Part II — Links and Description

Alexander Zhuk
Date: 11/23/2025

Deployed Shiny App URL

https://019ab2f0-c599-c119-a1cb-0ce4d5acc432.share.connect.posit.cloud/

Code Repository URL

https://github.com/AZhuk30/ShinyApp

App Description

I built Shiny app to compare forecasting models for Australian wine sales across six varietals from 1980-1994. Users can fit TSLM, ETS, and ARIMA models simultaneously, adjust training/validation splits, apply log transformations, and generate forecasts with prediction intervals. The interface lets you explore how different models handle trends and seasonality, which turned out to matter a lot more than I expected (validation errors varied by 50% depending on model choice). Everything runs interactively and includes diagnostic checks to validate model assumptions.

App Features

Core Functionality:

- **Multi-varietal selection:** Pick 2-6 wine types (Fortified, Red, Rose, Sparkling, Sweet White, Dry White) and compare their forecast performance side-by-side
- **Three model types with automatic specification:** TSLM, ETS (exponential smoothing with auto-selected components), and ARIMA (including seasonal components determined by the algorithm)
- **Model specification display:** Shows exactly what each model chose:
    - ETS forms like ETS(M,A,M) = multiplicative error, additive trend, multiplicative seasonality
    - Full ARIMA notation including seasonal orders, e.g., ARIMA(0,0,1)(1,1,1)[12] w/ drift showing (p,d,q) non-seasonal orders, (P,D,Q)[m] seasonal orders with period 12, plus drift term
    - TSLM formulas
- **Flexible train/test splits:** Slider from 60-95% lets you test model stability across different time periods. Red dashed line on plots shows exactly where the split happens
- **Adjustable forecast horizon:** Generate 6-24 month forecasts depending on your planning needs
- **Training vs validation accuracy:** See RMSE, MAE, and MAPE for both training fit and out-of-sample performance. This caught several cases where training looked fine but validation revealed overfitting

- **Interactive forecast plots:** Plotly charts with zoom/pan/hover, 80% and 95% prediction intervals shown as shaded bands, faceted by varietal and model for direct comparison
- **Clean interface:** Organized into Overview (data exploration), Model Results (specs and accuracy), Forecasts (projections), Diagnostics (assumption checks), and About tabs

Additional Features I Added:

- **Log transformation option:** Single checkbox applies log to all models and properly back-transforms forecasts. Helps with heteroscedasticity but turned out less critical than expected for this dataset
- **Seasonal pattern visualization:** Season plots overlay multiple years to show whether patterns stay consistent (they mostly do for wine sales)
- **Diagnostic suite:**
    - Residual scatter plots to check for remaining patterns or variance issues
    - ACF/PACF plots confirming residuals are white noise (no leftover autocorrelation)
    - Ljung-Box tests with pass/fail color coding (green = good, red = problems detected)
- **Summary statistics table:** Mean, SD, min/max, coefficient of variation per varietal to understand baseline patterns before modeling
- **Best model auto-identification:** Automatically ranks models by validation RMSE and highlights the winner for each varietal
- **Progress indicators:** Shows which model is currently fitting (ARIMA takes longer than TSLM)
- **Date range filter:** Focus analysis on specific subperiods if needed

Technical Details:

- Built with R Shiny + fpp3 package (tidyverse ecosystem for time series)
- Handles multiple time series efficiently using tsibble's key-based structure
- Auto-installs missing dependencies (like urca package needed for ARIMA unit root tests)
- Uses relative file paths so it deploys cleanly to shinyapps.io
- Error handling prevents crashes if you try weird combinations

Reproducibility:

- Packages required: shiny, tidyverse, fpp3, plotly, DT, bslib, urca
- Data: AustralianWines.csv (monthly sales 1980-1994, six varietals)
- Works on R 4.3+
- All settings exposed through the interface—no code modifications needed to reproduce any result