

Introduction to Time Series Modeling for Forecasting

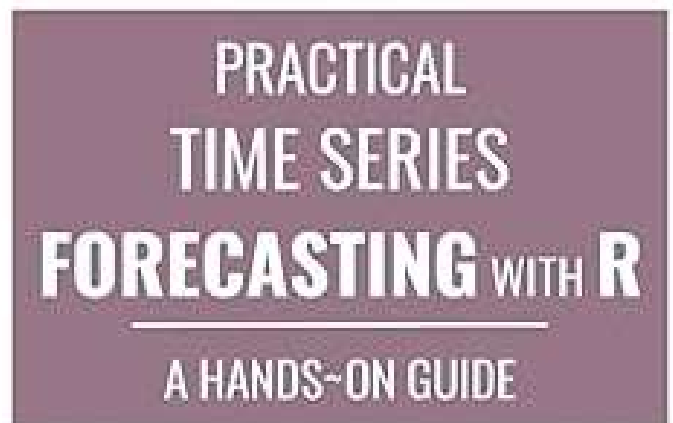
VectorByte Training 2024

Disclaimer and Credits

Most of the content in this presentation is adapted from *Practical Time Series Forecasting with R*, by Galit Shmueli and Julia Polak. Both the second and third editions were used. The example code was generated by Alicia Arneson using publicly available data and is a combination of code from the textbook and original code for time series forecasting.



THIRD EDITION
with R fable package



Galit Shmueli
Julia Polak

Forecasting Process

- Define Goal
- Get Data
- Explore/ Visualize
- Pre-Processing
- Partitioning
- Modeling
- Evaluation
- Implementation



Figure 1.1: Diagram of the forecasting process

Define Goal

- Think about what will be a useful output at the end of the process
 - Horizon and Update Regularity
 - Granularity
 - Accuracy
 - Implications/ Use
 - Automation
 - Descriptive vs. **Predictive**

Get Data



Forecasting challenges

Provide 'clean', pre-prepared data sets



Real life forecasting

Finding data can be messy and hard...

Explore and Visualize

Goals:

- Identify patterns
- Find errors
- Missing data
- Unequal spacing
- “Irrelevant” periods – is COVID sales data appropriate for forecasting tomorrow’s restaurant revenue?

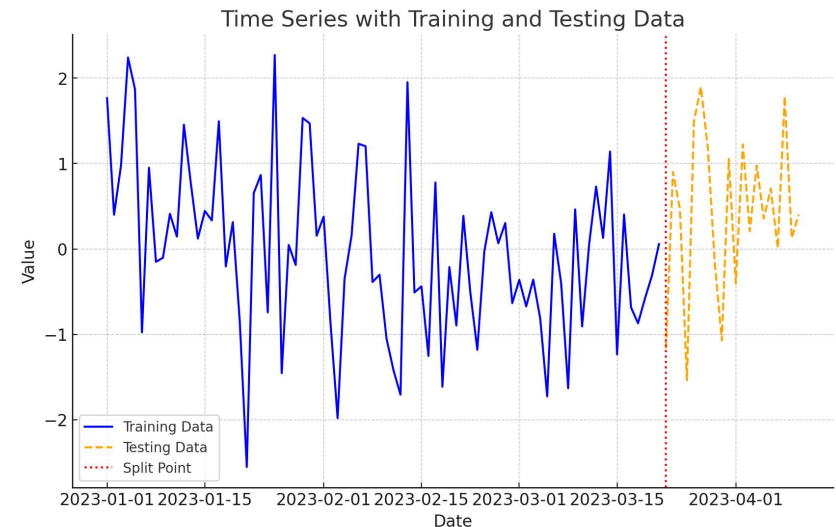
Pre-Process Data

- What needs doing will depend on the goal and the intended tool
- Unequal spacing – some approaches require equal spacing; imputation or averaging/ summing over a longer time scale sometimes possible; some models ok
- Extreme values – what do they mean? What is their effect?

Partition the series

NOT RANDOM

- Match the test forecast horizon to how far out you intend to forecast when possible
- Training data will be farther back in time than the validation data
- Sometimes in xsectional data – training, validation, and test; not usually the case in TS because does more harm than good to not validate the model on the most recent data



Apply Forecasting Model(s)

- AR
- ARMA
- ARIMA
- Regression Models
 - Linear
 - Logistic
 - GLMs
 - GAMs
- Machine Learning Models
 - Can't keep track of all of the kinds anymore
- GARCH
- Time-Varying Coefficient Models
- Ensemble Approaches
- Many More!

Evaluate and Compare Performance

- More visualizations!
 - Forecast overlaid on observed TS
 - TS plot of errors (is error getting worse as we go further out in time??)

Error = observation - forecast

- Accuracy Metrics

Common Accuracy Metrics

- MAE/MAD: $\frac{1}{v} \sum_{t=1}^v |e_t|$
 - Average absolute error – some people also use average error to see if the model tends to be biased negatively or positively.
- MAPE: $\frac{1}{v} \sum_{t=1}^v \left| \frac{e_t}{y_t} \right| \times 100$.
 - Comparing series with different scales
 - Biased toward under forecasting models
- RMSE: $\sqrt{\frac{1}{v} \sum_{t=1}^v e_t^2}$
 - Same units as the time series – nice for practical evaluation

There are several others that can be used, but these are the main ones.

Naïve Forecasts

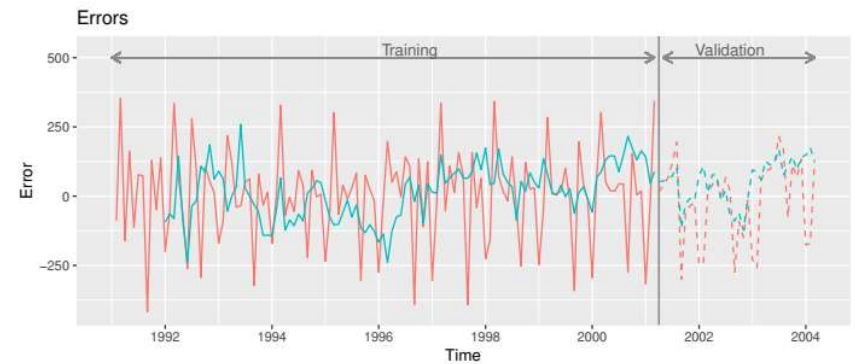
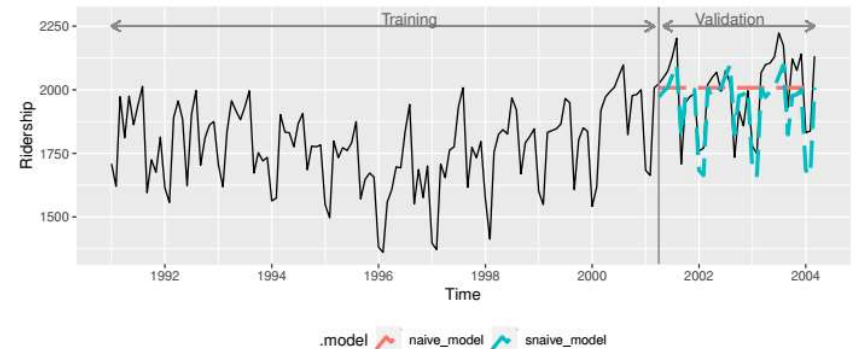
- We need a comparator to see if our model is worth applying at all

$$F_{t+k} = y_t$$

- Seasonally naïve forecasts

$$F_{t+k} = y_{t-M+k}$$

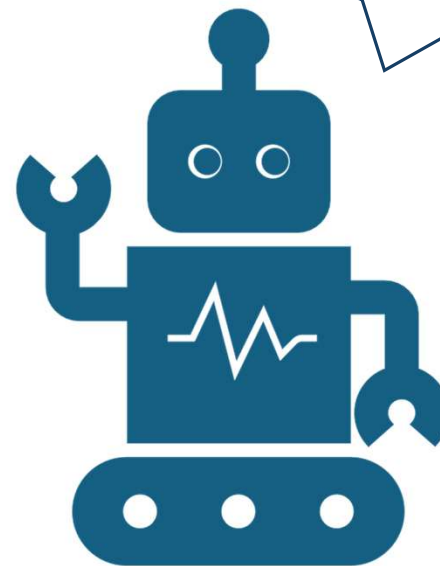
- From most recent comparable time point from a seasonal perspective



Implement Forecasting System

- A “System” can be a simple one time forecast or a more complex, automated system to do repeated forecasting over time
- Automation can also be a simple R loop or a more complex series of automatic downloads, model fitting, and uploads

Make sure you re-fit the model with all of the data before you make the real forecasts 😊

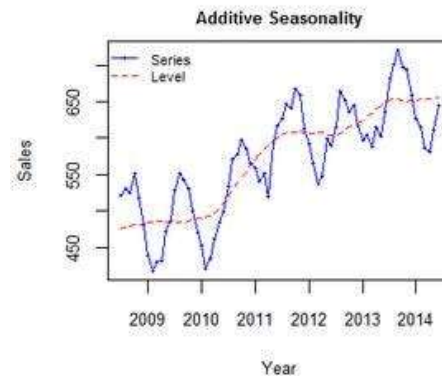


Time Series Models

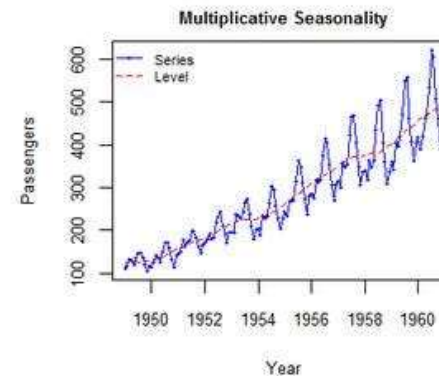
Review: Time Series Components

- Level
- Trend
- Seasonality
- Noise

Not directly
observable –
always observe
in conjunction
with noise



$$Y_t = \text{Level} + \text{Trend} + \text{Seasonality} + \text{Noise}$$



$$Y_t = \text{Level} \times \text{Trend} \times \text{Seasonality} \times \text{Noise}$$

Smoothing Methods

What is smoothing?

- Averaging over multiple time periods to reduce noise
 - Moving Average (MA) Models
 - Exponential Smoothing

Moving Average

- Averaging values across a window of consecutive periods
- Window - the number of consecutive values being averaged across each time

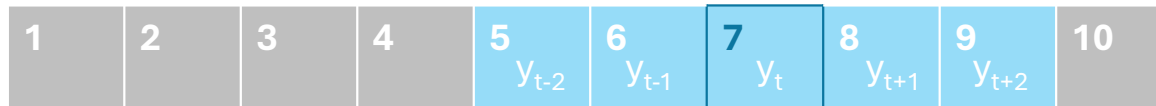
- Centered visualizing trends

$$MA_t = \left(y_{t-(w-1)/2} + \cdots + y_{t-1} + y_t + y_{t+1} + \cdots + y_{t+(w-1)/2} \right) / w$$

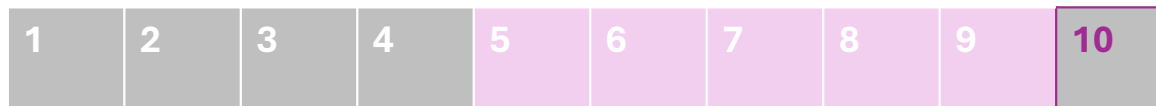
- Trailing forecasting

$$F_{t+k} = (y_t + y_{t-1} + \cdots + y_{t-w+1}) / w$$

Centered Moving Average



Trailing Moving Average



Window = 5

How to choose the right window

- Is the series seasonal?
 - Use the period over which the series is seasonal
- Performance
- Theoretical Appropriateness

Note:

Window = 1 creates a naïve forecast

Window = N uses all of the past observations

Limitations of MA Models

- Not ideal for series with strong trend or seasonality
 - Do not capture seasonality well
 - Lag behind on trends
- We can use techniques to de-trend and de-seasonalize time series before using MA models
 - Stay tuned!

Differencing

- Lag-1 differencing removes trend from a series
 - Does not assume a global trend
 - Have to do two rounds of lag-1 differencing if there is an exponential or quadratic trend present
- Lag-M differencing removes seasonality
 - M = the seasonal period
- We can add these components back in after we apply the MA model

Simple Exponential Smoothing

- Take a weighted average of all of the previous values

$$F_{t+1} = \alpha y_t + \alpha(1 - \alpha)y_{t-1} + \alpha(1 - \alpha)^2 y_{t-2} + \dots$$

- α is a smoothing constant between 0 and 1

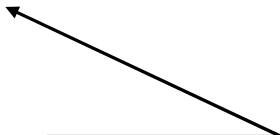
The exponential smoother can also be written as:

$$F_{t+1} = F_t + \alpha e_t$$

This frames it as an 'active learner'.

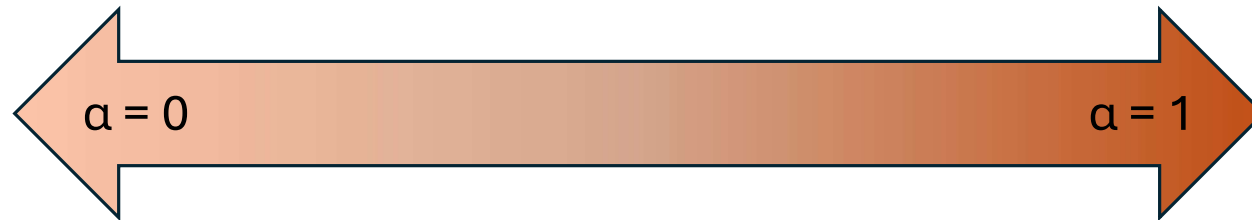
- Computationally cheaper
- Amount of correction depends on α

Exponentially decaying weights for earlier values in the series



Choosing a Smoothing Constant

- Rate of learning



- slow learning
- earlier values have considerable influence

- fast learning
- only the most recent values will be considered

- Default values often 0.1 - 0.2
 - Performance-based trial and error used to find optimal value

Limitations of Exponential Smoothing

- No trend or seasonality
 - Differencing is still okay to use

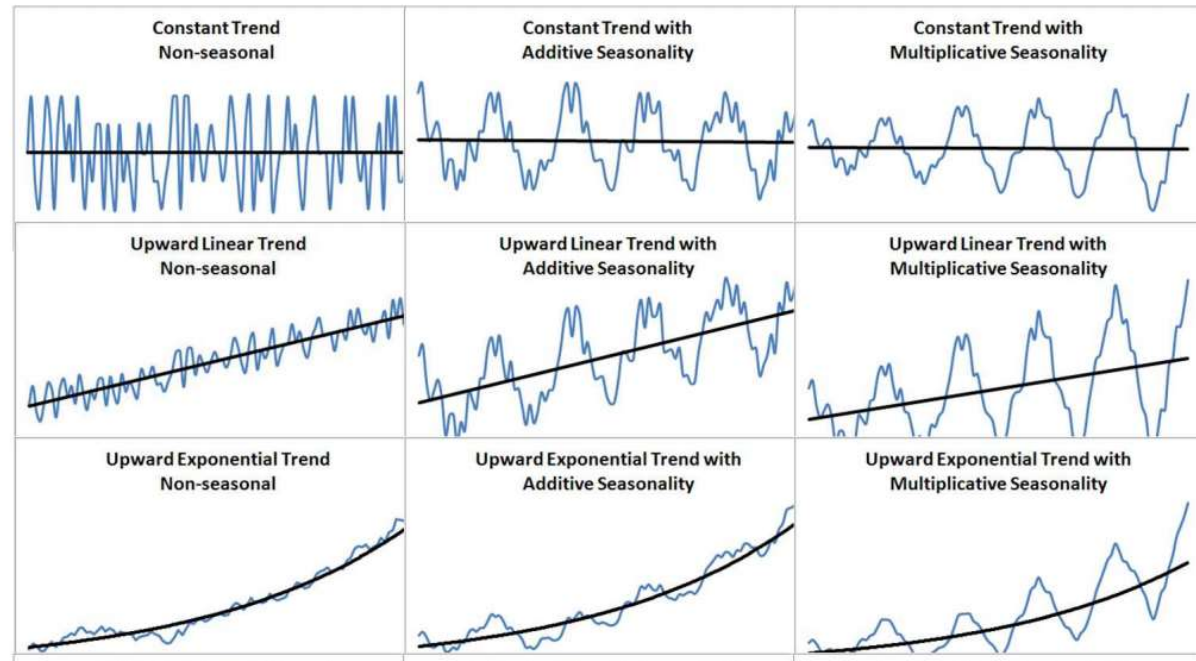
- One-step ahead forecasts only (applies to MA models too)
 - We can build automated generators (also R does this for you 😊)

Advanced Exponential Smoothing Methods

- We can account for additive or multiplicative seasonality, trend, and/ or error

The estimated trend is not global, it is able to change over time!

Can go further out than one-step ahead



Advanced Exponential Smoothing in R

`ETS(variable.name ~ error("A") + trend("A") + season("N"))`

A = Additive

M = Multiplicative

N = None/ Neither

2 error types X 3 trend types X 3 seasonality types = 18 models!

Technically there are 2 extra trend types – “Md” and “Ad” for dampening trends



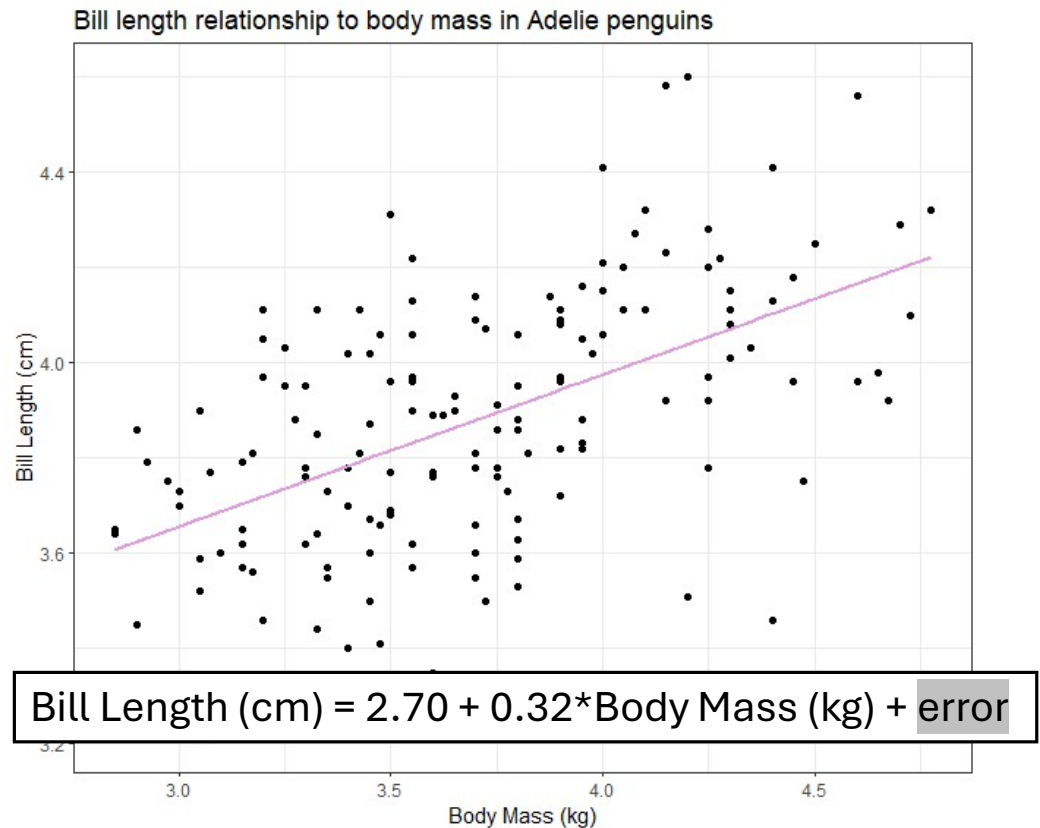
Example Time!

Regression Models

Linear Regression Recap

$$Y = B_0 + B_1 * X + \text{error}$$

How can we take advantage of this format to account for time series relevant components?



Adding Trend

- Must be a 'global' trend

Trend Type	Formula	R Syntax	
Linear	$Y = B0 + B1 * \text{time} + \text{error}$	<code>lm(Y ~ time)</code>	
Exponential	$\text{Log}(Y) = B0 + B1 * \text{time} + \text{error}$	<code>lm(log(Y) ~ time)</code>	Represents percentage growth
Polynomial	$Y = B0 + B1 * \text{time} + B2 * \text{time}^2 + \text{error}$	<code>lm(Y ~ time + time_sq)</code> <code>lm(Y ~ poly(time, 2, raw = T))</code> <code>lm(Y ~ time + I(time^2))</code>	Can be extended

Adding Seasonality

- Add categorical variable indicating the “season”
 - For multiplicative seasonality, fit to $\log(Y)$

Month	Ridership	Season
Jan-91	1709	Jan
Feb-91	1621	Feb
Mar-91	1973	Mar
Apr-91	1812	Apr

Mostly applies to ‘smooth’ seasonal patterns

- Sine and Cosine terms (Fourier terms) [Stolwijk et al., 1999](#)

$$f(t) = \beta_1 \times \sin\left(\frac{2\pi t}{T}\right) + \beta_2 \times \cos\left(\frac{2\pi t}{T}\right)$$

Period of seasonality (e.g. 12 for a year split into months)

Variable that represents where in the season we are (e.g. 1 for January)

The image features a black background with several abstract elements. A large, thin white circle is centered, with a thick light green ring around its bottom edge. To the left, a white zigzag line enters from the edge. In the top right, there is a light orange ring. In the bottom right, there is a large, solid light orange circle. To the right of the main circle, there are four parallel white diagonal lines. In the bottom left, there is a small light orange circle.

Example Time!

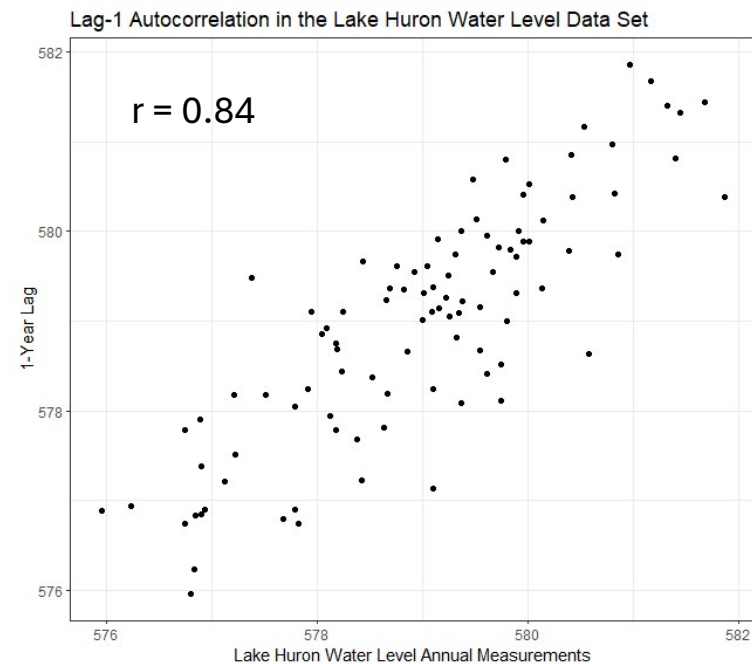
Autocorrelation

Capturing Autocorrelation

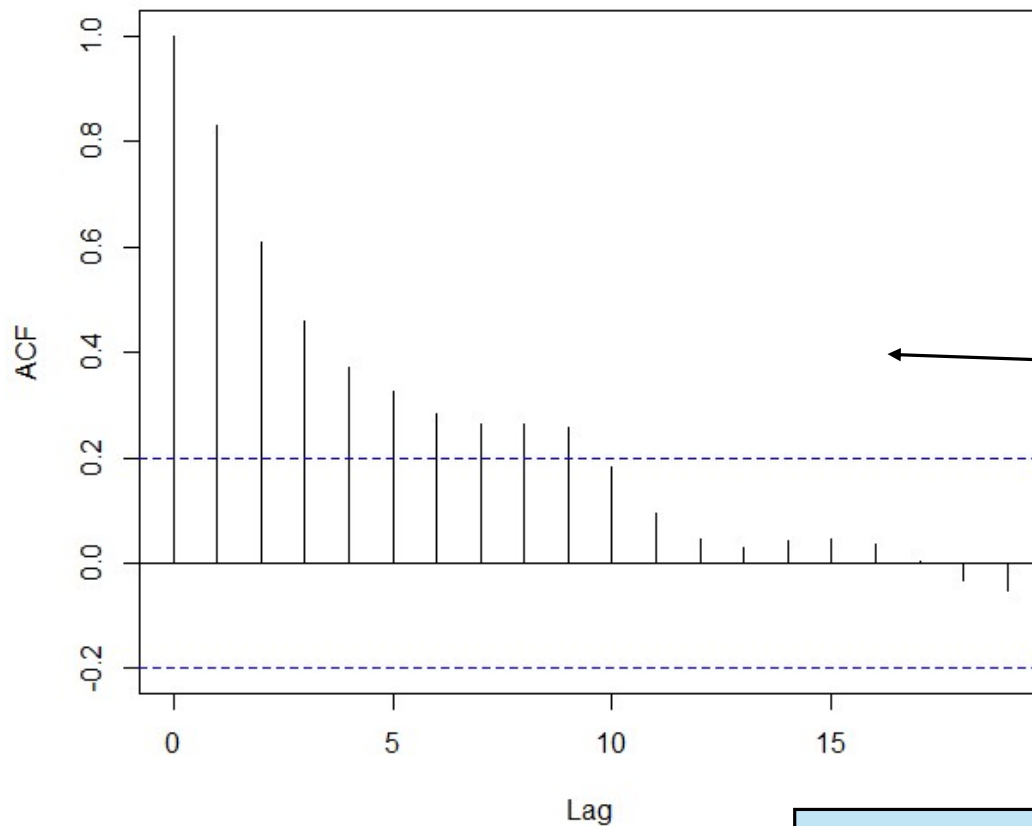
- Values in nearby periods tend to be correlated in time series data
- AR family of regression models

Computing Autocorrelation

Compute the correlation between the time series and lagged versions of itself.



Series LakeHuron



Strong autocorrelation at multiples larger than 1 suggests seasonality

- E.g. 12, 24, 36 months suggests annual pattern

Positive lag-1 autocorrelation (“stickiness”) means consecutive values tend to move in the same direction

- E.g. A linear trend

Negative lag-1 autocorrelation means there are swings in the time series

- E.g. High values immediately followed by low values

Note: We can evaluate autocorrelation in the RESIDUALS of a model too and adjust our forecasts to account for autocorrelated error.

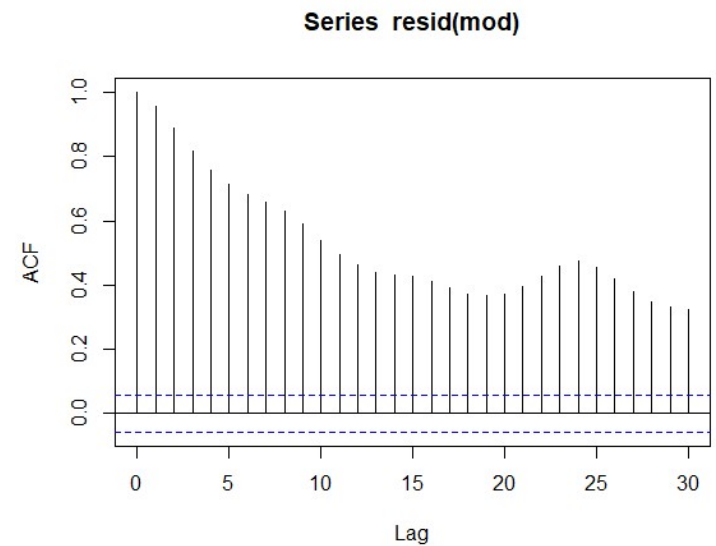
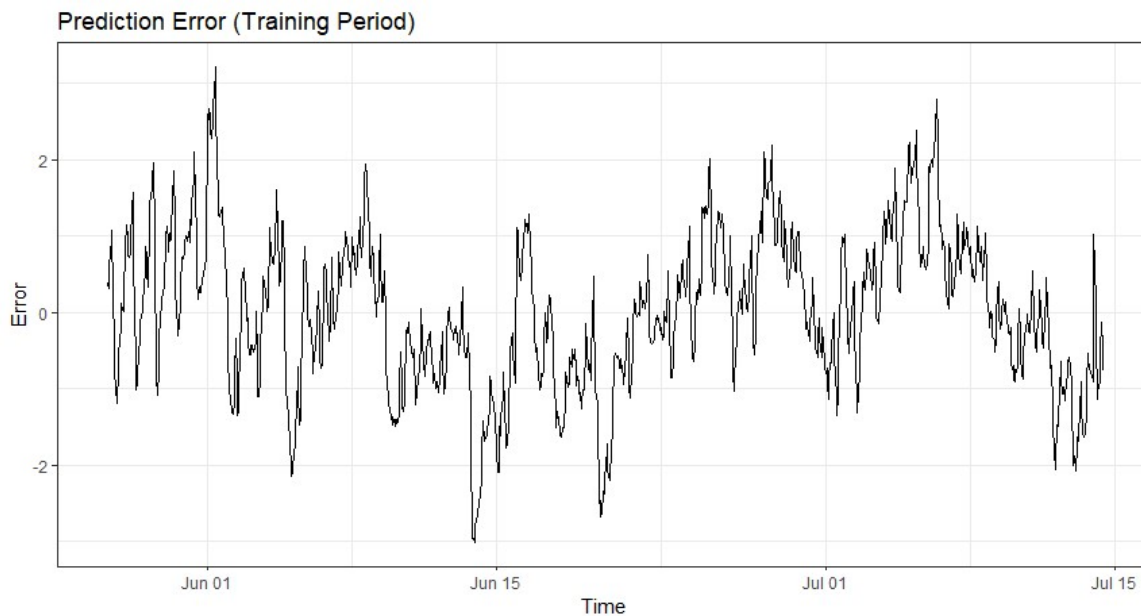
AR Models

The simplest way to capture autocorrelation in a regression model is to incorporate lagged outcomes as regressors.

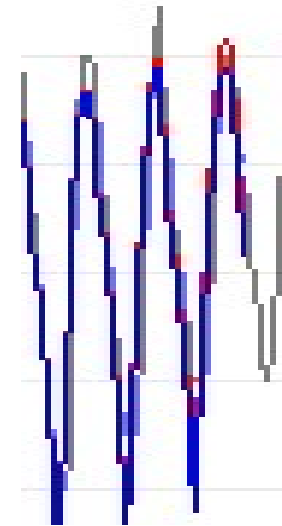
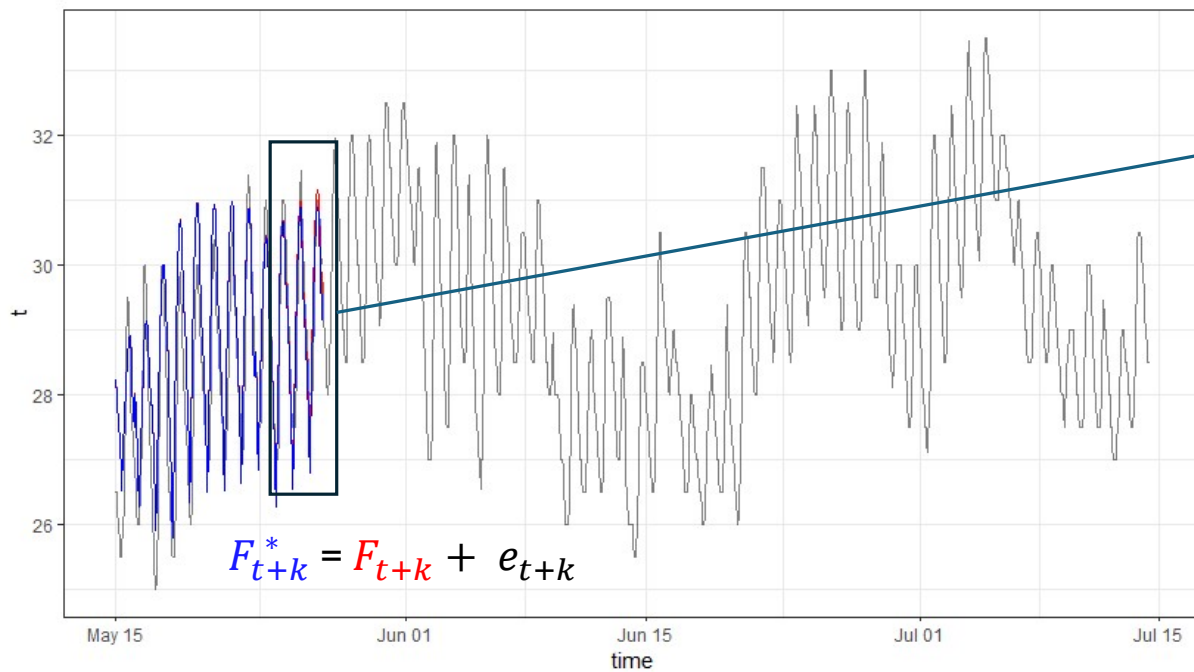
Model Equation	Order
$y_t = \beta_0 + \beta_1 y_{t-1} + \varepsilon_t$	AR(1)
$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \varepsilon_t$	AR(2)
$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \beta_3 y_{t-3} + \varepsilon_t$	AR(3)

AR as a Second-Layer Model

We can apply an AR model to the residuals of our model to forecast future error and adjust accordingly.



AR as a Second-Layer Model



Correction improves the model forecast for a short horizon.... Is there any other information that can be used?

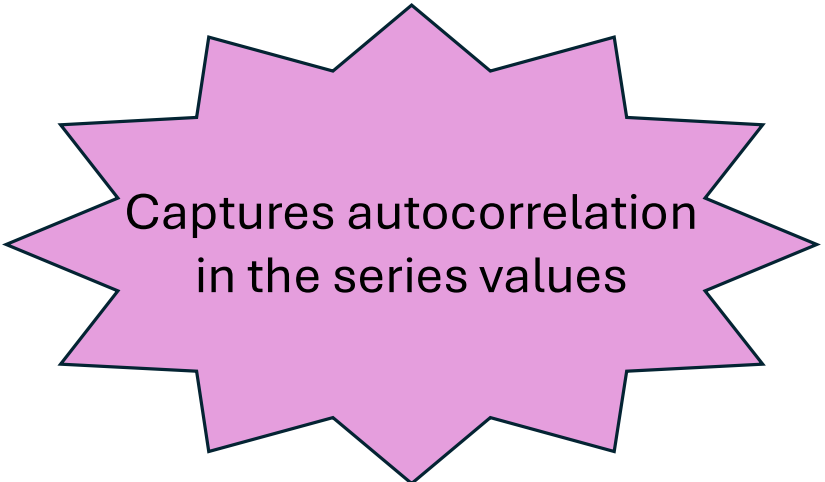
ARIMA Models

ARIMA Models

Auto-**R**egressive **I**ntegrated **M**oving **A**verage

AR(p) Model

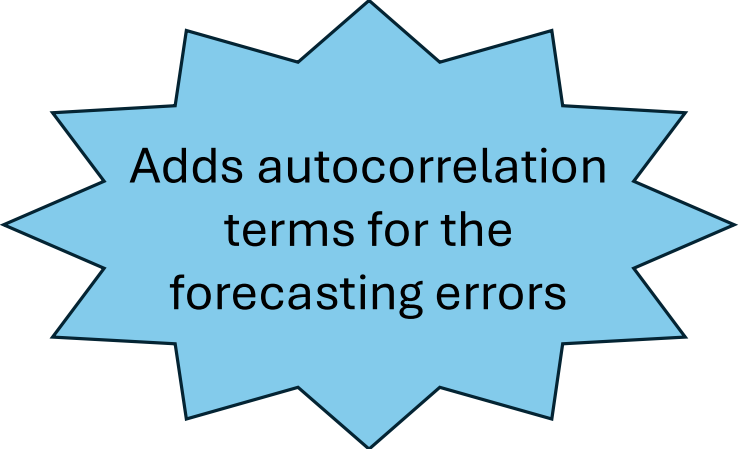
$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} \dots + \beta_p y_{t-p}$$



Captures autocorrelation
in the series values

ARIMA Models

Auto-Regressive Integrated Moving Average



Adds autocorrelation terms for the forecasting errors

ARMA(p,q) Model

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} \dots + \beta_p y_{t-p} +$$

$$\varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \dots + \theta_q \varepsilon_{t-q}$$

ARIMA Models

Auto-Regressive Integrated Moving Average

AR and ARMA models are not set up to account for trends or seasonality in the series, so we can INTEGRATE differencing into the ARMA model.

Basic ARIMA models handle trend by lag- d differencing.

- $d = 1$ accounts for a linear trend
- $d = 2$ accounts for a quadratic trend

What about seasonality?

Seasonal ARIMA Models

Seasonal ARIMA models require an extra set of parameters for seasonality left over in the model. We call them P, D, and Q.

- P – the number of seasonal autoregressive lags
- D – seasonal differencing order
- Q – the number of seasonal moving average lags



Example Time!

Including External Information

We often have extra information to use

- Outliers
- Special events
- Policy changes
- Correlated covariates

Outliers and Special Events

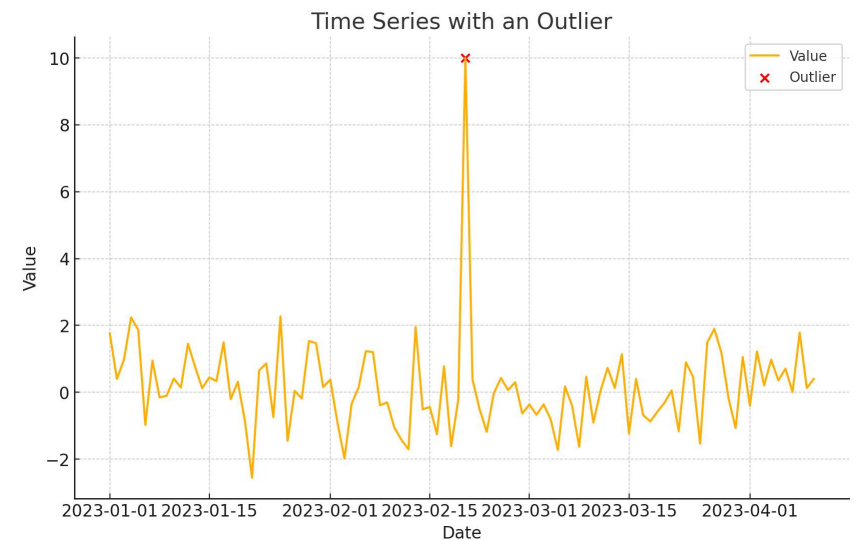
If outliers are present, we should test their effect on the forecast accuracy. If they negatively impact the predictions:

Option 1: Mark them using a dummy variable

- Only possible for regression models where extra predictors can be included

Option 2: Removal/ Replacement

- Regression – we can just remove
- Smoothing methods - we have to replace with an imputed value (like a centered moving average)



Thanks, ChatGPT!

Correlated Covariates

Considerations:

1. Lagged vs. Forecasted Predictors

- How good are the available forecasts?
- Maybe lagged is actually better than present
 - E.g. Plant growth

2. Data availability at the time of prediction

- In an ideal world, we will update the forecasts as new data becomes available.
 - How far in advance do we have access to data? Every week, every month, etc?? This will impact what lags or forecasts are available to us.



Time-Varying Coefficients

Just the basics

Time-Varying Coefficient Models

In some (many) cases, we might expect that the relationship between the predictors and the outcome may change over time. We can expand the usual parametric regression models to capture those changes via time-varying model coefficients. The coefficients can vary based on:

- deterministic processes
- stochastic processes
 - Stationary
 - Non-stationary
- state-space models

Time-varying coefficient models are considered “semi-parametric” or “locally parametric”.

The image features a black background with several abstract elements. A large, thin white circle is centered, with a thick light green ring around its bottom edge. To the left, a white zigzag line enters from the edge. In the top right, there is a light orange ring. In the bottom left, a small light orange circle is visible. In the bottom right, a large light orange semi-circle is partially shown. To the right of the main circle, there are four parallel white diagonal lines.

Example Time!

Neural Networks

Just some basics...

Neural Network Basics

Diagram of a Simple Artificial NN

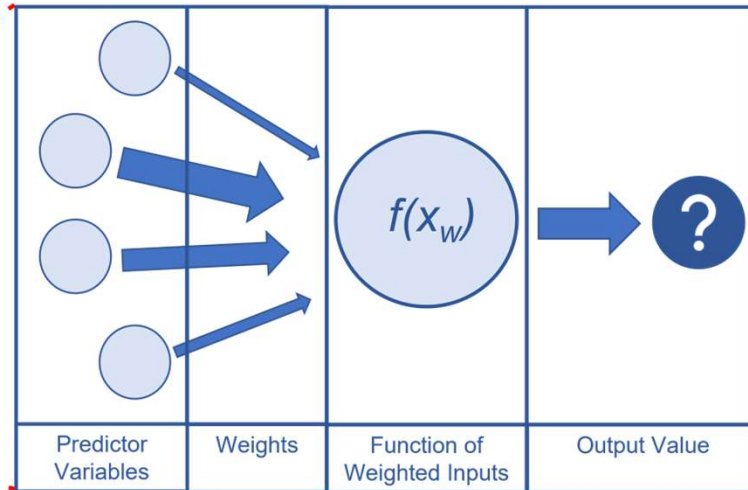
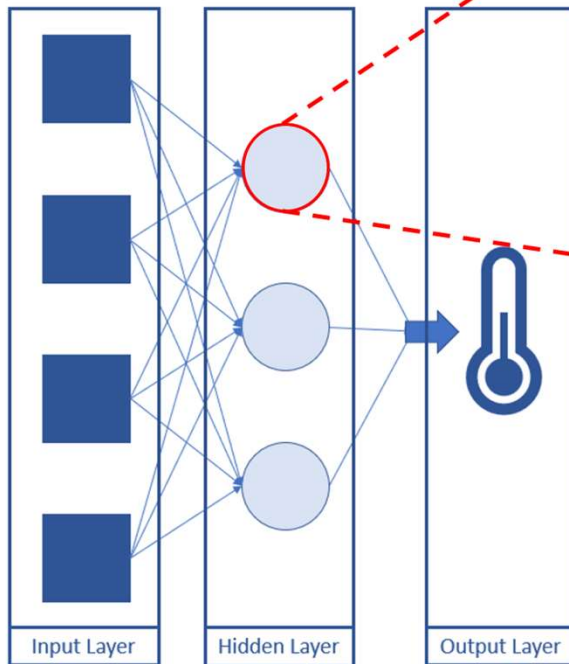


Diagram of a Simple ANN Node

Neural Networks for TS Forecasting

- Standard neural network models aren't so good at capturing time series data structures (like seasonality or autocorrelation).
 - We can include lagged response variables to help with that, BUT the pre-processing is.... Tedious....
 1. Make lagged response variables
 2. Center and scale them
 3. Make the same lags for the forecast period
 1. BUT WAIT some of those will need to be forecasted....
- Luckily there's a package for that!



Example Time!

Ensemble Models

Again just the basics...

Ensemble Models

- Ensemble forecasts are averages of several other models that are weighted in various ways
- A simple weighting scheme:

$$Weight_{Model A} = \frac{\left(\frac{1}{RMSE_{Model A}}\right)}{\sum_A^N 1/RMSE_{Model i}}$$

- The neural network function we used actually produces an ensemble of several neural network models.

Extra Tips

Binary Outcomes

- Sometimes we are more interested in predicting a yes/ no type of outcome
 - Will there be more than 20m of rainfall this season?
 - Will the species reach a critical threshold?
 - Will health resources be over-extended?
- Any model that can be applied to non-time series binary data can be applied here with the same adjustments we made in the previous sections

Evaluating Performance for Binary Outcomes

Sensitivity

- True positive rate

Specificity

- True negative rate

Area under the ROC Curve

- Quantifies overall classification performance (ability to discriminate)

		Predicted	
		Positive (Event)	Negative (No Event)
Actual	Positive (Event)	True Positives	False Negatives
	Negative (No Event)	False Positives	True Negatives

Evaluating Predictability

A predictable series is one where the future is able to be predicted **non-trivially** using past observations

$$\text{AR}(1): y_t = \beta_0 + \beta_1 y_{t-1} + \varepsilon_t$$

$$\text{Random Walk: } y_t = \beta_0 + y_{t-1} + \varepsilon_t$$

If β_1 is more than 2 SDs away from 1, we can call the series non-trivially predictable.

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	94.71257	32.23790	2.938	0.00415	**
lag1	0.83641	0.05568	15.022	< 2e-16	***

$$0.84 + 2*(0.06) = \mathbf{0.96}$$

0.96 < 1, so it is at least somewhat predictable!

Evaluating Predictability

Random Walk: $y_t - y_{t-1} = \beta_0 + \varepsilon_t$

Lag-1 Differenced Series

For this to be true, we should see no signs of autocorrelation in the lag-1 differenced series.

