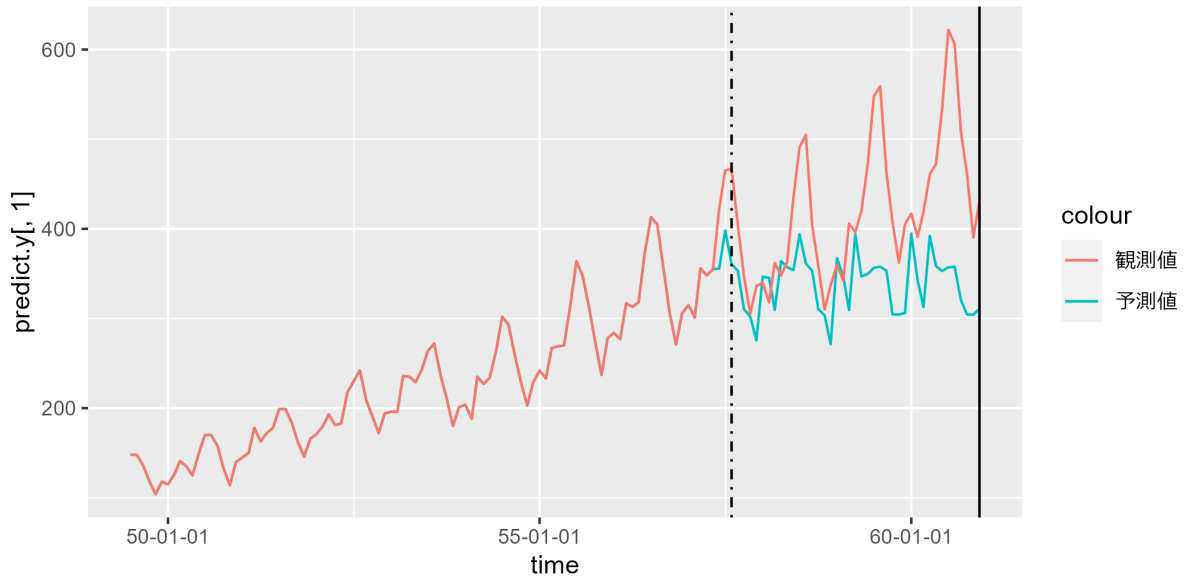
Application of xgboost to time series forecasting by taking advantage of its powerful forecasting performance

abstract

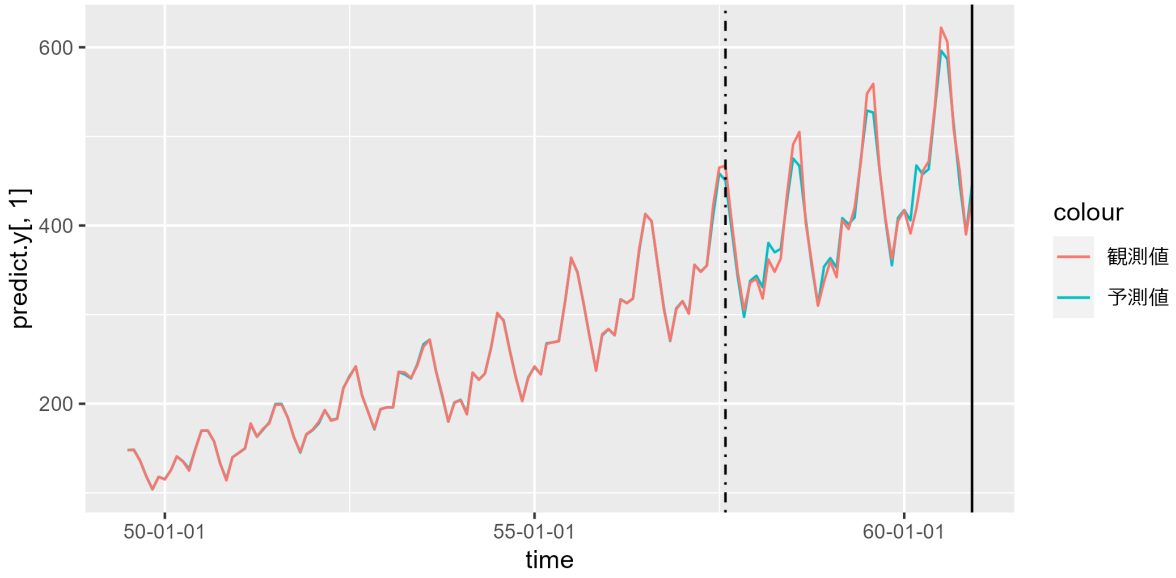
xgboost[1] has the best forecasting performance among **non-deep learning** methods. However, it works well for interpolation problems and regression, but not for future forecasting of time series data that requires extrapolation. I think it is difficult to avoid this tendency even if we add explanatory variables in the background of the data. Possible explanatory variables include lags of a day or several days from the data, months, days, days of the week, holidays, and so on. In fact, the increase or decrease in data values due to these factors is quite possible and can serve as explanatory variables. However, even if you do this, you will not be able to capture the trend.



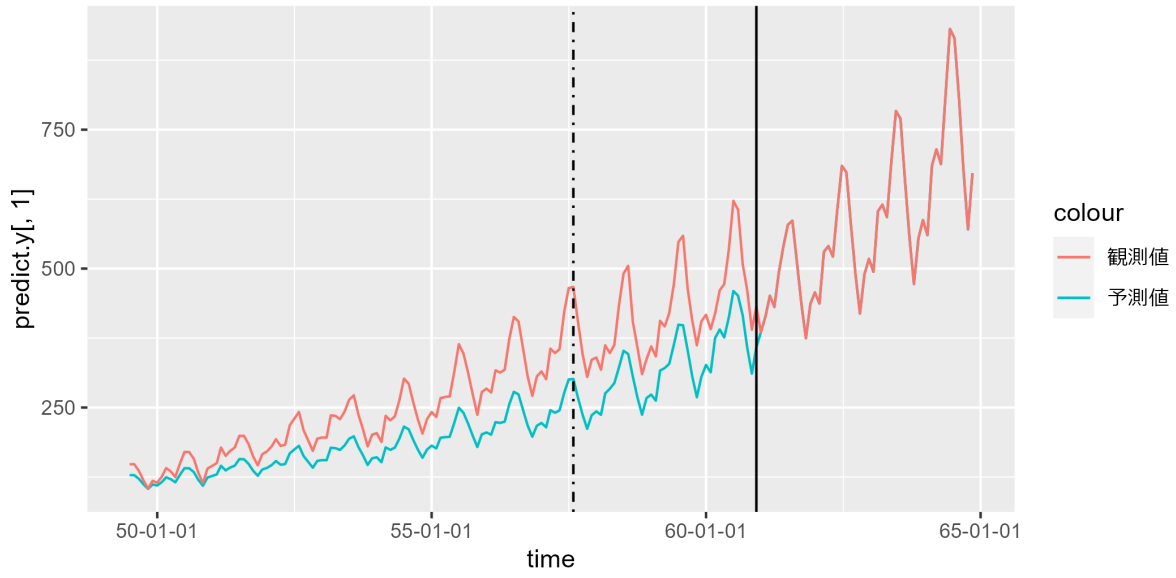
Introduction

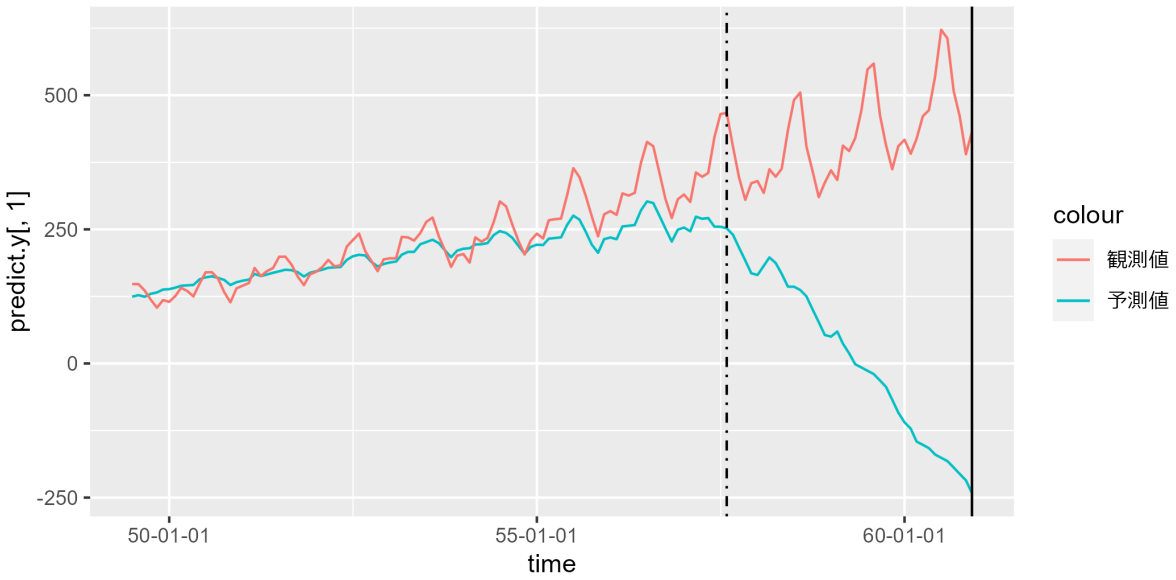
Time series data with a trend may work well in the training interval, but not in the test interval or future. The main reason is simple. The main reason is simple: it has very different basic statistics (e.g., mean) than the trained interval, which is information that is not acquired by training at all.

Normally, when we focus on stationarity, that is, the mean and variance of the data, we want it to show roughly the same trend at all points in time. Therefore, by taking the difference and logarithm of the data, we can make the data stationary, create a prediction model, and then reverse the difference and logarithm transformation of the prediction results.



For data that has been differenced and log-transformed in this way, **xgboost**[1] can be a seemingly good predictive model. However, this is only the case if the fact that the data is stationary by differencing can be observed at any point in time. If you change the situation of the data slightly, the discrepancy with the measured values will be so large that it will not be a usable prediction model.

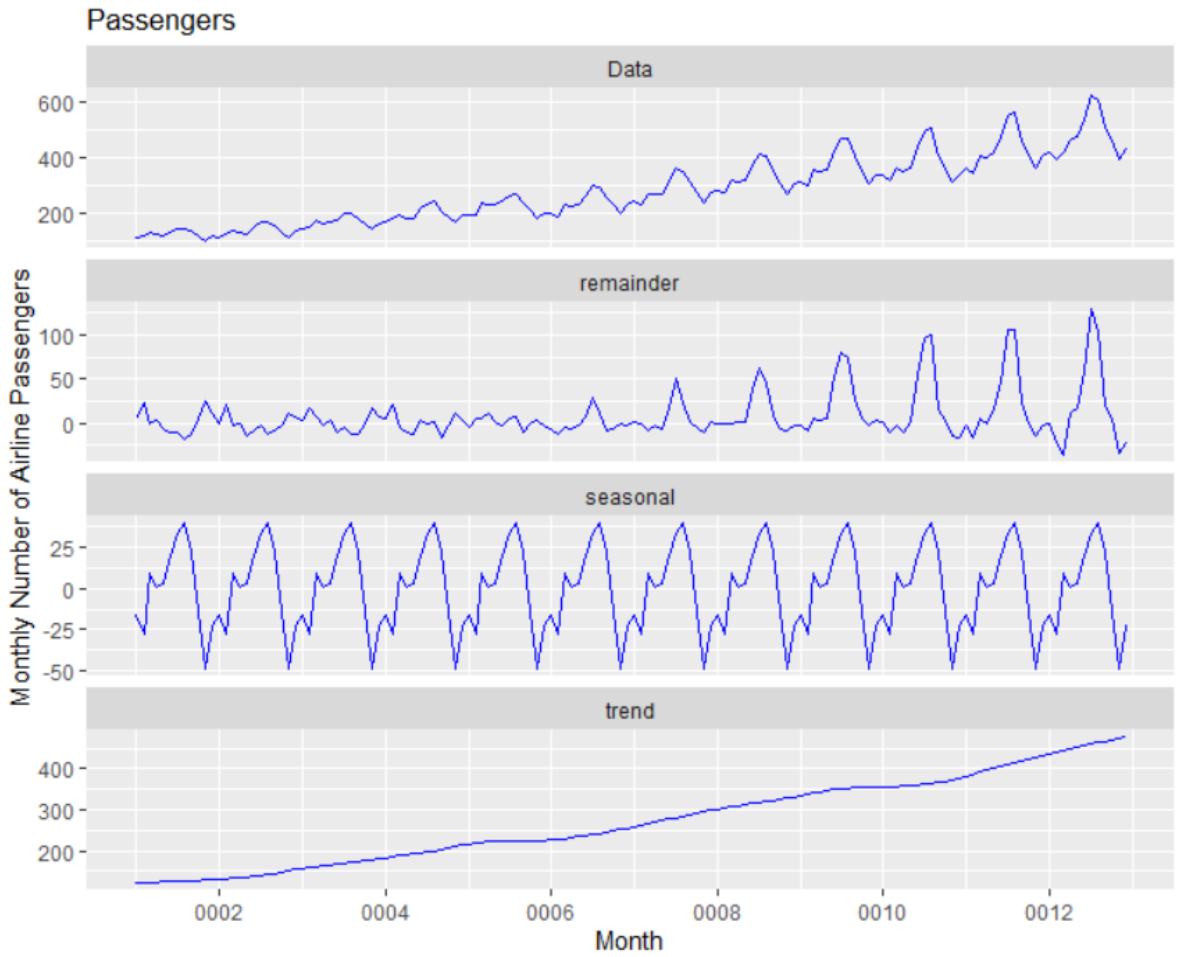




Overview of the proposed method

xgboost[1] can be given up as an option. For example, porohet[2] produces very good results, but lacks explanatory power. On the other hand, **xgboost**[1] has explanatory power and has been proven to have the best prediction performance among **non-deep learning** models, so it would be a shame to discard it. So the idea is to use the best parts of **xgboost**[1] in combination with other models as a compromise.

Fortunately, time series data can be used to isolate trends.



The data can be decomposed into **trend + seasonal + remainder**. If we let **xgboost**[1]take care of the part without **trend**, and model **trend** with **ARIMA**[3], we can build a good model. However, depending on the data, **xgboost**[1] may have weaknesses for repeated cycles.

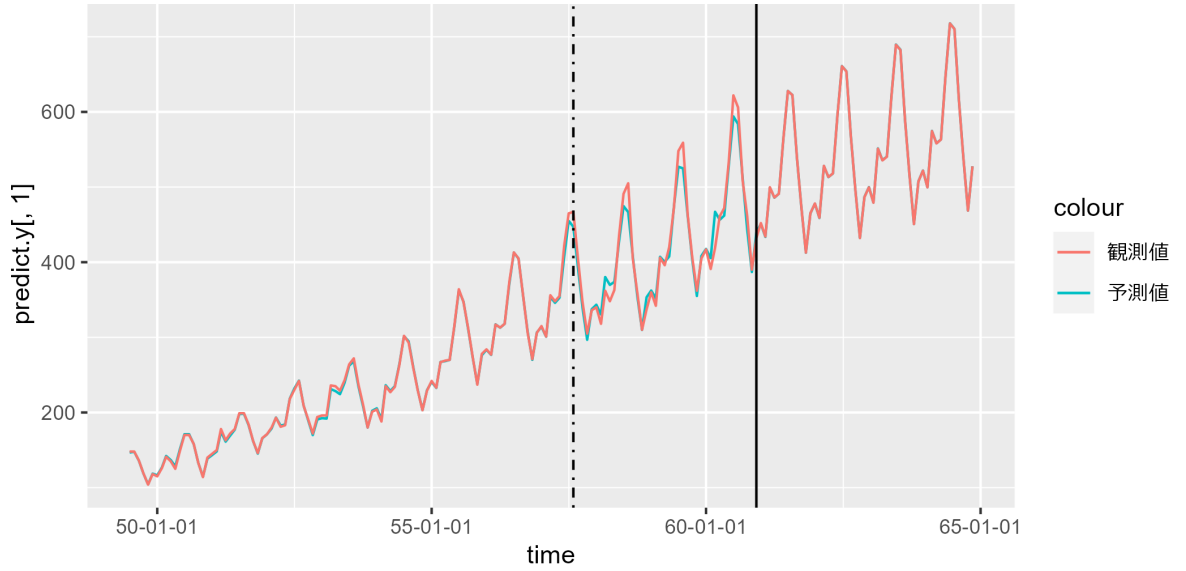
Therefore, I think it is possible to deal with this problem by adding the sin and cos terms of the Fourier expansion of the periodic component to the explanatory variables.

$$y_t = a + \sum_{k=1}^{K} \left[\alpha_k \sin(2\pi kt/m) + \beta_k \cos(2\pi kt/m)\right]$$

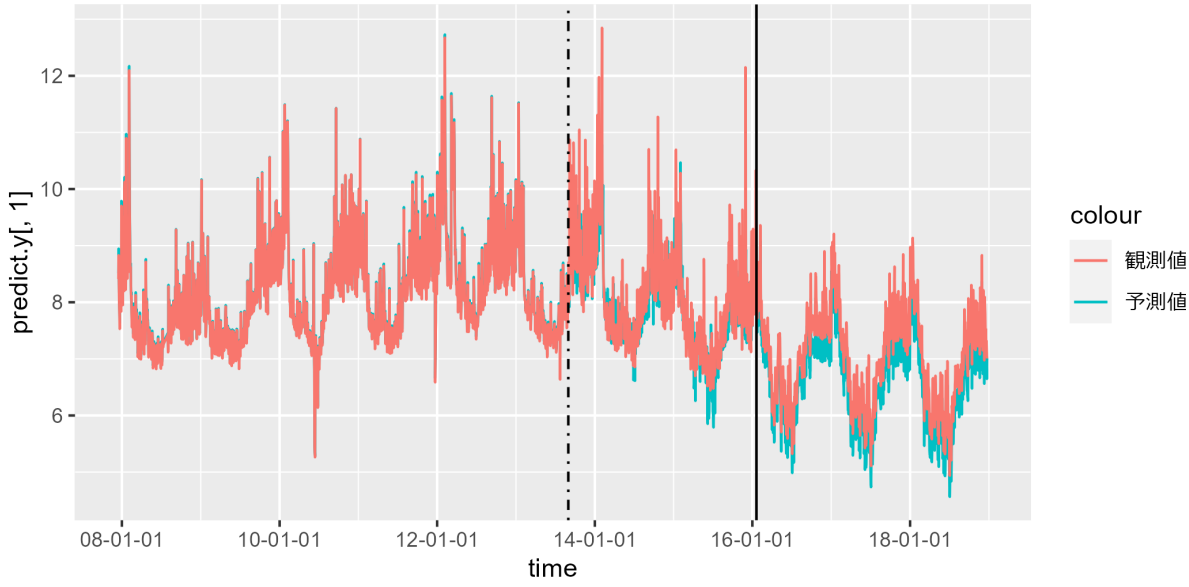
However, it is necessary to limit it to the day before the sin and cos terms are available. The reason is that the sin and cos terms on the day you want to predict are essentially unknown values. Therefore, they will be missing as explanatory variables on the day of the event, and we will have to fill them in by predicting. So, my idea is to use the previous day's sin and cos terms as they are to make a provisional forecast. This part of the explanatory variables can be refreshed

each time the prediction is made, since the sin and cos terms can be re-calculated as the prediction progresses, thus eliminating the problem of having only the copied sin and cos terms of the previous day. However, it is possible to reinforce this problem. As a result of decomposing the time series data, we get seasonal, which is generally a simple repetition, and by using this simple repetition data, we can modify the explanatory variables to be more reasonable.

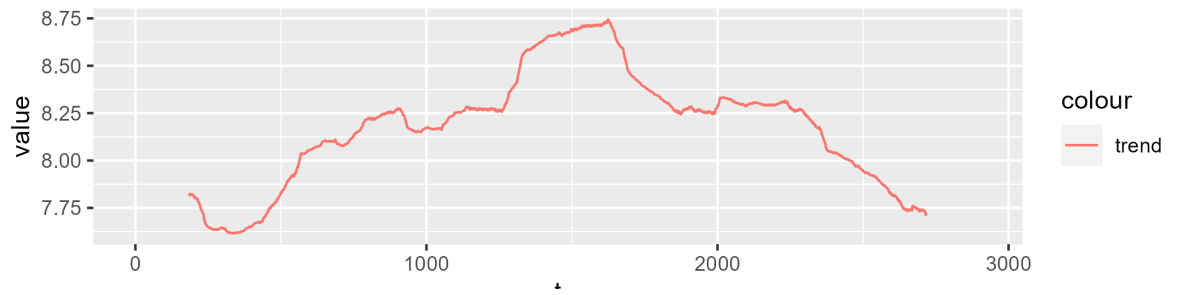
Experiment



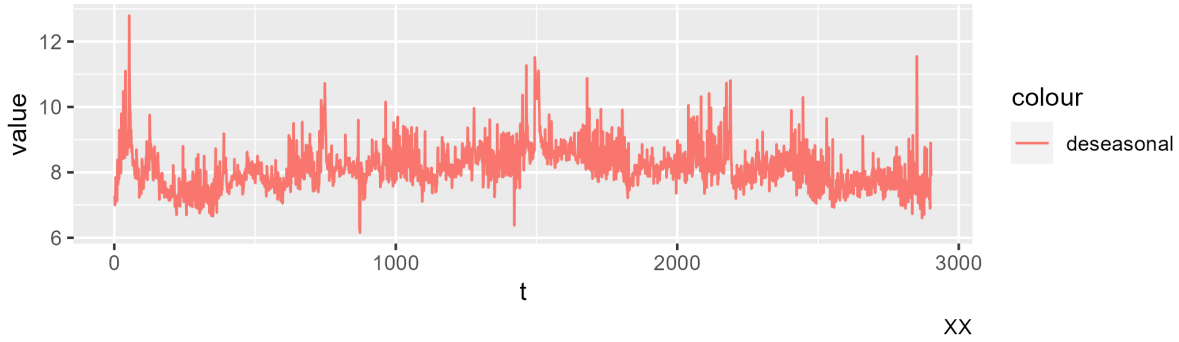
It seems that the prediction is reasonable even outside the data, i.e., in the extended future. Next, let's experiment with example_wp_log_peyton_manning.csv, which shows the results of extended prediction for 3 years (1096 steps).



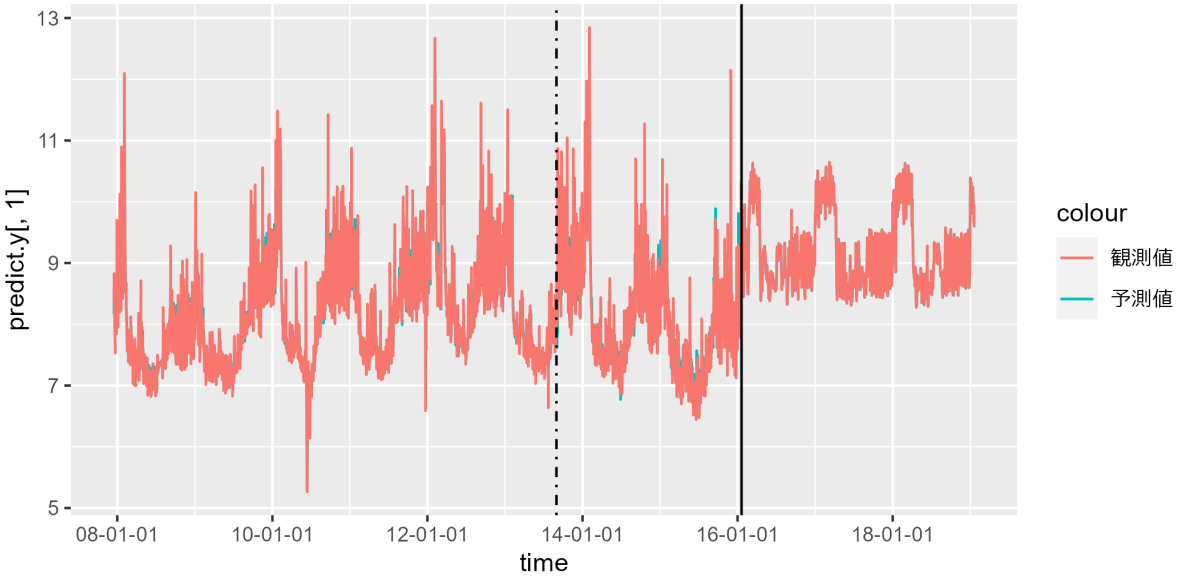
The trend component of this data is



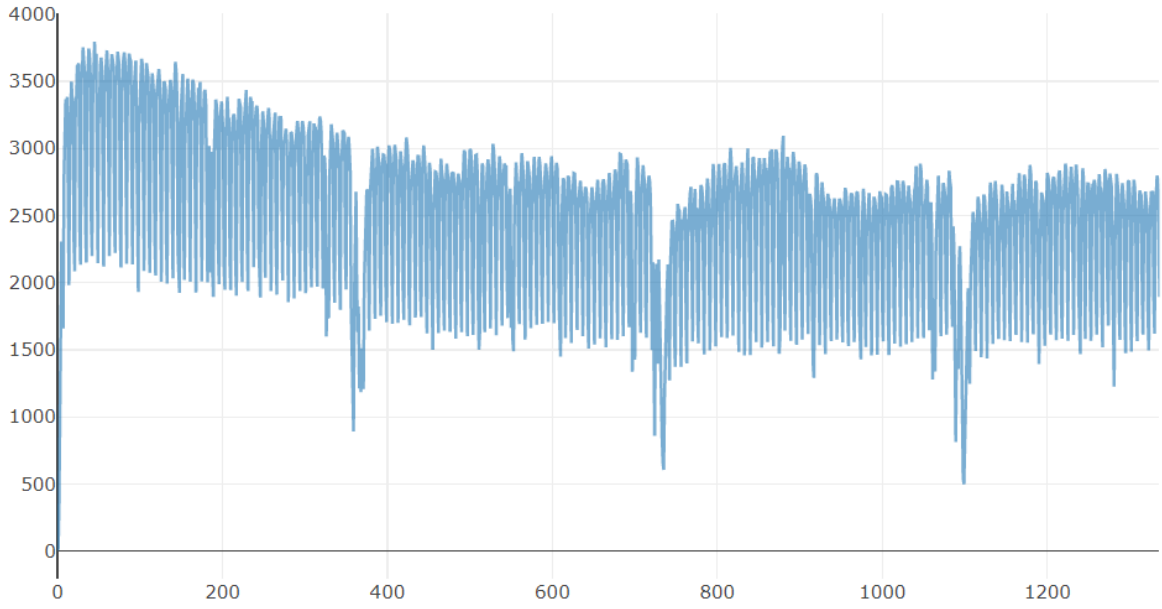
The downward trend shows that the forecast model follows the trend. The data for the component contributed by **xgtboost**[1] is shown in the figure below, which shows that the forecasting model fully utilizes the superior capabilities of **xgboost**[1].



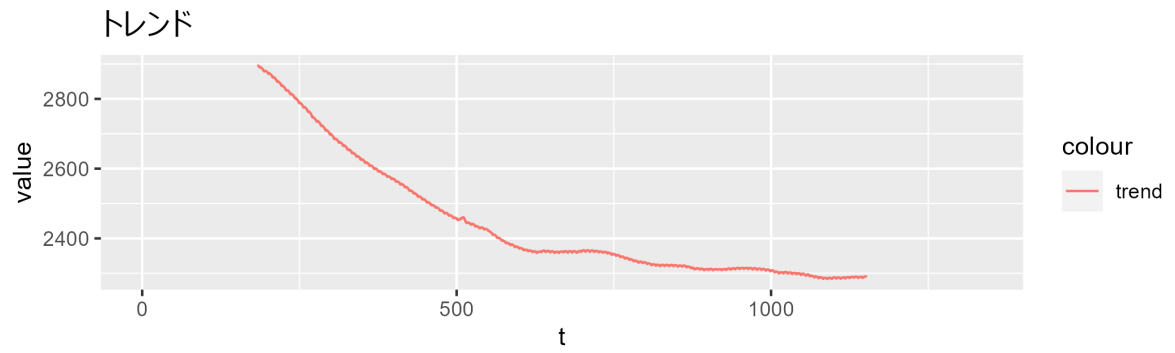
By the way, if we model it using only **xgboost**[1], we get the following.



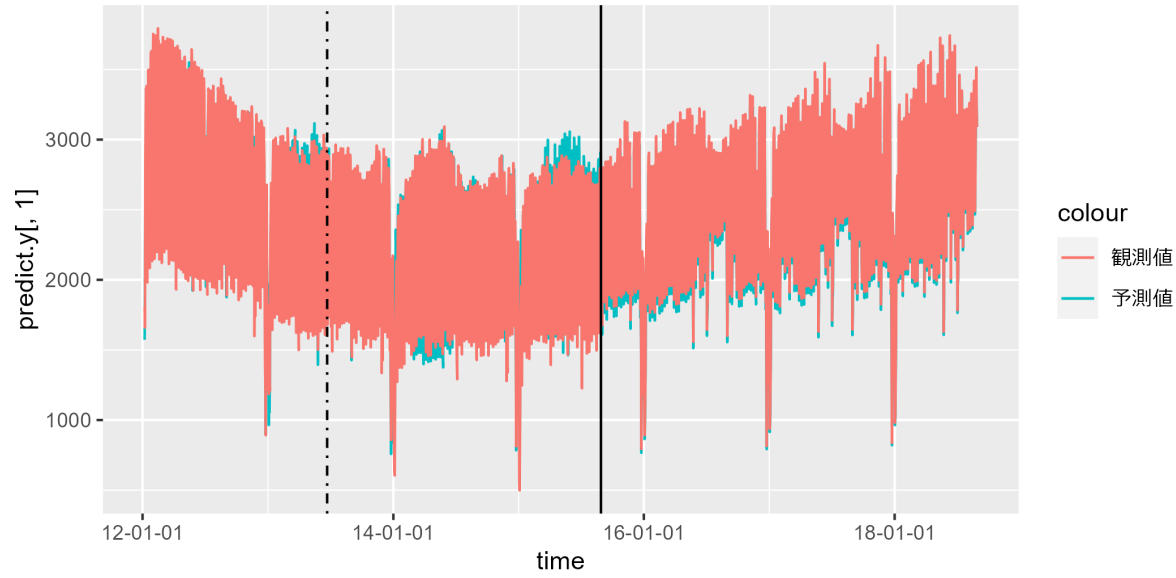
This suggests that the idea is producing reasonable results. Other data validation includes



his data appears to be right-side-up. In fact



The trend component is falling to the right. I tried to train this data by constraining the training interval to 40% of the total. The test interval is a very good predictor, but the extension to 3 years (1096 steps) shows a slight upward trend, but the characteristic increase and decrease seems to be a very good predictor. However, the characteristic increase/decrease looks like a very good prediction.



Discussion

In the experiment, I used **ARIMA**[3] in combination, but it is also possible to use **porphet**[2] for this part. The problem is that if you have data with very long periods, you will need data for multiple periods. The problem is that for data with very long periods, we need data until several of those periods appear, which increases the amount of training data and makes the computational cost very high for non-deep learning power.

References

[1]CHEN, Tianqi; GUESTRIN, Carlos. Xgboost: A scalable tree boosting system. In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 2016. p. 785-794.

[2]KHAYYAT, Mashael, et al. Time Series Facebook Prophet Model and Python for COVID-19 Outbreak Prediction. *CMC-COMPUTERS MATERIALS & CONTINUA*, 2021, 67.3: 3781-3793.

[3]Hyndman, RJ and Khandakar, Y (2008) "Automatic time series forecasting: The forecast package for R", *Journal of Statistical Software*, **26**(3).

Wang, X, Smith, KA, Hyndman, RJ (2006) "Characteristic-based clustering for time series data", *Data Mining and Knowledge Discovery*, **13**(3), 335-364.

PICCOLO, Domenico. A distance measure for classifying ARIMA models. *Journal of time series analysis*, 1990, 11.2: 153-164.

Brockwell, P. J. and Davis, R. A. (1996). *Introduction to Time Series and Forecasting*. Springer, New York. Sections 3.3 and 8.3.

Durbin, J. and Koopman, S. J. (2001). *Time Series Analysis by State Space Methods*. Oxford University Press.

Gardner, G, Harvey, A. C. and Phillips, G. D. A. (1980). Algorithm AS 154: An algorithm for exact maximum likelihood estimation of autoregressive-moving average models by means of Kalman filtering. *Applied Statistics*, **29**, 311--322. 10.2307/2346910.

Harvey, A. C. (1993). *Time Series Models*. 2nd Edition. Harvester Wheatsheaf. Sections 3.3 and 4.4.

Jones, R. H. (1980). Maximum likelihood fitting of ARMA models to time series with missing observations. *Technometrics*, **22**, 389--395. 10.2307/1268324.

Ripley, B. D. (2002) Time series in R 1.5.0. *R News*, **2/2**, 2--7. https://www.r-project.org/doc/Rnews/Rnews_2002-2.pdf

---

```
@inproceedings{y2021timeseriesxgboost,
  title={time series xgboost: time series xgboost},
  author={tatsuhiko.yamato},
  year={2021}
}
```