Georgia Treasury • Forecast Lab — Comprehensive User Guide

Version: 1.0   Date: November 13, 2025

https://github.com/WBG-ITS-Innovation/georgia-treasury-prototype

Table of Contents

Georgia Treasury • Forecast Lab — Comprehensive User Guide

Version: 1.0 | Date: November 13, 2025

This guide provides a complete reference for installing, operating, and understanding the Georgia Treasury Forecast Lab prototype. It includes step-by-step instructions, model and parameter definitions, validation guidance, and troubleshooting. The source repository lives at: https://github.com/WBG-ITS-Innovation/georgia-treasury-prototype.

1. Project Overview & Goals

The Forecast Lab enables Treasury analysts to evaluate multiple forecasting approaches—Statistical, Machine Learning, Deep Learning, and Quantile models—on Treasury time series (e.g., Revenues, Expenditure, State budget balance). The prototype is designed to be local-first, reproducible, and transparent, with standardized outputs and clear audit trails.

- Compare model families fairly using time-series cross-validation.
- Inspect Actual vs Operational baseline vs Predictions with intervals and residuals.
- Store each run in a self-contained folder (predictions, metrics, plots, configs, logs).
- Switch between an analyst-friendly UI (Streamlit) and a developer-friendly CLI (Python).

2. Architecture & Repository Layout

The system comprises a Streamlit frontend and a Python backend. The frontend launches backend runner scripts and monitors logs, while the backend executes pipelines, saves standardized outputs, and returns status to the UI.

Repository layout (key folders):

• backend/ — model pipelines, runners, preprocessing, and requirements

• frontend/ — Streamlit app (Overview, Data Pre-processing, Lab, Dashboard, History, Models)

• scripts/ — setup and run helpers (Windows & Unix)

• data_preprocessed/ — GENERATED at runtime by the pre-processor (ignored by Git)

• frontend/runs/ — GENERATED per experiment (ignored by Git)

• frontend/runs_uploads/ — GENERATED uploads (ignored by Git)

```
georgia-treasury-prototype/
├── backend/
│   ├── run_a_stat.py
│   ├── run_b_ml_univariate.py
│   ├── run_c_dl_univariate.py
│   ├── run_e_quantile_daily_univariate.py
│   ├── preprocess_data.py
│   ├── requirements.txt
│   └── data/processed/*.csv
├── frontend/
│   ├── Overview.py
│   ├── pages/
│   │   ├── 00_Data_Preprocessing.py
│   │   ├── 00_Lab.py
│   │   ├── 01_Dashboard.py
│   │   └── 02_History.py, 03_Models.py
│   ├── utils_frontend.py, backend_bridge.py, requirements.txt
│   ├── runs/, runs_uploads/, .tg_paths.json (GENERATED)
└── scripts/ (setup & run helpers)
```

3. Installation (Windows / macOS / Linux)

Prerequisites: Python 3.11+; optionally Git (otherwise use Download ZIP).

- Windows: install Python (check 'Add Python to PATH'), install Git if cloning.
- macOS/Linux: use system Python 3.11+ or install from python.org / pkg manager; install Git if cloning.

```
Windows (PowerShell):
  cd C:\Users\<your_name>\Documents
  git clone https://github.com/WBG-ITS-Innovation/georgia-treasury-prototype.git
  cd georgia-treasury-prototype
  scripts\setup_windows.bat
  scripts\run_app_windows.bat

macOS / Linux (Terminal):
  cd ~/Documents
  git clone https://github.com/WBG-ITS-Innovation/georgia-treasury-prototype.git
  cd georgia-treasury-prototype
  chmod +x scripts/*.sh
  ./scripts/setup_unix.sh
  ./scripts/run_app_unix.sh

Open http://localhost:8501 in your browser.
```

4. Data Pre-processing — From Excel/CSV to Standardized Daily Data

Use the '🧺 Data Pre-processing' page to convert raw Balance-by-Day files (Excel or CSV) into a standardized daily dataset consumed by all model pipelines.

- Input: Excel (Balance_by_Day_*.xlsx) or CSV (date, balance).
- Auto-detection of date and balance columns; override if ambiguous.
- Daily reindex; forward-fill balance (stock); compute net_flow = diff(balance).
- Calendar features: is_weekend, is_holiday, is_business_day, dow, month_end, quarter_end.
- Variants:
- • raw — minimal transformation.
- • clean_conservative — robust MAD clipping for outlier flows; reconstruct balance.
- • clean_treasury — non-business flows set to zero + 8-week same-weekday baseline of flows.
- Outputs saved under: data_preprocessed/<variant>/

```
Example (UI):
  1) Upload frontend/data/Balance_by_Day_2015-2025.xlsx
  2) Variant: raw
  3) Run preprocessing → Preview first rows → Download processed CSV
```

5. Lab — Running Experiments (Step by Step)

The Lab page lets you configure and launch experiments across model families, cadences, and horizons, with live logs and reproducible outputs.

- Choose data (upload CSV or select a preprocessed master).
- Select target (Revenues / Expenditure / State budget balance).
- Set cadence (Daily/Weekly/Monthly) and horizon (e.g., 1/6/20).
- Pick family (A/B/C/E), variant (Uni/Multi if available), and model(s).
- Select a run profile: Demo (fast), Balanced, Thorough.
- Optionally edit the JSON overrides (lags/windows/lookback, folds, min_train_years, etc.).
- Click Run → inspect logs → open results in Dashboard or History.

```
Run folder structure (example):
  frontend/runs/run_B_uni_Ridge_State_budget_balance_Daily_h1_2025xxxx_xxxx/
    backend_run.log
    outputs/
      predictions_long.csv
      metrics_long.csv
      leaderboard.csv
      plots/*.png
      artifacts/config.json
```

6. Validation Design & Cross-Validation Settings

Time-series cross-validation is implemented via sequential splits. The key controls are:

Use Balanced/Thorough profiles for credible evaluation. Validate robustness by checking metrics variance across folds and comparing to operational baselines.

- folds — number of CV splits; typical: 3–5.
- min_train_years — minimum training history for the first fold; typical: 3–5.
- demo_clip_months — optional truncation for very fast demo runs.
- Horizon (h) is respected within each fold; metrics are aggregated across folds.

7. Model Families & Hyperparameters (Deep Reference)

Below are detailed parameter references with suggested defaults and tuning tips. Start with defaults, then adjust based on the cadence, series volatility, and history length. Avoid overfitting by keeping the hypothesis space modest, using enough folds, and reviewing residual diagnostics.

8. Outputs, Metrics & Interpretation

Metric guidance:

• MAE — average absolute error; robust and interpretable.

• RMSE — penalizes large misses; check for outliers.

• sMAPE — scale-free % error; works across targets.

• R^2 — variance explained; be cautious on non-stationary series.

• MASE — relative to naive baseline; <1.0 means better than naive.

• Coverage — fraction of y_true within [y_lo, y_hi]; aim near target (e.g., 80% or 90%).

- predictions_long.csv — long format with date, target, model, horizon, y_true, y_pred, optional intervals (y_lo, y_hi).
- metrics_long.csv — MAE, RMSE, sMAPE, R^2, MASE, skill vs baseline, coverage (if intervals).
- leaderboard.csv — per-model ranking; primary metric configurable.
- plots/ — overlays, residuals, leaderboards; suitable for presentations.

- artifacts/config.json — JSON snapshot of all settings used in the run.

9. Dashboard & History — Reviewing Results

- Dashboard: overlay Actual vs Ops baseline vs Predictions; toggle Weekly/Monthly aggregation.
- Inspect residuals, error distributions, and interval coverage.
- History: newest-first run table with metadata, log tail, preview of CSVs, and ZIP downloads.

10. Backend CLI — Running Without the UI

Activate the backend virtual environment, set environment variables, and call the appropriate runner.

```
Windows (PowerShell) — ETS (A · Statistical):
  cd backend
  .\.venv\Scripts\Activate.ps1
  $env:TG_FAMILY="A_STAT"
  $env:TG_MODEL_FILTER="ETS"
  $env:TG_TARGET="Revenues"
  $env:TG_CADENCE="Monthly"
  $env:TG_HORIZON="6"
  $env:TG_DATA_PATH="..\backend\data\processed\master_daily_raw.csv"
  $env:TG_DATE_COL="date"
  $env:TG_PARAM_OVERRIDES='{"min_train_years":4}'
  $env:TG_OUT_ROOT="..\frontend\runs\cli_A_ETS_Revenues_Monthly_h6\outputs"
  python run_a_stat.py

Windows (PowerShell) — Ridge (B · ML):
  $env:TG_FAMILY="B_ML"
  $env:TG_MODEL_FILTER="Ridge"
  $env:TG_TARGET="State budget balance"
  $env:TG_CADENCE="Daily"
  $env:TG_HORIZON="1"

$env:TG_PARAM_OVERRIDES='{"lags_daily":[1,3,7,14],"windows_daily":[3,7,14],"folds":5}'
  $env:TG_OUT_ROOT="..\frontend\runs\cli_B_Ridge_SBB_Daily_h1\outputs"
  python run_b_ml_univariate.py

Windows (PowerShell) — LSTM (C · DL):
  $env:TG_FAMILY="C_DL"
  $env:TG_MODEL_FILTER="LSTM"
  $env:TG_TARGET="Revenues"
  $env:TG_CADENCE="Daily"
  $env:TG_HORIZON="6"
```

```
$env:TG_PARAM_OVERRIDES='{"lookback":48,"batch_size":128,"max_epochs":3}'
$env:TG_OUT_ROOT="..\frontend\runs\cli_C_LSTM_Revenues_Daily_h6\outputs"
python run_c_dl_univariate.py
```

11. Troubleshooting & Quality Checks

- Backend Python missing: run setup scripts; ensure frontend/.tg_paths.json points to backend/.venv.
- No predictions_long.csv: check backend_run.log inside the run folder for import/runtime errors.
- Excel read errors: specify Date/Balance columns explicitly in the pre-processor page.
- UnicodeEncodeError (Windows console): set PYTHONIOENCODING=utf-8 or run via the UI.
- ModuleNotFoundError (sklearn/torch/openpyxl): ensure you run backend .venv Python.
- Zero-line predictions: verify feature generation (lags/windows present, no all-zero columns), correct target selection, and that scaling/fit only uses train folds (avoid leakage).
- Overfitting warning signs: huge gap between train vs CV metrics; unstable residuals; poor coverage.
- Paths/CWD errors: ensure backend_dir exists and is the runner working directory; avoid special characters in paths.

12. Reproducibility, Governance & Security

- Every run captures config JSON, logs, and artifacts for auditability.
- Pin versions in backend/requirements.txt; avoid environment drift.
- Document data provenance and pre-processing variant (raw/conservative/treasury).
- Local-first by default; no external data is sent; comply with institutional security policies.

Appendix A. Parameter Tables — Statistical

Appendix B. Parameter Tables — Machine Learning

Appendix C. Parameter Tables — Deep Learning

Appendix D. Parameter Tables — Quantile

Appendix E. File Schemas & Glossary

Appendix A — Statistical Parameters

Model	Key Parameters	Suggested Defaults	Notes
Baselines (Naïve / Weekday / MA)	window (MA), weekday_weeks (weekday mean)	MA window: 7–28 (daily); 3–6 (weekly); 3–6 (monthly)	Use as sanity checks; Weekday mean needs >=8 weeks history for stable results.
ETS / Holt–Winters	trend ∈ {add, mul, none}; seasonal ∈ {add, mul, none}; seasonal_periods=m; damped_trend ∈ {T,F}	Monthly: m=12, trend=add, seasonal=add, damped_trend=T; Weekly: m=52; Daily: m=7 (weekly seasonality)	Multiplicative seasonality often good when variance grows with level; try additive if variance stable.
SARIMAX	order=(p,d,q); seasonal_order=(P,D,Q,m); exog	p,q ∈ [0..3], d ∈ [0..1]; P,Q ∈ [0..2], D ∈ [0..1]; m per cadence (12/52/7)	If trend present, use d/D minimally; add exog (holidays, calendar) judiciously to avoid leakage.
STL-ARIMA	STL seasonal/trend windows; ARIMA on remainder	m per cadence; robust=True; season_window tuned to history	Good when strong seasonality with level changes; resilient to outliers.

THETA	seasonal_periods; decomposition	m per cadence; simple config	Strong low-variance baseline with competitive accuracy on smooth series.

Appendix B — Machine Learning Parameters

Model	Key Parameters	Suggested Defaults	Notes
Ridge	alpha, fit_intercept, normalize/standardize	alpha ∈ {0.1,1,10}; standardize features	Robust linear baseline; avoid leakage when scaling (fit on train only).
Lasso	alpha, fit_intercept, standardize	alpha ∈ {0.001,0.01,0.1};	Sparsity can help feature selection; beware underfit on small windows.
ElasticNet	alpha, l1_ratio	alpha ∈ {0.01,0.1,1}; l1_ratio ∈ {0.2,0.5,0.8}	Balances Ridge/Lasso; good general-purpose linear model.
RandomForest / ExtraTrees	n_estimators, max_depth, min_samples_leaf, max_features	n_estimators=300–600; max_depth=None; min_samples_leaf=1–5	Nonlinear relationships; ExtraTrees faster & higher-variance; RF more stable.
HistGradientBoosting	learning_rate, max_depth, max_bins, l2_regularization	lr=0.05–0.2; max_depth=3–8; max_bins=64–255	Fast GBDT; strong default; monitor overfit on short samples.
LightGBM / XGBoost (optional)	num_leaves/max_depth, learning_rate, n_estimators, subsample, colsample	lr=0.05–0.2; depth=4–8; n_estimators=200–800	Only if installed; use early stopping if supported.
Common FE (all ML)	lags_{cad}, windows_{cad}, exog_top_k, folds, min_train_years	Daily lags=[1,3,7,14,21,28]; windows=[3,7,14,28]; folds=5; min_train_years=3–5	Calendar/holiday flags help; keep feature set compact to reduce leakage risk.

Appendix C — Deep Learning Parameters

Model	Key Parameters	Suggested Defaults	Notes
LSTM / GRU	lookback, hidden_size, num_layers, dropout, batch_size, max_epochs, device, valid_frac	lookback=48–90 (daily); hidden=64–128; layers=1–2; dropout=0–0.3; epochs=3–20	Good for long dependencies; use Demo mode first; scale inputs; set device=auto.
TCN	n_blocks, kernel_size, dilations, dropout	blocks=2–4; kernel=3–5; dilations=[1,2,4,8,...]	Fast & strong on many lags; stable training; tune receptive field ~lookback.
Transformer (encoder)	d_model, nhead, num_layers, ff_dim, dropout	d_model=64–128; nhead=4–8; layers=1–3; dropout=0–0.2	Powerful with enough data; more sensitive to config & compute.
MLP (window-based)	hidden_layers, activation, dropout	2–3 layers of 128–512; dropout=0–0.2	Strong baseline with engineered windows; cheap & reliable.
Common (DL intervals)	conformal_calib_frac (optional)	0.1–0.3	Calibrated P10/P50/P90 via conformal if enabled; validate coverage.

Appendix D — Quantile Parameters

Model	Key Parameters	Suggested Defaults	Notes
GBQuantile	quantiles, lags_{cad}, windows_{cad}, learning_rate, n_estimators	quantiles=[0.1,0.5,0.9]; lr=0.05–0.2; n_estimators=300–800	Pinball loss; P50 ~ median; P10/P90 for risk bands / cash buffers.

Appendix E — File Schemas

predictions_long.csv

- date (ISO), target (str), model (str), h (int), y_true (float), y_pred (float), y_lo (float), y_hi (float), fold (int), split (int)

metrics_long.csv

- metric (str: MAE/RMSE/sMAPE/R2/MASE/skill/coverage), value (float), by (model/horizon/split)

leaderboard.csv — columns: model, metric, value, rank, notes

Appendix F — Glossary

- Cadence: Forecast frequency (Daily/Weekly/Monthly).
- Horizon: Number of steps ahead to predict.
- Exogenous: Predictor not equal to the target variable.
- Conformal intervals: Data-driven prediction bands with empirical coverage.
- Ops baseline: Operational baseline used by Treasury for comparison.