Lecture 8: Advanced Topics: Advanced Forecasting Methods, Calibration, Scoring, and Ensembling

Introduction to Time Series, Fall 2023

Ryan Tibshirani

1 Advanced forecasters

- Thus far, we've learned in-depth about ARIMA and ETS as our two major forecasting frameworks. Undoutedbly, these are batted-tested frameworks that have been used for decades and will take you far, albeit with proper scrutiny
- Of course, there are actually many other forecasting methods out there, and this continues to be an active topic of research. Below, we briefly summarize three other forecasting methods, chosen based on their popularity (they also seem to constitute a common cast of characters, together with ARIMA and ETS, in R and Python forecasting packages)

1.1 Theta model

- First on our list is the *Theta model*, proposed by Assimakopoulos and Nikolopoulos (2000). This is on our list not as an advanced forecaster per se (as you will see, it's actually quite simple)
- Instead, it is a simple and popular method that began gaining a lot of attention in parts of the forecasting community, and is closely connected to something you already know: exponential smoothing
- Our presentation here follows Hyndman and Billah (2003), who give a nice, clear perspective on the Theta method and its connection to exponential smoothing
- Given data x_t, $t = 1, 2, 3, \dots$, the Theta method starts by defining a smoothed sequence $y_{\theta,t}$, $t = 1, 2, 3, \dots$ by the second-order difference equation:

  $$\Delta^2 y_{\theta,t} = \theta \Delta^2 x_t$$

  Here $\Delta^2 = (1-B)^2$ is the second-order difference operator, just like in our ARIMA lecture, and $\theta \geq 0$ is a parameter
- The solution to the above is given by

  $$y_{\theta,t} = a_\theta + b_\theta t + \theta x_t$$

  for some (constant in time) intercept and slope parameters a_θ, b_θ
- For fixed θ, the intercept and slope parameters are fit by minimizing the sum of squared errors to the original sequence

  $$\min_{a_\theta, b_\theta} \sum_{t=1}^n (x_t - y_{\theta,t})^2 \iff \min_{a_\theta, b_\theta} \sum_{t=1}^n \Big((1-\theta) x_t - a_\theta + b_\theta t \Big)^2$$

  which is simply a linear regression of $(1-\theta)x_t$ on time t
- Once these estimates $\hat{a}_\theta, \hat{b}_\theta$ are found, forecasts are made by running simple exponential smoothing (SES) on the sequence

  $$\hat{y}_{\theta,t} = \hat{a}_\theta + \hat{b}_\theta t + \theta x_t$$

  if $\theta > 0$, or by extrapolating the line $\hat{a}_0 + \hat{b}_0 t$ forward in time if $\theta = 0$

- Assimakopoulos and Nikolopoulos (2000) make the following general recommendation:
  - produce forecasts $\hat{y}_{0,t+h|t}$ with $\theta = 0$ (recall this is just extending the line $\hat{a}_0 + \hat{b}_0 t$);
  - produce forecasts $\hat{y}_{2,t+h|t}$ with $\theta = 2$ (recall this is given by just running SES on $\hat{y}_{2,t}$)
  - return their average: $(\hat{y}_{0,t+h|t} + \hat{y}_{2,t+h|t})/2$.

  Seasonality, if present, is estimated and removed before running this procedure, and added back in at the end
- Hyndman and Billah (2003) show that this procedure is quite similar to running Holt's linear trend method with an estimated slope $\hat{b}_0/2$ and with $\beta = 0$ (no evolution of the slope over time)
- They also compare the Theta method with Holt's linear trend method on the daa from the M3 forecasting challenge (where the Theta method performed well and subsequently gained popularity), and observe that Holt's linear trend performs competitively
- It is worth knowing about the close connections between Theta and exponential smoothing, because much of the literature on the Theta model does not seem to emphasize this aspect. There has been more recent work on the Theta model (which may further distinguish it from exponential smoothing—we cannot say, because we have not followed it) that you may be interested in reading

1.2 Prophet model

- Next on our list is the *Prophet model*, proposed by Taylor and Letham (2018), from Facebook. This has become popular for large-scale forecasting enterprises, and the popular opinion seems to be that its advantages over traditional ARIMA or ETS models are twofold: flexibility and speed. But make sure to read on, especially to the end of this subsection, for further discussion of this
- The Prophet model is a particular type of signal plus noise model, where we model the given time series as

$$x_t = g_t + s_t + h_t + \epsilon_t$$

  where ϵ_t, $t = 1, 2, 3, \dots$ is a white noise sequence, and:
  - g_t represents a trend component
  - s_t represents a seasonal component
  - h_t captures holiday/calendar effects
- This can be seen as a particular type of smoother, based on a particular model for the trend (which we will describe shortly; the seasonal and holiday components are fairly generic). We could also refer to it as a particular type of additive model, where the regressor is time
- The seasonal component is parametrized by a Fourier (cosine and sine) basis at given fixed, known periods chosen by the user. For frequencies ω_j, $j = 1, \dots, p$ (equivalently, periods $1/\omega_j$, $j = 1, \dots, p$), recall, this is:

$$g_t = \sum_{j=1}^{p} \Big(a_j \cos(2\pi\omega_j t) + b_j \sin(2\pi\omega_j t) \Big)$$

  for coefficients a_j, b_j, $j = 1, \dots, p$
- The holiday/calendar component is simply parametrized using indicator variables

$$h_t = \sum_{j=1}^{m} \alpha_j \cdot 1\{t \in D_j\}$$

  where α_j, $j = 1, \dots, m$ are coefficients and each D_j is a set of dates representing a particular holiday or calendar event (e.g., Christmas, Thanksgiving, etc.)

- Finally, the trend component is modeled in one of two ways. The first way is for *saturating* trends. For this, Taylor and Letham (2018) propose to model g_t using a sigmoid function with a piecewise growth rate:

$$g_t = \frac{C(t)}{1 + \exp\Big(c_0 + c_1 t + \alpha \sum_{j=1}^r \beta_j \cdot (t - t_j)_+\Big)}$$

  Here $C(t)$ is a (possibly) time-varying capacity, which it appears Taylor and Letham (2018) recommend be set externally (e.g., based on market sizes considerations)

- For *non-saturating* trends, Taylor and Letham (2018) propose to model g_t using a piecewise linear trend directly:

$$g_t = c_0 + c_1 t + \alpha \sum_{j=1}^r \beta_j \cdot (t - t_j)_+$$

- In either case (saturating or non-saturating), β_j, $j = 1, \dots, r$ are coefficients to be estimated. Also, t_j, $j = 1, \dots, r$ are knots (where the slope changes), which, in the simplest case, can be fixed ahead of time. Instead, Taylor and Letham (2018) recommend that knots be selected using ℓ_1 penalization from a large initial set of locations. That is, they use the ℓ_1 penalty

$$\sum_{j=1}^r |\beta_j|$$

  when fitting the model, which is like a special type of lasso regression. (In fact, though it may not be obvious at first pass, placing an ℓ_1 penalty on β_j, $j = 1, \dots, r$ here is actually equivalent to reparametrizing the entire sequence as $g_t = \theta_t$, and then using an ℓ_1 penalty on second differences of θ_t; recall, this is the penalization scheme used in *trend filtering*, which you learned about earlier in the course when we covered smoothing)

- Altogether, the Prophet model is fit by minimizing the sum of squared errors to the observed data, over all parameters $a_j, b_j, \alpha_j, c_0, c_1, \beta_j$ that determine the decomposition, with a squared ℓ_2 (ridge) penalty on the parameters a_j, b_j, α_j for the seasonal component s_t and holiday component h_t, and an ℓ_1 (lasso) penalty on the parameters β_j for the trend component g_t

- Forecasts are generated by extrapolating the fitted components forward in time. For s_t and h_t, this is straightforward, because they are periodic in nature. For g_t, this is done by holding the slope (i.e., growth rate in the saturating model) constant from its last value, as we move forward in time

- (Though unimportant for our purposes here, Taylor and Letham (2018) actually phrase all of this in the context of a hierarchical Bayesian model, with normal and Laplace priors that serve the purpose of regularization; this Bayesian machinery also provides added stochasticity in computation of prediction intervals)

- So, how does the Prophet model compare to ARIMA and ETS? It depends on who you ask. In their original paper, Taylor and Letham (2018) find ARIMA and ETS models to be too rigid in their motivating examples—take a look at Figure 3 in their paper, and compare Figure 4, which displays Prophet forecasts

- Indeed, in their traditional forms, ARIMA and ETS models lack the flexibility of the Prophet model, particularly the flexibility exhibited in the latter's trend component. However, both ARIMA and ETS can be extended to accommodate more sophisticated trends. With ARIMA, you have actually already seen a way to do this: we can phrase the problem as *regression with correlated errors* (where we use an ARIMA model for the errors), and we can use time for the regressor and create flexible basis functions to model trends just like Prophet does

- Hyndman and Athanasopoulos (HA) call this a *dynamic regression model* and study it in Chapter 10 of their book. The advantage this has over Prophet is that it is able to capture auto-correlations in the errors, which can lead to narrower prediction intervals. The advantage Prophet has is speed: it is usually more efficient to fit the Prophet model, since its error model (white noise) is simpler and this makes the optimization a version of penalized least squares

1.3 Neural network autoregression

- A *neural network* is a class of models that make predictions $f(x)$ from an input (feature vector) $x \in \mathbb{R}^p$ of the form:

$$
\begin{aligned}
f_1(x) &= \rho(W_1 x + b_1) \\
f_\ell(x) &= \rho\Big(W_{\ell-1} f_{\ell-1}(x) + b_{\ell-1}\Big), \quad \ell = 2, \ldots, L \\
f(x) &= f_L(x)
\end{aligned}
$$

- Here each $W_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ is a matrix of weights that maps from the dimension $d_{\ell-1}$ of layer $\ell - 1$ to the dimension d_ℓ of layer ℓ. Note that $d_0 = p$, and $d_L = 1$ (for real-valued predictions)
- Each $b_\ell \in \mathbb{R}^{d_\ell}$ is a vector of intercepts (often called *biases* in the deep learning community). Generically, the parameters W_ℓ, b_ℓ are all learned by minimizing the sum of squared errors of predictions on the training data
- The function ρ is a nonlinear *activation function* that is interpreted as being applied componentwise, and user-chosen; common choices are $\rho(u) = u_+$ and $\rho(u) = 1/(1 + e^{-u})$
- Lastly, L here is the number of layers, also a design choice, and often called the *depth* of the network
- For time series, one of the simplest things that can be done with neural networks is just to form input features by taking lags of the given response variable. We can use both nonseasonal and seasonal lags (as in ARIMA). This gives rise to what we call a *neural network autoregressive* (NNAR) model
- What we described above is actually just a particular type of neural network architecture, and indeed, the simplest kind, called a *feedforward* neural network. Many other architectures are possible, and some more appropriate for time series data, such as the *long short-term memory* (LSTM) network, a type of recurrent neural network
- A popular time series forecaster based on LSTMs is called *DeepAR*, proposed by Salinas et al. (2020), from Amazon. Relative to all other methods you have learned thus far, DeepAR is quite complicated to describe precisely (and to train). However, like many deep learning methods, it can work very well in data-rich prediction problems that have a high signal-to-noise ratio
- Perhaps not surprisingly, some authors are now re-purposing transformers in order to turn them into time series forecasters. As deep learning continues to grow, we will continue to see spillover into time series forecasting

2 Calibration

-

3 Scoring

-

4 Ensembling

-

4.1 Untrained methods

-

4.2 Trained methods

-

References

Vassilis Assimakopoulos and Konstantinos Nikolopoulos. The theta model: A decomposition approach to forecasting. *International Journal of Forecasting*, 16(4):521–530, 2000.

Rob J. Hyndman and Baki Billah. Unmasking the theta method. *International Journal of Forecasting*, 19(2): 287–290, 2003.

David Salinas, Valentin Flunkert an Jan Gasthaus, and Tim Januschowski. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191, 2020.

Sean J. Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.