* Definition of problem
* Optimization Function
* Data size
* Implicit label and sampling
* Funnel strategy
* Different modeling choice
* Loss Function
* Debug strategy
* Production Strategy
* Old style ML

## Definition of problem

## 

1. Click through rate
2. IG explore
3. Tiktok recommendation

## Sampling and Model calibration

For implicit feedback, the data does not have a clear label. For model training, we generally assume that the products that the user has interacted with are all positive examples, and through sampling, select a part of the products that the user has not interacted with as negative examples. The process of selecting negative examples based on a certain strategy from the user’s non-interactive product set is called Negative Sampling.

Heuristic

1. Random Negative Sampling - efficient, most common, dist is same
2. Popularity Based Sampling -

Model based

1. Dynamically Negative Sampling - The idea of DNS is to dynamically change the sampling distribution according to the current situation of the model to improve the sampling quality of each round.
2. Generative Adversarial Networks Based Negative Sampling
3. Simplify and Robustify Negative Sampling, SRNS [8] - This algorithm uses the observed statistical features as prior knowledge to distinguish between False Negative Examples and Strong Negative Examples to enhance the robustness of the model and uses a DNS-like structure for sampling to ensure sampling quality.

Model calibration

Negative downsampling can speed up training and improve model performance. Note that, if a model is trained in a data set with negative downsampling, it also calibrates the prediction in the downsampling space. For example, if the average CTR before sampling is 0.1% and we do a 0.01 negative downsampling, the empirical CTR will become roughly 10%.

We need to re-calibrate the model for live traffic experiment and get back to the 0.1% prediction with q = p / p+(1−p)/w

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

<https://medium.com/mlearning-ai/overview-negative-sampling-on-recommendation-systems-230a051c6cd7#:~:text=RNS%20is%20the%20most%20basic,deviations%20in%20the%20sampling%20process>.

<https://scontent-sjc3-1.xx.fbcdn.net/v/t39.8562-6/240842589_204052295113548_74168590424110542_n.pdf?_nc_cat=109&ccb=1-7&_nc_sid=ad8a9d&_nc_ohc=feFSKdycIpYAX8F3IPR&_nc_ht=scontent-sjc3-1.xx&oh=00_AfBgiQpMHwpu4oI-QOHin2FL_QftJzEQCnLqnTdYYVIgSw&oe=6431068A>

## OBjective functions

1. Click-through Rate (CTR) - [Others You May Like, Recommended for You, ADs click]
2. COversion Rate (CVR) - conversion rate maximizes [add to cart]
3. TikTok
   1. Awareness - Reach
   2. Consideration - traffic, video view, lead generation
   3. Conversion - Sales, app, website

https://dl.acm.org/doi/10.1145/3527449

### **Features**

There are many features that are useful in the recommendation system, The category of features (sparse and dense) or (categorical, numerical,

Features source

User - demography,

Complex - prior history, time spend,

Features Types

1. Numerical - transform by binning, demeaning (whitening), outlier.
2. Catgorial - small set then one-hot encoder, larger set then embedding, very large set then you will learn sparse embedding
3. Text - Word2vec, Bert
4. Video - frame, meta data,
5. User History - each post is an embedding we concatenate those, or have sequence of actions as representation

How are sparse and dense features learned

1. Sparse features are learned using embedding lookups ,
2. Facebook -> two dffferent feed forward networks, dot product to get interaction between them
3. Airbnb → sparse features are mapped into vectors via embedding lookup. A major difference is that the normalized dense features are directly concatenated with the sparse feature embeddings
4. The most common approaches are to normalize dense features or feed them to a separate MLP and to process sparse features using embedding lookups.
5. categorical value in the dataset by a n-dimensional embedding vector. The mapping from each categorical feature value to its embedding vector is learned via the embedding table.
6. representing the gradient matrix by a sparse tensor and only calculating gradients for embedding vectors which will be non zero

<https://quoraengineering.quora.com/Unifying-dense-and-sparse-features-for-neural-networks>

https://medium.com/nvidia-merlin/training-larger-and-faster-recommender-systems-with-pytorch-sparse-embeddings-53348a2cde3f

**Model**

Different types of models

1. MF - oldstyle
2. NCF - User - embedding, item – embedding, feed into neural network,
3. Recommendation - Large corpus of items - then we use the two towers
4. Personalized learning - search ranking
5. Extreme classification
6. Wide and shallow network -
7. Multi task multi label

https://calvinfeng.gitbook.io/machine-learning-notebook/supervised-learning/recommender/wide\_and\_deep\_learning\_for\_recommender\_systems

Different Loss function

1. CTR :
   1. NE - Normalized Entropy or more accurately, Normalized CrossEntropy is equivalent to the average log loss per impression divided by what the average log loss per impression would be if a model predicted the background click through rate (CTR) for every impression. In other words, it is the predictive log loss normalized by the entropy of the background CTR. The background CTR is the average empirical CTR of the training data set. It would be perhaps more descriptive to refer to the metric as the Normalized Logarithmic Loss. The lower the value is, the better is the prediction made by the model. The reason for this normalization is that the closer the background CTR is to either 0 or 1, the easier it is to achieve a better log loss. Dividing by the entropy of the background CTR makes the NE insensitive to the background CTR.
   2. Calibration : is the ratio of the average estimated CTR and empirical CTR. In other words, it is the ratio of the number of expected clicks to the number of actually observed clicks. Calibration is a very important metric since accurate and well-calibrated prediction of CTR is essential to the success of online bidding and auction. The less the calibration differs from 1, the better the model is. We only report calibration in the experiments where it is non-trivial

## 

## 

## Troubleshooting DL model

<http://josh-tobin.com/assets/pdf/troubleshooting-deep-neural-networks-01-19.pdf>

## Old Style ML

### **Collaborative filtering** :

similar user have similar taste The focus, in this case, is on customers, their opinions on products, and their interactions with the online platform, rather than on the items' features. This implies that recommender systems in this category will rely on machine learning algorithms (such as clustering models, K-nearest neighbors, matrix factorization, and Bayesian networks) to survey customers’ perception of products via user rating, understand who likes what, and offer items already bought by other users with comparable tastes.

### 

### **Matrix Factorization** :

### Content filtering

will investigate customers' purchase patterns and recommend other products sharing similar features with those previously bought and positively reviewed.

### 

### 

### Hybrid Approach

<https://www.itransition.com/machine-learning/recommendation-systems>

[What is a Recommendation System? | Data Science | NVIDIA Glossary](https://www.nvidia.com/en-us/glossary/data-science/recommendation-system/)

### **Cold start problem** :

### 

The cold start problem occurs when the system is unable to form any relation between users and items for which it has insufficient data.

1. User cold start
2. Product cold start

Mitigation

1. Representative based - what represent a user and items, easily plugin to MF
2. Content Based - easy hybrid system to MF, issue -> objective gets cluttered , feature engineering
   1. Indexing user - geo locations, gps, site from which it came, across app tracker,
3. Bandit approach
4. Deep learning - you Namely, among the many features on video watches, search tokens, geographic and biographic information, they included Days Since Upload. Training a deep network with this additional feature, they observed that their deep net learns that “fresh” videos are more important.

https://kojinoshiba.com/recsys-cold-start/

https://medium.com/@markmilankovich/the-cold-start-problem-for-recommender-systems-89a76505a7

Multi-stage system

[Recommender Systems, Not Just Recommender Models | by Even Oldridge | NVIDIA Merlin | Medium](https://medium.com/nvidia-merlin/recommender-systems-not-just-recommender-models-485c161c755e)

The role of a recommender model, whether it’s a simple collaborative filtering example or a deep learning model like DLRM is ranking, or more accurately Scoring, the interest that a user may have with a set of items. But these scores by themselves aren’t enough to serve recommendations to that user in most real world contexts. There are several reasons for this which we’ll dig into below before we explore the solutions and how they shape the system that we end up with.