Time series analysis in production

Many businesses are familiar with time series analysis. Often, it's an exercise undertaken regularly as part of operational planning – for example, conducting quarterly sales forecasts to fine-tune sales targets. In other cases, it's an exploratory exercise that informs a specific business decision – for example, forecasting customer growth to gauge upcoming customer service hiring needs.

Both scenarios are examples of manual, pre-planned analysis. These applications of time series analysis usually require advance planning and dedicated time from a data scientist. There's often significant effort required to collect and cleanse data; tune the model; and finally generate and present the results. Ultimately, this makes the analysis less repeatable and less timely, since decision makers can't get forecasts on demand.

How do we go beyond one-off analyses and instead deliver time series analysis as a production-level automated service?

Building a data pipeline

The first step is getting the right data. Not only must it capture the variables of interest, but it should also arrive on time and in a consistent format. Timely, high-quality data is the foundation of a relevant and accurate model. Implementing a data pipeline builds this foundation and allows on-demand forecasts by guaranteeing that inputs are available. Services such as AWS Glue and Azure Data Factory, as well as open source systems such as Apache Airflow or Luigi, are common choices for building a data pipeline.

Generating forecasts in the future and on demand

The nature of time series data poses unique challenges to implementing models in production. First, time series are not typically stationary. Trends in the data can be handled, but are not likely to remain constant in the long term. Cycles may expand or contract as the underlying process changes. **Forecasts generated from a model fitted today may not be accurate in the future**.

Second, time series models are **recursive**: the forecast for any point in time is dependent on prior forecast errors[1]. Due to potential changes to cycles or trends in the data, the model should always be refitted when examining a new time period – even if it's only the addition of a single day to the dataset.

If forecasting frequently – for example, every day – is required, labor-intensive model evaluation and tuning may be costly. At the same time, tuning and refitting is a necessary part of the process due to the challenges described above. With modern processing power and memory, manual modeling can be replaced by a grid search. Using this method, we comprehensively evaluate and select model parameters on the fly[2], choosing the model that performs the best using a predefined metric.

Evaluating and refining models

In order to automate the process while ensuring a high quality model, choose a metric that can be optimized for model selection. Amongst the myriad of metrics to pick for building and training models, described below are a couple of the more common ones. There is no particularly best metric and each one has strengths and drawbacks. A few common metrics are described below, each with its strengths and drawbacks:

- **AIC** penalizes a model with too many parameters. The smaller the AIC, the better fit the model will be to the data. This criterion tends to select a model that slightly overfits the data[4].
- **BIC** also penalizes a model with too many parameters, but with less impunity. This can lead to model selection that can slightly underfit the data[4].
- **HQC** – is similar to BIC in picking models. However, it can be prone to overfitting the data when sample sizes are not large[3], making it important to build a solid data pipeline upfront.

Weigh the strengths and drawbacks of each metric to decide which to use when selecting a model. No metric is perfect and selecting a metric may be an iterative process.

Communicating uncertainty

Forecasting with a time series model allows for reasonable glimpses into the future. As with all forecasting, we must understand the inherent uncertainty built into the data and forecasts so that we can provide precise results to end users. This understanding directly stems from the process of picking confidence levels and in return receiving confidence intervals of desired future predictions. Generally, when providing a forecast, we must balance the ability for the audience to be confident in the forecast while limiting the range of possible values enough that they can act on those forecasts. These two concepts work against each other; as the confidence levels increase, the width of the confidence interval also increases.

The purpose of building understanding around uncertainty is to communicate the forecasting capabilities of the model . Everyone should have a common understanding of what the model can say about the future. It's considered good practice to include confidence intervals in visualizations of forecasts. The two key points with communicating uncertainty is first that the more confident we want to be with our predictions, the more room for error there is. The second is the more variance within the data, the harder it is to manage predictions.

Scaling up and out

Some degree of error is inevitable when using a model to generate forecasts into the future. How can we minimize this error to optimize accuracy and precision of predictions? One of the easiest ways is to step back and readdress the scope of the problem. Usually, we're interested in general question such as "In which months did we have the most sales?". General questions can be made more specific: for example, "In which months did we sell the most of item A within the Midwest region?". Differentiating between geographies, demographic groups, and item categories to model a single variable against time will help provide stronger results in forecasting. Variances in forecasts can be reduced by disaggregating the scope and performing multiple forecasts over subsets of the input data.

One drawback of increasing granularity of the data is that instead of building a few different generalized time series forecasts, we are deploying hundreds of models. Performing that many forecasts after categorizing the fields can seem daunting. Luckily, through parallelization techniques, we can streamline the process of building, retraining and forecasting the models so long as the categories are independent from each other. This reduces manual intervention and large build times while generating consistency throughout the model building procedures.

References

1. https://otexts.com/fpp2/MA.html
2. https://ieeexplore.ieee.org/document/8882943
3. https://www.researchgate.net/publication/261377261_Comparison_of_Criteria_for_Estimating_the_Order_of_Autoregressive_Process_A_Monte_Carlo_Approach
4. Time Series Analysis with Applications in R by Jonathan D. Cryer and Kung-Sik Chan