机器学习算法系列（26）：因子分解机（FM）与场感知分解机（FFM）

本文转载自美团点评技术团队

FM和FFM模型是最近几年提出的模型，凭借其在数据量比较打并且特征稀疏的情况下，忍让能够得到优秀的性能和效果，屡次在各大公司举办的CTR预估比赛中获得不错的战绩。

在计算广告领域，点击率CTR（click-through rate）和转化率CVR（conversion rate）是衡量广告流量的两个关键指标。准确的估计CTR、CVR对于提高流量的价值，增加广告收入有重要的指导作用。预估CTR、CVR，业界常用的方法由人工特征工程+LR（Logistic Regression）、GBDT（Gradient Boosting Decision Tree）+LR、FM（Factorization Machine）和FFM（Field-aware Factorization Machine）模型。在这些模型中，FM和FFM近年来表现突出，分别在Criteo和Avazu举办的CTR预测竞赛中夺得冠军。

本文基于对FFM模型的深度调研和使用经验，从原理、实现和应用几个方面对FFM进行探讨，希望能够从原理上解释FFM模型在点击率预估上取得优秀效果的原因。因为FFM是在FM的基础上改进得来的，所以，我们首先引入FM模型。

一、FM（因子分解机）

1.1 FM的原理及推导

因子分解机（Factorization Machine，简称FM），又称分解机。是由德国康斯坦茨大学的Steffen Rendle（现任职于Google）于2010年最早提出的，旨在解决大规模稀疏数据下的特征组合问题。在系统介绍FM之前，先了解一下在实际场景中，稀疏数据是怎样产生的。

假设一个广告分类的问题，根据用户和广告位相关的特征，预测用户是否点击了广告。元数据如下：

Clicked?	Country	Day	Ad_type
1	USA	26/11/15	Movie
0	China	1/7/14	Game
1	China	19/2/15	Game

“Clicked? ”是label，Country、Day、Ad_type是特征。由于三种特征都是categorical类型的，需要经过独热编码（One-Hot Encoding）转换成数值型特征。

Clicked?	Country=USA	Country=China	Day=26/11/15	Day=1/7/14	Day=19
1	1	0	1	0	0
0	0	1	0	1	0
1	0	1	0	0	1

由上表可以看出，经过One-Hot编码之后，大部分样本数据特征是比较稀疏的。上面的样例中，每个样本有7维特征，但平均仅有3维特征具有非零值。实际上，这种情况并不是此例独有的，在真实应用场景中这种情况普遍存在。例如，CTR/CVR预测时，用户的性别、职业、教育水平、品类偏好、商品的品类等，经过One-Hot编码转换后都会导致样本数据的稀疏性。特别是商品品类这种类型的特征，如商品的末级品类约有550个，采用One-Hot编码生成550个数值特征，但每个样本的这550个特征，有且仅有一个是有效的（非零）。由此可见，数据稀疏性是实际问题中不可避免的挑战。

One-Hot编码的另一个特点就是导致特征空间大。例如，商品品类有550维特征，一个categorical特征转换为550维数值特征，特征空间剧增。

同时通过观察大量的样本数据可以发现，某些特征经过关联之后，与label之间的相关性就会提高。如：“USA”与“Thanksgiving”、“China”与“Chinese New Year”这样的关联特征，对用户的点击有着正向的影响。换句话说，来自“China”的用户很可能会在“Chinese New Year”有大量的浏览、购买行为，而在“Thanksgiving”却不会有特别的消费行为。这种关联特征与label的正向相关性在实际问题中是普遍存在的，如“化妆品”类商品与“女”性，“球类运动配件”的商品与“男”性，“电影票”的商品与“电影”品类偏好等。因此，引入两个特征的组合是非常有意义的。

表示特征之间的关联，最直接的方法的是构造组合特征。样本中特征之间的关联信息在one-hot编码和浅层学习模型（如LR、SVM）是做不到的。目前工业界主要有两种手段得到组合特征：

- 1）人工特征工程（数据分析＋人工构造）；
- 2）通过模型做组合特征的学习（深度学习方法、FM/FFM方法）

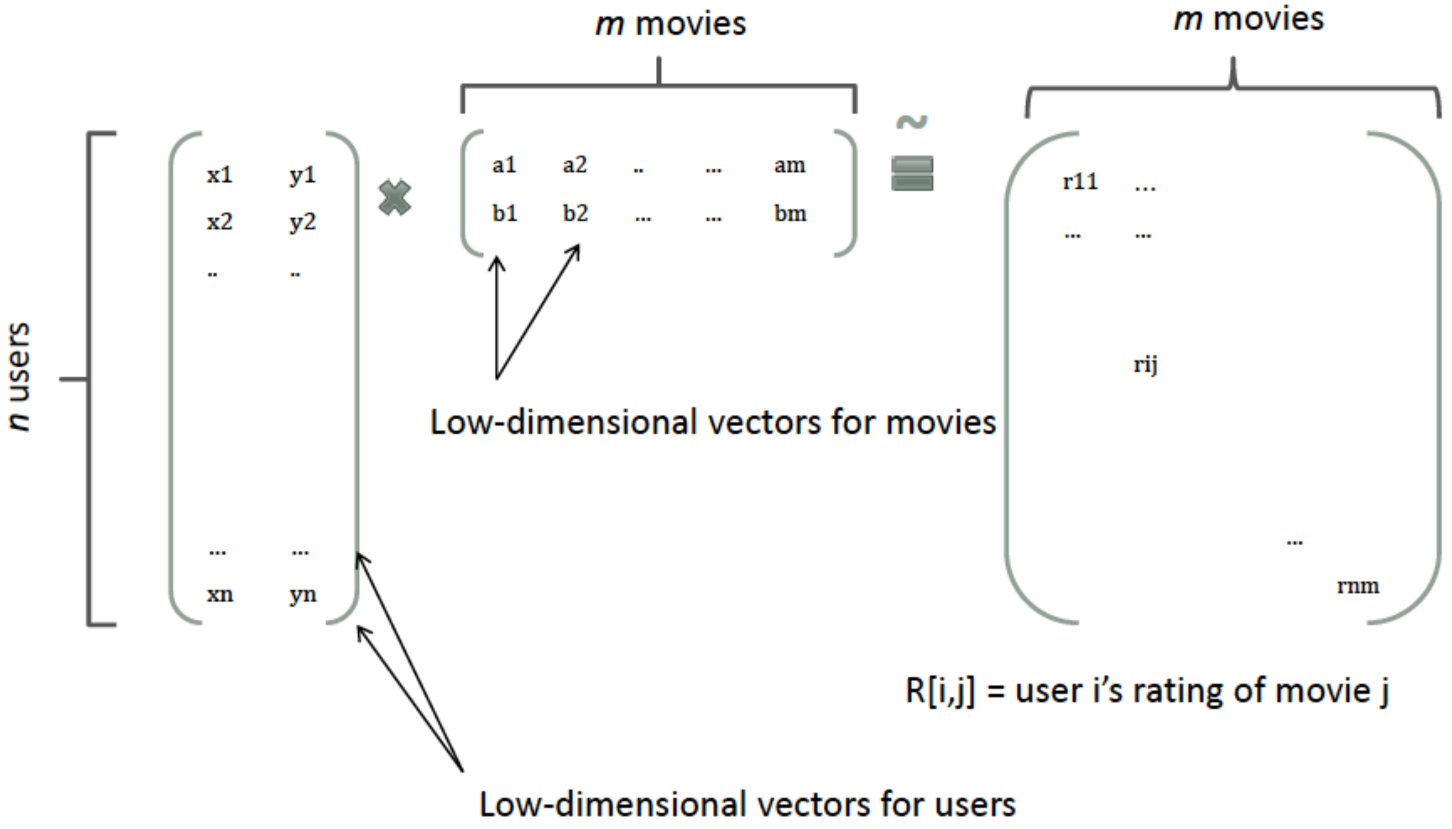
本章主要讨论FM和FFM用来学习特征之间的关联。多项式模型是包含特征组合的最直观的模型。在多项式模型中，特征 x_i 和 x_j 的组合采用 x_i 表示，即 x_i 和 x_j 都非零时，组合特征 x_ix_j 才有意义。从对比的角度，本文只讨论二阶多项式模型。模型的表达式如下：

$$y(x) = w_0 + \sum_{i=1}^{n} w_ix_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} w_{ij}x_ix_j$$

其中，n代表样本的特征数量，x_i是第i个特征的值，w_0、 w_i、 w_{ij}是模型的参数。

从这个公式可以看出，组合特征的参数一共有$\frac{n(n-1)}{2}$个，任意两个参数都是独立的。然而，在数据稀疏性普遍存在的实际应用场景中，二次项参数的训练是很困难的。其原因是，回归模型的参数w的学习结果就是从训练样本中计算充分统计量（凡是符合指数族分布的模型都具有此性质），而在这里交叉项的每一个参数w_{ij}的学习过程需要大量的x_i、x_j同时非零的训练样本数据。由于样本数据本来就很稀疏，能够满足“x_i和x_j都非零”的样本数就会更少。训练样本不充分，学到的参数w_{ij}就不是充分统计量结果，导致参数w_{ij}不准确，而这会严重影响模型预测的效果（performance）和稳定性。

那么，如何解决二次项参数的训练问题呢？矩阵分解提供了一种解决思路。在Model-based的协同过滤中，一个rating矩阵可以分解为user矩阵和item矩阵，每个user和item都可以采用一个隐向量表示。比如在下图中的例子，我们把每个user表示成一个二维向量，同时把每个item表示成一个二维向量，两个向量点积就是矩阵中user对item的打分。



类似地，所有二次项参数 w_{ij}可以组成一个对称阵 W（为了方便说明FM的由来，对角元素可以设置为正实数），那么这个矩阵就可以分解为 $W = V^TV$，V 的第j列便是第 j 维特征的隐向量。换句话说，每个参数 $w_{ij} = \langle v_i, v_j\rangle$，这就是FM模型的核心思想。因此，FM的模型方程为（本文不讨论FM的高阶形式）

$$y(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n}\sum_{j=i+1}^{n} \langle vi, vj\rangle x_i x_j \quad \cdots \quad (2)$$

其中，v_i是第i维特征的隐向量，$\langle \cdot, \cdot \rangle$代表向量点积，计算公式为

$$\langle v_i, v_j \rangle = \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f}$$

隐向量的长度为$k(k << n)$，包含k个描述特征的因子。
具体解读一下这个公式

- 线性模型+交叉项：直观地看FM模型表达式，前两项是线性回归模型的表达式，最后一项是二阶特征交叉项（又称组合特征项），表示模型将两个互异的特征分量之间的关联信息考虑进来。用交叉项表示组合特征，从而建立特征与结果之间的非线性关系。
- 交叉项系数 → 隐向量内积：由于FM模型是在线性回归基础上加入了特征交叉项，模型求解时不直接求特征交叉项的系数w_{ij}（因为对应的组合特征数据稀疏，参数学习不充分），故而采用隐向量的内积$\langle v_i, v_j \rangle$表示w_{ij}。具体的，FM求解过程中的做法是：对每一个特征分量x_i引入隐向量$v_i = (v_{i,1}, v_{i,2}, \cdots, v_{i,k})$，利用$v_i v_j^T$内积结果对交叉项的系数$w_{ij}$进行估计，公式表示：$\hat{w}_{ij} = v_i v_j^T$

根据上式，二次项的参数数量减少为kn个，远少于多项式模型的参数数量。

此外，参数因子化表示后，使得$x_h x_i$的参数与$x_i x_j$的参数不再相互独立。这样我们就可以在样本系数的情况下相对合理地估计FM模型交叉项的参数。具体地：

$$\langle v_h, v_i \rangle = \sum_{f=1}^{k} v_{h,f} \cdot v_{i,f}$$

$$\langle v_i, v_j \rangle = \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f}$$

$x_h x_i$与$x_i x_j$的系数分别为$\langle v_h, v_i \rangle$和$\langle v_i, v_j \rangle$，它们之间有共同项v_i，也就是说，所有包含x_i的非零组合特征（存在某个$j \neq i$,使得$x_i x_j \neq 0$）的样本都可以用来学习隐向量v_i，这在很大程度上避免了数据系数行造成参数估计不准确的影响。而在多项式模型中，w_{hi}和w_{ij}是相互独立的。

显而易见，公式(2)是一个通用的拟合方程，可以采用不同的损失函数用于解决回归、二元分类等问题，比如可以采用MSE（Mean Square Error）损失函数来求解回归问题，也可以采用Hinge、Cross-Entropy损失来求解分类问题。当然，在进行二元分类时，FM的输出需要经过Sigmoid变换，这与Logistic回归是一样的。

FM应用场景	损失函数	说明
回归	均方误差（MSE）损失	Mean Square Error，与平方误差类似
二类分类	Hinge/Cross-Entopy损失	分类时，结果需要做sigmoid变换

排序		

直观上看，FM的复杂度是$O(kn^2)$，但是，通过下面的等价转换，可以将FM的二次项化简，其复杂度可以优化到$O(kn)$，即：

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle v_i, v_j\rangle x_i, x_j = \frac{1}{2}\sum_{f=1}^{k}[(\sum_{i=1}^{n} v_{i,f}x_i)^2 - \sum_{i=1}^{n} v_{i,f}^2 x_i^2]$$

下面给出详细推导：

$$\sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle v_i, v_j\rangle x_i x_j = \frac{1}{2}\sum_{i=1}^{n}\sum_{f=1}^{n}\langle v_i, v_j\rangle x_i x_j - \frac{1}{2}\sum_{i=1}^{n}\langle v_i, v_i\rangle x_i x_i = \frac{1}{2}(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{f=1}^{k} v_{i,f}v_{j,f}x_i x_j - \sum_{i=1}^{n}\sum_{f=1}^{k} v_{i,f}v_{i,f}x_i x_i) =$$

解读第一步到第二部，这里用A表示系数矩阵V的上三角元素，B表示对角线上的交叉项系数。由于系数矩阵V是一个对称阵，所以下三角和上三角相等，有下式成立：

$$A = \frac{1}{2}(2A + B) - \frac{1}{2}B$$

其中，

$$A = \sum_{i=1}^{n}\sum_{j=i+1}^{n}\langle v_i, v_j\rangle x_i x_j, B = \sum_{i=1}^{n}\langle v_i, v_j\rangle x_i x_i$$

如果用随机梯度下降（SGD）法学系模型参数。那么模型各个参数的梯度如下：

$$\frac{\partial}{\partial\theta}y(x) = \begin{cases} 1,\ \ if\ \theta\ is\ w_0(\text{常数项}) \\ x_i,\ \ if\ \theta\ is\ w_i(\text{线性项}) \\ x_i\Sigma_{j=1}^{n} v_{j,f}x_j - v_{i,f}x_i^2,\ if\ \theta\ is\ v_{i,f}(\text{交叉项}) \end{cases}$$

其中，$v_{j,f}$是隐向量v_j的第f个元素。

由于$\Sigma_{j=1}^{n} v_{j,f}x_j$只与f有关，在参数迭代过程中，只需要计算第一次所有f的$\Sigma_{j=1}^{n} v_{j,f}x_j$，就能够方便地得到所有$v_{i,f}$的梯度。显然，计算所有f的$\Sigma_{j=1}^{n} v_{j,f}x_j$的复杂度是$O(kn)$；已知$\Sigma_{j=1}^{n} v_{j,f}x_j$时，计算每个参数梯度的复杂度是$O(n)$；得到梯度后，更新每个参数的复杂度是$O(1)$；模型参数一共有$nk + n + 1$个。因此，FM参数训练的时间复杂度为$O(kn)$

1.2 FM的优势

综上可知，FM算法可以再线性时间内完成模型训练，以及对新样本作出预测，所以说FM是一个非常高效的模型。FM模型的核心作用可以概括为以下三个：

- 1）FM降低了交叉项参数学习不充分的影响：one-hot编码后的样本数据非常稀疏，组合特征更是如此。为了解决交叉项参数学习不充分、导致模型有偏或不稳定的问题。作者借鉴矩阵分解的思路：每一维特征用k维的隐向量表示，交叉项的参数w_ij用对应特征隐向量的内积表示，即$\langle v_i, v_j \rangle$。这样参数学习由之前学习交叉项参数w_{ij}的过程，转变为学习n个单特征对应k维隐向量的过程。很明显，单特征参数（k维隐向量v_i）的学习要比交叉项参数w_{ij}学习的更加充分。示例说明： 假如有10w条训练样本，其中出现女性特征的样本数为3w，出现男性特征的样本数为7w，出现汽车特征的样本数为2000，出现化妆品的样本数为1000。特征共现的样本数如下：

共现交叉特征	样本数	注
<女性，汽车>	500	同时出现<女性，汽车>的样本数
<女性，化妆品>	1000	同时出现<女性，化妆品>的样本数
<男性，汽车>	1500	同时出现<男性，汽车>的样本数
<男性，化妆品>	0	样本中无此特征组合项

<女性，汽车>的含义是女性看汽车广告。可以看到，但特征对应的样本数远大于组合特征对应的样本数。训练时，但特征参数相比交叉项特征参数会学习地更充分。因此，可以说FM降低了因数据稀疏，导致交叉项参数学习不充分的影响。

- 2）FM提升了模型预估能力。依然看上面的示例，样本中没有没有<男性，化妆品>交叉特征，即没有男性看化妆品广告的数据。如果yoga多项式模型来建模，对应的交叉项参数$w_{男性，化妆品}$是学不出来的，因为数据中没有对应的共现交叉特征。那么多项式模型就不能对出现的男性看化妆品广告场景给出准确地预估。
  FM模型是否能得到交叉项参数$w_{男性，化妆品}$呢？答案是肯定的。由于FM模型是把交叉项参数用对应的特征隐向量内积表示，这里表示为$w_{男性，化妆品}=<v_{男性}，v_{化妆品}>$，即用男性特征隐向量$v_{男性}$和化妆品特征隐向量$v_{化妆品}$的内积表示交叉项参数$w_{男性，化妆品}$由于FM学习的参数就是单特征的隐向量，那么男性看化妆品广告的预估结果可以用$<v_{男性}，v_{化妆品}>$得到。这样，即便训练集中没有出现男性看化妆品广告的样本，FM模型仍然可以用来预估，提升了预估呢不给力。

- 3）FM提升了参数学习效率：这个显而易见，参数个数由$(n2+n+1)(n2+n+1)$变为$(nk+n+1)(nk+n+1)$个，模型训练复杂度也由$O(mn^2)$变为$O(mnk)$。mm为训练样本数。对于训练样本和特征数而言，都是线性复杂度。此外，就FM模型本身而言，它是在多项式模型基础上对参数的计算做了调整，因此也有人把FM模型称为多项式的广义线性模型，也是

恰如其分的。从交互项的角度看，FM仅仅是一个可以表示特征之间交互关系的函数表法式，可以推广到更高阶形式，即将多个互异特征分量之间的关联信息考虑进来。例如在广告业务场景中，如果考虑User-Ad-Context三个维度特征之间的关系，在FM模型中对应的degree为3。

最后一句话总结，FM最大特点和优势：**FM模型对稀疏数据有更好的学习能力，通过交互项可以学习特征之间的关联关系，并且保证了学习效率和预估能力。**

与其他模型相比，它的优势如下：

- FM是一种比较灵活的模型，通过合适的特征变换方式，FM可以模拟二阶多项式核的SVM模型、MF模型、SVD++模型等；
- 相比SVM的二阶多项式核而言，FM在样本稀疏的情况下是有优势的；而且，FM的训练/预测复杂度是线性的，而二项多项式核SVM需要计算核矩阵，核矩阵复杂度就是N平方。
- 相比MF而言，我们把MF中每一项的rating分改写为 $r_{ui} \sim \beta_u + \gamma_i + x_u^T y_i$，从公式(2)中可以看出，这相当于只有两类特征 u 和i 的FM模型。对于FM而言，我们可以加任意多的特征，比如user的历史购买平均值，item的历史购买平均值等，但是MF只能局限在两类特征。SVD++与MF类似，在特征的扩展性上都不如FM，在此不再赘述。

二、FFM（场感知分解机器）

2.1 FFM的原理及推导

场感知分解机器（Field-aware Factorization Machine ，简称FFM）最初的概念来自Yu-Chin Juan(阮毓钦，毕业于中国台湾大学，现在美国Criteo工作)与其比赛队员，是他们借鉴了来自Michael Jahrer的论文中的field概念提出了FM的升级版模型。通过引入field的概念，FFM把相同性质的特征归于同一个field。以上面的广告分类为
例，“Day=26/11/15”、“Day=1/7/14”、“Day=19/2/15”这三个特征都是代表日期的，可以放到同一个field中。同理，商品的末级品类编码生成了550个特征，这550个特征都是说明商品所属的品类，因此它们也可以放到同一个field中。简单来说，同一个categorical特征经过One-Hot编码生成的数值特征都可以放到同一个field，包括用户性别、职业、品类偏好等。在FFM中，每一维特征 x_i，针对其它特征的每一种field f_j，都会学习一个隐向量 v_{i,f_j}。因此，隐向量不仅与特征相关，也与field相关。也就是说，“Day=26/11/15”这个特征与“Country”特征和“Ad_type"特征进行关联的时候使用不同的隐向量，这与“Country”和“Ad_type”的内在差异相符，也是FFM中“field-aware”的由来。

假设样本的 nn 个特征属于 ff 个field，那么FFM的二次项有 nfnf个隐向量。而在FM模型中，每一维特征的隐向量只有一个。FM可以看作FFM的特例，是把所有特征都归属到一个field时的FFM

模型。根据FFM的field敏感特性，可以导出其模型方程。

$$y(x) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle v_{i,f_j}, v_{j,f_i} \rangle x_i x_j$$

其中，f_j是第j个特征所属的field。如果隐向量的长度为k，那么FFM的二次参数有nfk个，远多于FM模型的nk个。此外，由于隐向量与field相关，FFM二次项并不能够化简，其复杂度为$O(kn^2)$。

下面以一个例子简单说明FFM的特征组合方式。输入记录如下

User	Movie	Genre	Price
YuChin	3Idiots	Comedy, Drama	$9.99

这条记录可以编码成5个特征，其中“Genre=Comedy”和“Genre=Drama”属于同一个field，“Price”是数值型，不用One-Hot编码转换。为了方便说明FFM的样本格式，我们将所有的特征和对应的field映射成整数编号。

Field name	Field index	Feature name	Feature index
User	1	User=YuChin	1
Movie	2	Movie=3Idiots	2
Genre	3	Genre=Comedy	3
		Genre=Drama	4
Price	4	Price	5

那么，FFM的组合特征有10项，如下图所示。

$$\begin{aligned} \langle \mathbf{v}_{1,2}, \mathbf{v}_{2,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{1,3}, \mathbf{v}_{3,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{1,3}, \mathbf{v}_{4,1} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{1,4}, \mathbf{v}_{5,1} \rangle \cdot 1 \cdot 1 \\ + \langle \mathbf{v}_{2,3}, \mathbf{v}_{3,2} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{2,3}, \mathbf{v}_{4,2} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{2,4}, \mathbf{v}_{5,2} \rangle \cdot 1 \cdot 1 \\ + \langle \mathbf{v}_{3,3}, \mathbf{v}_{4,3} \rangle \cdot 1 \cdot 1 + \langle \mathbf{v}_{3,4}, \mathbf{v}_{5,3} \rangle \cdot 1 \cdot 1 \\ + \langle \mathbf{v}_{4,4}, \mathbf{v}_{5,3} \rangle \cdot 1 \cdot 1 \end{aligned}$$

其中，红色表示Field编码，蓝色表示Feature编码，绿色表示样本的组合特征取值（离散化后的结果）。二阶交叉项的系数是通过与Field相关的隐向量的内积得到的。如果单特征有n个，全部做二阶特征组合的话，会有$C_n^2 = \frac{n(n-1)}{2}$个。

2.2 FFM的应用

在DSP的场景中，FFM主要用来预估站内的CTR和CVR，即一个用户对一个商品的潜在点击率和点击后的转化率。

CTR和CVR预估模型都是在线下训练，然后用于线上预测。两个模型采用的特征大同小异，主要有三类：用户相关的特征、商品相关的特征、以及用户-商品匹配特征。用户相关的特征包括年龄、性别、职业、兴趣、品类偏好、浏览/购买品类等基本信息，以及用户近期点击量、购买量、消费额等统计信息。商品相关的特征包括所属品类、销量、价格、评分、历史CTR/CVR等信息。用户-商品匹配特征主要有浏览/购买品类匹配、浏览/购买商家匹配、兴趣偏好匹配等几个维度。

为了使用FFM方法，所有的特征必须转换成“field_id:feat_id:value”格式，field_id代表特征所属field的编号，feat_id是特征编号，value是特征的值。数值型的特征比较容易处理，只需分配单独的field编号，如用户评论得分、商品的历史CTR/CVR等。categorical特征需要经过One-Hot编码成数值型，编码产生的所有特征同属于一个field，而特征的值只能是0或1，如用户的性别、年龄段，商品的品类id等。除此之外，还有第三类特征，如用户浏览/购买品类，有多个品类id且用一个数值衡量用户浏览或购买每个品类商品的数量。这类特征按照categorical特征处理，不同的只是特征的值不是0或1，而是代表用户浏览或购买数量的数值。按前述方法得到field_id之后，再对转换后特征顺序编号，得到feat_id，特征的值也可以按照之前的方法获得。

CTR、CVR预估样本的类别是按不同方式获取的。CTR预估的正样本是站内点击的用户-商品记录，负样本是展现但未点击的记录；CVR预估的正样本是站内支付（发生转化）的用户-商品记录，负样本是点击但未支付的记录。构建出样本数据后，采用FFM训练预估模型，并测试模型的性能。

	#(field)	#(feature)	AUC	Logloss
站内CTR	39	2456	0.77	0.38
站内CVR	67	2441	0.92	0.13

由于模型是按天训练的，每天的性能指标可能会有些波动，但变化幅度不是很大。这个表的结果说明，站内CTR/CVR预估模型是非常有效的。

在训练FFM的过程中，有许多小细节值得特别关注。

第一，样本归一化。FFM默认是进行样本数据的归一化，即 pa.normpa.norm 为真；若此参数设置为假，很容易造成数据inf溢出，进而引起梯度计算的nan错误。因此，样本层面的数据是推荐进行归一化的。

第二，特征归一化。CTR/CVR模型采用了多种类型的源特征，包括数值型和categorical类型等。但是，categorical类编码后的特征取值只有0或1，较大的数值型特征会造成样本归一化后categorical类生成特征的值非常小，没有区分性。例如，一条用户-商品记录，用户为“男”性，商

品的销量是5000个（假设其它特征的值为零），那么归一化后特征“sex=male”（性别为男）的值略小于0.0002，而“volume”（销量）的值近似为1。特征“sex=male”在这个样本中的作用几乎可以忽略不计，这是相当不合理的。因此，将源数值型特征的值归一化到 [0,1][0,1] 是非常必要的。

第三，省略零值特征。从FFM模型的表达式可以看出，零值特征对模型完全没有贡献。包含零值特征的一次项和组合项均为零，对于训练模型参数或者目标值预估是没有作用的。因此，可以省去零值特征，提高FFM模型训练和预测的速度，这也是稀疏样本采用FFM的显著优势。

2.3 FFM实现

Yu-Chin Juan实现了一个C++版的FFM模型，源码可从Github下载[10]。这个版本的FFM省略了常数项和一次项，模型方程如下。

$$\phi(w, x) = \sum_{j1, j2 \in C_2} \langle w_{j_1, f_2}, w_{j_2, f_1} \rangle x_{j_1} x_{j_2}$$

其中，C_2是非零特征的二元组合，j_1是特征，属于field f_1，w_{j_1, f_2}是特征 j_1对field f_2 的隐向量。此FFM模型采用logistic loss作为损失函数，和L2惩罚项，因此只能用于二元分类问题。

$$minw \sum_{i=1}^{L} log(1 + exp{-y_i \phi(w, x_i)}) + \frac{\lambda}{2} \|w\|2$$

其中，$y_i \in -1, 1$是第 i个样本的label，L是训练样本数量，λ 是惩罚项系数。模型采用SGD优化，优化流程如下。

Algorithm 1 SGD(tr, va, pa)

$model = \textbf{init}(tr.n, tr.m, pa)$
$R_{tr} = \mathbf{1}, R_{va} = \mathbf{1}$
if $pa.norm$ **then**
  $R_{tr} = \textbf{norm}(tr), R_{va} = \textbf{norm}(va)$
end if
for $it = 1, \cdots, pa.itr$ **do**
  if $pa.rand$ **then**
    $tr.X = \textbf{shuffle}(tr.X)$
  end if
  for $i = 1, \cdots, tr.l$ **do**
    $\phi = \textbf{calc}\boldsymbol{\Phi}(tr.X[i], R_{tr}[i], model)$
    $e\phi = \exp\{-tr.Y[i] * \phi\}$
    $L_{tr} = L_{tr} + \log\{1 + e\phi\}$
    $g_{\Phi} = -tr.Y[i] * e\phi/(1 + e\phi)$
    $model = \textbf{update}(tr.X[i], R_{tr}[i], model, g_{\Phi})$
  end for
  for $i = 1, \cdots, va.l$ **do**
    $\phi = \textbf{calc}\boldsymbol{\Phi}(va.X[i], R_{va}[i], model)$
    $L_{va} = L_{va} + \log\{1 + \exp\{-va.Y[i] * \phi\}\}$
  end for
end for

参考 Algorithm1, 下面简单解释一下FFM的SGD优化过程。
算法的输入 tr、va、pa 分别是训练样本集、验证样本集和训练参数设置。

1. 根据样本特征数量（tr.ntr.n）、field的个数（tr.mtr.m）和训练参数（papa），生成初始化模型，即随机生成模型的参数；

2. 如果归一化参数 pa.normpa.norm 为真，计算训练和验证样本的归一化系数，样本i的归一化系数为

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

3. 对每一轮迭代，如果随机更新参数 pa.randpa.rand 为真，随机打乱训练样本的顺序；

4. 对每一个训练样本，执行如下操作:

   - 计算每一个样本的FFM项，即公式中的输出 ϕ；
   - 计算每一个样本的训练误差，如算法所示，这里采用的是交叉熵损失函数 $log(1+e\phi)$；
   - 利用单个样本的损失函数计算梯度 $g\Phi$，再根据梯度更新模型参数；

- 5. 对每一个验证样本，计算样本的FFM输出，计算验证误差；
- 6. 重复步骤3~5，直到迭代结束或验证误差达到最小。

在SGD寻优时，代码采用了一些小技巧，对于提升计算效率是非常有效的。

第一，梯度分步计算。采用SGD训练FFM模型时，只采用单个样本的损失函数来计算模型参数的梯度。

$$L = \quad L_{err} + L_{reg} = log(1 + exp\{-y_i\phi(w, x_i)\}) + \frac{\lambda}{2}\|w\|^2$$

$$\frac{\partial L}{\partial w} = \frac{\partial L_{err}}{\partial \phi}\frac{\partial \phi}{\partial w} + \frac{\partial L_{reg}}{\partial w}$$

上面的公式表明，$\frac{\partial L_{err}}{\partial \phi}$与具体的模型参数无关。因此，每次更新模型时，只需计算一次，之后直接调用$\frac{\partial L_{err}}{\partial \phi}$的值即可。对于更新 nfk个模型参数，这种方式能够极大提升运算效率。

第二，自适应学习率。此版本的FFM实现没有采用常用的指数递减的学习率更新策略，而是利用 nfk 个浮点数的临时空间，自适应地更新学习率。学习率是参考AdaGrad算法计算的[11]，按如下方式更新

$$w'_{j_1,j_2} = w_{j_1,f_2} - \frac{\eta}{\sqrt{1+\sum_t (g^t_{w_{j_1,f_2}})^2}} \cdot g_{w_{j_1,f_2}}$$

其中，w_{j_1,f_2}是特征 j_1 对field f_2 隐向量的一个元素，元素下标未标出；$g_{w_{j_1,f_2}}$是损失函数对参数 w_{j_1,f_2}的梯度；$g^t_{w_{j_1,f_2}}$ 是第 t 次迭代的梯度；η是初始学习率。可以看出，随着迭代的进行，每个参

数的历史梯度会慢慢累加，导致每个参数的学习率逐渐减小。另外，每个参数的学习率更新速度是不同的，与其历史梯度有关，根据AdaGrad的特点，对于样本比较稀疏的特征，学习率高于样本比较密集的特征，因此每个参数既可以比较快速达到最优，也不会导致验证误差出现很大的震荡。

第三，OpenMP多核并行计算。OpenMP是用于共享内存并行系统的多处理器程序设计的编译方案，便于移植和多核扩展[12]。FFM的源码采用了OpenMP的API，对参数训练过程SGD进行了多线程扩展，支持多线程编译。因此，OpenMP技术极大地提高了FFM的训练效率和多核CPU的利用率。在训练模型时，输入的训练参数ns_threads指定了线程数量，一般设定为CPU的核心数，便于完全利用CPU资源。

第四，SSE3指令并行编程。SSE3全称为数据流单指令多数据扩展指令集3，是CPU对数据层并行的关键指令，主要用于多媒体和游戏的应用程序中。SSE3指令采用128位的寄存器，同时操作4个单精度浮点数或整数。SSE3指令的功能非常类似于向量运算。例如，a 和 b 采用SSE3指令相加（a 和 b 分别包含4个数据），其功能是 a 中的4个元素与 b 中4个元素对应相加，得到4个相加后的值。采用SSE3指令后，向量运算的速度更加快捷，这对包含大量向量运算的FFM模型是非常有利的。

除了上面的技巧之外，FFM的实现中还有很多调优技巧需要探索。例如，代码是按field和特征的编号申请参数空间的，如果选取了非连续或过大的编号，就会造成大量的内存浪费；在每个样本中加入值为1的新特征，相当于引入了因子化的一次项，避免了缺少一次项带来的模型偏差等。

后记

本文主要介绍了FFM的思路来源和理论原理，并结合源码说明FFM的实际应用和一些小细节。从理论上分析，FFM的参数因子化方式具有一些显著的优势，特别适合处理样本稀疏性问题，且确保了较好的性能；从应用结果来看，站内CTR/CVR预估采用FFM是非常合理的，各项指标都说明了FFM在点击率预估方面的卓越表现。当然，FFM不一定适用于所有场景且具有超越其他模型的性能，合适的应用场景才能成就FFM的“威名”。

参考文献

1. http://blog.csdn.net/lilyth_lilyth/article/details/48032119
2. http://www.cnblogs.com/Matrix_Yao/p/4773221.html
3. http://www.herbrich.me/papers/adclicksfacebook.pdf
4. https://www.kaggle.com/c/criteo-display-ad-challenge
5. https://www.kaggle.com/c/avazu-ctr-prediction
6. https://en.wikipedia.org/wiki/Demand-side_platform
7. http://www.algo.uni-konstanz.de/members/rendle/pdf/Rendle2010FM.pdf

8. http://www.cs.cmu.edu/~wcohen/10-605/2015-guest-lecture/FM.pdf
9. http://www.csie.ntu.edu.tw/~r01922136/slides/ffm.pdf
10. https://github.com/guestwalk/libffm
11. https://en.wikipedia.org/wiki/Stochastic_gradient_descent#AdaGrad
12. http://openmp.org/wp/openmp-specifications/
13. http://blog.csdn.net/gengshenghong/article/details/7008704
14. https://kaggle2.blob.core.windows.net/competitions/kddcup2012/2748/media/Opera.pdf