Machine Learning for Time Series Analysis

Forecasting

Fortunato Nucera[1]

[1]Department of Mathematics
Imperial College London

Final Project Preparation - Week 2, 16th May 2023

Table of Contents

Table of Contents

About the code

I have decided to host the code on github rather than providing a screenshot on the slides. This is because I want to make sure that the code is up-to-date and versioned. You are welcome to
check out the GitHub repository I have created for sharing the Python code.

https://github.com/tinosai/ThesisRepo

Table of Contents

Gaussian Processes for Forecasting

This section assumes prior knowledge of Gaussian Processes (GPs). More information about GPs can be found at [1]. The main idea in the use of GPs for Forecasting is that **the covariates in the GP are the timestamps of the series**: $t_1, t_2, \ldots, t_n$.

Let us define a time series $\{X_t\}$ as sampled from a GP with prior mean $\mu_t = 0$ and prior covariance matrix $\mathbf{K}$, which is determined by a kernel k.

$$\{X_t\} \sim \text{GP}(0, k)$$

where the prior covariance matrix is determined by the kernel function $k(t, t')$. In literature, we can find a variety of kernel used for Gaussian processes such as:

- **Constant kernel**: $k(t, t') = c \quad \forall t, t'$ where c is a constant $c \in \mathbb{R}$.
- **White Noise kernel**: $k(t, t') = c\mathbf{1}_{\{t=t'\}}$
- **Dot-Product Kernel** : $k(t, t') = \sigma_0^2 + tt'$
- **Matern kernel**: $k(t, t') = \frac{1}{\Gamma(\nu)2^{\nu-1}} \left(\frac{\sqrt{2\nu}}{l} d(t, t') \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}}{l} d(t, t') \right)$ where K_ν is a modified Bessel function.

Gaussian Processes for Forecasting

- **Rational Quadratic Kernel** : $k(t,t') = \left(1 + \frac{d(t,t')^2}{2\alpha l^2}\right)^{-\alpha}$
- **Radial Basis Function kernel**: $k(t,t') = \exp\left(-\frac{d(t,t')}{2l^2}\right)$ where d is $L2$ distance.
- **Exp-Sine Squared Kernel** : $k(t,t') = \exp\left(-\frac{2\sin^2 \pi d(t,t')/p}{l^2}\right)$

The appropriate choice of the kernel is essential, since the posterior distribution mean is affected by the kernel function (see Equations 2.25 and 2.26 from [1]). While there are no strict rules on the choice of the kernel function, [2] provides some "rules of thumb". For example, if the underlying function is believed to be periodic, a periodic kernel - Exp-Sine-Squared - should be used, etc.
Usually, most users only rely on the Radial Basis Function (RBF) kernel or on the Dot-Product kernel, but we will see that these options do not lead to satisfactory results for time series with a seasonal component.

Gaussian Processes for Forecasting

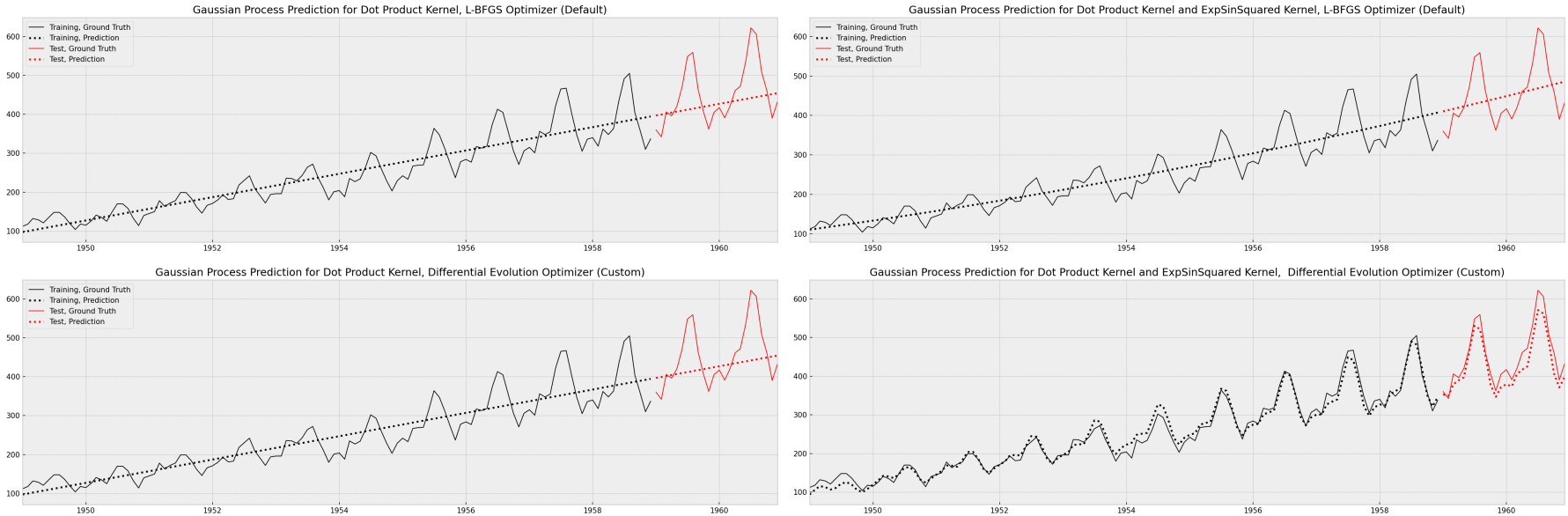
The parameters of the kernels are to be selected by maximizing the MLE. Equation 2.30 from [1] provides the equation for the marginal likelihood of the GP.

$$\log p(y|x) = -\frac{1}{2} y^T (\mathbf{K} + \sigma^2 I)^{-1} y - \frac{1}{2} \log |\mathbf{K} + \sigma^2 I| - \frac{n}{2} \log(2\pi) \tag{1}$$

Where $|\mathbf{K} + \sigma^2 I|$ represents the determinant of matrix $\mathbf{K} + \sigma^2 I$. The maximization of Equation 1 is not easy! Nelder-Mead and L-BFGS methods fail more often than not. As a result, we need to make sure that the optimization algorithm for marginal log-likelihood is an effective one!
The code for the following images can be found in the notebook `GaussianProcessForecasting.ipynb`.

Gaussian Processes for Forecasting

In the Figure below, we fit a GP to the Air Passengers time series. The color indicates whether the data is used for training (black) or test (red). The solid line represents the ground truth, whereas the dotted line represents the prediction.
The models on the left column only use a Dot-Product kernel, whereas the models on the right column also include the Exp-Sine-Squared kernel (which captures the seasonality). The upper row includes the models whose kernels have been fit using L-BFGS, whereas the bottom row indicates the models using the evolutionary algorithm Differential Evolution for training.

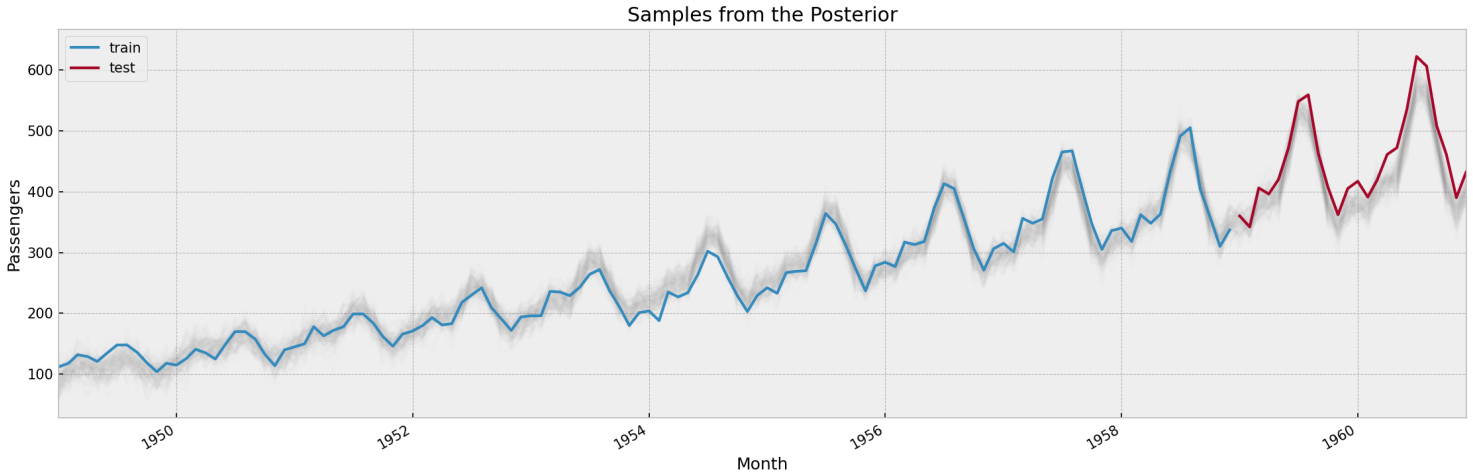


Gaussian Processes for Forecasting

We notice that:

- **The models trained with L-BFGS (the default optimization algorithm) are completely unable to capture any patterns in the time series other than the trend**
- If the Exp-Sine-Squared kernel is not included, the seasonal component (periodic) cannot be learnt.
- Exp-Sine-Squared kernel coupled with an appropriate optimization algorithm (Differential Evolution) provides a very good fit even on the test set.

The samples of the posterior (figure below) are close to each other, which supports our belief that the model is correctly specified. However, we need to note that the posterior samples tend to underestimate the ground truth values on the test set.



Gaussian Processes for Forecasting

Why would one use GPs instead of ARIMA/Holt-Winters?
Benefits:

- the time series does not necessarily have to be stationary, as the GP is - at least in theory - able to fit arbitrary complex trends.
- the appropriate choice of kernels and their parameters' tuning via the maximization of the marginal log-likelihood allows the model to learn the seasonality automatically.
- the GP does not require that the data be equally spaced in time. It is also robust against missing data.

Drawbacks:

- GPs are extremely flexible models, prone to overfitting the training set, and their marginal log-likelihood may be very difficult to optimize. It is essential that an appropriate maximization algorithm is chosen as the commonly used Nelder-Mead and L-BFGS may fail, since the former does not deal well with multivariate optimization, and the latter requires the computation of gradients, which may be quite unstable when the marginal likelihood is too "wiggly". For this reason, we use a Differential Evolution algorithm, which is much more stable and less likely to get trapped into local minima.

Table of Contents

Hybrid Methods

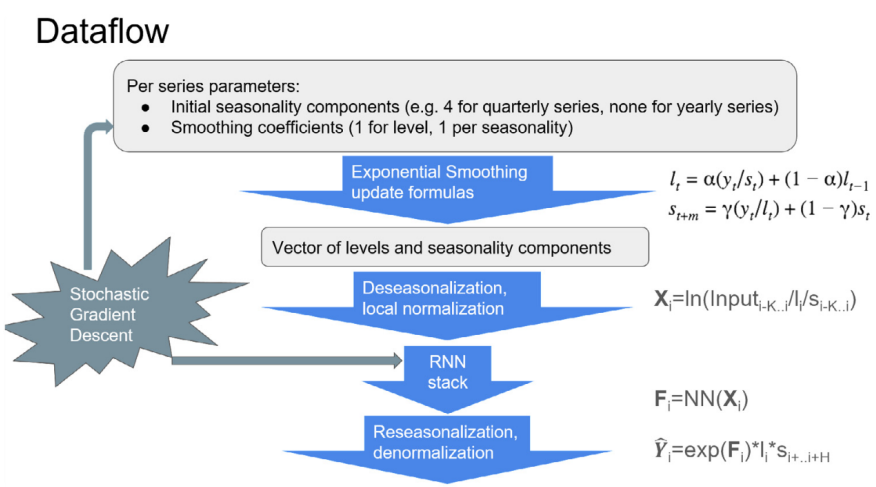
Background Information:

- In the past years, it has become evident that the use of ML models - and by extension also DL models - is unable to reach the accuracy of traditionally statistical models such as Holt-Winters and ARIMA [3].
- The history of forecasting competitions can be found in [4], and the details of the dataset for the most recent competition - M4 - can be found in [5].

The family of *Hybrid Methods* defines a group of time series forecasting models which leverage both statistical and ML methods. One such model is described in [6], which was the winner of the M4 competition [5].

Smyl's ES-RNN Hybrid Model

Smyl's [6] novelty mainly consists in combining a statistical model (Holt-Winters) with a ML model (LSTM). A sketch of the model can be found below.



Non-seasonal models:

$$l_t = \alpha y_t + (1-\alpha)l_{t-1} \quad (1)$$

Single seasonality models:

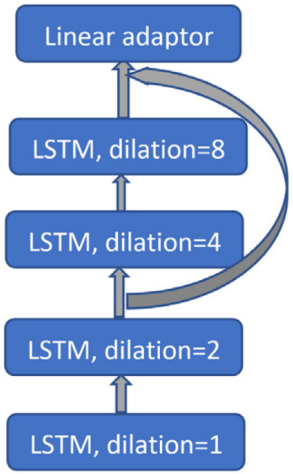
$$l_t = \alpha y_t/s_t + (1-\alpha)l_{t-1}$$
$$s_{t+K} = \beta y_t/l_t + (1-\beta)s_t \quad (2)$$

Double seasonality models:

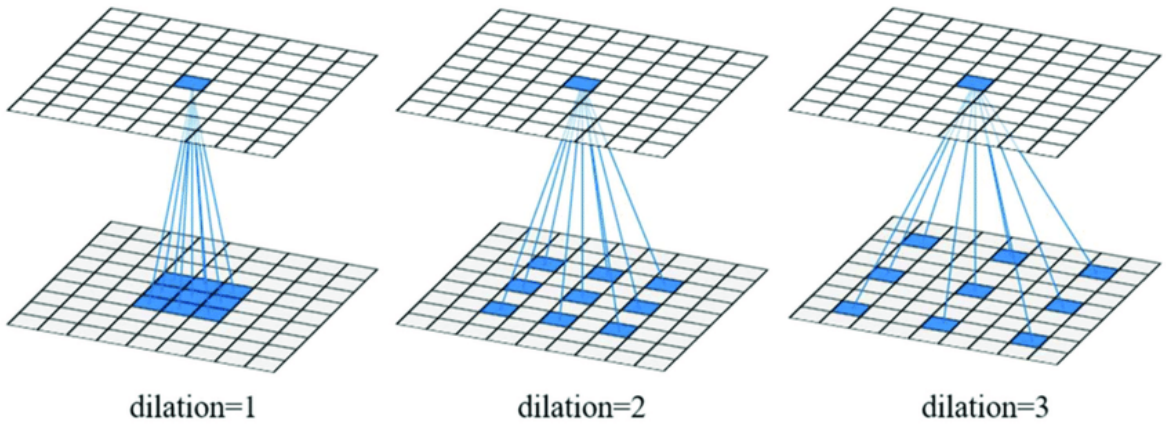
$$l_t = \alpha y_t/(s_t u_t) + (1-\alpha)l_{t-1}$$
$$s_{t+K} = \beta y_t/(l_t u_t) + (1-\beta)s_t \quad (3)$$
$$u_{t+L} = \gamma y_t/(l_t s_t) + (1-\gamma)u_t,$$

- Time Series are assumed to follow a multiplicative Holt-Winters Model. Depending on whether the incoming time series is *not seasonal*, *single-seasonal* or *double-seasonal* a specific Holt-Winters model (without trend) is assumed.
- The parameters of the Holt-Winters model (smoothing parameters and initial seasonality values) are *local* and *learnt*, i.e. time-series specific.
- The time series is split into chunks via sliding window for both inputs and outputs. The time series in these two windows are normalized by the last *level* and the *seasonality component* in the input window and the logarithm is applied.

Smyl's ES-RNN Hybrid Model



- The normalized time series - rescaled and stripped of their seasonality - are passed through an LSTM model **shared across all the time series in the data set**. The LSTM does not appear in its vanilla implementation, but *dilated*. Dilation in NNs refers to the practice of skipping input chunks in a predictable manner in order to process a larger input while maintaining the total amount of parameter constant (example provided for CNNs).
- The output of the LSTM is scaled back using the seasonality and level the series was divided by at the normalization step.



Smyl's ES-RNN Hybrid Model

The model showed a positive bias in the forecast. To mitigate this, the author trained the model using the sum of two different losses

- **Pinball Loss**:

$$L_t = \begin{cases} (y_t - \hat{y}_t)\tau & \text{if } \hat{y}_t \leq y_t \\ (y_t - \hat{y}_t)(\tau - 1) & \text{if } \hat{y}_t > y_t \end{cases}$$

  The pinball loss uses $\tau \in [0, 1]$ in order to control the penalty given in case of overestimation/underestimation of the ground truth. Typically, values in the range $\tau \in [0.45, 0.55]$ are chosen. Since the ES-RNN model tends to experience positive bias, a value smaller than 0.5 to penalize overestimation is employed.

- **Level Wiggliness Penalty**:

$$L_{\text{wiggle}} = \frac{c}{N-2} \sum_{i=1}^{N-2} (\log y_{t+2} - 2 \log y_{t+2} + \log y_t)^2$$

  Note that this loss is applied to the inputs of the LSTM, corresponds to second-order differencing, and employs a constant c which is used to scale up its contribution to the total loss to be minimized via gradient descent.

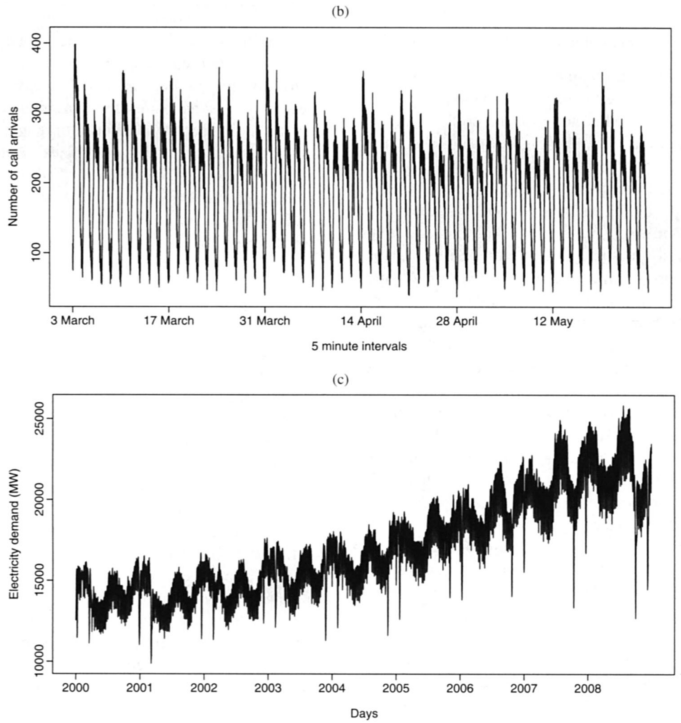
The final prediction is the average of an ensemble of the described model trained on different non-random subsets, a technique called *Ensemble of Specialists*.

Table of Contents

Hyndman's BATS and TBATS

Although time series with a single seasonality component are common, there exist time series which exhibit multiple seasonality patterns. For example:

- the number of call arrivals at a call center in a given time interval has daily and weekly seasonality
- the electricity demand may have both yearly and 6-month seasonality



In [7], two novel approaches for time series forecasting with multiple seasonalities, similar to Holt-Winters, are introduced.

Hyndman's BATS and TBATS

Given a positive-valued time series $\{x_t\}$, the BATS models proceeds as described below:

- A Box-Cox transformation is applied:

$$x_t^{\omega} = \begin{cases} \frac{x_t^{(\omega)} - 1}{\omega} & \omega \neq 0 \\ \log x_t & \omega = 0 \end{cases}$$

- An additive Holt-Winters model, with damped trend (via the parameter ϕ) is used. We name it BATS$(\omega, \phi, p, q, m_1, m_2, \ldots, m_T)$

$$x_t^{(\omega)} = \ell_{t-1} + \phi b_{t-1} + \sum_{i=1}^{T} s_{t-m_i}^{(i)} + d_t$$

$$\ell_t = \ell_{t-1} + \phi b_{t-1} + \alpha d_t$$

$$b_t = (1-\phi)b + \phi b_{t-1} + \beta d_t$$

$$s_t^{(i)} = s_{t-m_i}^{(i)} + \gamma_i d_t$$

$$d_t = \sum_{i=1}^{p} \psi_i d_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-1} + \epsilon_t$$

Here, T represents the number of seasonal components in the model. Consider that each seasonality component $s_t^{(i)}$ involves the optimization of one seed initializer with m_i parameters (i.e. for component i, we need to optimize also $s_1^{(i)}, s_2^{(i)}, \ldots, s_{m_i}^{(i)}$).

The model for TBATS (*Trigonometric BATS*) follows exactly the same structure as BATS. the only difference is the way the seasonality $s_t^{(i)}$ is specified. In TBATS, the seasonality is the sum of some Fourier harmonics up to a given index denoted by k_i for the i-th seasonal component. TBATS is denoted as TBATS$(\omega, \phi, p, q, \{m_1, k_1\}, \ldots, \{m_T, k_T\})$

$$s_t^{(i)} = \sum_{j=1}^{k_i} s_{j,t}^{(i)}$$

$$s_{j,t}^{(i)} = s_{j,t-1}^{(i)} \cos \lambda_j^{(i)} + s_{j,t-1}^{*(i)} \sin \lambda_j^{(i)} + \gamma_1 d_t$$

$$s_{j,t}^{*(i)} = -s_{j,t-1}^{(i)} \sin \lambda_j^{(i)} + s_{j,t-1}^{*(i)} \cos \lambda_j^{(i)} + \gamma_2 d_t$$

In general, TBATS is more flexible and, thanks to the Fourier representation, it is able to capture non-integer seasonality as well. For each seasonality, we have $2k_i$ parameters (γ_1 and γ_2 for each of the harmonics in the Fourier series). If $k_i = m_i/2$, then TBATS and BATS feature the same model compexity. However, if the seasonal pattern does not change over time, fewer harmonics are needed, thus leading to a more parsimonious model.
Model selection is usually performed via AIC.

References


C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge: MIT Press, 2005, available also in a print ed.


D. Duvenaud, "Automatic model construction with gaussian processes," Ph.D. dissertation, University of Cambridge, Pembroke College, 2014.


S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," *PloS one; PLoS One*, vol. 13, no. 3, p. e0194889, 2018.


R. J. Hyndman, "A brief history of forecasting competitions," *International Journal of Forecasting*, vol. 36, no. 1, pp. 7–14, 2020, m4 Competition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016920701930086X


S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The m4 competition: 100,000 time series and 61 forecasting methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020, m4 Competition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169207019301128


S. Smyl, "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting," *International Journal of Forecasting*, vol. 36, no. 1, pp. 75–85, 2020, m4 Competition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0169207019301153


A. M. De Livera, R. J. Hyndman, and R. D. Snyder, "Forecasting time series with complex seasonal patterns using exponential smoothing," *Journal of the American Statistical Association*, vol. 106, no. 496, pp. 1513–1527, 2011, 15. [Online]. Available: http://www.jstor.org/stable/23239555