原创

腾讯技术工程 | 基于Prophet的时间序列预测

腾讯技术 关注

2018-01-30 16:40:11 81734人阅读 0人评论

预测未来永远是一件让人兴奋而又神奇的事。为此，人们研究了许多时间序列预测模型。然而，大部分的时间序列模型都因为预测的问题过于复杂而效果不理想。这是因为时间序列预测不光需要大量的统计知识，更重要的是它需要将问题的背景知识融入其中。为此，Prophet充分的将两者融合了起来，提供了一种更简单、灵活的预测方式，并且在预测准确率上达到了与专业分析师相媲美的程度。如果你还在为时间序列预测而苦恼，那就一起走进兴奋而又神奇的Prophet世界吧。

前言

时间序列预测一直是预测问题中的难点，人们很难找到一个适用场景丰富的通用模型，这是因为现实中每个预测问题的背景知识，例如数据的产生过程，往往是不同的，即使是同一类问题，影响这些预测值的因素与程度也往往不同，再加上预测问题往往需要大量专业的统计知识，这又给分析人员带来了难度，这些都使得时间序列预测问题变得尤其复杂。

传统的时间序列预测方法，例如ARIMA（autoregressive integrated moving average）模型，在R与Python中都有实现。虽然这些传统方法已经用在很多场景中了，但它们通常有如下缺陷：

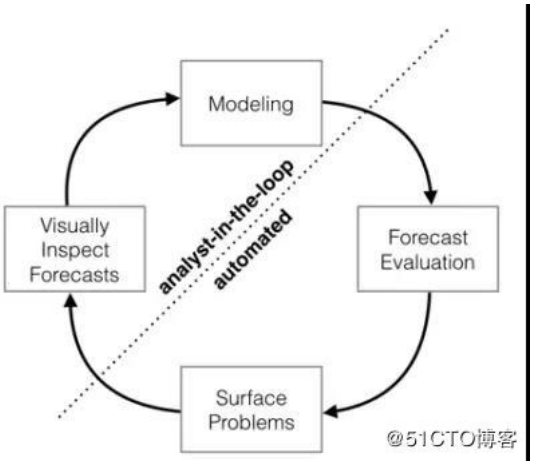
a.适用的时序数据过于局限
例如最通用的ARIMA模型，其要求时序数据是稳定的，或者通过差分化后是稳定的，且在差分运算时提取的是固定周期的信息。这往往很难符合现实数据的情况。
b.缺失值需要填补
对于数据中存在缺失值的情况，传统的方法都需要先进行缺失值填补，这很大程度上损害了数据的可靠性。
c.模型缺乏灵活性
传统模型仅在于构建数据中的临时依赖关系，这种模型过于不够灵活，很难让使用者引入问题的背景知识，或者一些有用的假设。
d.指导作用较弱
当前，虽然R与Python中实现了这些方法并提供了可视化效果，降低了模型的使用门槛。但由于模型本身的原因，这些展现的结果也很难让使用者更清楚地分析影响预测准确率的潜在原因。

总之，传统的时间序列预测在模型的准确率以及与使用者之间的互动上很难达到理想的融合。

近期，facebook发布了prophet（“先知”）项目，它以更简单、灵活的预测方式以及能够获得与经验丰富的分析师相媲美的预测结果引起了人们的广泛关注。下面我们介绍一下Prophet。

Prophet介绍

2.1整体框架



上图是prophet的整体框架，整个过程分为四部分：Modeling、Forecast Evaluation、Surface Problems以及Visually Inspect Forecasts。从整体上看，这是一个循环结构，而这个结构又可以根据虚线分为分析师操纵部分与自动化部分，因此，整个过程就是分析师与自动化过程相结合的循环体系，也是一种将问题背景知识与统计分析融合起来的过程，这种结合大大的增加了模型的适用范围，提高了模型的准确性。按照上述的四个部分，prophet的预测过程为：

a.Modeling：建立时间序列模型。分析师根据预测问题的背景选择一个合适的模型。

b.Forecast Evaluation：模型评估。根据模型对历史数据进行仿真，在模型的参数不确定的情况下，我们可以进行多种尝试，并根据对应的仿真效果评估哪种模型更适合。

c.Surface Problems：呈现问题。如果尝试了多种参数后，模型的整体表现依然不理想，这个时候可以将误差较大的潜在原因呈现给分析师。

d.Visually Inspect Forecasts：以可视化的方式反馈整个预测结果。当问题反馈给分析师后，分析师考虑是否进一步调整和构建模型。

2.2适用场景

前文提到，不同时间序列预测问题的解决方案也各有不用。Prophet适用于有如下特征的业务问题：

a.有至少几个月（最好是一年）的每小时、每天或每周观察的历史数据；

b.有多种人类规模级别的较强的季节性趋势：每周的一些天和每年的一些时间；

c.有事先知道的以不定期的间隔发生的重要节假日（比如国庆节）；

d.缺失的历史数据或较大的异常数据的数量在合理范围内；

e.有历史趋势的变化（比如因为产品发布）；

f.对于数据中蕴含的非线性增长的趋势都有一个自然极限或饱和状态。

2.3 模型原理

模型的整体构建如下：

$$y(t) = g(t) + s(t) + h(t) + \epsilon_t.$$ @51CTO博客

模型（1）整体由三部分组成：growth(增长趋势)、seasonality(季节趋势)以及holidays(节假日对预测值的影响)。其中g(t)表示增长函数，用来拟合时间序列中预测值的非周期性变化；s(t)用来表示周期性变化，比如说

每周，每年中的季节等；h(t)表示时间序列中那些潜在的具有非固定周期的节假日对预测值造成的影响。最后ε_t为噪声项，表示模型未预测到的波动，这里假设ε_t是高斯分布的。

可以看出这是一种类似generalized additive model(GAM)的模型，不同于以往的时间序列预测模型（例如ARIMA），上述的模型将预测问题视作曲线拟合问题。这样做具有很多实践价值：

a.灵活度高，许多具有不同周期以及不同假设的季节性趋势能很容易的被引入；

b.时间序列中无需有一个固定的周期，也不需要在拟合前对缺失值进行填补，这是传统的（例如ARIMA）模型所办不到的；

c.拟合非常快，允许分析师交互式的探索模型的效果；

d.模型中参数的解释性很强，可以让分析师根据启发来增强某部分假设。

下面分别介绍模型中各部分的构建。

2.3.1 增长趋势

增长趋势是整个模型的核心组件，它表示认为整个时间序列是如何增长的，以及预期未来时间里是如何增长的。这部分为分析师提供了两种模型：Non-linear growth（非线性增长）和Linear growth（线性增长）。

1.Non-linear growth

非线性增长的公式采用了逻辑回归的模型：

$$g(t) = \frac{C}{1 + \exp(-k(t-b))}, \qquad (2)$$

这里，C是承载量，它限定了所能增长的最大值，k表示增长率，b为偏移量。
当然，实际的增长模型远没有这么简单，Prophet主要考虑了两个现实问题：

（1）C值并不一定是常数；（2）增长率也不一定是一沉不变的。对于（1），将C构建成随时间变化的函数：C(t) = K 或者 C(t) = Mt + K。下面详细论述。

（2）的解决：首先模型定义了增长率k发生变化时对应的点，我们将其称作**changepoints**，用s_j表示，这些点对应的斜率调整值用表示，所有的斜率调整值形成一个向量δ_j。此时，每个changepoint点对应的增长率就变为$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise} \end{cases}$。如果有如下定义：

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise} \end{cases}$$

则t时刻的增长率就可以表示为：$k + a(t)^T\delta$

当增长率k调整后，每个changepoint点对应的偏移量b也应该相应调整以连接每个分段的最后一个时间点，表达式如下：

$$\gamma_j = \left(s_j - b - \sum_{l<j}\gamma_l\right)\left(1 - \frac{k + \sum_{l<j}\delta_l}{k + \sum_{l\le j}\delta_l}\right)$$

综上，结合（1）和（2），最终的分段式逻辑回归增长模型为：

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^\intercal\boldsymbol{\delta})(t - (b + \mathbf{a}(t)^\intercal\boldsymbol{\gamma})))}. \quad (3)$$

2.Linear growth

如果认为时间序列的整体增长趋势是线性的，那么就可以采用线性模型：

$$g(t) = (k + \mathbf{a}(t)^\intercal\boldsymbol{\delta})t + (b + \mathbf{a}(t)^\intercal\boldsymbol{\gamma}),$$

这里的参数定义与非线性增长一样，唯一不同的是每个changepoint对应的

结合上述两种增长模型，我们可以看到，对于增长趋势的预测，最重要的就是对这些changepoint的指定。使用时，既可以手动指定这些changepoint，也可以根据公式（3）和（4）自动识别。此时，认为

$\delta_j \sim$ Laplace(

其中 控制着模型整体的平滑程度。

2.3.2 季节性趋势

由于时间序列中有可能包含多种周期类型的季节性趋势，因此，傅里叶级数可以用来近似表达这个周期属性，公式如下：

$$s(t) = \sum_{n=1}^{N} c_n e^{i\frac{2\pi nt}{P}}$$

其中，P表示某个固定的周期（例如用"天"做单位统计的数据中，年数据的P = 365.25，周数据的P = 7）。2N表示我们希望在模型中使用的这种周期的个数，较大的N值可以拟合出更复杂的季节性函数，然而也会带来更多的过拟合问题。按照经验值，年周期的N取10，周周期的N取3。

当将s(t)中的所有季节性时间序列模型组合成一个向量X(t)，那么最终的季节性模型为：

$$s(t) = X(t)\boldsymbol{\beta}.$$

$\boldsymbol{\beta} \sim$ Normal(

其中， ，以此提高季节性模型的平滑性。

2.3.3 节假日模型

很多实际经验告诉我们，节假日或者是一些大事件都会对时间序列造成很大影响，而且这些时间点往往不存在周期性。对这些点的分析是极其必要的，甚至有时候它的重要度远远超过了平常点。

鉴于每个节假日（或者某个已知的大事件）的日期与影响程度存在差异，节假日模型将不同节假日在不同时间点下的影响视作独立的模型。同时为每个模型设置了时间窗口，这主要是考虑到节假日的影响有窗口期（例如中秋节的前几天与后几天），模型将同一个窗口期中的影响设置为相同的值。例如，i表示节假日 表示窗口期中包含的时间t，则节假日模型h(t)可表示为：

$$h(t) = \sum_{i=1}^{L} \kappa_i \mathbf{1}(t \in D_i). \quad (7)$$

$$Z(t) = [\mathbf{1}(t \in D_1), \ldots, \mathbf{1}(t \in D_L)]$$

那么

$$h(t) = Z(t)\kappa.$$

其中

$$\kappa \sim \text{Normal}(0, \sigma^2)$$

Prophet的使用

3.1参数使用

下面是这个模块的参数解释，使用者可充分利用这些参数调整模型：

a.增长趋势的模型参数

growth：增长趋势模型。整个预测模型的核心组件，分为两种：” linear”与”logistic” ，分别代表线性与非线性的增长，默认值：” linear”。

cap：承载量。非线性增长趋势中限定的最大值，预测值将在该点达到饱和。当选择非线性增长时，该项值必须给出。

changepoints(growth模型中的)：改变点。使用者可以自主填写已知时刻的标示着增长率发生改变的”改变点” ，如果不填则系统自动识别。默认值：“None”。

n_changepoints：用户指定潜在的”changepoint”的个数，默认值：25。

changepoint_prior_scale(growth模型中的)：增长趋势模型的灵活度。调节”changepoint”选择的灵活度，值越大，选择的”changepoint”越多，从而使模型对历史数据的拟合程度变强，然而也增加了过拟合的风险。默认值：0.05。

b.季节趋势的模型参数

seasonality_prior_scale（seasonality模型中的）：调节季节性组件的强度。值越大，模型将适应更强的季节性波动，值越小，越抑制季节性波动，默认值：10.0。

c.节假日的模型参数

holidays_prior_scale（holidays模型中的）：调节节假日模型组件的强度。值越大，该节假日对模型的影响越大，值越小，节假日的影响越小，默认值：10.0。

holidays：节假日的定义，设置节假日的json格式的配置文件，例如：

其中"holiday"表示某类节假日的名称，"ds"指定具体的节假日期，"lower_window"表示该节假日包括指定日期之前的多少天，"upper_window"表示该节假日包括指定日期之后的多少天，上述四个参数均需要配置。

d.预测中需要的其他参数

freq：数据中时间的统计单位（频率），默认为" D" ，按天统计，具体可参考这里。

periods：需要预测的未来时间的个数。例如按天统计的数据，想要预测未来一年时间内的情况，则需填写365。

mcmc_samples：mcmc采样，用于获得预测未来的不确定性。若大于0，将做mcmc样本的全贝叶斯推理，如果为0，将做最大后验估计，默认值：0。

interval_width：衡量未来时间内趋势改变的程度。表示预测未来时使用的趋势间隔出现的频率和幅度与历史数据的相似度，值越大越相似，默认值：0.80。当mcmc_samples = 0时，该参数仅用于增长趋势模型的改变程度，当mcmc_samples > 0时，该参数也包括了季节性趋势改变的程度。

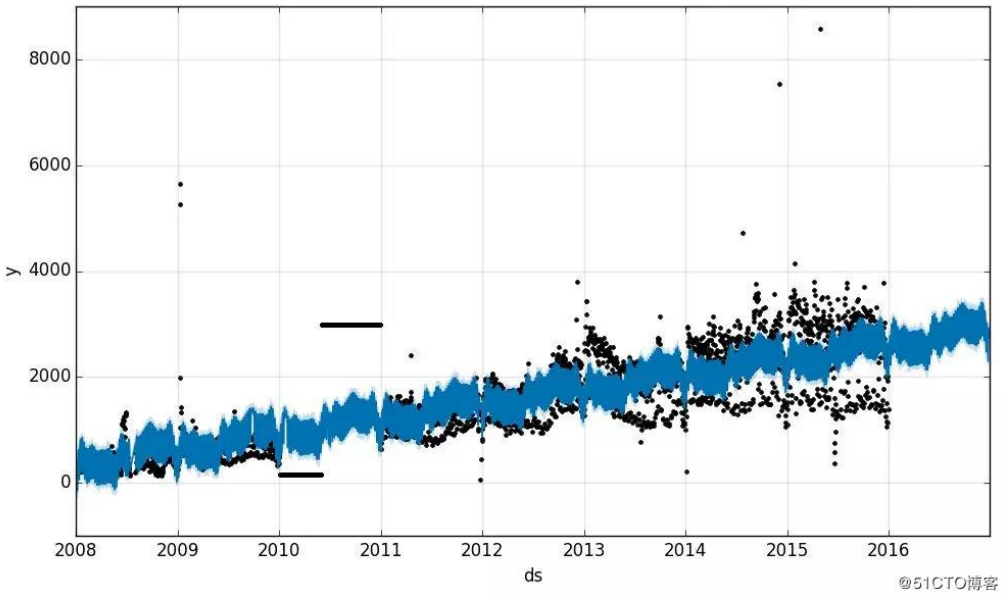
uncertainty_samples：用于估计未来时间的增长趋势间隔的仿真绘制数，默认值：1000。

3.2 结果读取与分析

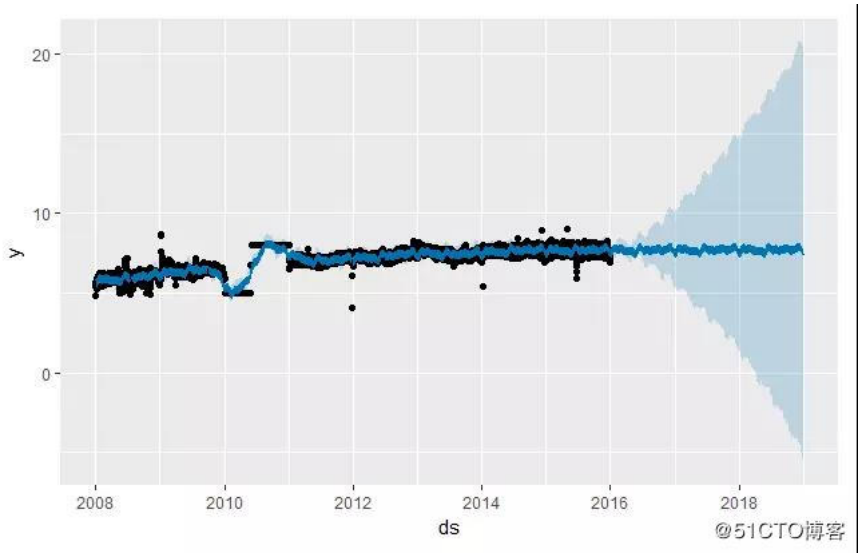
完成以上的配置后，接下来就可以直接运行模型并获得结果了。

3.2.1 可视化结果

整体预测情况是我们衡量模型整体预测效果的一个最直接的方式，它是我们评估当前模型的预测水平的重要来源。同时可视化的展示可以帮助我们有效分析预测结果中各个时间阶段的预测效果。

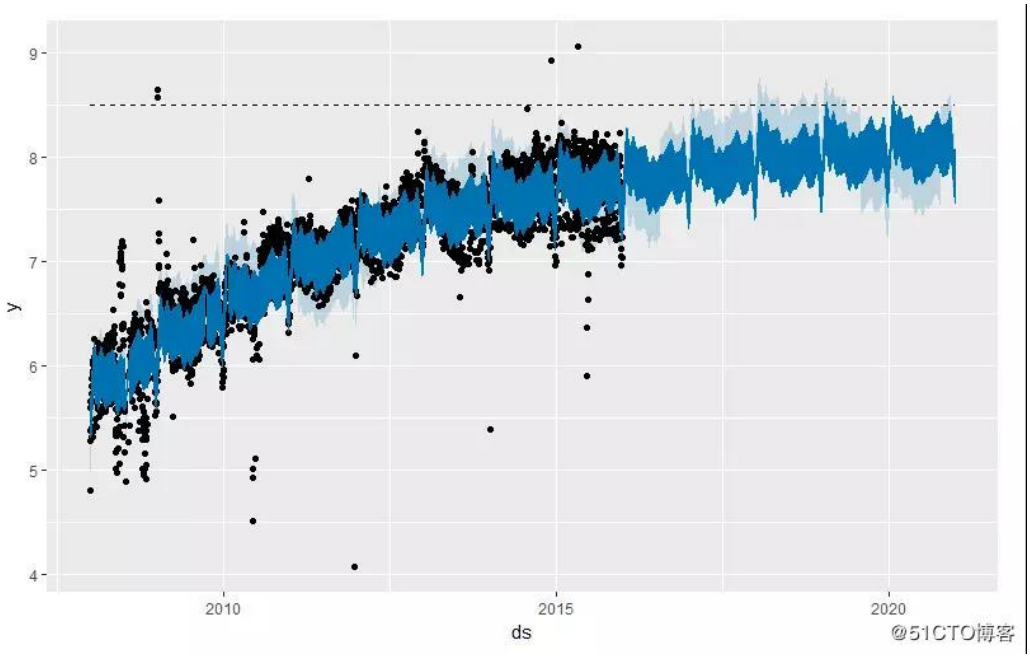


上图是一个整体的预测结果图，它包含了从历史数据的时间起点到期望预测的未来时间终点的结果。图中的ds坐标表示时间，y坐标对应预测值。图中的黑点表示已知的历史数据，由图上我们很容易发现数据中的异常点，蓝色曲线表示模型的预测值。仔细查看蓝色曲线，我们可以发现，曲线轮廓的上下边界有浅蓝色区域，它表示模型预测值的上、下边界。在评估结果时，我们将蓝色曲线的预测值视作主预测值，上、下边界的预测值作为参考。除此之外，浅蓝色区域还可以很好的用于模型评估，例如对于下面这个图：



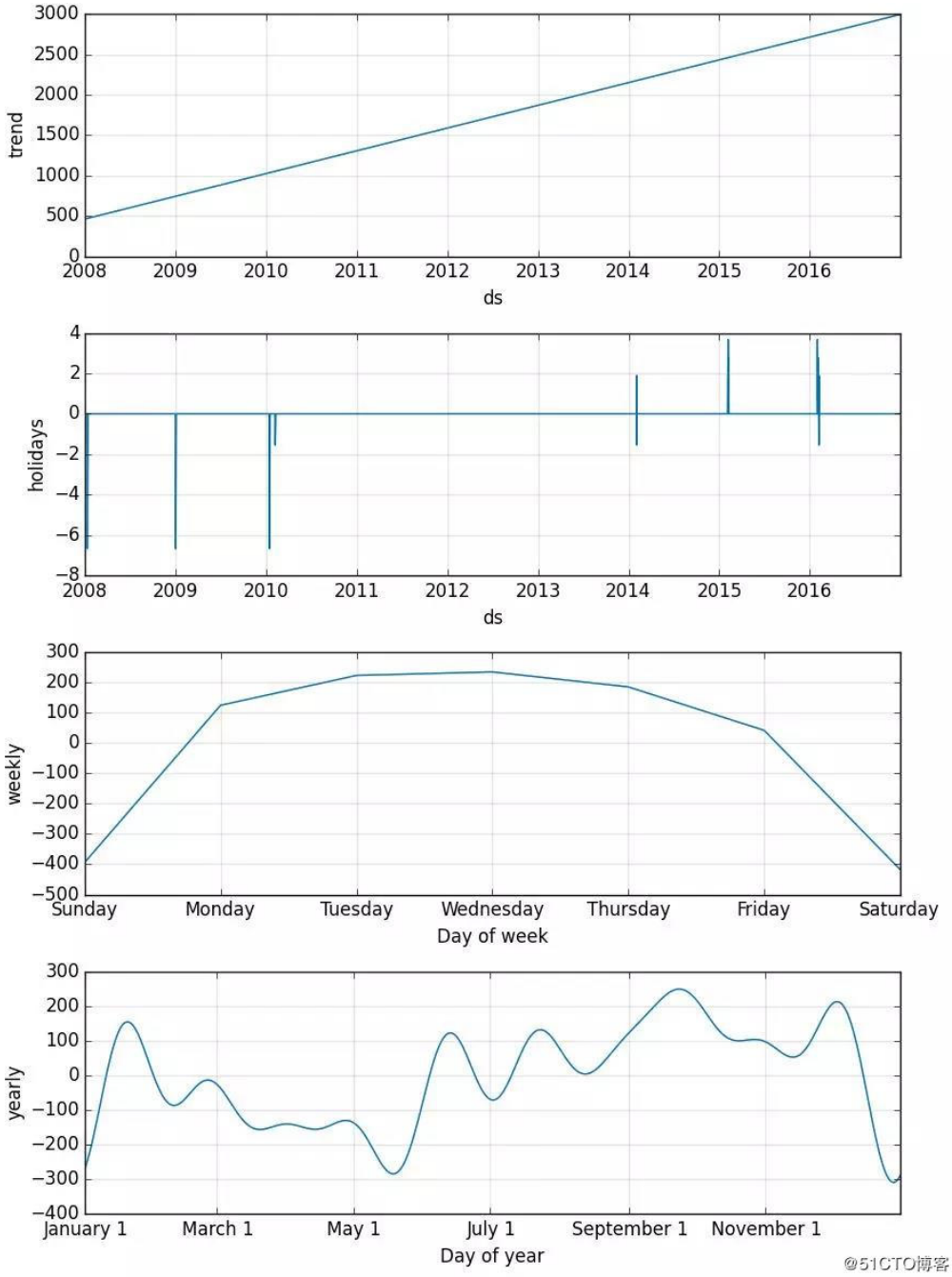
在2016年之后的模型预测部分，浅蓝色区域就过于宽泛，模型预测的上、下边界被逐渐放大很多倍。这说明模型的平滑性过大，导致异常点对结果造成了很大影响。因此，该模型不够合理，需要使用者重新设置参数或者对历史数据中的异常点进行预处理。

上述图是growth选择”linear”时的结果，如果认为时间序列呈非线性增长趋势，我们用如下的图例来说明：



体上与线性增长的结果表达没有太大差异，唯一需要注意的是，上图中的水平虚线表示了非线性增长趋势的承载量cap，预测结果将在该虚线处达到饱和。

除了上述的整体预测情况外，Prophet还提供了组成成分分析（简称成分分析），所谓成分分析就是指对公式（1）中的三大部分模型单独进行分析，成分分析有助于我们考察模型中的各个组件分别对预测结果的影响，通过可视化的展示，我们可以准确判断影响预测效果的具体原因，从而针对性的解决。成分分析是我们提高模型准确性的重要来源。例如下图结果：



上述四个图从上至下依次是对增长趋势模型（trend）、节假日模型（holidays）以及季节性模型（weekly和yearly）的展示。需要注意的是，如果没有在holidays参数里注明具体的节假日信息，模块也不会自动对这一部分进行分析。如果对于上面的结果你觉得有不合理的地方，那么可以根据2.1中参数使用说明更改相应的成分影响，这里应该尽可能的利用你的专业背景知识，以使各部分组成的影响更符合实际。举个例子，如果在每年趋势”yearly”中你认为当前的效果过拟合了，那么就可以调解seasonality_prior_scale这个参数，值越小，这里的季节性波动就越小。

对于上面的可视化分析，这里**总结几点建议**，方便大家定位预测中的问题：

a.如果预测结果的误差很大，考虑选取的模型是否准确，尝试调整增长率模型（growth）的参数，在必要的情况下也需要调整季节性（seasonality）参数。

b.如果在尝试的大多数方法中，某些日期的预测依然存在很大的误差，这就说明历史数据中存在异常值。最好的办法就是找到这些异常值并剔除掉。使用者无需像其他方法那样对剔除的数据进行插值拟合，可以仅保留异常值对应的时间， 并将异常值修改为空值（NA），模型在预测时依然可以给出这个时间点对应的预测结

c.如果对历史数据进行仿真预测时发现，从一个截点到下一个截点误差急剧的增加，这说明在两个截点期间数据的产生过程发生了较大的变化，此时两个截点之间应该增加一个"changepoint"，来对这期间的不同阶段分别建模。

参考文献

Sean J. Taylor and Benjamin Letham.Forecasting at Scale.https://research.fb.com/prophet-forecasting-at-scale

腾讯 大数据 时间序列

0 收藏 分享

上一篇：如何节省1TB图片带宽？解密极致... 下一篇：腾讯技术工程 | 新一代企业级容...

腾讯技术
116篇文章，49W+人气，21粉丝
公众号 腾讯技术工程
关注

提问和评论都可以，用心的回复会被更多人看到和认可

Ctrl+Enter 发布 取消 发布

在线客服

推荐专栏 更多

1 分享 腾讯技术 关注