2022.3.14 Global 组合尝试

工作回顾

概率预测组合→→间断需求 (intermittent demand) 的概率预测组合：基准方法池构建与组合方式

- 概率预测方法池构建（上学期）：基于 ***Recent advances in electricity price forecasting: A review of probabilistic forecasting*** (Jakub Nowotarski, Rafał Weron, 2018) 的综述中对概率预测方法的分类：
  - **历史模拟/经验预测区间**：从历史数据的经验分布中抽样估计，在间断需求中用 WSS (Willemain et al. , 2004) 与 VZ（Zhou and Viswanathan ,2011) 这两种基于 Bootstrap 的方法实现。（已完成）
  - **基于分布的概率预测**：假设数据服从某一分布，之后去拟合。对间断需求考虑两种方法：
    1. 不使用外生变量：基于 Snyder 等（2012）提出的概率预测方法，拟合泊松分布、负二项分布、Hurdle shifted Poisson 分布并考虑均值不同的时变方式。（部分完成，现只有 Poisson 分布的两种情形已做出预测结果，但其它方式也只需对似然稍作调整即可获得）
    2. 使用外生变量：这里考虑借鉴 M5 比赛的思路，用网络去利用数据自身及各种外生信息来预测。（尚未完成）
  - **Bootstrap预测区间**：这里指的是拟合均值后用 Bootstrap 抽取残差，用均值+残差伪数据来拟合模型。（对于间断需求，其服从一个离散分布，且去掉均值后残差分布往往与均值有关，故抽取残差也没有意义，这种方法暂不考虑）
  - **分位数回归**：拟合自变量与分位数的关系。间断需求中使用 Gaillard 等 (2016) 在 GEFCom2014 提出的 quantGAM 在intermittent demand 的改进（GAM+计数数据分位数回归）。（已完成）（但是还可以尝试用其它方法进行分位数回归，比如 Load probability density forecasting by transforming and combining quantile forecasts 一文中 (Zhang et al., 2020) 做分位数回归组合时，其使用 GRU (RNN 变种)、随机森林 (RF)、梯度增强回归树 (GBRT)、LGBM 等方法进行分位数预测）
- 组合方式：暂考虑使用分位数预测的组合，理由是：易于组合参数与非参数方法的结果；分位数预测接近于点预测的方法，上手易，方法多；间断需求是计数数据、离散分布，有一定分布函数的形状信息，由足够的分位数也可以得到其概率密度/分布。

  至于怎么组合，接下来给出两种尝试。

固定效应面板分位数回归的Global组合

思路产生

Granger 与 Ramanathan (1984) 提出用线性回归方式组合模型进行点预测的方法，即模型的权重之和不受限，且加入常数项；其结果如下：

- 线性加权组合预测可以看作受约束的回归，这可能导致信息使用不充分；
- 从理论上证明引入回归截距项可以减小 MSE 并修偏（预测误差平均为0，即使有点预测有偏）。数据实证进一步验证。

在此基础上，对于分位数组合，Granger等 (1989) 提出了一个实践性的方法：使用分位数回归组合不同的分位数预测，即进行真值与各基准预测值的分位数回归。该方法实验效果很好，但是没有理论证明。此外，其方法只针对美国失业率与国库券利率的单条序列。

面对M5的数据集，如将参考集（14期分位数预测，5个基准模型的结果*30490条序列）视作一个面板，则是否可以考虑使用面板数据的分位数回归方法，**用固定效应捕捉不同时间序列个体的特性，实现一种Global机制?**

模型设定

固定效应分位数回归使用 R 的 `rgpd` 实现，其基于 Koenker(2004) 的方法，模型表示如下。其中 α_i 表示第 i 个个体的固定效应，$x_{ij}^T\beta(\tau)$ 是第 i 个个体第 j 个观测的线性项：

$$Q_{y_{ij}}(\tau|x_{ij}) = \alpha_i + x_{ij}^\top\beta(\tau) \quad j = 1, \ldots m_i, \quad i = 1, \ldots, n.$$

固定效应项相当于为每个样本设置一个虚拟变量，当个体较多时，固定效应项较多，故在优化目标函数时对 α_i 施以L1惩罚：

$$\min_{(\alpha,\beta)} \sum_{k=1}^{q}\sum_{j=1}^{n}\sum_{i=1}^{m_i} w_k\rho_{\tau_k}(y_{ij} - \alpha_i - x_{ij}^\top\beta(\tau_k)) + \lambda\sum_{i=1}^{n}|\alpha_i|.$$

该式是对多个分位数一起优化损失函数，各分位数的损失权重用 w_k 表示。$\rho_{\tau_k}()$ 代表 pinball loss 损失。 求解与一般的分位数回归类似，原文使用内点法求解。

实验设置

M5的数据最长是1941，取最后14期作为测试集，倒数15-28期作为训练组合模型的参考集（下一节的数据也是如此设置）。

目前已有5种基准模型的分位数预测：基于 Bootstrap 的 WSS 与 VZ；Gaillard 等 (2016) 在 GEFCom2014 提出的 quantGAM 在intermittent demand 的改进（GAM+计数数据分位数回归）；Snyder 等（2012）提出的基于模型的 intermittent demand 概率预测（原文3种分布*3种均值变化模式，共9种；目前只有泊松分布，均值为 Damped dynamic 与 Undamped dynamic 的结果，共两个模型）。

考虑分位数为0.995与0.005（以0.99与0.01代替）、0.975与0.025、0.835与0.165、0.75与0.25，以及0.5；实验时以指定分位数的预测结果为自变量，参考集的值为因变量，作指定分位数的分位数回归，只考虑1个分位数，故 w_k 取1。（由于一开始做基准预测时以0.01递增分位数，故0.835由0.84与0.83的平均值代替，其他情况同理。）

结果展示

以测试集 Pinball loss 在个体与时间的总平均为对比指标，结果如下：

	0.01	**0.025**	**0.165**	**0.25**	**0.5**	**0.75**	**0.835**	**0.975**	**0.99**
quantGAM	0.0195	0.0455	0.2459	0.3494	0.5638	0.5604	0.4787	0.1601	0.0901
VZ	**0.0146**	0.0364	0.2296	0.3320	0.5374	0.5342	0.4566	0.1446	0.0742
WSS	0.0146	0.0365	0.2336	0.3427	0.5740	0.5672	0.4793	0.1519	0.0781
poisson_damped	0.0237	0.0491	0.2394	0.3325	0.5135	0.5114	0.4454	0.1766	0.1141
poisson_undamped	0.0244	0.0496	0.2371	0.3280	0.5073	0.5100	0.4498	0.2020	0.1438
简单平均	0.0174	0.0400	0.2260	0.3227	0.5200	0.5148	0.4386	0.1394	0.0734
面板回归组合	0.0146	**0.0363**	**0.2168**	**0.3093**	**0.4943**	**0.4873**	**0.4165**	**0.1340**	**0.0710**

可见，这种组合方式几乎总能战胜基准预测与简单平均。令人意外的是，组合结果没有出现负值（截距被控制的很好）。

惩罚项；截距项效果；加入特征?

在这个回归中，暂未考虑时变模式；或许可以考虑引入时变系数/截距（用时变截距相对好一些，因为涉及变量较少）来描述时变性，但是没有其它的时变变量，这种时变性仅限于从第一个预测期到第14个预测期的变化，感觉意义不是很大?

LSTM 基于特征元学习组合

思路产生

FFORMA (Montero-Manso, 2020)方法使用 XGBoost 将时间序列的特征与模型组合权重联系起来，但是其只产生了一个在预测期固定的权重，而非时变权重。**如果能有一组随时间变化的特征，可否据此产生一组时变的 Global 权重?** 为了适应时变特征的数据，这里考虑将黑箱方法由 XGBoost 改为使用 LSTM。

实验设置

目前的思路是：

- 数据：取最后14期作为测试集，倒数15-28期作为训练组合模型的参考集；首先计算参考集阶段的时间序列特征，即计算时间序列从有正数据的第一期至参考集第一期、至参考集第二期、……、至参考集第十四期的14条时间序列的特征，所涉及的24个特征如下：
  - 间断数据特征：ADI、CV2、零值比例（这个和ADI相关性有点高）
  - 熵（使用 `tsfeatures` 的 `entropy()` 计算）
  - 基于 STL 的特征（使用 `tsfeatures::stl_features()` 计算）
  - 基于自相关的特征 （使用 `tsfeatures::acf_features()` 计算，数据的频率设为7，即考虑7天的季节相关）
  - 滞后1-7日与滞后1-28日内的均值与标准差

  对于测试集，由于不知道时间序列的真值，以上特征不可直接计算，采用如下方法估计特征：将测试集14期的所有方法的分位数预测结果（0.01-0.99）都进行简单平均，之后利用这14期的分位数情况去抽样获得100条测试期的可能数据（取整以保证是一个需求数据），利用这100个时间序列数据与之前的真值获得测试期的时间序列特征估计（100条时间序列特征结果取均值）。
- 模型搭建：目前网络取一个最简单的形式，**一层LSTM层接一个线性层**；组合方式是以网络的5维输出结果做softmax变换转换为加和为1的权值，用此权值做线性组合；**损失函数是各基准模型 Pinball loss 的加权平均（权是学习的结果）在个体与时间上的总均值**（即 FFORMA 的损失定义）

$$\underset{w}{\operatorname{argmin}} \sum_{n=1}^{N} \sum_{m=1}^{M} w(\boldsymbol{f}_n)_m L_{nm}.$$

（不是组合结果的 Pinball loss 损失函数值，事实上这样做结果不好）；训练时循环100次。下面是网络结构、损失函数与训练过程的核心代码：

```
class lstm(nn.Module):
    def __init__(self,input_size=24,hidden_size=48,output_size=5,num_layer=2):
        super(lstm,self).__init__()
        self.layer1 = nn.LSTM(input_size,hidden_size,num_layer)
        self.layer2 = nn.Linear(hidden_size,output_size)
```

```
    def forward(self,x):
        x,_ = self.layer1(x)
        s,b,h = x.size()
        x = x.view(s*b,h)
        x = self.layer2(x)
        x = x.view(s,b,-1)
        return x

def pinball(x,y,tau):
    return torch.mean((y-x)*tau*torch.ge(y,x)+(x-y)*0.5*torch.gt(x,y))
def pinball_indi(x,y,tau):
    return (y-x)*tau*torch.ge(y,x)+(x-y)*(1-tau)*torch.gt(x,y)

#.....
#训练过程
for e in range(100):
    out = model(data1)
    out1=torch.exp(out)/(torch.sum(torch.exp(out),dim=2).unsqueeze(-1))
    #损失函数

 loss=torch.mean(out1[:,:,0]*pinball_indi(gam0,real_t14,0.975)+out1[:,:,1]*pinba
ll_indi(vz0,real_t14,0.975)+out1[:,:,2]*pinball_indi(wss0,real_t14,0.975)+out1[:
,:,3]*pinball_indi(poi0,real_t14,0.975)+out1[:,:,4]*pinball_indi(poiun0,real_t14
,0.975))
    optimizer.zero_grad()
    loss.backward()
    optimizer.step()

    if (e + 1) % 10 == 0: # 每 100 次输出结果
        print('Epoch: {}, Loss: {:.5f}'.format(e + 1, loss))
```

结果展示

最后构造的组合是由LSTM学习的权重的加权平均。此外作为对比，加入了 XGBoost (FFORMA) 进行模型组合试验，其只能利用14期误差平均值与训练器上的特征。测试集结果如下：

	0.01	**0.025**	**0.165**	**0.25**	**0.5**	**0.75**	**0.835**	**0.975**	**0.99**
简单平均	0.0174	0.0400	0.2260	0.3227	0.5200	0.5148	0.4386	0.1394	0.0734
面板回归组合	0.0146	0.0363	0.2168	0.3093	0.4943	0.4873	0.4165	0.1340	0.0710
LSTM元学习	0.0146	0.0364	0.2207	0.3142	0.5058	0.5018	0.4354	0.1518	0.0716
XGBoost	0.0178	0.0404	0.2287	0.3300	0.6191	0.5659	0.4729	0.4516	0.4218

可以看到，LSTM元学习方法在大多数情形下都可以战胜简单平均，但不能战胜面板回归的结果。可能原因分析：

- （过拟合/网络设置问题）：这个应该不是主要因素，因为能战胜简单平均（和几乎所有基准模型结果），说明网络是有可靠性的。
- **组合方式**：可能需要考虑引入“截距项”，放松权值的设定。关于分位数组合引入截距项的优势，Granger 做了实证，但未理论证明。个人对分位数组合引入截距的理解是：
    - 线性组合：权重不受限更好，因为在线性规划的角度讲，**增加了一个约束后可行域是放松权重的子集**，Pinball loss 应该更差；

    - 线性组合与引入截距项：可能存在类似的修偏作用；根据 Granger 与 Ramanathan (1984) 的研究，常数项可以看作是条件期望的常数估计器与其它模型组合，因此可以认为加入常数项能获得更多模型组合的好处。但是无法更理论的说明/证明，难点在于分位数回归的参数求解不解析。
- 引入截距项，目前尝试了两种办法：
    - 将截距项视为网络的第六个输出，优化目标变为优化组合预测结果的 Pinball loss （与之前的加权损失不同）；
    - 将加权组合后的结果作为一个变量代入面板分位数固定效应回归（单变量回归）。

  结果均不理想，表现为预测方法的权值接近于0，仅靠截距项拟合结果，是一种过拟合。因此在**基于特征的模型组合中引入截距项**是一个问题，而且目前甚至只能做加权为1的组合，否则学习结果也不佳。

目前可能的问题：

- **时间序列不够长**：可供训练的时间序列就14期，纵向长度可能不够（但是加纵向长度意味着需要计算更长时间的基准预测模型以及更长时间的特征）；LSTM的性能可能被浪费
- **特征变化不够**：除了滞后期特征外，其它特征都是对一整条时间序列的特征概括，而M5数据本身长度很长（最长1900+，多数都有上百上千的长度），增加一个新值对特征的影响不会很大。

  如果使用局部序列的特征去解决，是否合适?
- 网络结构设置问题
- **是否需要加额外的偏置/截距**：如何放松权重范围、引入截距；损失函数如何设计（最好不要优化组合预测的 Pinball loss，因为这样更像是学习而不是元学习）
- M5数据在比赛中的预测过程，两个时期是28/28，实验中为了节省计算是14/14，是否需要扩充预测时期。

分位数

Pinball 倒数权重组合

看计量书