两种缺失值补全算法实现与测试

脚本：FillNaN.py

时间：2017.11.16

by小虎

FillNaN算法特点

- 一、输入的原始数据，即可返回补全结果，不论原始数据有没有经过处理；
- 二、可以采用两种方案来对缺失值补全，分别为**‘dt’**和**‘ht’**。两种不同的算法思路：
- ‘dt’为：首先采用‘both’方法建立一颗树，然后通过叶子节点的样本分组（同一个叶子节点下的样本分为一组），然后组内进行缺失值补全。如果一组内的一个字段的值全部为NaN，则向上寻找对应的第一个父节点，直到该父节点下有不为NaN的样本；
- ‘ht’为：首先，用户可以自己选择是否选择特征，选择特征的方案为tree的feature importance.默认采用所有的特征，对类别型特征做one-hot编码，消除除类别型缺失，对连续性特征取均值填充，然后计算距离D，D的计算公式为：

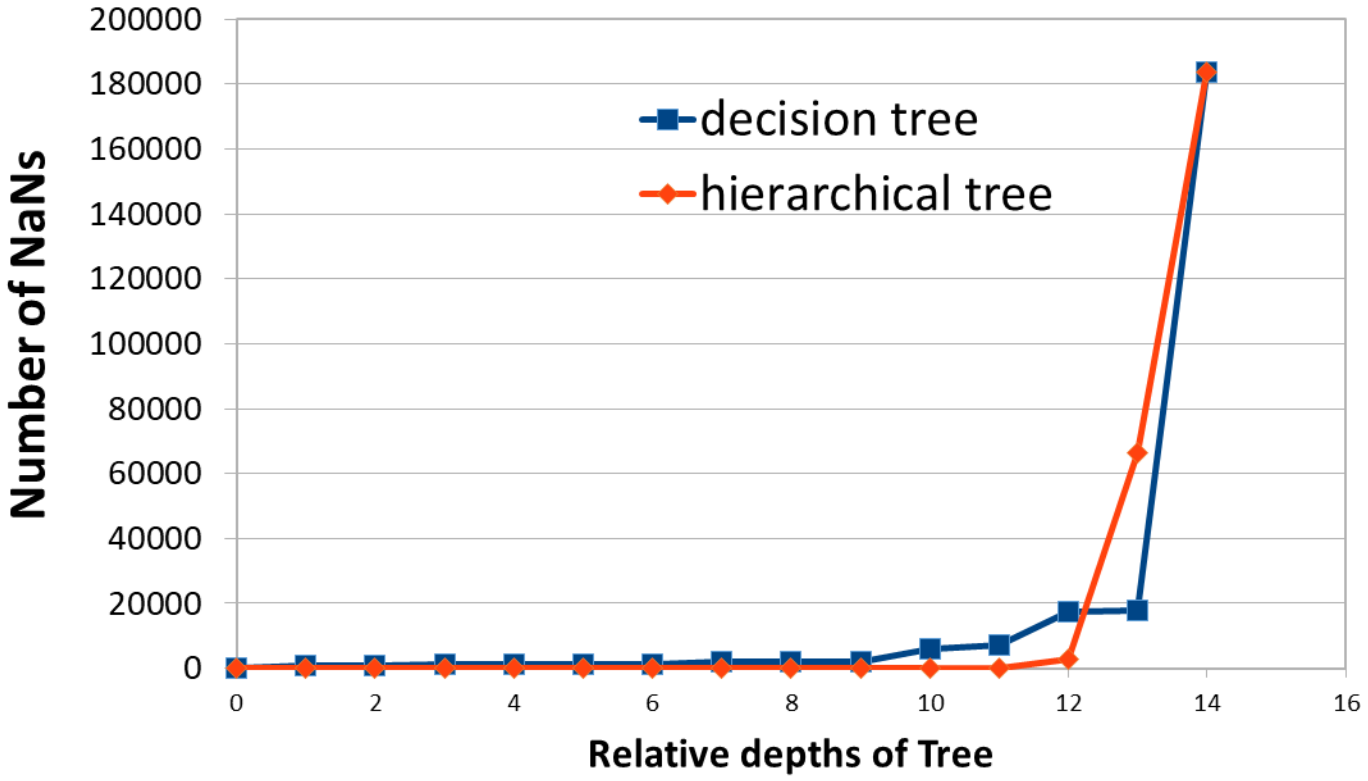
‘ht’方法的距离D公式

- D=D1*W1 + D2*W2
- D1为类别型特征， default使用Tanimoto距离；
- D2为连续型特征， default使用Correlation距离；
- W1为类别型特征权重， default为类别型特征占所有特征的比例；
- W2为连续型特征权重， default为连续型特征占所有特征的比例；

分组后数据补全方案：
类别型：组内出现次数最多值
连续型：组内均值

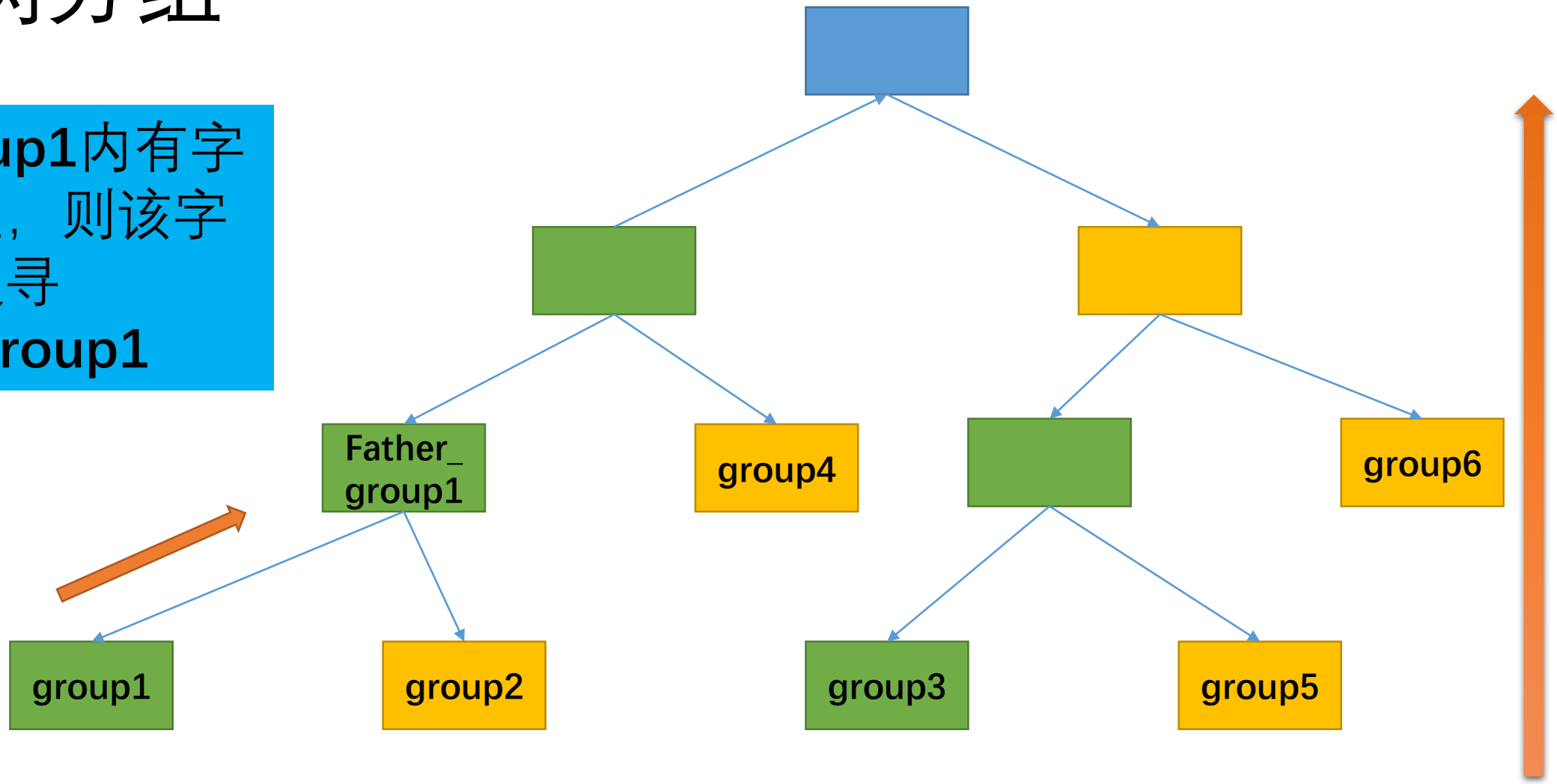
数据填充过程

- 对树自下而上遍历，直到所有的数据都没有空为止

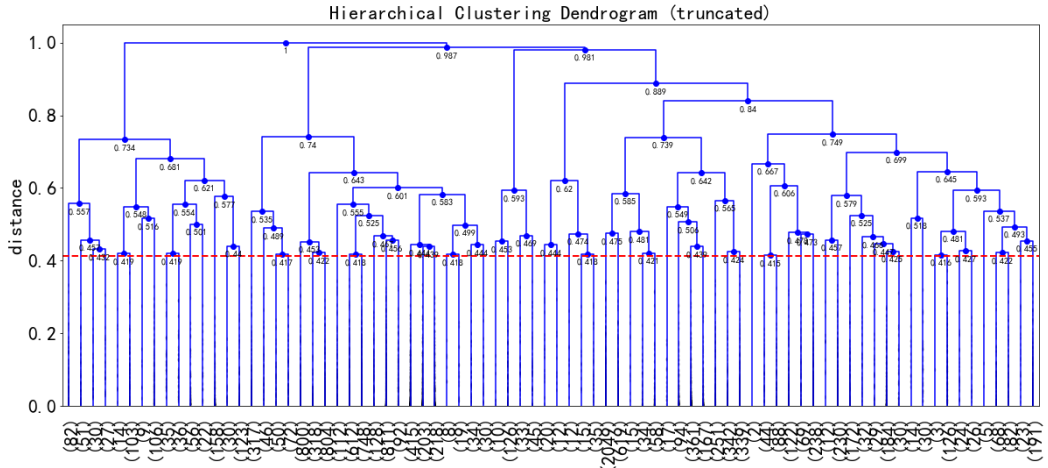


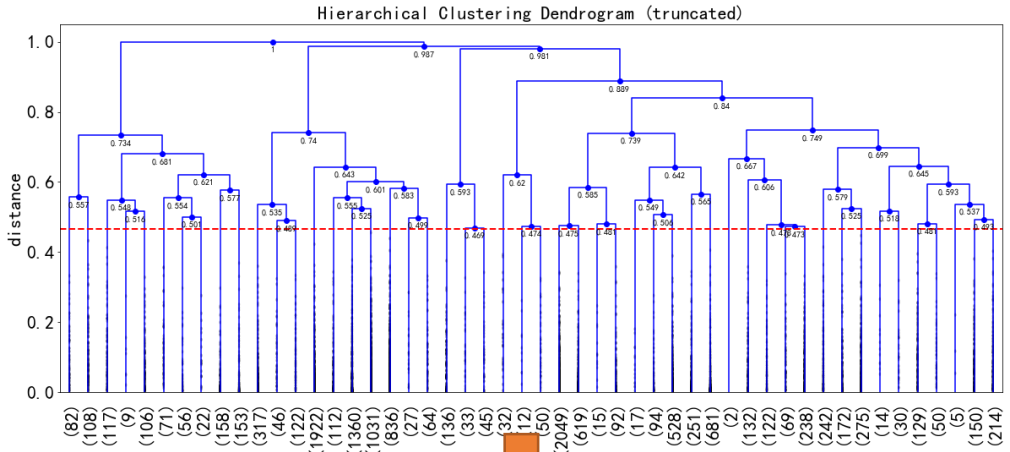
决策树分组

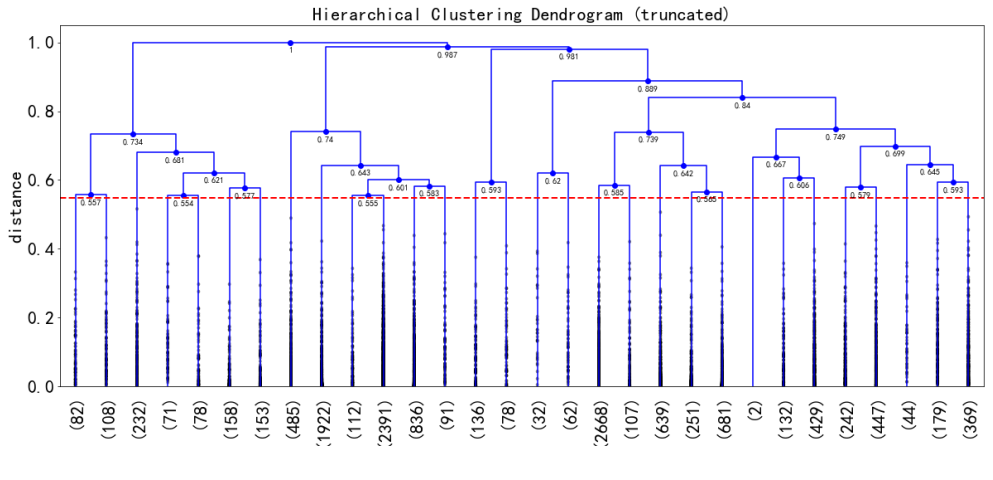
如果**group1**内有字段全为空，则该字段向上搜寻**Father_group1**

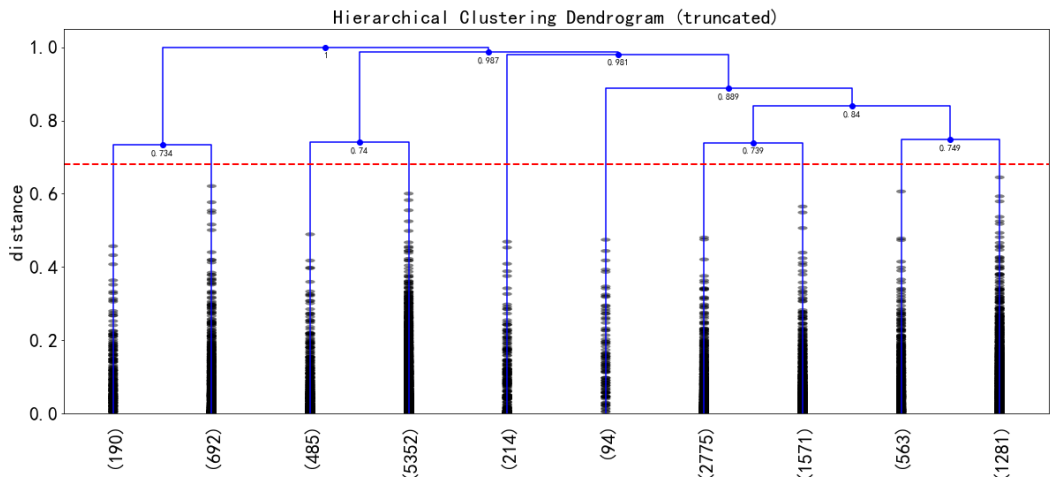


层次树分组


Hierarchical Clustering Dendrogram (truncated)
distance


Hierarchical Clustering Dendrogram (truncated)
distance


Hierarchical Clustering Dendrogram (truncated)
distance


Hierarchical Clustering Dendrogram (truncated)
distance

初步测试方案

I. 将王老师给我的车险数据集分为训练（0.75）和测试（0.25）集合

II. 采用开发的FWD方法对数据集合进行缺失补全，分别采用了**决策树**方法（decision tree，DT）和**层次树**方法（hierarchical tree，HT）

III. 补全结果和原始的数据见附件（dt_df_train.csv，ht_df_train.csv等）

IV. 对以上数据集处理后，采用决策树（DT），剪枝后决策树（DT_prune）,随机森林（RF）和梯度提升树（GBDT）模型测试

初步测试结果

	DT	DT_Prune	RF	GBDT
Original_Data	0.830	0.939	0.929	0.973
DT_Fill_NaN	0.832	0.940	0.918	0.971
HT_Fill_NaN	0.824	0.939	0.926	0.973

结果与讨论

- 1.使用decision tree方法补全可能会存在的问题：有一些变量没有用上，特别是剪枝后，有一些变量剔除了。如车险数据集原始的特征80几个，采用‘both’方法的树用到的特征只有十几个。最后在分组的时候，实际上是基于这十几个特征来分组的，导致其他的特征被忽略，在补全的时候就没有考虑它们

- 2.从结果上看， decision tree方法补全在单颗树上表现较好，但是放在RF和GBDT里面就不行了。说明其他特征也很关键

- 3.使用hierarchical tree补全可以考虑所有的特征，但是特征越多，每个样本的独立性越大，分组越多，导致时间比较长。选择哪些特征就很关键