Capstone Project 2 -- Forecast Web Traffic

Xiao Zhang

1. Background and motivation

This project is to forecast future web traffic for approximately 145,000 Wikipedia articles. Originally a kaggle competition at

https://www.kaggle.com/c/web-traffic-time-series-forecasting

Main goals of this project include:

1) explore the time series data and analyze trend, seasonality and stationarity

2) get familiar with state-of-the-art time series forecast methods including ARIMA, Prophet and RNN

This project was built in Google Colaboratory because it could utilize the GPU from google's server and make the computation more efficient.

2. Data exploration

The data packages include three files:

1) Training data of 145063 Wikipedia pages, including page names and visit time series from 2015-7-1 to 2017-9-10
2) Key data of 8993906 pages, matching their page names with IDs
3) A sample submission spreadsheet with 8993906 page IDs and forecasted total visits during 2017-9-13 ~ 2017-11-13

Here the data exploration focuses on the training data with visit time series.

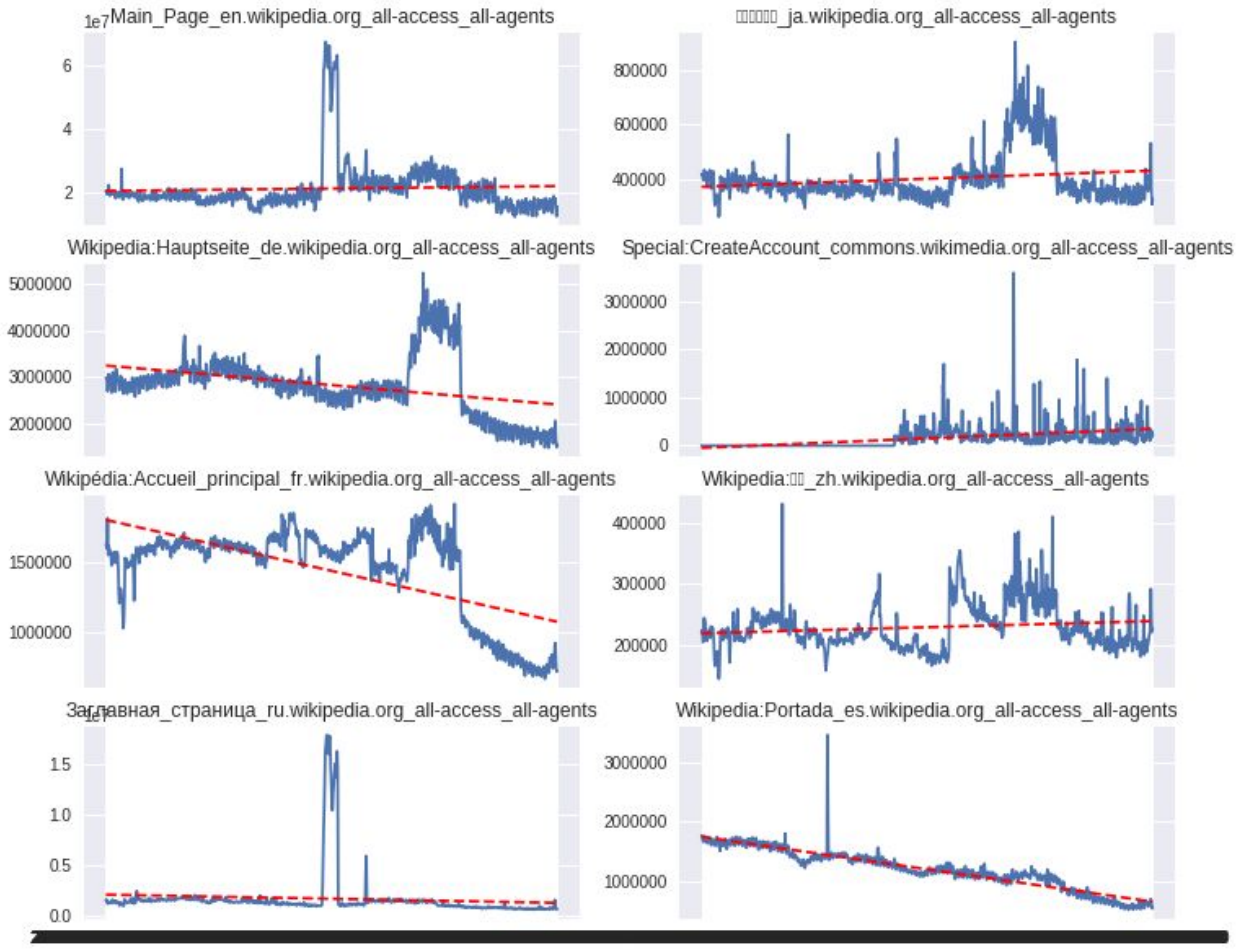
2.1 Whether visits are related to language?

I checked the time series of total visits per language and found

1) English articles in general has more visits than other languages as expected because English is the most widely used language in regions having access to Wikipedia
2) media pages have greater variations compared to other articles
3) English and Russian articles experienced spike in visits during 2016-7 ~ 2016-8 probably due to US election and Olympics
4) Spanish articles had dip around January of each year probably due to celebration of Día de los Reyes Magos.

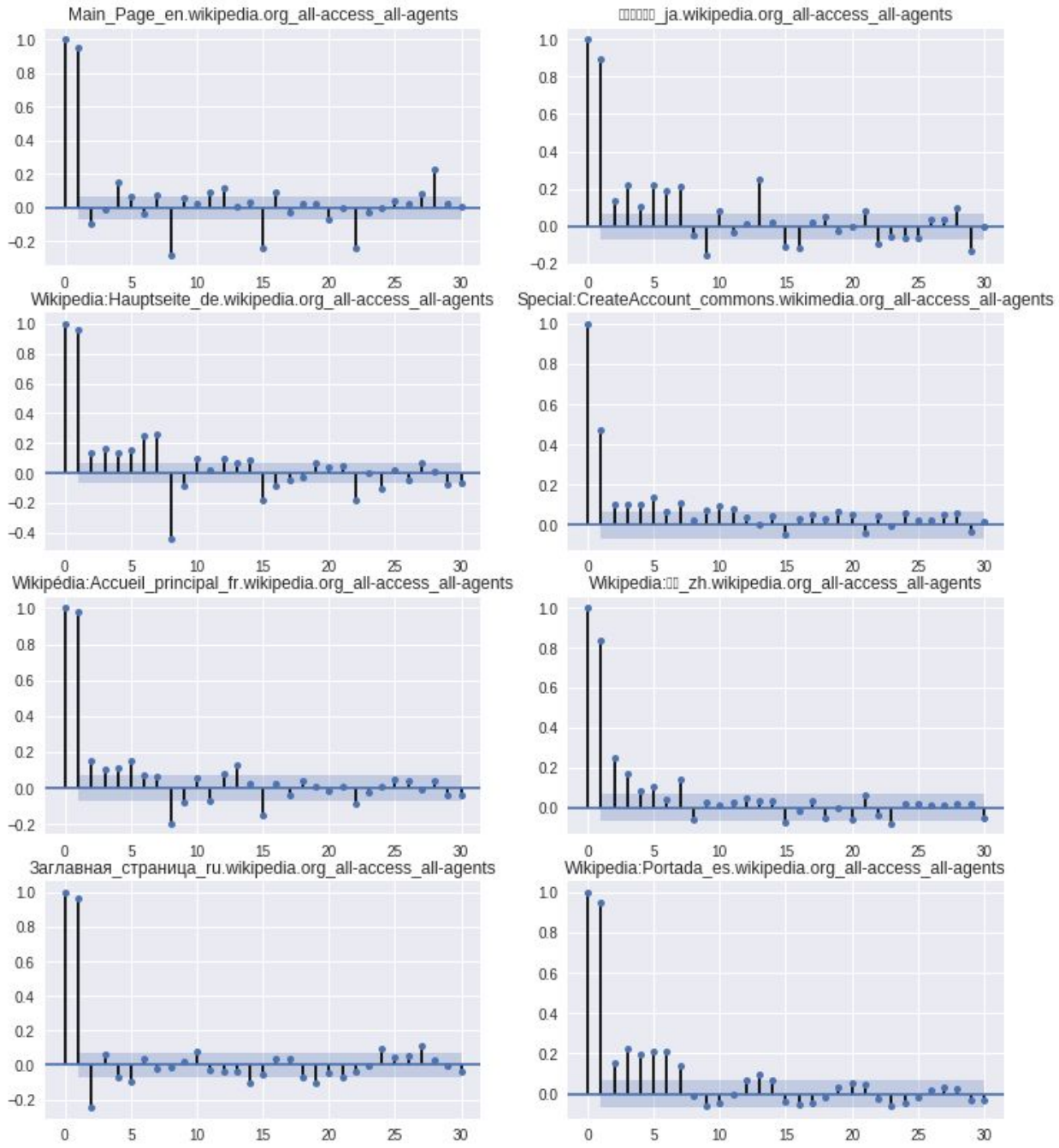
2.2 Trend

Here I checked the trends of the most visited page per language using linear regression and found the most visited German, French and Spanish page show a clear decreasing trend, while other pages do not demonstrate a significant trend.



2.3 Seasonality

Here I used the most visited page per language as an example to explore seasonality. PACF (partial autocorrelation function) and ACF (autocorrelation function) were applied but PACF gave a clearer pattern than ACF because it removes the intermediate correlation. Based on PACF, English based page has a clear weekly seasonality, German and French pages show similar patterns. Visits of most pages are correlated to the visits of the prior day, while Japanese, Spanish and Chinese pages are correlated to the visits of the past week.



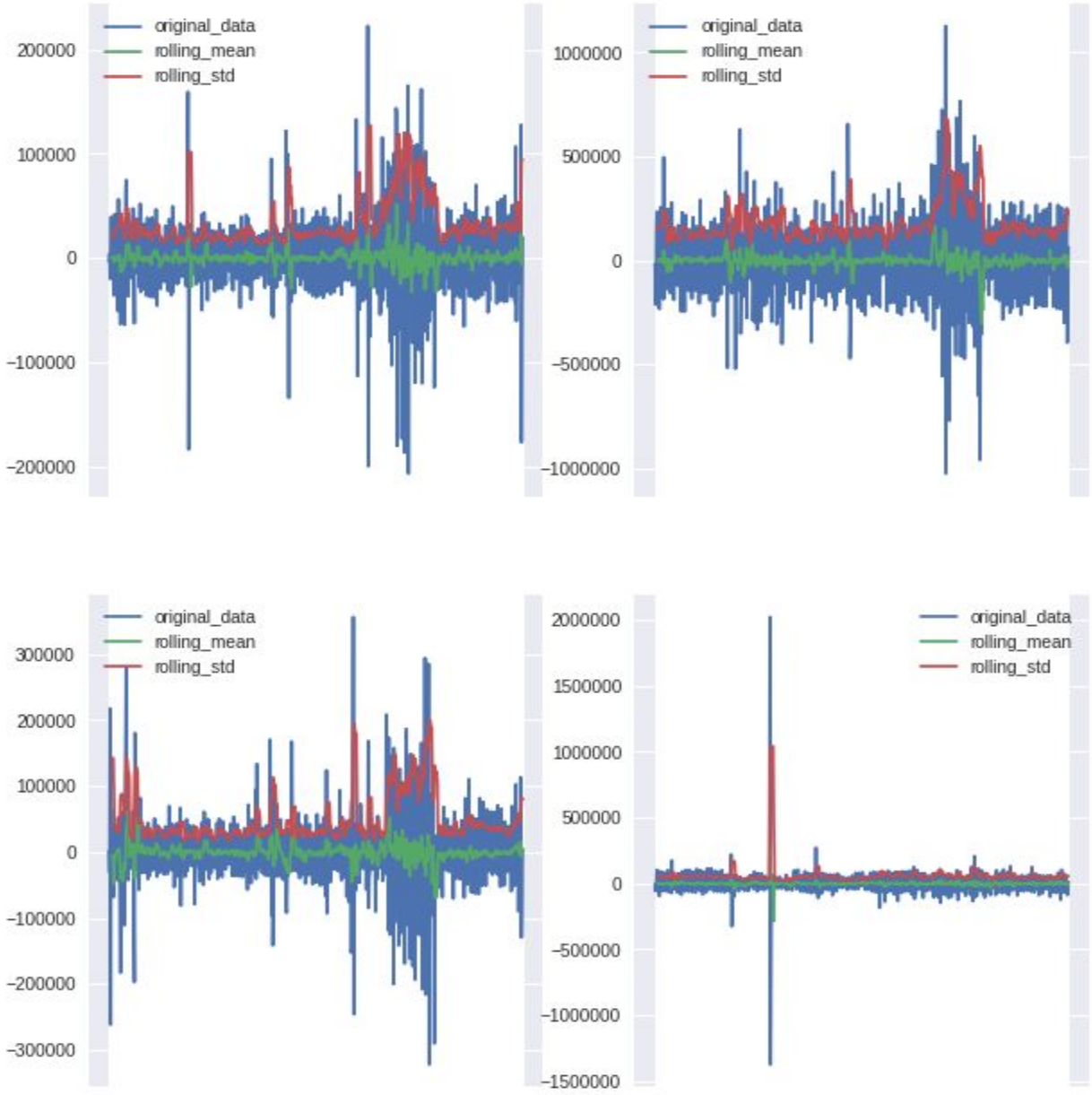
2.4 Stationarity

Time series are stationary if they do not have trend or seasonal effects. Summary statistics calculated on the time series are consistent over time, like the mean or the variance of the observations.

Some models like ARIMA require stationarity to forecast. To detect stationarity, **Augmented Dickey-Fuller test** was used. The null hypothesis of the test is that the time series is not stationary. If failed to be rejected (p-value > a threshold, say 0.05), it suggests the time series

is non-stationary. If rejected when p-value < 0.05, the null hypothesis is rejected and it means the time series is stationary.

Detrending and seasonality removal may be needed to transform the data to stationary. Common approaches include *transformation* (Log, square root, etc), *smoothing* (rolling average, monthly average etc), *differencing* (first order difference, difference with average). Based on the p-value of the adf test, the 1st, 4th, 6th and 7th web article are stationary (p-value < 0.05), while the rest are non-stationary. Here I used first order differencing to stationarize the data. After differencing, all the four non-stationary time series become stationary with *p-value* < 0.05.



3. Modeling

To explore various approaches of forecast, the most visited page per language were used as a case study. Three approaches were tested here: ARIMA, Prophet and LSTM.

3.1 ARIMA

ARIMA = Auto-Regressive Integrated Moving Average. Assumptions. The time-series is stationary. Depends on:

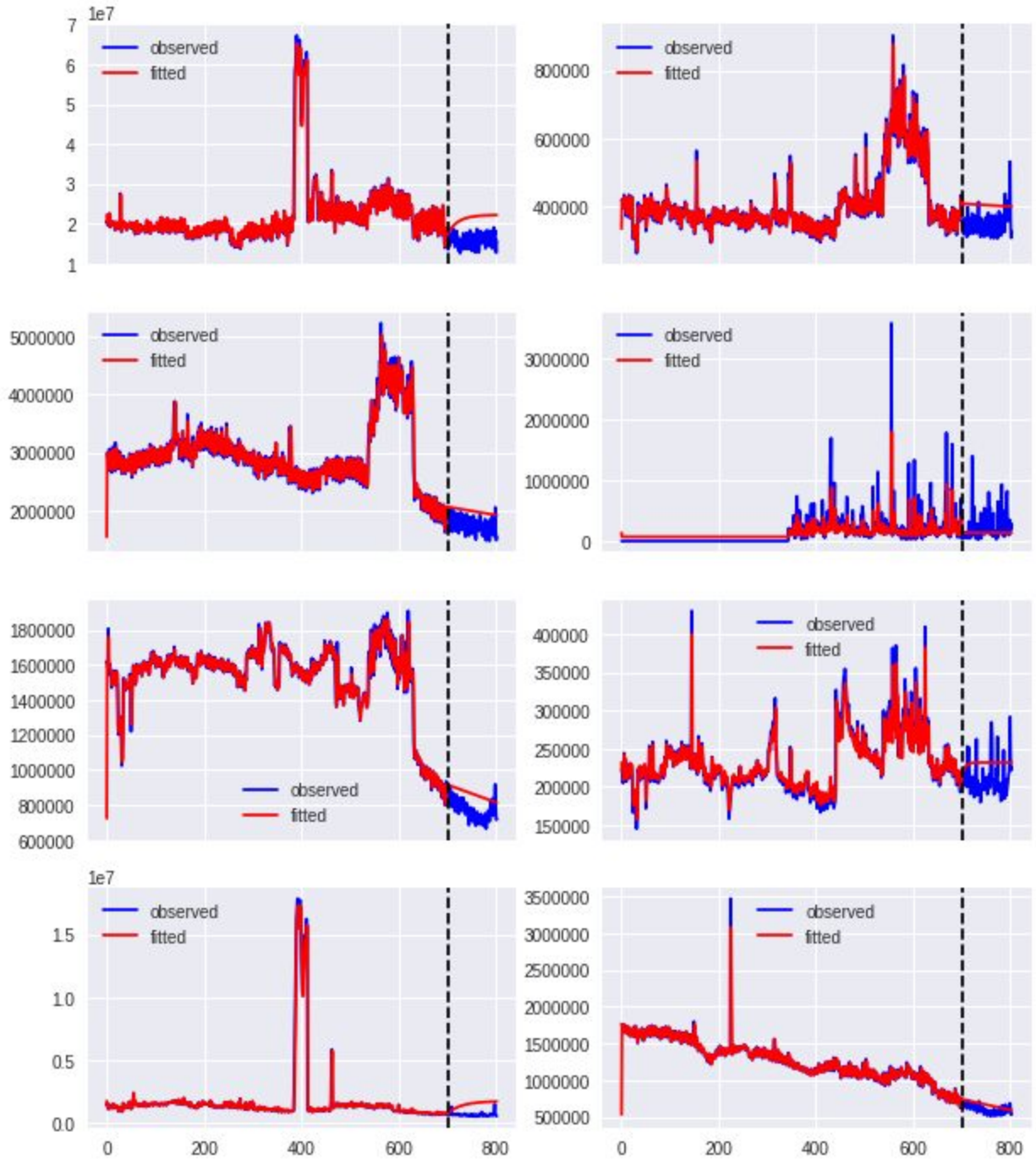
1. Number of AR (Auto-Regressive) terms (p).
2. Number of I (Integrated or Difference) terms (d).
3. Number of MA (Moving Average) terms (q).

p and q can be determined using ACF and PACF plots. Here I chose 1 for *p* and 0 for *q*, *d* as 1 for non-stationary series and 0 for stationary series. The total period of 803 days was divided into training period of 2015-7-1 ~ 2017-5-31 (701 days) and test period from 2017-6-1 ~ 2017-9-10 (102 days) to examine model performance. **SMAPE** (Symmetric mean absolute percentage error) is used as metric to evaluate forecast model performance.

ARIMA model was applied to the most visited pages per language. Use 702 days of data to train and 101 days to test. Order of differencing in ARIMA model was set to 1 for the four non-stationary time series, which is equivalent to first-order differencing which convert them stationary. When d = 1, ARIMA model in python returns the difference/residual. An inversion is needed to add the difference to the real data for comparison.

Mean SMAPE value of the 8 forecasts is 29.0. In general, ARIMA performs well for in-sample fitting but tends to converge to the sample mean for out-of-sample forecast when predicting long forecasting periods. This is because the AR process could not capture seasonality or other main features in the time series.

In summary, *advantages* of ARIMA model: easy to understand, *disadvantages*: parameter tuning, requirement of stationarity, limited performance for out-of-sample prediction.



3.2 Prophet

The Prophet model decomposes time series to three main model components: trend, seasonality, and holidays

(https://www.analyticsvidhya.com/blog/2018/05/generate-accurate-forecasts-facebook-prophet-python-r/). They are combined in the following equation:

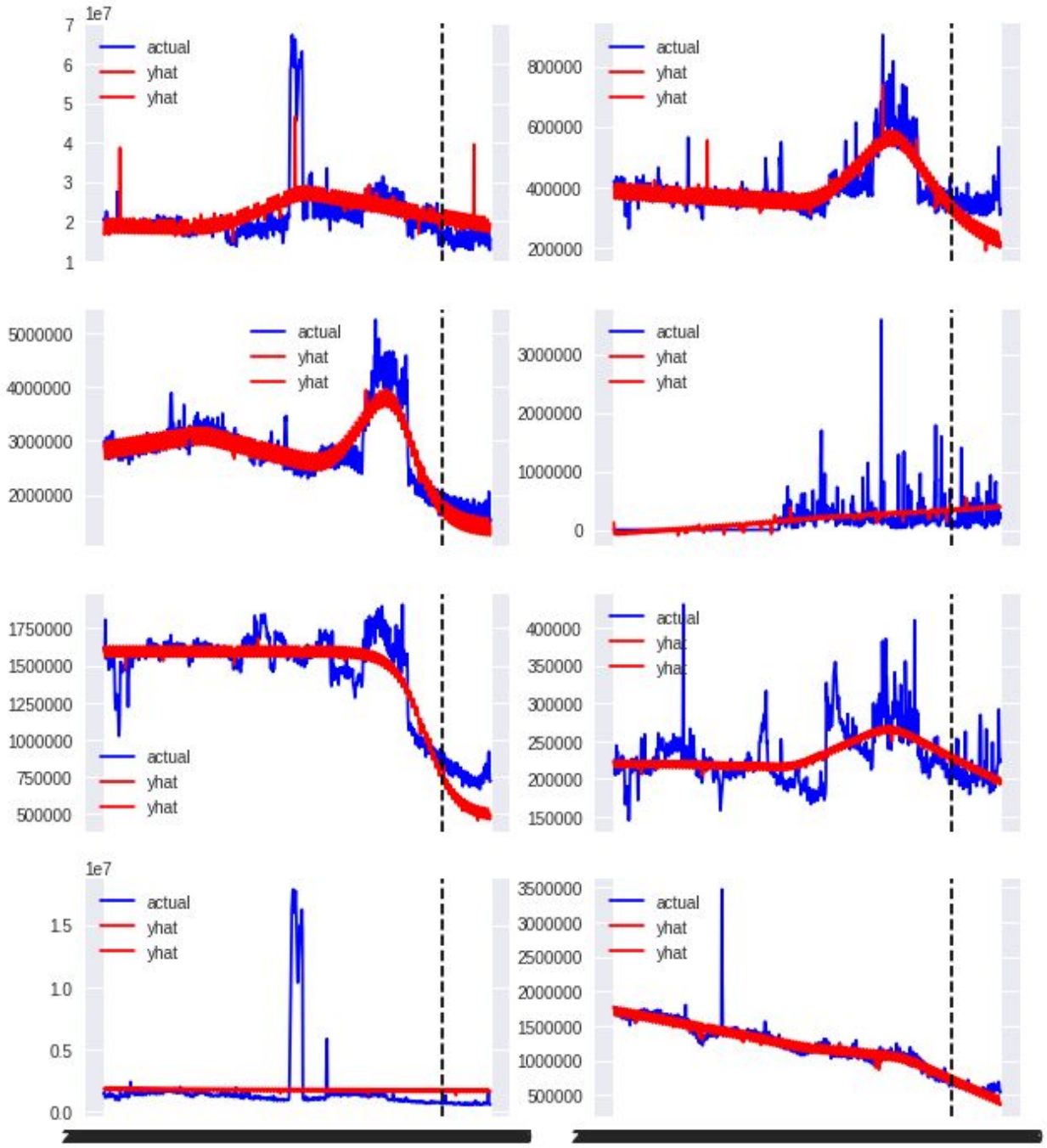
$$y(t) = g(t) + s(t) + h(t) + \epsilon_t$$

- **g(t)**: piecewise linear or logistic growth curve for modelling non-periodic changes in time series
- **s(t)**: periodic changes (e.g. weekly/yearly seasonality)
- **h(t)**: effects of holidays (user provided) with irregular schedules
- **εt**: error term accounts for any unusual changes not accommodated by the model

Prophet does not need to assume stationarity to predict, which is a great advantage over ARIMA. Here the parameters I tuned include:

- growth = 'logistic',
- n_changepoints = 15, #no of change points,default 25
- changepoint_range=0.95, #change points exist in the first 90% of time series,default 0.8
- changepoint_prior_scale = 0.1, #adjust trend flexibility,default 0.05,increase if underfit
- daily_seasonality=True,
- holidays=holiday # language specific holiday is defined

The mean SMAPE of the predicted page visits over the test period is 34.6, higher than the value from ARIMA. By contrast, Prophet does not model the in-sample series as well as ARIMA since it is based on trend and seasonality. If change point position is not specified in parameter setting, it is unlikely to be captured. The out-of-sample predictions are greatly affected by the trend near the end of the training period.



3.3 LSTM

Recurrent neural network **(RNN)** is able to model time or sequence-dependent process.**Long Short-Term Memory Network (LSTM)** is a a type of RNN that handles the vanishing gradient problem. Instead of neurons, LSTM networks have memory blocks that are connected through layers. A block has components that make it smarter than a classical neuron and a memory for recent sequences. A block contains gates that manage the block's state and

output. A block operates upon an input sequence and each gate within a block uses the sigmoid activation units to control whether they are triggered or not, making the change of state and addition of information flowing through the block conditional.

There are three types of gates within a unit:

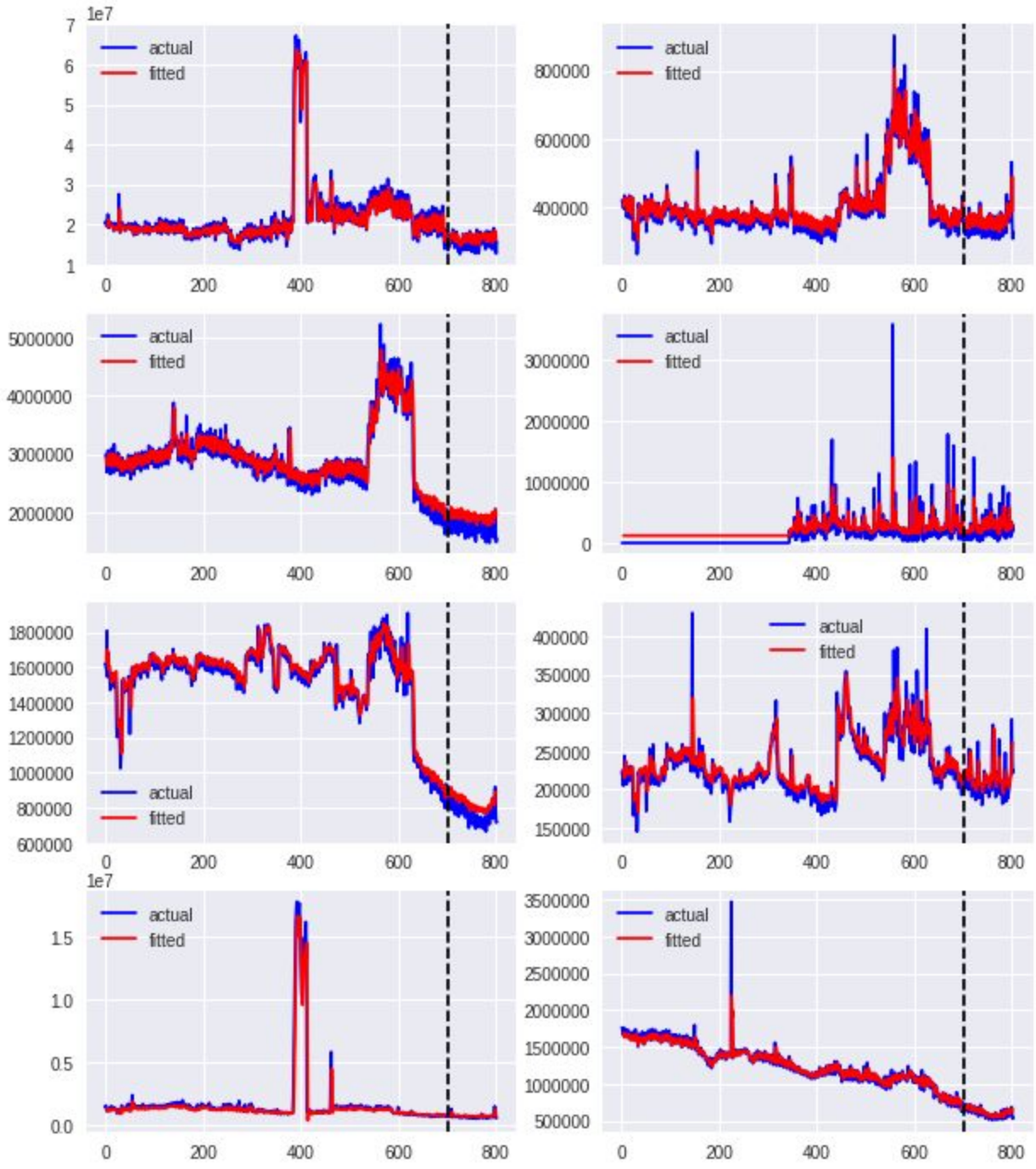
Forget Gate: conditionally decides what information to throw away from the block.

Input Gate: conditionally decides which values from the input to update the memory state.

Output Gate: conditionally decides what to output based on input and the memory of the block.

Here a LSTM network was built. The network has a visible layer with 1 input, a hidden layer with 4 LSTM blocks or neurons, and an output layer that makes a single value prediction. The default sigmoid activation function is used for the LSTM blocks. The network is trained for 10 epochs and a batch size of 1 is used. A window method is used so that multiple recent time steps can be used to make the prediction for the next time step. In this case, with a look back of 3 steps, it used data of t-2, t-1, and t to predict the value at t+1.

The mean SMAPE value of the 8 pages over the test period is 16.1, lower than the SMAPE gotten from ARIMA and Prophet. Therefore, LSTM outperforms ARIMA and Prophet in forecast page visits of the eight articles. Also, LSTM predicts satisfactory values for both in-sample and out-of-sample forecast.



4. Summary

Three models including ARIMA, Prophet and LSTM were explored in this study to examine their performance on predicting web traffic. I summarized their advantages, disadvantages and results into the following table.

	Advantages	Disadvantages	Results
ARIMA	easy to understand	parameter tuning (p, d, q), requirement of stationarity	Great fitting for in-sample prediction, out-of-sample forecast converge to sample mean, **SMAPE** of test period: 29
Prophet	Work for both stationary and non-stationary data, easy to implement	Parameter tuning (seasonality, change points, holiday etc)	Perform ok for in-sample forecast, out-of-sample forecast depend on trend at the end of training period, **SMAPE** of test period: 34.6
LSTM	Great performance in forecast	Difficult to understand, parameter tuning (no of cells, look-back steps)	Great performance for both in-sample and out-of-sample forecast, **SMAPE** of test period: 16.1