Bayesian Regression and Bitcoin

https://arxiv.org/abs/1410.1231 (https://arxiv.org/abs/1410.1231)

Seoul AI Meetup, September 2017

Cinyoung Hur, cinyoung.hur@gmail.com

Bayesian Regression

The Problem

- predict the unknown label $y \in \mathbb{R}$ for given $x \in \mathbb{R}^d$
- train n labeled data points
  - $(x_i, y_i), 1 \leq i \leq n$
  - $x_i \in \mathbb{R}^d$
  - $y_i \in \mathbb{R}$

The classical approach

- function approximation for $y = f(x) + \varepsilon$
  - $f(x) = x^T\theta^*$
  - ε : noise(an independent random variable)

- least-squares estimation for θ^* or f
  - assume $n \gg d$, d is fixed

$$\hat{\theta}_{LS} \in \underset{\theta \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - x_i^T\theta)^2$$

$n \ll d$ and Sparsity

The practical appoach

- regularized least-square estimation for θ^* or f
  - In reality,
    - $n \asymp d$
    - $n \ll d$
    - $\|\theta^*\|_0 \ll d$, $\|\theta^*\|_0 = \left|\{i : \theta_i^* \neq 0\}\right|$
  - for appropriate choice of $\lambda > 0$

$$\hat{\theta}_{LASSO} \in \underset{\theta \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{i=1}^{n} (y_i - x_i^T\theta)^2 + \lambda\|\theta\|_1$$

Choosing a reasonable parametric function space θ^* or f is hard!

- data is very high dimensioanl(e.g. time-series)
- parametric space is too complicated or meaningless

Latent source model

- capture that underlying events exhibits itself
- K distinct latent sources
  - $s_1, \ldots, s_K \in \mathbb{R}^d$
- a latent distribution over $\{1, \ldots, K\}$ with associated probabilities
  - $\{\mu_1, \ldots, \mu_K\}$
- K latent distributions over $\mathbb{R}$
  - $P_1, \ldots, P_K$

Latent source model

- data point (x, y) is generated as follows
  - $x = s_T + \varepsilon$
    - $T \in \{1, \ldots, K\}$ with $P(T = k) = \mu_k$ for $1 \leq k \leq K$
  - $y \in \mathbb{R}$ as per distribution P_T

Regression → Simple Bayesian Inference

- conditional distribution

$$\begin{aligned} P(y|x) &= \sum_{k=1}^{T} P(y|x, T = k)P(T = k|x) \\ &\propto \sum_{k=1}^{T} P(y|x, T = k)P(x|T = k)P(T = k) \\ &= \sum_{k=1}^{T} P_k(y)P(\varepsilon = (x - s_k))\mu_k \\ &= \sum_{k=1}^{T} P_k(y) \exp(-\frac{1}{2}\|x - s_k\|_2^2)\mu_k \end{aligned}$$

Lack of knowledge

- 'latent' parameters of the source model
  - sources: $(s_1, \ldots, s_K)$
  - probabilities: $(\mu_1, \ldots, \mu_K)$
  - probability distribution: $P_1, \ldots, P_K$

Use empirical data as proxy

- empirical conditional probability
$$P_{emp}(y|x) = \frac{\sum_{i=1}^{n} \mathbb{1}(y = y_i) \exp(-\frac{1}{4}\|x - x_i\|_2^2)}{\sum_{i=1}^{n} \exp(-\frac{1}{4}\|x - x_i\|_2^2)}$$
- conditional expectation of y, given x
$$\mathbb{E}_{emp}[y|x] = \frac{\sum_{i=1}^{n} y_i \exp(-\frac{1}{4}\|x - x_i\|_2^2)}{\sum_{i=1}^{n} \exp(-\frac{1}{4}\|x - x_i\|_2^2)}$$

- in binary classification, y takes values in $\{0, 1\}$
- classifiation rule: compute ratio
  - if ratio is > 1, $y = 1$, else $y = 0$ $\frac{P_{emp}(y=1|x)}{P_{emp}(y=0|x)} = \frac{\sum_{i=1}^{n} \mathbb{1}(y=1) \exp(-\frac{1}{4}\|x-x_i\|_2^2)}{\sum_{i=1}^{n} \mathbb{1}(y=0) \exp(-\frac{1}{4}\|x-x_i\|_2^2)}$

- 'linear' estimator:
  - $X(x) \in \mathbb{R}^n$
  - $X(x)_i = \exp(-\frac{1}{4}\|x - x_i\|_2^2)/Z(x)$
  - $Z(x) = \sum_{i=1}^{n} \exp(-\frac{1}{4}\|x - x_i\|_2^2)$

- $y \in \mathbb{R}^n$ with ith component being y_i
  - $\hat{y} = X(x)y$

Trading Bitcoin

Relevance of Latent Source Model

- price movements follow a set of patterns
- one can use past price movements to predict future returns to some extent
- e.g. geometric patterns
  - heads-and-shoulders, triangle and double- top-and-bottom

Data

- 0kcoin.com, China
- 7 months:
  - Feb 2014~ Jul 2014
- 200 million data points
- interval:
  - raw: 2 sec
  - train: 10 sec

Trading Strategy

- position of Bitcoin
  - +1
  - 0
  - -1
- Average price movement over the 10 sec interval
  - Δp
- Bitcoin price threshold
  - t

- if $\Delta p > t$ and position ≤ 0, buy a bitcoin
- elif $\Delta p < -t$ and position ≥ 0, sell a bitcoin
- else, do nothing

Predicting Price Change

Terms

- subsets of time-series data of 3 different lengths:
  - S_1: time-length 30 min
  - S_2: time-length 60 min
  - S_3: time-length 120 min
- previous time-series of different lengths at t:
  - x^1: previous 30 min
  - x^2: previous 60 min
  - x^3: previous 120 min
- at t, predict the future change Δp using these time-series

Predicting Price Change

Bayesian regression

- Δp^j: prediction of cost change using x^j with samples S^j
- $r = \frac{v_{bid} - v_{ask}}{v_{bid} + v_{ask}}$
  - v_{bid} is total volumn of bid in top 60 orders based on the current order book
- $\mathbf{w} = (w_0, \ldots, w_4)$: learnt parameters

$$\Delta p = w_0 + \sum_{j=1}^{3} w_j \Delta p^j + w_4 r$$

How to find S_j, How to learn $\mathbf{w}$

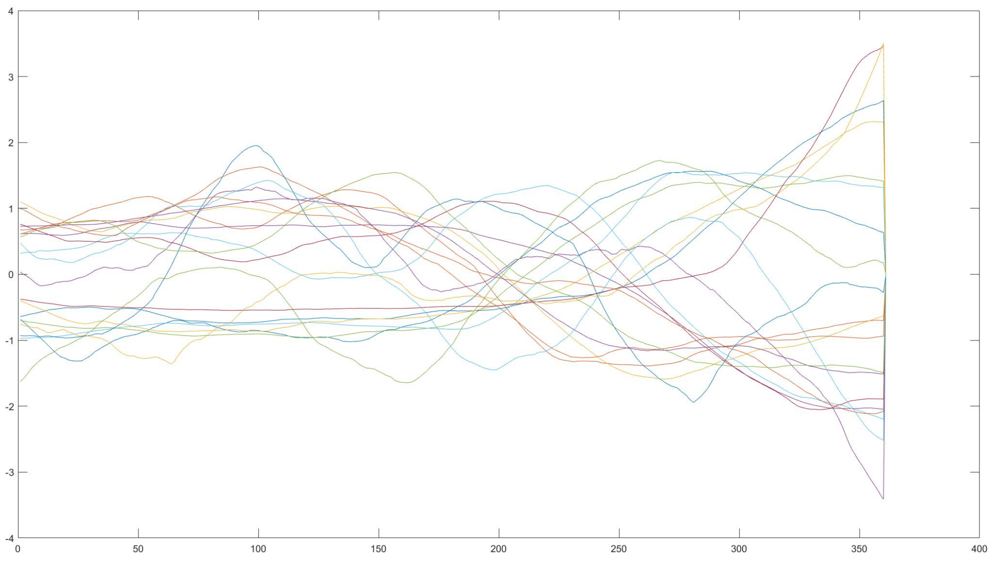
- divide entire 7 months into three periods
- 1st period: find patterns $S_j, 1 \leq j \leq 3$
  - x_i = all possible time series of appropriate length (180, 360, and 720 from S_1, S_2, S_3)
  - y_i = $avg(\Delta p)$ of 10 sec interval at the end of x_i
  - ------------------t
  - |------x_i^1-------|
  - -------|---x_i^2---|
  - -------------|x_i^3|
  - ------------------|y_i|

- pick 20 clusters out of 100 clusters (k-means algo)
  - distance: $\exp(-\|x - x_i\|_2^2/4)$ --> $\exp(c \cdot s(x, x_i))$
    - becuase of computational cost of squared $l_2 - norm$
  - similarity between two vectors a, b

$$s(a, b) = \frac{\sum_{z=1}^{M} (a_z - mean(a))(b_z - mean(b))}{M\ std(a)\ std(b)}$$

  - computing similarity
    - precomputed and normalized S_1, S_2, S_3
    - 1 sec / 10 million vectors similarity

- 20 patterns



How to find S_j, How to learn $\mathbf{w}$

- 2nd period: learn parameter $\mathbf{w}$ having best linear fit over all S_j

$$\Delta p = w_0 + \sum_{j=1}^{3} w_j \Delta p^j + w_4 r$$

Evaluation

- 3rd period: evaluate the performance of the algorithm
  - different threshold t
  - evaluation metric: Sharpe ratio (https://en.wikipedia.org/wiki/Sharpe_ratio)
    - how well the strategy performs compared to the risk-free strategy
    - how consistently it performs
    - Good: greater than 1
    - Very Good: higher than 2
    - Excellent: 3 or higher
- In 50 days, 89% return, a Sharp ratio of 4.10
  - 2,872 trades
  - total profit peaked 3,362 yuan (581,525.14 KRW, 513.81 USD)
  - total average investment 3,781 yuan (653,999.57 KRW, 577.74 USD)

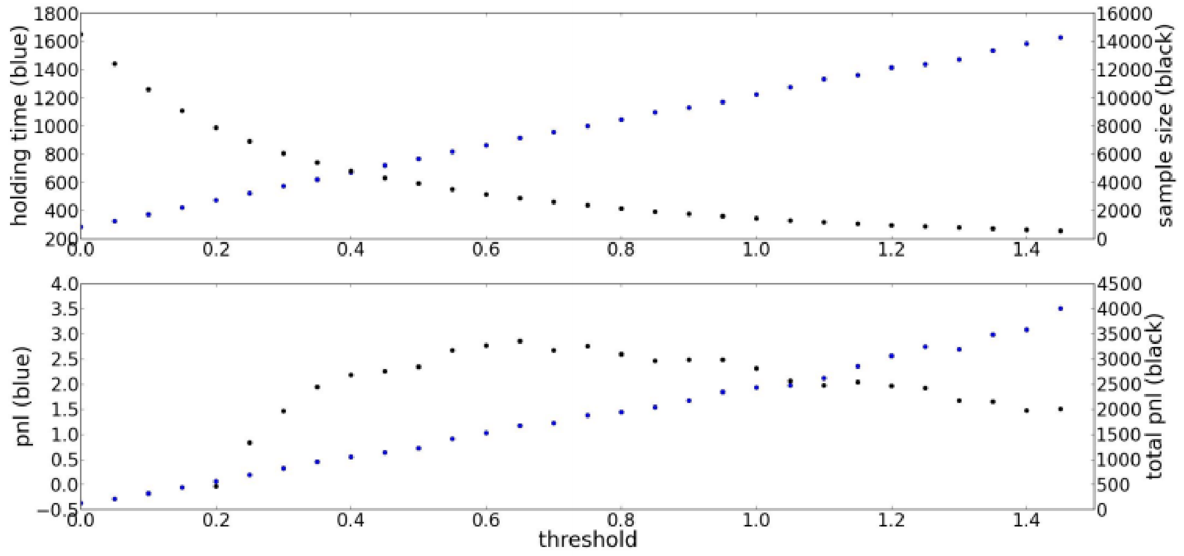


Fig. 1: The effect of different threshold on the number of trades, average holding time and profit

- increase threshold, number of trades decreases, avg holding time increases
- increase threshold, profit&loss increases, total profit&loss saturates

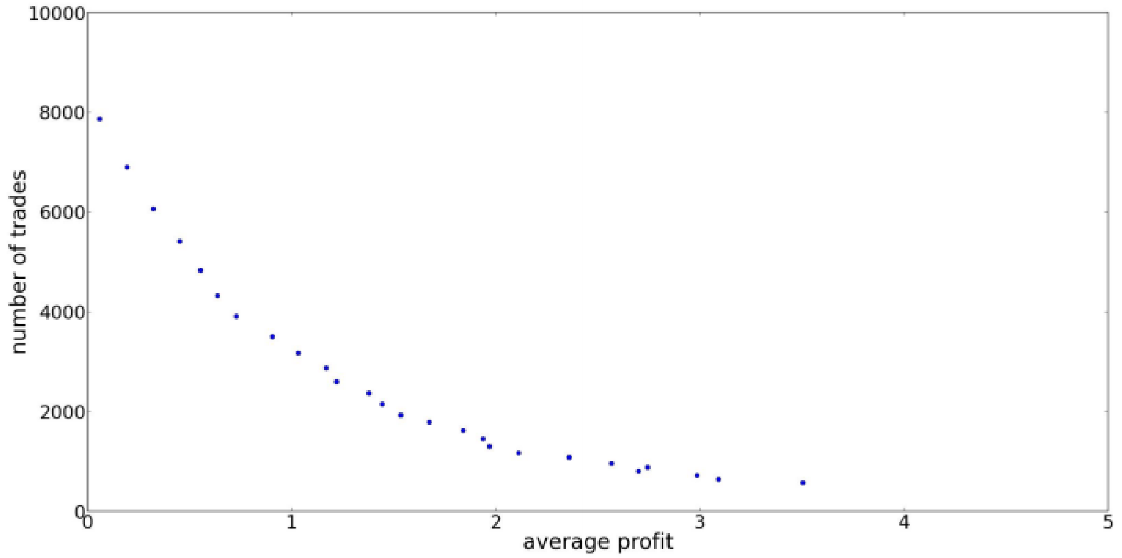
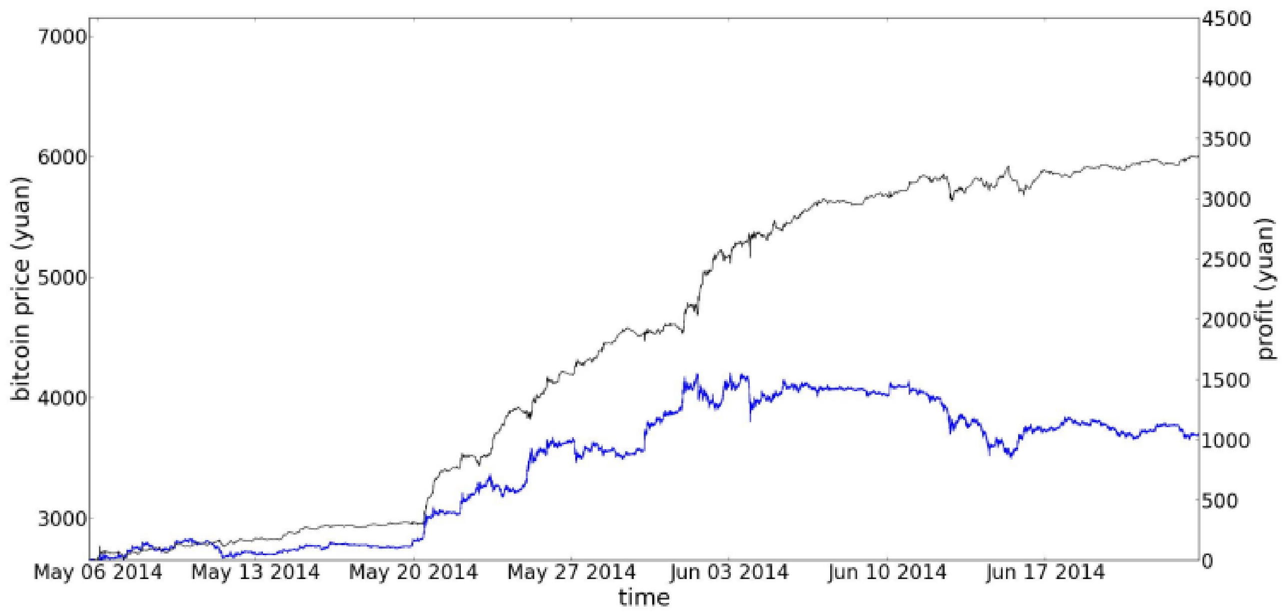


Fig. 2: The inverse relationship between the average profit per trade and the number of trades



- blue: bitcoin price, black: cumulative profit
- strategy performs better in the middle section when the market volatility is high.

Discussion

Are there Interesting Patterns?

- head-n-shoulder, triangle, and so on

Scaling of Strategy

- +1/-1 Bitcoin
- flexibility in the Bitcoin position

Scaling of Computation

- not'representative' prior time-series
- use all possible time-series could have improved the prediction power