Improving Bond Trading Workflows by Learning to Rank RFQs

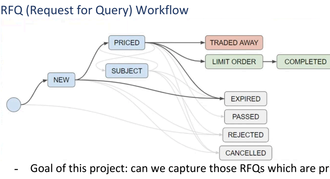
10/1/21 4:17 PM

Speaker: Andy Almonte (Bloomberg AI Engineering)
Date: 09/17/2021

Overview: Stock Market vs. Fixed Income Market

- *Stock market*
  - Trades on centralized exchanges (e.g. NYSE, NASDAQ, London Stock Exchange)
  - Strict trade reporting requirements
  - Higher volatility
  - Largely automated
- *Fixed income market*
  - Trades over-the-counter, largely through major large banks (sell side)
  - Looser trade reporting requirements
  - Lower volatility
  - Human effort required to facilitate trade

RFQ (Request for Query) Workflow



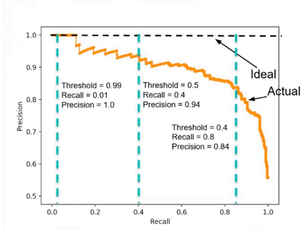
- Goal of this project: can we capture those RFQs which are priced by trader?

Machine Learning Problem Statement

- Task
  - Rank RFQs in an useful way
- Approach
  - Rank by the probability that an RFQ will be priced. This transforms our problem into <u>binary classification</u>
- Implementation
  - Use a Random Forest model to predict the probability that an RFQ will be priced based on its characteristics (amount, price, side, etc.) trained on RFQs from the previous months. We retrain periodically.

Binary Classification Metrics

- Precision: among all selected, how many are relevant?
- Recall: among all relevant, how many are selected?
  - High threshold (fewer selection): higher precision, lower recall
  - Low threshold (more selection): lower precision, higher recall
- Area under Precision-Recall Curve (AUPRC)
  - Measures how well the model performs across all possible confidence thresholds



Ranking metrics

1. *Precision @ N*

$$Precision@N = \frac{P_N}{N}$$

P_N = True positives in N highest ranked samples

- Problem here: choosing of different N value can give very different results, and sometimes even misleading. Two examples:

Example 1
Precision@10 = 0.6

ML Score	Outcome
0.67	PRICED
0.52	PRICED
0.49	PRICED
0.43	PRICED
0.31	NOT PRICED
0.19	PRICED
0.09	NOT PRICED
0.05	NOT PRICED
0.03	PRICED
0.03	NOT PRICED
0.02	NOT PRICED
0.01	PRICED
0.00	NOT PRICED

Example 2
Precision@10 = 0.3

ML Score	Outcome
0.98	PRICED
0.92	PRICED
0.71	PRICED
0.10	NOT PRICED
0.09	NOT PRICED
0.09	NOT PRICED
0.09	NOT PRICED
0.05	NOT PRICED
0.04	NOT PRICED
0.02	NOT PRICED
0.02	NOT PRICED
0.00	NOT PRICED
0.00	NOT PRICED

2. *Discounted Cumulative Gain @ N (DCG@N)*

$$DCG_{@N} = \sum_{i=1}^{N} \frac{(Scored\ Correctly)}{\log_2(i+1)}$$

Higher ranked orders matter more

Example 1

ML Score	Outcome
0.67	PRICED
0.52	PRICED
0.49	PRICED
0.43	PRICED
0.31	NOT PRICED
0.19	PRICED

Precision@5 = 0.8
DCG@5 = 2.56

Example 2

ML Score	Outcome
0.78	NOT PRICED
0.61	PRICED
0.42	PRICED
0.39	PRICED
0.35	PRICED
0.21	PRICED

Precision@5 = 0.8
DCG@5 = 1.95

- Problem here: hard to interpret

3. *Normalized DCG (NDCG@N)*

$$NDCG@N = \frac{DCG_{@N}}{IDCG_{@N}}$$

$DCG_{@N}$ = DCG for N highest ranked samples

$IDCG_{@N}$ = Best possible DCG for N samples

Example 1
NDCG@10 = 0.97

ML Score	Outcome
0.67	PRICED
0.52	PRICED
0.49	PRICED
0.43	PRICED
0.31	NOT PRICED
0.19	PRICED
0.09	NOT PRICED
0.05	NOT PRICED
0.03	PRICED
0.03	NOT PRICED
0.02	NOT PRICED
0.01	PRICED
0.00	NOT PRICED

Example 2
NDCG@10 = 1.0

ML Score	Outcome
0.98	PRICED
0.92	PRICED
0.71	PRICED
0.10	NOT PRICED
0.09	NOT PRICED
0.09	NOT PRICED
0.09	NOT PRICED
0.05	NOT PRICED
0.04	NOT PRICED
0.02	NOT PRICED
0.02	NOT PRICED
0.00	NOT PRICED
0.00	NOT PRICED

Calibration

- If a group of orders have probability of approximately 20%, then around 20% of them should be priced
- A separate model monotonically transforms the output of random forest output "probability" so that the above statement becomes more correct, while preserving order