Return analysis

November 25, 2020

FRE 7871_I Fall 2020 HW 1 Part 2

- Jinfeng Hong jh6011
- Eric Sun zs861
- Raye Shen rs6981

[1]:
```python
import numpy as np
import pandas as pd
import datetime
import matplotlib.pyplot as plt
from pandas.tseries.offsets import BDay
```

0.1 Import score data from previous section

[19]:
```python
index = pd.read_csv("result.csv")
index.head()
```

[19]:

	keys	lm_score	hiv4_score	lm_pw	hiv4_pw
0	20100125_AAPL_10-KA_d10ka	0.600711	62.285790	0.034532	0.079106
1	20100208_BA_10k1_d10k	5.259690	279.376111	0.058846	0.087030
2	20100210_UNH_10k_d10k	2.955430	159.018505	0.042775	0.079799
3	20100212_HON_10k_c60039_10k	2.735993	169.977563	0.043494	0.083818
4	20100216_MMM_10k_a09-35783_110k	2.755357	433.202071	0.048741	0.082106

[20]:
```python
split = index["keys"].str.split("_", n = 3, expand = True)
index["ticker"] = split[1]
index["date"] = split[0]
index["keys"] = split[0]+"_"+split[1]
index["filling date"] = pd.to_datetime(index["date"], format='%Y%m%d')
index['end date'] = index["filling date"].apply(lambda x: x + BDay(3))
index["ticker"] = index["ticker"].replace('RTX', 'UTX')
index.head()
```

[20]:

	keys	lm_score	hiv4_score	lm_pw	hiv4_pw	ticker	date	\
0	20100125_AAPL	0.600711	62.285790	0.034532	0.079106	AAPL	20100125	
1	20100208_BA	5.259690	279.376111	0.058846	0.087030	BA	20100208	
2	20100210_UNH	2.955430	159.018505	0.042775	0.079799	UNH	20100210	
3	20100212_HON	2.735993	169.977563	0.043494	0.083818	HON	20100212	

```
4   20100216_MMM  2.755357  433.202071  0.048741  0.082106    MMM  20100216

  filling date   end date
0   2010-01-25 2010-01-28
1   2010-02-08 2010-02-11
2   2010-02-10 2010-02-15
3   2010-02-12 2010-02-17
4   2010-02-16 2010-02-19
```

0.2 Get data from WRDS database

[21]:
```
import wrds
db = wrds.Connection()
```

```
Enter your WRDS username [zhong]:ericsunnyu
Enter your password:········
WRDS recommends setting up a .pgpass file.
You can find more info here:
https://www.postgresql.org/docs/9.5/static/libpq-pgpass.html.
Loading library list…
Done
```

In CRSP, we cannot search by ticker as many tickers have duplicates. We fetch the permno number of the stock and merge it with our 10-K dataframe.

[22]:
```
company = db.get_table(library='crsp', table='stocknames')
```

After fetching the remove permno number list, we filter by DOW30 ticker and remove non related tickers manually.

[23]:
```
company = company.loc[company['ticker'].isin(index["ticker"])]
company = company.loc[company['comnam'] != "VIVENDI"]
company = company.loc[company['comnam'] != "VIVENDI UNIVERSAL"]
company = company.loc[company['comnam'] != "HONEYWELL INC"]
company = company.loc[company['comnam'] != "TRAVELERS INC"]
company = company.loc[company['comnam'] != "TRAVELERS GROUP INC"]
company = company.loc[company['comnam'] != "THOUSAND TRAILS INC DEL"]
company = company.loc[company['comnam'] != "MORGAN J P & CO INC"]
company = company.loc[company['comnam'] != "CLEVELAND ELECTRIC ILLUM CO"]
company = company.loc[company['comnam'] != "CHEVRONTEXACO CORP"]
company = company.loc[company['comnam'] != "MINING CORP CDA"]
company = company.loc[company['comnam'] != "CRAMER ELECTRONICS INC"]
company = company.loc[company['comnam'] != "GILLETTE CO"]
company = company.loc[company['comnam'] != "C R I INSURED MORTGAGE INVS"]
company = company.loc[company['comnam'] != "CORIMON C A S A C A"]
company = company.loc[company['comnam'] != "C R I INSURED MORTGAGE INVS LP"]
company = company.loc[company['comnam'] != "CORIMON S A C A"]
company = company.loc[company['comnam'] != "CORIMON C A"]
```

```
company = company.loc[company['comnam'] != "HUDSON BAY MNG & SMLT LTD"]
company = company.loc[company['comnam'] != "COMPARATOR SYSTEMS CORP"]
company = company.loc[company['comnam'] != "NEW YORK NEW HAVEN & HARTFORD R"]
company = company.loc[company['comnam'] != "IRVING BANK CORP"]
company = company.loc[company['comnam'] != "VIKING GENERAL CORP"]
company = company.loc[company['comnam'] != "VIVRA INC"]
company = company.loc[company['comnam'] != "DUPONT DE NEMOURS INC"]
company = company.loc[company['comnam'] != "AMERICAN TELEPHONE & TELEG CO"]
company = company.loc[company['comnam'] != "A T & T CORP"]
company = company.loc[company['comnam'] != "ALCOA CORP"]
company = company.loc[company['comnam'] != "ALUMINUM COMPANY AMER"]
company = company.loc[company['comnam'] != "BACHE & CO INC"]
company = company.loc[company['comnam'] != "BACHE GROUP INC"]
company = company.loc[company['comnam'] != "BANKAMERICA CORP"]
company = company.loc[company['comnam'] != "UNION TANK CAR CO"]
company = company.groupby(['ticker']).mean().astype(int)
```

[24]:
```
result = pd.merge(index,
                  company['permno'],
                  left_on='ticker',
                  right_on=company.index,
                  how='left')
result.head()
```

[24]:
```
              keys  lm_score  hiv4_score     lm_pw   hiv4_pw ticker      date  \
0  20100125_AAPL  0.600711   62.285790  0.034532  0.079106   AAPL  20100125
1    20100208_BA  5.259690  279.376111  0.058846  0.087030     BA  20100208
2   20100210_UNH  2.955430  159.018505  0.042775  0.079799    UNH  20100210
3   20100212_HON  2.735993  169.977563  0.043494  0.083818    HON  20100212
4   20100216_MMM  2.755357  433.202071  0.048741  0.082106    MMM  20100216

  filling date   end date  permno
0   2010-01-25 2010-01-28   14593
1   2010-02-08 2010-02-11   19561
2   2010-02-10 2010-02-15   92655
3   2010-02-12 2010-02-17   10145
4   2010-02-16 2010-02-19   22592
```

[25]:
```
crsp = pd.read_csv('CRSP value index.csv')
```

0.3 Pull stock price from CRSP

[26]:
```
stock_ret = []
for i in range(result.shape[0]):
    sql = "select permno, date, prc, ret from crsp.dsf where permno = " + \
        str(result['permno'][i]) + " and date>='" + str(result['filling␣
 ↪date'][i])[:10] +\
```

```
        "' and date<='" + str(result['end date'][i])[:10] + "'"
    df = db.raw_sql(sql, date_cols=['date'])
    try: ret = (df.iloc[-1,2] - df.iloc[0,2])/df.iloc[0,2]
    except: ret = 0
    stock_ret.append(round(ret,5))
```

[27]:
```
mkt_ret = []
for i in range(result.shape[0]):
    n = crsp[crsp['date'].astype(str)==result['date'][i]].index
    n = n[0]
    ret = (crsp.iloc[n+3,1] - crsp.iloc[n,1])/crsp.iloc[n,1]
    mkt_ret.append(round(ret,5))
```

[28]:
```
result['ret'] = stock_ret
result['mkt_ret'] = mkt_ret
result['excess_ret'] = result['ret'] - result['mkt_ret']
result.head()
```

[28]:
```
            keys  lm_score   hiv4_score     lm_pw   hiv4_pw ticker      date  \
0  20100125_AAPL  0.600711    62.285790  0.034532  0.079106   AAPL  20100125
1    20100208_BA  5.259690   279.376111  0.058846  0.087030     BA  20100208
2   20100210_UNH  2.955430   159.018505  0.042775  0.079799    UNH  20100210
3   20100212_HON  2.735993   169.977563  0.043494  0.083818    HON  20100212
4   20100216_MMM  2.755357   433.202071  0.048741  0.082106    MMM  20100216

  filling date   end date  permno      ret  mkt_ret  excess_ret
0   2010-01-25 2010-01-28   14593 -0.01864 -0.01105    -0.00759
1   2010-02-08 2010-02-11   19561  0.04664  0.02467     0.02197
2   2010-02-10 2010-02-15   92655 -0.01620  0.02915    -0.04535
3   2010-02-12 2010-02-17   10145  0.03249  0.02980     0.00269
4   2010-02-16 2010-02-19   22592  0.01305  0.01384    -0.00079
```

[29]:
```
result['lm_rank'] = result['lm_score'].rank(method='first')
result['hiv4_rank'] = result['hiv4_score'].rank(method='first')
result['lm_pw_rank'] = result['lm_pw'].rank(method='first')
result['hiv4_pw_rank'] = result['hiv4_pw'].rank(method='first')

bin_labels = ['5', '4', '3', '2', '1']
result['lm_quantile'] = pd.qcut(result['lm_rank'].astype(int), 5,␣
 ↪labels=bin_labels)
result['hiv4_quantile'] = pd.qcut(result['hiv4_rank'].astype(int), 5,␣
 ↪labels=bin_labels)
result['lm_pw_quantile'] = pd.qcut(result['lm_pw_rank'].astype(int), 5,␣
 ↪labels=bin_labels)
result['hiv4_pw_quantile'] = pd.qcut(result['hiv4_pw_rank'].astype(int), 5,␣
 ↪labels=bin_labels)
```

[30]:
```python
quantile = result[['keys', 'excess_ret', 'lm_quantile', 'hiv4_quantile',␣
↪'lm_pw_quantile', 'hiv4_pw_quantile']]
quantile.head()
```

[30]:
```
            keys  excess_ret  lm_quantile  hiv4_quantile  lm_pw_quantile  \
0  20100125_AAPL    -0.00759            5              5               5
1    20100208_BA     0.02197            2              3               1
2   20100210_UNH    -0.04535            3              4               4
3   20100212_HON     0.00269            4              4               4
4   20100216_MMM    -0.00079            4              2               2

   hiv4_pw_quantile
0                 4
1                 2
2                 3
3                 3
4                 3
```

[31]:
```python
lm_quantile = quantile.groupby(['lm_quantile']).median()
lm_quantile
```

[31]:
```
             excess_ret
lm_quantile
5             -0.002300
4              0.000625
3             -0.001510
2              0.002375
1              0.002965
```

[32]:
```python
hiv4_quantile = quantile.groupby(['hiv4_quantile']).median()
hiv4_quantile
```

[32]:
```
               excess_ret
hiv4_quantile
5               -0.004705
4                0.002505
3               -0.001510
2               -0.001275
1                0.004420
```

[33]:
```python
lm_pw_quantile = quantile.groupby(['lm_pw_quantile']).median()
lm_pw_quantile
```

[33]:
```
                excess_ret
lm_pw_quantile
5                 -0.00208
4                  0.00039
```

```
3                       0.00104
2                       0.00008
1                      -0.00128
```

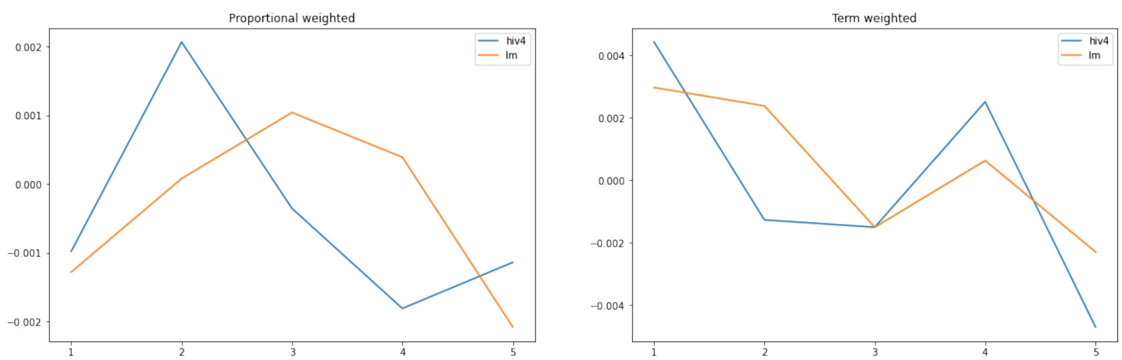
[34]:
```
hiv4_pw_quantile = quantile.groupby(['hiv4_pw_quantile']).median()
hiv4_pw_quantile
```

[34]:
```
                  excess_ret
hiv4_pw_quantile
5                  -0.001140
4                  -0.001810
3                  -0.000355
2                   0.002065
1                  -0.000980
```

[35]:
```
fig, (ax1, ax2) = plt.subplots(1,2, figsize=(20, 6))
ax1.plot(hiv4_pw_quantile, label='hiv4')
ax1.plot(lm_pw_quantile, label='lm')
ax1.invert_xaxis()
ax1.legend()
ax1.set_title("Proportional weighted")

ax2.plot(hiv4_quantile, label='hiv4')
ax2.plot(lm_quantile, label='lm')
ax2.invert_xaxis()
ax2.legend()
ax2.set_title("Term weighted")

plt.show()
```



In the plots above, 1 is the least negative, 5 is the most negative. If the model is perfect, the plot should show downward sloping lines.

[ ]:
```
db.close()
```

0.4 Observation

In the proportional weighted plot, both lines did not show a generally decreasing trend, which means under proportional weighting, both the Harvard and LM-negative dictionaries did not manage to produce a good result.

However, in the term weighted plot, both lines show a generally decreasing trend. There is one exception at quantile 4 for both lines, but the general downwards trend is clear. This indicates that the term weighting methodology presented by Loughran and McDonald in the original paper is really useful. Out of all four lines, the LM-term weighted line shows the best result.

0.5 Analysis

The reason that proportional weighting did not generate a good result is because the model did not account for different length of the 10-K reports, which means there is no normalization. This will cause some entries to be improperly weighted more than others. At the same time, propotional weighting also did ont account for the importance of a term, both within a document and the entire corpus, which makes the analysis more inaccurate.

On the other hand, term weighting address the above mentioned problems by reassigning a weight to each word, and calculate the document score by multiplying the word count within the document and the word weightings, followed by summing them up. This generates a better result as shown in the term weighing plot. Out of the two trends in the term-weighted plot, LM dictionary shows a better result by generating a trend that better fits a strictly downward sloping line. This is because the words in the LM dictionary were carefully chosen to fit the specific context of the financial reports. Due to polysemes, some words might have different meanings in different context. Therefore, using a seanario-specific dictionary is always a better choice thana general dictionary.

Our result is not as good as the plot in the original paper. We have two hypothesis. 1. The sample size is small. The original paper used ~50000 financial statements, while we only have ~350. This might cause the result to be biased. 2. We used data from 2010-2019, mostly after the original paper has been published. Companies might be aware of the exsistence of this LM dictionary. Therefore, they will try to use more neutral words in the financial statements, causing the result to be not as good as before.