RS | 深度讨论FM和FFM：不仅是推荐

原创 机智的叉烧 CS的陋室 2019-03-31

点击上方蓝色文字立刻订阅精彩

Can't Feel My Face

The Weeknd - Beauty Behind The Madness

【RS】

本栏目是结合我最近上的七月在线的课、自己自学、以及一些个人的经验推出的专栏，从推荐系统的基础到一些比较好的case，我都会总结发布，当然，按照我往期的风格，更加倾向于去讨论一些网上其实讲得不够的东西，非常推荐大家能多看看并且讨论，欢迎大家给出宝贵意见，觉得不错请点击推文最后的好看，感谢各位的支持。

往期回顾：

- 技术向：推荐学习推荐系统（深度思考，不是广告）
- 【RS】推荐系统的评估
- 【RS】协同过滤-user_based
- 【RS】协同过滤-user_based
- 提问回复0324 | 秋招求职

我看到很多人都已经写过有关FM(Factorization Machine)和FFM(Field-aware Factorization Machine)模型的原理和实现方法，有关论文、实现方法和一些我看的比较好的博客，我都放在这里，有需要的小伙伴可以直接传送过去：

> FM论文：https://www.csie.ntu.edu.tw/~b97053/paper/Rendle2010FM.pdf
> FFM论文：https://www.csie.ntu.edu.tw/~cjlin/papers/ffm.pdf
> CTR预估算法之FM, FFM, DeepFM及实践：
> https://blog.csdn.net/john_xyz/article/details/78933253
> 深入FFM原理与实践：https://tech.meituan.com/2016/03/03/deep-understanding-of-ffm-principles-and-practices.html

东西我都放在这里了，很明显，下面的内容，肯定就不会是上面文章提到过的，而是FM和FFM的一些细节思路，我抽取出来详细和大家讨论。

网上搜FM、FFM全都是，一般都会顶着推荐系统或者相关的帽子，CTR之类的，但是在我最近的学习看来，绝对不是一个只能用在推荐系统问题上的模型，很多实际的问题都能用到，而在思想上，很多文章也没有谈到FM和FFM的一些实现细节，所以我在这里展开讨论一下。

懒人目录：

FM中特征工程的问题

从我的经验和理解看来，FM中其实是非常建议大家对特征进行离散化和one-hot的，这点我再下一章里面谈，这里先谈这两个特征工程方法的细节。

首先，什么是one-hot特征，首先，对于初始数据，必须是离散型才能转为one-hot，举个例子，性别有男、女，则转为one-hot，则变成“性别=男”和“性别=女”两个特征，如果该名用户为男性，则“性别=男”=1，“性别=女”=0，，这种特征就能实现用乘的方式组合。

离散化特征，是针对连续型特征而言的，举个简单的例子，商品售价，而严格的当然有一些离散的由于单个值的样本比较少或者意义不大所以也需要进行离散化，例如考试成绩，对小的间隔只有0.5，这个85和85.5其实差别不会很大，这种建议最好放在一起，进行离散化。离散化最直接的方式就是“分桶”，把整个特征空间平均分为若干份，例如成绩，90到100，80到90等，然后用户的考试成绩再用是否在该区间内来进行one-hot化即可，当然还有更加复杂的，例如地理位置，精度和纬度加起来进行哈希化，得到GeoHash，也是一种离散化的方法，然后通过该用户在不在该位置，就能one-hot化。

而且，我对于离散化特征，其实是非常喜欢做one-hot的，尤其是类别比较多的，例如用户所在省，国内34个，如果只是由0-33来表示，在衡量距离的时候，就会有问题，例如0是北京，10是河北，20是广东，一旦计算，北京和河北并不一定就比北京和广东近（不是地理上，而是综合特点上）而one-hot化后，大家的距离都是2（汉明距离），比较公平。

进行了离散化、one-hot化后，就能够进行FM了。

补充一下，这里只是谈到了有关FM中特别提到的两种特征工程方法，但是特征工程远远不止如此，有关的拓展大家可自行拓展阅读，这还是一个在实际运用中比模型本身还要重要的点，望大家能重视。

FM中的组合特征问题

可能有人会问，为什么FM中要进行离散化和one-hot化，主要是因为最终放入模型的特征要进行相乘计算，相乘其实是一个非常不稳定的计算，主要由于两者相乘的会有过大的变化（即使是归一化后），在FFM的论文中也曾经提到过这么一句话：

> It is more difficult to apply FFMs on numerical data sets.

可见，还是非常建议大家去做离散化的，离散化后，其实one-hot只是随手的事情了。

有关组合特征的问题，似乎由于被看成一种trick，所以没有被很多有关领域的书作为重点来讨论，有些文章写的挺好的，例如下面这篇，会比较全面，大家看完了会有比较深入的了解。

> https://segmentfault.com/a/1190000014799038

书上讨论的不多（连《百面机器学习》这样的书中讨论的都很少），但是在现实问题上，通过特征的组合其实能够令模型效果有新的提升，在工业界，甚至可以体现十分个性化的信息，男性不一定喜欢球鞋，但是如果是某个圈子里的，就很可能非常喜欢了，可见，组合信息可能会产生十分特别的效果。

回头看看FM是怎么解决的，这里就要请出FM最核心的公式（公式1），论文截图一出，原汁原味。

A. Factorization Machine Model

1) Model Equation: The model equation for a factorization machine of degree $d = 2$ is defined as:

$$\hat{y}(\mathbf{x}) := w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v}_i, \mathbf{v}_j \rangle x_i x_j \quad (1)$$

where the model parameters that have to be estimated are:

$$w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^n, \quad \mathbf{V} \in \mathbb{R}^{n \times k} \quad (2)$$

And $\langle \cdot, \cdot \rangle$ is the dot product of two vectors of size k:

$$\langle \mathbf{v}_i, \mathbf{v}_j \rangle := \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f} \quad (3)$$

A row $\mathbf{v}_i$ within $\mathbf{V}$ describes the i-th variable with k factors. $k \in \mathbb{N}_0^+$ is a hyperparameter that defines the dimensionality of the factorization.

第一项和第二项大家都很熟悉，分别是常数项和一次项，而第三项，就是一个组合项，任取两个特征相乘，在给予特定的组合权重（其实是两个特征向量的点乘），这里的特征，应该是离散化、one-hot化的特征。

首先看两个特征，xi和xj，one-hot化后，两者相乘的优势就会变得非常明显了，其实就表达了一个“且”的概念，当且仅当两个特征同时不为0，这项才有值（这里绝对注意哈，在实际的特征中，为0不是指没有意义，但是在one-hot化后，就是了！注意区分和立即），于是就能真正体现FM的真实含义。

再来看v，vi和vj其实都是向量，在FM论文的公式(3)中已经定义了，f是转化特征的维数，可以表示更加丰富的含义（大家可以想象一下矩阵分解），这种高纬度化能够令一个特征的描述更为丰富，从单一值转化为更为丰富的含义，而此时，具体这个特征是什么就显得不是很重要，他用一个抽象的向量表示，且高纬度的表达也更为精准，这也是NLP领域里面提到的embedding的一大重要意义。

所以，可以看到FM用了一种非常巧妙的方式去进行了特征的组合，且这种方式的效率很高，同时复杂度也很低（FM论文中详细证明了是线性复杂度，与特征个数和向量特征维数f有关）。

同时强调，特征组合的方式非常丰富，《百面机器学习》中还提到了基于决策树方法的组合，这些都建议去看看，有的时候，组合特征能够一定程度的提升性能，比换模型、在模型加attention之类的要高效很

多。

FFM的Field-aware

FFM的核心创新点就在于引入了Field-aware的概念，在于把几个相同性质的特征归结为一个field，例如“Day=1/3/19”、“Day=1/1/18”、“Day=21/3/15”（日/月/年）都是日期特征，上面提到的广东省北京市河北省都是地点省级别的特征，应该放在一个field里面，每一个特征（onehot）xi，对对应的fieldf学习一个隐向量vif，此时隐向量就是连接特征和field的桥梁，即“sex=male”这个特征，就和Date这个field进行了连接，于是模型就更新为这样：

$$\phi_{\text{FFM}}(\boldsymbol{w}, \boldsymbol{x}) = \sum_{j_1=1}^{n} \sum_{j_2=j_1+1}^{n} (\boldsymbol{w}_{j_1,f_2} \cdot \boldsymbol{w}_{j_2,f_1}) x_{j_1} x_{j_2}, \qquad (4)$$

where f_1 and f_2 are respectively the fields of j_1 and j_2. If

引入了field的概念，核心目标在于，很多时候没有必要衡量任意两个小特征的关系，而只需要衡量小特征和每个field之间的关系，这样能一定程度降低稀疏性，提升隐向量的实际含义和泛化能力，隐向量的个数确实大大缩小，但是field的个数却也有关，因此不好说谁的复杂度高了，和实际问题有关，可以说的是，其实在FFM作者的实验中，FFM的提升相比FM并不多。

Model and implementation	parameters	training time (seconds)	public set logloss	public set rank	private set logloss	private set rank
LM-SG	$\eta = 0.2, \lambda = 0, t = 13$	527	0.46262	93	0.46224	91
LM-LIBLINEAR-CD	$s = 7, c = 2$	1,417	0.46239	91	0.46201	89
LM-LIBLINEAR-Newton	$s = 0, c = 2$	7,164	0.46602	225	0.46581	222
Poly2-SG	$\eta = 0.2, \lambda = 0, B = 10^7, t = 10$	12,064	0.44973	14	0.44956	14
Poly2-LIBLINEAR-Hash-CD	$s = 7, c = 2$	24,771	0.44893	13	0.44873	13
FM	$\eta = 0.05, \lambda = 2 \times 10^{-5}, k = 40, t = 8$	2,022	0.44930	14	0.44922	14
FM	$\eta = 0.05, \lambda = 2 \times 10^{-5}, k = 100, t = 9$	4,020	0.44867	11	0.44847	11
LIBFM	$\lambda = 40, k = 40, t = 20$	23,700	0.45012	14	0.45000	15
LIBFM	$\lambda = 40, k = 40, t = 50$	131,000	0.44904	14	0.44887	14
LIBFM	$\lambda = 40, k = 100, t = 20$	54,320	0.44853	11	0.44834	11
LIBFM	$\lambda = 40, k = 100, t = 50$	398,800	0.44794	9	0.44778	8
FFM	$\eta = 0.2, \lambda = 2 \times 10^{-5}, k = 4, t = 9$	6,587	0.44612	3	0.44603	3

(a) Criteo

Model and implementation	parameters	training time (seconds)	public set logloss	public set rank	private set logloss	private set rank
LM-SG	$\eta = 0.2, \lambda = 0, t = 10$	164	0.39018	57	0.38833	64
LM-LIBLINEAR-CD	$s = 7, c = 1$	417	0.39131	115	0.38944	119
LM-LIBLINEAR-Newton	$s = 0, c = 1$	650	0.39269	182	0.39079	183
Poly2-SG	$\eta = 0.2, \lambda = 0, B = 10^7, t = 10$	911	0.38554	10	0.38347	10
Poly2-LIBLINEAR-Hash-CD	$s = 7, c = 1$	1,756	0.38516	10	0.38303	9
Poly2-LIBLINEAR-Hash-Newton	$s = 0, c = 1$	27,292	0.38598	11	0.38393	11
FM	$\eta = 0.05, \lambda = 2 \times 10^{-5}, k = 40, t = 8$	574	0.38621	11	0.38407	11
FM	$\eta = 0.05, \lambda = 2 \times 10^{-5}, k = 100, t = 9$	1,277	0.38740	17	0.38531	15
LIBFM	$\lambda = 40, k = 40, t = 20$	18,712	0.39137	122	0.38963	127
LIBFM	$\lambda = 40, k = 40, t = 50$	41,720	0.39786	935	0.39635	943
LIBFM	$\lambda = 40, k = 100, t = 20$	39,719	0.39644	747	0.39470	755
LIBFM	$\lambda = 40, k = 100, t = 50$	91,210	0.40740	1,129	0.40585	1,126
FFM	$\eta = 0.2, \lambda = 2 \times 10^{-5}, k = 4, t = 4$	340	0.38411	6	0.38223	6

(b) Avazu

FM和FFM的应用场景

文章开头就讲过，不要把FM和FFM局限在推荐系统尤其是CTR的问题，很多时候能做好多别的问题，都能够借鉴，其他领域的小伙伴也可以把这整个思路当做是一个trick，在合适的问题中使用。在FM论文中，作者就已经抽象化的提到了FM的应用场景，分别是**回归问题、二分类问题和pairwise排序问题**，而且，随着技术的演进，甚至被放入了深度学习中的某一层中，例如DeepFM中，甚至有结合wide&deep进行组合的新模式，可谓是十分丰富，**他不再是一个独立的模型**，例如也有人用支持向量机来替代softmax或者sigmoid作为最后一层的输出层计算，都是有的，希望大家能够从一些比较局限的思维里面走出来。

小结

开始想写理论，我自己代码也有，但是感觉写进来并无必要，写写删删，最终到了这个状态，一方面网上的大量文章其实都有，另一方面是感觉对模型深层次的理解远比理论本身和代码有用，这两个是带领你把事情完成的基础，但是不是你进行理解、改进和深化的动力，例如DeepFM的提出就是依赖于对模型的理解。

FM的作者在论文中提到FM的三大优点分别是**可处理高度稀疏的数据**、**线性复杂度**、**实数域可用**（虽然我前面还是建议做离散化和one-hot），而在我看来，其实FM和FFM之所以厉害，有下面几个原因，供大家参考：

- 在线性模型的基础上，提出了通过组合特征来提升性能的方案
- 通过使用离散化特征来解决线性模型的缺陷，实现灵活的非线性化
- 适用于特征多样化（如用户特征）、且具有一定稀疏性的问题

参考文献

[1] FM论文：https://www.csie.ntu.edu.tw/~b97053/paper/Rendle2010FM.pdf

[2] FFM论文：https://www.csie.ntu.edu.tw/~cjlin/papers/ffm.pdf

[3] CTR预估算法之FM, FFM, DeepFM及实践：https://blog.csdn.net/john_xyz/article/details/78933253[4] 深入FFM原理与实践：https://tech.meituan.com/2016/03/03/deep-understanding-of-ffm-principles-and-practices.htm