FM 与 FFM 算法

FM 算法全称叫因子分解机(Factorization Machines)，而 FFM(Field-aware Factorization Machines)算法是 FM 算法的特例，这两个算法通常解决稀疏数据下的特征组合问题。

1. FM 算法

FM 算法的模型是多项式模型，模型的表达式如下：

$$y(\boldsymbol{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{ij} x_i x_j$$

上式中，$\boldsymbol{x}$ 表示样本向量，n 表示特征个数，x_i 表示样本的第 i 个特征值，w_0、w_i 和 w_{ij} 是模型参数。由于在数据稀疏普遍存在的应用场景中，二项式系数 w_{ij} 是很难训练的，因此将二项式系数 w_{ij} 拆分为两个特征隐向量的点积，故 FM 算法的模型公式为：

$$y(\boldsymbol{x}) = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} < \mathbf{v}_i, \mathbf{v}_j > x_i x_j$$

上式中，$\mathbf{v}_i$ 和 $\mathbf{v}_j$ 分别是 x_i 和 x_j 的隐向量。其中，

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

上式中，k 表示隐向量的维度，$v_{i,s}$ 和 $v_{j,s}$ 分别表示样本第 i 个和第 j 个特征隐向量的第 s 个值。由于上式是针对单个样本的二次交叉项，但是在 tensorflow 中通常针对的是多个样本的二次交叉项，因此在 tensorflow 中二次交叉项公式如下：

$$\boldsymbol{I}_{m\times 1} = \frac{1}{2} sum\_row\left(square\left(\boldsymbol{X}_{m\times n} \boldsymbol{V}_{n\times k} \right) - square\left(\boldsymbol{X}_{m\times n} \right) square\left(\boldsymbol{V}_{n\times k} \right) \right)$$

上式中，m 表示样本个数，函数 sum_row 表示对矩阵行求和，函数 $square$ 表示对矩阵每个元素求平方，$\boldsymbol{I}_{m\times 1}$ 表示 m 个样本的二次交叉项值矩阵，$\boldsymbol{X}_{m\times n}$ 表示特征个数为 n 的 m 个样本矩阵，$\boldsymbol{V}_{n\times k}$ 表示 n 个特征每个隐向量长度为 k 的矩阵，用 tensorflow 代码表示如下：

```
interaction = 0.5 * tf.reduce_sum(
tf.subtract(
tf.pow(
tf.matmul(sample, embedding), 2),
tf.matmul(tf.pow(sample, 2), tf.pow(embedding, 2))
```

), axis=1, keepdims=True)

其中，interaction 表示$\boldsymbol{I}_{m\times 1}$，sample 表示$\boldsymbol{X}_{m\times n}$，embedding 表示$\boldsymbol{V}_{n\times k}$，然而利用梯度下降法训练单个样本时，模型各个参数的梯度如下：

$$\frac{\partial}{\partial\theta}y(\boldsymbol{x})=\begin{cases}1 & \theta=w_0\\ x_i & \theta=w_i\\ x_i\sum_{j=1}^{n}v_{j,s}x_j-v_{i,s}x_i^2 & \theta=v_{i,s}\end{cases}$$

在随机梯度下降法下，FM 算法的损失函数一般分为两种：

(a).回归问题：最小均方误差(the least square error)

$$loss(\overline{y}^{(i)},y^{(i)})=\frac{1}{2}(\overline{y}^{(i)}-y^{(i)})^2$$

上式中，$\overline{y}^{(i)}$与 $y^{(i)}$分别表示第 i 个样本的模型输出和标签值，因此上式的损失函数对 θ(包含：w_0、w_i 和 $v_{i,s}$)的偏导为：

$$\frac{\partial}{\partial\theta}loss(\overline{y}^{(i)},y^{(i)})=(\overline{y}^{(i)}-y^{(i)})\frac{\partial}{\partial\theta}\overline{y}^{(i)}$$

其中$\frac{\partial}{\partial\theta}\overline{y}^{(i)}=\frac{\partial}{\partial\theta}y(\boldsymbol{x})$；

(b).二分类问题

对于二分类问题，采用 logit loss 函数作为损失函数，即：

$$loss(\overline{y}^{(i)},y^{(i)})=-\ln\sigma(\overline{y}^{(i)}y^{(i)})$$

其中，$\sigma(x)=\frac{1}{1+e^{-x}}$，为了防止 $\sigma(x)$函数的输入是零，将期望输出 $y^{(i)}$正例与负例置为 1 与-1，因此上式的损失函数对 θ(包含：w_0、w_i 和 $v_{i,s}$)的偏导为：

$$\begin{aligned}\frac{\partial}{\partial\theta}loss(\overline{y}^{(i)},y^{(i)})&=-\frac{1}{\sigma(\overline{y}^{(i)}y^{(i)})}\sigma(\overline{y}^{(i)}y^{(i)})\left[1-\sigma(\overline{y}^{(i)}y^{(i)})\right]y^{(i)}\frac{\partial}{\partial\theta}\overline{y}^{(i)}\\&=y^{(i)}\left[\sigma(\overline{y}^{(i)}y^{(i)})-1\right]\frac{\partial}{\partial\theta}\overline{y}^{(i)}\end{aligned}$$

其中$\frac{\partial}{\partial\theta}\overline{y}^{(i)}=\frac{\partial}{\partial\theta}y(\boldsymbol{x})$；

最后根据随机梯度下降法的迭代公式更新每个参数的值。

2. FFM 算法

FFM 算法是 FM 算法的改进。在 FM 算法中，每个特征对应一个隐向量(即：该特征与任何特征进行特征组合都用相同的隐向量)。但是在 FFM 算法中，每个特征有 f-1 个隐向量(其中 field 的个数为 f，同一个 field 中的特征不会进行组合)。因此 FFM 算法的模型公式为：

$$\begin{aligned}y(\boldsymbol{x})&=w_0+\sum_{i=1}^{n}w_ix_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}<\mathbf{v}_{i,f_j},\mathbf{v}_{j,f_i}>x_ix_j\\&=w_0+\sum_{i=1}^{n}w_ix_i+\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(\mathbf{v}_{i,f_j}\cdot\mathbf{v}_{j,f_i})x_ix_j\end{aligned}$$

其中，f_j 是第 j 个特征所属的 field，f_i 是第 i 个特征所属的 field。

由于 FFM 算法中的每个特征的隐向量与 field 有关，所以 FFM 算法的二次项并不能够化简。又因为 FFM 算法与 FM 算法在回归问题与二分类问题中的损失函数一样，所以只需要计算 $\mathbf{v}_{i,f_j}$ 和 $\mathbf{v}_{j,f_i}$ 的梯度。同时需要注意，在计算 $\mathbf{v}_{i,f_j}$ 和 $\mathbf{v}_{j,f_i}$ 的梯度时，x_i 和 x_j 都不为零，否则无法计算它们的梯度，故：

$$\boldsymbol{g}_{i,f_j} = \kappa \cdot \mathbf{v}_{j,f_i} x_i x_j$$
$$\boldsymbol{g}_{j,f_i} = \kappa \cdot \mathbf{v}_{i,f_j} x_i x_j$$

上式中，$\boldsymbol{g}_{i,f_j}$ 和 $\boldsymbol{g}_{j,f_i}$ 分别为 $\mathbf{v}_{i,f_j}$ 和 $\mathbf{v}_{j,f_i}$ 的梯度，在回归问题中 $\kappa = \overline{y}^{(i)} - y^{(i)}$，在二分类问题中 $\kappa = y^{(i)}\left[\sigma(\overline{y}^{(i)} y^{(i)}) - 1\right]$，其中 $\overline{y}^{(i)} = y(\boldsymbol{x})$，$y^{(i)}$是样本的期望输出(1 或者-1)。因此，$\mathbf{v}_{i,f_j}$ 和 $\mathbf{v}_{j,f_i}$ 隐向量第 s 个维度的梯度为：

$$\left(\mathbf{v}_{i,f_j}\right)_s \leftarrow \left(\mathbf{v}_{i,f_j}\right)_s - \frac{\eta}{\sqrt{\left(\boldsymbol{G}_{i,f_j}\right)_s}}\left(\boldsymbol{g}_{i,f_j}\right)_s$$
$$\left(\mathbf{v}_{j,f_i}\right)_s \leftarrow \left(\mathbf{v}_{j,f_i}\right)_s - \frac{\eta}{\sqrt{\left(\boldsymbol{G}_{j,f_i}\right)_s}}\left(\boldsymbol{g}_{j,f_i}\right)_s$$

上式中，η 是学习率，$\boldsymbol{G}_{i,f_j}$ 和 $\boldsymbol{G}_{j,f_i}$ 分别为：

$$\left(\boldsymbol{G}_{i,f_j}\right)_s \leftarrow \left(\boldsymbol{G}_{i,f_j}\right)_s + \left(\boldsymbol{g}_{i,f_j}\right)_s^2$$
$$\left(\boldsymbol{G}_{j,f_i}\right)_s \leftarrow \left(\boldsymbol{G}_{j,f_i}\right)_s + \left(\boldsymbol{g}_{j,f_i}\right)_s^2$$

在初始化时，$\boldsymbol{G}_{i,f_j}$ 和 $\boldsymbol{G}_{j,f_i}$ 可以为 1，这样可以防止分母为 0。