Ad Click Prediction

Ruichen Wang

December 21, 2018

Abstract

There are a lot of algorithms designed for ad ctr prediction. Those algorithms take contextual and historical features as inputs, often combines massive feature engineering work, then predict whether or not the user will click some certain ads. Here I'd like to summarize some popular traditional solutions and recent deep learing technics.

Contents

1 Online Advertising and Linear Model Sparsity

For learning in massive scale, generalized linear models have many advantages. Easily implement, memory saving, efficient training on streaming data.

Supposing we choose logistic regression as our model. Given feature $x_t \in R^d$, model parameter w_t. we can predict $p_t = \sigma(w_t \cdot x_t)$, where $\sigma(x) = \frac{1}{1+exp(-x)}$

is the sigmoid function. And our observation $y_t \in \{0, 1\}$. The log loss can be easily denoted as :

$$\mathcal{L}(w_t) = -y_t log p_t - (1 - y_t) log(1 - p_t)$$

It is obvious that:

$$\frac{\partial \mathcal{L}}{\partial w} = (\sigma(w \cdot x_t) - y_t)x_t = (p_t - y_t)x_t$$

And this gradient is all we need for optimization purpose.

1.1 Online Gradient Descent (OGD)

OGD is essentially the same with SGD. OGD can provide excellent prediction accuracy with minimum computing resources. However, as x might have billions of dimensions, the size of the final model is another key consideration. Here we want the w be sparse, while keep the accuracy or little accuracy loss.

However, OGD is not good at producing sparse result. In fact, simply adding subgradient penalty will never induce sparsity.

1.2 Truncated Gradient (TG)

There is a very simple solution to produce sparse solution. Just rounding small weights to 0. but doing so may cause some problems, because a weight may be useless or small because it has been updated only once, or at the begining of the training. Truncated gradient[1, 3] introduce another natural method to round small coefficients.

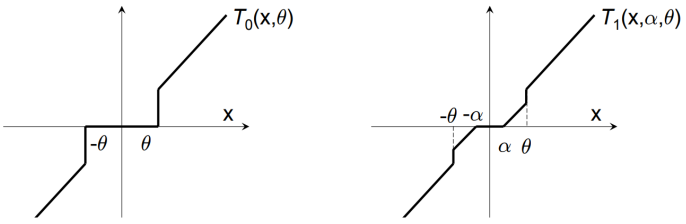

Figure 1: Plots for the truncation functions below.

For each K steps, perform simple coefficient rounding.

$$T_0(v_j, \theta) = \begin{cases} 0 & if \ |v_j| \leq \theta \\ v_j & otherwise \end{cases}$$

For each K steps, truncate gradient.

$$T_1(v_j, \alpha, \theta) = \begin{cases} max(0, v_j - \alpha) & if \ v_j \in [0, \theta] \\ min(0, v_j + \alpha) & if \ v_j \in [-\theta, 0] \\ v_j & otherwise \end{cases}$$

1.3 Forward-Backward Splitting (FOBOS)

In FOBOS[2], the problem is considered as two parts,

$$f(w) + r(w)$$

where f is the empirical loss and r is a regularization term that penalized for excessively complex vectors. FOBOS updates the w in 2 steps, forward-looking subgradients and forward-backward spliting. May sound complex, Actually it's very simple.

$$w_{t+\frac{1}{2}} = w_t - \eta_t g_t$$

$$w_{t+1} = \arg\min_{w_t} \left\{ \frac{1}{2} ||w_t - w_{t+\frac{1}{2}}||^2 + \eta_{t+\frac{1}{2}} r(w_t) \right\}$$

The first step is apply the gradient to the weight. The second step mainly contributes to the sparsity.It can be viewed as (i)let the updated weight be close to the previous one (ii) attain a low complexity.

1.4 Regularized Dual Averaging Algorithm (RDA)

RDA[6] condsider the learning variables are adjusted by solving a simple minimization problem that involves the running average of all past subgradients of the loss function and the whole regularization term, not just its subgradient.

$$w_{t+1} = \arg\min_{w} \{ \frac{1}{t} \sum_{i=1}^{t} \langle g_i, w \rangle + \Psi(w) + \frac{\beta_t}{t} h(w) \}$$

At each iteration, this method minimizes the sum of three items: (i) all previous subgradients (dual average). (ii)the original regularization funciton ($\lambda ||w||_1$). (iii) an additional strongly convex regularization term($\frac{1}{2}||w||_2^2$). It can be denoted as :

$$w_{t+1} = \arg\min_{w} \{ \frac{1}{t} \sum_{i=1}^{t} \langle g_i, w \rangle + \lambda ||w||_1 + \frac{1}{2} ||w - 0||_2^2 \}$$

In result, RDA producs even better accuracy vs sparsity tradeoffs than FOBOS.

1.5 Follow the Proximally Regularized Leader (FTRL-Proximal)

FTRL [4] is a hybrid of FOBOS and RDA, and significantly outperforms both on large real-world dataset.

$$w_{t+1} = \arg\min_{w} \{ g_{1:t} \cdot w + t\lambda ||w||_1 + \frac{1}{2} \sum_{s=1}^{t} \sigma_s ||w - w_s||^2 \}$$

where σ is the learning rate schedule, such that $\sigma_{1:t} = \frac{1}{\eta_t}$. The update of this fomula is actually very simple. As the fomula above can also be written as :

$$(g_{1:t} - \sum_{s=1}^{t} \sigma_s w_s) \cdot w + \frac{1}{\eta_t}||w||_2^2 + \lambda_1 ||w||_1 + const$$

If we store $z_{t-1} = g_{1:t-1} - \sum_{s=1}^{t-1} \sigma_s w_s$ in memory. At each round t, we update z_t:

$$z_t = z_{t-1} + g_t + (\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}})w_t$$

and the w_{t+1} can be solved in closed form on:

$$w_{t+1} = \begin{cases} 0 & if\ |z_t| \leq \lambda_1 \\ -\eta_t(z_t - sgn(z_t)\lambda_1) & otherwise \end{cases}$$

Thus FTRL-Proximal stores z in memory, whereas OGD stores w.

Per-coordinate leaning Standard online learning gradient descent suggest using a global learning rate schedule $\eta = \frac{1}{\sqrt{t}}$, which is common for all coordinates. As some coordinate updates frequently and some not. It makes more sense to represent the gradient per coordinate:

$$\eta_{t,i} = \frac{\alpha}{\beta + \sqrt{\sum_{s=1}^{t} g_{s,i}^2}}$$

β is simply for numeric stable sake, ensures that early learning rate not too high. $\beta = 1$ is usually good enough.

2 Collaborative Filtering

2.1 Factorization Machines (FM)

In recommender system, we may want the model to be able to estimate interactions between large feature dimensions, while keeping the sparsity. FM [5] is designed to deal with highly sparse features.It has linear complexity, and can work with any real valued feature vector. It can be used in many prediction task like regression, binary classification and ranking.

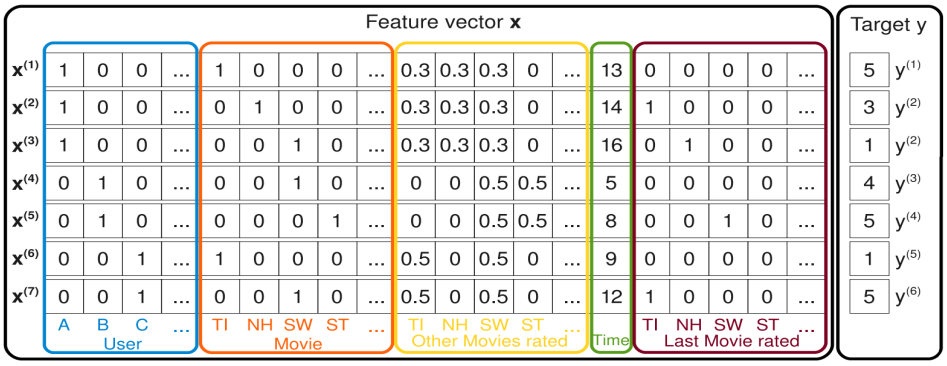

Figure 2: Sparse user features and target y.

Model equation Given $x \in R^n$, the model parameters that have to be estimated are $w_0 \in R, \mathbf{w} \in R^n, \mathbf{V} \in R^{n \times k}$.

$$\widehat{y} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j$$

where

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle = \sum_{m=1}^{k} v_{i,m} \cdot v_{j,m}$$

Based on the theory that for any positive definite matrix (semi-positive definite) $\mathbf{w}$, there exists a matrix $\mathbf{v}$ such that $\mathbf{w} = \mathbf{v} \cdot \mathbf{v}^{\mathrm{T}}$ if k is not limited. In practice, we often pick a small k for better generalization and thus improved interaction matrices under sparsity.

Computation If we follow the straight forward computation, the cost will be $O(kn^2)$. We can apply some mathematics transformations to reduce the computation to linear time $O(kn)$.

$$\begin{aligned}
\sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j &= \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j - \frac{1}{2} \sum_{i=1}^{n} \langle \mathbf{v}_i, \mathbf{v}_i \rangle x_i x_i \\
&= \frac{1}{2} \sum_{m=1}^{k} \left(\left(\sum_{i=1}^{n} v_{i,m} x_i \right)^2 - \sum_{i=1}^{n} v_{i,m}^2 x_i^2 \right)
\end{aligned}$$

2.1.1 ALS

2.2 FFM

3 Incorporate with Deep Learing

3.1 FNN

3.2 PNN

3.3 Deep and Wide

3.4 DeepFM

3.5 DNN

References

[1] John C. Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[2] John C. Duchi and Yoram Singer. Efficient online and batch learning using forward backward splitting. *Journal of Machine Learning Research*, 10:2899–2934, 2009.

[3] John Langford, Lihong Li, and Tong Zhang. Sparse online learning via truncated gradient. *Journal of Machine Learning Research*, 10:777–801, 2009.

[4] H. Brendan McMahan. Follow-the-regularized-leader and mirror descent: Equivalence theorems and implicit updates. *CoRR*, abs/1009.3240, 2010.

[5] Steffen Rendle. Factorization machines. In Geoffrey I. Webb, Bing Liu, Chengqi Zhang, Dimitrios Gunopulos, and Xindong Wu, editors, *ICDM 2010, The 10th IEEE International Conference on Data Mining, Sydney, Australia, 14-17 December 2010*, pages 995–1000. IEEE Computer Society, 2010.

[6] Lin Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11:2543–2596, 2010.