点击率预估算法：FM与FFM

原创 jediael_lu 2017-09-01 14:17:03 31113 已收藏 37 版权

分类专栏： 8.CTR_推荐系统 文章标签： 计算广告 FFM FTRL 点击率预估 FM

文章目录

1、FM

1.1 背景

1.1.1 线性模型

常见的线性模型，比如线性回归、逻辑回归等，它只考虑了每个特征对结果的单独影响，而没有考虑特征间的组合对结果的影响。

对于一个有n维特征的模型，线性回归的形式如下：

$$\begin{aligned} f(x) &= \omega_0 + \omega_1 x_1 + \omega_2 x_2 + ... + \omega_n x_n \\ &= \omega_0 + \sum_{i=1}^{n} \omega_i x_i \end{aligned} \qquad (1)$$

其中$(\omega_0, \omega_1 ... \omega_n)$为模型参数，$(x_1, x_2 ... x_n)$为特征。
从(1)式可以看出来，模型的最终计算结果是各个特征的独立计算结果，并没有考虑特征之间的相互关系。

举个例子，我们认为“USA”与"Thanksgiving"，"China"与“Chinese new year”这样的组合特征是很有意义的，在这样的组合特征下，会对某些商品表现出更强的购买意愿，而单独考虑国家及节日都是没有意义的。

1.1.2 二项式模型

我们在（1）式的基础上，考虑任意2个特征分量之间的关系，得出以下模型：

$$f(x) = \omega_0 + \sum_{i=1}^{n} \omega_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \omega_{ij} x_i x_j \qquad (2)$$

这个模型考虑了任意2个特征分量之间的关系，但并未考虑更高阶的关系。
该模型涉及的参数数量为：

$$1 + n + \frac{n(n-1)}{2} = \frac{1}{2}(n^2 + n + 2) \qquad (3)$$

对于参数ω_i的训练，只要这个样本中对应的x_i不为0，则可以完成一次训练。
但对于参数ω_{ij}的训练，需要这个样本中的x_i和x_j同时不为0，才可以完成一次训练。
在数据稀疏的实际应用场景中，二次项ω_{ij}的训练是非常困难的。因为每个ω_{ij}都需要大量x_i和x_j都不为0的样本。但在数据稀疏性比较明显的样本中，x_i和x_j都不为0的样本会非常稀少，这会导致ω_{ij}不能得到足够的训练，从而不准确。

1.2 FM

1.2.1 FM基本原理

为了解决二项式模型中由于数据稀疏引起的训练不足的问题，我们为每个特征维度x_i引入一个辅助向量：

$$V_i = (v_{i1}, v_{i2}, v_{i3}, ..., v_{ik})^T \in \mathbb{R}^k, i = 1, 2, 3, ..., n \qquad (4)$$

其中k为辅助变量的维度，依经验而定，一般而言，对于特征维度足够多的样本，$k << n$。
将ω_{ij}表示为：

$$\omega_{ij} = V_i^T V_j = \sum_{l=1}^{k} v_{il} v_{jl} \qquad (5)$$

简单的说，我们不再简单的使用样本训练具体的ω_{ij}，而是先

（5）求出最终的ω_{ij}。

具体而言，$\omega_{ij}=V_i^T V_j$与$\omega_{hi}=V_h^T V_i$有相同的项V_i，也就是只要样本中的x_i不为0，且最少具有一个其它特征，则这个样本则可用于训练V_i，这就解决了数据稀疏性导致的问题。

于是，在FM中，模型可以表达为：

$$f(x)=\omega_0+\sum_{i=1}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(V_i^T V_j)x_i x_j \qquad (6)$$

1.2.2 数据分析

我们的目标是要求得以下交互矩阵W：

$$W=\begin{pmatrix}\omega_{11} & \omega_{12} & \cdots & \omega_{1n}\\ \omega_{21} & \omega_{22} & \cdots & \omega_{2n}\\ \vdots & \vdots & \ddots & \vdots\\ \omega_{n1} & \omega_{n2} & \cdots & \omega_{nn}\end{pmatrix}_{n\times n} \qquad (7)$$

由于直接求解W不方便，因此我们引入隐变量V：

$$V=\begin{pmatrix}v_{11} & v_{12} & \cdots & v_{1k}\\ v_{21} & v_{22} & \cdots & v_{2k}\\ \vdots & \vdots & \ddots & \vdots\\ v_{n1} & v_{n2} & \cdots & v_{nk}\end{pmatrix}_{n\times k}=\begin{pmatrix}V_1^T\\ V_2^T\\ \cdots\\ V_n^T\end{pmatrix} \qquad (8)$$

令

$$V\,V^T=W \qquad (9)$$

如果我们先得到V，则可以得到W了。
现在只剩下一个问题了，是否一个存在V，使得上述式（9）成立。
理论研究表明：当k足够大时，对于任意对称正定的实矩阵$W\in\mathbb{R}^{n\times n}$，均存在实矩阵$V\in\mathbb{R}^{n\times k}$，使得$ W=VV^T$。

理论分析中要求参数k足够的大，但在高度稀疏数据的场景中，由于 没有足够的样本，因此k通常取较小的值。事实上，对参数k的限制，在一定程度上可以提高模型的泛化能力。

1.2.3参数个数

假设样本中有n个特征，每个特征对应的隐变量维度为k，则参数个数为$1+n+nk$。
正如上面所言，对于特征维度足够多的样本，$k<<n$。

1.2.4 计算时间复杂度

下面我们分析一下已经知道所有参数，代入式（6）计算预测值时的时间复杂度。从式（6）中一看，

$$f(x)=\omega_0+\sum_{i=1}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(V_i^T V_j)x_i x_j \qquad (6)$$

可以看出时间复杂度是$O(kn^2)$。但我们对上述式子的最后一项作变换后，可以得出一个$O(kn)$的时间复杂度表达式。

$$\begin{aligned}\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(V_i^T V_j)x_i x_j&=\frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}(V_i^T V_j)x_i x_j-\sum_{i=1}^{n}(V_i^T V_i)x_i x_i\right)\\&=\frac{1}{2}\left(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1}^{k}v_{il}v_{jl}x_i x_j-\sum_{i=1}^{n}\sum_{l=1}^{k}v_{il}^2x_i^2\right)\\&=\frac{1}{2}\sum_{l=1}^{k}\left(\sum_{i=1}^{n}(v_{il}x_i)\sum_{j=1}^{n}(v_{jl}x_j)-\sum_{i=1}^{n}v_{il}^2x_i^2\right)\\&=\frac{1}{2}\sum_{l=1}^{k}\left(\left(\sum_{i=1}^{n}(v_{il}x_i)\right)^2-\sum_{i=1}^{n}v_{il}^2x_i^2\right)\end{aligned} \qquad (10)$$

上述式子中的$\sum_{i=1}^{n}(v_{il}x_i)$只需要计算一次就好，因此，可以看出上述模型的复杂度为$O(kn)$。
也就是说我们不要直接使用式（6）来计算预测结果，而应该使用式（10），这样的计算效率更高。

1.2.5 梯度

FM有一个重要的性质：multilinearity：若记$\Theta=(\omega_0,\omega_1,\omega_2,...,\omega_n,v_{11},v_{12},...,v_{nk})$表示FM模型的所有参数，则对于任意的$\theta\in\Theta$，存在与$\theta$无关的$g(x)$与$h(x)$，使得式（6）可以表示为：

$$f(x)=g(x)+\theta h(x) \qquad (11)$$

从式（11）中可以看出，如果我们得到了$g(x)$与$h(x)$，则对于参数θ的梯度为$h(x)$。下面我们分情况讨论。

- 当$\theta=\omega_0$时，式（6）可以表示为：

$$f(x)=\sum_{i=1}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(V_i^T V_j)x_i x_j+\omega_0\times 1 \qquad (12)$$

上述中的蓝色表示$g(x)$，红色表示$h(x)$。下同。
从上述式子可以看出此时的梯度为1.

- 当$\theta=\omega_l,l\in(1,2,...,n)$时，

$$f(x)=\omega_0+\sum_{i=1,i\neq l}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(V_i^T V_j)x_i x_j+\omega_l\times x_l \qquad (13)$$

此时梯度为x_l。

- 当$\theta=v_{lm}$时

$$f(x)=\omega_0+\sum_{i=1}^{n}\omega_i x_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\left(\sum_{s=1,i\neq lm,j\neq lm}^{k}v_{is}v_{js}\right)x_i x_j+v_{lm}\times x_l\sum_{i=1,i\neq l}^{n}v_{im}x_i \qquad (14)$$

此时梯度为$x_l\sum_{i\neq l}v_{im}x_i$。

综合上述结论，$f(x)$关于θ的偏导数为：

$$\frac{\partial f(x)}{\partial\theta}=\begin{cases}1, & \theta=\omega_0\\ x_l, & \theta=\omega_l,l\in(1,2,...,n)\\ x_l\sum_{i=1,i\neq l}^{n}v_{im}x_i & \theta=v_{lm}\end{cases} \qquad (15)$$

1.2.6 训练时间复杂度

由上述式（15）可以得到：

$$x_l \sum_{i=1,i\neq l}^{n} v_{im}x_i = x_l \sum_{i=1}^{n} v_{im}x_i - v_{lm}x_l^2 \qquad (16)$$

对于上式中的前半部分$\sum_{i=1}^{n} v_{im}x_i$，对于每个样本只需要计算一次，所以时间复杂度为$O(n)$，对于k个隐变量的维度分别计算一次，则复杂度为$O(kn)$。其它项的时间复杂度都小于这一项，因此，模型训练的时间复杂度为$O(kn)$。

详细一点解释：
（1）我们首先计算$\sum_{i=1}^{n} v_{im}x_i$，时间复杂度为n，这个值对于所有特征对应的隐变量的某一个维度是相同的。我们设这值为C。

（2）计算每一个特征对应的$x_l\sum_{i=1}^n v_{im}x_i-v_{lm}x_l^2 =Cx_l-v_{lm}x_l^2 $，由于总共有n个特征，因此时间复杂度为n，至此，总的时间复杂度为n+n。

（3）上述只是计算了隐变量的其中一个维度，我们总共有k个维度，因此总的时间复杂度为$k(n+n) = O(kn)$.

2、FFM

2.1 背景及基本原理

在FM模型中，每一个特征会对应一个隐变量，但在FFM模型中，认为应该将特征分为多个field，每个特征对应每个field分别有一个隐变量。

举个例子，我们的样本有3种类型的字段：publisher, advertiser, gender，分别可以代表媒体，广告主或者是具体的商品，性别。其中publisher有5种数据，advertiser有10种数据，gender有男女2种，经过one-hot编码以后，每个样本有17个特征，其中只有3个特征非空。

如果使用FM模型，则17个特征，每个特征对应一个隐变量。
如果使用FFM模型，则17个特征，每个特征对应3个隐变量，即每个类型对应一个隐变量，具体而言，就是对应publisher, advertiser, gender三个field各有一个隐变量。

2.2模型与最优化问题

2.2.1 模型

根据上面的描述，可以得出FFM的模型为：

$$f(x) = \omega_0 + \sum_{i=1}^{n} \omega_i x_i + \sum_{j1=1}^{n-1} \sum_{j2=j1+1}^{n} (V_{j1,f2}^T V_{j2,f1}) x_{j1} x_{j2} \qquad (17)$$

其中$j1, j2$表示特征的索引。我们假设$j1$特征属于$f1$这个field，$j2$特征属于$f2$这个field，则$V_{j1,f2}$表示$j1$这个特征对应$f2$($j2$所属的field)的隐变量，同时$V_{j2,f1}$表示$j2$这个特征对应$f1$($j1$所属的field)的隐变量。

事实上，在大多数情况下，FFM模型只保留了二次项，即：

$$\phi(V, x) = \sum_{j1=1}^{n-1} \sum_{j2=j1+1}^{n} (V_{j1,f2}^T V_{j2,f1}) x_{j1} x_{j2} \qquad (18)$$

2.2.2 最优化问题

根据逻辑回归的损失函数及分析，可以得出FFM的最优化问题为：

$$\min \frac{\lambda}{2} ||V||_2^2 + \sum_{i=1}^{m} \log(1 + exp(-y_i \phi(V, x))) \qquad (19)$$

上面加号的前面部分使用了L2范式，后面部分是逻辑回归的损失，y_i表示样本的真实值（如是否点击的-1/1），$\phi(V, x)$表示使用当前的

注意，以上的损失函数适用于样本分布为{-1,1}的情况。

2.2.3 自适应学习率

与FTRL一样，FFM也使用了累积梯度作为学习率的一部分，即：

$$V_{j1,f2} = V_{j1,f2} - \frac{\eta}{\sqrt{1 + \sum_t (g_{v_{j1,f2}}^t)^2}} g_{v_{j1,f2}} \qquad (20)$$

其中$g_{v_{j1,f2}}$表示对于$V_{j1,f2}$这个变量的梯度向量，因为V_{j1,f2}是一个向量，因此g_{v_{j1,f2}}也是一个向量，尺寸为隐变量的维度大小，即k。而\sum_t(g_{v_{j1,f2}}^t)^2$表示从第一个样本到当前样本一直以来的累积梯度平方和。

2.2.4 FFM算法的最终形式

$$(V_{j1,f2})_d = (V_{j1,f2})_{d-1} - \frac{\eta}{\sqrt{(G_{j1,f2})_d}} \cdot (g_{j1,f2})_d$$
$$(V_{j2,f1})_d = (V_{j2,f1})_{d-1} - \frac{\eta}{\sqrt{(G_{j2,f1})_d}} \cdot (g_{j2,f1})_d$$

其中G为累积梯度平方：

$$(G_{j1,f2})_d = (G_{j1,f2})_{d-1} + (g_{j1,f2})_d^2$$
$$(G_{j2,f1})_d = (G_{j2,f1})_{d-1} + (g_{j2,f1})_d^2$$

g为梯度，比如$g_{j1,f2}$为$j1$这个特征对应$f2$这个field的梯度向量：

$$g_{ji,f2} = \lambda \cdot V_{ji,f2} + \kappa \cdot V_{j2,f1}$$
$$g_{j2,f1} = \lambda \cdot V_{j2,f1} + \kappa \cdot V_{j1,f2}$$

其中κ为：

$$\kappa = \frac{\partial \log(1 + exp(-y_i \phi(V, x)))}{\partial \phi(V, x)} = \frac{-y}{1 + \exp(y \phi(V, x))}$$

2.3完整算法流程

使用随机梯度下降（SGD）训练FFM模型的完整过程如下：

2.3.1 计算梯度

对于每一个样本的每一对特征组合都要计算以下梯度向量。

$$g_{ji,f2} = \lambda \cdot V_{ji,f2} + \kappa \cdot V_{j2,f1}$$
$$g_{j2,f1} = \lambda \cdot V_{j2,f1} + \kappa \cdot V_{j1,f2} \qquad (21)$$

其中κ为式(19)后半部分对应的梯度，即：

$$\kappa = \frac{\partial \log(1 + exp(-y_i \phi(V, x)))}{\partial \phi(V, x)} = \frac{-y}{1 + \exp(y \phi(V, x))} \qquad (22)$$

每重申一次，g与V都是k维的向量，在python中可以作为一个向量计算，在java/c++等需要通过一个循环进行计算。

详细推导（21）式如下：
（1）在SGD中，式（19）可以转化为：

$$\min \frac{\lambda}{2} ||V||_2^2 + \log(1 + exp(-y_i$$

（2）上式对$V_{j1,f2}$求偏导，可得：

$$\begin{aligned}&\frac{\partial\frac{\lambda}{2}||V||_2^2+\log(1+exp(-y_i\phi(V,x)))}{\partial V_{j1,f2}}\\&=\lambda\cdot V_{j1,f2}+\frac{\partial\log(1+exp(-y_i\phi(V,x)))}{\partial V_{j1,f2}}\\&=\lambda\cdot V_{j1,f2}+\frac{\partial\log(1+exp(-y_i\phi(V,x)))}{\partial\phi}\cdot\frac{\partial\phi}{V_{j1,f2}}\\&=\lambda\cdot V_{j1,f2}+\frac{-y}{1+\exp(y\phi(V,x))}\cdot V_{j2,f1}\end{aligned}\tag{24}$$

2.3.2 计算累积梯度平方和

计算从第一个样本，到当前样本（第d个）以来的累积梯度平方和：

$$\begin{aligned}(G_{j1,f2})_d&=(G_{j1,f2})_{d-1}+(g_{j1,f2})_d^2\\(G_{j2,f1})_d&=(G_{j2,f1})_{d-1}+(g_{j2,f1})_d^2\end{aligned}\tag{26}$$

2.3.3 更新隐变量

$$\begin{aligned}(V_{j1,f2})_d&=(V_{j1,f2})_{d-1}-\frac{\eta}{\sqrt{(G_{j1,f2})_d}}\cdot(g_{j1,f2})_d\\(V_{j2,f1})_d&=(V_{j2,f1})_{d-1}-\frac{\eta}{\sqrt{(G_{j2,f1})_d}}\cdot(g_{j2,f1})_d\end{aligned}\tag{26}$$

2.3.4 关于初始参数的设定

文献1中如此建议：

（1）η：没有具体的建议，用户根据经验指定即可，一般会取0.1，0.01，0.001。

（2）V：在区间$[0,1/\sqrt{k}]$间的随机值，均匀分布即可。

（3）G：设置为1，以避免$(G_{j1,f2})_d^{-\frac{1}{2}}$出现很大的值。

2.4 时间复杂度

2.4.1 计算时间复杂度

由于式(18)无法做类似于式（10）的简化，因此FFM的计算时间复杂度为$O(kn^2)$。

2.4.2 训练时间复杂度

由于训练时，需要先根据式（18）计算ϕ，复杂度为$O(kn^2)$，计算得到ϕ后，还需要按照式（22）计算1次，按照式（21）计算2k次，按照式（23）计算2k次，按照式（24）计算2k次，也就是说，总的训练时间复杂度为：

$$O(kn^2)+1+2k+2k+2k=O(kn^2)$$

因此，训练时间复杂度为$O(kn^2)$。

2.5 计算速度优化

2.5.1 openMP

OpenMP提供的这种对于并行描述的高层抽象降低了并行编程的难度和复杂度，这样程序员可以把更多的精力投入到并行算法本身，而非其具体实现细节。对基于数据分集的多线程程序设计，OpenMP是一个很好的选择。同时，使用OpenMP也提供了更强的灵活性，可以较容易的适应不同的并行系统配置。线程粒度和负载平衡等是传统多线程程序设计中的难题，但在了部分这两方面的工作。

OpenPM原生支持C/C++/Fortran，但java可以通过jomp等引入，未测试。

2.5.2 SSE3

SSE3 中13个新指令的主要目的是改进线程同步和特定应用程序领域，例如媒体和游戏。这些新增指令强化了处理器在浮点转换至整数、复杂算法、视频编码、SIMD浮点寄存器操作以及线程同步等五个方面的表现，最终达到提升多媒体和游戏性能的目的。Intel是从Prescott核心的Pentium 4开始支持SSE3指令集的，而AMD则是从2005年下半年Troy核心的Opteron开始才支持SSE3的。但是需要注意的是，AMD所支持的SSE3与Intel的SSE3并不完全相同，主要是删除了针对Intel超线程技术优化的部分指令。
SSE3指令采用128位的寄存器，可以同时操作4个单精度浮点数或者整数，因此非常类似于向量运算。这对于有大量向量计算的的FFM模型是有用的。
但事实上，计算ϕ是几乎无用，而这是最耗时间的部分。

2.5.3 ParameterServer

https://www.zybuluo.com/Dounm/note/517675
Paraeter Server框架中，每个server都只负责分到的部分参数（server共同维持一个全局共享参数）。server节点可以和其他server节点通信，每个server负责自己分到的参数，server group共同维持所有参数的更新。server manage node负责维护一些元数据的一致性，例如各个节点的状态，参数的分配情况。

2.6模型优化

2.6.1 特征编码连续

如果特征的编码不连续，比如编码是有意义的，或者预留空间给之后的编码。如果直接使用最大编码值的作为参数数据尺寸，则会导致大量内存空间的浪费，因此有2种解决方案：
（1）使用hashmap，而非数组。
（2）将有意义的编码映射到一个连续的编码空间。
目前我们使用方式（1），理论上方式（2）的计算速度会更快。

2.6.2 一次项缺失的影响

正如上面式（18）所言，我们经常会忽略了一次项的影响，因此我们可以为每个样本加上一个辅助特征，这相特征的值恒为1，这相当于引入了一次项。

2.6.3 样本归一化

文献1还建议，将样本向量的长度归一化后，性能有少量的提升。

$$R[i]=\frac{1}{||X||}$$

2.6.4 特征归一化

某些特征（如购买个数）有可能很大，而一些类别参数则恒为1，这将导致不同特征最终对模型的影响相关很大，这很不合理。