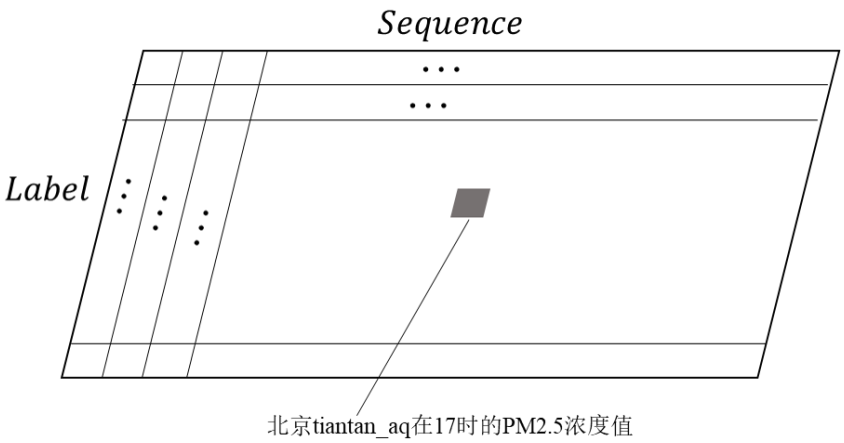
KDD CUP实验报告

丁雨晖　计54班　2015010866

马子轩　计54班　2015012283

数据处理

对于所给历史数据，我们读取并解析了csv文件，将数据保存为二维numpy数组的形式，一维表示数据标签（例如北京某一观测站的PM2.5浓度），另一维则表示该类数据在不同时间节点上的取值（如下图所示）。时间序列从2017-01-01 0:00:00开始，逐小时递增。



在处理过程中我们发现存在数据缺失的情况，部分数据甚至在很长的一段时间内全部缺失。初始化时我们暂时将这些缺失值置为0，这样可以使不同种类的数据在存储格式上保持一致，便于之后的计算。接下来我们尝试了几种方式对缺失数据进行填补：

1. 使用均值进行填补，这样做的问题在于当遇到大量连续缺失的数据时填补的值可能仍然是0
2. 使用非0值的中位数进行填补
3. 使用盒图，判断置为0值的这些点是否离群。缺失数据较少时，这些离群点可用1/4点或3/4点的值来替代。若遇到大量连续缺失的情况，则用等差数列进行填补。我们最终采用了这一方法。

我们专门编写了一部分代码来获取官方发布的新的观测数据并对本地数据进行更新，在每次跑预测模型之前都会先运行这部分代码。这部分代码的功能很简单，只是将新的数据按照历史数据的组织方式构成二维numpy数组，然后和历史数据在时间序列这一维上拼接起来。我们用文件记录了当前数据的截至时间，这样可以保证每次数据更新完后时间序列始终是连续的。将新获取的数据与历史数据拼接完成后，我们再按照上述填补缺失数据的方法对数据进行处理。

获取新数据时我们遇到了一个问题，不同种类数据的最新时间是不一致的。例如北京的污染物浓度可能更新到了15时，而气象数据可能只更新到了12时。我们的做法是选取较早的作为我们新的截至时间，在上面的例子中也就是12时。这样污染物浓度13时-15时的数据我们可以在下次更新时获得，不会遗漏，而那时气象数据可能也更新了13时-15时，因此获得了更多的真实数据，减小了填补缺失数据可能对预测结果造成的不利影响。

此外，我们也利用了天气预报数据，以使我们在预测未来某一时间点的污染物浓度时可以结合该时间点的气象预报数据。

模型构建

ARIMA

ARIMA(Autoregressive Integrated Moving Average Model)是一种时间序列预测算法，其数学表示为：

$$\widehat{y_t} = \mu + \phi_1 * y_{t-1} + \ldots + \phi_p * y_{t-p} + \theta_1 * e_{t-1} + \ldots + \theta_q * e_{t-q}$$

其中p为滞后参数，q为预测误差滞后参数，ϕ为AR的参数，θ为MA的参数。

ARIMA算法要求被预测序列是稳定的，如果不稳定，则通过差分来使序列稳定。序列的稳定条件为：序列的均值与方差仅与取的窗口长度相关，而与时间无关。在应用这一模型时，通过观察我们发现数据有24小时的周期性，于是我们使用24小时进行差分，即$y'_t = y_t - y_{t-24}$，经过验证得知，差分后序列达到稳定。

考察差分后的数据，早于48小时的数据对当前时间点的数据几乎没有影响，因此我们使用前48小时的数据进行拟合，即$p = 48$。

关于q的取值，我们发现在剔除前17小时数据点的干扰后，第前18小时数据对当前时间点的数据没有干扰，因此$q = 18$。

设d表示差分阶数，我们对数据的处理方式已经将序列转化为稳定序列，不需要再额外进行差分，因此我们不考虑d的取值。

从结果上来看，ARIMA算法在每一天大部分站点都能达到0.7左右，但仍有部分站点得分大于1。

应用ARIMA模型的过程中，我们发现由于ARIMA的训练过程中有大量方程求解，运算过于复杂，导致每天更新占用运算资源过多，计算时间过长，因此在实用性上有较大不足，我们考虑使用更高效的方法。

OLS

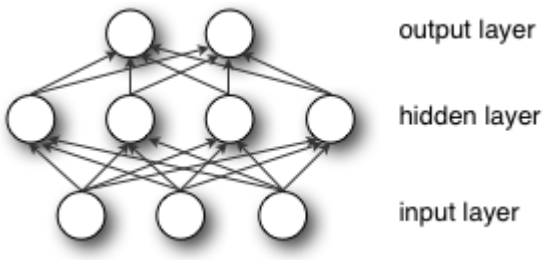
在考虑到ARIMA的不足后，我们决定直接使用线性回归分析。我们仍然利用了ARIMA模型中的结论，即使用24小时进行差分得到稳定序列，以及使用前48小时数据进行回归。此外，我们额外使用了网格气象数据，并添加了$cos(t/24)$一项。添加这一项是因为原数据本身具有24小时的周期性，使用$cos(t/24)$能够较好地捕捉数据特征。

经过验证，使用这一方法大部分预测站点能达到0.5左右的分数，仍有少部分站点的分数大于1。这种方法在我们尝试的方法中表现最好，因此我们选取它为最终方案。

这些回归方法之外，我们又对神经网络模型进行了尝试。

MLP

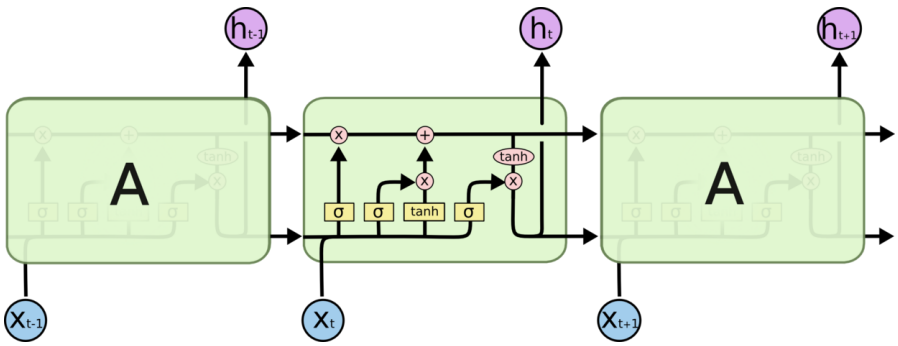
多层感知机MLP(Multi-Layer Perceptron)是最简单的一类前馈人工神经网络，其结构可用图表示为：



我们使用当前时间节点前72小时的污染物浓度数据预测当前节点的污染物浓度，进而递归预测48小时的污染物浓度。我们的网络结构可以表示为$FC(1000)-FC(35*6)$，其中$FC(c)$表示有c个输出的全连接层，两层之间有$batch\ normalization$和$ReLUs$。我们使用均方误差作为损失函数，采用$mini-batch\ gradient\ descent$对模型进行训练。这一模型预测的结果得分可以达到0.6左右。

LSTM

LSTM(Long Short-Term Memory)网络是对RNN网络的一种改进，它引入了一种名为$cell$的结构，可以更有效地应对长期依赖问题，更适合于发现较长时间之前的数据对此次预测的影响。LSTM的典型结构如下：



我们将每一时间点前48小时的污染物浓度数据及此时间点的气象数据作为网络的输入，输出待预测的48小时的空气质量数据。

我们设计的网络结构为：

$$LSTM(64) - LSTM(32) - DENSE(64) - ReLUs - DENSE(1)$$

我们将batchsize设为64，一共训练了8个epoch，预测结果的得分在0.7左右，但不同站点的波动比较大。

我们的网络结构设计是经验上的，对于这一具体问题应该有更优化的方案。

实现工具

全部代码使用python实现，用到的包主要有：pickle，numpy，sklearn，keras

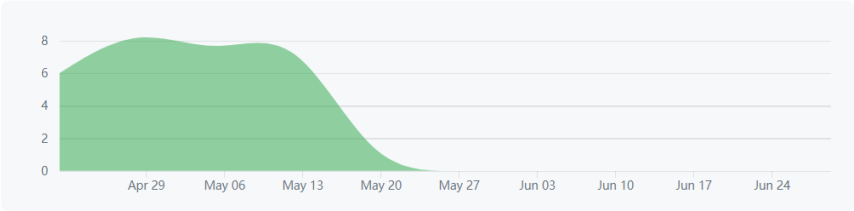
代码架构

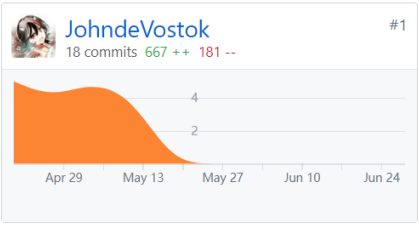
NAME	DESCRIPTION
init.py	组织历史数据
update.py	爬取新发布的数据并更新本地数据
getforecast.py	获取天气预报数据
fix.py	利用盒图填补缺失数据，处理离群点
fit.py	训练回归模型
predict.py	预测
submit.py	提交

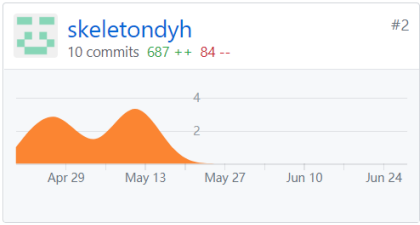
运行时按照列表中由上到下的顺序依次运行。

工作量及小组分工

丁雨晖同学负责进行数据处理并实现了MLP模型，马子轩同学实现了ARIMA及LSTM模型。全部代码位于https://github.com/JohndeVostok/KDD-CUP。


8
6
4
2
0
Apr 29
May 06
May 13
May 20
May 27
Jun 03
Jun 10
Jun 17
Jun 24


JohndeVostok
#1
18 commits 667 ++ 181 --
4
2
Apr 29
May 13
May 27
Jun 10
Jun 24


skeletondyh
#2
10 commits 687 ++ 84 --
4
2
Apr 29
May 13
May 27
Jun 10
Jun 24