []
    for i in range(10, 30000, 10):    # Running for-loop
        info_url = "https://financialpost.com/search/?
 ↪search_text="+company_name +"&date_range=-3650d&sort=asc&from="+str(i)
        page = requests.get(info_url, headers = headers)
        parser = bs(page.content, "html.parser" )
        date = parser.body.find_all('div', attrs={'class':␣
 ↪'article-card__meta-bottom'})
        for span in date:
            dates.append(span.text.split("   ")[1])
        headline = parser.body.find_all('h3', class_ = 'article-card__headline␣
 ↪text-size--extra-large--sm-up')
        for x in headline:
            headlines.append(x.text)
    dates = dates[:len(headlines)]
    file = {'date' : dates, "headline" : headlines}
    file = pd.DataFrame(file)
    print(file.head())
    file['ticker'] = ticker
```

```
    return file

#calculate sentiment scores for each headlines and append to dataset
def analyze_sent(df):
    analyze_obj = SentimentIntensityAnalyzer()
    df['sentiment']=df['headline'].apply(lambda headline: analyze_obj.
 ↪polarity_scores(str(headline))['compound'])
    df.fillna(0, inplace = True)
    return df

def final_sentiment(df):
    return df.withColumn("sent_score", df.mean_sentiment*(df.
 ↪headline_count**2)).drop('headline', 'headline_count', 'mean_sentiment')
```

3 Process Sentiment Scores and Write to Database

[5]:
```
import dateparser
ticker_list = ['MSFT','GOOG','NFLX','TSLA', 'AMZN']
company_list = ['microsoft', 'google', 'netflix', 'tesla', 'amazon']


def process_headlines(ticker_list, company_list):
    dfs = []
    for tick, company in zip(ticker_list, company_list):
        data = gather_headlines(company, tick)
        dfs.append(data)
    full_df = pd.concat(dfs)
    dates = []
    for index, row in full_df.iterrows():
        date = dateparser.parse(row['date'], date_formats = ["%d-%m-%y"])
        dates.append(date.date())
    full_df['date'] = dates
    full_df = ps.from_pandas(full_df)
    print(full_df.head())
    full_df = analyze_sent(full_df)
    full_df = full_df.to_spark()
    full_df.show()
    aggregated = full_df.groupBy('date', 'ticker').agg(count('headline').
 ↪alias('headline_count'), mean('sentiment').alias("mean_sentiment"))
    final_news = final_sentiment(aggregated)
    final_news.write.format("jdbc")\
        .option("url", "jdbc:postgresql://localhost:5432/financials") \
        .option("driver", "org.postgresql.Driver").option("dbtable",␣
 ↪"sentiment") \
        .option("user", "adam").option("password", "green").mode('append').
 ↪save()
```

```
process_headlines(ticker_list, company_list)
```

```
             date                                              headline
0  April 10, 2013   Personal computer shipments shrink 14% in wor…
1  April 11, 2013         What you need to know before markets open
2  April 11, 2013   Microsoft's Windows 8 gets blame for worst PC…
3  April 11, 2013   Microsoft falls after Goldman warns of PC los…
4  April 11, 2013   Electronic Arts lays off employees at Montrea…
            date                                              headline
0  April 5, 2013   4.05.13: 'Flurry of disappointment' for Canad…
1  April 5, 2013   Facebook Home Q&A with mobile engineering dir…
2  April 6, 2013   Discontinued products that we still miss dearly
3  April 8, 2013   How DIRTT has built a successful green manufa…
4  April 8, 2013   Review: The HTC One is the most beautiful And…
             date                                              headline
0  April 22, 2013   Google chairman Schmidt explores future of In…
1  April 22, 2013   Netflix shares surge as profit, U.S. streamin…
2  April 22, 2013   Closing Bell: TSX closes modestly higher amid…
3  April 23, 2013   4.23.13: Mark Carney, travelling light but ta…
4  April 23, 2013   Netflix becomes S&P's top performer as it stu…
            date                                              headline
0   July 9, 2013         What you need to know before markets open
1   July 9, 2013   Tesla rises to record high as stock heads for…
2  July 13, 2013   13 expensive side projects keeping the bigges…
3  July 16, 2013   Tesla CEO Elon Musk morphs from Tony Stark to…
4  July 18, 2013   GM is so afraid of Tesla it has assembled a s…
          date                                             headline
0  May 1, 2013         Transforce ready to ride out energy slump
1  May 6, 2013   How Al Gore amassed a $200-million fortune af…
2  May 7, 2013      The 40 most undervalued stocks in the market
3  May 7, 2013   Desire2Learn launches software that predicts …
4  May 9, 2013   Canadian TV producers DHX Media, Nelvana, OUT…
23/03/30 14:01:26 WARN TaskSetManager: Stage 0 contains a task of very large
size (1907 KiB). The maximum recommended task size is 1000 KiB.
```

```
         date                                                          headline
ticker
0  2013-04-10      Personal computer shipments shrink 14% in worst-ever decline
MSFT
1  2013-04-11                          What you need to know before markets open
MSFT
2  2013-04-11   Microsoft's Windows 8 gets blame for worst PC decline on record
MSFT
3  2013-04-11                  Microsoft falls after Goldman warns of PC losses
MSFT
```

```
4  2013-04-11            Electronic Arts lays off employees at Montreal studio
MSFT
23/03/30 14:01:29 WARN TaskSetManager: Stage 1 contains a task of very large
size (1907 KiB). The maximum recommended task size is 1000 KiB.

/home/cis6180/anaconda3/lib/python3.9/site-packages/pyspark/pandas/utils.py:975:
PandasAPIOnSparkAdviceWarning: If `index_col` is not specified for `to_spark`,
the existing index is lost when converting to Spark DataFrame.
  warnings.warn(message, PandasAPIOnSparkAdviceWarning)

23/03/30 14:01:29 WARN TaskSetManager: Stage 2 contains a task of very large
size (1907 KiB). The maximum recommended task size is 1000 KiB.


+----------+--------------------+------+---------+
|      date|            headline|ticker|sentiment|
+----------+--------------------+------+---------+
|2013-04-10| Personal compute…|  MSFT|      0.0|
|2013-04-11| What you need to…|  MSFT|      0.0|
|2013-04-11| Microsoft's Wind…|  MSFT|  -0.7579|
|2013-04-11| Microsoft falls …|  MSFT|  -0.4767|
|2013-04-11| Electronic Arts …|  MSFT|      0.0|
|2013-04-12| 4.12.13: BlackBe…|  MSFT|      0.0|
|2013-04-12| Motocross Madnes…|  MSFT|  -0.4939|
|2013-04-12| Who says account…|  MSFT|  -0.3182|
|2013-04-15| 4.15.13: Gold an…|  MSFT|      0.0|
|2013-04-15| Microsoft smartw…|  MSFT|      0.0|
|2013-04-16| Facebook, Apple …|  MSFT|      0.0|
|2013-04-16| Facebook Home se…|  MSFT|      0.0|
|2013-04-17| Buying defensive…|  MSFT|  -0.1531|
|2013-04-17| 4.17.13: Stickin…|  MSFT|      0.0|
|2013-04-17| TSX tumbles as g…|  MSFT|  -0.2023|
|2013-04-17| The bull case fo…|  MSFT|   0.5216|
|2013-04-17| Learning to use …|  MSFT|      0.0|
|2013-04-18| Private equity t…|  MSFT|      0.0|
|2013-04-18| Microsoft CFO Kl…|  MSFT|   0.4019|
|2013-04-19| What you need to…|  MSFT|      0.0|
+----------+--------------------+------+---------+
only showing top 20 rows

23/03/30 14:01:34 WARN TaskSetManager: Stage 3 contains a task of very large
size (1907 KiB). The maximum recommended task size is 1000 KiB.
```

4 Retrieve, Process and Store Financial Data

[3]:
```
def get_financials(ticker, start):
    time_delt = dt.timedelta(days = 150)
    start_day = start - time_delt
    data = yf.download(str(ticker), start_day)
    data['ticker'] = ticker
    data = data.reset_index()
    data = data.rename(columns = {'Date':'date', 'Open':'open', 'High':'high',␣
 ↪'Low':'low', 'Close':'close', 'Adj Close': 'adj_close', 'Volume':'volume'})
    print('success!')
    return data


def EWMA(data, ndays):
    EMA = pd.Series(data['close'].ewm(span = ndays, min_periods = ndays - 1).
 ↪mean(),
                    name = 'EWMA_' + str(ndays))
    data = data.join(EMA)
    return data

def rsi(close, periods = 14):

    close_delta = close.diff()

    # Make two series: one for lower closes and one for higher closes
    up = close_delta.clip(lower=0)
    down = -1 * close_delta.clip(upper=0)

    ma_up = up.ewm(com = periods - 1, adjust=True, min_periods = periods).mean()
    ma_down = down.ewm(com = periods - 1, adjust=True, min_periods = periods).
 ↪mean()

    rsi = ma_up / ma_down
    rsi = 100 - (100/(1 + rsi))
    return rsi

def BBANDS(data, window):
    MA = data.close.rolling(window).mean()
    SD = data.close.rolling(window).std()
    data['MiddleBand'] = MA
    data['UpperBand'] = MA + (2 * SD)
    data['LowerBand'] = MA - (2 * SD)
    return data

def prep_financials(df):
    df = pd.DataFrame(df)
```

```
    #df.set_index('date')
    df['target'] = (df['close'])
    df['tenmda'] = df['close'].rolling(10).mean()
    df['twentymda'] = df['close'].rolling(20).mean()
    df['fiftymda'] = df['close'].rolling(50).mean()
    df['hundredmda'] = df['close'].rolling(100).mean()
    df = EWMA(df, 20)
    df = EWMA(df, 50)
    df = EWMA(df, 100)
    df['rsi'] = rsi(df['close'])
    df = BBANDS(df, 40)
    df.dropna(inplace = True)
    df.reset_index()
    print(df.head())
    return df
```

[ ]:
```
def process_finance(ticker_list):
    finance_dfs = []
    for tick in ticker_list:
        data = get_financials(tick, dt.date(2015,1, 1))
        data = prep_financials(data)
        finance_dfs.append(data)
    final_finance = pd.concat(finance_dfs)
    final_finance = spark.createDataFrame(final_finance)
    final_finance.write.format("jdbc")\
        .option("url", "jdbc:postgresql://localhost:5432/financials") \
        .option("driver", "org.postgresql.Driver").option("dbtable",␣
 ↪"company_data") \
        .option("user", "adam").option("password", "green").mode('append').
 ↪save()

ticker_list = ['MSFT','GOOG','NFLX','AMZN', 'TSLA']
process_finance(ticker_list)
```

[ ]:
```
spark.stop()
```