4 Idiots' Approach for Click-through Rate Prediction

Team Members

4 Idiots consist of:

Name	Kaggle ID	Affiliation
Yu-Chin Juan	guestwalk	National Taiwan University
Wei-Sheng Chin	mandora	National Taiwan University
Yong Zhuang	yolicat	National Taiwan University
Michael Jahrer	Michael Jahrer	Opera Solutions

Our final model is an ensemble of NTU's model and Michael's model. Michael's model is based on his work in Opera Solutions, so he cannot release his part. Therefore, in the released codes and documents we only present NTU's solution.[1]

[1] The private leaderboard score of NTU's solution is 0.3796, so the rank keeps unchanged.

Data Set

all features are categorical

Label	hour	banner_pos	site_id	site_domain	...	C20	
+1	14102100	0	1fbe01fe	f3845767	...	-1	40M
-1	14102100	1	fe8cc448	9166c161	...	100084	
			⋮				
-1	14103023	1	f61eaaae	25d4cfcd	...	100077	
?	14103100	0	8fda644b	7e091613	...	100084	4M
?	14103100	1	e151e245	f3845767	...	100019	
			⋮				
?	14103123	0	1fbe01fe	bb1ef334	...	-1	

Evaluation

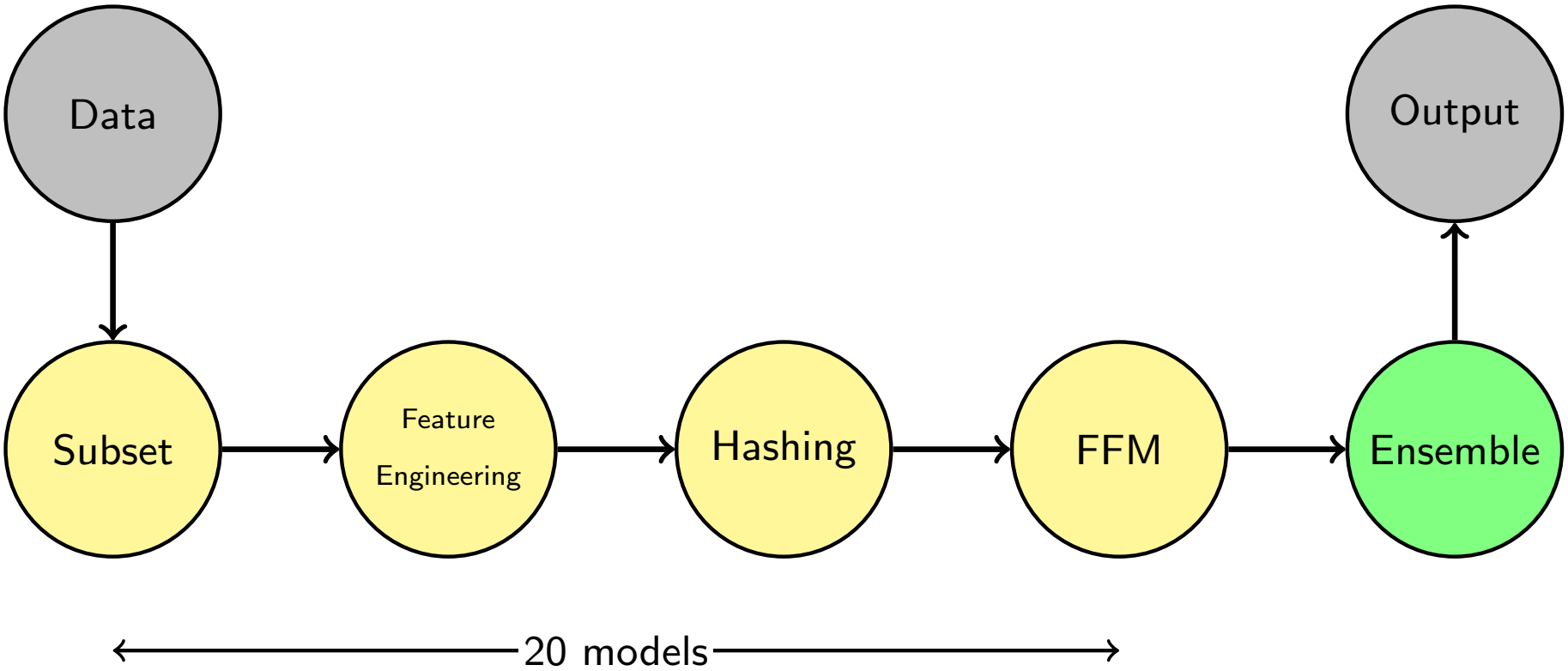
Logarithmic loss is used in this competition:

$$\text{logloss} = -\frac{1}{L}\sum_{i=1}^{L} y_i \log p_i + (1 - y_i) \log (1 - p_i),$$

where L is the number of instances, $y_i \in \{0, 1\}$ is the label of the ith instance, and p_i is the probability of that the ith instance is clicked.

Flow Chart

Our best model is an ensemble of 20 models. These models are built under the yellow part of the flow chart below with different settings.



Subset

Instead of using the whole dataset, in this competition we find splitting data into small parts works better than directly using the entire dataset. For example, in one of our models we select instances whose site_id is 85f751fd; and in another one we select instances whose app_id is ecad2386.

Feature Engineering

Except the raw features, we generate the following additional features:

- Counting features
- Bag features
- Click history

Counting Features

Counting features include:

- `device_ip` count
- `device_id` count
- hourly user count
- user count
- hourly impression count

Here, **user** is defined as:

$$\begin{cases} \texttt{device\_ip + device\_model}, & \text{if } \texttt{device\_id} \text{ is } \texttt{a99f214a} \\ \texttt{device\_id}, & \text{otherwise} \end{cases}$$

An **impression** is defined as concatenating all raw features together.

Bag Features

For each user, we add bags of features. For example, if we have `user1` associated with `app_id-A` and `app_id-B`, and `user2` associated with `app_id-C` and `app_id-D`, then we generate an additional feature `bag_of_app_id`:

user	app_id	bag_of_app_id
user1	A	A, B
user1	B	A, B
user2	C	C, D
user2	D	C, D

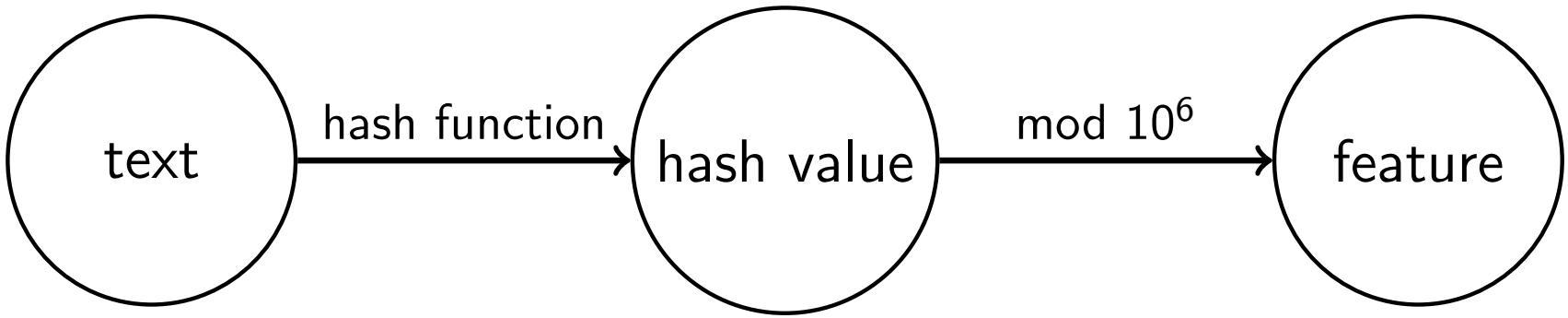
Click History

We generate a click history feature for users who have `device_id` information. For example:

label	user	history
0	user1	
1	user1	0
1	user1	01
0	user1	011

Hashing

We use hashing trick to transform text features. For example:



site_id-68fd1e64	739920192382357839297	839297
app_id-80e26c9b	839193251324345167129	167129

Field-aware Factorization Machines (FFM)

For details of FFM, please check the following slides:

`http://www.csie.ntu.edu.tw/~r01922136/slides/ffm.pdf`

This model is also used in another CTR competition.[2] We are interested to see if it can be more widely used. If you want to use this model, we have released a package **LIBFFM** at:

`http://www.csie.ntu.edu.tw/~r01922136/libffm`

[2] `https://www.kaggle.com/c/criteo-display-ad-challenge`

Ensemble

By using different settings for subset / feature engineering / FFM, we totally built 20 models. We use a simple average approach to blend them. For example, if an impression has three predictions 0.1, 0.15, and 0.08 from three different models, then the averaged prediction is:

$$p = f\left(\frac{f^{-1}(0.1) + f^{-1}(0.15) + f^{-1}(0.08)}{3}\right) = 0.1067,$$

where f is logistic function and f^{-1} is the inverse function of f.

Source Codes

The source codes of our solution can be obtained at:

`https://github.com/guestwalk/kaggle-avazu`

If you want to re-use our model, please download **LIBFFM** at:

`http://www.csie.ntu.edu.tw/~r01922136/libffm`

Miscellaneous

- Our solution includes many parameters (e.g. number of iteration in the FFM solver). Most of parameters are tuned by running experiments on a 10% subset of the raw dataset.
- In these slides, we focus on presenting important concepts of our solution. For ease of understanding, some details are not disclosed. For example, for each counting feature, actually we only consider those smaller than a certain threshold. To understand all details, please trace our code. Of course, you can also ask questions on the forum. It's very welcomed!
- In this competition, FFM is an effective model. However, because our competitors also use FFM,[3] it is not the key to win this competition. We conclude that the keys are feature engineering and ensemble. It is worth noting that our ensemble is blending the same model (i.e., FFM) built from different subsets of data and features.

[3] We are really happy to see some teams use our codes released in Criteo's CTR competition!