Comparison of models for Ubiquant Market Prediction

Jiabao Li, Zhihan Zhu

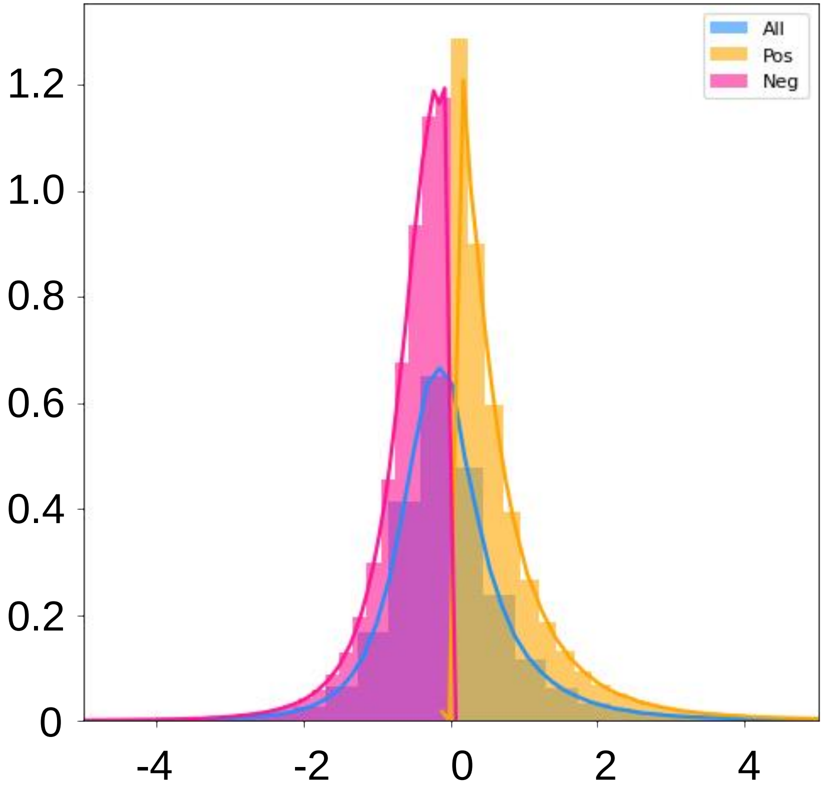
MATH 5470

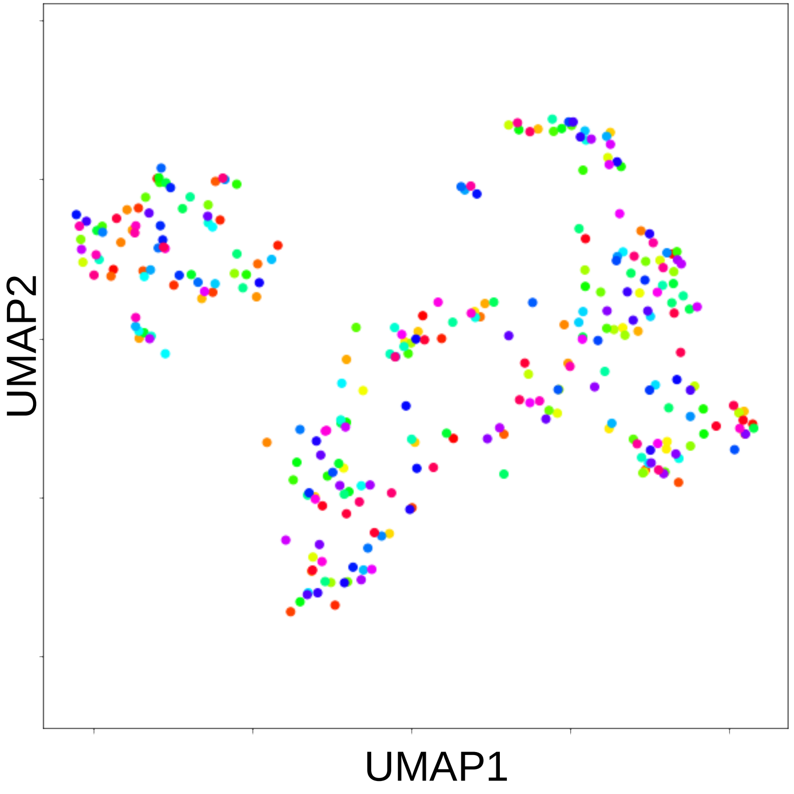
2022.4.22

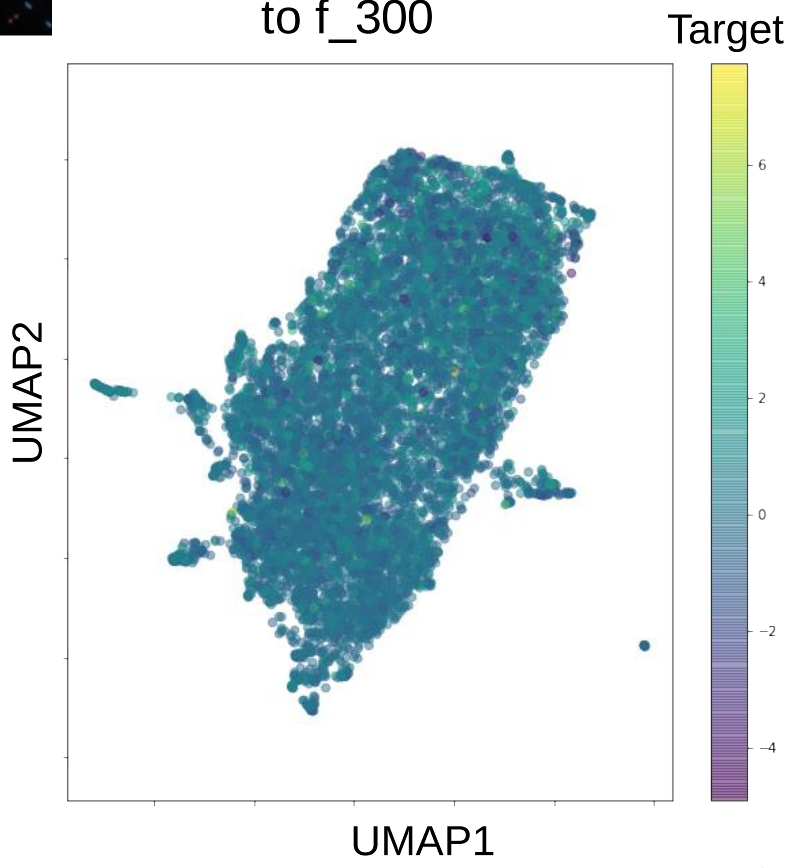
https://youtu.be/Wzn00pUDEYY

Introduction

- 3,141,410 transaction data
  - Target value
  - Time_ID
  - Investment_ID
  - 300 features named f_1 to f_300



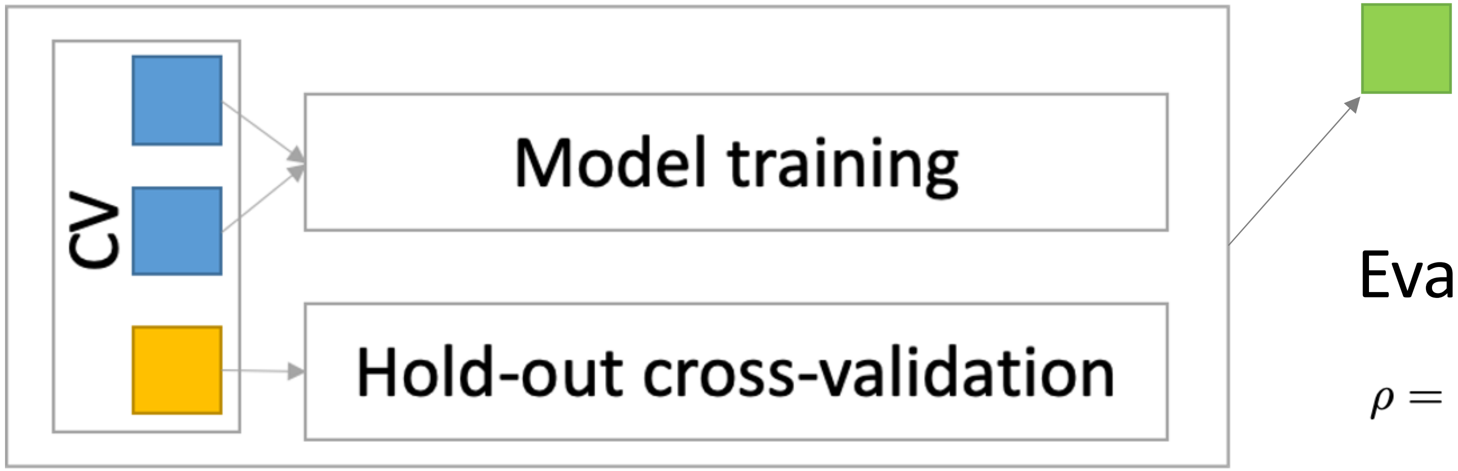




Machine Learning Framework

- Which model can predict the market data better?

3-fold cross-validation



Independent test dataset

Hyper-parameter tuning

Evaluation metric:

$$\rho = cor(\hat{y}, y) = \frac{\Sigma(\hat{y}_i - \bar{\hat{y}})(y_i - \bar{y})}{\sqrt{\Sigma(\hat{y}_i - \bar{\hat{y}})^2 \Sigma(y_i - \bar{y})^2}}$$

Methods - LASSO

Formula: $$\hat{y}_i = \sum_j \beta_j x_{ij}$$

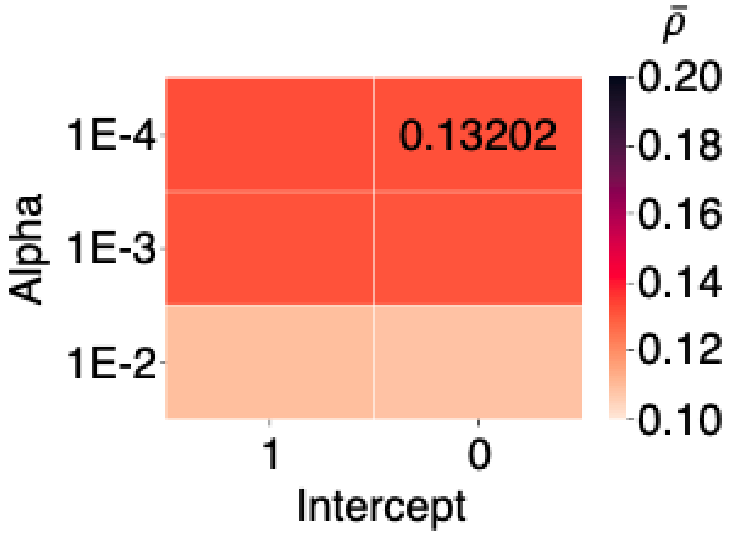
Objective function: $$L = \sum_{i=1}^{n} \left(y_i - \beta_0 c - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \alpha \sum_{j=1}^{p} |\beta_j|$$

Hyper-parameter:

C: fitting intercept

alpha: Controlling the strength of the L1 term

Cross-validation performance:



Methods - GBDT

Formula: $$\hat{y}_i = \sum_{t=1}^{T} f_t(x_i), f_t \in \mathcal{F}$$

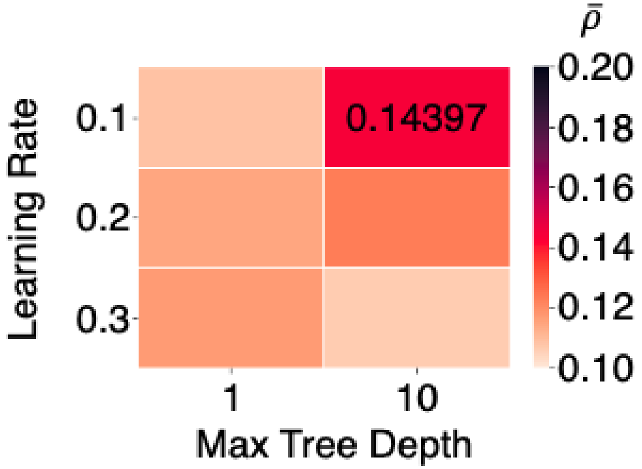
Objective function: $$L^{(t)} = \sum_{i=1}^{n} \left(y_i - \hat{y}_i^{(t-1)} - \alpha f_t(\mathbf{x}_i)\right)^2 + \Omega(f_t)$$

Hyper-parameter:

Learning rate: weight of a new generated tree

Max tree depth: how complex of a tree

Cross-validation performance:



Methods - DNN

Platform: GPU&Kaggle
Pipeline: Kaggle::greydolphin

```
  | Name         | Type       | Params
------------------------------------------
0 | id_embedding | Embedding  | 110 K
1 | layers       | Sequential | 96.4 K
------------------------------------------
206 K     Trainable params
0         Non-trainable params
206 K     Total params
0.826     Total estimated model params size (MB)
```

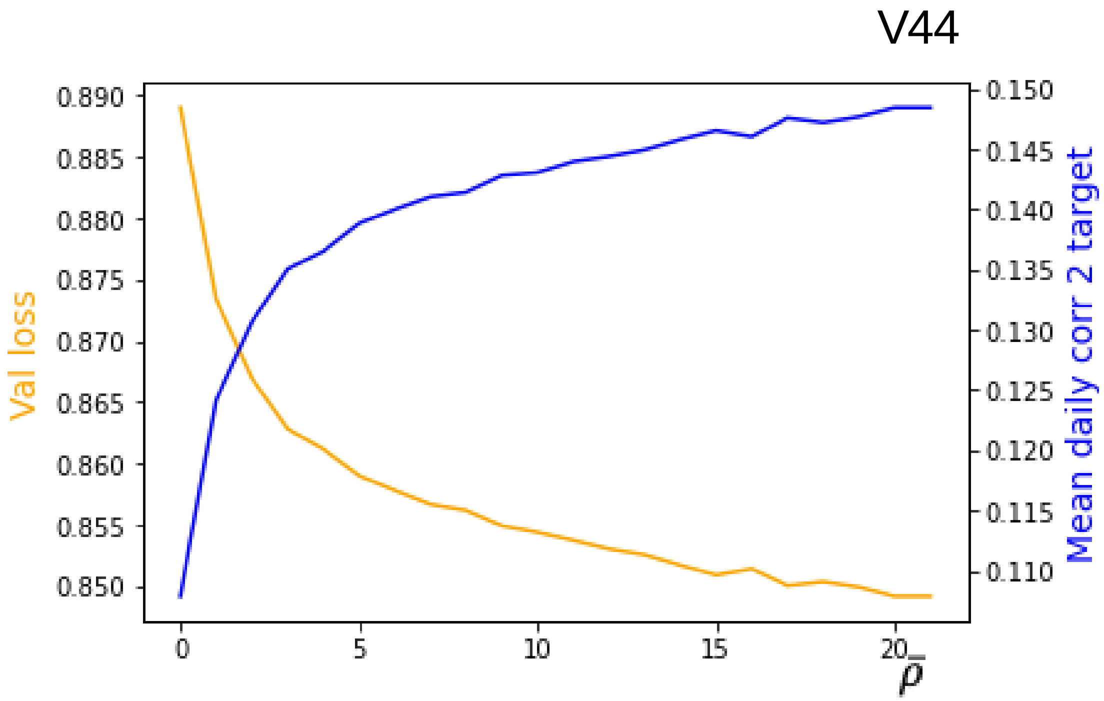
Simple

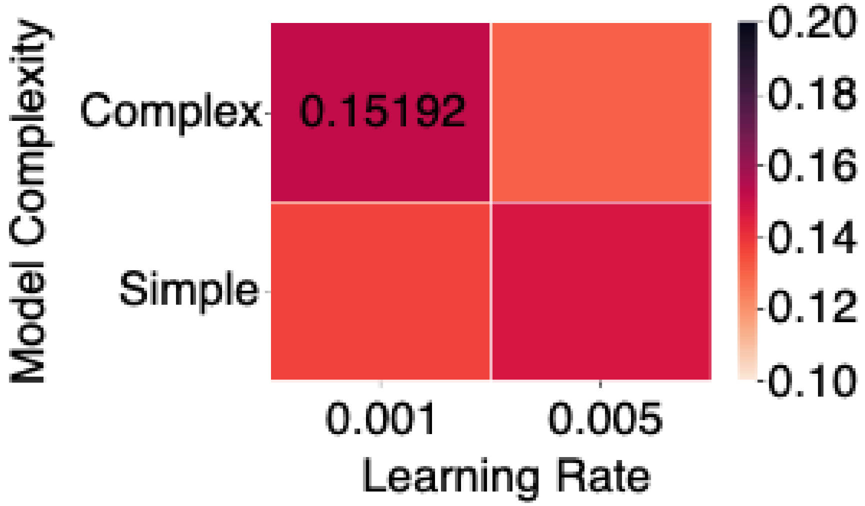
```
  | Name         | Type       | Params
------------------------------------------
0 | id_embedding | Embedding  | 110 K
1 | layers       | Sequential | 360 K
------------------------------------------
470 K     Trainable params
0         Non-trainable params
470 K     Total params
1.883     Total estimated model params size (MB)
```

Complex

Table 1: Results for DNN models

Model	Model Complex	Learning Rate	Pearson Cor	Name
DNN	complex	0.001	0.15191829	V4
DNN	complex	0.005	0.1302046	V42
DNN	simple	0.005	0.13668779	V43
DNN	simple	0.001	0.14695018	V44





Methods - LSTM

```
  | Name           | Type       | Params
-------------------------------------------
0 | id_embedding | Embedding  | 110 K
1 | layers1      | Sequential | 20.1 K
2 | lstm1        | LSTM       | 99.3 K
3 | lstm2        | LSTM       | 49.7 K
4 | layers2      | Sequential | 545
-------------------------------------------
279 K     Trainable params
0         Non-trainable params
279 K     Total params
1.119     Total estimated model params size (MB)
```

Complex

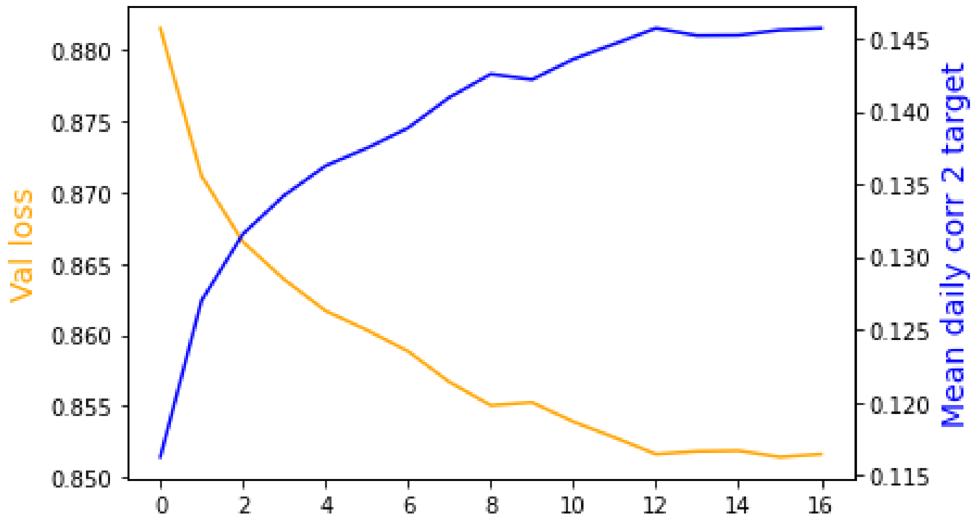
```
  | Name           | Type       | Params
-------------------------------------------
0 | id_embedding | Embedding  | 110 K
1 | layers1      | Sequential | 28.7 K
2 | lstm1        | LSTM       | 395 K
3 | lstm2        | LSTM       | 197 K
4 | layers2      | Sequential | 1.1 K
-------------------------------------------
732 K     Trainable params
0         Non-trainable params
732 K     Total params
2.930     Total estimated model params size (MB)
```

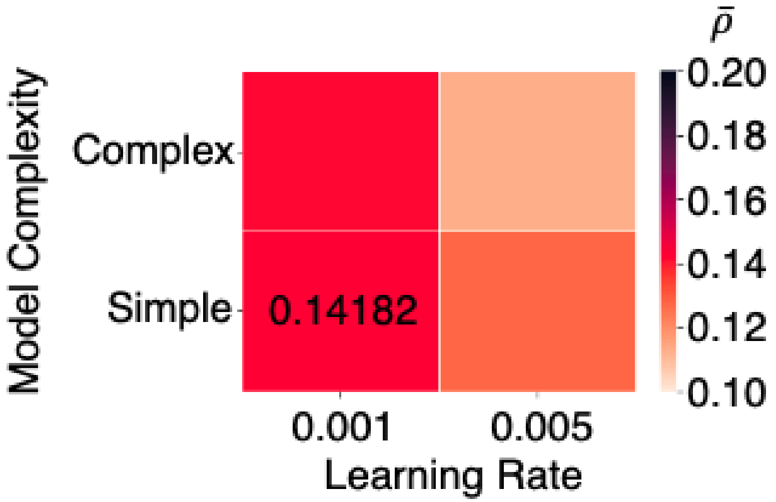
Simple

Table 2: Results for LSTM models

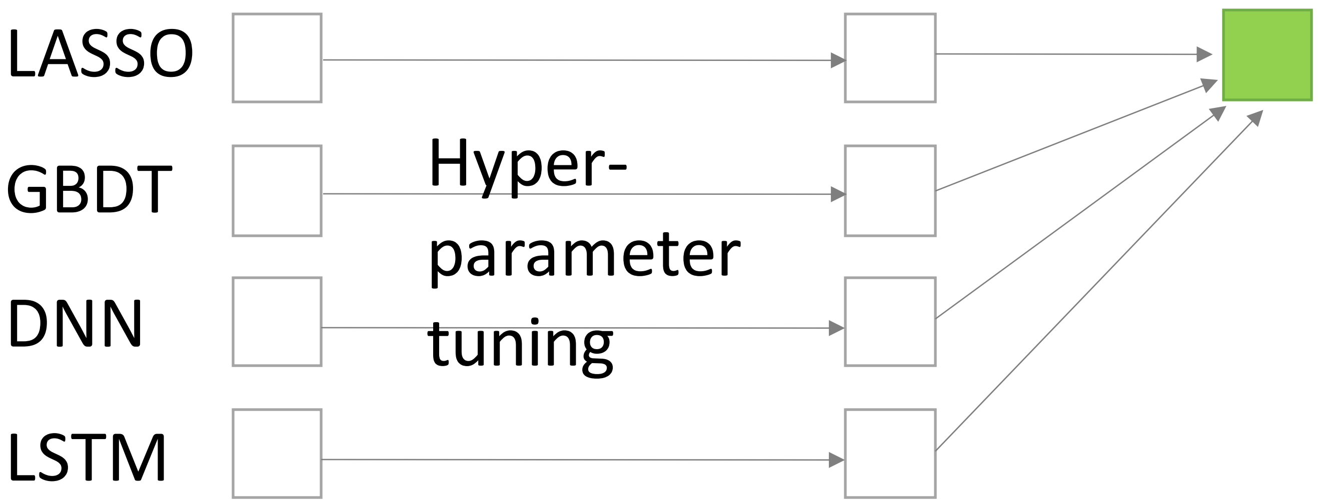
Model	Model Complex	Learning Rate	Pearson Cor	Name
LSTM	complex	0.001	0.14113208	V5
LSTM	complex	0.005	0.11254942	V52
LSTM	simple	0.001	0.14182154	V53
LSTM	simple	0.005	0.12764723	V54

V53





Results & conclusion



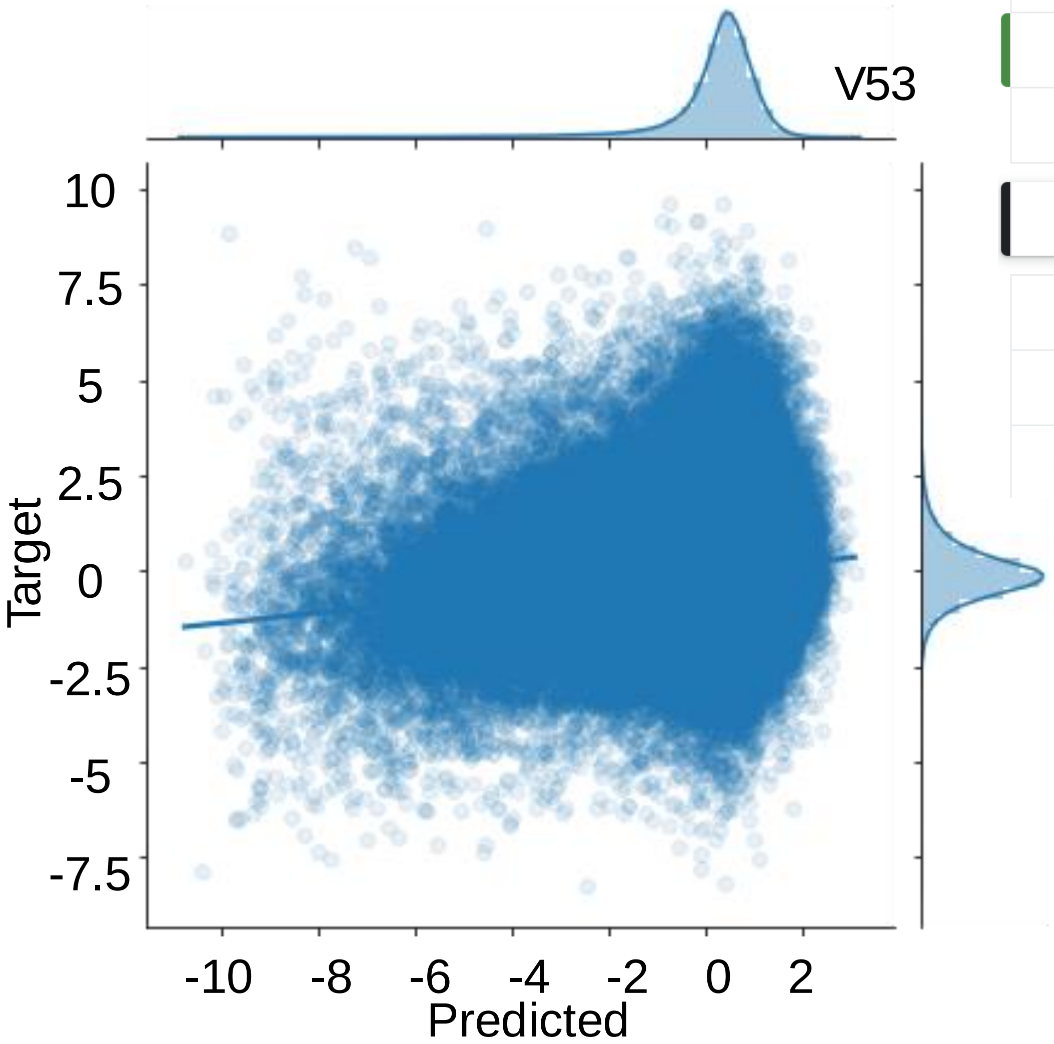
Independent test dataset
(25% data)

Performance:

Model	ρ in validation dataset	ρ in Kaggle
LASSO	0.13063	0.1076
GBDT	0.15513	0.1193
DNN	0.15191	0.1348
LSTM	0.14182	NA

Summited for the final round competition

Case study & Discussion



#	Team	Members	Score	Entries	Last	Code
1	MGLSY		0.1680	416	11h	
289	Head of Meme Trading		0.1555	44	20h	
387	**math5470_Li_Zhu**		0.1554	33	20h	
450	Bryan Arnold		0.1553	11	20d	
670	MathGhost		0.1542	22	2d	
2505	vhack		0.1127	25	8h	

- Many transactions are not well predicted
- Metrics
  - Best final scores: 0.1680 (0.1554)
  - Our best rank: 386/2885 (13.4%)
- The features are not enough
  - More data should be collected (Actually some teams do in Kaggle)

Thank you