Evolution of limit order book dynamics: One machine learning high frequency trading model

Jian Wang

wangjian790@gmail.com

Financial math Ph.D Candidate
Florida State University

©©©©

6th High Frequency data Conference 2015

Table of contents

Contents

- Our main goal is to use boosting machine learning method to predict the limit order book price cross over opportunity.
- Use the high frequency data to predict relatively long time future price changing trend.
- Features selection: choose what kind of data as our independent variables(choose x_i s).
- Compare the accuracy rate and calculation time among different machine learning methods, and show that the boosting method can improve the predicting performance to some extent.

Contents

High frequency trading

High frequency trading is a specialized case of algorithmic trading involving the frequent turnover of many small positions of a security.

Positive impact

- Increased liquidity
- Narrowing spreads
- Improve market efficiency
- Increase fees for Exchanges

Negative impact

- Impact on the institutional investors.
- Increase volatility
- Disadvantages to the small Investors(asymmetric information)

HFT Strategies:

Market Making

place bets on both sides of the trade by placing a limit order to sell slightly above the current market price, or to buy slightly below the current market price, thereby profiting from the difference between the two.

Statistical Arbitrage

Firms and traders looking to make profits from market arbitrage essentially exploit the momentary inconsistencies in factors such as rates, prices, and other conditions between different exchanges or asset classes

Liquidity Rebate Trading

look for large orders, fill a part of that order, and then offer these shares back to the market by placing a limit order, which makes them eligible to collect the rebate fee for providing liquidity,with or without them making a capital gain.

Momentum Ignition

ignition strategies involve initiating and canceling a number of trades and orders with a certain security in a particular direction, which may ignite a rapid market price movement.

Contents

Dataset

Limit order book data

The dataset contains limit order book prices of specific stock from NASDAQ. For each stock, it divided into two major components: the message book and the order book.

- Message book: Contains Time, Prices, Volume, Event Type, Direction
- Order book: Contains price levels, price and volume in each level for every event.

More details can be found in the following two charts.

Message Book:

Message book

	Time(sec)	Price($)	Volume	Event Type	Direction
$k-1$	34203.011926972	585.68	18	execution	ask
k	34203.011926973	585.69	16	execution	ask
...	...	...	...	...	...
$k+4$	34203.011988208	585.74	18	cancellation	ask
$k+5$	34203.011990228	585.75	4	cancellation	ask
...	...	...	...	...	...
$k+8$	34203.012050158	585.70	66	execution	bid
$k+9$	34203.012287906	585.45	18	submission	bid
k+10	34203.089491920	586.68	18	submission	ask

Time is in sec and minimum time change is nanosecond, Price is in dollars and each tick is one cent, 7 Event type, such as execution, cancellation and so on, 2 Direction ask and bid.

Order Book:

Order book

	Ask^1		Bid^1		Ask^2		Bid^2		Ask^3		Bid^3		...
	Price	Vol.	Price	Vol.	Price	Vol.	Price	Vol.	Price	Vol.	Price	Vol.	...
$k-1$	585.69	16	585.44	167	585.71	118	585.40	50	585.72	2	585.38	22	...
k	585.71	118	585.44	167	585.72	2	585.40	50	585.74	18	585.38	22	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...
$k+4$	585.71	118	585.70	66	585.72	2	585.44	167	585.75	4	585.40	50	...
$k+5$	585.71	118	585.70	66	585.72	2	585.44	167	585.80	100	585.40	50	...
...	...	...	...	...	...	...	...	...	...	...	...	...	...
$k+8$	585.71	100	585.44	167	585.80	100	585.40	50	585.81	100	585.38	22	...
$k+9$	585.71	100	585.45	18	585.80	100	585.44	167	585.81	100	585.40	50	...
$k+10$	585.68	18	585.45	18	585.71	100	585.44	167	585.80	100	585.40	50	...

From level 1 to level 10, where the first level is the best bid and ask.

Contents

Methodology

Logistic regression

$$ln\frac{F(x)}{1-F(x)} = \beta_0 + \sum_i \beta_i x_i$$

Ridge regression

$$\hat{\beta}^{ridge} = argmin_\beta \{\sum_{i=1}^{p} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} \beta_j^2\}$$

Lasso regression

$$\hat{\beta}^{lasso} = argmin_\beta \{\sum_{i=1}^{p} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p} |\beta_j|\}$$

Methodology

Comparison of L1 and L2 Penalized Model

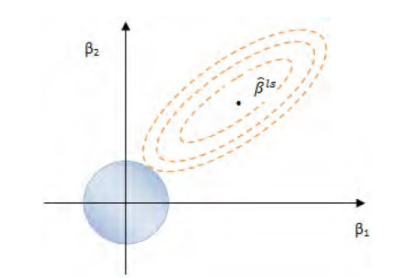
Ridge regression

$\hat{\beta}^{ridge} = argmin_{\beta}\{\sum_{i=1}^{p}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}\beta_j^2\}$

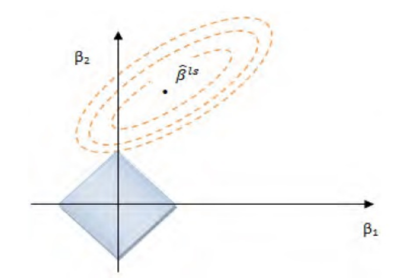
Lasso regression

$\hat{\beta}^{lasso} = argmin_{\beta}\{\sum_{i=1}^{p}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}|\beta_j|\}$

Coefficients:



Coefficients:



Methodology

Comparison of L1 and L2 Penalized Model

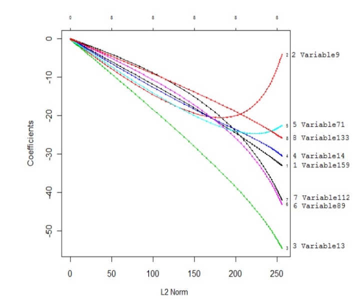
Ridge regression

$$\hat{\beta}^{ridge} = argmin_{\beta}\{\sum_{i=1}^{p}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}\beta_j^2\}$$

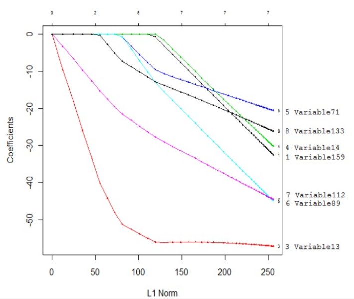
Lasso regression

$$\hat{\beta}^{lasso} = argmin_{\beta}\{\sum_{i=1}^{p}(y_i - \hat{y}_i)^2 + \lambda\sum_{j=1}^{p}|\beta_j|\}$$

Path::



Path::



Methodology

Support vector machine

- Introduced in COLT-92 by Boser, Guyon & Vapnik. Became rather popular since.
- Theoretically well motivated algorithm: developed from Statistical Learning Theory (Vapnik & Chervonenkis) since the 60s.
- Empirically good performance: successful applications in many fields (bioinformatics, text, image recognition, . . .)

Try to maximize the margin:

$r = 1/||w||, y_j = 1, -1$

primal form:

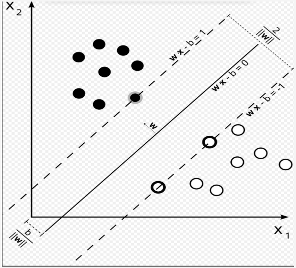
$$\max_{W,b} r = 1/||W||$$

$$s.t.(W^T x_j + b)y_j >= 1$$

Dual form:

$$\max_{\alpha_1,\dots,\alpha_M} \sum \alpha_l - \frac{1}{2}\sum_{j=1}^{M}\sum_{k=1}^{M} \alpha_j\alpha_k y_j y_k < X_j, X_k >$$

$$\text{s.t.}\alpha_l \geq 0, \sum_{l=1}^{M} \alpha_l y_l = 0$$



Methodology

Kernel functions

We can use the kernel function to calculate the inner product in high dimensional cases in its original feature spaces.

Example:two dimension polinomial

$$k(x, z) = (x^T z)^2$$
$$= (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T (z_1^2, \sqrt{2} z_1 z_2, z_2^2)$$
$$= \Phi(x)^T \Phi(z)$$

Kernel functions that we used

- Linear kernel: $k(x, y) = x^T y + c$
- Polynomial Kernel: $k(x, y) = (\alpha x^T y + c)^d$
- Radial basis function kernel(RBF): $k(x, y) = exp(-\gamma ||x - y||^2)$

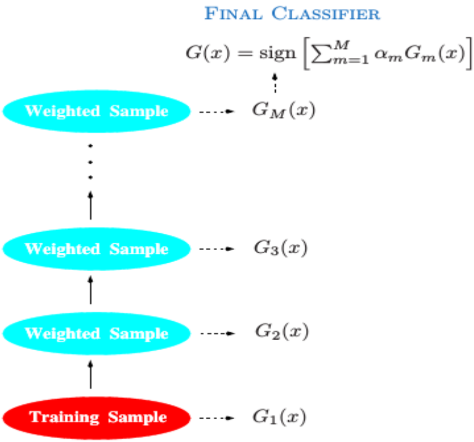
Methodology

Boosting methods

- Introduced in 1990s
- Originally designed for classification problems
- Later extended to regression
- Motivation - a procedure that combines the outputs of many "weak" classifiers to produce a powerful "committee"

Methodology

Boosting methods



Methodology

Adaboosting algorithm:

1. Initialize the observation weights $w_i = 1/N,\ i = 1, 2, \ldots, N$.
2. For $m = 1$ to M:
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
   $$\mathrm{err}_m = \frac{\sum_{i=1}^N w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^N w_i}.$$
   (c) Compute $\alpha_m = \log((1 - \mathrm{err}_m)/\mathrm{err}_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))],\ i = 1, 2, \ldots, N$.
3. Output $G(x) = \mathrm{sign}\left[\sum_{m=1}^M \alpha_m G_m(x)\right]$.

source:ESL

- Put more weights on the false classification data

Gradient tree boosting algorithm:

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^N L(y_i, \gamma)$.
2. For $m = 1$ to M:
   (a) For $i = 1, 2, \ldots, N$ compute
   $$r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{m-1}}.$$
   (b) Fit a regression tree to the targets r_{im} giving terminal regions $R_{jm},\ j = 1, 2, \ldots, J_m$.
   (c) For $j = 1, 2, \ldots, J_m$ compute
   $$\gamma_{jm} = \arg\min_\gamma \sum_{x_i \in R_{jm}} L\left(y_i, f_{m-1}(x_i) + \gamma\right).$$
   (d) Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$.
3. Output $\hat{f}(x) = f_M(x)$.

source:ESL

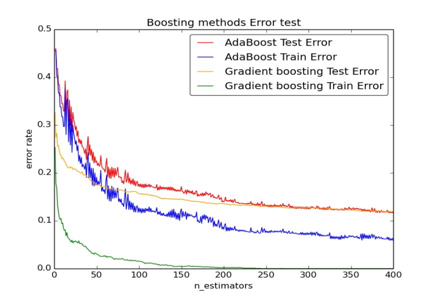
- Use gradient descent methods to minimize the residual in each step

Boosting methods error rate evolution

Boosting method can dramatically increase the performance of even a very weak classifier.We further implement the figure 10.2 in the ESL for example. Suppose features $X_1, X_2, ..., X_10$ are standard independent Gaussian, and the deterministic target Y is defined by:

$$Y = \begin{cases} 1 & \textit{if } \sum_{j=1}^{10} X_j^2 > \chi_{10}^2(0.5) \\ -1 & \textit{otherwise} \end{cases}$$

where $\chi_{10}^2(0.5)$ is the median of a chi square random variable with 10 degrees of freedom.

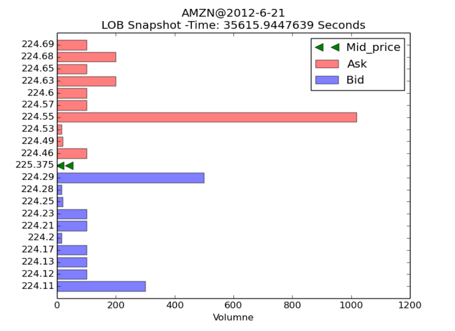


- Boosting methods can reduce the prediction error rate to around one third of the original.
- In this case, Gradient boosting method performance better

Contents

Model built

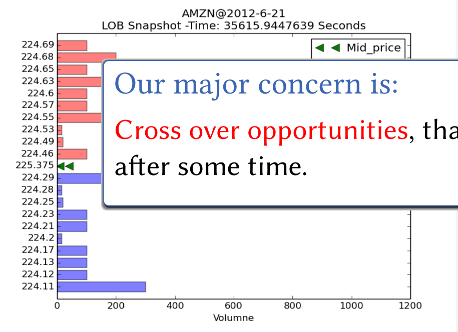
order book snapshot:



- At Time t: $P_t^A > P_t^B$, no arbitrage
- At Time t+ Δt, there are three situations:
  •$P_{t+\Delta t}^A < P_t^B$: ask lower,denote as 1 in our model
  •$P_{t+\Delta t}^B > P_t^A$: bid higher, denote as -1 in our model
  •otherwise(implies thatno direction change)

Model built

order book snapshot:



- At Time t: $P_t^A > P_t^B$, no arbitrage
- At Time t+ Δt, there are three

enote as

denote

as -1 in our model

•otherwise(implies thatno direction change)

Our major concern is:

Cross over opportunities, that is bid higher or ask lower after some time.

Build features:

Basic Set	Description(i = *level index*, $n = 10$)
$v_1 = \{P_i^{ask}, V_i^{ask}, P_i^{bid}, V_i^{bid}\}_{i=1}^n,$	*price and volume (n levels)*

Time-insensitive Set	Description(i = *level index*)
$v_2 = \{(P_i^{ask} - P_i^{bid}), (P_i^{ask} + P_i^{bid})/2\}_{i=1}^n,$	*bid-ask spreads and mid-prices*
$v_3 = \{P_n^{ask} - P_1^{ask}, P_1^{bid} - P_n^{bid}, \lvert P_{i+1}^{ask} - P_i^{ask}\rvert, \lvert P_{i+1}^{bid} - P_i^{bid}\rvert\}_{i=1}^n,$	*price differences*
$v_4 = \{\frac{1}{n}\sum_{i=1}^n P_i^{ask}, \frac{1}{n}\sum_{i=1}^n P_i^{bid}, \frac{1}{n}\sum_{i=1}^n V_i^{ask}, \frac{1}{n}\sum_{i=1}^n V_i^{bid}\},$	*mean prices and volumes*
$v_5 = \{\sum_{i=1}^n (P_i^{ask} - P_i^{bid}), \sum_{i=1}^n (V_i^{ask} - V_i^{bid})\},$	*accumulated differences*

- contain price,volume, bid ask spread, price difference and volume difference for each level, mean of price and volume.
- total 86 variables, can be treated as high dimensional problems.

Numerical results:

AAPL Predict(100 events):

Table : AAPL Accuracy rate and CPU time

Methods	Accuracy rate	CPU time
Logistic	57.15%	0.06
Ridge(alpha=1)	59.83%	0.01
Lasso(alpha=0.001)	60.75%	16.54
SVM	60.15%	4.04
Decision tree	51.40%	0.24
Ada Boosting Tree	72.90%	30.19
Gradient Boosting Tree	71.95%	14.53

remark: training samples 8000 and test samples 2000. The parameter for adaboosting and gradient boosting are: depth =3 and iterations =500.Computer is 8G memory and Intel Xeon E3 processor(4 cores)

Numerical results:

AMZN Predict(100 events):

Table : AMZN Accuracy rate and CPU time

Methods	Accuracy rate	CPU time
Logistic	66.97%	1.75
Ridge(alpha=1)	64.32%	0.01
Lasso(alpha=0.001)	65.60%	16.60
SVM	64.54%	6.19
Decision tree	56.01%	0.26
Ada Boosting Tree	74.03%	29.40
Gradient Boosting Tree	77.23%	14.41

remark: training samples 8000 and test samples 2000. The parameter for adaboosting and gradient boosting are: depth =3 and iterations =500. Computer is 8G memory and Intel Xeon E3 processor(4 cores)

Contents

Future work

- Use high performance method such as parallel computing to improve the speed.
- Add more meaning features.
- Conduct cross validation or bagging methods.
- Test the Profit and Loss(PNL) which traders mainly concern.
- Try to publish in our Quantitative Finance :P

Contents

QA

Thanks a lot and Questions