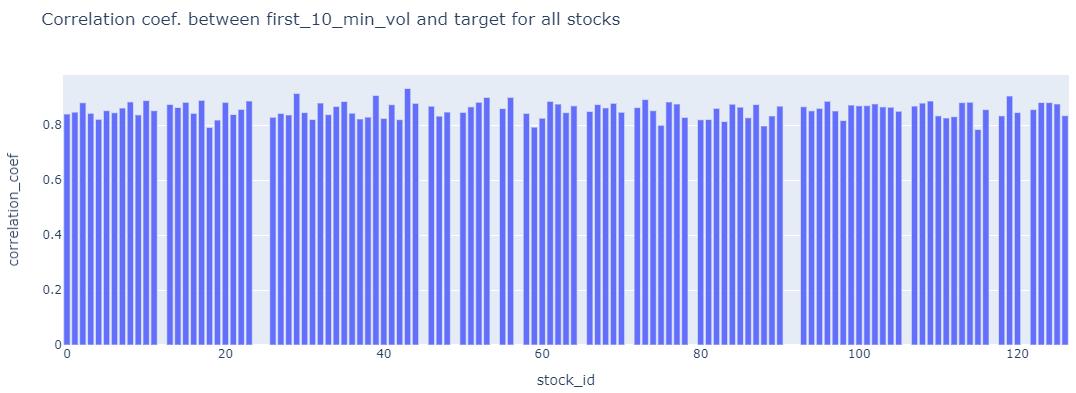
**IDEAS to gain insights**

# **Feature DEFINITION**

##### 

##### **1) Check correlation between target and book WAP realized\_vol in first 10 mins. Aggregate book WAP price using the log returns and realised volatility formulas for each time id. Plot scatterplot this against the corresponding target for all time id.**

yes confirmed!



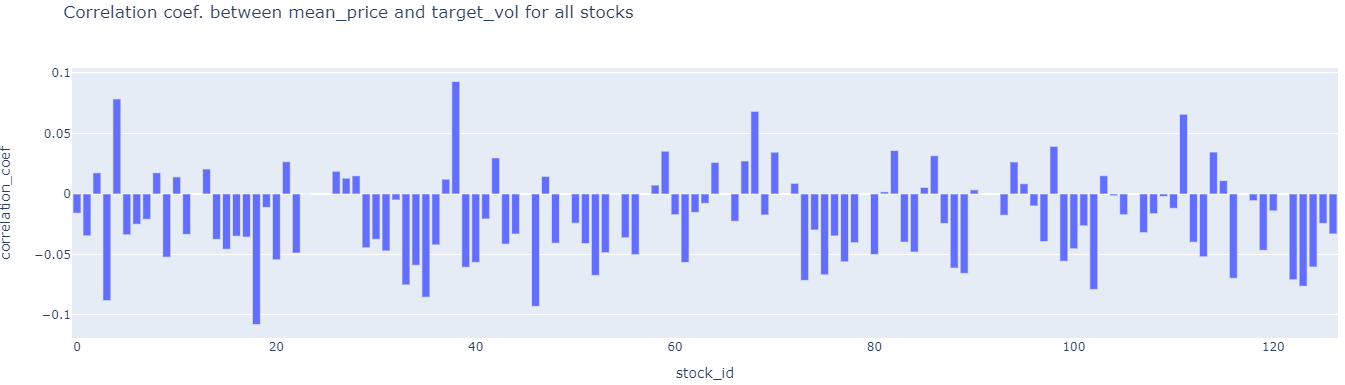
##### **2) Check correlation between**

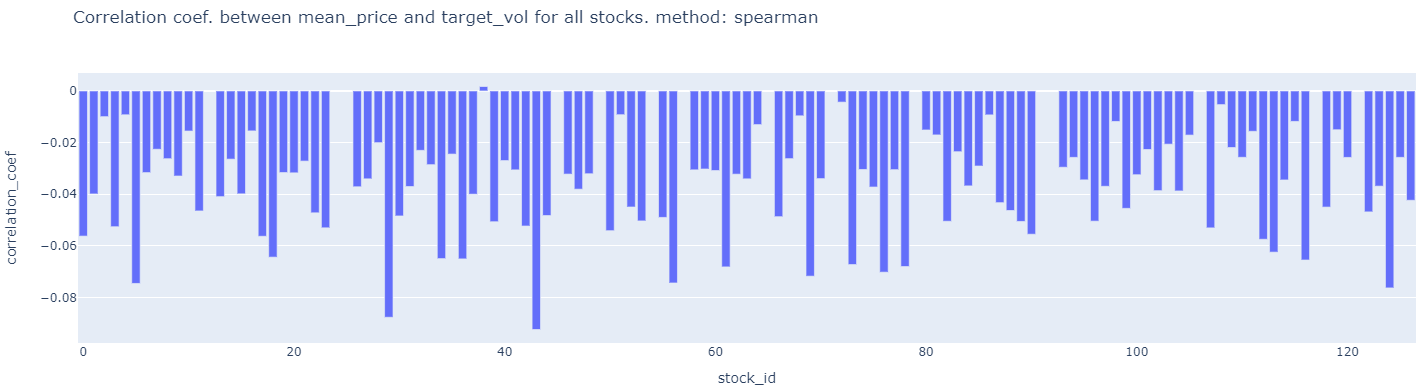
##### **i) target (2nd 10 mins vol.) and trade execution stock price at the available times.**

Aggregate **trade execution price using the log returns and realised volatility formulas for each time id. Plot scatterplot this against the corresponding target for all time id.**

**Yes confirmed**

****

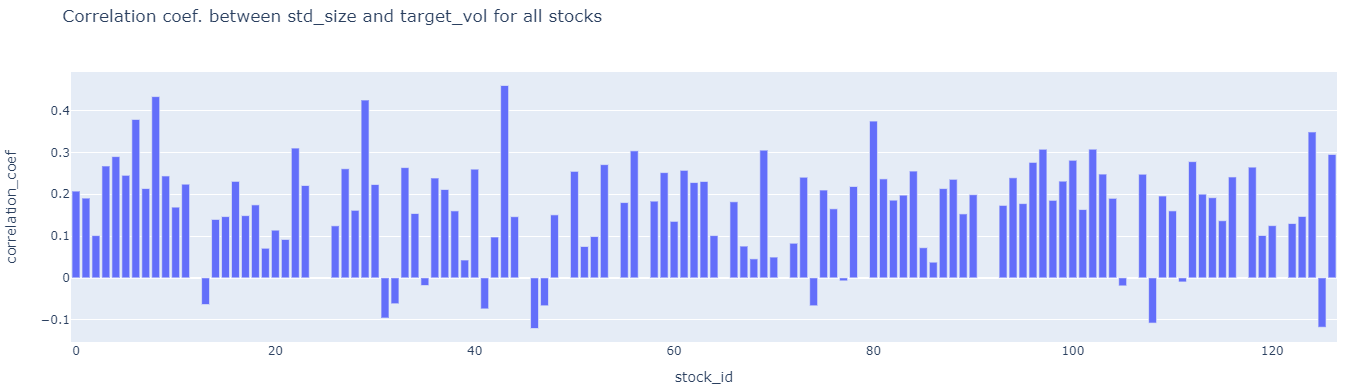
****

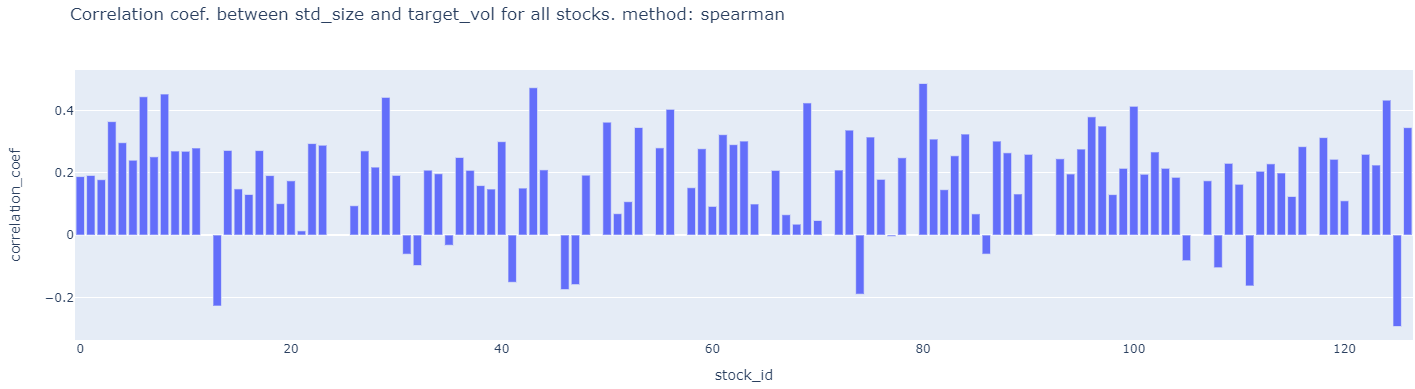
****

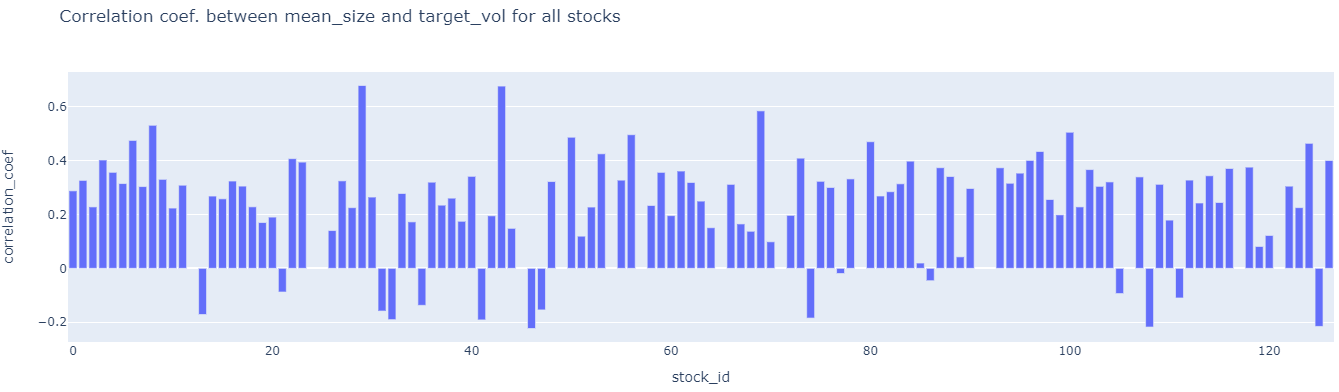
****

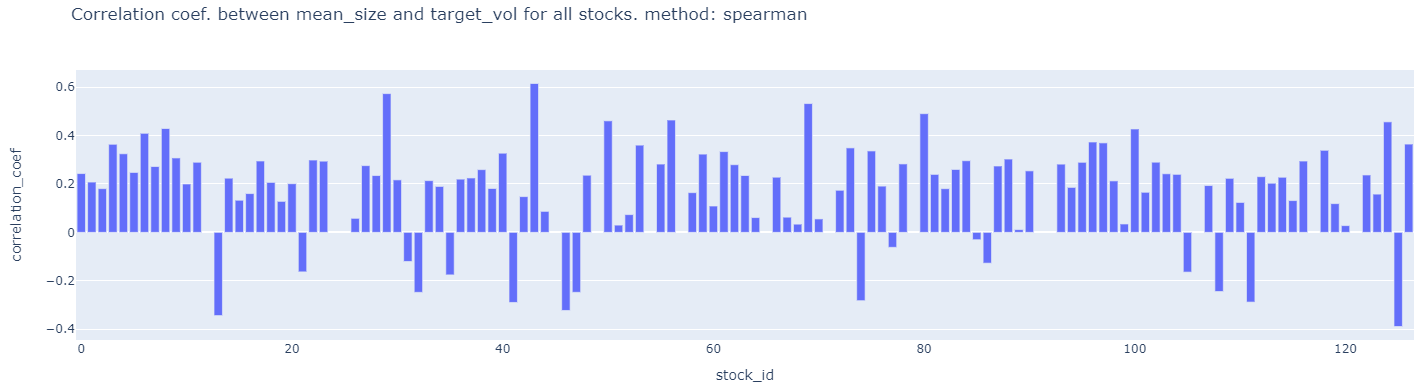
##### **ii) target (2nd 10 mins vol.) and size at the available times.**

Aggregate size by taking standard deviation or mean at each time id. **Plot scatterplot of this against the corresponding target for each time id.**

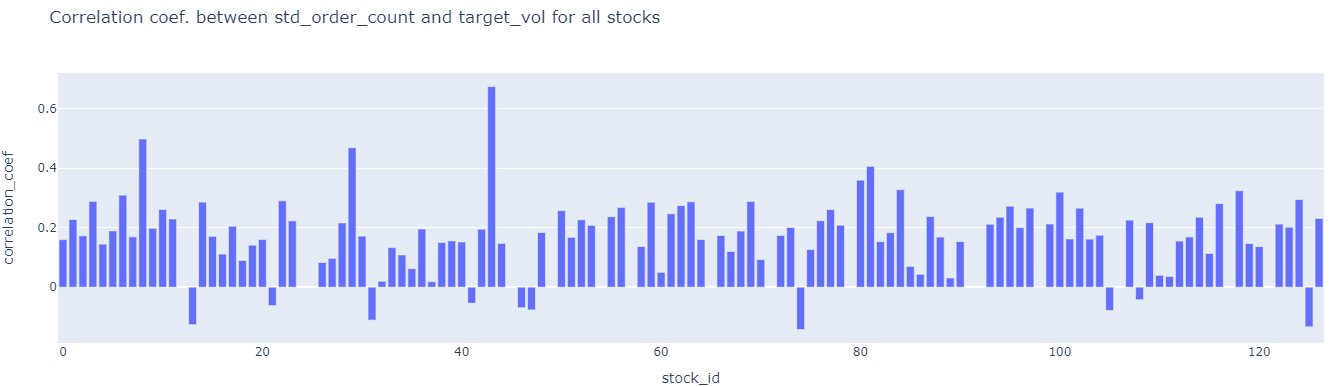
****

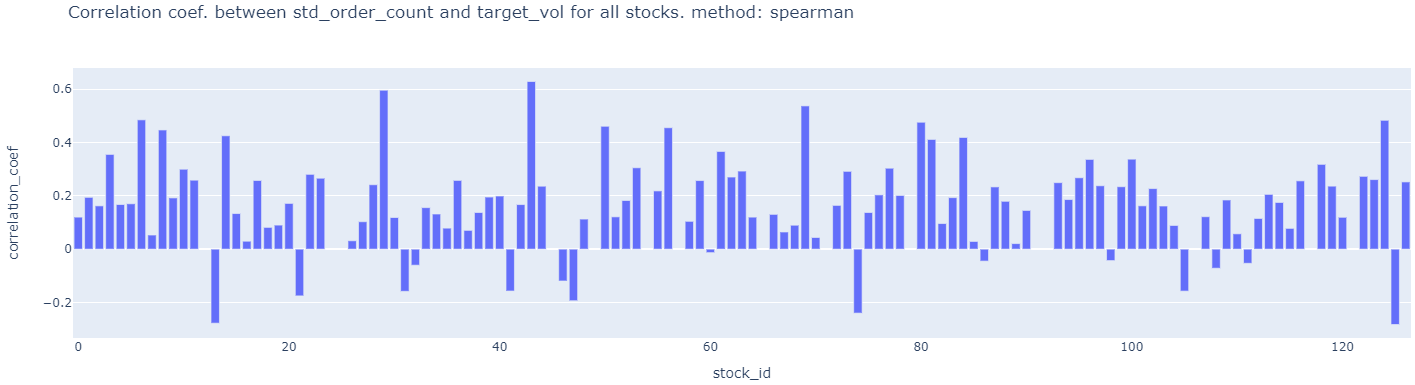
****

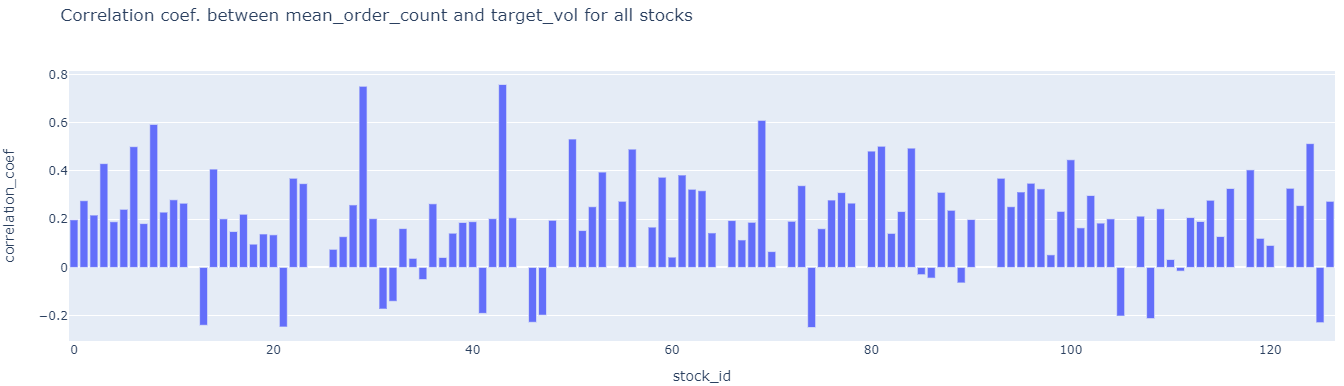
****

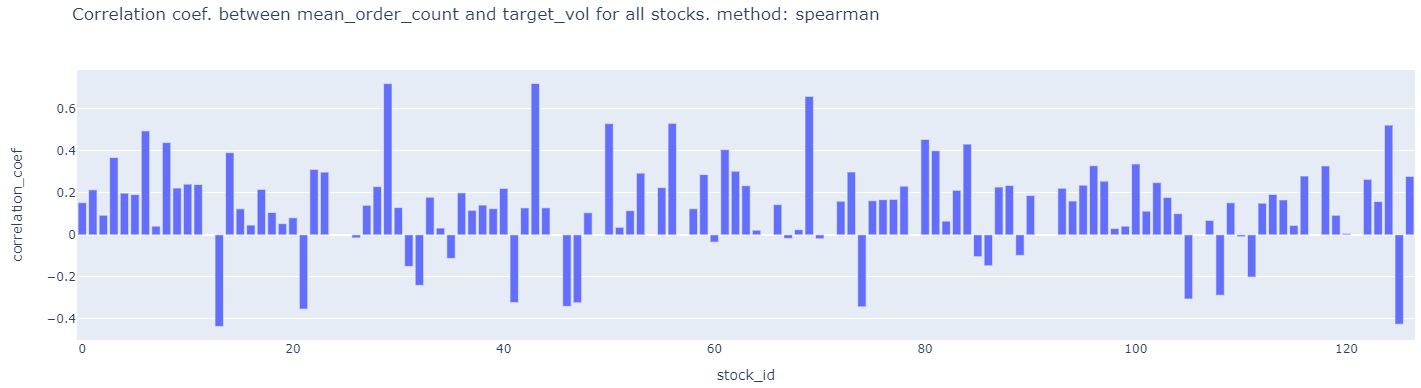
****

##### **ii) target (2nd 10 mins vol.) and order\_count at the available times.** Aggregate **order\_count** by taking standard deviation or mean at each time id. **Plot scatterplot of this against the corresponding target for each time id.**







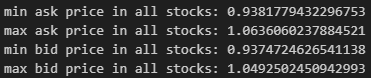


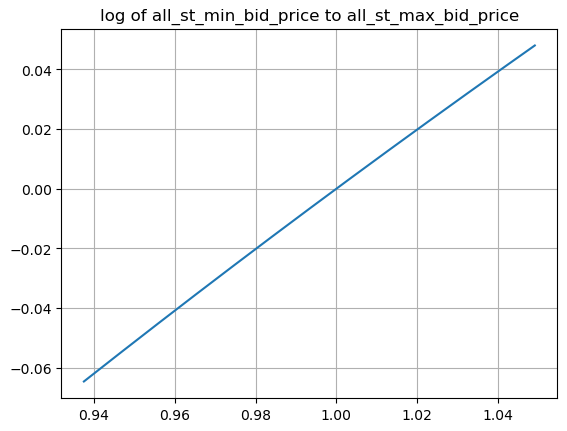
##### **2a)** Try to calculate spearman’s rank correlation for the cases of low linear pearson’s correlation. It might be higher due to non-linear correlation??? **DONE ABOVE. VERY SIMILAR.**

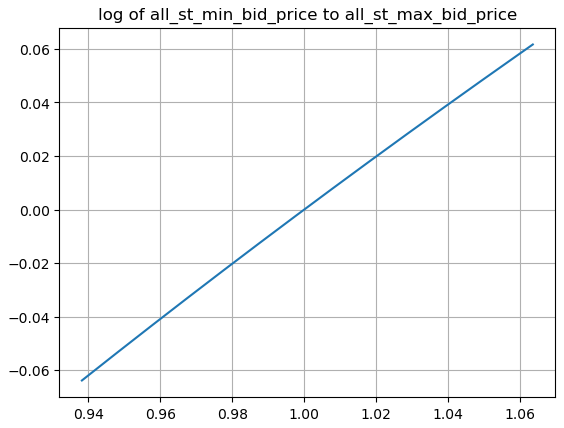
##### **3) check correlation in the first 10 mins. Target volatility between any two stocks and then identify which stocks have highest correlation. This is possible because even though time\_id is shuffled, they are the same for each stock so the scatter plot does not care about time. 112 choose 2 = 6216. compute this numerically. DO NOT PLOT. Rank the correlations.**

**DONE and updated in Key Insights file.**

4) ## check how prices are distributed in the book data so that we can see how log(s\_t2/s\_t1) transformation of prices below and above 0 affects volatility. ## due to log nature check if non-linearity is visible. A point in graph below is for a single stock.



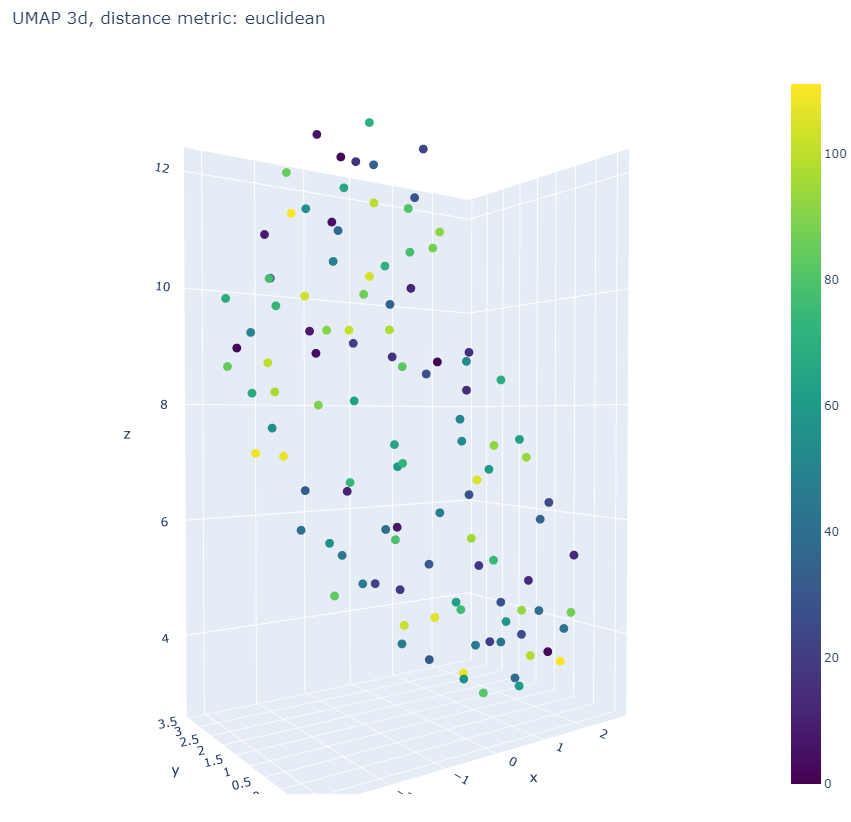




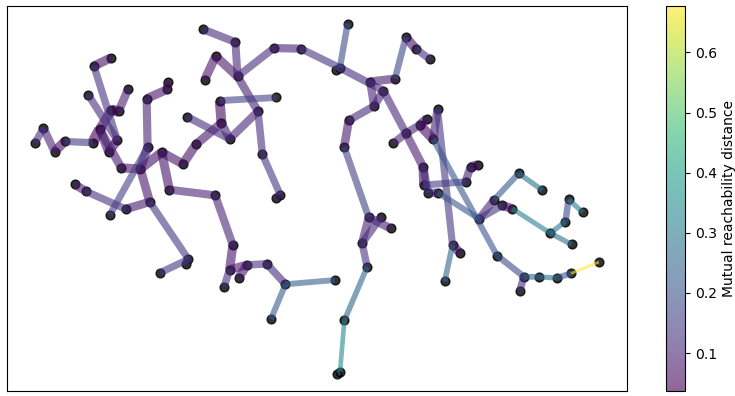
## Non-linearity is NOT visible as variance in price is low. so NO need to treat prices < 1 differently from prices > 1.

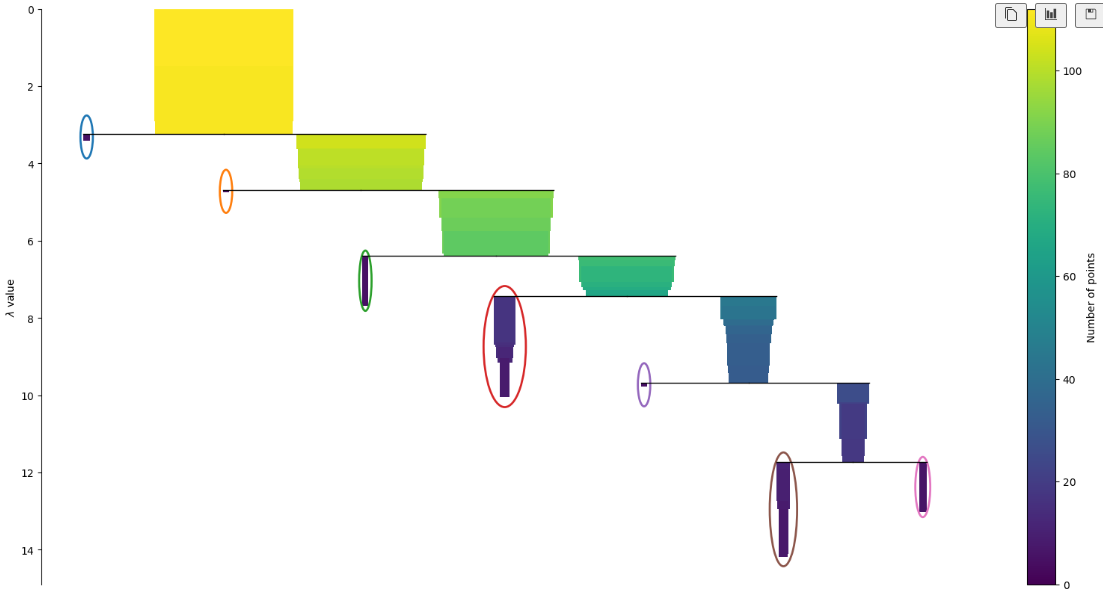
5) perform target volatility clustering across stocks using summary statistics features like mean, median, min, max etc..

Ans: The stocks volatilities are so homogeneous that it's difficult to separate/cluster them using summary stats features and hdbscan clustering algorithm. Maybe better features and/or different clustering algo. Might work. This is done in target\_eda\_across\_stocks.ipynb file

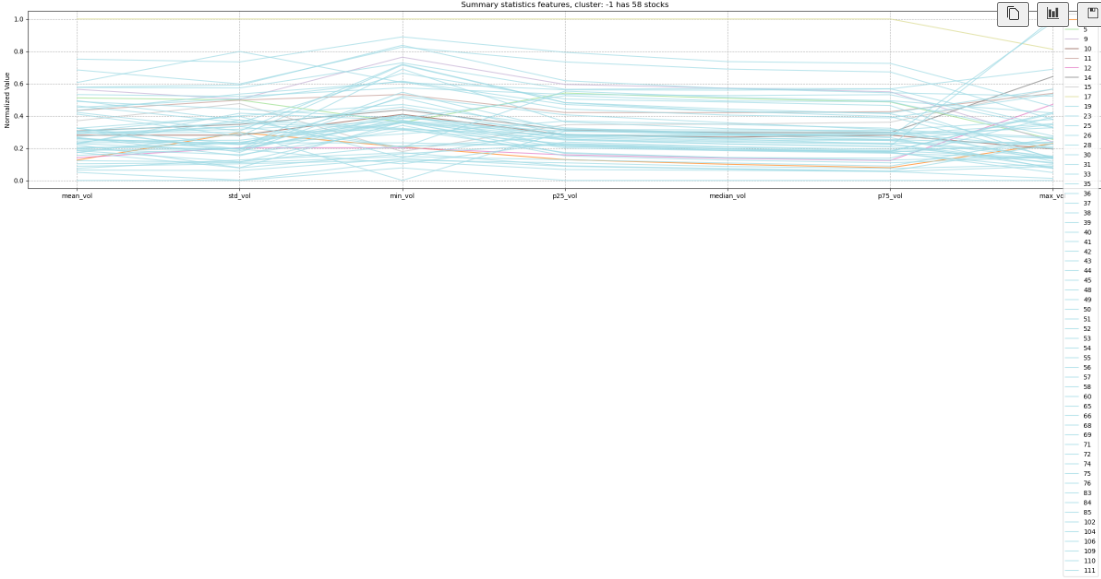
Colors represent different stocks. 

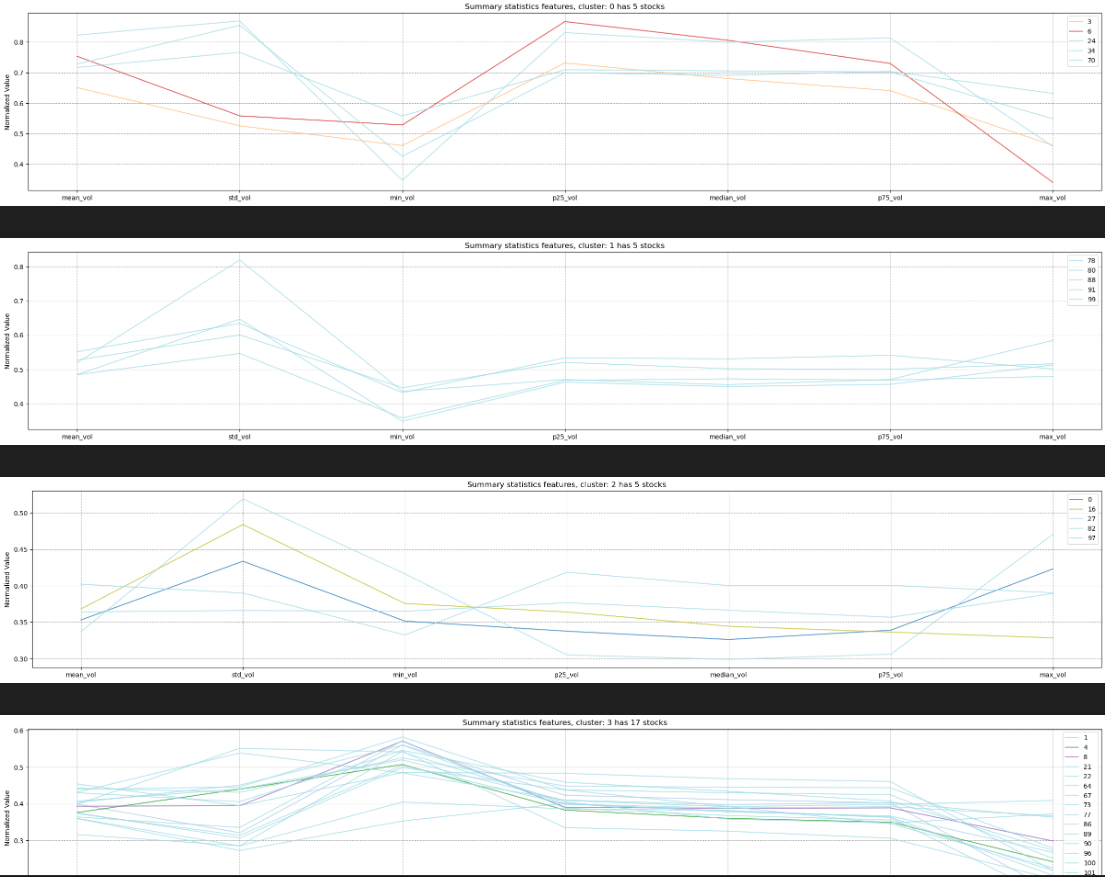
HDBSCAN analysis below was NOT fruitful!! .

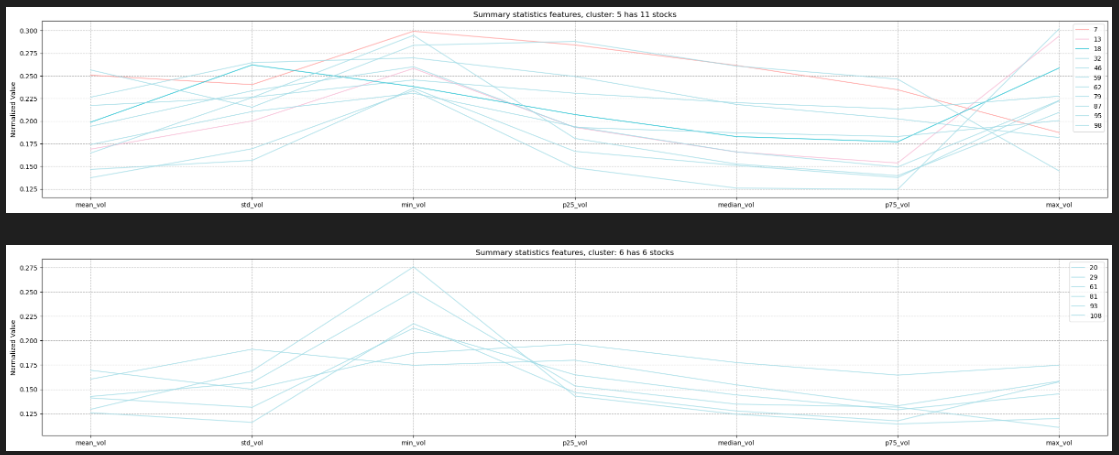




There are too many outliers in this clustering method.

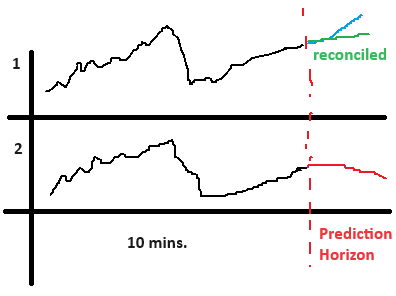






Hierarchical Agglomerative clustering using WARD linkage is better.

6) If two stocks (i.e. time series within a time\_id i.e. 10 min interval) are highly correlated for many time\_ids then we can improve the independent forecast of one stock by using the independent prediction of the correlated other stock. Reconcile the two predictions for the first stock to get more accurate forecast for 1st stock. Correlation captures shared patterns between the two stocks while independent stock prediction is based solely on within stock patterns. Do similarly for stock 2 using stock 1’s prediction. The other stock is acting like a covariate.



This analysis is SAME as in suggestion 3) which is **DONE and updated in Key Insights file.** Although data analysed is from TARGET and NOT from input features i.e. book\_train and trade\_train data. It’s better still.

7a) Check if minimum/maximum of bidsize1 and asksize1 in a time\_id correlated with target realized volatitlity for the same time\_id?

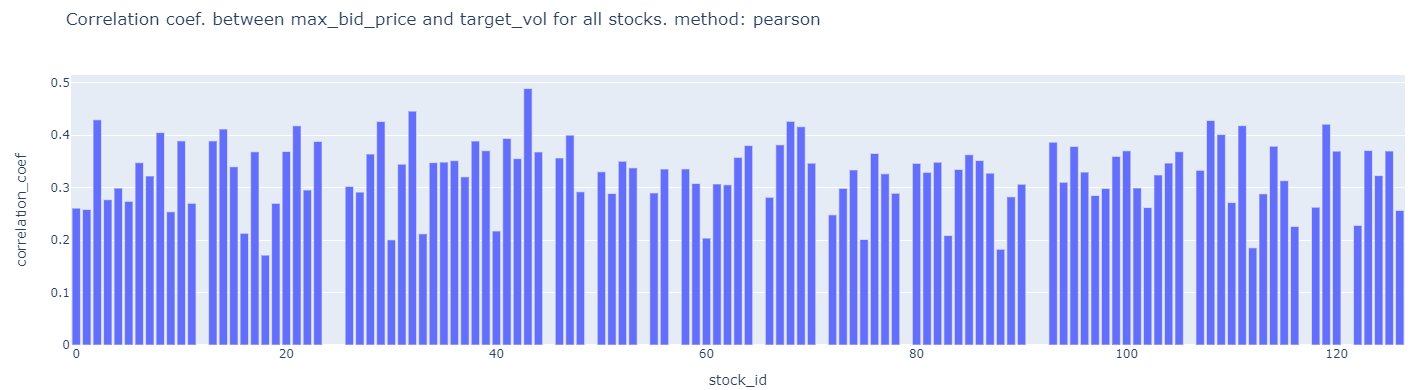
Graphs without level are level 1

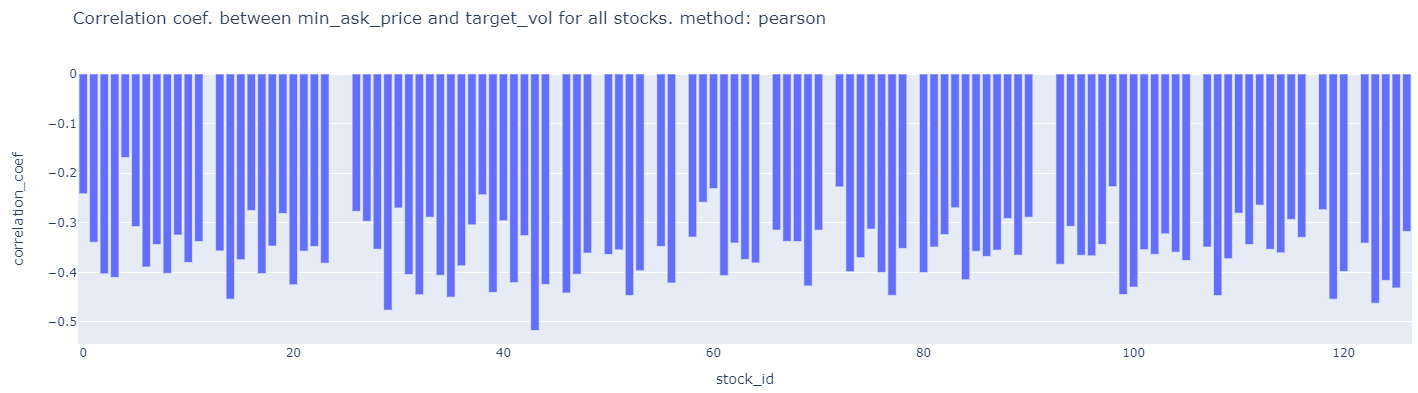


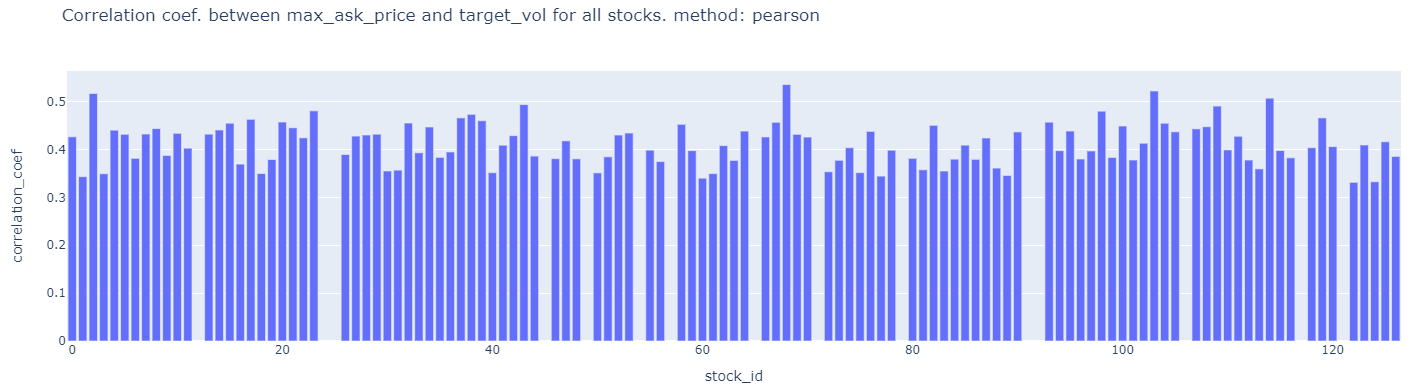


7b) Check if minimum/maximum of bidprice1 and askprice1 is correlated with target realized volatitlity for the same time\_id?





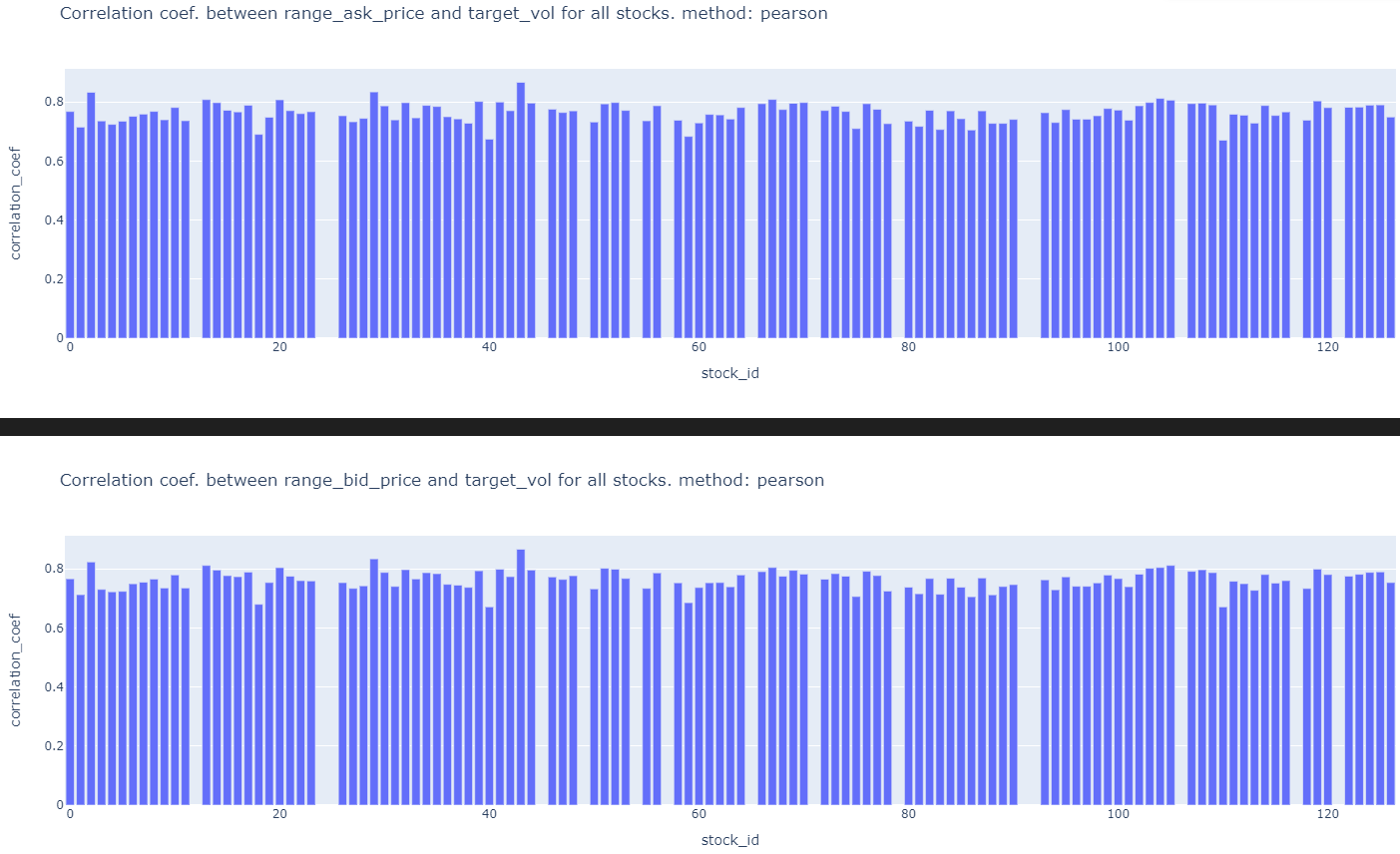




7c) check if the difference between minimum and maximum (i.e. range) of bidsize1 and range of asksize1 is correlated with target realized volatitlity for the same time\_id?

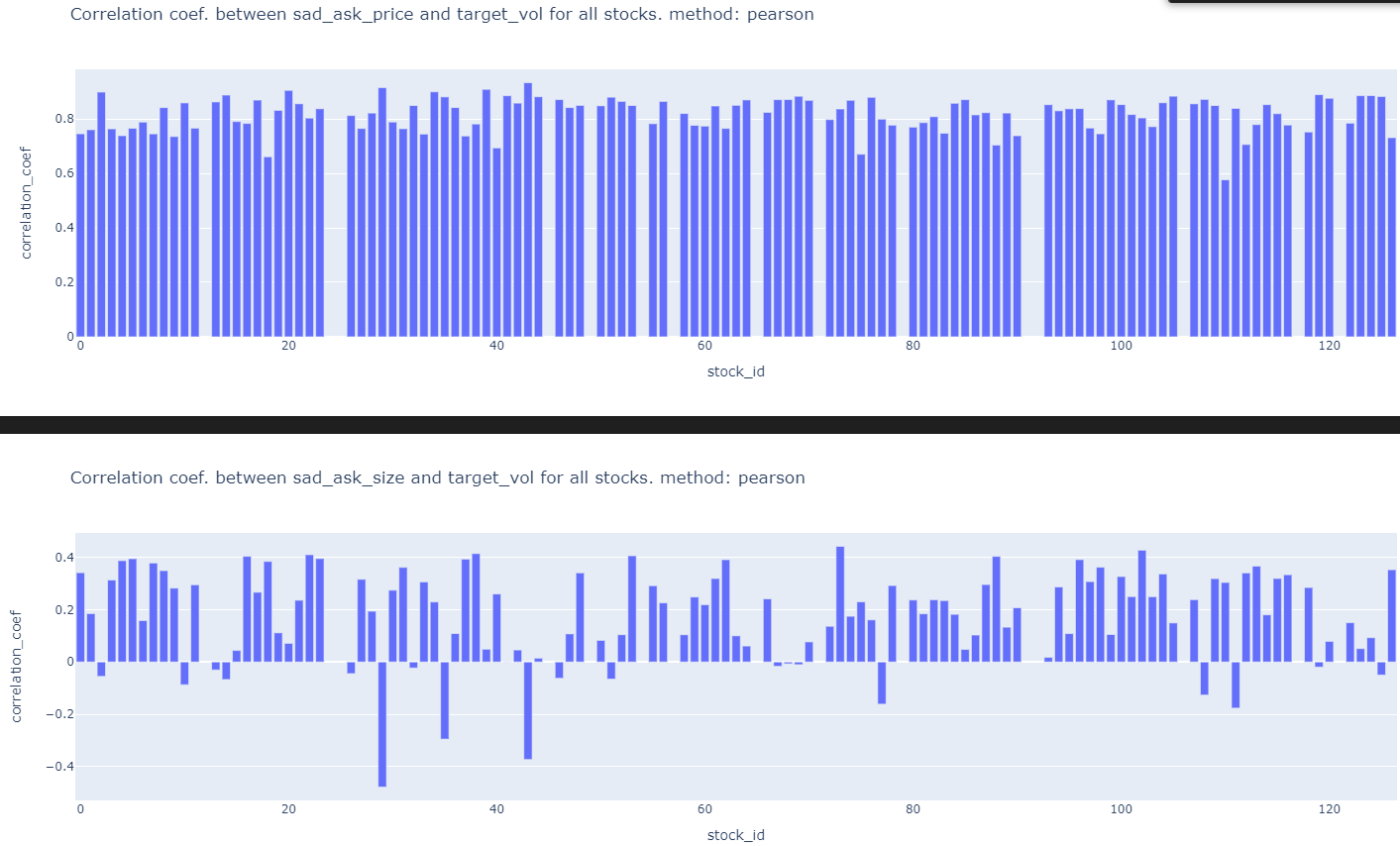


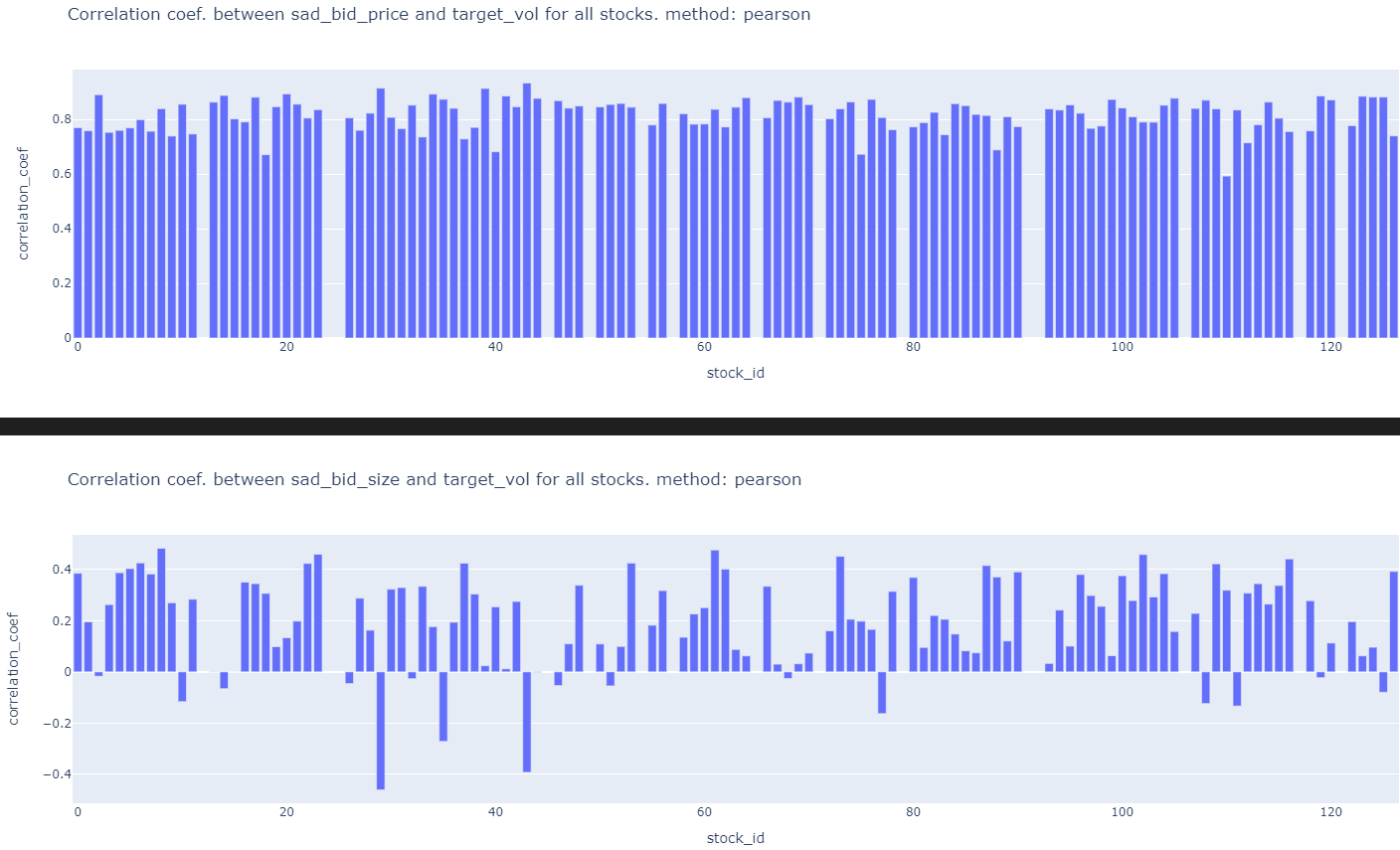
7d) check if the difference between minimum and maximum (i.e. range) of bidprice1 and range of askprice1 is correlated with target realized volatitlity for the same time\_id?



7e)Also check the correlation between (minimum – maximum) (i.e. range) vs. realized volatitliy of target. WHAT??

7f) Also check if the ~~average~~ sum of absolute bid\_price1 in t2 minus bid\_ price1 in t1 within a time\_id is correlated with target. Check the same for ask\_price1. Sum of absolute differences is better than average because price changes are too small and there are 600 seconds in a bucket so average values will be too small.





7g) calculate correlation between bid\_price1 and ask\_price1 in a time\_id (positive correlation leads to a larger/smaller wap’s numerator) then check correlation with target realized voaltitlity.

7h) calculate correlation between bid\_size1 and ask\_size1 in a time\_id (positive correlation leads to a larger/smaller wap’s denominator)

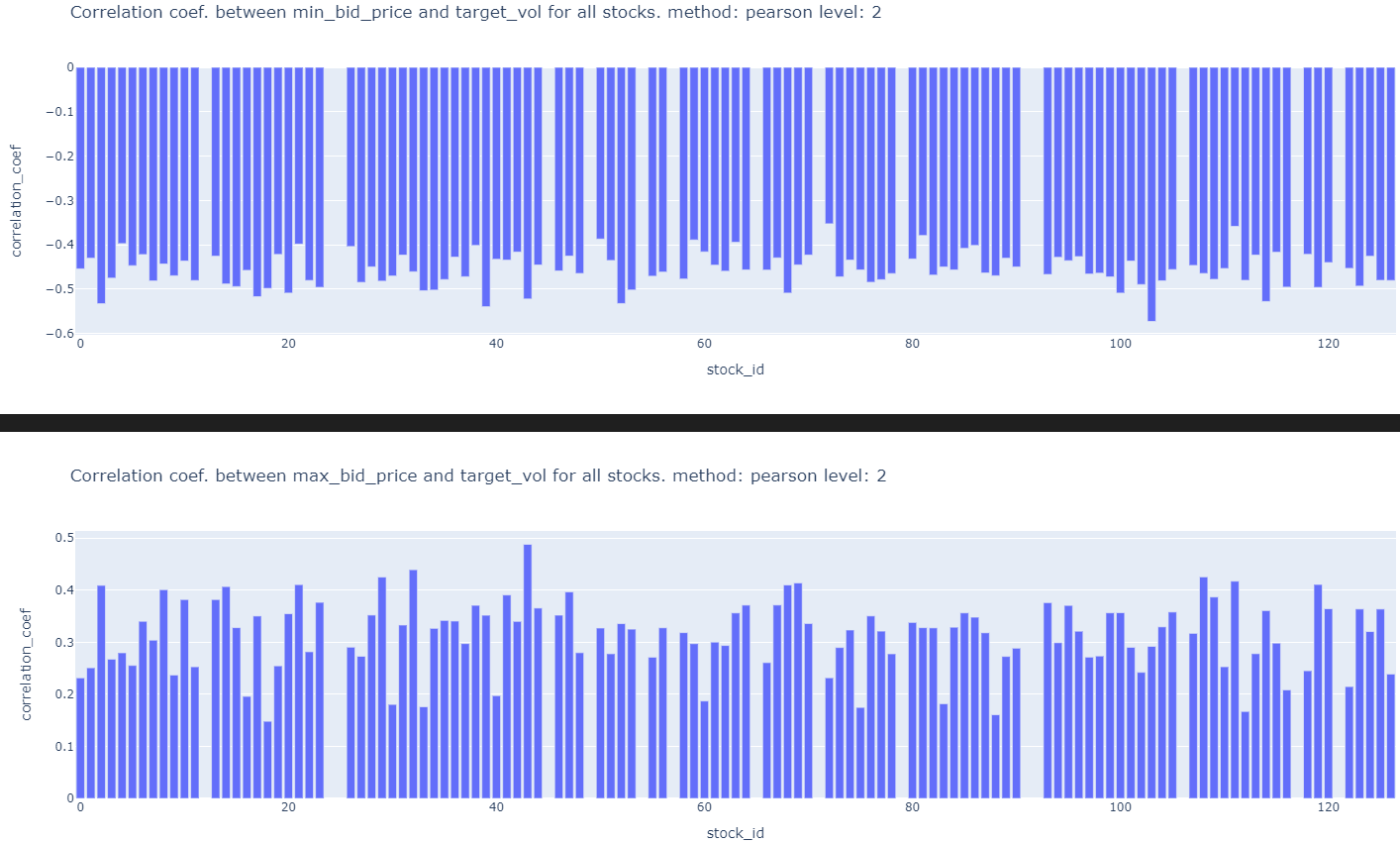
7i) Calculate pairs of correlation between a time series in this list [bid\_price1, ask\_price1] and this list [bid\_size1, ask\_size1] in a time\_id (negative correlation leads to a larger/smaller wap) i.e. corr(bid\_price1, ask\_size1), corr(bid\_price1, bid\_size1), corr(ask\_price1, bid\_size1), corr(ask\_price1, bid\_size1)

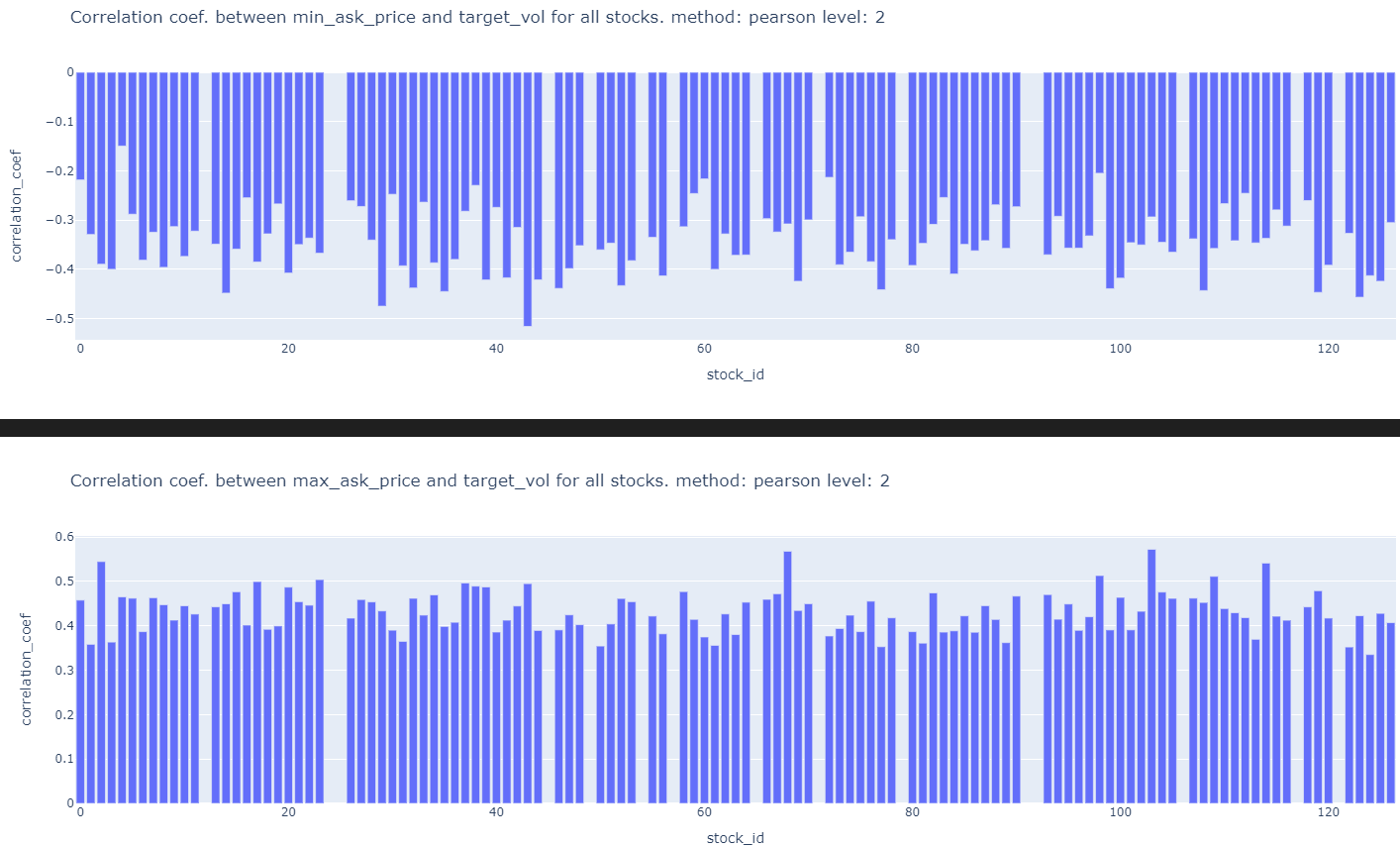






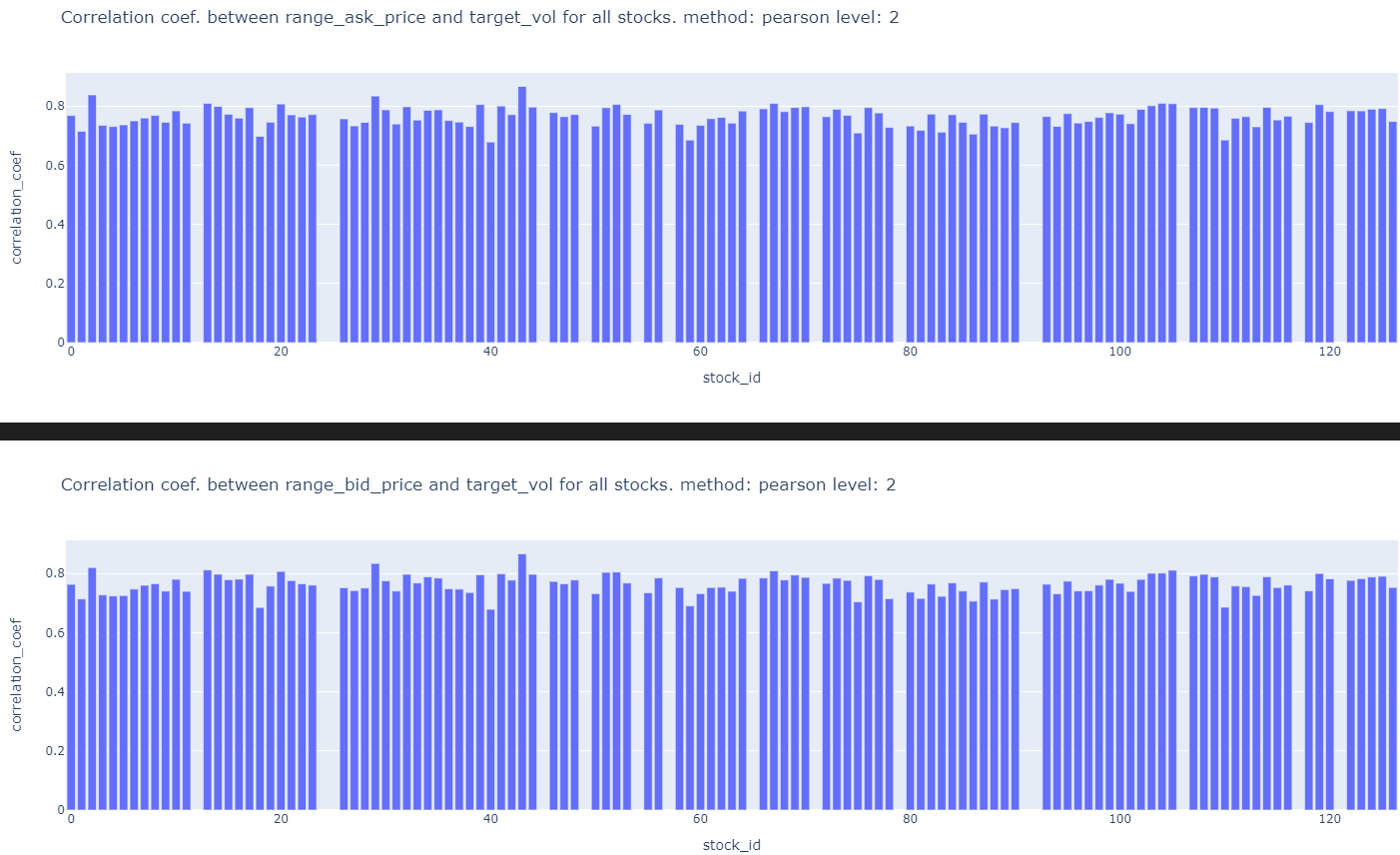
7j) Do the above for bid/ask\_price2, bid/ask\_size2 as well.

















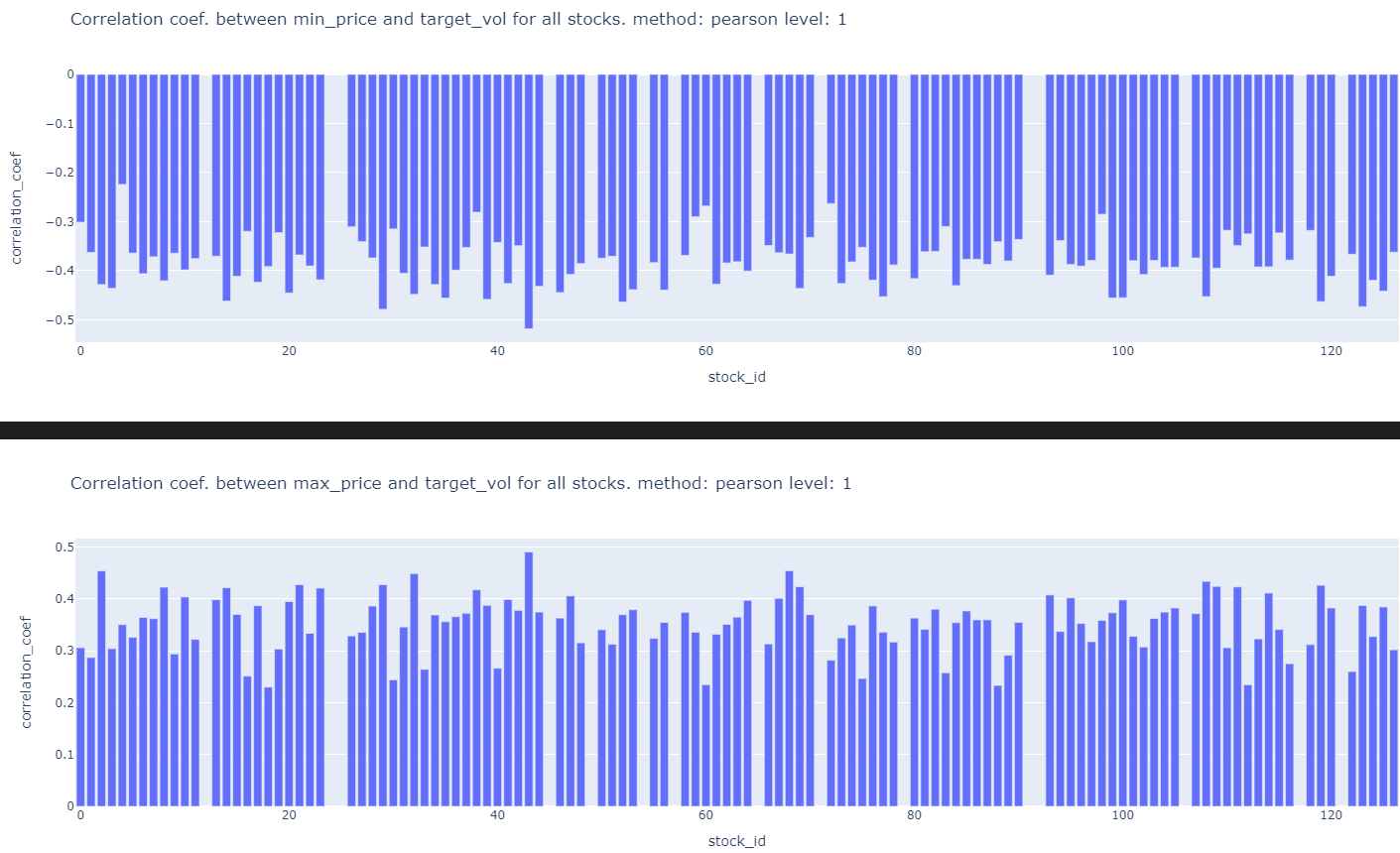


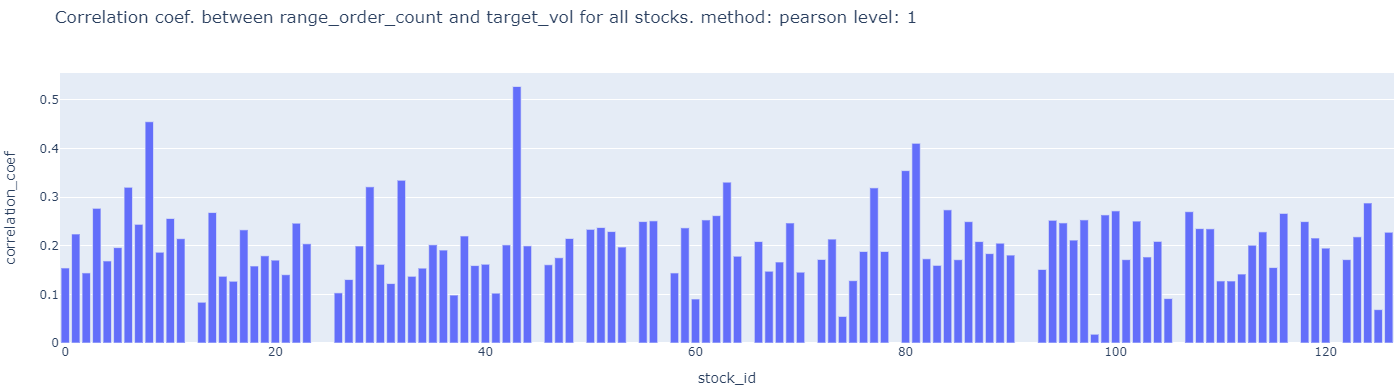
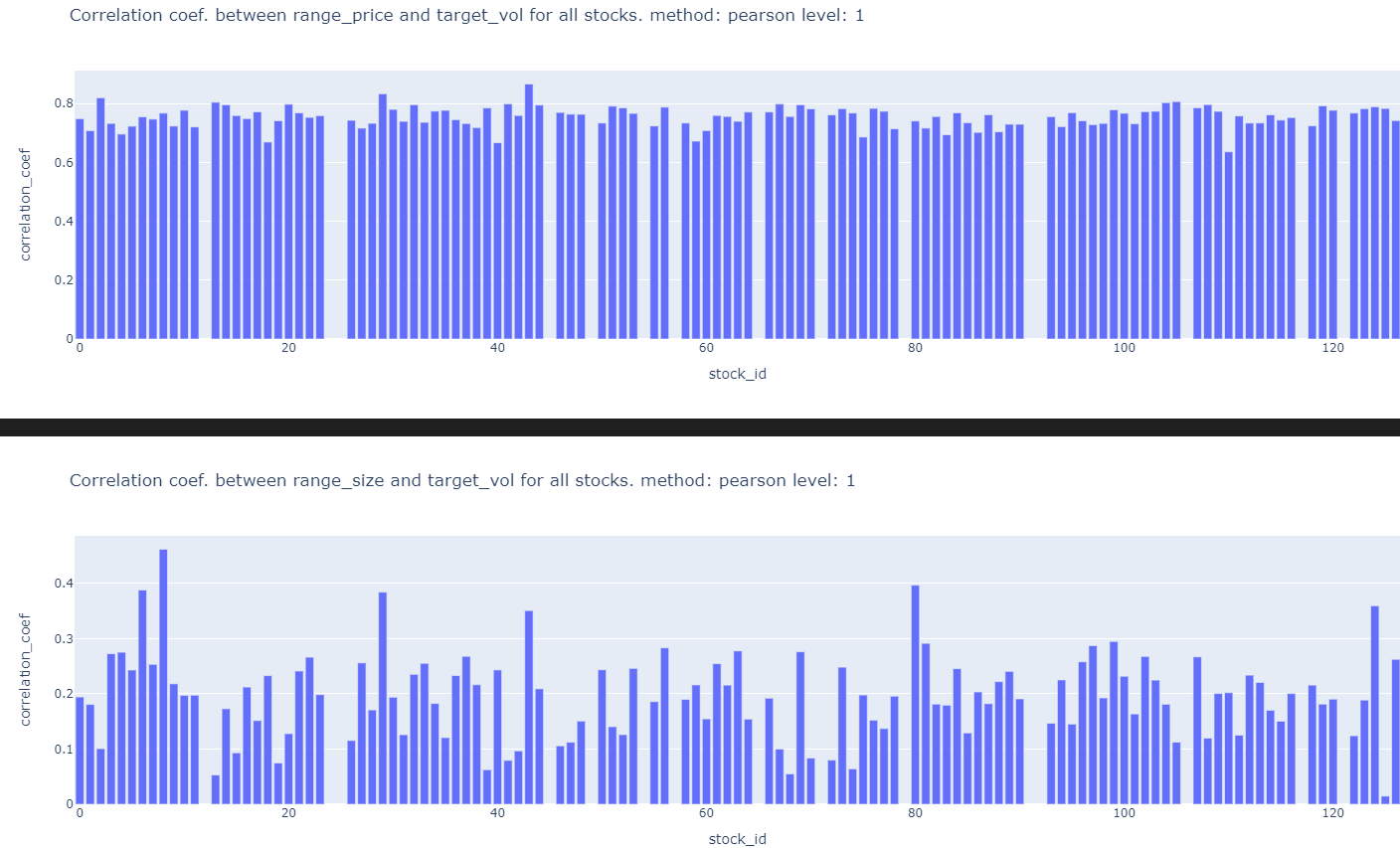


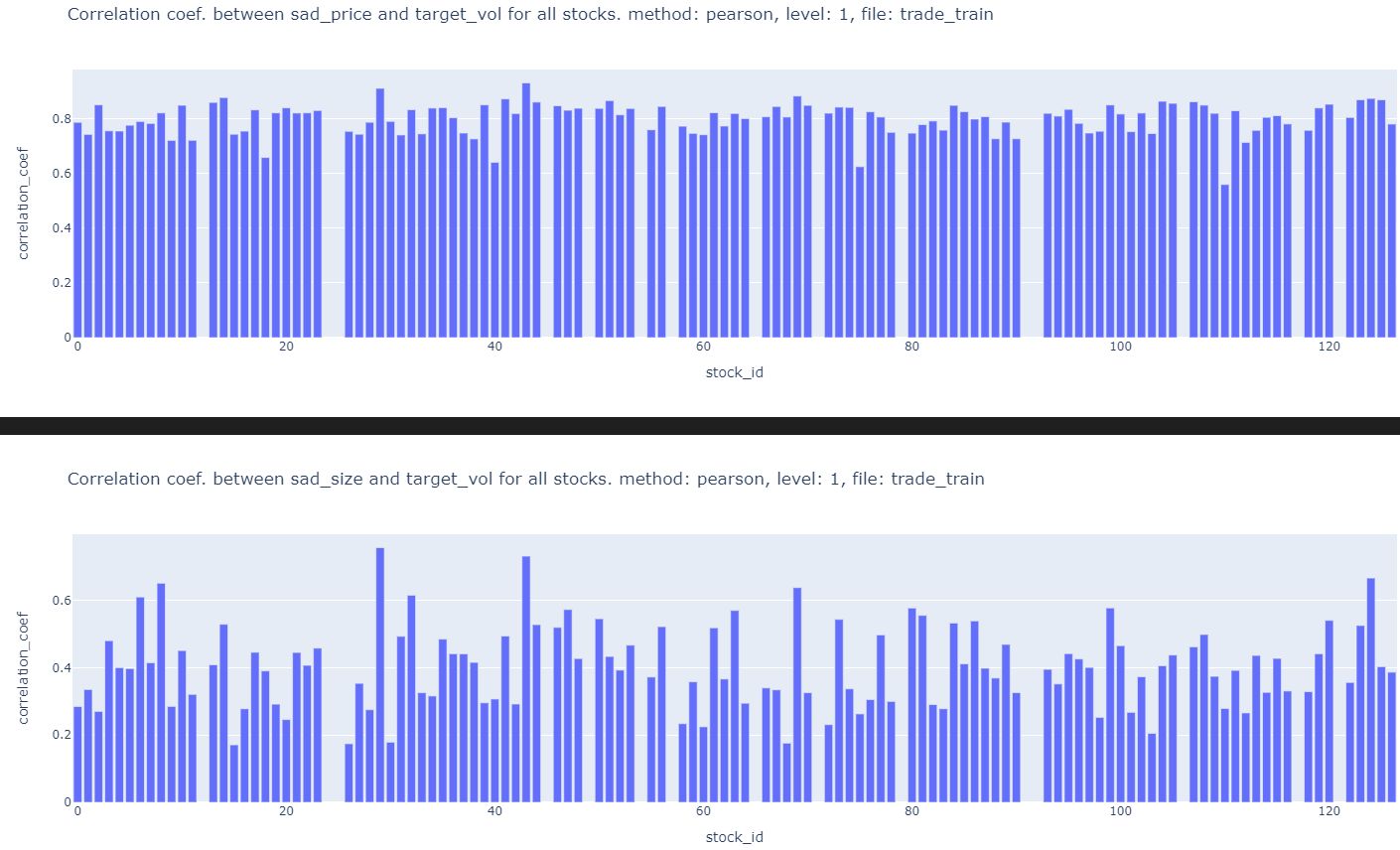


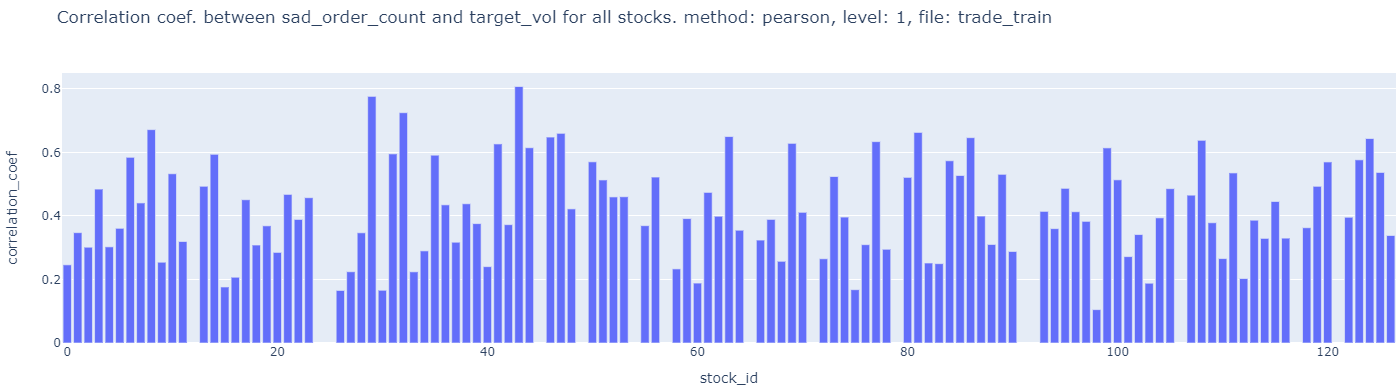
7k) Do the above for trade\_train.parquet

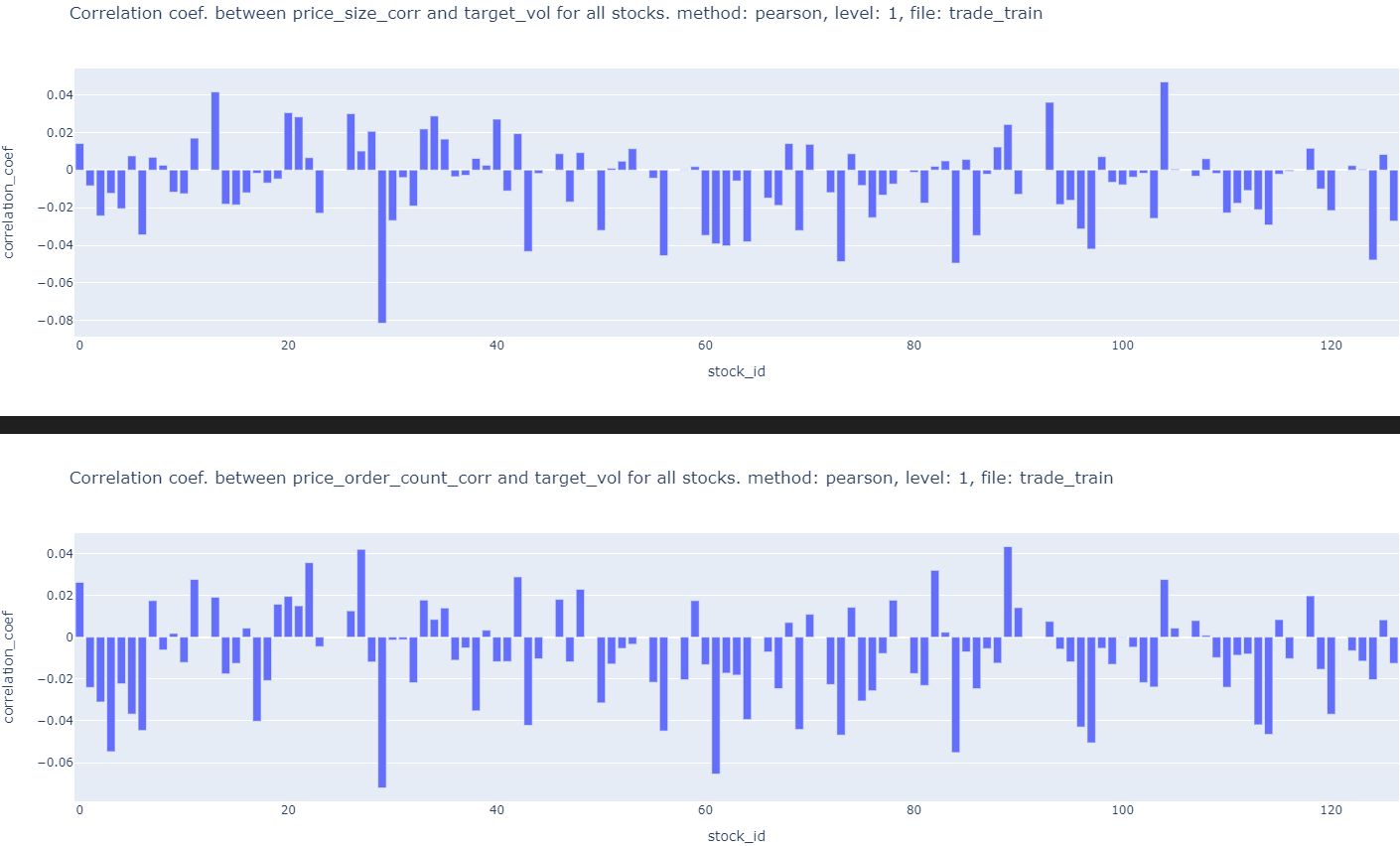
Slight code modification has to be done.









￼

This is because from the WAP formula we can see that there is large variation in wap time series caused by min and max values of bidprice1 and askprice1 bidsize1 and asksize1 respectively. This affects returns and volatility. ALL of suggestion 7 is done in features\_eda\_within\_stocks.ipynb

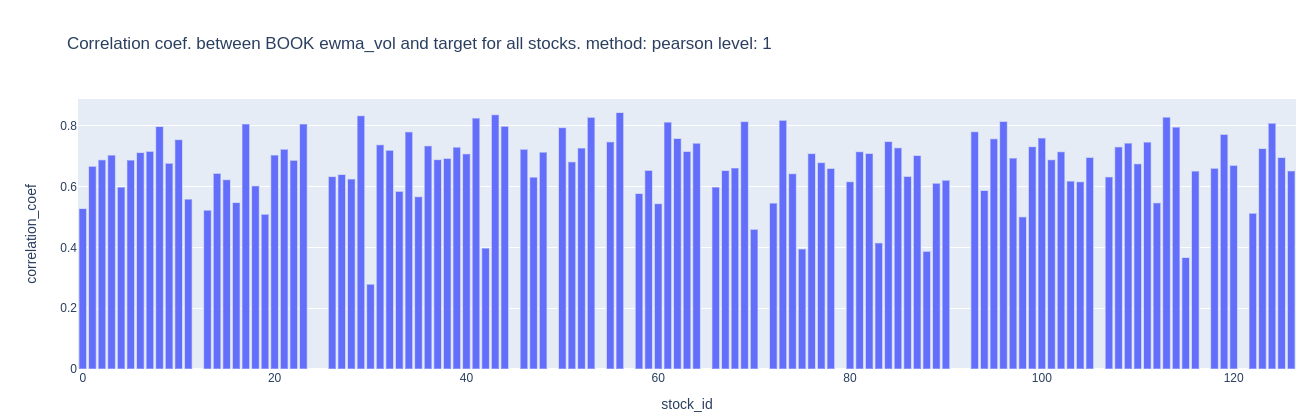
8) What features in the first 10 min that affect volatility in the next 10 mins?

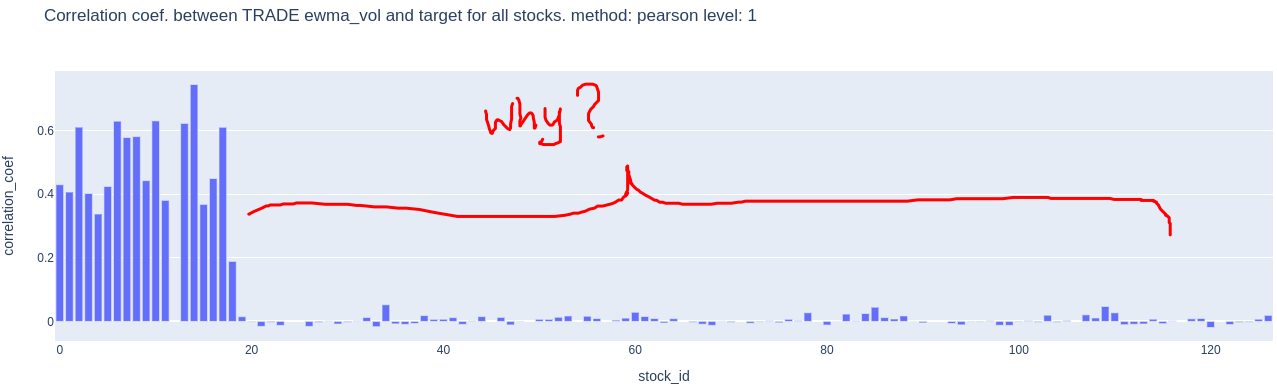
Feature that has Delayed effect on volatility?

Ans: Rolling average or moving average volatility can identify a lasting trend of high or low volatility. (peeterson)

This is not good because summed volatility is an increasing function

Exp. moving average of INstantaneous volatility (squared returns) can be used





9) Use a GARCH model on the time series available in the first 10 mins. to predict volatility into X (X<10) minutes of the second 10 minutes. Check correlation for different average x minutes realized volatility against the target realized volatility and choose the best x minutes of average volatility for each stock. (peeterson)

9a) The cons of a GARCH model is that the GARCH model does not distinguish between the impact of positive and negative negative returns. However, the leverage effect states negative returns have a greater impact on volatility and volatility clustering than positive returns.Therefore use GJR-GARCH or EGARCH.

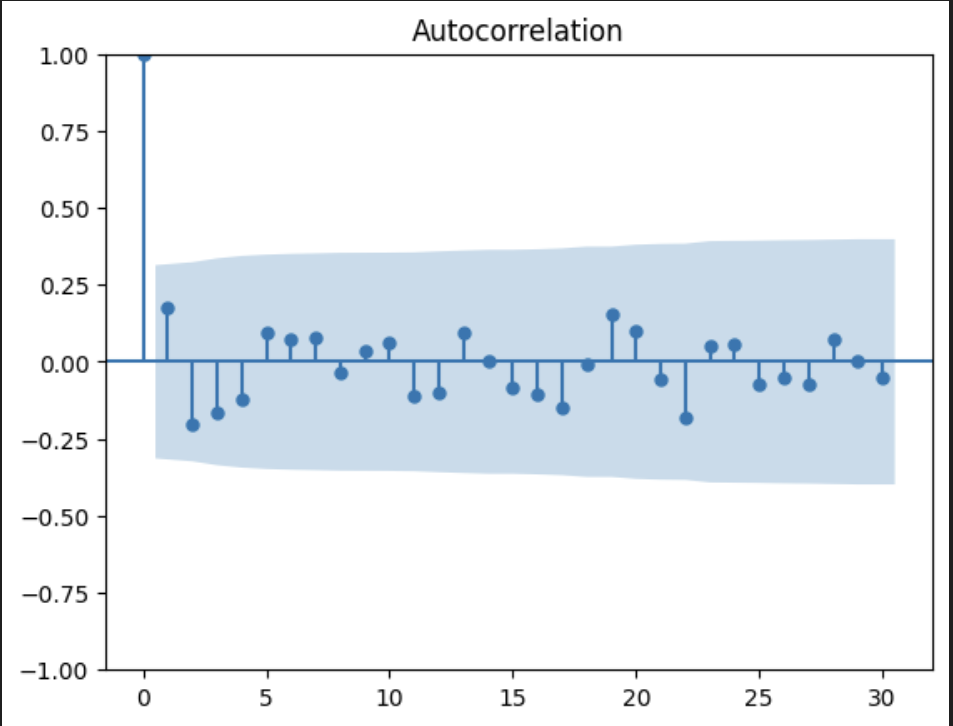
Use Apache spark to analyse 166 Million rows.

9b) use the clustering (of correlation matrix) done on pearson correlation of target in fitting a multivariate GARCH model.

9c) Can you reverse engineered time\_ids to create more contiguous trainings set and together with time\_id can be used in prediction.

https://www.kaggle.com/c/optiv￼er-realized-volatility-prediction/discussion/275825

10) Plot ACF of squared returns time series to leverage the Volatility clustering (persistence) phenomenon to predict/extend into second 10 minutes. Remember volatility is square root of sum of squared returns.(jx)



All autocorrelation values are under blue range so they are all close to zero

Series is random?

Yes, so it’s random or coming from a white noise distribution. We cannot predict instantaneous values but we can learn a probability distribution from these returns and predict things like averages, standard deviations and long term behaviours

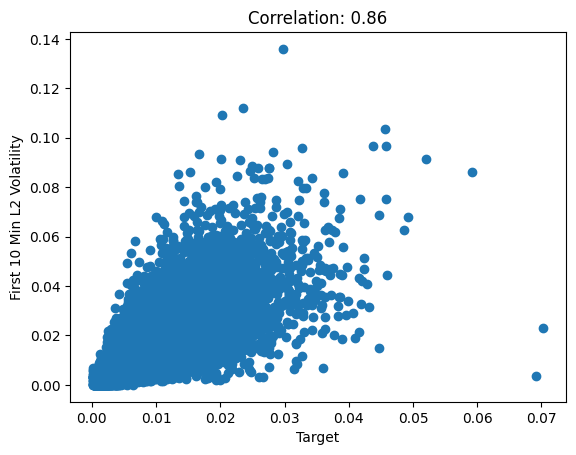
Ohh

~~11) WAP price trend up/down in the second 10 mins is negatively correlated with realized volatility in the 2nd 10 mins. This is called the leverage effect. Can we forecast trend in the 2nd 10 mins? Try using moving average of WAP of different window sizes and check if current price is above or below average. to predict short term trend in 2nd 10 mins.~~

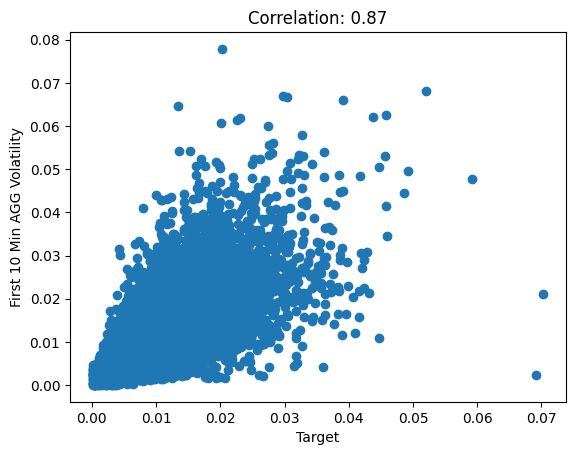
12) When calculating realized volatility did you first ffill for the missing book order seconds?

It does not make a difference because constant price/size/order\_count lead to zero volatility.

13) Compute Wap2 using ask/bid\_price/size 2.find the correlation of realized vol. Using level 2 wap with target and find the correlation of realized vol. Using average wap with target. Code is already available in features\_eda\_within\_stocks.ipynb in cells 1, 2, 225, 230, (jiaxu)

￼

Correlation for wap2 and target

￼

Correlation for aggregate wap and target

aggregated wap calculated using ((bidprice1 \* asksize1) + (bidprice2 \* asksize2) + (askprice1 \* bizsize1) + (askprice2 \* bizsize2)) / (bidsize1 + asksize1 + bidsize2 + asksize2)

14) Just use skew, kurtosis, min, max, std,and all other statistics for each bid ask wap etc… in each time id. (peeterson)

15) Check if Standard Deviations of Price Variables is POSITIVELY Highly Correlated with Target (peeterson)

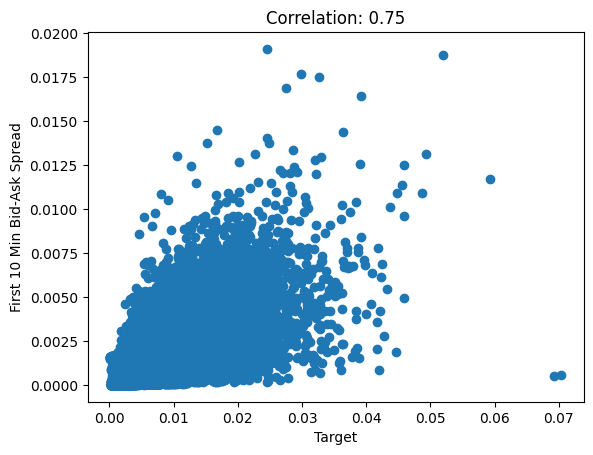
16)Check if Minimums of Prices Variables is NEGATIVELY Highly Correlated with Target; (peeterson)

17) the std of bid price was often more highly correlated with the target than the wap calculation. (peeterson)

18) Use bid ask spread. ask\_price1 / bid\_price1 - 1 (jiaxu)

(ask-bid)/bid

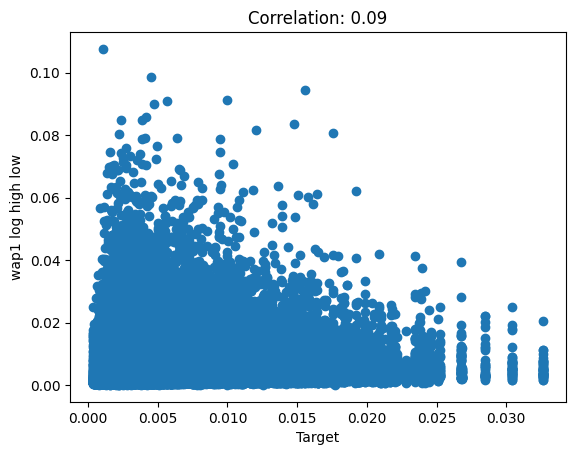
￼



Used mean to aggregate bid ask spread for each row\_id

19)wap1\_log\_high\_low - this is log(highest wap1 in interval) - log(lowest wap1 in interval)

Implement above feature (jiaxu)



20) Liquidity features:

Spread-related measures, Volume-based measures , Price-based measures

<https://www.kaggle.com/competitions/optiver-realized-volatility-prediction/discussion/276137> and Chatgpt 🙂

LIquidity describes how much of an asset can be purchased or sold immediately without moving the market (i.e. significant price changes). It is inversely proportional to volatility and bid-ask spread. It is directly proportional to trade volume/ size.

Generally, trades happen smoothly when the trade execution price (i.e. let’s call it “realized” WAP) is a compromise between the bid and ask price i.e. it lies in between to keep both the buying and selling party happy. The bid and ask price should approach this WAP price with time so that trades can happen at this price. This is like price discovery (equilibrium price). In this equation we are measuring how far away we are from this to happen. incorporating the order size also improves this information and this is defined as liquidity. Current time’s liquidity is an indicator of the future volatility to come. WAP is calculated all the time but “realized” WAP is when trade execution happens at that time.

First we need to find the wap (i.e. equilibrium price) for a given bid price, ask price, bid size and ask size. This can be found when we minimize the liquidity function. MInimizing liquidity does not mean liquidity is low value. E.g. 1000 ican be a minimum of a function similarly 10 can be a minimum of a function. Value of 10 is lower liquidity than 1000. The minimum is the correctly measured value of liquidity at that point in time. At different times we calculate different liquidity. The computed objective function (at the minimum) is computing liquidity based on parameters of the objective function. MInimizing this liquidity objective function helps us find the WAP that the bid\_price and ask\_price is likely to move towards in the future.

Given bid\_size, bid\_price, ask\_size and ask\_price, let us try to calculate the “realized” WAP at which trade execution is likely to happen in the future.

If WAP is far away from bid\_price/ask\_price then it indicating the bid ask spread is high. This is not ideal for trades or we can say it is ILLIQUID. Volatility has to come in the future and move prices in order to make it more LIQUID.

## **Defining Liquidity**

In finance, liquidity describes how much of an asset can be purchased or sold immediately without moving the price significantly. A reasonable numerical definition of liquidity then should start from the following assumptions:

liq\_1 = sum\_i[ bid\_size\_i/(wap\_1 - bid\_price\_i) + ask\_size\_i/(ask\_price\_i - wap1)]

or for wap\_2:

liq\_2 = sum\_i[ bid\_size\_i/(wap\_2 - bid\_price\_i)\*\*2 + ask\_size\_i/(ask\_price\_2 - wap1)\*\*2]

1. moving bids closer to asks always increases liquidity
   1. I.e. smaller (wap\_1 - bid\_price\_i)and smaller (ask\_price\_i - wap1)
2. increasing order sizes always increases liquidity
   1. I.e. Bigger bid\_size\_i and Bigger ask\_size\_i
3. adding additional orders to the book always increases liquidity
   1. I.e. sum\_i Refers to levels of the order book.

The above two behaviours are captured in the following equation as well.

f(x) = -bid\_size\*log(x-bid\_price) - ask\_size\*log(ask\_price-x) over the range [bid\_price, ask\_price].

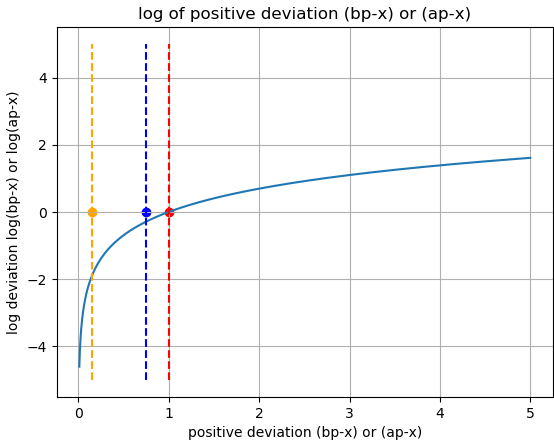
X-bid\_price = missing price on bid side. Ask\_price - x = missing price on ask side.

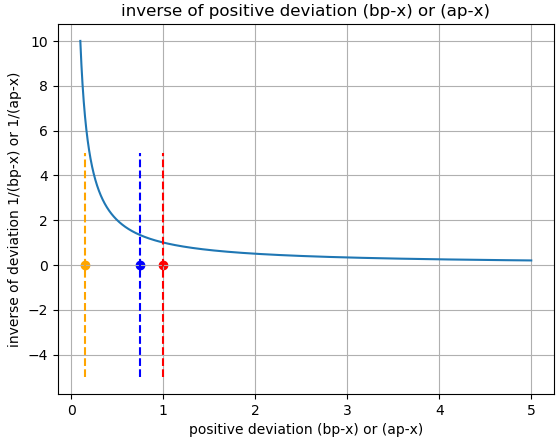
bid\_size\*log(x-bid\_price) = missing bid volume. ask\_size\*log(ask\_price-x) = missing ask volume.

Think of f(x) as the total missing volume. We want to find a price that minimizes the total missing volume. Missing volume means no trade can take place. When missing volume is minimized/less then trade can take place.

f(x) = bid\_size\*1/(x-bid\_price) + ask\_size\*1/(ask\_price-x) over the range [bid\_price, ask\_price].

In this equation we assume WAP i.e. x, x <= ask price or x >= bid price. AND bid price != ask price.





Instead of log we can also use other functions that can be computed faster. The function just has to have an asymptote at x=0. E.g. 1/x . Note: sign change in cost function.

Firstly, the deviation is highly unlikely to be greater than 1. Deviation closer to zero deviation incurs more cost.

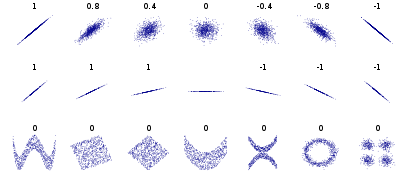
In the code the wap (equilibrium price) is computed using the function full\_book\_wap\_bisect( df\_book, lvl=0) using a bisection algorithm iteratively and the wap moves towards the price (either bid or ask price) which has bigger bid or ask size and smaller . increasing lvl increases importance/weight of distance with respect to size. Higher lvl. Additionally, odd values of lvl make s negative and if bid size/price > ask size/price then p moves towards ask price. Similarly, opposite is true. For even values, always pushed towards ask price??

21) Explore the rest of the 46 features in clustering analysis to see if there are other feature that give a high score?

Ans: No need to try this because Jager must have tried this already. Actually we can try our features through this clustering method.

# **Feature SELECTION**

22) always check correlation between features to ensure no multicollinearity issues. (However, this method has the downfall of only checking “LINEAR/Spearman” correlation )



How to capture last row correlation patterns in fig above? Need a non-linear measure or feature transformation to make them like row 1 or row 2.

23) features target real vol

7 pc target real vol

<https://neptune.ai/blog/feature-selection-methods>

<https://machinelearningmastery.com/feature-selection-for-regression-data/>

<https://machinelearningmastery.com/feature-selection-with-real-and-categorical-data/>

Filter-based methods

Borutapy

QQ-plot

<https://stats.stackexchange.com/questions/101274/how-to-interpret-a-qq-plot>

<https://xiongge.shinyapps.io/QQplots/>

***Be sure to REMOVE all the columns with the word ” TARGET “ in it before splitting the data into x and y for training. E.g. log\_target and log\_target\_standardized***

Try to use these different columns as targets in training and see if it can improve the prediction performance.

**EXPLORING FEATURES AND INITIAL ELIMINATION.**

* Jager’s features from 0:152 , stock\_id to target
* Our features from 153:237 ,log\_target to pear\_corr\_90\_clusters\_labels

1. V1proj\_25\_vvc2: v1 = volatility 1, proj means some ratio or subtraction type of comparison. 25 is time index. Vv = volume , c1 = cluster 1. Stocks clustered according to trade volume. WHat is the ratio of wap1\_log\_price\_ret\_vol\_buks in last 25 bins to all bins. C2 has different number of trade volume clusters from c1.
2. V1proj\_25\_vvc3: similar to above. C3 has different number of trade volume clusters from c2 and c1.
3. V1spprojt15f25\_c1: v1 = volatility 1, sp = spread , proj means some ratio or subtraction type of comparison. t15 is to time index. f25 is from time index, c1 = cluster 1. fraction of null values: 0.7365782921302211 num of null values: 315942

DISCARD

1. V1spprojt15f25\_c2:Same as above, fraction of null values: 0.7249680602053472, num of null values: 310962 DISCARD
2. V1spprojt15f25\_c3: fraction of null values: 0.7081821827236019, num of null values: 303762, num of inf values: 126, num of unique values: 4592, fraction of unique values: 0.010705659638357595 DISCARD
3. V1spprojt15f25\_c4:fraction of null values: 0.688297912023351, num of null values: 295233, num of inf values: 85, num of unique values: 7323, fraction of unique values: 0.01707263622205851 DISCARD
4. V1spprojt15f25\_vc1: fraction of null values: 0.7340650732517042 DISCARD
5. V1spprojt15f25\_vc2: DISCARD
6. V1spprojt15f25\_vc3: DISCARD
7. Tvpl2\_rmed2v1: tvpl1 = trade volume per liquidity\_wavg 2. rmed2v1= r, ratio of Med, median, 2, to v1, stocks overall wap1 return volatility1. Median along stock dimension.
8. Tvpl2\_rmed2v1lf25: similar as above but using liquidity\_wavg from bin 25 to 30 only.
9. Tvpl2\_rmed2v1lf29: similar as above but using liquidity\_wavg from bin 29 to 30 only.
10. Tvpl2: ratio of average of all buckets trade\_volume to average of all buckets liquidity2. num of -inf values: 19

Removing the 19 negative inf values using x = x[x > -np.inf] solves the problem. Log (0) is causing -inf. This is ok because LGBM can handle -inf values.

1. Tvpl2\_liqf10: num of -inf values: 19

Similar to above except liquidity\_wavg starts from index 10 onwards.

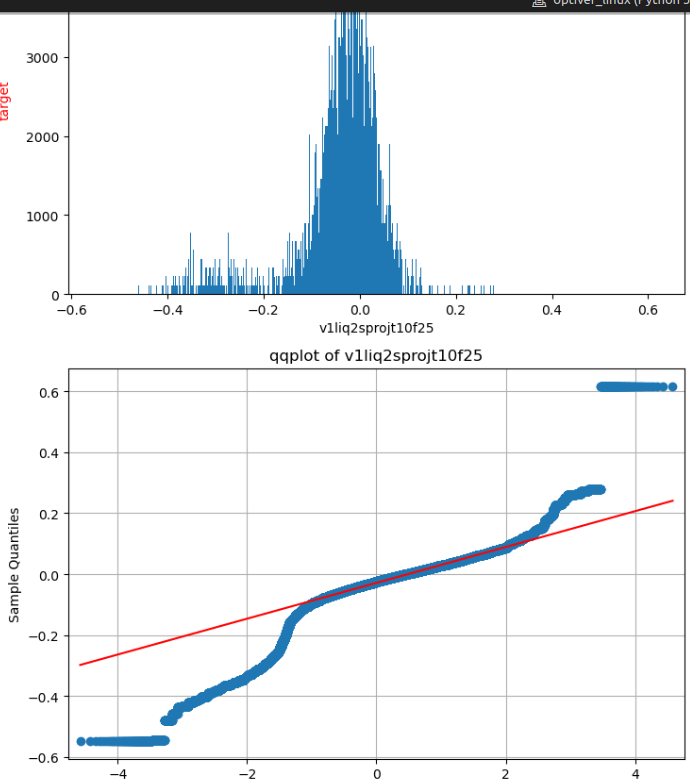
1. Tvpl2\_liqf20: num of -inf values: 19

Similar to above

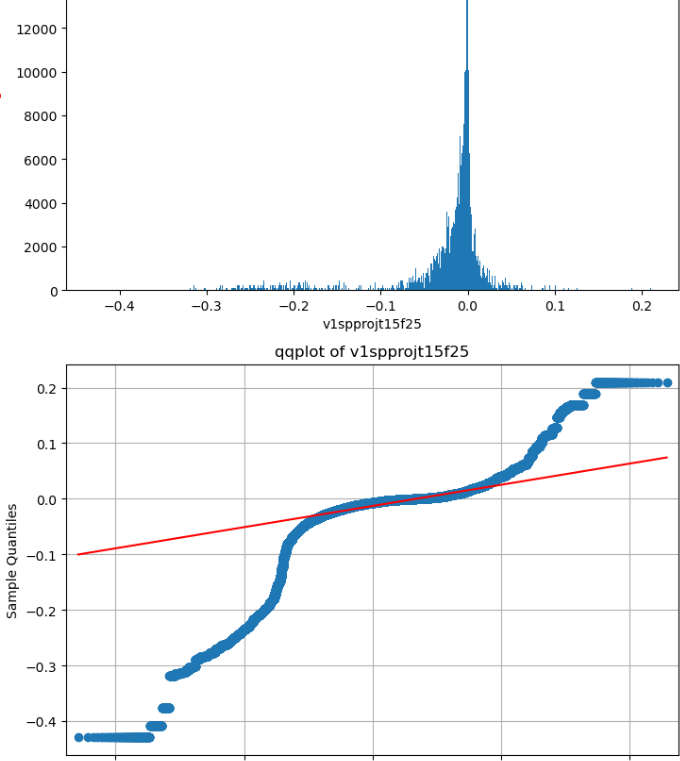
1. Tvpl2\_liqf29: num of -inf values: 19Similar to above
2. Tvpl2\_smean\_vol: ratio of average of all time ids and buckets trade\_volume to average of all bucket liquidity2, shape of (3830,112,1). Tvpl2 = trade volume averaged over all buckets and time\_ids (1,112,1) per liquidity2\_wavg average across all buckets (3830,112,1). Smean\_vol = stock’s mean volume. How liquidity at each time id compares with the full trade volume (at all time time ids) of every stock.

1. Tvpl2\_smean\_vol\_liqf10: same as above but liquidity2\_wavg starts from bin index 10
2. Tvpl2\_smean\_vol\_liqf20: same as above but liquidity2\_wavg starts from bin index 20
3. Tvpl2\_smean\_vol\_liqf29: same as above but liquidity2\_wavg starts from bin index 29
4. V1liq2projt5: To check the trend or see changes/ratio in liquidity2\_wavg between first 5 bins vs last 2 bins from 28. NOt exactly sure why need to do \*\*⅛??
5. V1liq2projt10: same as above but starts at liquidity2\_wavg 10 divided by index 28: (last 2 bins)
6. V1liq2projt20: same as above starts at bin index 20divided by index 28: (last 2 bins)
7. Liqt10rf29: ratio of average of first 10 buckets liquidity2\_wavg to average of last 1 buckets liquidity2, shape of (3830,112,1)
8. Liqt20rf29: ratio of average of first 20 buckets liquidity2\_wavg to average of last 1 buckets liquidity2, shape of (3830,112,1)
9. V1liq2sprojt10f25: how liquidity2\_wavg changes/ratio in first 10 bins to last 5 bins.

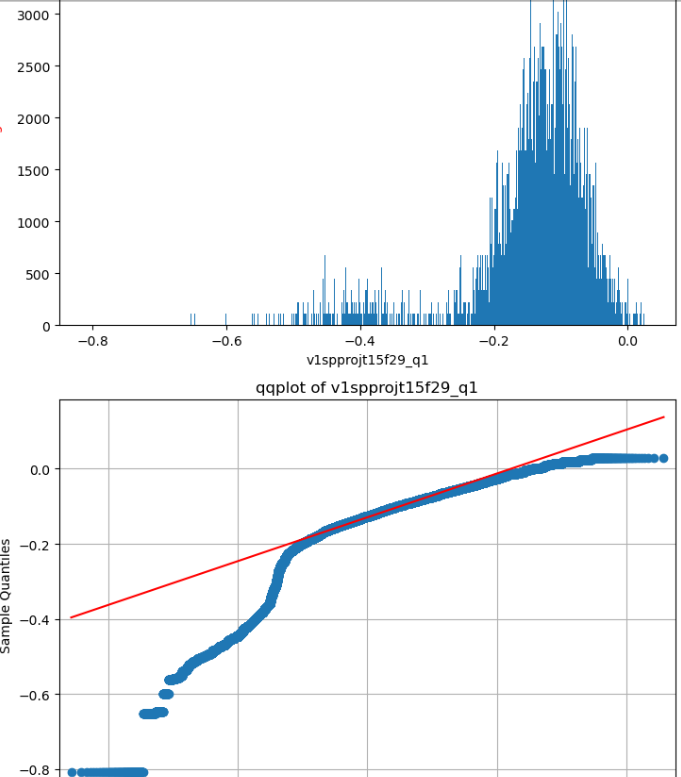
Very few outliers due to gap in qqplot. Bimodal distribution



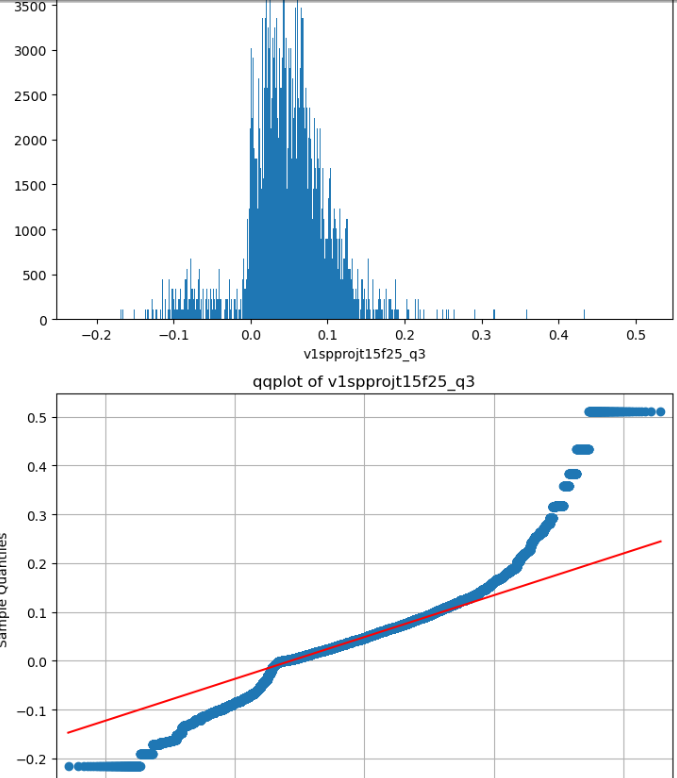
1. V1liq2sprojt5f25: similar to above but starting time bins is 5. Also a Bimodal distribution
2. V1spprojt10f29: median along stock id axis of ratio of mean of all buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). starting time bins is 10. Also a Bimodal distribution
3. V1spprojt15f25: median along stock id axis of ratio of mean of first 15 buckets log\_spread2\_wavg to mean of last 5 bucket log\_spread2\_wavg , shape of (3830,1,1). Multimodal distribution



1. V1spprojt15f29: # median along stock id axis of ratio of mean of first 15 buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). Similar to above.
2. V1spprojt15f29\_q1: 25% quantile along stock id axis of ratio of mean of first 15 buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). Multimodal.



1. V1spprojt15f29\_q3: # 75% quantile along stock id axis, of ratio of mean of first 15 buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). Bimodal.
2. V1spprojt15f25\_q1: similar to above. bimodal.
3. V1spprojt15f25\_q3: similar to above. bimodal.



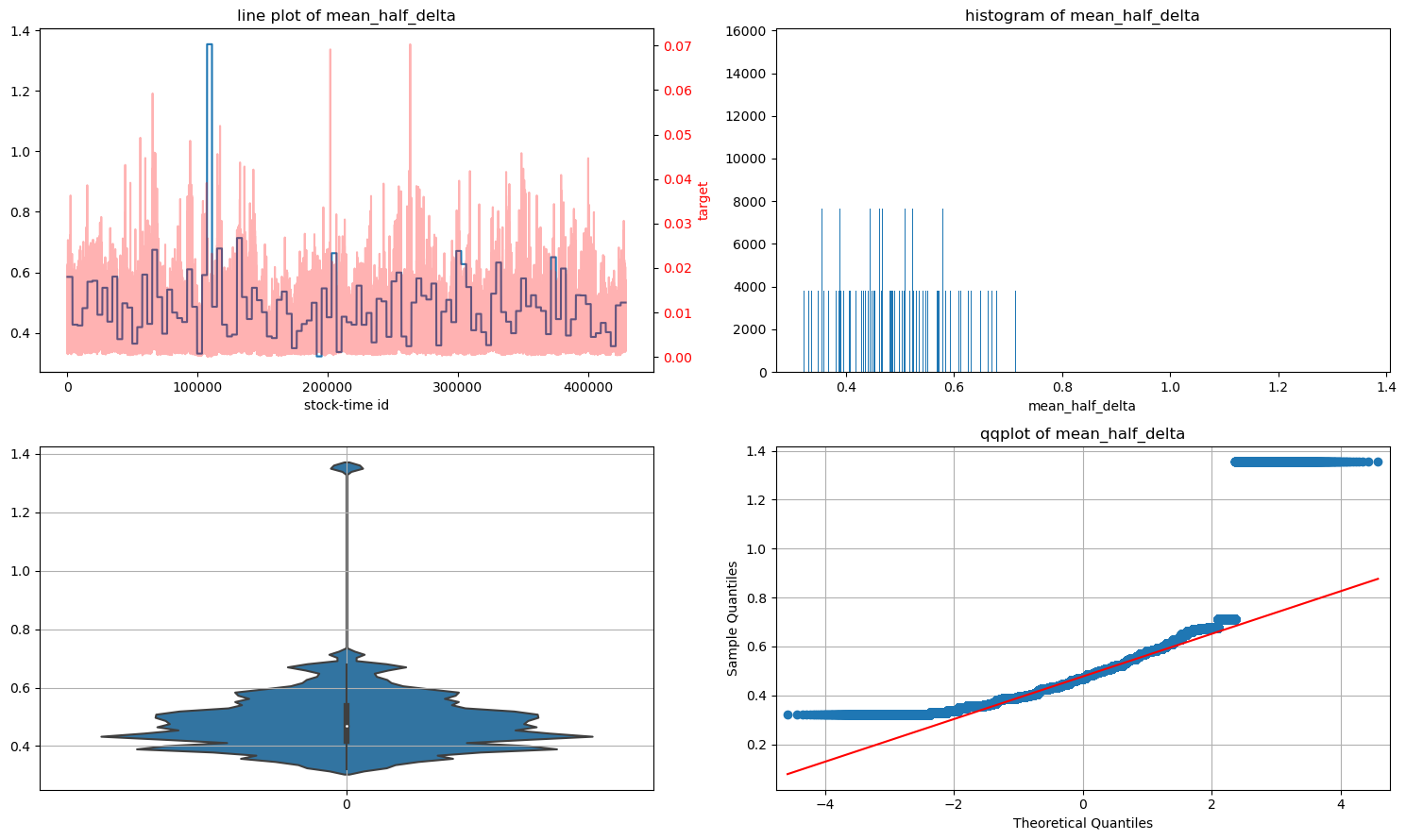
1. V1spprojtf29\_q1: # 25% quantile along stock id axis,of ratio of mean of all buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). BImodal.
2. V1spprojtf29\_q3: # 75% quantile along stock id axis,of ratio of mean of all buckets log\_spread2\_wavg to mean of last 1 bucket log\_spread2\_wavg , shape of (3830,1,1). unimodal
3. V1spprojtf25\_q1: # 25% quantile along stock id axis,of ratio of mean of all buckets log\_spread2\_wavg to mean of last 5 bucket log\_spread2\_wavg , shape of (3830,1,1).bimodal
4. V1spprojtf25\_q3: # 75% quantile along stock id axis,of ratio of mean of all buckets log\_spread2\_wavg to mean of last 5 bucket log\_spread2\_wavg , shape of (3830,1,1). bimodal
5. Wap1\_log\_price\_ret\_vol\_from\_0: taking log of Wap1\_log\_price\_ret\_vol/Wap1\_log\_price\_ret\_vol which is log(1) so all are zero. DISCARD
6. Wap1\_log\_price\_ret\_volstock\_mean\_from\_0: Mean across stock ids,

FORMULA = np.median( np.mean(wap1\_log\_price\_ret\_vol[:,:,ffrom:]/stocks\_overall\_wap1\_log\_price\_ret\_mean, 2, keepdims=True), 1, keepdims=True)

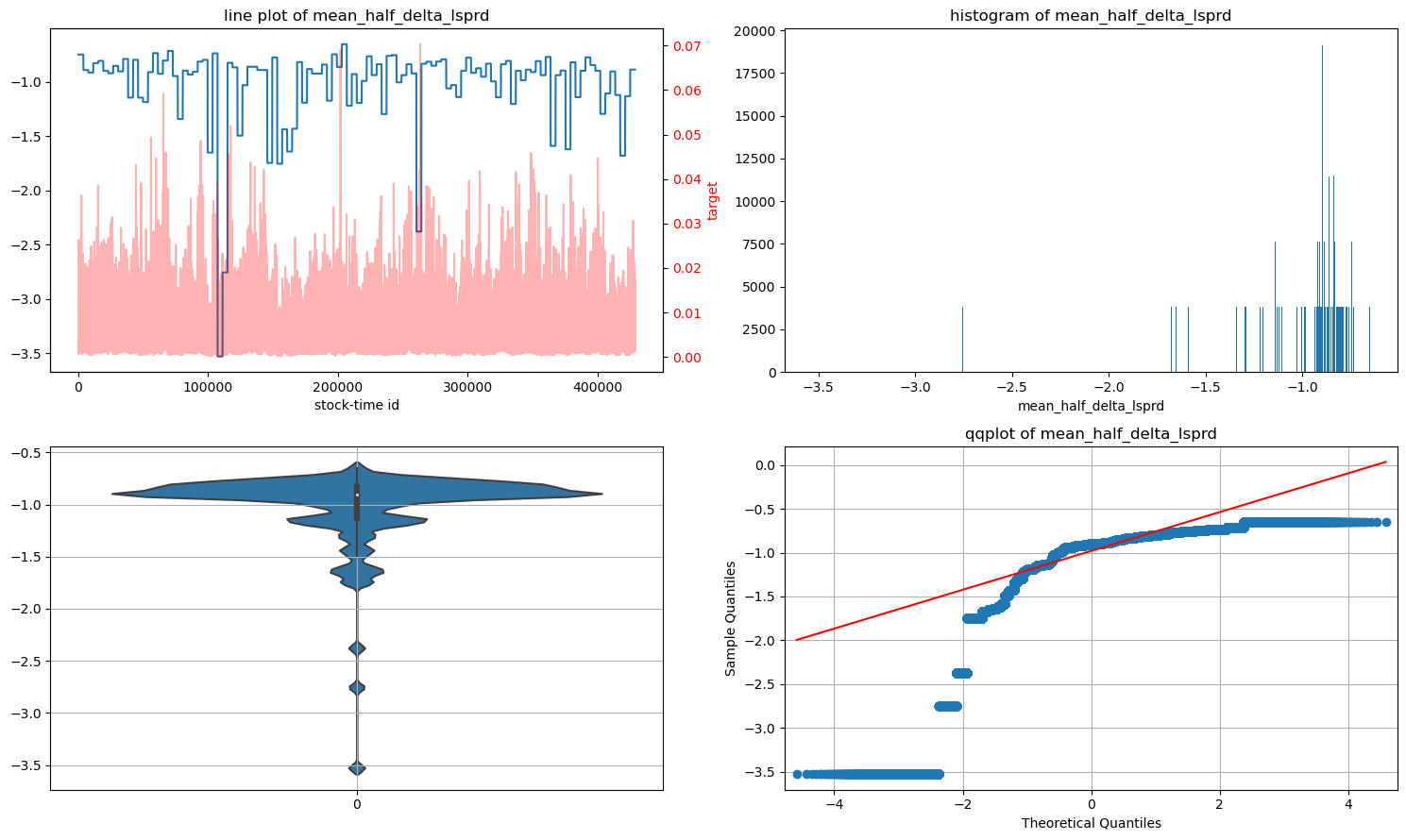
1. Wap1\_log\_price\_ret\_vol\_from\_10: similar to above
2. Wap1\_log\_price\_ret\_volstock\_mean\_from\_10: similar to above
3. Wap1\_log\_price\_ret\_vol\_from\_20: similar to above
4. Wap1\_log\_price\_ret\_volstock\_mean\_from\_20: similar to above
5. Wap1\_log\_price\_ret\_vol\_from\_25: similar to above num of inf values: 5
6. Wap1\_log\_price\_ret\_volstock\_mean\_from\_25: similar to above
7. Vol1\_mean: # standardize wap1\_log\_price\_ret\_vol by dividing by mean of wap1\_log\_price\_ret\_vol over all time periods for each stock.
8. Mean\_half\_delta: num of unique values: 112 (DISCARD)

fraction of unique values: 0.0002611136497160389

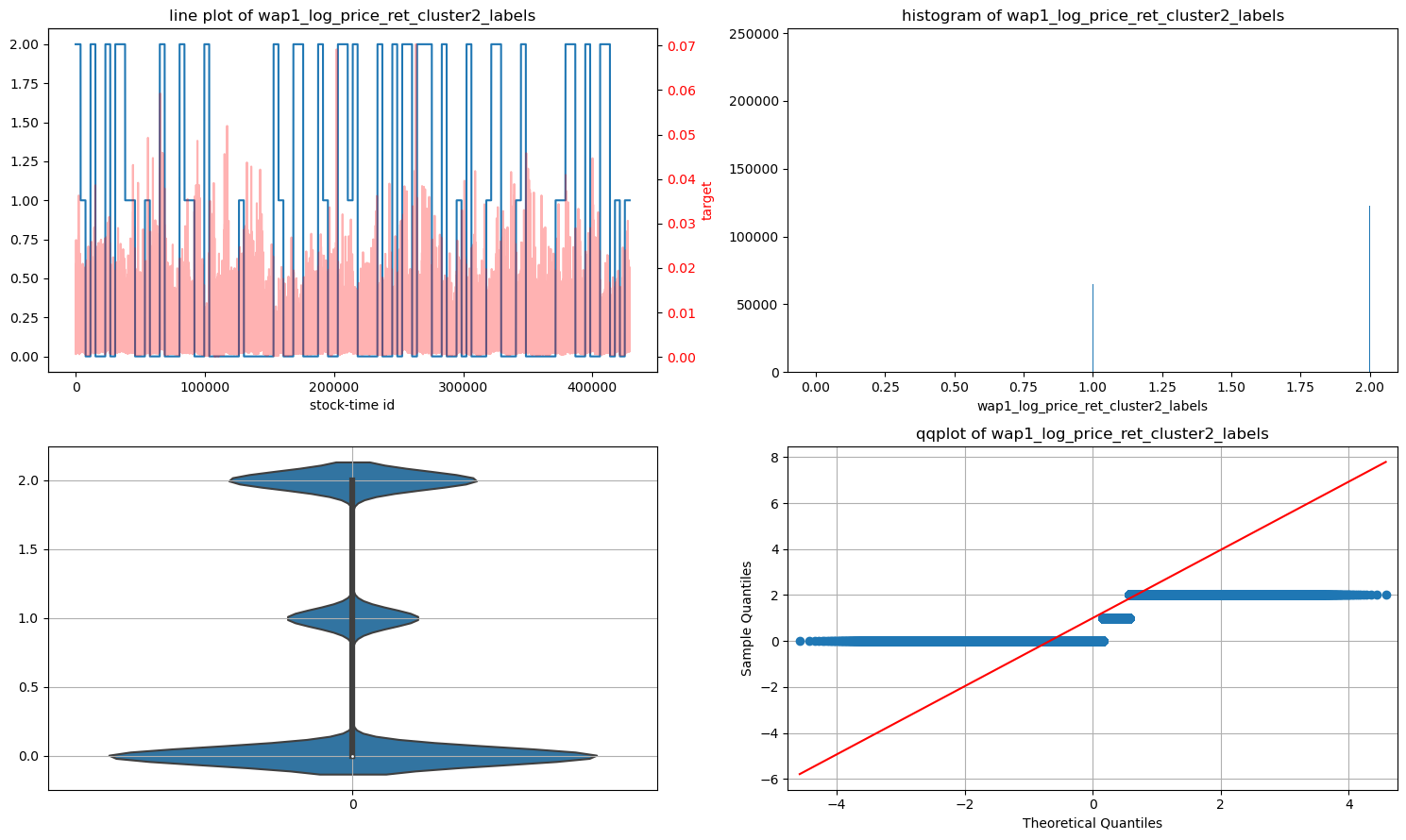
Very few Ourliers lie very far away from bulk of data.



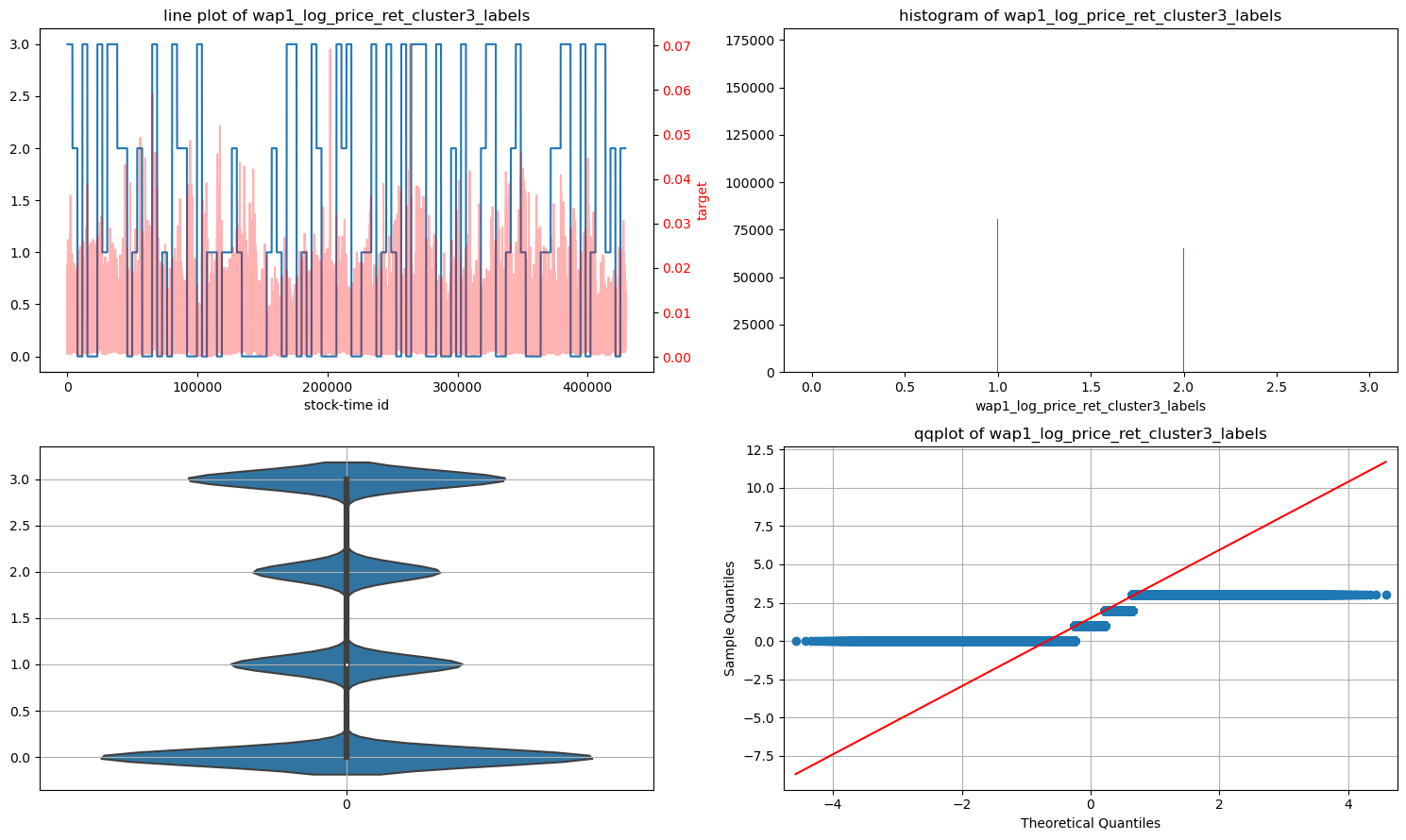
1. Mean\_half\_delta\_lsprd: Similar to above, num of unique values: 112 (DISCARD)



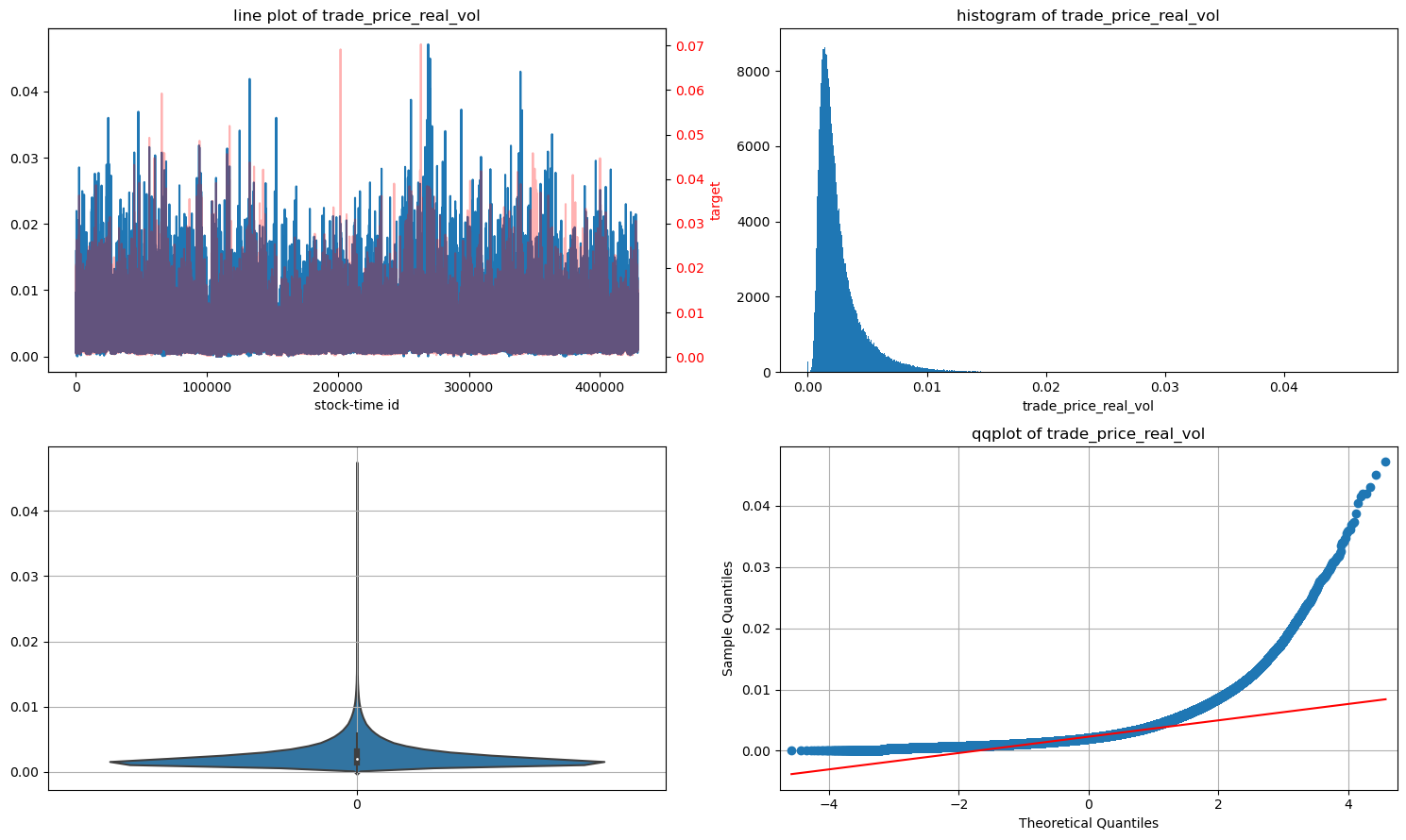
1. Log\_wap1\_log\_price\_ret\_vol: log of volatility in wap1 log price returns.
2. Wap1\_log\_price\_ret\_cluster2\_labels: CATEGORICAL feature.



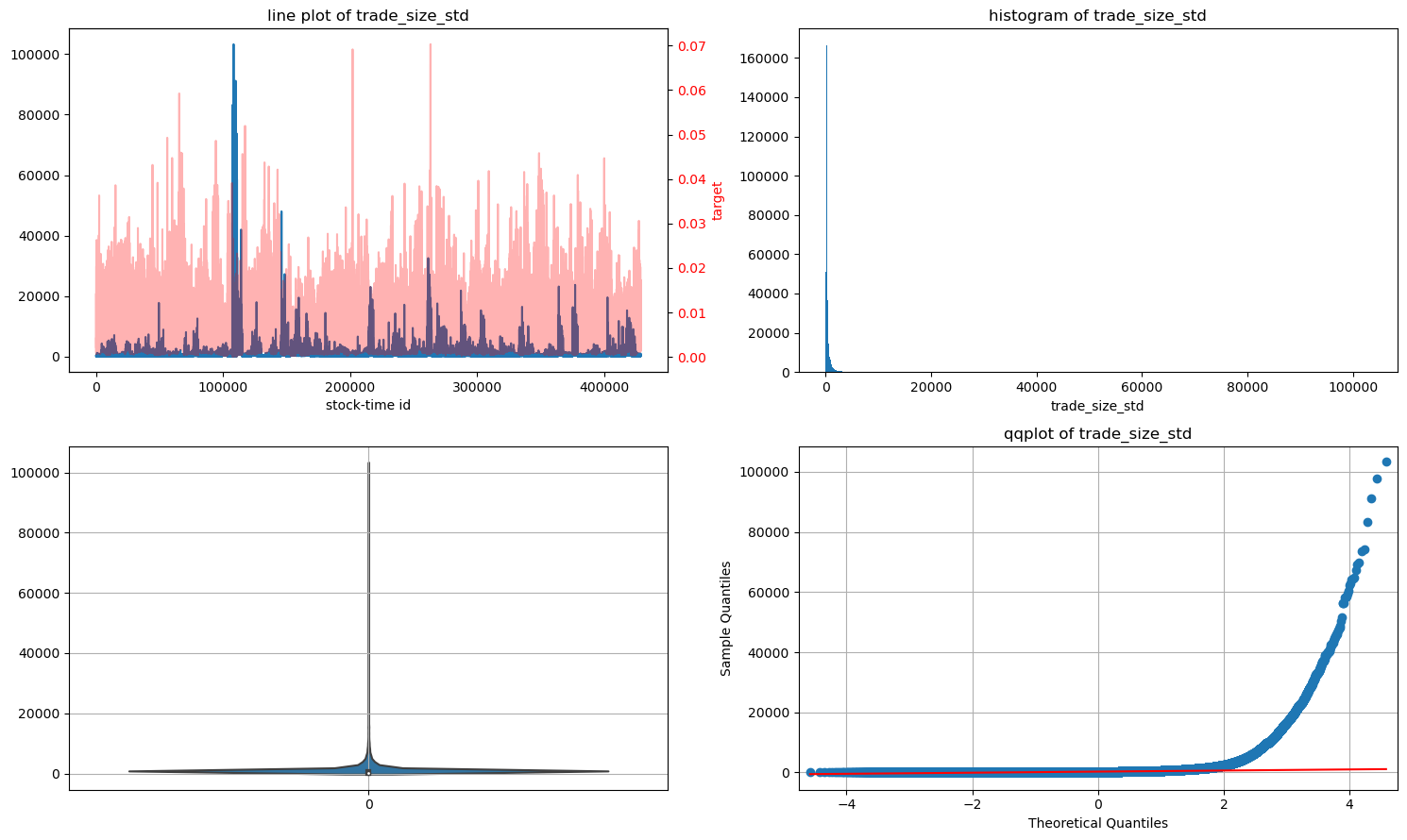
1. Wap1\_log\_price\_ret\_cluster3\_labels: CATEGORICAL feature.



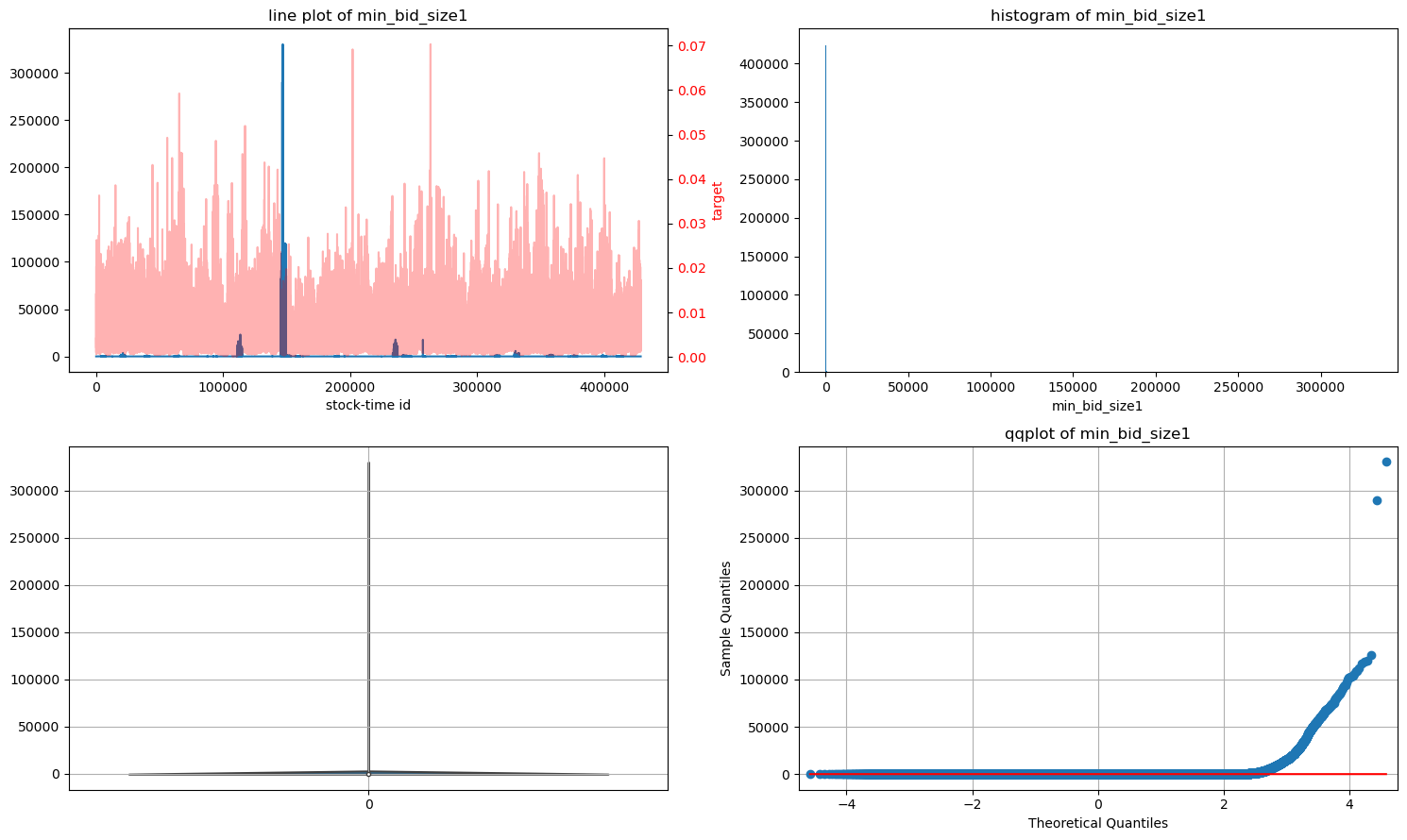
1. Target:
2. Log\_target:
3. Log\_target\_standardized:
4. First\_10\_min\_vol: first 10 min volatility calculated from wap1 price alone. Looks very similar to target. It can be log transformed in the future to look more normal from right skew. Log transform required. Yes done.
5. Trade\_price\_std: right skew.
6. Trade\_price\_real\_vol: right skew,



1. Trade\_size\_std: HIghly skewed with a heavy right tail. This has to be log1p transformed!!



1. Trade\_size\_mean: similar case as aboveLog1p transform required.done
2. Trade\_order\_count\_std: SIMilar ot above.Log1p transform required.done
3. Trade\_order\_count\_mean: SIMilar ot above.Log1p transform required. done
4. Target\_vol\_sum\_stats\_4\_clusters: similar to above. Log transform required. done
5. Target\_vol\_sum\_stats\_10\_clusters: similar to above. Log transform required. done
6. Target\_vol\_sum\_stats\_16\_clusters: similar to above. Log transform required. Done
7. Target\_vol\_sum\_stats\_30\_clusters: similar to above. Log transform required. Done
8. Target\_vol\_corr\_32\_clusters: similar to above. Log transform required. Done
9. Target\_vol\_corr\_4\_clusters: similar to above. Log transform required. Done
10. Target\_vol\_corr\_49\_clusters: similar to above. Log transform required. Done
11. target\_vol\_corr\_90\_clusters : similar to above. Log transform required. Done
12. Min\_bid\_price1: no change ok.
13. Max\_bid\_price1: no change ok.
14. min\_ask\_price1: no change
15. Max\_ask\_price1: no change
16. Min\_bid\_size1: similar to above. Log transform required. Done



1. Max\_bid\_size1: similar to above. Log transform required. Done
2. Min\_ask\_size1: similar to above. Log transform required. Done
3. Max\_ask\_size1: similar to above. Log transform required. Done
4. Range\_ask\_price1:
5. Range\_bid\_price1:
6. Range\_ask\_size1: log1p applied
7. Range\_bid\_size1: log1p transformed, log1p\_range\_bid\_size1\_stnd
8. log\_+1e-2\_sad\_ask\_price1\_stnd: ALL Null values
9. log1p\_sad\_ask\_size1\_stnd:
10. log\_+1e-2\_sad\_bid\_price1\_stnd:
11. log1p\_sad\_bid\_size1\_stnd:
12. Bs\_bp\_corr1: exponential of left skew.
13. Bs\_as\_corr1: exponential of left skew.
14. Bs\_ap\_corr1: no change
15. Bp\_as\_corr1: no change
16. 3\_bp\_ap\_corr1: subtracted 0.0001 and then took arctanh and then standardized . this is similar to fishers z transform.
17. As\_ap\_corr1:
18. Min\_price1:
19. Max\_price:
20. Min\_price:
21. Max\_price1:
22. Min\_size1: log transform applied. Still highly skewed can be Discarded if needed
23. Max\_size1: log transform applied. This is good.
24. Min\_order\_count1: highly SkewedDISCARD
25. Max\_order\_count1: LOG TRansfrom it.
26. Range\_price1: fixed,
27. Range\_size1: log1p transform.
28. Range\_order\_count1: log1p transform.
29. Sad\_price1
30. Sad\_size1: log1p transform
31. Sad\_order\_count1: log1p transform
32. Size\_order\_count\_corr1:
33. Book\_ewma\_vol: log transform
34. Log1p\_trade\_ewma\_vol\_stnd:
35. Min\_bid\_price2:
36. Max\_bid\_price2
37. Min\_ask\_price2
38. Max\_ask\_price2:
39. Min\_bid\_size2: CAN be DISCARDED
40. Max\_bid\_size2: LOG Transform
41. Min\_ask\_size2: DISCARDED
42. Max\_ask\_size2: lop transform
43. Range\_ask\_price2:
44. Range\_bid\_price2:
45. Range\_ask\_size2: log transform
46. Sad\_ask\_size2: log transform
47. 'log\_sad\_ask\_size2\_stnd',
48. 'sad\_bid\_price2',
49. 'bs\_bp\_corr2',
50. 'bs\_as\_corr2',
51. 'bs\_ap\_corr2'
52. ‘Bp\_as\_corr2’
53. Bp\_ap\_corr2: subtracted 0.0001 and then took arctanh and then standardized . this is similar to fishers z transform.
54. As\_ap\_corr2:
55. Sum\_stats\_4\_clusters\_labels: categorical feature.
56. 'sum\_stats\_10\_clusters\_labels', : categorical feature.
57. 'sum\_stats\_16\_clusters\_labels',: categorical feature.
58. 'sum\_stats\_30\_clusters\_labels', : categorical feature.
59. 'pear\_corr\_32\_clusters\_labels',: categorical feature.
60. 'Pear\_corr\_4\_clusters\_labels',: categorical feature.
61. 'pear\_corr\_49\_clusters\_labels',: categorical feature.
62. 'pear\_corr\_90\_clusters\_labels': categorical feature.

* **Identify if any columns still have null and infinity values**

**NUmber of Null values**

wap2\_logprice\_ret\_changes\_n\_wap1\_logprice\_ret\_constant\_sqr\_vol\_15\_ratio : 589

wap2\_logprice\_ret\_changes\_n\_wap1\_logprice\_ret\_constant\_sqr\_vol\_15\_ratio\_median\_stock : 54086

trade\_count\_15\_15 : 19

root\_trade\_count\_15\_15 : 19

v1proj\_25\_15\_std : 560

v1proj\_29\_15\_std : 282002

v1proj\_25\_std : 560

v1proj\_29\_std : 51517

v1proj\_25\_c1\_std : 325

v1proj\_25\_c2\_std : 284

v1proj\_25\_c3\_std : 177

v1proj\_25\_c4\_std : 112

v1proj\_25\_c5\_std : 89

v1spprojt15f25\_c1 : 315942

v1spprojt15f25\_c2 : 310962

v1spprojt15f25\_c3 : 303762

v1spprojt15f25\_c4 : 295233

v1spprojt15f25\_vc1 : 314864

v1spprojt15f25\_vc2 : 310913

v1spprojt15f25\_vc3 : 298756

log\_log\_log\_max\_ask\_size1\_stnd\_stnd\_stnd : 29575

**NUmber of Inf values**

wap1\_log\_price\_ret\_per\_liq2\_vol\_15\_ratio : 1

wap1\_log\_price\_ret\_per\_spread\_sqr\_vol\_15\_ratio : 1

wap2\_logprice\_ret\_changes\_n\_wap1\_logprice\_ret\_constant\_sqr\_vol\_15\_ratio : 2831

wap2\_logprice\_ret\_changes\_n\_wap1\_logprice\_ret\_constant\_sqr\_vol : 589

lsvol : 139

trade\_count : 19

root\_trade\_count : 19

root\_trade\_count\_var : 19

trade\_count\_15\_15 : 397

root\_trade\_count\_15\_15 : 397

v1spprojt15f25\_c3 : 126

v1spprojt15f25\_c4 : 85

v1spprojt15f25\_vc1 : 63

v1spprojt15f25\_vc2 : 153

v1spprojt15f25\_vc3 : 133

tvpl2 : 19

tvpl2\_liqf10 : 19

tvpl2\_liqf20 : 19

tvpl2\_liqf29 : 19

wap1\_log\_price\_ret\_vol\_from\_25 : 5

**Ensemble Models based on LGBM**

1. GBRT
2. DART
3. Decision Trees
4. GJR-GARCH

<https://machinelearningmastery.com/information-gain-and-mutual-information/>