Replication and Extension of the Loughran and MacDonald analysis of 10Q Statements Report

Daoyu Li 93286958

By following the instructions of the assignment, I will list all the coding files related to each section.

Section 2.a

a. First, leverage the WRDS program to obtain ciks of S&P500 stocks. I download the names of those stocks and find the cik according to the “ticker.txt” provided by www.sec.gov.

By following the sample jupyter notebook codes provided by WRDS, I wrote the download_tickers_cik.ipynb to query the SP500 tickers and their corresponding years and quarters to sp500_tickers_2020_2024_by_quarter.csv. Then I merged it with ticker.txt to a csv file with ticker, quarters and ciks.

Section 2.b,c

b. Then, when generating MASTERINDEX objects, I check whether the cik is in S&P500 and whether the form is “10-Q”.
c. The original code has sleep time between downloading documents and restrains the time from 9pm to 6am. I modified those restraints to make it quicker.

I used the csv obtained from section 2a, and modified EDGAR_Pac.py & EDGAR_DownloadForms.py.
In order to download from sec.gov, I add headers as below

```
headers = {
    'User-Agent': 'dl5312@nyu.edu',
    'Accept-Encoding': 'gzip, deflate',
    'Host': 'www.sec.gov'
}
```

Since from class we know it’s necessary to get both 10-Q and 10-K for the full analysis, I modified the codes to download either 10-Q or 10-K for each company in their listed quarters in the csv.
Eventually I downloaded 9984 reports.

Section 3

a. The origin file only counts the number of negative words, but if we need to calculate the tf-idf weight, we need a weight for each word. I create a word_list to give each negative word in lm_dictionary a number.
b. Then I create 3 matrixes, for the 1st matrix(tf), the size is (#of documents * # of negative words in lm_dictionary), so it means the occurrence of each negative word in txt files.
   The 2nd matrix(idf), the size is also (# of documents * # of negative words in lm_dictionary), it means whether this word appeared in the txt files. The 3rd matrix(doc_length), the size is (#of documents *1). It counts the number of words in a file.
   This step is done by get_proportion and get_data functions.
c. Then calculate the tfidf and proportion weight using the 3 matrixes according to the formula provided by Loughran & McDonald (2011).
d. Finally, 1 X 2 matrixes, one is tfidf, one is term weight. They are all (#of documents *1) matrix and can be used during regression.

Specifically, these modified lines of code capture the idea of section 3

```
self.fin_neg_words = [word for word in self.lm_dict if self.lm_dict[word].negative]
tf_matrix = np.zeros((num_docs, len(shared.fin_neg_words)), dtype=np.float32)
doc_lengths = np.zeros(num_docs, dtype=np.uint32)
df = np.sum(tf_matrix > 0, axis=0) + 1e-6
idf = np.log(num_docs / df)
tf = tf_matrix / doc_lengths[:, np.newaxis]
tfidf = tf * idf
```

I used two dictionaries in the Generic_Parser.py

1. LoughranMcDonald_MasterDictionary_2014.csv(asked by the provided Generic parser)
2. Harvard_IV_Negative_Word_List_Inf.txt(found link in the first ever announcement of this course)

I am also using multiprocessing to speed up the processing time and finally get a final_results.csv containing all the attributes.

Section 4

1. The first step is to get the date of release for the 10-Q, it can be done by using the file name of the txt files.
2. can use CRSP to get the 4-day and 3-day excess returns. This is available from your WRDS account. Finally, sort the excess returns against tf_idf and term weight to get the necessary results and to replicate Figure 1 in Loughran & McDonald (2011). Alternatively, segregate the tf_idf by quintiles and plot those against their respective excess returns.

I got the date of release of 10-Q and 10-K from the file names, and then retrieve the excess returns from CRSP as below

Step 1: Choose your date range.

Date Variable:

2020-01-01 to 2024-12-31

Step 2: Apply your company codes.

What format are your company codes?

Autocomplete
ticker Permanent Security Number (permno)
Permanent Company Number (permco) ncusip siccd
CUSIP Header (cusip)
Standard Industrial Classification Code (hsiccd)

Select an option for entering your company codes:

Search Name or Ticker
Please enter company codes separated by a space.
Example: IBM MSFT AAPL
[Code Lookup: CRSP Stock (Annual)]

Code List Name
Save this code list to Saved Codes

---------Select Saved Codes List---------
Choose from your saved code lists.

Browse... sp500_tickers.txt
Upload a plain text file (.txt), having one code per line.

Search the entire database
This method allows you to search the entire database of records. Please be aware that this method can take a very long time to run because it is dependent upon the size of the database.

Step 3: Choose query variables.

How does this work?

Search All 5/61 | Identifying Information 3/20 | Time Series Information 1/10 | Share Information

Select All

cik
No variables available

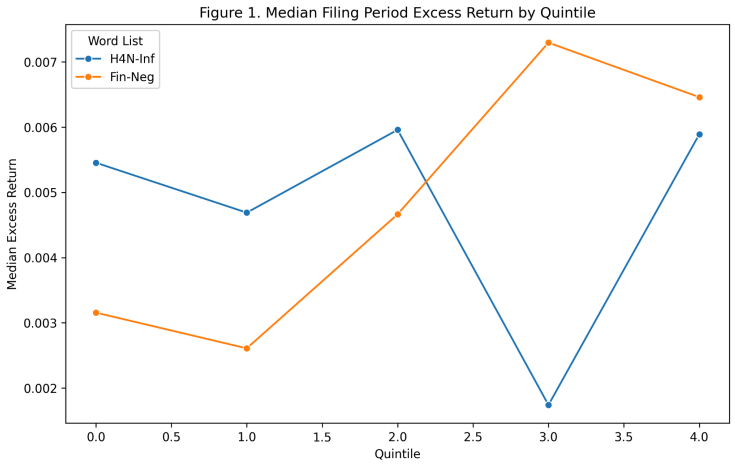
Selected Clear All (5)

- Cusip (cusip)
- Company Name (comnam)
- Ticker (ticker)
- Returns (ret)
- Return on S&P Composite Index (sprtrn)

Query 9751770, for crsp_a_stock

Status:	Success — Rerun
Product:	crsp_a_stock
Results:	Result Size: 7.5 MiB; Result Count: 737,306 Rows Output Files: Download .zip Output Results are also in your home directory at: **~/web_query_output/**. See WRDS Cloud: Access Web Query Output for more details.
Timing:	Elapsed Time: 29 seconds; Submitted: 2025-04-11 17:39 Work Begun: 2025-04-11 17:39; Work Finished: 2025-04-11 17:40
Input Parameters:	Toggle Input Parameters

Then I write the main.py based on the final_results.csv from section 3 and excess returns from CRSP to replicate the figure 1.



Surprisingly, both lines are having positive excess returns and don't actually perform like the original figure 1. I think the reason is due to Covid 19 and the market going well from 2022 to 2024 as the trend of AI. In this view, the Fin-Neg do have a better correlation to how the market moves and H4N-Inf didn't.