RecSys Challenge 2020 Workshop

A Stacking Ensemble Model for Prediction of Multi-type Tweet Engagements

Team Wantedly's 3rd place solution

26.Sep.2020 - Shuhei Goda, Naomichi Agata, Yuya Matsumura

RECSYS 2020

CHALLENGE TASK

A Task of predicting different types of engagement on Twitter

- Multi-label binary classification that predicts each engagement per (tweet ID, engaging user ID).
- Four types of engagements: *Like*, *Reply*, *RT* and *RT with comment*

Evaluation Metrics

- PR-AUC
- RCE (Relative Cross Entropy)

$$RCE = \frac{(CE_{naive} - CE) * 100}{CE_{naive}}$$

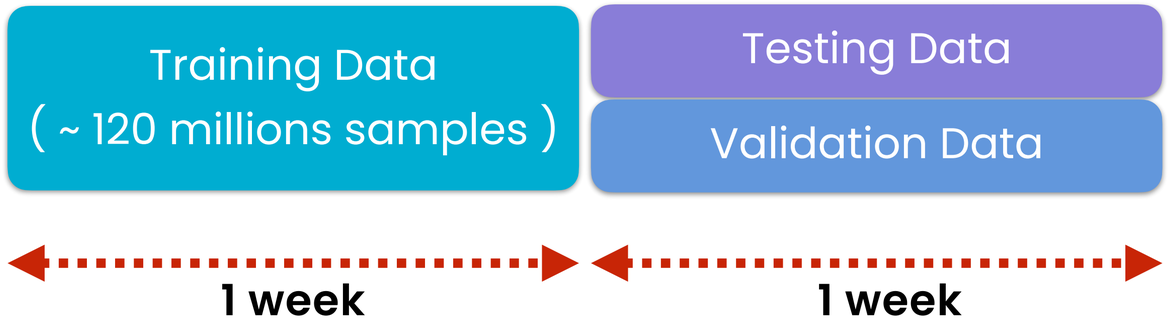
Indicates the improvement of prediction relative to the naive prediction.

DATASET DESCRIPTION

The information provided for the challenge dataset

- Tweet info. : *tweet ID, timestamp, text token,* etc.
- Engaging User info. : *user ID, following count, follower count,* etc.
- Engaged with User info. : *user ID, following count, follower count,* etc.
- Engagement info. : *timestamps of the engagements*

Train / Test split for evaluating



DATASET CHARACTERISTICS (1)

Label Imbalance

- The number of positive samples: RT with Comment < Reply < RT < Like
- The positive ratio of Like is 43%, while that of RT with Comment is only 0.7% .

Table 1. Label Imbalance in Each Engagement

engagement type	positive sample	negative sample	positive ratio
Like	52,719,548	68,666,883	0.434
Reply	3,108,287	118,278,144	0.026
RT	13,264,420	108,122,011	0.109
RT with comment	884,082	120,502,349	0.007

DATASET CHARACTERISTICS (2)

High correlation between engagement types

- Users sometimes make multiple types of engagements for one tweet.
- High co-occurrences are observed in some pairs.
  - e.g. *RT and Like , RT and RT with comment*

Table 2. Engagement Co-occurrence Matrix

	Like	Reply	RT	RT with comment
Like	1.000	0.023	0.132	0.007
Reply	0.383	1.000	0.092	0.014
RT	0.523	0.022	1.000	0.067
RT with comment	0.415	0.048	1.000	1.000

The co-occurrence matrix of engagements in the training dataset. Labels at Y-axis represent referenced engagements E_r , while labels at X-axis are potential co-occurrence engagements E_p. The conditional probability $P(E_p|E_r)$ of each engagement pair is presented in the corresponding block.

OVERVIEW OF OUR SOLUTION

Model Architecture

- Stacking LightGBMs

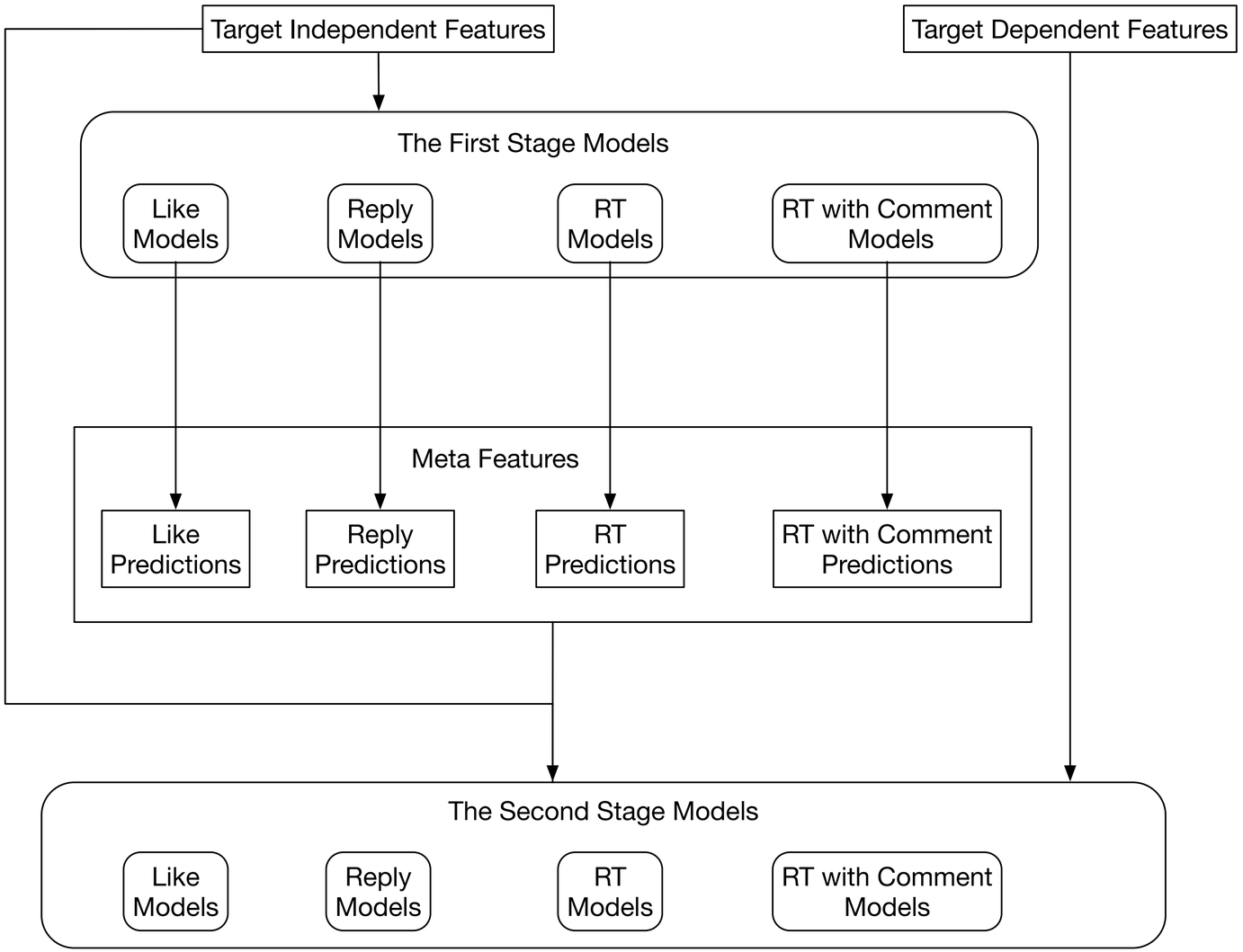
Features

- Categorical Features
- Network Features
- Text Features

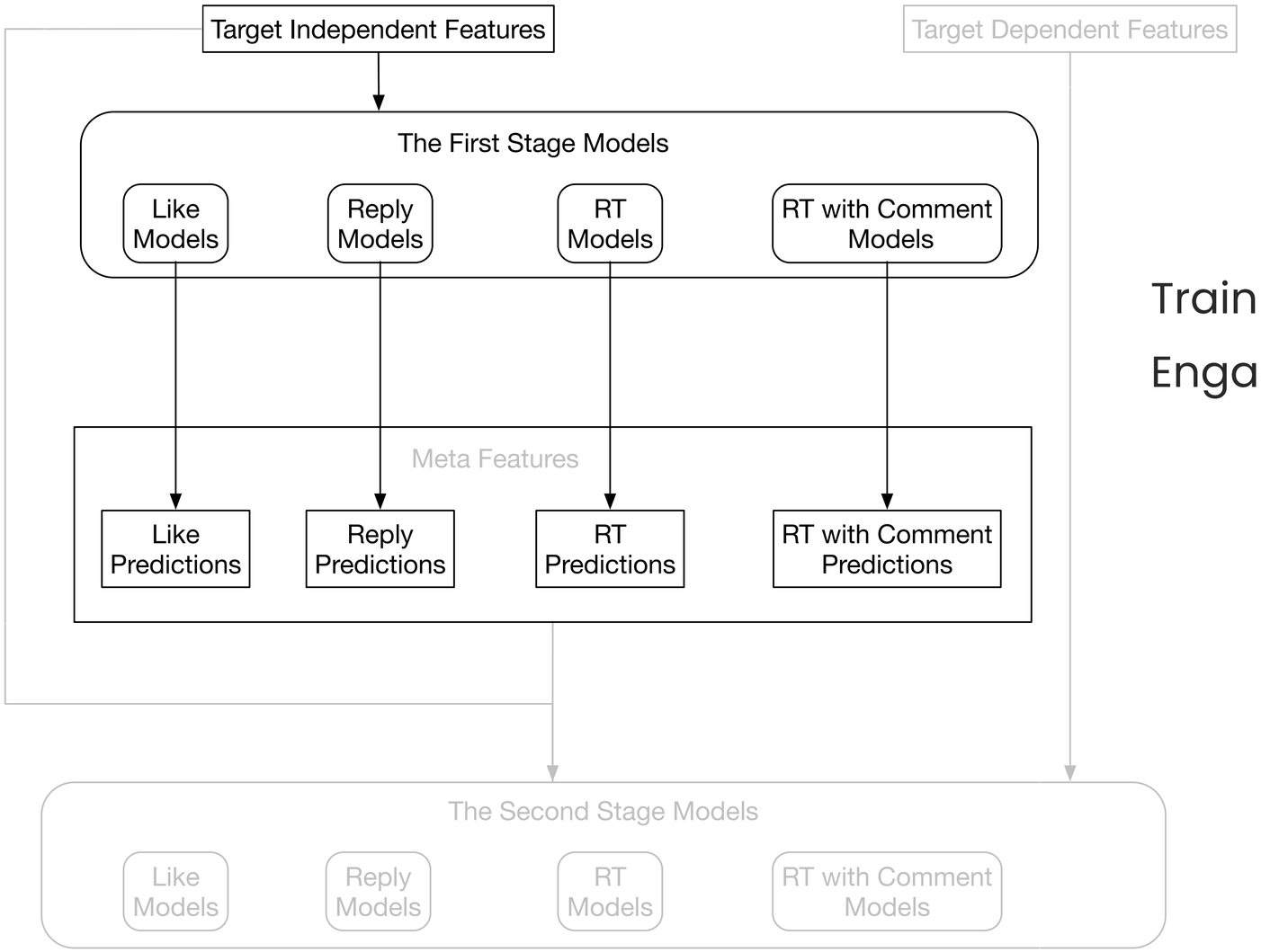
Training Process

- Bagging with negative under sampling
- Stratified K-Folds over Retweet with Comment

MODEL ARCHITECTURE

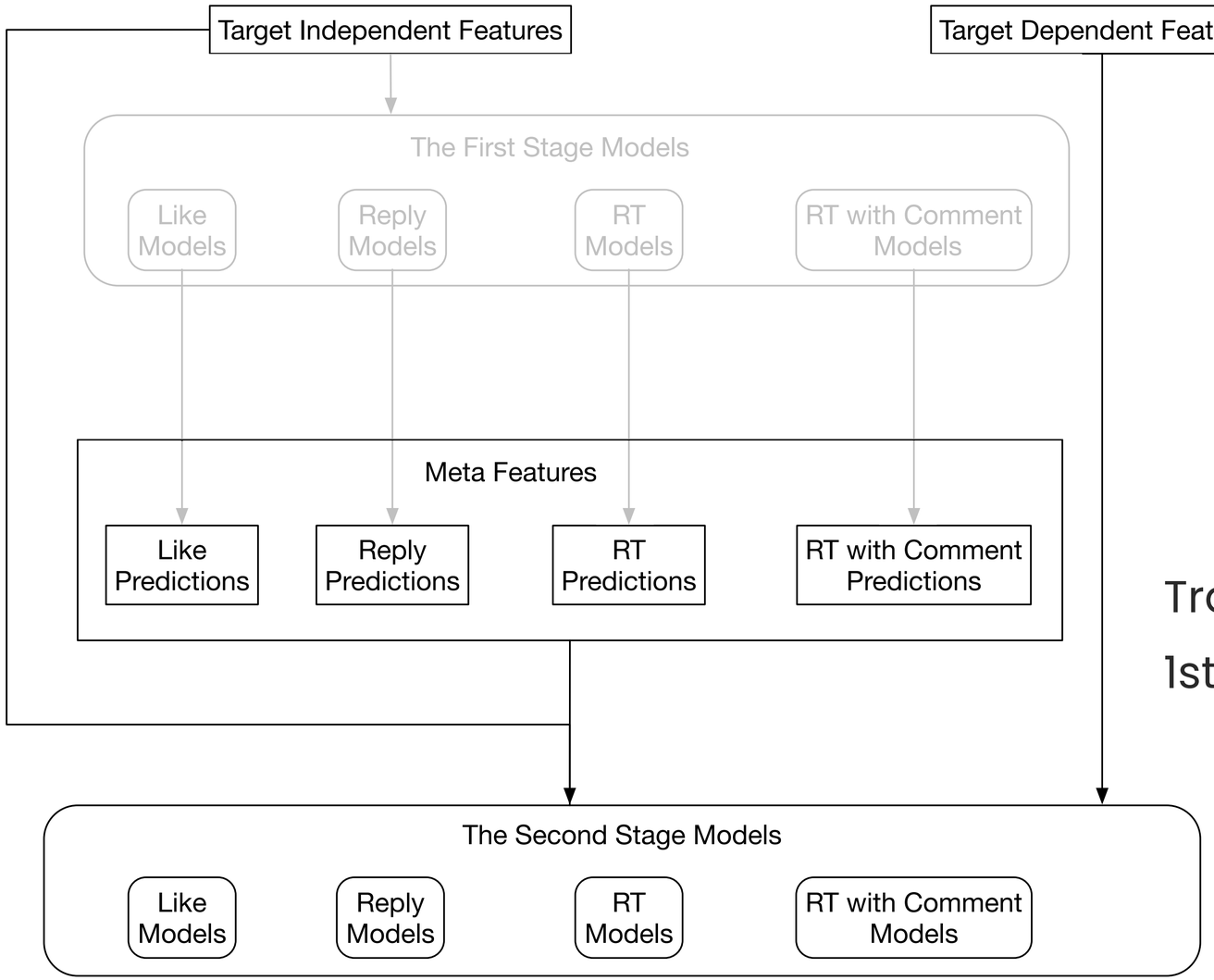


MODEL ARCHITECTURE 1st Stage



Train LightGBMs for each Engagement type

MODEL ARCHITECTURE 2nd Stage



Train LightGBM with

1st stage models predictions

Applying different encoding methods to each categorical variable

- Low-cardinality categories: Label Encoding
  - e.g. *language, tweet type*
- High-cardinality categories: Frequency Encoding & Target Encoding
  - e.g. *tweet ID, user ID*

Considering the combination of categorical variables

- Capture more complex relationships among categorical variables
  - e.g. *Hashtag* × *engaging user ID*

Social Follow Graph: considering relationships between users and their social influence

- flags that represent whether there are first or second degree connections
- PageRank

Like Graph: considering user similarities in terms of their preferences

- each node represents a user and each edge represents Like engagement
- Random Walk with Restarts: the number of visits to engaged with users from engaging users

A text-based estimation of Engaging User's preferences

- Considering two types of preferences
  - Preferences for the contents of Tweets
  - Preferences for the Engaged with Users

- Express preferences as the similarity by inner products of the vectors
  - Tweet: The outputs of pretrained multi-lingual BERT
  - Engaging User: The averaged vectors of the Tweets users are engaging
  - Engaged with User: The averaged vectors of the users' Tweets

FEATURES Other Features

- **Count:** The number of hashtags, media
- **Following/Follower:** Following count, Follower count, F/F Ratio
- **Account Age:** The time elapsed since user accounts were created
- **User Activity:** Relative active time for each user
- **Main Language:** The main language and its ratio in each user's Home timeline

Use the predictions of the 1st Stage Models of each engagement as features of the 2nd Stage Models.

- Since every engagement is highly co-occurring, the information on other engagements is important to predict one engagement.
- Take the aggregation of the predictions by categories such as user ID and tweet ID.
  - Express the tendency of the engagements in each category.

Bagging with Negative Under-Sampling

- Use Bagging to create high performance models efficiently with small training dataset for each model.
- Sampling process is as below:
  1. Apply Negative Under-Sampling to reduce the data size and make the number of positive and negative samples is equal.
  2. Apply Random Sampling to make the data size even smaller.

Re-Calibration

- The predicted probabilities are based on the downsampling space because of Negative Under-Sampling in training.
- Predicting the engagement probability is required since RCE is one of the metrics.
- Apply re-calibration below as a post-processing.

$$\frac{p}{p + \frac{1 - p}{w}}$$

where p is the prediction in downsampling space
and w the negative downsampling ratio.

Use Stratified K-Folds so that the ratio of positive samples of RT with comment in each fold is equal.

- We use the same splits for trainings to predict every engagement targets.
  - if we use different splits, the negative influence of the leakage due to meta features and target encoding gets bigger.
  - This is because each engagement is not actually independent.
- Considering the calculation time, we set the number of folds to 3.

All our experiments were conducted on resources as follows:

- Google BigQuery
- Google Dataflow
- Google Compute Engine
  - vCPUs: 64
  - Memory: 600GB

Our code is available at

- https://github.com/wantedly/recsys2020-challenge

The score of the 2nd stage models is considerably better than the 1st stage models.

- The 2nd stage models outperform the 1st stage models on both metrics.
- This result supports the effectiveness of our stacking architecture.

Table 5: Final Results

	training		validation	
Model name	Ave. PRAUC	Ave. RCE	Ave. PRAUC	Ave. RCE
first stage models	0.383	22.216	0.344	17.965
second stage models	0.439	28.583	0.390	22.535

The difference between the training and validation score of the 2nd stage models is larger than the 1st stage models

- In the case of RCE, the difference in the 1st and 2nd stages is as follows.
  - 4.251 (1st stage models) < 6.048 (2nd stage models)
- We finally considered that it is not a problem as both the training and validation scores improved.

Table 5: Final Results

	training		validation	
Model name	Ave. PRAUC	Ave. RCE	Ave. PRAUC	Ave. RCE
first stage models	0.383	22.216	0.344	17.965
second stage models	0.439	28.583	0.390	22.535

In the 2nd stage models, the larger the number of training data, the worse the validation score.

- This is due to the use of meta features and target encoding.
- Change the number of training data in the 2nd stage models depending on the target.
  - 100,000 for Like, 1,000,000 for other targets

Table 3: Experiment Results of Training Data Size

	training		validation	
Limit of the training data	Like RCE	RT RCE	Like RCE	RT RCE
100,000	36.494	35.500	24.728	29.568
1,000,000	38.326	36.675	23.687	30.122
2,000,000	38.709	36.918	22.991	30.104

We set the number of models to 5.

CONCLUSION

- We described ***Team Wantedly's*** solution for RecSys Challenge 2020, which won **the 3rd place**.
- We train **two stage stacking models** to capture the characteristics of high co-occurrence between engagements effectively and efficiently.

Competition Results

Rank	Team/User Name	Overall Score	PRAUC Retweet	RCE Retweet	PRAUC Reply	RCE Reply	PRAUC Like	RCE Like	PRAUC RT with comment	RCE RT with comment
1	NVIDIA RAPIDS.AI	9	0.6111	37.91	0.2185	24.23	0.9108	53.01	0.0796	17.40
2	learner	14	0.5395	30.96	0.2218	21.94	0.7761	25.42	0.0757	14.61
3	Team Wantedly	18	0.5266	30.06	0.1918	20.44	0.7716	24.76	0.0724	14.86

