AtsPy: Automated Time Series Forecasting in Python

Derek Snow[1],✉

[1]Alan Turing Institute, British Library, 96 Euston Rd, London NW1 2DB, United Kingdom

This short report deals with the recent rise of programmatic time series methods. This decade has witnessed the proliferation of commercial and open source time-series tooling, which calls for an exposition of what is publicly available. In tandem with this survey, AtsPy, an open source automated time series framework is developed as a working prototype to showcase the ability of state of the art univariate time series methods.

Automated | Time Series | Machine Learning | AtsPy | Python | Software
Correspondence: ***dsnow@turing.ac.uk***

Introduction

A Python-centric view on the recent growth of time series tools shows the development of the Prophet by Facebook, the GluonTS toolkit by Amazon, and the ForecastTCN algorithm by Microsoft. Among others, these tools have put forecasting methods in the hands of the everyday user. We have also seen contributions from academia and independent developers with algorithms and packages like N-Beats, Auto-Arima, and TBATS.[1] The recent surge in automated time series methods is the direct result of new algorithms (TBATS), procedures (Prophet), and frameworks (GluonTS). In the following section we will seek to understand how these methods have led to the automation of time series forecasting and also discuss how existing method can be used to automate predictions.

Comparison

TBATS. It is often true that your daily data might exhibit weekly and annual patterns. This is true for many tasks like call center volume, electricity usage, or deposit withdrawal requests. TBATS is a method that accounts for these multiple seasonalities (1). It uses a combination of Fourier terms, exponential smoothing models, and Box-Cox transformations in a completely automated manner. TBATS also allows for the seasonal patterns to dynamically adjust over time.

Prophet. Has the benefit of identifying non-linear trends through saturated logistic growth models as well as shifts in these trends, whereas before this would have largely been a manual effort. Prophet also takes care of seasonal effect from annual to daily granularity; that includes holiday effects. And as a last innovation, it is robust to missing data. Both of which would have historically required some sweat of the brow. In summary, Prophet is an automated curve fitting exercise using several linear and non-linear functions of time (2).

GluonTS. Allows you to quickly experiment with different models for univariate and multivariate forecasting (3). It has a rich set of pre-built models that can be used. They particularly emphasise deep learning models. You can use their abstractions and building blocks to create your own custom time series models. A framework like this has much promise, as of now, a lot is left to be improved.

AtsPy. Can be seen as a univariate instantiation of GluonTS with an emphasis on model diversity. AtsPy is built on top of Auto-Arima, TBATS, Prophet, and GluonTS. It is an extremely fast method to test which model best fits your data (4). AtsPy's final innovation is an ensemble time series protocol developed with the LightGBM flavour Gradient Boosting Machine and extracted time series features.

Package

Implements all your favourite automated time series models in a unified manner by simply running AutomatedModel(df). This package can be downloaded using *pip install atspy*. The six main commands will be summarised in this section.

```
from atspy import AutomatedModel
models = ["HWAMS","TBATS1","Prophet"]
am = AutomatedModel(df,models,length)
```

The first parameter, df, houses the univariate dataset, the second parameter; models, lists a selection of the 12 models available to users; length is the out-of-sample prediction length. The current selection of exposed models are ARIMA - automated ARIMA modelling; Prophet - modeling multiple seasonalities with linear or non-linear growth; HWAAS - exponential smoothing with additive trend and additive seasonality; HWAMS - exponential smoothing with additive trend and multiplicative seasonality; NBEATS - neural basis expansion analysis (now fixed at 20 Epochs); Gluonts - RNN-based model (now fixed at 20 Epochs); TATS - seasonal and trend without Box-Cox transformation; TBAT - trend and Box-Cox transformation; TBATS1 - trend, seasonal and Box-Cox transformation;

```
fore_ins, perf = am.forecast_insample()
```

A percentage of the original data is used to make an in-sample forecast of which the performance can be captured.

[1]The majority of these have been implemented in AtsPy which is hosted on GitHub. https://github.com/firmai/atspy

The reason for this in-sample forecast is to establish the performance of the different models so as to build the final ensemble model.

```
fore_out = am.forecast_outsample()
```

It is also possible to use all the data to forecast out-of-sample. By definition we do not have data to test the out-of-sample performance so we simply return a data frame of predictions.

```
i,o,p = am.ensemble(fore_in, fore_out)
```

We can now make use of the in-sample and out-of-sample forecast dataframes to develop an ensemble model that would attempt to beat individual model performance. Generally speaking, the more models initially included in the forecast problem, the better the performance of the ensemble.

```
am.models_dict_in,  am.models_dict_out
```

At the end we can call on the in-sample and out-of-sample models that we have developed and save these models for them to be used in a production environment.

Performance

In various domains, AtsPy can reduce the structural model errors of existing singular models with 30%-50% by using a Gradient Booosting Model (GBM) with time-series extracted features. AtsPy automatically identifies the seasonalities in data using singular spectrum analysis, periodograms, and peak analysis. It identifies and makes accessible the best model for your time series problem using in-sample validation methods and combines the predictions of all these models in a simple (average) and complex (GBM) ensembles for improved performance.

Future Development

The next iteration of AtsPy should include in-sample validation steps to stop deep learning models from over and underfitting. Extra performance metrics like MAPE and MAE should be added. Work should be done to improve methods that select training and calibration window lengths. The framework should be made robust to missing and dirty data e.g., interpolation, imputation methods. Further attention should be given to develop functions to resample data to a larger frequency.

The automated framework should also have the ability to algorithmically select an optimal subset of a large dataset to balance performance and time to train. More internal model optimisation methods can be included using AIC, BIC, and AICC. Further work should be done to force seasonal stability between in and out-of-sample training models. At a package level, more work should be done to write quality annotations for other developers to follow and improve on the work being done. And lastly there should be a particular focus on making AtsPy less dependency heavy; currently it draws on tensorflow, pytorch, and mxnet for neural network developement.

Conclusion

Automated time series prediction is a relatively new area, and the expectation is that it will only grow from here. Almost every business need to make forecast in order to effectively allocate resources. AtsPy can help users to significantly reduce their time to development.

Bibliography

1. SJ Taylor and B Letham. Prophet: forecasting at scale. facebook research. 2018.
2. Alysha M De Livera, Rob J Hyndman, and Ralph D Snyder. Forecasting time series with complex seasonal patterns using exponential smoothing. *Journal of the American statistical association*, 106(496):1513–1527, 2011.
3. Alexander Alexandrov, Konstantinos Benidis, Michael Bohlke-Schneider, Valentin Flunkert, Jan Gasthaus, Tim Januschowski, Danielle C Maddix, Syama Rangapuram, David Salinas, Jasper Schulz, et al. Gluonts: Probabilistic time series models in python. *arXiv preprint arXiv:1906.05264*, 2019.
4. Derek Snow. AtsPy: Automated time series models in python (1.15)., 2020.