Jeremy Zhou, jzhou83
Miles Lee, mlee276

Modifications

We created several new questions after phase A after having a better idea of our project and which questions would be better suited for our project. Additionally, after phase A we decided to enrich our project with three new databases: Bond.txt, OptionVolume.txt, and Company.txt. In particular, we manually created every entry of Company.txt by searching up information about each CEO. We also created a script to draw data for OptionVolume. This is described more in detail in process.txt

From phase D: Since a lot of our queries relied on weekly/monthly price change, we needed more than 15 data points in our "small" data files. Thus, we enlarged every small file except for Company-small.txt, per the following specifications.
* Used 2 months of data per ticker.
* For SectorETF-small and OptionVolume, we included almost all the tickers. This way we are able to test all the tickers for.
Unfortunately, the "small" datasets are not very small, but that is necessitated by us doing a project on stock data - it's next to impossible to derive anything meaningful from a few days of data, especially comparing monthly returns.

Process

1) Briefly document the process (which data came from which URL and in what format, what downloaded columns you needed to remove, which downloaded files contained duplicate values or anomalies and how you removed them, what splitting of files was needed, etc.)

We searched for data according to these categories:

- Each category correlates to a relation in our relation schema.
- Bond yield (relation: Bond)
    - Downloaded data from these websites, which correlate to each type of treasury bond we are interested in.
        - https://www.investing.com/rates-bonds/u.s.-5-year-bond-yield-historical-data
        - https://www.investing.com/rates-bonds/u.s.-6-month-bond-yield-historical-data
        - https://www.investing.com/rates-bonds/u.s.-10-year-bond-yield-historical-data
        - Dates: 4/05/2011 to 4/05/2021
    - Needed to sign up for a free account in order to download data for free.
    - We added a "ticker" column with the duration of the bond. Then, we combined all three bond data files using python.
    - Converted date format to YYYY-MM-DD using python.
        - relation_processing.ipynb located in dbase_setup folder.

- Stock option volume (relation: OptionVolume)
    - Downloaded data from this website -

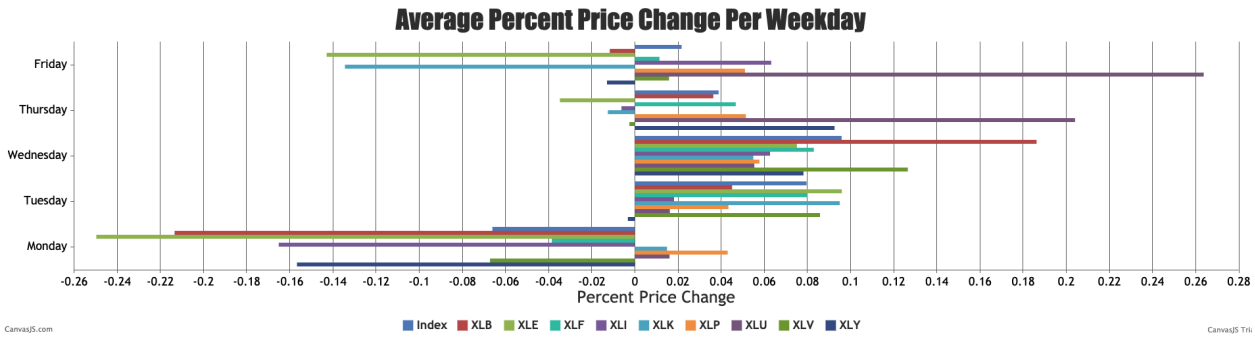
        - https://www.theocc.com/Market-Data/Market-Data-Reports/Volume-and-Open-Interest/Volume-Query
    - In order to download the daily stock data for free, we needed to program a python script to download option calls and put volume for specific stock tickers and dates via HTTP calls for free. This script is available in our submission.
        - get_option_volume.ipynb located in dbase_setup folder.
    - Converted date format to YYYY-MM-DD using python.
        - relation_processing.ipynb located in dbase_setup folder.
    - We shifted the OptionVolume dataset backwards by 6 years in order to align this data with the dates of our other datasets.

- Stock price and order volume (relation: SectorETF and MarketIndex)
    - Downloaded data from this website
        - https://www.kaggle.com/borismarjanovic/price-volume-data-for-all-us-stocks-etfs
    - Downloaded all the stock historical data files.
    - We removed the “open interest” column since all the values are zero.
    - For all sector ETF data files, we added a “ticker” column then appended them together using python.
        - Same thing was done for market index data files.
        - relation_processing.ipynb located in dbase_setup folder.

- Identifying ETFs (by sector) and Companies (relation: Company)
    - Searched online the ETF sectors in SPDR
    - Per sector, found the ETF’s top five holdings (companies)
    - Researched the ceo, ceo demographics, and market cap per company.

Successes

One particular query we were proud of was question 11. First, we created tables of the average price change and put/call ratio for each weekday, for both the given index and all sectors. This required executing another small query beforehand to get all the tickers, then looping through it in php. We created a multi-column bar chart of the average percent price change per weekday, to highlight average price change in particular and answer questions like “is Tuesday more green than Monday”. We pulled data from both SectorETF.txt, MarketIndex.txt, and OptionVolume.txt to create the data in the tables. The graph is an example run; it showed that in the time frame selected, Monday was the most red and Tuesday and Wednesday were the most green!



Another thing we were proud of was pulling the option volume data. Option data was difficult to find online, but we found that https://marketdata.theocc.com gave good data but only for specific dates at a time. However, they provided an API we could use to pull data, so we wrote a python script to pull option volume data using HTTP requests for all the tickers and dates we wanted and throw it into a csv file. The code is on the next page!

```
import pandas as pd
import requests
import urllib3
import datetime
urllib3.disable_warnings()

# Generate dates
base = datetime.datetime.today()
date_list = [base - datetime.timedelta(days=x) for x in range(2*365+5)]
dates = []
for dt in date_list:
    if dt.weekday() < 5:
        dates.append(dt.strftime("%Y%m%d"))

# Tickers
tickers = ['SPY','XLC','XLY','XLP','XLE','XLF','XLV','XLI','XLB','XLRE','XLK','XLU']

# CSV Columns
cols = ['Ticker','Date','CallVol','PutVol','P/C']
df = pd.DataFrame(columns=cols)

# Main loop
for ticker in tickers:
    for date in dates:
        a = [ticker, date]
        for PC in ['C','P']:
            r =
requests.get(f"https://marketdata.theocc.com/volume-query?reportDate={date}&format=csv
\

&volumeQueryType=O&symbolType=O&symbol={ticker}&reportType=D&accountType=ALL&productKi
nd=ALL\
            &porc={PC}", verify=False, allow_redirects=False)
            total = -1
            for line in r.iter_lines(chunk_size=1024):
                # Skip first line
                if total == -1:
                    total += 1
                    continue
                if line:
                    lineStr = str(line)
                    commaIndex = lineStr.find(',')
                    total += int(lineStr[2:commaIndex])
            a.append(total)
        if a[-2] == 0:
            continue
            a.append(0)
        else:
            a.append(a[-1]/a[-2])
        dfT = pd.DataFrame([a], columns=cols)
        df = df.append(dfT, ignore_index=True)
df.to_csv('OptionVolume.csv', index=False, header=False)
```

Known Issues
We fixed all the bugs from part D, so there are no known bugs in the public_html files.

From phase D: we noticed that we only had stock price data up until 2017, but option volume data only started in 2019. Unfortunately, there is no way to obtain the correct data online without paying a high fee. To fix this, we shifted the option volume data back by six years, so that the dates would align to still be on weekdays.

Extensions
If we had more time, we could've:

- Implemented queries that addressed some additional questions relating to volatility, such as the ones Alex suggested after phase D (e.g. finding correlations between # of puts and calls and realized volatility of the ETF).
- Drawn data from a much larger period. Our data was limited by what we could find, especially the option volume data. Having access to multiple decades of data for all the relations would have made our project data a lot more complete.
- Improved the user-friendliness of the HTML page. We could've added CSS and additional formatting to style the webpage and make it look better. Additionally, we would make the visualizations interactive, so the user can adjust their input and have the graph change instantly.