optiver

OPTIBOOK

STOCK QUOTING & NEWS FEED PROCESSING

1. INTRODUCTION

In this assignment you will modify an existing stock quoting algorithm to deal with incoming news messages and social media feeds using recently development automated text processing tools in Python.

A basic quoting strategy, also known as a market making strategy, places simultaneous bid and ask limit orders into the order book on an exchange on a continuous basis, and in doing so attempts to capture the bid-ask spread, that is, the difference between the prices of such orders.

Such a quoting strategy is successful if the direction of opposing market orders, those who trade against you, is essentially random. In such cases, it is likely that a buy trade might be followed by a sell trade, or vice versa, leaving you with a zero net position, and returning a profit of the difference in the bid price and the ask price you traded on, the elusive "scalp" trade. This strategy adds value to the market through "adding liquidity", it ensures other market participants always have someone to trade against, you, the market maker. It is called a passive strategy, as through the insertion of our limit orders we show the market our intentions to trade, and wait for other market participants to come to us to affect the trade, if they so desire.

It sounds almost perfect, but in practice, things aren't as straight-forward as might appear. Market prices are not static and there are competing market makers offering the same service. It is therefore crucial to determine the right prices and price offsets to use, and to manage the risk of the outstanding positions. For example, a quoting strategy can fail to be profitable if the opposing market orders are not random but informed. That is, whenever someone buys from you, so when you sell, the market tends to go up, and whenever you buy again, it happens to be just before the market goes down. That's exactly what you don't want! The small profit from the bid-ask spread you quoted can quickly evaporate under such circumstances. This is called adverse selection and is a very real effect. It makes sense, if someone trades, they usually have a reason for doing so.

In this assignment we investigate, based on an example we encountered in our real trading, one of the ways in which changing market circumstances can affect the viability of a market making strategy, and what kinds of modifications to the algorithm that requires to ensure the strategy still has a future.

Good luck!

2. EXISTING QUOTING STRATEGY

You are a market making firm that has been running a simple quoting strategy on five liquid and well-known stocks profitably for several years. So far, this strategy has worked out quite well for you, but recent market developments have caused your strategy's profit to see sudden downswings, become highly volatile, and even turn overall negative at times. Just as you are about to cut your losses and switch off the strategy, one of your researchers points out what may be going on.

Before we get there however, let's first dig into how the current quoting strategy works. A full Python implementation of this strategy can be found in the Jupyter Notebook *Basic Quoter.ipynb*, which runs it for five stocks simultaneously.

2.1 FUNCTIONING OF A BASIC QUOTING STRATEGY WITH POSITIONAL RETREATS

The algorithm described here is one of the most basic forms of a liquidity providing, passive, market making strategy. The goal of this strategy is to provide two prices on a stock, a bid and an ask, or a buy and a sell price, in more common parlance, broadcast to the market our availability to trade on those prices, and wait for others to trade with us. As they do, we hope to find two-way flow, buyers and sellers, that trade with us, so that we can profit from the difference.

In the current implementation we do not have any other products available to hedge our risks with, the stocks are not very correlated, so if there is a big difference in time between when a buyer and a seller trade with us, we run an open position, and associated positional price risk, in the meanwhile. As we don't like risks, to ensure that we are more likely to trade in the direction that reduces our open position instead of increasing it, we skew our prices depending on the position we currently have. A higher set of prices is more likely to trigger others to sell to us, whereas a lower set of prices is more likely to trigger others to buy to us. That's basic supply and demand, and also works out that way in the order book.

STEP 1. Determine mid price

Obtain an estimate of the price of the product we're quoting, in this case a stock. What's a neutral price where we'd neither buy or sell? There are many ways to obtain this. In more advanced strategies we might have an alternative input for this, but here, let's just take it from the market. The products are quite active & liquidly traded, so the average between the best market bid and ask prices, would be a quite good estimate of what the stock is truly worth at this moment. We assume the market prices to be efficient.

Let's take a look at the order book.

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	
	20,00	
	19,90	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

The best bid price is $19{,}80$ and the best ask price is $20{,}20$, therefore the market midpoint is $\frac{(19{,}80+20{,}20)}{2} = 20{,}0$. That is what we will consider the neutral price.

STEP 2. Determine own bid and ask prices

We intend to capture the bid-ask spread, so our bid price needs to be lower than our ask price. But by how much? There are several options to determine this, but the most basic one is to use a "fixed credit", a minimum difference we desire to earn to trade as compared to our determined fair price. In the above order book, using a fixed credit of 10cts would result in inserted bid and ask prices that are better than anything that's currently being quoted in the market by others, so let's do that.

Our bid price will be $20{,}0 - 0{,}10 = 19{,}90$ and our ask price will be $20{,}0 + 0{,}10 = 20{,}10$.

STEP 3. Insert limit orders and wait

We are willing to buy at a price of $19{,}90$ and willing to sell at $20{,}10$. We can't immediately do so, in the present order book the best seller (who we'd be able to directly buy from) has a price of $20{,}20$ and the best buyer (who we'd be able to directly sell to) has a price of $19{,}80$.

That is the idea, this is a passive trading strategy. We're not going to try to directly trade, but rather wait for others to come to us. We do so with so-called limit orders, orders that show our intention and after we insert them become part of the market order book.

Let's insert our orders with a fixed volume of 10 lots.

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	10
	20,00	
10	**19,90**	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

The new market order book includes our own orders (highlighted in yellow). Due to the price-time priority matching of orders, anybody that affects a market trade, would have us as the counterparty. Let the trading commence!

Let's wait 5 seconds to see if something happens.

STEP 4. Observe trades and determine new position

We are now dependent on the other participants on the market to see what happens next. There are several things that might happen, our orders may either go unfilled entirely (nobody traded with us on either side), partially (they traded some amount of lots < 10 on either the bid or the ask or both) or fully (they traded the full 10 lots on both sides). The first and third scenario leave us with an open position of zero, with the crucial difference that in the third scenario we've made a small profit ($10 * (20{,}10 - 19{,}90) = 2{,}0$) whereas in the first scenario we've done nothing.

The second scenario is more interesting, as we may have some degree of locked in profit as in the third scenario, as well as an open position in the stocks. Note that this open position may be long (positive) or short (negative), as it is often possible to sell before you buy on financial markets and on Optibook especially. Let's assume we bought 8 out of the 10 lots on our bid, and sold 5 of the 10 lots on our ask. That results in the following market order book, with the yellow cells still highlighting our own limit orders:

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	5
	20,00	
2	**19,90**	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

We bought 8 lots and sold 5 of them to someone else, so we have an open position in the stock of 3 lots. For the five lots we've managed to turn over directly, we can consider the profit fully locked in: $5 * (20{,}10 - 19{,}90) = 1{,}00$. Nice!

We have an open position of 3 stocks long. That is, we now own 3 of these stocks. We are a market neutral participant without a fundamental opinion of what the fair stock price is, so we don't necessarily want this, it's risky! What if the stock price starts to move!

Well, we can deal with a little bit of risk. On the next iteration of our strategy, that is, in the next 5 seconds, who knows what will happen, maybe we'll sell 3 more stocks than we bought, and we'd end up at zero anyway. So, it's all part of the game. We're certainly not going to try to sell these actively, we'd have to sell them to the next buyer at $19{,}80$, which would lead to a reduced total profit of $5 * (20{,}10 - 19{,}90) + 3 * (19{,}80 - 19{,}90) = 0{,}70$. We don't like wasting money like that.

STEP 5. Pull the quotes

Pulling quotes, deleting orders, removing orders, etc., are all different terms for removing any outstanding limit orders that we've previously inserted but have not traded yet from the market. Here we'll do so in preparation of the next iteration of our strategy, starting with a clean slate again at STEP 1.

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	
	20,00	
	19,90	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

Our orders are gone, and we are ready to insert new ones on the next iteration.

STEP 6. Repeat but modify STEP 1 for open position

We may now repeat the entire strategy: insert 10 lots at fair prices, wait for trades for a while, and hopefully the trades that we do balance out somewhat. Ideally, they also balance out against whatever imbalance / open position we had remaining from our previous iteration(s).

We van force this issue a little, given we don't like to hold a position, and we currently have a long (positive) position, wouldn't it be nice if we could make it more likely to sell on the next iteration, than to buy?

We can do so via a price retreat, an adjustment to the prices to make trading in the desired direction more likely. Looking at the order book again, which we assume for the example has remained unchanged otherwise, the midpoint is still $20{,}0$. If we want to make it more likely to sell, we need to insert a lower ask and if we want to make it less likely to buy we need to insert a lower bid price. Again, this is just supply and demand. Both of these actions make it more likely that we will reduce our position on this iteration than to increase it. So let's adjust the fair price to $19{,}90$, a positional price retreat of $0{,}10$.

If we use the same required credit as before, 10 cents, this gives us bid and ask prices of $19{,}80$ and $20{,}00$ respectively.

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	
	20,00	(10)
	19,90	
(10) 500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

Can you see the ways in which this makes it more likely for us to sell and reduce the open position? Our ask price is quite aggressive compared to the best market buying and selling prices, it stands out as very low. It may trade quickly indeed. Our bid price now enters the queue behind the best market bid of 500 lots. Based on the price-time priority mechanic used on almost all exchanges, if the price is the same, the order matching happens on first-come-first-serve basis. There would need to be a lot of selling activity on the market, selling all 500 stocks that were already being bid for there, before we'd finally get a trade. So seeing a buy trade on this iteration is not very likely.

Now, the retreat of 10 cents for a small position of only 3 outstanding lots is would be considered quite aggressive, it is used here for the example only. Choosing the right retreat to use is a balancing act. In the implemented algorithm in the Jupyter notebook, the initial setting is 0.5ct for every lot held, which should work out pretty well.

Note of course, that for short (negative) positions, we do the exact opposite, we raise the prices to encourage getting more buys.

3. THE PROBLEM

So far, the above strategy has worked out gloriously, it's made market prices more efficient, added liquidity to the order book, did not carry any large position risks, and returned a slow and steady profit every day.

However, recently, we've suddenly started to see much worse PnLs associated with the strategy, even turning negative at times. Just as we were ready to give up, a bright researcher pointed out an insight: "the strategy losses always come in quick bursts, and happen around large news events about the company!"

After scouring the internet for sources, we've determined that markets are starting to price in the effect of social media feeds/shares and news on an almost instantaneous basis. Some of the published articles and headlines trigger a direct and large move in the underlying stock price, which is persistent.

How would a big stock move on a news item cause a loss in our quoting strategy?

STEP 1. Insert quotes

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	10
	20,00	
10	**19,90**	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

STEP 2. A news item is published

It is very negative news for the company involved, markets respond immediately and market sentiment on the stock goes negative. It's a seller's world. Immediately our 10 lots bid at $19{,}90$ trade, then the 500 lots at $19{,}80$, then the 2000 at $19{,}70$, then the 2000 at $19{,}60$, …. Where will it end!?

Stock Order Book		
# Bid	Price	# Ask
	20,10	(10) 2000
	20,00	2000
	19,90	2000
	19,80	2000
	19,70	2000
	19,60	2000
	19,50	2000
	19,40	2000
	19,30	2000
	19,20	2000
	19,10	2000
	19,00	500
	18,90	50
	18,80	
50	**18,70**	
250	**18,60**	

STEP 3. Oops!

Our ask order at $20{,}10$ remains but is completely unlikely to trade. We are long 10 lots, which we need to sell, eventually. The move is persistent, so waiting won't help us, this order book is the new reality. Hoping for the stock price to move back up is wishful thinking at best and gambling at worst. The company is simply worth less. In jargon: "we carried our long position through the downtick".

Even if we sell the lots of our open position on the next iteration passively, let's be generous and say that we do so at a price of $18{,}90$, that's a net loss of $10 * (18{,}90 - 19{,}90) = -10{,}00$. Oops indeed. That loss is a lot bigger than the small profits we were making through scalps during normal operations. If these types of moves are frequent enough, and large enough, they might completely evaporate the gains that our strategy normally makes.

4. INCORPORATING NEWS MESSAGES

Our researchers have identified the important news streams that seem to be affecting stock prices and connected them to our Python API through the function *exchange.poll_new_social_media_feeds()* which returns a list of all new relevant news items published since the last time the function was called.

4.1 A BANDAID: STABILIZING THE STRATEGY

First things first. We're losing a lot of money on news effects interacting with the stock quoting strategy and PnLs are going all over the place. That is not good. Based on our new news feed, can we find an initial fix that'll help alleviate that problem?

After you understand the functioning of the core algorithm in the Jupyter Notebook *Basic Quoter.ipynb*, run it, observe it for ten to fifteen minutes or so and try to understand its actions using the reference above. Add additional logging if it aids your understanding. Especially observe that the strategy is indeed losing money currently on large impactful news events.

Explore the *exchange.poll_new_social_media_feeds()* function in the notebook *Social Media Feeds API.ipynb*. Do you see the news items coming in? Not all messages seem equally relevant to the stocks we quote, but let's forget about that for now.

Our quoting strategy is still fine under most circumstances, but loses money whenever a large news item hits. Let's stabilize this.

1. Get out of the way. Use the social media feed functionality such that on any new news item, you pull all outstanding quotes (delete outstanding orders) to avoid further bad trades like in the scenario under Section 3.

2. (Optionally) After getting the quotes out of the way, neutralize any open positions in products. You can do so by quickly inserting an opposing IOC order against the best available market price. E.g. in the below order book, with an open position for a stock of 3 lots long (positive), quickly sell using an IOC sell (ask) order for 3 lots at the price of $19{,}80$. If we're quick enough, we should have a position of zero lots during the move and avoid a large positional loss. Note that if we were short instead, we should use an IOC buy (bid) order at the price of $20{,}20$.

   There is a cost in trading out of our position this way, but that's nothing compared to the large risk we're avoiding.

3. Wait for some time for markets to stabilize before re-entering the market (suggested: 15 to 20 seconds).

Stock Order Book		
# Bid	Price	# Ask
	20,50	2000
	20,40	2000
	20,30	2000
	20,20	500
	20,10	
	20,00	
	19,90	
500	**19,80**	
2000	**19,70**	
2000	**19,60**	
2000	**19,50**	

After implementing these fixes, the strategy should avoid the worst losses on the big events. Timing is quite crucial, so it might happen that we still sometimes get taken out adversely, but at least the problem should be much reduced and we are back in control.

However, this does imply the strategy is out of the market for long periods of time. Has applying these been enough already to make the strategy return to a good profitability overall?

4.2 ANALYZING THE DATA: DETERMINING THE RIGHT STOCK

Responding to all news messages in all five of our instruments is quite inefficient. We are paying a price to cancel our outstanding positions against unfavourable prices, and most of those trades are not necessary, as the news message or social media feed does not even relate to that stock. Staying out of the market is also costly, it means we do not obtain the normal gains that the quoting strategy provides.

A dataset of news messages has been provided in csv format in the file *training.csv*. It shows a set of historical messages and the effects on each of the five stocks we quote.

The next step to improving our strategy is to modify the code of the previous section to only take those actions on relevant stocks.

1. Explore the *training.csv* dataset in the notebook *Dataset Exploration.ipynb* using Pandas to gain an intuitive idea of how the news messages affect the stocks.

2. Walk through the notebook *Classifier Introduction.ipynb* to explore how we might go about converting a text message from the news feeds into an affected stock. Test this out on the large dataset. Is your classification working well?

3. Use the classification code you developed and incorporate it into the quoter strategy, such that you only pull and neutralize positions on the likely affected stocks.

   This requires some sharp decision-making. Given that text to class-label conversion is probabilistic in nature by definition, how sure is sure enough before we decide to take action? You can define these choices based on first-principles reasoning, back of the envelope calculations, trial and error, or any discretionary combination of such.

   Keep in mind that markets are ever-changing and that the end goal is to minimize bad trades, while maximizing good ones.

4.3 IMPROVING THE STRATEGY

We have now mitigated the main risks and are running a pretty efficient algorithm again. It simply ducks the largest risks, and works in the basic way in other occasions. However, now that we have our hands on the news feeds, there appears to be a treasure trove of information the contents we may make further use of.

Continue analyzing the data for patterns, and creatively expand the quoting strategy to make use of the resulting classifications. For example, can you determine the direction of the expected move based on the messages? The expected size of the move? It's reliability and likelihood to trigger a move? What's the optimal amount of time to stay out of the market after a pull?

Besides getting out of the way, you might even actively take a position in the correct direction. And then what do you do with that position afterwards? How do you deal with the uncertainty of your news message evaluation?

As you progress, you can take your development in several directions depending on your preference. Will you focus more on understanding the incoming text messages and improving the chances of making the right conclusions on their impact? Or will you instead focus on expanding the strategy itself, and optimizing the trading actions you take based on the imperfect information that the messages provide?

Optionally, if you are familiar with them, feel free to use other Python packages such as scikit-learn and torch, which come pre-installed in your Python environment. Also feel free to modify the strategy in any other way you see fit.

Be creative, reason well, and try to see how you can come out ahead of the competition.

Good luck!