《时序数据处理算子开发——数据画像》项目文档

——Minmax, Mode, Mvavg, Pacf, Percentile, Period 函数实现及测试

张梦遥　致理书院　2020012610　zhang-my20@mails.tsinghua.edu.cn

一、归一化 Minmax 实现及测试

参数：

compute ：若设置为"batch"，则将数据全部读入后转换；若设置为"stream"，则需用户提供最大值及最小值进行流式计算转换。默认为"batch"。

min ：使用流式计算时的最小值。

max ：使用流式计算时的最大值。

不忽略输入序列中空缺，利用下式逐个计算得到输出序列，其中 MIN 是输入序列中的最小值，MAX 是输入序列中的最大值，min、max 对应输入参数，在‘batch’模式下固定为 0、1。输出序列时间戳与输入序列保持一致，空缺对应输出为 NaN。

$$output = \frac{input - MIN}{MAX - MIN} \times (\max - \min) + \min$$

单元测试：

测试用例见“root_test_zmy”、“root_test_d1”（作业文档提供的使用示例，第 i 个函数对应第 si 组数据，下同）。

Minmax 对应“root_test_zmy”中数据组“s1”（INT，正数，存在空缺），“s2”（FLOAT，正负数，无空缺）。

1. "s1"	无参数（batch）
2. "s1", "compute": "batch", "min": 0, "max": 10	batch 下 min、max 输入不影响结果
3. "s2", "compute": "stream", "min": 1, "max": 10	stream
4. "s2", "compute": "stream", "min": 1, "max": -1	stream 下 min、max 异常输入，对调计算（若 min、max 相等则得到常值输出序列），但不修改输出序列对应 column 名
5. 使用示例	

利用“pytest --count=m MinmaxUT.py”进行重复测试，耗时如下：

m	Time/s
100000	15.66

1000000	19.78

二、众数 Mode 实现及测试

忽略输入序列中空缺，利用 pandas 中 DataFrame.mode()函数进行计算，输出序列时间戳为 0。

单元测试：

Mode 对应“root_test_zmy”中数据组“s1”,“s2”,“s3”。

1. "s1"	存在空缺
2. "s2",	存在多个众数
3. "s3"	输入序列类型混杂
5. 使用示例	

利用“pytest --count=m ModeUT.py”进行重复测试，耗时如下：

m	Time/s
100000	2.99
1000000	3.69

三、移动平均 Mvavg 实现及测试

参数：

window ：移动窗口的长度。默认值为 10

利用 DataFrame.rolling(window).mean()，不忽略输入序列中空缺，移动窗口中存在空缺时不进行计算，输出序列中无对应时间戳。

单元测试：

Mvavg 对应“root_test_zmy”中数据组“s1”,“s2”。

1. "s1"	无参数（window=10），输入序列中存在空缺
2. "s1", "window": 4	输入参数且为小于输入序列数值个数的正整数
3. "s2", "window": 30	输入参数且为大于输入序列数值个数的正整数，输出序列为空 Dataframe
4. "s2","window": -1	输入参数且为负整数，按默认值 window=10 计算，修改输出序列对应 column 名
5. 使用示例	

利用“pytest --count=m MvavgUT.py”进行重复测试，耗时如下：

m	Time/s
100000	2.70
1000000	3.02

四、偏自相关系数 Pacf 实现及测试

参数：

lag ：最大滞后阶数。默认值为 min(10log$_{10}$ n, n-1) ，n 表示数据点个数。

不输入序列中空缺，空缺存在时 Yule-Walker 方程一般无解，输出序列共 lag+1 个数据，时间戳与输入序列前 lag+1 个时间戳对应。

将等式 $x_{i+1}=\sum_{j=1}^{k}(\phi_j x_{i-j+1})+\xi_{i+1}$ 两边同乘x_{i-k+1}，得到：

$$x_{i-k+1}.x_{i+1}=\sum_{j=1}^{k}(\phi_j x_{i-j+1}.x_{i-k+1})+x_{i-k+1}.\xi_{i+1}$$

求两边数学期望：

$$\langle x_{i-k+1}.x_{i+1}\rangle=\sum_{j=1}^{k}(\phi_j\langle x_{i-j+1}.x_{i-k+1}\rangle)+\langle x_{i-k+1}.\xi_{i+1}\rangle$$

噪声值的数学期望为 0，因此可化简为：

$$\langle x_{i-k+1}.x_{i+1}\rangle=\sum_{j=1}^{k}(\phi_j\langle x_{i-j+1}.x_{i-k+1}\rangle)$$

等式两边同除 N-k，后同除以c_0：

$$c_k=\sum_{j=1}^{k}\phi_j c_{j-k}$$

$$r_k=\sum_{j=1}^{k}\phi_j r_{j-k}$$

即：

$$\begin{pmatrix} r_1\\ r_2\\ .\\ r_{k-1}\\ r_k\end{pmatrix}=\begin{pmatrix} r_0 & r_1 & r_2 & . & r_{k-2} & r_{k-1}\\ r_1 & r_0 & r_1 & . & r_{k-3} & r_{k-2}\\ . & . & . & . & . & .\\ r_{k-2} & r_{k-3} & r_{k-4} & . & r_0 & r_1\\ r_{k-1} & r_{k-2} & r_{k-3} & . & r_1 & r_0\end{pmatrix}\times\begin{pmatrix}\phi_1\\ \phi_2\\ .\\ \phi_{k-1}\\ \phi_k\end{pmatrix}$$

其中$r_0 = 1$，我们令：

$$r = \begin{pmatrix} r_1 \\ r_2 \\ . \\ r_{k-1} \\ r_k \end{pmatrix}, R = \begin{pmatrix} 1 & r_1 & r_2 & . & r_{k-2} & r_{k-1} \\ r_1 & 1 & r_1 & . & r_{k-3} & r_{k-2} \\ . & . & . & . & . & . \\ r_{k-2} & r_{k-3} & r_{k-4} & . & 1 & r_1 \\ r_{k-1} & r_{k-2} & r_{k-3} & . & r_1 & 1 \end{pmatrix}, \Phi = \begin{pmatrix} \phi_1 \\ \phi_2 \\ . \\ \phi_{k-1} \\ \phi_k \end{pmatrix}$$

可得到 r = RΦ，由于 R 为满秩 Toeplitz 矩阵（若输入序列中不存在空缺），其逆存在，利用 np.linalg.solve()解Φ即得所求。

单元测试：

Pacf 对应“root_test_zmy”中数据组“s1”,“s2”、“s6”(取 s2 中前四个数据，$10\log_{10} 4 > 4$-1)。

1. "s1“, “lag": 2	存在空缺，Yule-Walker 方程无解
2. "s2"	无输入（lag=35，取 $10\log_{10} n$作为默认值）
3. "s2", “lag": 8	输入大小小于数据个数的参数；若参数大小大于数据个数，超出数据个数的偏自相关系数解为 NaN
4. "s2", “lag": -1	输入负数，取默认值（lag=35）并修改输出序列对应 column 名
5. "s6"	无输入（lag=3，取 n-1 作为默认值）
6. 使用示例	

利用“pytest --count=m PacfUT.py”进行重复测试，耗时如下：

m	Time/s
100000	3.27
1000000	3.89

五、分位数 Percentile 实现及测试

参数：

rank ：所求分位数在所有数据中的排名百分比，取值范围为 (0,1]，默认值为 0.5。

error ：近似分位数的基于排名的误差百分比，取值范围为[0,1)，默认值为 0。当 error =0 时，计算结果为精确分位数。

不忽略输入序列中空缺，error=0 时利用 ndarray.percentile()进行计算，error 不为 0 时将输入序列排序并取[rank-error, rank+error]区间内对应值。输出序列时间戳为 0。

单元测试：

Percentile 对应“root_test_zmy”中数据组“s1”,“s2”、“s6”(取 s2 中前四个数据，$10\log_{10} 4 >$ 4-1)。

1. "s1"	默认值（rank=0.5，error=0），有空缺
2. "s1", “rank": 0.75, “error": 0.03	rank 和 error 均有输入值
3. "s2", “rank": 0.25	
4. 使用示例	

利用“pytest --count=m PercentileUT.py”进行重复测试，耗时如下：

m	Time/s
100000	2.53
1000000	2.39

六、周期 Period 实现及测试

不忽略输入序列中空缺，实现如下：

1.遍历找到与首个元素相等的第 i 个元素，且 i-1 是数据个数的因数，转 2；

2.判断[1,i-1](1 对应首个元素)与[(A-1)*(i+1), Ai]是否相等，A=2,3,…出现不相等则转 1，循环至 Ai 与数据个数相等时说明周期即为 i-1。

输出序列时间戳为 0。

单元测试：

Period 对应“root_test_zmy”中数据组“s4”（有周期，数据类型混杂），“s5”（无周期，开始部分数据存在周期，确保能够测试代码中各个循环），“s7”（有周期，将 s4 周期中某一值全部替换为空缺）。

1. "s4"	有周期，数据类型混杂
2. "s5",	无周期，开始部分数据存在周期
3. "s7"	有周期，存在空缺
5. 使用示例	

利用“pytest --count=m PeriodUT.py”进行重复测试，耗时如下：

m	Time/s
100000	3.40
1000000	3.50