EDELWEISS HACKATHON- MACHINE LEARNING

Solution 2 - Predict Price Movement

Author - Anurag Sharma

PROBLEM STATEMENT

- To predict short term (intraday) price movement of various stocks on the basis of its price and combination of multiple features provided.
- Training dataset contains 5-min tick data for a universe of more than 150 stocks. The data has the following features:
  - Mid_Price: The price over which the returns need to be calculated
  - Date and Time
  - 46 anonymized features

TOOLS USED

- Python
- Pandas/Numpy
- Matplotlib
- Sklearn
- XGBoost

SOLUTION - 1

- Read the dataset into pandas dataframe.
- Merged all three files into one file for train and test data.
- Replaced all null values with 999 and infinite values with a large number 99999.
- Created a new datetime feature Date_mod by merging Date and Time column.

SOLUTION - II

- In the next step I did feature engineering.
- I created several new features based on the newly created Date_Mod feature like DayOfMonth, Month, Minute, Hour, WeekDay etc.
- These features were used along with anonymized features in the MidPrice prediction.

MODELING

- For model training, I used XGBoost. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. XGBoost is an algorithm that has recently been dominating applied machine learning and ML competitions for structured or tabular data.

- XGBoost with tuned parameters is able to capture the trends in the time series data.

- After tuning hyper parameters, I got these as the best on this dataset,

    - num_rounds = 900
    - learning rate = 0.01
    - max_depth = 8
    - sub_sample = 0.9
    - colsample_bytree = 0.8

ALGORITHM FOR BUY-SELL TRADING

- Using XGBoost, I predicted MidPrice for each stock.

- After this I implemented an algorithm for Stock trading. In this algorithm I tried to predict the Sell/Buy position for each stock on each day.

- First I sliced the dataframe based on time. As we can take a position only between 9:30 to 15:10 (both included) so I took only those datapoint for predicting Buy/Sell position.

CONTINUE…

- Then I took a window of 5 for each stock and for each window interval I tried to find out the absolute maximum between two MidPrices.
- I used a flag to check whether the absolute difference was due to an uptrend or downtrend. Which means whether the value of A[i] > A[j] or A[i] < A[j] (given i>j and 0 based index is used)
- If A[i] > A[j], then from ‘i’ to ‘j-1’ we take -1(buy) position and the jth position will be 0.
- If A[i] > A[j], then from ‘i’ to ‘j-2’ we take -1(buy) position. For j-1 position we take 1(sell) position and for jth position we take 0 position.
- Same thing we repeat for next window of 5 points and so on.

CONTINUE…

- Let's understand with an example.
- In this example, entry price will be 1500
- Exit price will be 2000
- Position would be 0 - (1) = -1
- So return would be
  (-1) * position * (exit-entry)/entry = 0.33

MIDPRICE	POSITION
1450	0
1500	-1
1600	-1
1700	-1
1800	1
2000	0

CONTINUE…

- Let's understand with another example.
- In this example, entry price will be 5000
- Exit price will be 1000
- Position would be 0 - (-1) = 1
- So return would be
  (-1) * position * (exit-entry)/entry = 4

MIDPRICE	POSITION
1450	0
5000	-1
4000	-1
3000	-1
2000	-1
1000	0

RESULTS

- From this solution I got 8086399.21659 return score on the leaderboard.

"Torture the data, and it will confess to anything.""

-Ronald Coase

THANK YOU