CTD是无监督算法，在计算数据源权重，以便进一步更新迭代的真值时，需要调用embedding模型，得到数据源中不同value的距离。需要注意的是这里embedding训练用的数据集是动态变化的，CTD算法最后会返回最新版的model。

CTD算法的返回值，我们设定为CTD算法迭代计算的数据源权重。同时，我们也提供了接口，直接返回计算的真值calcTruth，最终保存的模型，便于在遗传算法中适应度函数的计算。

CTD算法是在GA（遗传算法）中被调用的。对于GA，初始子代设置为按照超参结构，随机生成的，结构显示如下：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Length | 属性正态分布均值 | 属性正态分布标准差 | 值正态分布平均值 | 值正态分布标准差 | 样本正态分布均值 | 样本正态分布标准差 |

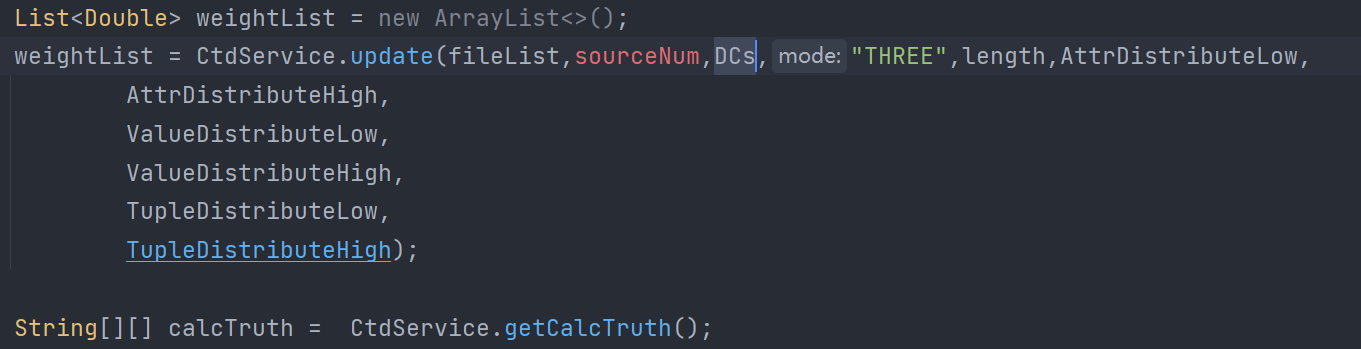
其中，length长度设定为6位，其余长度为3位，都用二进制表示。Dropout边的操作，直接写死到embedding训练中了，不必找参数了。初始的超参组合是随机生成的，在经过交叉，变异等操作后，我们需要计算子代适应度。此时调用changeX，负责解析我们的二进制基因编码（代码中用染色体类Chromone表示），负责按照上图顺序解析基因，然后输出一些提示信息，便于观察超参解析情况是否出错。之后调用changeY函数，利用基因，解析计算适应度函数。适应度函数我们采用了wilcoxon test方法，需要针对当前子代，计算邻域质优度，获取CTD返回的数据源权重，以及利用CTD中训练好的EMBDI模型计算的数据源两两之间的相似度。其中，邻域质优度需要计算CTD迭代计算出的真值和golden standard的rmse排名。这里没有进行抽样操作，直接全都比较了。这里的排名，是建立在k-δ-之上的，也就是说我们需要先计算k次初始超参组合，获取他们的参数，迭代真值等，然后在后续过程中，维护一个长度为K的列表。每当有满足deta上届要求，并且自身rmse计算结果能够更新topK的，就更新这个列表。最后可以通过当前rmse数值在列表中的排名，计算得到Qi质优度，进而计算出适应度函数B（r）。这里，我认为rank应该按照降序排序，因为rmse较小的，降序后index大，带入B后评估分数大，反馈给遗传算法的适应度就高，可以被留下。



为了实现上述功能，我们将CTD和遗传算法，通过CTD\_Algorithm类对象的方法调用之外，还附加了读取golden standard，计算rmse，以及利用保存的模型直接计算distance的方法distanceUseSavedModel。

需要特别注意的是，在此过程中，迭代的真值可能出现为空，distance可能发现空指针异常，找不到词。我们直接捕获异常，返回distance/rmse为0了。

对于CTD的参数输入，除了调整的超参外，还有原始数据集，否定约束DC（对于stock，里面有今日最低和最高股价，所以我利用这两个的最高>最低，设置了否定约束输入给了CTD），采用的数据源数目55，采用的训练模式（三分图，传入的mode=”THREE”）



五分图的对应调整分布的版本还没修改，如果有必要的话可以修改（呜）