A Solution for the M5 Forecasting Competition

Rakshitha Godahewa

Outline

M5 Competition Overview

- The fifth round of the famous M competitions.
- Accurately forecast a set of hierarchically organized time series representing the sales demand of 3049 products sold by Walmart.
  - Required to submit 30,490 point forecasts for the lowest level of the hierarchy (store-product combinations).
  - Prediction horizon of 28 days.
  - Validation phase: Allowing the teams to fine-tune the model performance, Test phase: Used to evaluate the final performance of the teams.

M5 Time Series Structure

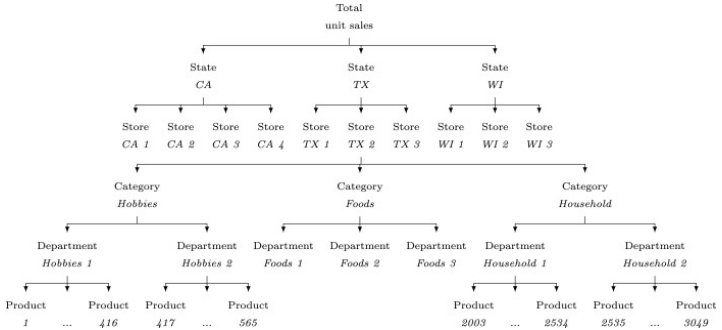

Figure 1: Time series hierarchy used in the M5 competition Makridakis et al. (2021)

Time Series Aggregation Structure

Level id	Level Description	Aggregation Level	Number of series
1	Unit sales of all products, aggregated for all stores/states	Total	1
2	Unit sales of all products, aggregated for each State	State	3
3	Unit sales of all products, aggregated for each store	Store	10
4	Unit sales of all products, aggregated for each category	Category	3
5	Unit sales of all products, aggregated for each department	Department	7
6	Unit sales of all products, aggregated for each State and category	State-Category	9
7	Unit sales of all products, aggregated for each State and department	State-Department	21
8	Unit sales of all products, aggregated for each store and category	Store-Category	30
9	Unit sales of all products, aggregated for each store and department	Store-Department	70
10	Unit sales of product x, aggregated for all stores/states	Product	3,049
11	Unit sales of product x, aggregated for each State	Product-State	9,147
12	Unit sales of product x, aggregated for each store	Product-Store	30,490
	Total		**42,840**

Figure 2: Aggregation levels of the M5 competition Makridakis et al. (2021)

Outline

Key Considerations

- Highly non-stationary historical data
- Irregular sales patterns
- Data intermittency
- Influence of exogenous variables and holiday effects
- Hierarchically organized large collection of time series

Solution

- A four-layered cross-learning-based retail demand forecasting framework.
- Achieved 17th position in the accuracy track (Top 1%)
- The implementation of the framework is available at: https://github.com/rakshitha123/M5
- Pre-processing layer, a Model prediction layer, a Post-processing layer, and an Ensembling layer
  - Pre-processing: time series grouping, normalisation
  - Model prediction: application of global models
  - Post-processing: data denormalisation
  - Ensembling: model combination strategy

Data Pre-processing Layer

- Time series grouping
  - Time series grouping at the department level
  - 70 clusters (store-department combinations)
- Data normalisation
  - Using the *meanscale* transformation strategy to account for sales scale differences
- Feature engineering
  - *day-of-week, day-of-month, month, is-working-day, is-weekend, snap, and events* as temporal features
  - 400 days of sales lags (longer memory)

Global Forecast Models (GFMs)

- Methods that estimate model parameters jointly from all available time series.
- A unified forecasting model that is built using a collection of time series.
  - Borrow similar behaviours and structures from other related time series.
  - Improves model generalizability.
  - Adequate data for model fitting.
  - Ability to exploit the cross-series information.
- Forecasting a large quantity of related time series.

LightGBM Model (Ke et al., 2017)

- A popular and computationally efficient machine learning algorithm.
- A variant of Gradient Boosting Models (GBM).
  - Combines many weak learners to come up with one strong learner.
  - The initial "weak" decision tree is "boosted" to produce more accurate forecasts.
  - Used the implementation available from the R package *lightgbm*.

Pooled Regression (PR) Model

- A global version of an Auto-Regression (AR) model of order 400 (lags of sales) along with the external variables.
- The term pooling indicates that one model is built using many series.
- Using the implementation available in the **glm** function from the R package *glmnet*.

Forecast Engine

- LightGBM and PR as the main prediction models.
  - Both trained globally across a collection of time series using exogenous variables.
  - 70 LightGBM models and 70 PR models.
  - The recursive strategy to generate multi step-ahead forecasts.
- Loss functions
  - Poisson loss as the loss function of LightGBM.
  - Re-weighted Least-Squares as the loss function of PR models.
- Hyperparameter selection and optimization
  - The last 28 days prior to the forecast horizon is used as the validation set.
  - A grid-based methodology to minimise the Weighted Root Mean Squared Scaled Error (WRMSSE).

Post-processing Layer

- Reversing the initial preprocessing steps.
  - Multiplying the forecasts by the mean of the respective series (for each GFM).
- The sales forecasts generated for each department id is collated (forecasts generated for each cluster).

Ensembling Layer

- In time series forecasting, ensemble models are mostly used in the form of forecast combinations.
- The model diversity to the forecast combination was introduced by employing both linear (PR) and nonlinear (LightGBM) global models in the forecast framework.
  - Compute the simple average of PR and LightGBM model forecasts.

Retail Demand Forecasting Framework

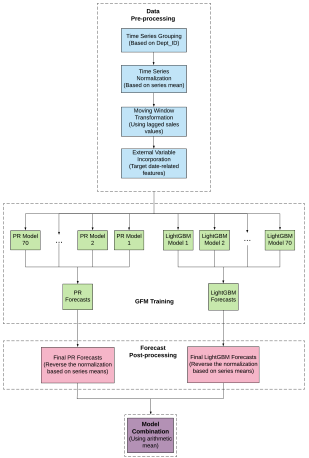

Figure 3: The overall summary of the proposed retail demand forecasting framework

Failed Experiments

- **Using local models such as Croston to obtain forecasts.**
  Can't learn cross-series patterns.
- **Running GFMs with a smaller number of lags.**
  The GFMs cannot capture yearly seasonality related patterns.
- **Running GFMs across the whole dataset at once.**
  Has not considered the similarity of series.
- **Running GFMs only with past lags.**
  Has not considered the holiday effects/exogenous features.
- **Using a single GFM to obtain final forecasts.**
  Ensembling can mitigate the weaknesses of each sub-model.

Outline

Limitations

- Only considers **simple averaging** to aggregate the sub-model forecasts.
- Only performs **recursive forecasting** to obtain the sub-model forecasts.
- Only considers **department-level clusters** for sub-model training.
- Execution time complexity (as the GFMs are trained with a **large number of lags and iterations**).

Possible Improvements

- Considering **hand-crafted features** such as rolling means, maximum and minimum series values for model training.
- Considering **promotional related features** and analyzing their effect on the model performance.
- Improving the running time complexity of the framework by using **parallel programming**.
- Analyzing the applicability of **transfer learning** to obtain forecasts for new series.
- Optimizing the forecasts across the **hierarchy**.

Possible Improvements Cont.

- Using a rolling-origin method for **hyperparameter optimization**.
- Considering both **recursive and direct methods** to obtain sub-model forecasts.
- Using **more sub-models** in the framework.
- Analyzing the effect of using **a machine learning model to aggregate** the sub-model forecasts.
- Obtaining series clusters using different methods, e.g.: **feature clustering, category-level clusters, store-level clusters, sales series types (smooth, erratic, intermittent, lumpy**) etc.

Thank you

References I

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T. (2017). Lightgbm: a highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 3149–3157, Red Hook, NY, USA. Curran Associates Inc.

Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2021). The M5 competition: Background, organization, and implementation. *Int. J. Forecast.*

Evaluation

- Error measures
  - Weighted root mean squared scaled error: WRMSSE

$$\text{RMSSE} = \sqrt{\frac{\frac{1}{h}\sum_{t=n+1}^{n+h}(y_t - \widehat{y}_t)^2}{\frac{1}{n-1}\sum_{t=2}^{n}(y_t - y_{t-1})^2}}$$

$$\text{WRMSSE} = \sum_{i=1}^{42840} w_i * RMSSE_i$$

y_t: actual observation at time t
$\widehat{y}_t$: Forecast at time t
h: Number of data points in the test set (forecast horizon)
n: Number of data points in the training set