Quick and Easy Time-Series Forecasting in PowerBI: A Practical Guide

How to set up and configure a forecasting system in just a few clicks

Toward Data Science
https://towardsdatascience.com/quick-and-easy-time-series-forecasting-in-powerbi-a-practical-guide-c31a858ee220

Thomas A Dorfer
May 1, 2023

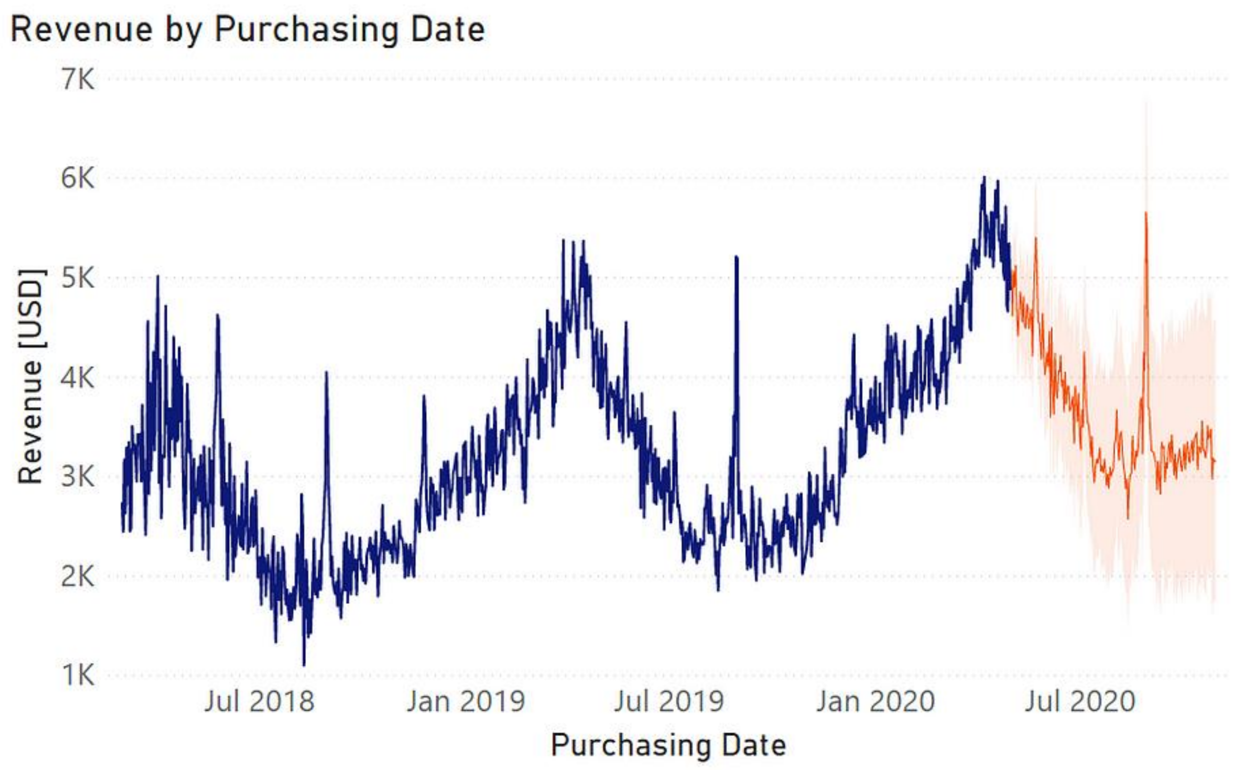

Image by the Author.

Introduction

Time-series forecasting has become a ubiquitous tool for businesses, governments, and individuals alike. It is nowadays being applied in almost every domain imaginable: in finance to predict stocks and interest rates, in healthcare to predict hospital bed capacity, in transportation to plan routes and traffic patterns, in energy to predict power supply and demand, and the list goes on.

Consequently, there exists a need for a user-friendly and easy-to-onboard tool that is quick and seamless to set up. Luckily, PowerBI has got you covered. With their built-in tool, users are able to set up and configure a forecasting system in just a few clicks.

This article will provide you with a step-by-step guide on how to achieve this. But before diving into the practical part, let’s go through the algorithm that’s running under the hood.

The Algorithm: Exponential Smoothing

PowerBI is using exponential smoothing — a robust algorithm capable of capturing trends in time-series data, while at the same time suppressing noise and unwanted variation.

In simple terms, exponential smoothing takes a weighted average of past observations, with more weight given to more recent observations. This means that the weights decrease exponentially as the observations get older. The idea behind this is that recent observations are more informative about future behavior than distant ones.

There are several variations of exponential smoothing, with each method using a different combination of the level, trend, and seasonality components to make predictions.

PowerBI’s tool automatically chooses between two algorithms from the ETS model family, depending on the seasonality of the historical data the user provides: (1) ETS AAA for seasonal data, and (2) ETS AAN for non-seasonal data.

Seasonal Data: ETS AAA

Let’s first break down this rather daunting acronym. Interestingly — or confusingly, depending on how you look at it — the second part of the acronym describes the nature of the components that the first part represents.

The first part, *ETS*, tells us the components that the time-series model is taking into account. In this case, it’s error (E), trend (T), and seasonality (S). The *A* in *AAA* stands for *additive*. With this information, we can now conclude that our time-series model takes into account *additive error*, *additive trend*, and *additive seasonality*.

- **Additive error** refers to the assumption that the errors or random fluctuations in the time-series are added to the expected values.
- **Additive trend** means that the expected value of the time-series changes by a constant amount over time. For instance, if a time-series has a trend component of 2, then the expected value at time *t+1* will be 2 units higher than the expected value at time *t*.
- **Additive seasonality** means that the seasonal component of the time-series adds a constant amount to the expected value for each season. For instance, if a time-series has a seasonal component of 5 for the month of July, then the expected value for July will be 5 units higher than that of any other month.

Non-Seasonal Data: ETS AAN

Based on the explanation in the previous section on how to interpret this acronym, you’ve probably already come to the conclusion that *ETS AAN* is also using *additive error* and *additive trend*. The *N* at the end of *AAN* simply means *non-seasonal*, indicating that the model doesn’t take into account seasonal patterns in the time-series.

Implementation in PowerBI

Let’s now take a look at the individual steps you need to follow in order to set up a time-series forecasting system in PowerBI.

Step 1: Load your time-series data into PowerBI.

Step 2: Create a *Line Chart* containing your time-series and make sure the X-axis type is set to *continuous*. For the illustrations in this article, I’m using sample data made available by Microsoft.

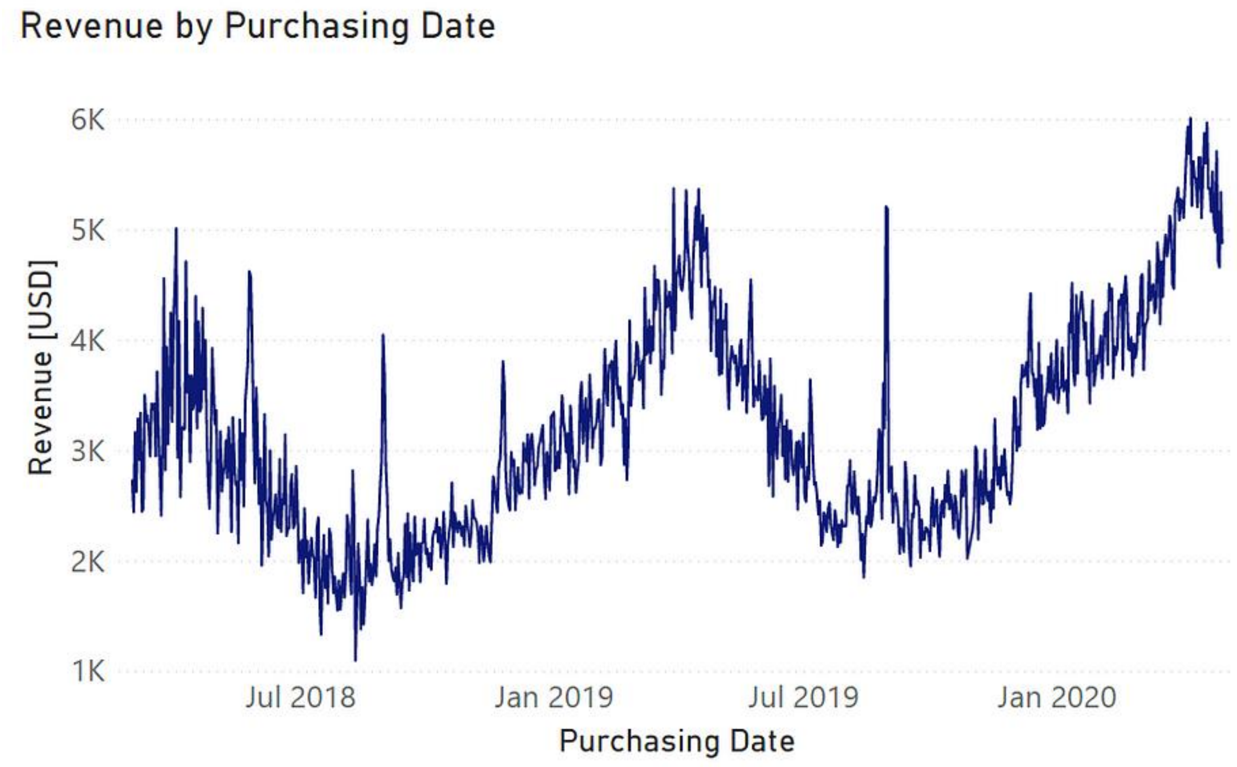

Image by the Author. License information for data usage: MIT License.

Step 3: In the *Visualizations* pane, navigate to *Add further analyses to your visual* and switch on *Forecast*.

Step 4: Under *Options*, you can set some parameters and custom configurations such as *units*, *forecast length*, *ignore the last*, *seasonality*, and *confidence interval*.

The *units* parameter applies to *forecast length* and *ignore the last* (we'll see later what this parameter means). In our example, I set *units* to "Months" and *forecast length* to "6", indicating that I'm attempting to make a projection 6 months into the future.

Without providing any further input, the result would look like this:

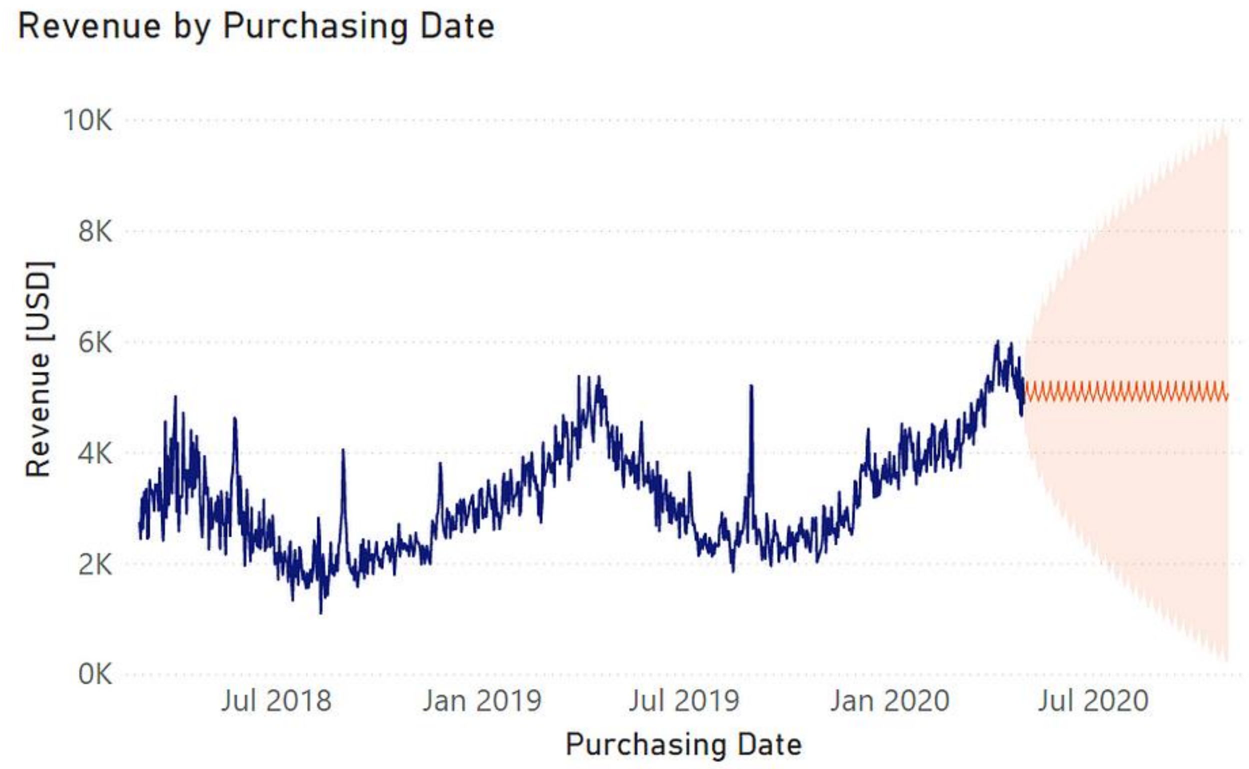

Image by the Author.

Doesn't look so good, does it? That's because the default value for *seasonality* is set to "Auto", and, presumably, our data doesn't include enough seasonality cycles for the tool to accurately detect it.

Just by looking at the line chart, we can see that our data has yearly seasonality. Revenue usually bottoms out in the summer months, then rises steadily until mid April, before starting to fall again, and the cycle repeats.

Therefore, we can manually set the *seasonality* to “365” points. Points in this case refers to the granularity of the time-series data. In our case, we have daily granularity — each data point in the chart represent a single day. Thus, 365 points means 365 days.

After tweaking the parameters as described, we get the following outcome:

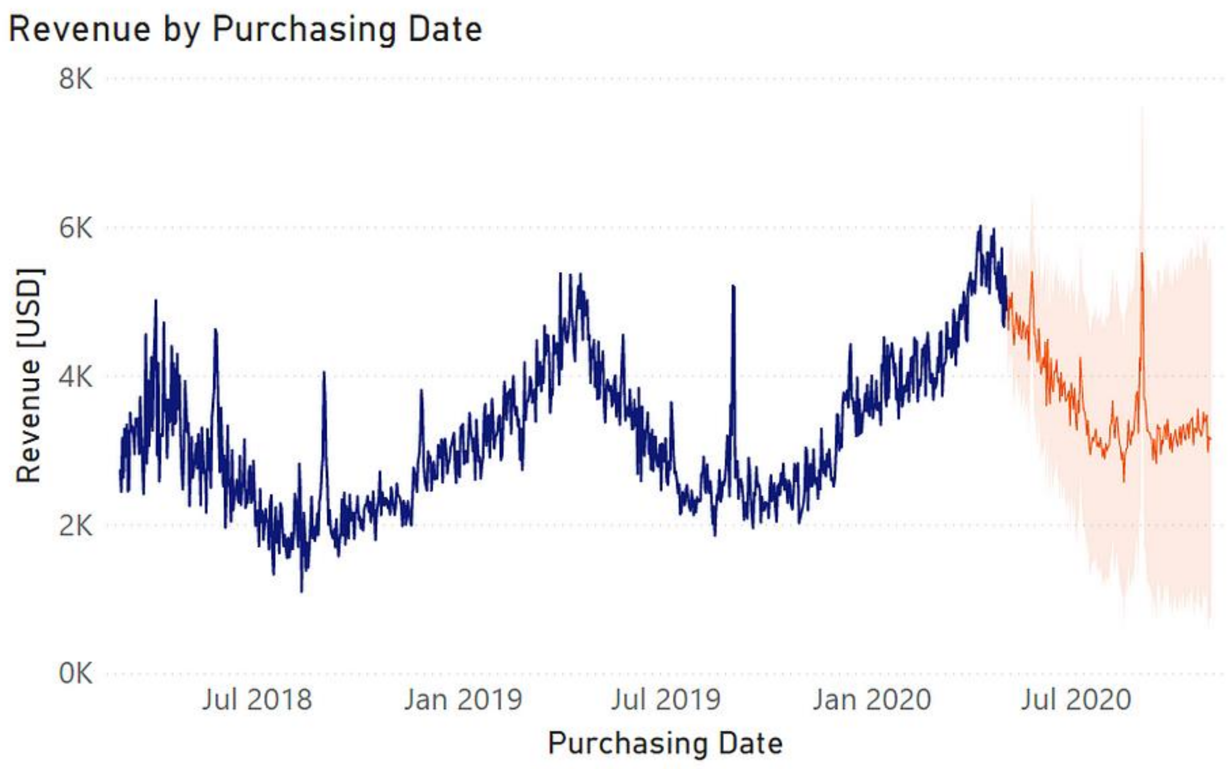

Image by the Author.

Much better! The model clearly picked up the seasonality in the data and also predicted the peak at the end of August that occurred in the previous two years as well. Moreover, by looking at the bottoms that occur every year in mid August, we can see that it also picked up the slight up-trend that’s present in this time-series.

When hovering over the plot, we can also inspect the exact value of the forecast as well as the upper and lower bounds of the confidence interval, which is set to 95% by default. This means that our model predicts with a 95% probability that the data will fall within this predicted range.

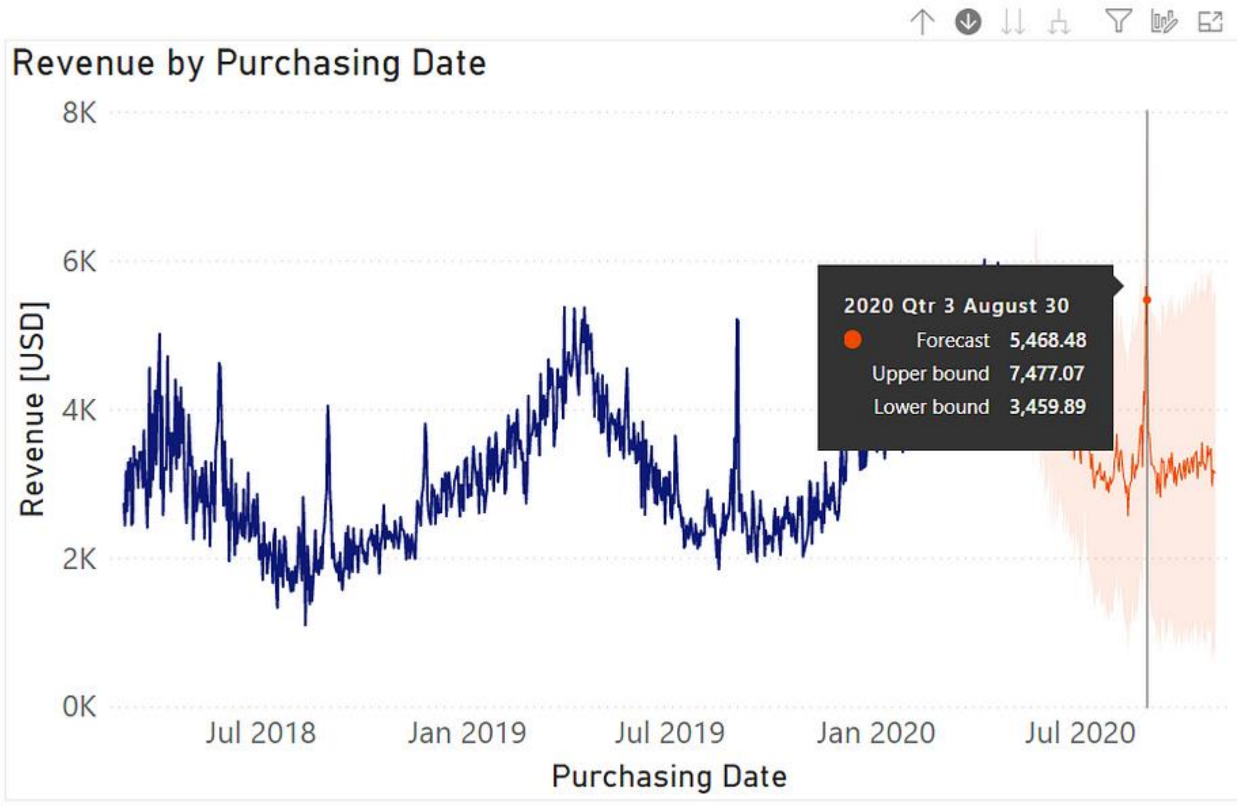

Image by the Author.

The further into the future we project the data, the wider the confidence interval becomes. This is because the uncertainty around the future values increases (think about the weather forecast — we can predict the weather with reasonable certainty for the upcoming day or two, but for 7-day or 14-day forecasts, the uncertainty is much higher).

Model Evaluation

Once you’ve set up your forecasting system, you can evaluate how well the model is performing through a process called *hindcasting*. Instead of predicting future values, hindcasting involves making “predictions” about past values and comparing those predictions to the actual values.

Using this method can give us an idea about the model's performance by evaluating how well it would have performed in the past.

This is the time to make use of the *ignore the last* parameter. Setting this parameter to "6", we're telling the model to perform a hindcast on the last 6 months of our time-series data.

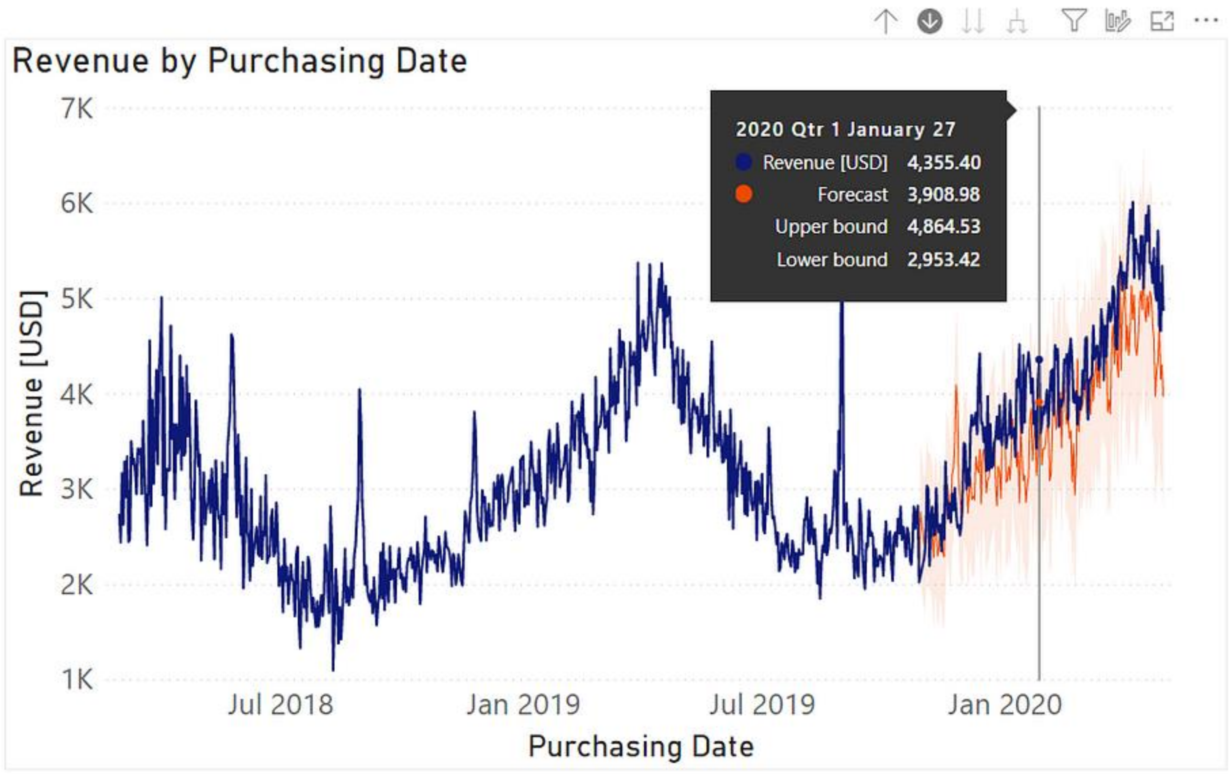

Image by the Author.

Looking at this hindcast, we can see that while the model is not 100% accurate — no model is, really — it does perform quite well overall. On average, the forecasts seem a little lower than the actual values. This could be due to the fact that it didn't capture well enough the overall up-trend of the data, or that the slope of the up-trend is actually increasing in this cycle. More historical data would typically resolve this issue.

Conclusion

PowerBI provides a truly seamless way of setting up a time-series forecasting system that only requires minimal tweaking in a matter of minutes. By leveraging this tool, organizations and individuals can make use of their historical data to generate actionable predictions about future trends and events. However, it is also important to keep in mind that forecasting requires constant refinement and evaluation, and that the accuracy of the projections are highly dependent on the quality and quantity of the underlying historical data.