**Predicting First Day of IPO Stock Return Using XGBoost**

**Abstract**

Machine Learning has been adopted long in finance and the area of stock trading. Models are trained to identify technical indicators, optimise portfolio diversification and build automated trading system. First day of IPO stock return is one of the market anomalies being studied by researchers. Pricing of IPO stock is an arena between the management of the listing issuer, stock underwriters which then sell the shares to investors and the outcome of the subscription price is often thought to deviate from the true market value, which can be reflected by the public market on the first day of trading.

Before and shortly after the 21st century, the initial IPO return in the field of artificial intelligence was not studied extensively other than linear classification. Since then, genetic algorithms to predict IPO underpricing were proposed by Luque, Quintana and Isasi in 2012. In the same year, Huang et al. proposed genetic-search model IPO first day return based on common financial metrics as data inputs. Artificial neural networks (ANN) and support vector machines (SVM) were also proposed later. Quintana, Saez and Isasi proposed an ensemble method, namely random forest, to predict IPO underpricing in 2017, and ensemble method has become a mainstream in studying the topic of IPO returns. In this study, we will explore other artificial intelligence algorithms, namely XGBoost of the ensemble method, to learn IPO data and predict the first day of IPO return. The investable universe of this study is IPO in Hong Kong stock market.

**Introduction**

Extreme Gradient Boosting, often known as “XGBoost”, is an emerging algorithm in machine learning for classification or regression. XGBoost, which is also an ensemble machine learning method, is powerful and efficient that a number of competition winning machine learning algorithms have been based on XGBoost. This study is motivated to apply XGBoost to learn first day IPO return in the Hong Kong Stock Exchange (SEHK).

**Input and Output**

A dataset is obtained from AAStocks.com, which is a leading Hong Kong financial market real-time data provider. Table 1 below provides the attributes which we obtained from AAStocks.com and employed for our XGBoost model.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Company** | **Mkt Cap (Offer Price)** | **Market (GEM/Main)** | **Industry** | **Oversubscription (%)** | **Allotment chance** | **First day return** |

Table 1 (to be updated)

**About XGBoost**

Boosting is the most widely used ensemble learning methods, where multiple weak learners are trained to solve a learning problem and configured to improve the combined prediction results. Weak learners individually have low variance. They have fast learning but due to the low model complexity, bias associated with them is high too. [Generally, the weak learners are short decision trees known as stumps (also the model choice of XGBoost), and ensemble models configure the weights of each weak learners to maximise the likelihood of the outputs.] Common boosting algorithms are Adaptive Boosting (AdaBoost) and Gradient Boosting, and Tianqi Chen and Carlos Guestrin of the University of Washington modified the Gradient Boosting and coined the term XGBoost as a supervised learning.

The objective function of XGBoost comprises training loss function and regularising overfitting:

obj(θ) = L(θ) + Ω(θ)

where L is the loss function and Ω is the regularisation term where mean squared error or logistic loss are often chosen as the loss function.

XGBoost is a tree ensemble model that consists of a set of classification and regression trees (CART) where a CART has a score on each leaf which is the prediction score (some refer it as the similarity score). The prediction scores of each individual tree are summed up to get the final score and the prediction model is in the form:

Where *K* is the number of CARTs, *f* is a function in the set of all CARTs .

In training the trees, the key is to define an objective function and optimise it. XGBoost regularises the structure of the trees and configures the prediction scores of the CARTs. An additive strategy is employed, learning a tree one at a time and adding a new tree next. Therefore, the prediction model at step t is shown in the formula:

For mean squared error to be considered for optimising the objective function, the objective function should be:

where c is a constant.

It is important to regularise the learning objective to prevent overfitting. The complexity is defined by:

**Dataset and Features**

The target investable universe is Hong Kong stock market and our objective is to examine the first day of return of IPO stocks. The dataset is a small dataset obtained from AAStocks.com and contains [\*] number of shares, each of which has been assigned with [\*] attributes.

Consistent other studies, first day of return (*R*) is defined as (closing price of first day of trading – final offer price) / (final offer price), expressed as percentage. The closing price of each stock is also easily accessible from AAStocks.com.

**Experiments / Results / Discussion**

*Library and parameters*

We use Python as our programming language. To use XGBoost, the library [py-xgboost] is installed, together with other necessary libraries including [numpy, pandas, scikit-learn].

*Hyperparameters of XGBoost*

Documentations of XGBoost set out general parameters and parameters for tree boosters. The following are the important one we modelled:

learning\_rate: step size initially set at [0.1]

max\_depth: which govern the training stumps on the depth level they could have.

subsample: subsample ratio of the training instances. setting it to [0.5] means that XGBoost would randomly sample half of the training data prior to growing trees. and this will prevent overfitting.

colsample\_bytree: the subsample ratio of columns when constructing each tree, will lead to overfitting if it is set too high. The value is set at [0.3].

n\_estimator: number of training stumps to be built, which is set at [100]

objective: the default of objective is squarederror (regression with squared loss), [which is our objective]

[We also tried to adjust [some of the parameters above] to evaluate the impact to convergence. The results are:]

**Results**

[Plot]

**Conclusion**

**References**

1. T. Chen and C Guestrin. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. August 2016. Pages 785-794.
2. Luque, C., Quinlana, D., & Isasi, P. (2012). Predicting IPO underpricing with genetic algorithms. International Journal of Artificial Intelligence, 8(S12), 133-146.
3. Quintana, D., Sacz, Y., & Isasi, P. (2017). Random forest prediction of IPO underpricing. Applied Sciences, 6(7).
4. B. Baba, G. Sevil. (2020). Predicting IPO initial returns using random forest. Borsa Istanbul Review 20-1(2020) 13-23.
5. S. Russell, P. Norvig. (2009). Artificial Intelligence: A Modern Approach. Prentice Hall Press. 3rd Edition.
6. [Documentations of XGBoost (xgboost.readthedocs.io/)