**Choosing the Best ML Time Series Model for Your Data**

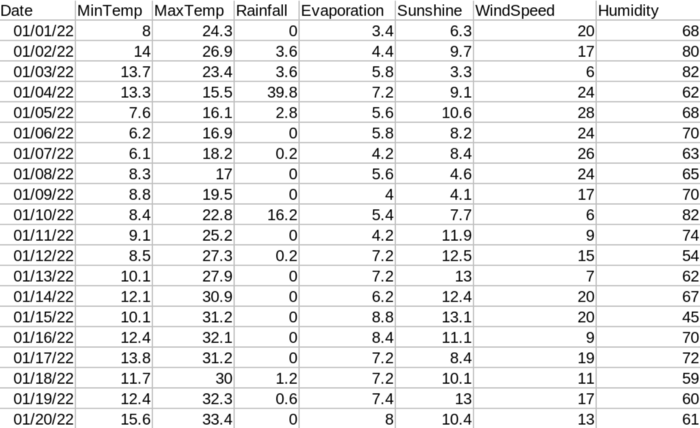
**With new developments every year — it’s becoming more challenging to decide on a proper model**



Photo by [Chris Liverani](https://unsplash.com/@chrisliverani?utm_source=medium&utm_medium=referral) on [Unsplash](https://unsplash.com?utm_source=medium&utm_medium=referral)

ARIMA, Prophet, LSTMs, CNNs, GPVAR, Seasonal Decomposition, DeepAR, and more. When it comes to time series models, there are a plethora of methods, meaning it is important to consider your options before committing to a model.

There are two key decisions that need to be made regarding beginning to hone in on a model: whether it will be local or global, and whether the predictions need to be univariate or multivariate. To explain these, let's introduce a modified [sample weather dataset](https://www.kaggle.com/datasets/zaraavagyan/weathercsv?resource=download) from Kaggle:



[Weather Data (](https://www.kaggle.com/datasets/zaraavagyan/weathercsv?resource=download)[CC0: Public Domain](https://creativecommons.org/publicdomain/zero/1.0/)[)](https://www.kaggle.com/datasets/zaraavagyan/weathercsv?resource=download)

[A **local** model](https://arxiv.org/abs/2202.02262) (also sometimes called an iterative or traditional model) only uses the prior values of a *single data column* to predict future values. Since local models only rely on a single data column, they must also be **univariate**, or we are predicting a single variable over time. In this example, a local, univariate model would be using the MaxTemp from days 1–20 to predict what the MaxTemp will be on day 21.

in contrast, [a **global** model](https://business-science.github.io/modeltime/articles/modeling-panel-data.html) uses many data columns to predict future values, generally time-independent variables. There are two types of global models, global univariate, in which we use many values to predict a single value, and global multivariate models, where we use many values to predict many values. A **global univariate** model would use MaxTemp, Evaporation, and Humidity in order to predict the future maxTemp value. A global multivariate model would utilize MaxTemp, Evaporation, and Humidity in order to predict the future value of MaxTemp, Evaporation, and Humidity.

**Local Univariate Models**

**Local univariate models are best for these use cases:**

1. your data is trivial

If your data is simple, univariate, and easy to predict, a classical approach to time series prediction may be best. Why? It can be trained immediately, requires little computing resources, and more complex models may be overkill and [overfit your data](https://www.ibm.com/cloud/learn/overfitting). An example of this may be predicting how many of a certain item will be sold in a store.

2. you need accurate predictions but only have 1 variable

If you only have 1 usable variable, [iterative forecasting](https://business-science.github.io/modeltime/articles/modeling-panel-data.html)/ensemble local models have proven time and time again that they can predict data more accurately than many ML models due to their ability to negate the downsides and weaknesses of any one particular model, therefore, while training may be relatively expensive, it remains one of the most popular methods of time series forecasting. [A popular example of this is predicting the stock market using past data.](https://medium.com/p/f7b4d6f3bb2) (If you are interested in doing so, [here](https://medium.com/p/f7b4d6f3bb2) is where you can get data)

3. Models that are predictably seasonal

If you know your data follows predictable seasonal patterns, many time series such as SARIMA (Seasonal Autoregressive moving average) is built to handle data when you are confident in what your “season” is. An example of this may be web traffic, where you know data follows a regular pattern on a daily basis. In this case, you can define a model with daily seasonality.

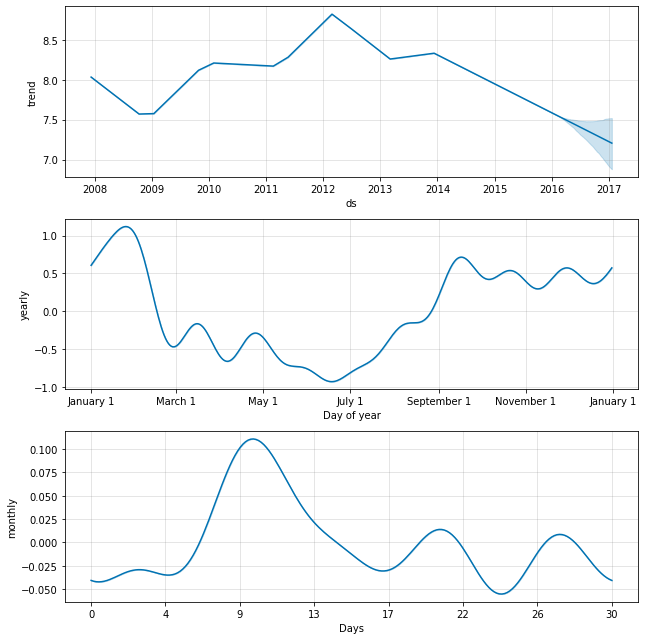
**Examples of Local Univariate models (from simplest to most complex — with brief explanations and links to deeper reads)**

Moving Average — Simplest method and can be [computed with 1 line of pandas](https://stackoverflow.com/questions/40060842/moving-average-pandas)

[Exponential Smoothing/Holt-Winters](https://timeseriesreasoning.com/contents/holt-winters-exponential-smoothing/) — Exponential Smoothing predicts values using a weighted average of all previous values, where more recent values are weighted higher. In Holt-Winters, seasonality and trends are taken into the equation as parameters

[ARIMA/SARIMA/Auto-ARIMA](https://towardsdatascience.com/time-series-forecasting-with-arima-sarima-and-sarimax-ee61099e78f6) — At its base, it is a derivative of a moving average plus an autoregressive term ([using past values with noise](https://en.wikipedia.org/wiki/Autoregressive_model" \t "_blank)) in order to predict future values. SARIMA takes into consideration seasonality and [Auto-ARIMA](https://github.com/alkaline-ml/pmdarima) will perform a search to try and find the optimal model parameters

[Prophet](https://research.facebook.com/blog/2017/2/prophet-forecasting-at-scale/) — Developed by Facebook and since open-sourced, Prophet is a regression model that incorporates a linear or logistic growth trend, seasonal components, and changepoint detection. A cool thing about Prophet is its ability to separate out these trends and plot them for you!



<https://research.facebook.com/blog/2017/2/prophet-forecasting-at-scale/>

[Iterative Forecasting](https://business-science.github.io/modeltime/articles/modeling-panel-data.html#traditional-modeling-iteration) — Iterative Forecasting is simply using many of the above models in conjunction to create a prediction!

**Global Univariate Models**

**Global univariate models are best for these use cases:**

1. You have a lot of supplementary variables and are looking to predict a singular value in the future. An example of this may be temperature forecasting, like in our toy example, using variables such as humidity, wind speed, season, etc to predict the daily temperature.
2. You are unaware of the seasonality or trends of your model — a benefit of ML models is that they, by design, are able to detect patterns in the data not immediately visible to an observer. An example of this may be predicting a [hardware failure](https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7887121/). While a human may have a hard time diagnosing what variables may put hardware at risk, these global time series models have much more success
3. You need to train many time series in a single model — for many of the deep learning implementations, the model can simultaneously learn many time series models. An example of this, going back to our temperature example, is if we have data from multiple regions, we can train a single ML model that can predict any of the individual regions and even learn the patterns between regions!

**Examples of Global Univariate models (from simplest to most complex — with brief explanations and links to deeper reads)**

[SARIMAX](https://towardsdatascience.com/time-series-forecasting-with-arima-sarima-and-sarimax-ee61099e78f6) — SARMAX is simply SARIMA (discussed earlier) that takes into account exogenous (outside) variables to allow the time series to adapt to changing variables faster

[Tree-Based Methods](https://www.sciencedirect.com/science/article/pii/S0169207021001679#b37) — Trees can be thrown at almost every problem with some success, and time series are not much different. They tend to be advantageous if data is sparse, are accurate, and are relatively fast to train in comparison to deep methods (discussed below). The most popular current implementations are [lightgbm](https://pypi.org/project/lightgbm/) and [xgboost](https://www.kaggle.com/code/robikscube/tutorial-time-series-forecasting-with-xgboost)

MLP-Based Methods — Using a classic fully-connected neural network for prediction produces some pretty fantastic results, [even winning international competitions](https://arxiv.org/abs/1905.10437). The current most popular implementations are [N-BEATS](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.n_beats.html) and [GP Forecaster](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.gp_forecaster.html), which mixes an MLP with [gaussian embeddings](https://openreview.net/forum?id=HJ7O61Yxe), or density-based distributions

CNN-Based Methods — [Convolutional Neural Networks are similar to MLPs](https://medium.com/data-science-bootcamp/multilayer-perceptron-mlp-vs-convolutional-neural-network-in-deep-learning-c890f487a8f1) except they are not fully connected. CNNs are widely used because they tend to be smaller, less wasteful, and easier to train. Unfortunately, implementations are sparse, but there are [many scholarly articles detailing how they work](https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00599-y)

RNN/LSTM-Based Methods — The current “state-of-the-art” among researchers, [RNNs are neural networks that “loop” into each other.](https://towardsdatascience.com/temporal-loops-intro-to-recurrent-neural-networks-for-time-series-forecasting-in-python-b0398963dc1f#:~:text=It%20makes%20recurrent%20connections%20by,of%20inputs%20without%20loosing%20track.) The reason this works so well is it allows subsequent data points to “remember” what has already been processed in the few preceding points allowing for more dynamic predictions, as time series are naturally dependent on the previous values. [LSTMs](https://colah.github.io/posts/2015-08-Understanding-LSTMs/) are a more specific and popular type of RNN and stand for “Long short term memory”. A disadvantage of this method is because it relies so heavily on previous data points, long-term predictions with RNNs tend to be less reliable than some other methods. RNN methods are wildly popular and some of the more state-of-the-art implementations are [DeepAR](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.deepar.html), [DeepState](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.deepstate.html), and [DeepFactor](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.deep_factor.html).

**Global Multivariate Models**

**Global multivariate models are best for these use cases:**

1. You have a lot of supplementary variables and are looking to predict many or all of the values in the future. Going back to our temperature example, we would be using variables such as temperature, humidity, wind speed, season, etc, and generating predictions into the future *for many or all* of these variables.
2. Like above, global multivariate models are also good if you are unaware of the seasonality or trends of your model
3. You need to train many time series for many different variables, all wrapped into a single model efficiently— An example of this, going back to our temperature example, is if we have data from multiple regions, we can train a single ML model that can predict ANY variable from ANY region!

**Examples of Global Multivariate models**

RNN/LSTM-Based Methods — Almost every single Global Multivariate implementation is some variant of an RNN/LSTM model, having small differences between one another, and some of these even being adapted from their univariate versions to make predictions for any and all variables. Popular implementations are [DeepVAR](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.deepvar.html) (a variant of DeepAR), [GPVAR](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.gpvar.html), which incorporates gaussian process into RNNs, and [LSTNet](https://ts.gluon.ai/stable/api/gluonts/gluonts.model.lstnet.html), an LSTM variant.

**Conclusion**

Every model listed above has its merit, its strengths, and its weaknesses. I hope this article provided a useful guide to get you started on your time series journey, as the options can be overwhelming.

If you found the guide useful, be sure to save/bookmark the story if you think you will need to come back for reference! If you enjoyed this article, feel free to [follow me](https://jerdibattista.medium.com/), and read more of what I write, or use me [as a referral](https://jerdibattista.medium.com/membership) so I can continue to make content I love. I write a lot in the Data Science/ML space!