Python for finance and optimization
Logisitic regressions in finance

Download the csv file `trading_data.csv` on the EPI.

Dataset

The CSV contains a table from an artificial bond dealer. This table contains the following columns corresponding to requests from clients:

- `midprice`: the mid-price of the bond at the time of the request.[1]
- `id`: identification of the client. There are four client ids in the table.
- `buy/sell`: side of the request (+1 for a client willing to buy, −1 for a client willing to sell).
- `answeredprice`: the price answered by the dealer to the client as a response to his/her request.
- `deal`: the first 2000 rows contain 1 if the client accepted the offer of the dealer, and 0 otherwise. The last 200 contain `NaN`s.

Plot of few graphs to be sure you understand the dataset.

`sklearn`

1. Use `sklearn` to fit a logistic regression for each client.
2. Find how to get the value of the coefficients.
3. Can we tier clients in two categories?
4. Try to predict the probability of a deal for the last 200 rows of the dataset.

`statsmodels`

1. Use `statsmodels` to fit a logistic regression (with no penalty) for each client.
2. Do you get the same result? Why? Propose a code with `sklearn` that replicates the result of `statsmodels`.
3. How can you use a more complex regression to assess the quality of the tiering?

Gradient ascent

1. Code by yourself a gradient ascent to obtain the coefficients given above.

[1] Nobody can trade at that price but it evaluates the current price based on consensus data.