**应用时间序列**

**课程作业实验报告**

|  |
| --- |
| 金融工程 颜一凡 171098506 |
| 金融工程 丁安节 171870528 |
| 金融工程 雷欣南 171870600 |
| 工业工程 徐懋成 171870536 |
| 金融工程 陈 凯 171870553 |

解决思路：ARIMA 🡪 Fourier级数法 🡪 长短记忆神经网络（LSTM）

最好结果：smape=0.39（LSTM提供）

团队分工：

资料论文查询：许凯

报告写作：丁安节、徐懋成

ARIMA：雷欣南

Fourier级数法、LSTM：颜一凡

**1. 数据预处理**

我们拿到数据时，首先就进行了对数据的初步分析和基于数据特征的预处理工作。

我们看到我们的数据中存在未给出的空值，空值必须最先处理，考虑到数据的合理性，我们选择使用数据在各剩余时间上的均值来替代空值。

此外，随着时间的推移，互联网和人们的思维越发开放，维基百科在中国的点击率也随之上升，这是每个数据都可能存在的趋势。所以我们对数据使用了ln函数来去趋势化。

我们还注意到了数据的规模各不相同，规模不一的数据会导致参数规模也不一，从而导致训练错误。所以我们在数据预处理时，就需要统一数据规模，对数据做归一化处理。我们将最小值都设为0，预处理的代码为preprocess.m, 之后将pre中的数据复制到preprocessed.xlsx中。

通过这个处理，我们将数据都成比例变化到0和1之间，消除了数据规模的影响。

经过预处理的通过这次大作业，我们认识到，完成一个工程和掌握一个算法是有很大差别的。在工程当中，原始数据质量往往参差不齐，空值很多。而数据预处理则是根据数据本身呈现出来的结构特性，结合模型需求，对原始数据预先处理的过程。没有数据预处理，算法中对数据所作出的假设就可能被违背，导致整个工程的结果大打折扣。数据预处理是工程中极为重要的一步。

**2. ARIMA模型**

**2.1 模型简述**

ARIMA模型是一种时间序列的预测方法。中，AR代表“自回归”，为自回归项数，MA为“滑动平均 ”，为滑动平均项数，指的是使其成为平稳序列所做的差分次数。

假设表示时刻的Y的差分，有：

假设已知，可以表示为：

通常情况下，可以表示为：

ARIMA模型不需要直接考虑其他相关随机变量的变化，使用ARIMA模型的步骤如下：

|  |
| --- |
| 1. 判断时间序列数据的平稳性，即通过时间序列的图像，自相关系数和偏自相关系数图像识别数据的平稳性；  2. 对非平稳的数据进行平稳处理；  3. 根据识别出来的特征建立相应的模型，如AR模型，MA模型和ARMA模型；  4. 进行参数估计，检验是否具有统计意义；  5. 进行假设检验，判断残差是否为白噪音；  6. 利用所得模型进行预测。 |

**2.2 具体实施**

我们通过SAS实现ARIMA模型，对于100个序列的SAS代码位于code.sas中。具体步骤如下：

1.对于训练集数据进行预处理，采用log(x+1)，然后归一化（每个值除以最大值）对于空值，采用周围平均值补齐，处理过的训练集为train\_processed.xlsx，对应的各自最大值为max.xlsx。为了方便操作，训练集使用EXCEL转置，并且将其名称换成了‘ob1’,'ob2'...'ob100'对应'2NE1\_zh.wikipedia.org\_all-access\_spider'...'TFBOYS\_zh.wikipedia.org\_all-access\_spider'；

2.经过研究，序列数据一阶差分后较为平稳，采用ARIMA模型对训练集进行拟合与预测；

3.对于序列的ACF/PACF图像进行观察，并且结合SAS自带的minic选项，使用BIC信息量查看较为适合的模型；

4.选定模型进行预测和比较，选择合理的模型进行预测；

5.根据训练集进行60步预测，得到对应的结果（各自结果放在‘单个预测’文件夹内），再将结果乘最大值后取exp()处理并且-1，还原至原来值，四舍五入取整（此步骤在EXCEL中完成，由于处理过程简单，不在文件夹中包括），最后把结果总结至predict.csv中；

6.用python绘制对比图和计算，存储在Comparison.pdf中，smape计算得出为0.5186838552940602>0.5这不是一个好的预测，ARIMA模型可能并不能较好地提取出训练集的信息，python处理代码在proccess.py中，预测结果在submisson\_arima.csv中

**2.3 结果展示**

下面展示通过ARIMA模型预测的部分结果，其中蓝色折线表示真实值，橙色折线表示预测值（全部结果见Comparison.pdf）。





由于ARIMA模型本质上只能捕捉线性关系，而不能捕捉非线性关系，ARIMA模型可能并不能较好地提取出训练集中的信息，因此，我们有下面的处理。

**3. Fourier级数法**

通过观察经预处理训练集的图像，为了把握周期性，对每个训练集做Fourier级数展开，阶数展开到6阶，对训练集重新做一次平滑。

之后，为捕获长期的特征，对每个要预测的值，分别取该日期前6、12、18、30天的经过Fourier平滑的平均值，将这四个平均值再做一次平均值，得到预测结果（注：直接用预测值用于下一天的预测），最后经过还原得到的结果为smape=0.4939

**3.1 实现细节**

通过MATLAB读取proprecessed.xlsx，在用fit(x,y,’fourier6’)做拟合，之后做两次平均值即可。代码在fourier\_analysis.m中，预测结果在submisson\_fourier.csv中

**3.2 模型优缺点**

该模型具有一定启发式算法的特点，预测速度快，标压i7处理器可以在10秒内完成

但该模型预测结果大多也是一条直线，精度不高，还是无法把握长期特征。只能对周期性有一定处理。

为了把握长期特征，采用下面的LSTM模型。

**4. LSTM模型**

**4.1 梯度消失现象**

RNN（循环神经网络）也常用于时间序列预测。

为什么要采用LSTM而不直接使用RNN呢？

在RNN模型中，若序列过长，将会导致梯度消失现象，使得模型的训练收敛缓慢，无法进行。

在一个RNN模型中，输入和输出往往满足这样的关系：

我们经常使用随机梯度下降法来训练这样的RNN模型的参数：

这里的表示训练模型损失函数的梯度。所以我们可以看到，的绝对值的大小，对模型的收敛速度有着很大的影响。而在误差反向传播的过程中，我们看到：

当每层的输入通过sigmoid函数来增加非线性时，通过导数的链式法则看出，梯度需要乘上一个小于1的导数。序列每增加一层，sigmoid函数就会多使用一次；每一次sigmoid函数的嵌套，都会使得在误差反向传播时，梯度越来越小。如此，最后得到的梯度将会远小于1，导致梯度下降算法的每一步都很小，参数学习速度变慢，收敛缓慢。

**4.2 LSTM模型**

由于在本题中，恰好是序列非常长的情形，RNN模型势必会因为梯度消失问题而瘫痪，所以我们想到了Long Short Term Memory（LSTM）模型来增加模型的记忆力。

相较于RNN，LSTM在其简单框架的基础上，加入了三个门来达到长期记忆的效果，分别是遗忘门，输入门和输出门。在LSTM中长期状态被表示为cell state。每一次新的长期状态的产生，都是当前状态和长期状态的结合。但由于三个门的存在，LSTM对当前状态的记忆具有选择性，而不是简单的按比例分配新长期状态中两个部分的构成。这种选择性记忆，使得LSTM就具有了注意力机制，可以忽略不重要的信息，而记住有效的信息。这种模型的设计，既避免了RNN由于序列过长导致的梯度消失现象，也提高了模型对过去的记忆能力。

**4.3 具体实现**

代码在LSTM.py中，采用了python中Keras的LSTM的Sequential、Dense、LSTM包和sklearn.preprossing中的预处理包MinMaxScaler。

在数据方面，没有使用我们的预处理数据，直接用的MinMaxScaler做预处理。

**特别注意：﻿要顺利运行此模型一定要用我的train.csv**

**该文件末尾复制了test.csv数据**

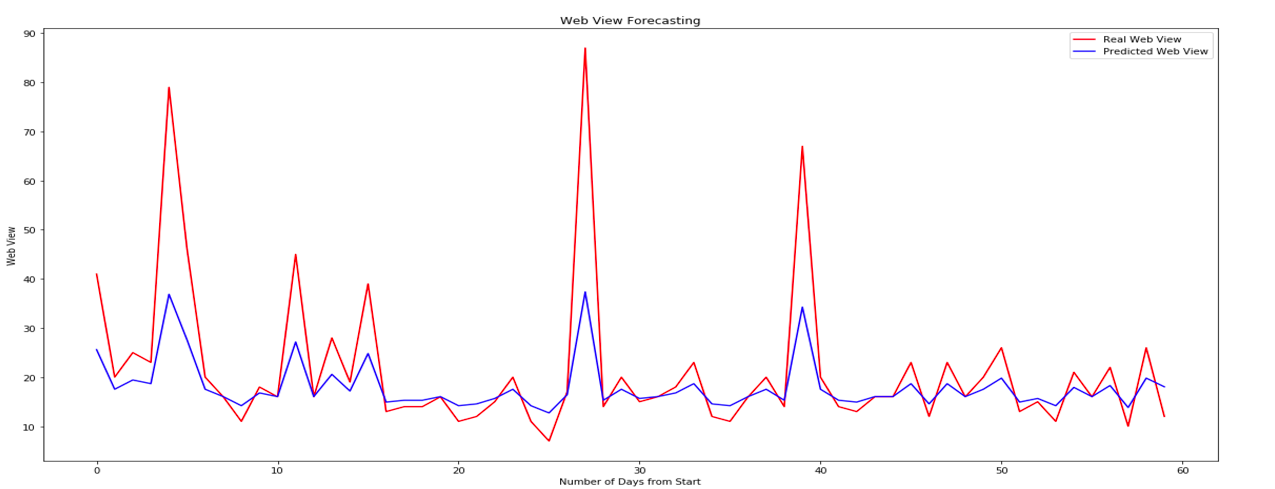
**只是为了读数据方便!!!直接一次索引即可**

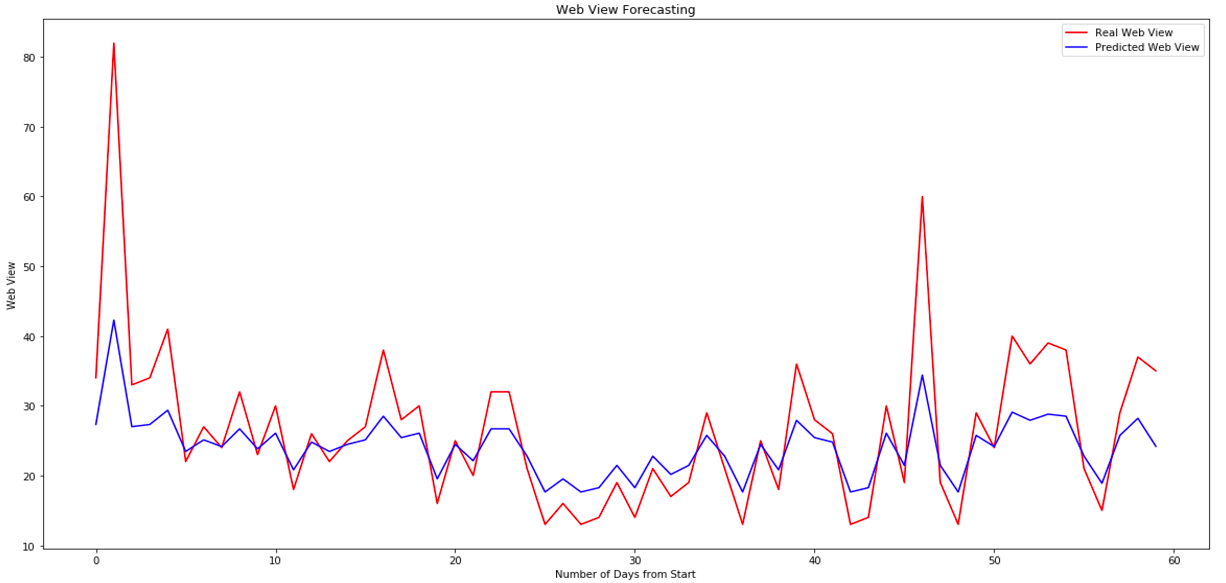
**否则文件读取在划分训练集的时候下标处理麻烦**

**没有使用测试集训练!!!**

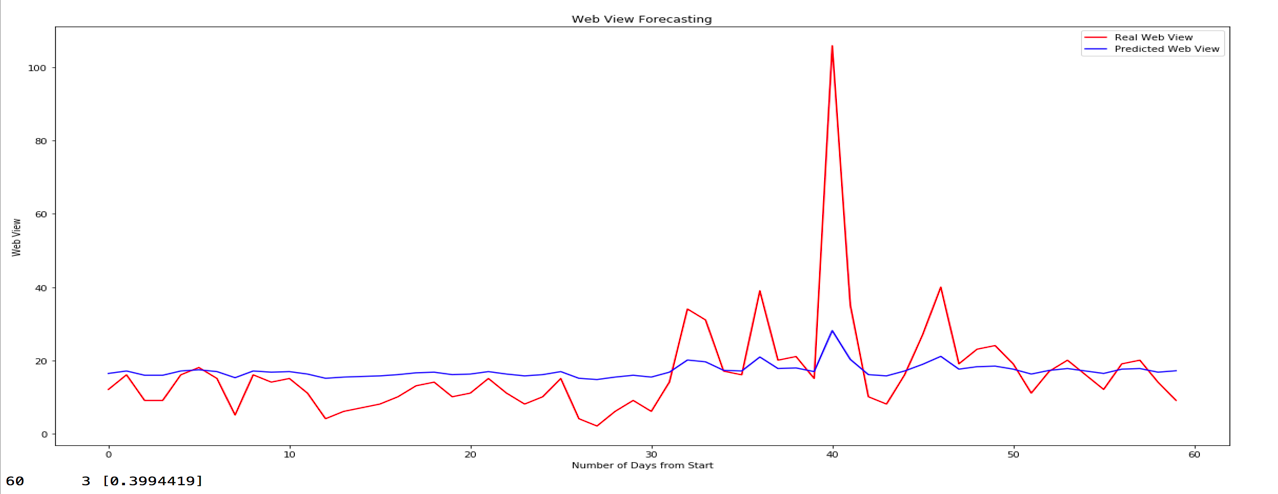
**cross-validation的过程完全在训练集中自动进行**

用LSTM预测，有的预测趋势比较完美，比如第5、8条数据：

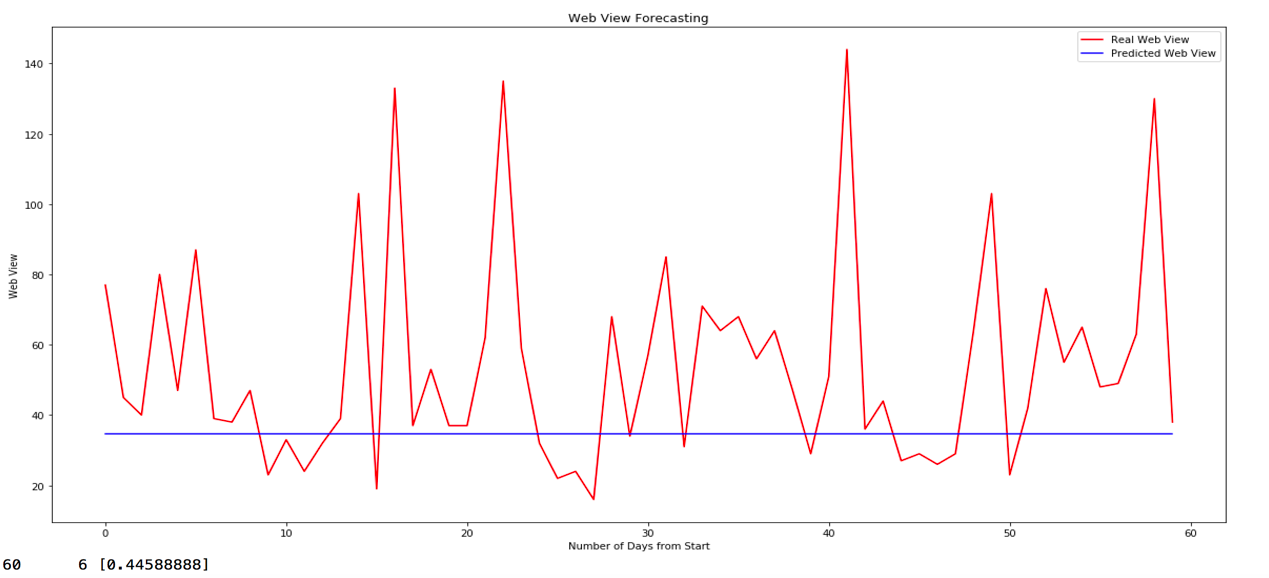




有的预测比较中庸，比如第4条预测



有的也比较一般，比如第7条数据：



这证明我们没有使用测试集过拟合，否则误差可以无限逼近0

另外，由于LSTM是神经网络，训练时是分批梯度递降，每次训练都有可能收敛到局部最优解，整个100次训练标压i7耗时1小时30分钟左右，每次的smape会在0.3944到0.4059之间，故提交的最后最好结果的0.3944。结果在submisson\_LSTM.csv中

**4.4 LSTM总结**

LSTM对这个时间序列的预测比较适合，能较好的把握长期特征

**4.5模型改进**

未来可以加入其他数据的影响，不同数据间的相互影响。