NotesS4

Andrew Wang

2024-01-18

Summary

This document outlines my methodology and implementation for calculating the HHI index in alignment with step 4 of the notes document.

First, we conduct basic filtering and feature engineering to extract a quarter and year pair for grouping

```r
#Basic filtering conducted on the read-in data
df <- df[df$trade_cost > 0, ]
df$quarter <- quarters(df$report_date)
df$year <- year(df$report_date)
df$Time <- as.yearqtr(paste(df$year, df$quarter))
```

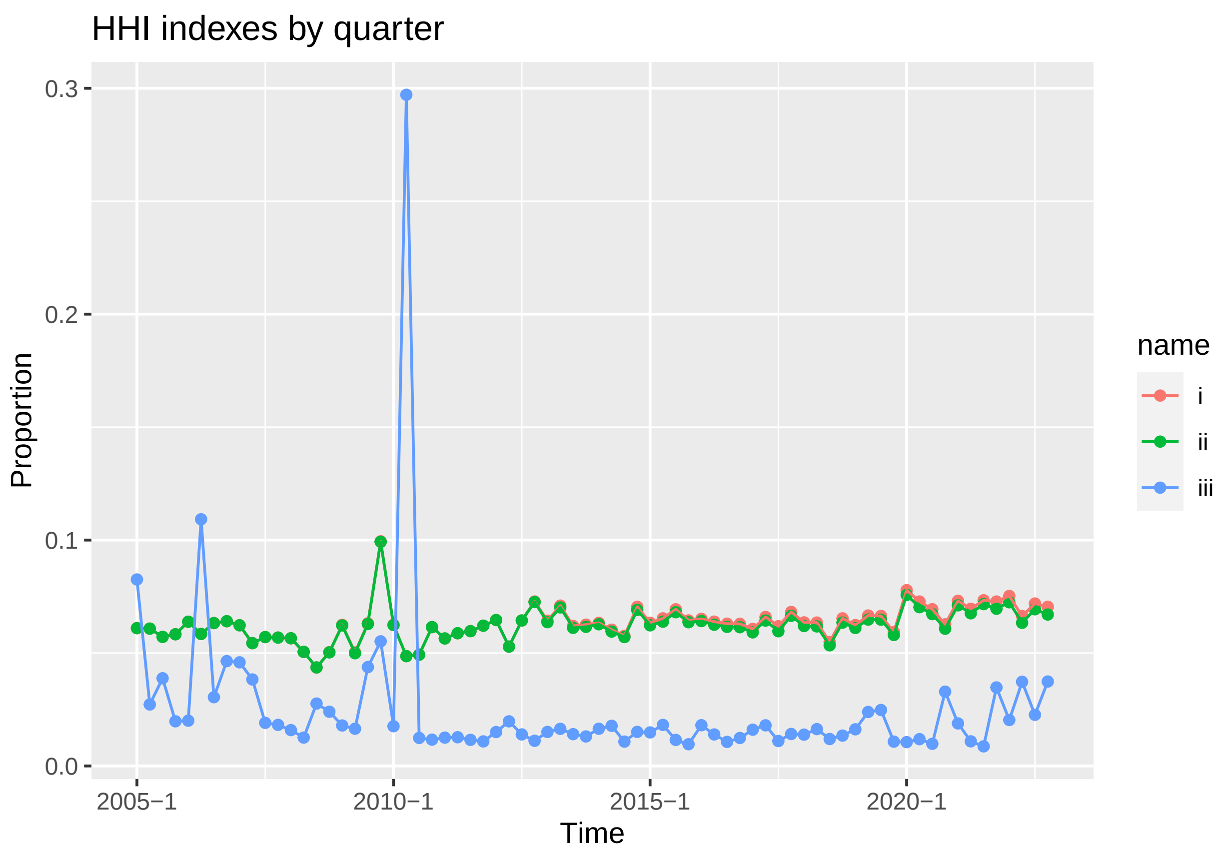
I implemented the following function to take in a filtered dataframe and return a table consisting of quarters and years in one column and the associated HHI index value for the time period in the other column. We want to calculate volume by broker, so we group by two variables and sum the transaction amounts. We then want some macro data about the set, in which we calculate N and total volume. We merge this back into the dataset to provide a basis for calculating s, which is the squared proportion of broker volume over total volume. We then group by time again and calculate the indexes for each period, before returning the output.

```r
makeHHI <- function(df) {
  df$trns_amount <- abs(df$trns_amount)
  hhi1 <- df %>% group_by(Time, brok_id) %>% transmute(
    vol = sum(trns_amount)
  ) %>% as.data.frame() %>% unique()

  hhi1Meta <- hhi1 %>% group_by(Time) %>% transmute(
    num = n(),
    quartVol = sum(vol)
  ) %>% as.data.frame() %>% unique()
  hhi1 <- merge(hhi1, hhi1Meta, all.x = TRUE)
  hhi1$s <- (hhi1$vol / hhi1$quartVol)^2
  hhi1 <- hhi1 %>% group_by(Time) %>% transmute(
    index = (sum(s) - (1/num)) / (1 - 1/num)
  ) %>% as.data.frame() %>% unique()
  return(hhi1)
}
```

We apply this function to both the (i) and (ii) calculations, while we use an altered version for demand side concentration for (iii)

```r
#4.1 among dealers only - grouping brok_id
subset <- df %>% filter(elec_platf == 0) %>% filter(ATS == 0)
hhi1 <- makeHHI(subset)

#4.2 among dealers and ep/ats
hhi2 <- makeHHI(df)

#4.3 among investors
hhi3 <- df %>% group_by(Time, fund_id) %>% transmute(
  vol = sum(trns_amount)
) %>% as.data.frame() %>% unique()

hhi3Meta <- hhi3 %>% group_by(Time) %>% transmute(
  num = n(),
  quartVol = sum(vol)
) %>% as.data.frame() %>% unique()
hhi3 <- merge(hhi3, hhi3Meta, all.x = TRUE)
hhi3$s <- (hhi3$vol / hhi3$quartVol)^2
hhi3 <- hhi3 %>% group_by(Time) %>% transmute(
  index = (sum(s) - (1/num)) / (1 - 1/num)
) %>% as.data.frame() %>% unique()
```

```r
hhi1$i <- hhi1$index
hhi1$index <- NULL
hhi2$ii <- hhi2$index
hhi2$index <- NULL
hhi3$iii <- hhi3$index
hhi3$index <- NULL
out <- merge(hhi1, hhi2)
out <- merge(out, hhi3)

out2 <- out %>%
  pivot_longer(c(i, ii, iii))

ggplot(out2, aes(Time, value, color = name)) + geom_point() + geom_line() + scale_x_yearqtr() + ggtitle
```




```r
print(out)
```

```
##       Time          i         ii         iii
## 1  2005 Q1 0.06100646 0.06099568 0.082550561
## 2  2005 Q2 0.06079480 0.06079724 0.027212074
## 3  2005 Q3 0.05713667 0.05713266 0.038811869
## 4  2005 Q4 0.05830849 0.05828746 0.019756326
## 5  2006 Q1 0.06384305 0.06384305 0.020041850
## 6  2006 Q2 0.05841936 0.05843424 0.109203623
## 7  2006 Q3 0.06324045 0.06325687 0.030429728
## 8  2006 Q4 0.06406929 0.06408010 0.046400623
## 9  2007 Q1 0.06226563 0.06226563 0.045870623
## 10 2007 Q2 0.05436512 0.05436512 0.038288194
## 11 2007 Q3 0.05703977 0.05703977 0.019074704
## 12 2007 Q4 0.05683622 0.05683622 0.018174949
## 13 2008 Q1 0.05651062 0.05651964 0.015870976
## 14 2008 Q2 0.05048052 0.05049064 0.012552364
## 15 2008 Q3 0.04359568 0.04360899 0.027620451
## 16 2008 Q4 0.05037226 0.05028006 0.023999590
## 17 2009 Q1 0.06247528 0.06221492 0.017912094
## 18 2009 Q2 0.05026096 0.04989505 0.016478721
## 19 2009 Q3 0.06312754 0.06286407 0.043763424
## 20 2009 Q4 0.09944688 0.09922154 0.055158263
## 21 2010 Q1 0.06250264 0.06236090 0.017582813
## 22 2010 Q2 0.04861814 0.04861228 0.297120375
```

```
## 23 2010 Q3 0.04926725 0.04924668 0.012391493
## 24 2010 Q4 0.06145354 0.06146564 0.011643009
## 25 2011 Q1 0.05643098 0.05644581 0.012538846
## 26 2011 Q2 0.05875749 0.05875305 0.012711368
## 27 2011 Q3 0.05968230 0.05966954 0.011552272
## 28 2011 Q4 0.06207812 0.06208968 0.010871297
## 29 2012 Q1 0.06459151 0.06458656 0.015010831
## 30 2012 Q2 0.05281693 0.05283556 0.019753941
## 31 2012 Q3 0.06439752 0.06439787 0.013959819
## 32 2012 Q4 0.07281009 0.07254465 0.011134424
## 33 2013 Q1 0.06431141 0.06365631 0.015050872
## 34 2013 Q2 0.07097652 0.07018651 0.016426898
## 35 2013 Q3 0.06198844 0.06112407 0.014074980
## 36 2013 Q4 0.06255735 0.06157087 0.013063244
## 37 2014 Q1 0.06333984 0.06271981 0.016469317
## 38 2014 Q2 0.06031935 0.05947375 0.017764255
## 39 2014 Q3 0.05761465 0.05705340 0.010777217
## 40 2014 Q4 0.07045950 0.06909544 0.015076809
## 41 2015 Q1 0.06339322 0.06225051 0.014847022
## 42 2015 Q2 0.06534179 0.06387301 0.018152522
## 43 2015 Q3 0.06934732 0.06805241 0.011520083
## 44 2015 Q4 0.06443349 0.06361425 0.009618351
## 45 2016 Q1 0.06511366 0.06412529 0.018023413
## 46 2016 Q2 0.06388550 0.06251963 0.013943721
## 47 2016 Q3 0.06298834 0.06157412 0.010657234
## 48 2016 Q4 0.06291395 0.06140780 0.012372915
## 49 2017 Q1 0.06064708 0.05913713 0.016047958
## 50 2017 Q2 0.06596840 0.06436925 0.017980042
## 51 2017 Q3 0.06184907 0.05963279 0.011028592
## 52 2017 Q4 0.06817279 0.06647872 0.014147599
## 53 2018 Q1 0.06355581 0.06193582 0.013867981
## 54 2018 Q2 0.06347607 0.06177796 0.016300859
## 55 2018 Q3 0.05484384 0.05340884 0.011917642
## 56 2018 Q4 0.06534595 0.06344890 0.013437826
## 57 2019 Q1 0.06235525 0.06107937 0.016220042
## 58 2019 Q2 0.06660002 0.06479196 0.023895004
## 59 2019 Q3 0.06640201 0.06477041 0.024766473
## 60 2019 Q4 0.05928388 0.05797891 0.010762408
## 61 2020 Q1 0.07782161 0.07572148 0.010546081
## 62 2020 Q2 0.07278957 0.07027609 0.011863244
## 63 2020 Q3 0.06938735 0.06721419 0.009802120
## 64 2020 Q4 0.06269687 0.06075347 0.032896858
## 65 2021 Q1 0.07310441 0.07111123 0.018834994
## 66 2021 Q2 0.06957556 0.06757080 0.010890720
## 67 2021 Q3 0.07334866 0.07162548 0.008671694
## 68 2021 Q4 0.07279101 0.06958374 0.034756348
## 69 2022 Q1 0.07525360 0.07245692 0.020335002
## 70 2022 Q2 0.06635582 0.06337129 0.037256322
## 71 2022 Q3 0.07192331 0.06935871 0.022641085
## 72 2022 Q4 0.07045530 0.06703308 0.037374559
```

At the end we perform some basic merging functions to get the aggregate dataset.