**Data cleaning notes**

The representation of a zero position in client position data appears inconsistent. If it's ordered by date it's observable that sometimes positions are recorded as having NULL for the quantity, other times they are simply not recorded. (For example, client id = 17, product id =203 is inconsistent about reporting whether a position is null)

This seems to change throughout the data set. It's important that this is consistent, otherwise is makes it harder to track position changes that would indicate trading activity. Products that there is a position in get consistent prices as expected.

To simplify the analysis, I've made the decision to fill all records that represent a NULL quantity with 0.

None of this is perfect, as without trade data the closest I can get to estimating activity is a change in portfolio positions. A better measure of activity would be the number of trades that resulted in that change in position.

Some price data is missing for some instruments on some days. For example 04/05/2020 - 06/05/2020 has missing prices for some instruments e.g. (16,44… )

Making the assumption that potatoes are traded on the same days as financial markets, I looked at fincal for LSE and NYSE for 2020 expecting this to be holiday related. It is the Early May Bank Holiday for LSE on the 04/05/2020 but not for the following days. Other instruments have prices for those days but they could just be trading in a different market / country.

I don’t really have enough information at the moment to infer why prices are missing. I can see it tends to happen in groups of 2-3 days which could indicate a holiday but it’s not definitive. I would need a more comprehensive financial calendar to check other markets. It could just be that the potato product has had trading halted on that day.

I’m going to deal with this by forward filling the prices for a product as that is the last known good price we have. Similar to how if an instrument had trading halted, we would only have the last known good price from before trading halted.

**Data Observations**

* Client 38 is by far the biggest trader in our sample.
* July saw hardly any trades placed, a drastic reduction when compared to the other months. Our top clients did not trade in this month.
* Excluding July our top three active clients are (Jamel Caruso (38), Nakisha Southerland (74), Valeri Burchfield (94))

*I Didn’t get as far here as I would have liked…*

**Part 2**

These questions all boil down the fundamental question - how can I take unstructured data from multiple sources and turn it into something useful we can use to improve trading performance. We could get additional pricing information via Bloomberg messages, market prediction via e-mail from different brokers or a change in market conditions related to a particular product. If we can get access to more information, or process information more efficiently than other parties it’s going to result in an edge. This could be in the form of more efficient price discovery, quicker reactions to what’s going on in the market or a better sense of a price trend for a particular product.

Data from word of mouth and telephone is going to be the most difficult to deal with in an automated way, in the instance the human is the interface and they will need to process that unstructured data into something useful. As suggested a simple web app that allows submission with the minimum amount of information, but with the option of adding additional information would be the best option. This could then be stored as structured data.

Another option is to have an external webform and ask the brokers to submit ideas / market information / prices directly, paying them for the accuracy of the information / predictions / ideas.

For Bloomberg messages and e-mail messages we can take a more automated approach. For example, e-mails can be saved in a relatively structured manner. Although the content of theses e-mails will be inconsistent, e-mails do still have certain characteristics that we could use to add structure, which will add value. For example, there will always be a from e-mail address, usually a subject title and then some content.

We could store it in a JSON format in a NOSQL database that might look something like this:

{

subject: "Balmoral potatoes are blowing up!!",

sender: "John Doe <john.doe@potatokings.com>",

recipient: "Henry <exeuction.trader@ms.com>",

message: "There are rumours that several of the large producers have been hit be a blight and the crops may be threatened. Balmorals have historically been more resistant so we are seeing a surge in interest. Latest price: 202p ",

attachments: [

{name: "balmoral\_price\_history.pdf"},]

}

There are a few things we can do once we have it stored to turn it into something more useful and structured

* Run the message through an e-mail parser that can pick out key terms such as ‘price’.
* Perform sentiment analysis on the message and any attachments – are we getting any strong signals regarding a particular variety of potato?
* Give a weight to this message based off of reliability / quality of the individual or broker. The weightings could initially come from traders (e.g. john is really good, I rate his predictions highly) but would be adjusted over time based upon some historical analysis of how close predictions and price were to the market. Perhaps there is someone who is always wrong that we could use as a negative signal.

After processing we can then then add these to a traditional more structured SQL database.

How can we make this useful?

Once we have successfully given the data some structure, we could we could look to give an aggregate to the traders. For example, perhaps we could have a live updating widget that gives a weighted average of where we think the current price is, based on multiple sources. Any sentiment that is particularly strong or negative could also be highlighted.

Conclusions

This was a task that was very different to what I’m used to so I had to learn as I went along. Generally, I do data manipulation in SQL rather than numpy/pandas.

I spent far too much time on Task1 trying to analyse, clean, shape and visualise the data. I only get as far as starting a skeleton for a flask web app. I probably should have started here where I’m much more confident as a developer.

For the trends and factor questions I’m sure there is some more advanced analysis I could perform once I become more familiar with statsmodels/scikit-learn.

I enjoyed this task, It’s a great take-home project. I look forward to continuing to explore this data.

Thanks