



## **M5 Forecasting - Accuracy**

Estimate the unit sales of Walmart retail goods

Wan-Yen Lin, Chieh-Yun Yeh,  
Shao-Ning Yen, Yun-Chien Yen

MSBA 6421  
Dr. Yicheng Song



UNIVERSITY OF MINNESOTA  
Driven to Discover™

# Table of Content

1. Project Overview
  - 1.1. Project Definition & Business Context
  - 1.2. Project Goal
2. Data Exploration
  - 2.1. Tools Used
  - 2.2. Datasets Used
  - 2.3. Exploratory Data Analysis
3. Model Pipeline
  - 3.1. Format Transformation
  - 3.2. Feature Engineering
  - 3.3. Feature Importance
  - 3.4. Algorithm Selection
  - 3.5. Model Selection
  - 3.6. Model Evaluation
4. Project Conclusion
  - 4.1. Key Insights
  - 4.2. Forward Thinking
5. Appendix

# 1. Project Overview

## 1.1. Project Definition & Business Context

M5 forecasting is a Kaggle competition, which consists in estimating the unit sales of Walmart retail goods. The document sums up our insights and learnings from the competition and will include but is not limited to the following topics:

- What are the data characteristics gained from exploratory data analysis?
- Which features have larger predictive power?
- What is the pipeline of model training?
- What algorithm and model work the best for this task?
- How do we enhance model performance through different methods?
- How to ensure model robustness at the end?

Walmart is one of the world's biggest multinational retail corporations that operates a chain of hypermarkets, discount department stores, and grocery stores from the United States. In today's competitive business landscape, companies want to increase their revenues and have better strategic planning. That's why it is always a main topic on how to make accurate sales forecasting because it provides knowledge about the nature of future conditions, avoid waste and shortage, especially because precisely predicting sales is challenging. If we are able to successfully predict sales volume, Walmart could use the algorithm to achieve better operational efficiency and efficient resource allocations. For example, they can make decisions about inventory ordering as well as staffing scheduling. As a result, the M5 sales forecasting challenge was launched.

## 1.2. Project Goal

In this project, we aim at predicting sales for the future 28 days at the product level for Walmart using five years of historical sales data, price data, and calendar data. The success of the result is measured by WRMSSE (Weighted Root Mean Squared Scaled Error), which is a metric for prediction error evaluation. One of the reasons for adopting this metric is to balance the influence of products of high price and low volume with products of low price and high volume. Below is example of calculating WRMSSE for predictions of two products:

$$\text{WRMSSE}(A,B) = \text{RMSSE\_A} * 1/2 * \text{Sales\_A} / (\text{Sales\_A} + \text{Sales\_B}) + \text{RMSSE\_B} * 1/2 * \text{Sales\_B} / (\text{Sales\_A} + \text{Sales\_B}) + \text{RMSSE}(A,B) * 1/2 * \text{Sales}(A,B) / (\text{Sales\_A} + \text{Sales\_B})$$

## 2. Data Exploration

### 2.1. Tools Used

#### 2.1.1. Tool List

We use below codes to import all the packages and functions needed.

```
import lightgbm as lgb
import numpy as np
import pandas as pd
from mlforecast import Forecast
from numba import njit
from window_ops.expanding import expanding_mean
from window_ops.rolling import seasonal_rolling_mean, rolling_mean
```

#### 2.1.2. MLForecast Function

Particularly, we use MLForecast API to build machine learning based models. The forecasting steps of this function are as below, with contents and demo codes referenced from the github page of developers.

- Input
  - Data
    - Store time series in a pandas dataframe in long format
    - Each row represents an observation for a specific series and timestamp
  - Flow configuration
    - Set lag, lag-based transformations, data attributes, drop nulls or not, multithreading or not, static features or not
    - Lag and lag transformation: A lag is the value of the target variable shifted by a certain period, providing useful recent trends for more robust future sales predictions. For example, for any specific product in a given store, the 3-days lag value would be the sales made three days ago for this particular product and store. Different shifting time periods, average of several lags, or different levels of aggregated sales like at product level or at store level values can be considered
  - Model
    - Use either local mode scikit-learn compatible regressor or LightGBM / XGBoost for distributed training
  - Preprocessing
    - Store all time series in a grouped array
      - Values are stored in a single numpy 1D array and group boundaries are stored in another 1D array, which allows fast iteration and updates
    - Use Numba jitted functions for transformation
      - Accelerate transformation and allow computation multithreading

- Forecasting
    - Create an MLForecast object with the chosen models and features
    - The features can include lags, transformations on the lags and other date features
      - Lag
      - Lag transformations (such as rolling mean or expanding mean based on chosen window): defined as Numba jitted functions that transform an array
      - Other data features: can also define differences to apply to the series before fitting that will be restored when predicting
- 
- How to update lags
    1. Update features using latest values from a series
    2. Used updated features to predict next time stamp
    3. Update series values with predictions
    4. Run above three steps recursively

```
@njit
def rolling_mean_28(x):
    return rolling_mean(x, window_size=28)

models = [
    lgb.LGBMRegressor(),
    xgb.XGBRegressor(),
    RandomForestRegressor(random_state=0),
]

fcst = MLForecast(
    models=models,
    freq='D',
    lags=[7, 14],
    lag_transforms={
        1: [expanding_mean],
        7: [rolling_mean_28]
    },
    date_features=['dayofweek'],
    differences=[1],
)
```

- Training
  - Fit a model into training data
  - Some parameters are available to define:
    - Identify each series (id\_col)
    - Identify contained timestamps (time\_col)
    - Identify target series values (target\_col)
    - Identify static features (static\_features)

```
fcst.fit(series, id_col='index', time_col='ds', target_col='y', static_features=['static_0'])
```

- Predicting
  - Call predict(n) to get the forecasts for the next n days, which will automatically handle the updates required by the features using a recursive strategy

```
fcst.fit(series, id_col='index', time_col='ds', target_col='y', static_features=['static_0'])
```

## 2.2. Datasets Used

All data used for this analysis were provided by the Kaggle competition in the CSV format. There are three states (Wisconsin, Texas and California), and each state has several stores with different categories (Hobbies, Foods and Households) and items.

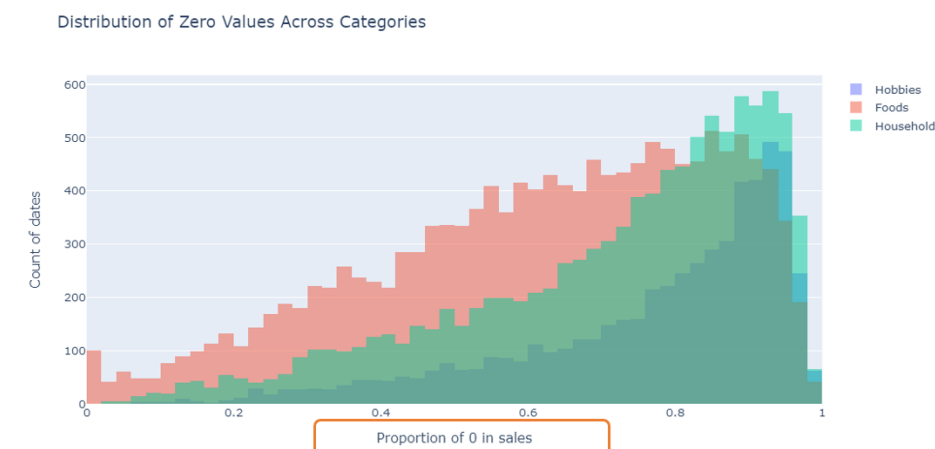
- Sales\_train\_validation.csv -- file contains sales for different levels including item, department, category, store and state from day1 to day1913.
- calendar.csv -- file is a mapping table for dates with special events or holidays.
- sell\_prices.csv -- file contains sell prices for each item in 10 stores in different weeks.
- sales\_train\_evaluation.csv -- file is for evaluating prediction performance which contains one month (28 days) more than sales\_train\_validation file.

## 2.3. Exploratory Data Analysis

To better understand the data, we do the data exploration on sales, calendar and price dataset. The following presents visualizations about summary statistics and time series analysis.

- Intermittent Sales

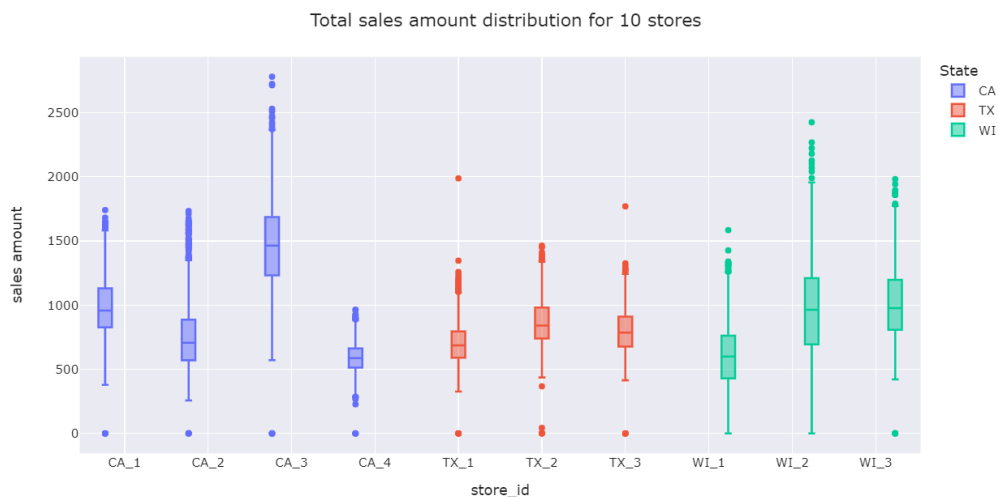
There are many zero values in time series indicating the intermittent demand for time series in three categories.



- Sales distribution across stores

10 stores have different mean and variance of daily total sales. For example, California has a high variance which includes the highest and the lowest sales within 10 stores. To be more specific, CA\_3 has 1472 sales (the highest). CA\_4 has 584 daily sales (the lowest). On the other hand, Texas and

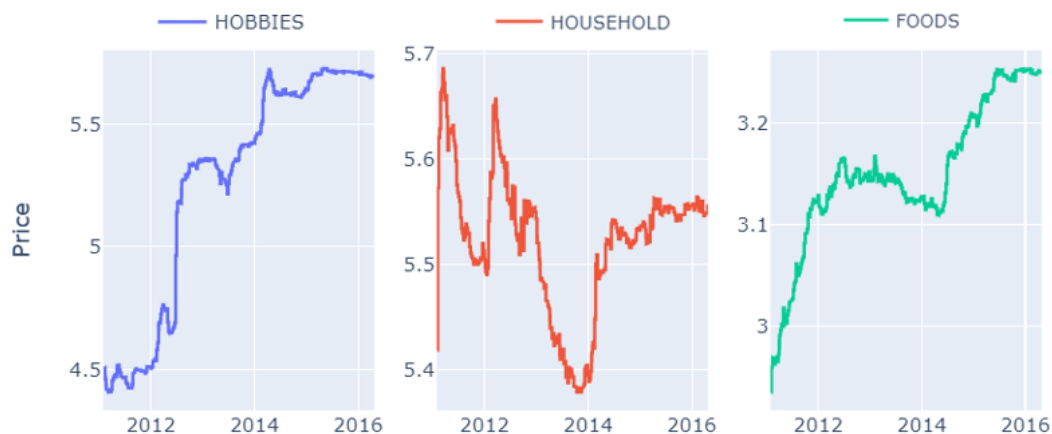
Wisconsin have more averagely distributed. Mean of daily total sales in TX\_1 to TX\_3 are about 800.



- Average Price Trend

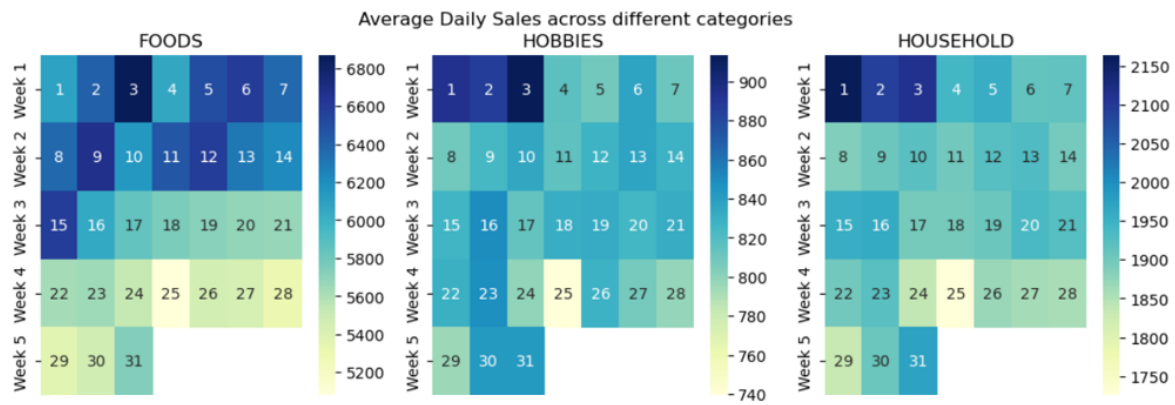
The average daily sales price trend for 3 categories have quite different patterns. Prices of three categories remained constant after 2015. Hobbies and Foods show an overall increasing trend, while Household decreased from 2012 to 2014.

Average Price in 3 categories



- Sales distribution across category

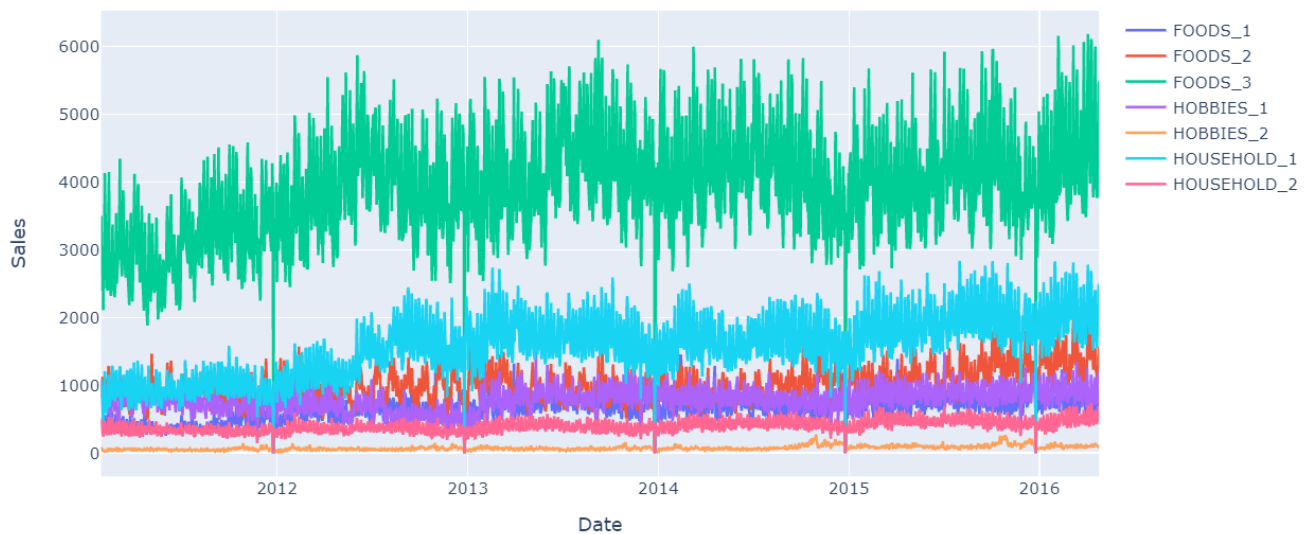
On each day of month, food items have more average daily sales in the first half of the month, especially on the third day of each month. Hobbies and Households have relatively higher sales in the first 3 days of the month.



- Sales distribution across departments

Overall, seven departments have increased trend over year. Foods\_3 has the highest total daily sales amount followed by the Hobbies.

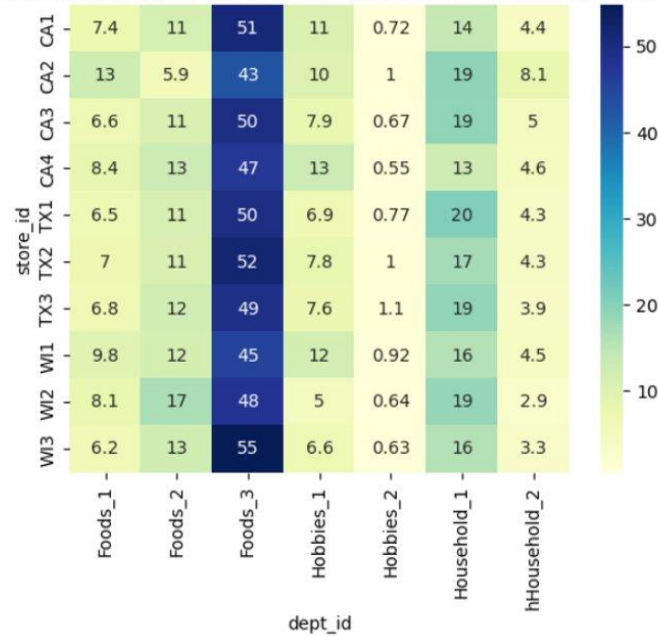
Total sales amount in 7 departments by date



For the distribution percentage in each store across 7 departments, store WI\_3 has the most skewed distribution that 55% sales come from food\_3 and about 7% is from hobbies(6.6%+0.63%). Also, store CA\_2 has the lowest sales in FOODS\_2 (5.9%) compared to other stores(e.g. 43% in Foods\_3 and 19% in Household\_1).

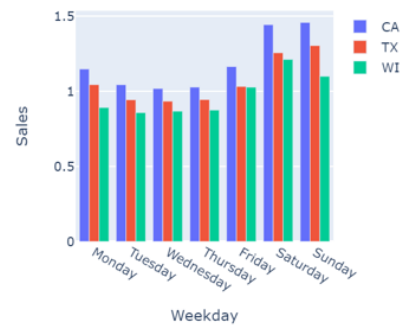
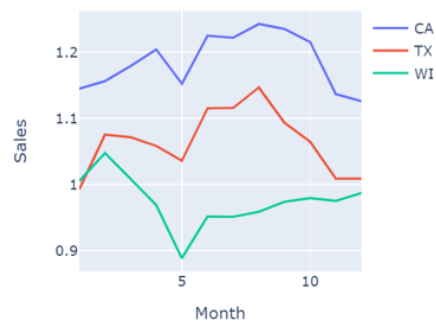


Sales distribution(%) in each store across different departments



Based on the below 3 line charts, we can see that yearly average sales in California has remained the largest, and sales in Wisconsin suppressed Texas since 2015. On a monthly basis, summer such as July and August have the most sales. For the weekday level, there is a significant increase on weekends.

Sales Trend in 3 states



## 3. Model Pipeline

### 3.1. Format Transformation

To fit data into the LGBM model, we need to transform the data into long format, where the index of the table is concatenation of item id and day.

### 3.2. Feature Engineering

- Calendar data features
  - Day
  - Week of year
  - Event\_name\_1
  - Event\_type\_1 - event types include Religious, Cultural, National, and Sporting
  - Event\_name\_2
  - Event\_type\_2
  - Sanp\_CA
  - Snap\_TX
  - Snap\_WI
- Sales Price of day1 to day1969
- Sales Units of dat1 to day1941
- Created features
  - Item price mean - Historical average item price
  - Item sales mean - Historical average item sales
  - Store price mean - Historical average store price
  - Store sales mean - Historical average store sales
  - Number of unique price of item in the history
  - Item release week
  - Pre-event - whether the day is 7 days before an event
  - Post-event - whether the day is 7 days after an event
- Features Extracted by MLForecast
  - Sales Lag
    - Sales shifted by a given period of time
    - Provide useful recent trends for more robust future sales predictions
    - Lag 7, 14, 21, 28, 35, 42, 49, 56 days
  - Lag Transformation
    - Rolling sales mean of 7, 14, 28, 60 days

### 3.3. Feature Importance

Understanding the context, complications, and primary inquiry is the first stage of problem analysis. The data needs to be processed before modeling can start since the data given to the algorithm is crucial for producing correct results. We undertake feature engineering and data transformations in the procedures outlined in the section below to extract the pertinent features with high predictive potential.

	CA	TX	WI
<b>Feature1</b>	Item id	Item id	Item id
<b>Feature2</b>	Item sales mean	Item price mean	Item price mean
<b>Feature3</b>	date	Sell price	Sell price
<b>Feature4</b>	Sell price	Date	Date
<b>Feature5</b>	Item price mean	Department	Department
<b>Feature6</b>	Department	Store	Item sales mean
<b>Feature7</b>	Store	Item sales mean	snap_WI
<b>Feature8</b>	Category	Event name	Product release week

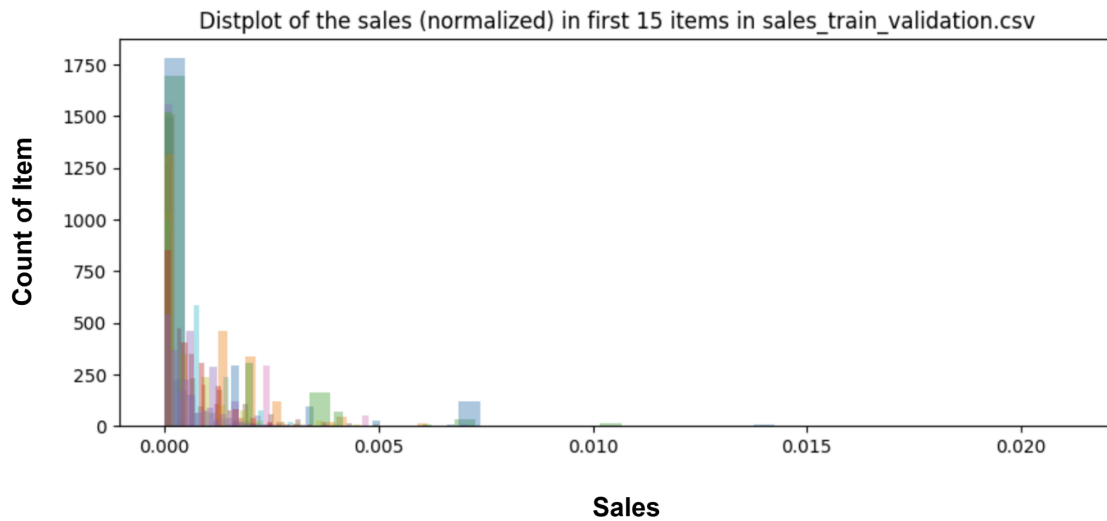
The top 6 features are similar in each state. For the top 1 important feature, item id, it is reasonable that it is the most critical feature in the predicted model since each product has unique characteristics that affect the sales. The mean price and sales of the item are also relevant features because these two do not vary a lot over time, making it reliable to predict future sales. Moreover, the department of the item is also critical since the sales in different departments vary significantly, making the feature affect predicted sales a lot.

The differences of important features between states are, in Wisconsin, the week that the product was released was more important than in other states. In California, item category is more relevant. As for Texas, the event name plays a more important role. Also, whether the nutrition supply was provided on the day(snap\_WI) stands out in the prediction model of Wisconsin.

### 3.4. Algorithm Selection

Light GBM is a gradient-boosting framework that uses a tree-based learning algorithm, which grows trees vertically (leaf-wise) while other algorithms grow trees horizontally (level-wise). It will choose the leaf with max delta loss to increase so that when developing the same leaf, a leaf-wise algorithm can reduce more loss than a level-wise algorithm.

We select the LightGBM algorithm in this case for several reasons. Due to its rapid speed, Light GBM is prefixed with "Light." Large amounts of data can be handled using Light GBM, which requires less memory. Most significantly, the data distribution is heavily right-skewed in the sales amount because we are forecasting intermittent demand, so we must expect lots of zero. We can effortlessly adopt the loss function of Tweedie, which is created to maximize the negative log-likelihood of the Tweedie distribution by setting the parameters in LightGBM.



Besides the LightGBM model, we utilize a high-level API function called: MLForecast, which could simplify our process to train efficiently and customize the lag feature on the product level to update our features based on the latest values. The following is our training cycle. We first transform our data into an item-based format, then define the model we use, LightGBM. After that, we customize the lag features in the MLForecast function. Last, we can train our model based on the previous settings and predict the best model.



### 3.5. Model Selection

We run the model by store and by state level with different features.

- Store level using original features
- State level using original features
- Store level using original features plus created features
- State level using original features plus created features
- Ensembling model that combined four models above

	By Store	By State
Original Columns	0.5637	0.5640
New Features	0.5663	0.5709
Ensemble Model	0.5633	

We then experiment by running the model by state, by the store with different features, and try the various hyperparameter combinations of `num_leaves`, `n_estimators`, and learning rate. In the above table, we list the top results to demonstrate. The performance metric we used here is WRMSSE (appendix 5.2), which means the lower the score is, the better. The store's performance on the left is slightly better than the state's. In addition, the ensemble model that combines the top 4 performances of all gives the best outcome of .5633 overall.


### 3.6. Model Evaluation

After careful hyper-parameters tuning and testing, we found that the store level models always perform better than the state level models. Moreover, using the ensemble method that combined model 1~4 above resulted in the best predicting performance. We believe that the ensemble model keeps a mixed portfolio across features and observations to reduce the variability and thus is more robust. Hence we chose the ensembled model as our final model, getting  $WRMSSE = 0.5633$ .

Featured Prediction Competition

## M5 Forecasting - Accuracy

Estimate the unit sales of Walmart retail goods

 University of Nicosia · 5,558 teams · 2 years ago

\$50,000

Prize Money

Overview

Data

Code

Discussion

Leaderboard

Rules

Team

Submissions

Late Submission

...


### Submissions

You selected 0 of 1 submissions to be evaluated for your final leaderboard score. Since you selected less than 1 submission, Kaggle auto-selected up to 1 submissions from among your public best-scoring unselected submissions for evaluation. The evaluated submission with the best Private Score is used for your final score.

Submissions evaluated for final score

AllSuccessfulSelectedErrors

Private Score ▾

Submission and Description	Private Score ⓘ	Public Score ⓘ	Selected
<div> <b>submission_ensem1.csv</b> Complete (after deadline) · 4d ago</div>	<b>0.56337</b>	<b>1.4663</b>	<input type="checkbox"/>

## 4. Project Conclusion

### 4.1. Key Insights

All in all, we earned superior performance and precious experiences in the M5 competition. First, we rank around 31 on the private leaderboard with WRMSSE 0.5633. Aside from low prediction error, our data preprocessing steps of our approach is relatively easy because you only need to specify data types per column and some useful features for time series problems are built-in. Last but not least, our model training time is highly efficient. It takes only around 90 seconds to predict approximately 300 products. This will allow companies to periodically re-train models for optimization or scale up models for more stores.

At the same time, we gained valuable insights in this competition:

- Select efficient machine learning methods for sales forecasting: since past sales is a good indicator of future sales, sales forecasting will generally be treated as time series problem thus need good model structure to deal with loaded time series computations and ensure constant update with time
- Take advantage of lag features as much as possible: use lags features to capture trends and seasonality as need but remember considering complexity additionally when there is any external or internal change
- Leverage value of cross-learning: take information originating from different aggregation level, different models, or different features to get final prediction for better model robustness
- Feature engineering is important when building flexible machine learning model: more good features can ensure the quality and relevance models receive, and we can adopt selection techniques to filter needed ones when evaluation

### 4.2. Forward Thinking

In order to gain profitability, retailers like Walmart can make use of our techniques to gather knowledge about future sales and allocate their resources accordingly. Here, we have some more recommendations in terms of model deployment, optimization, and maintenance.

- Pay attention to data leakage in future engineering process and make sure all features are still available
- Incorporate new reliable data into model to keep most up-to-date trends and advance model performance
- Monitor model to make sure its effectiveness does not degrade with time and re-train models whenever it is below acceptable performance threshold

- Consider making predictions using different granularity like in department level to gain diverse information about sales drivers
- Reinforce direct, latent, explanatory or exogenous features such as competitor sales, weather, or oil selling prices into models might help improvements in forecasting uncertainty
- Create models for different forecasting periods depending on business needs because staffing and ordering might need to be finalized months or weeks ahead of time
- Changes in the competitors sales, economic factors, or shifting consumer tastes need to be monitor and integrate into models when needed because they might influence forecasting much

## 5. Appendix

### 5.1. Hypothesis

Based on the data given some of the factors that may affect sales are:

- Day- Customers shopping time and spending mostly depends on the weekend. Many customers may like to shop only at weekends.
- Special Events/Holidays: Depending on the events and holidays customers purchasing behavior may change. For holidays like Easter, food sales may go up and for sporting events like Superbowl finals Household item sales may go up.
- Product Price: The sales are affected the most by the product price. Most customers will check the price tag before making the final purchase.
- Product Category: The type of product greatly affects sales. For instance, products in the household like TV will have fewer sales when compared with sales of food products.
- Location: The location also plays an important role in sales. In states like California, the customers might buy products they want irrespective

### 5.2. WRMSSE

The accuracy of the point forecasts will be evaluated using the Root Mean Squared Scaled Error (RMSSE), which is a variant of the well-known Mean Absolute Scaled Error (MASE). The measure is calculated for each series as follows:

$$RMSSE = \sqrt{\frac{1}{h} \frac{\sum_{t=n+1}^{n+h} (Y_t - \hat{Y}_t)^2}{\frac{1}{n-1} \sum_{t=2}^n (Y_t - Y_{t-1})^2}},$$

where  $Y_t$  is the actual future value of the examined time series at point  $t$ ,  $\hat{Y}_t$  the generated forecast,  $n$  the length of the training sample (number of historical observations), and  $h$  the forecasting horizon.

After estimating the RMSSE for all the 42,840 time series of the competition, the participating methods will be ranked using the Weighted RMSSE (WRMSSE), as described latter in this Guide, using the following formula:

$$WRMSSE = \sum_{i=1}^{42840} (W_{2i} * \sqrt{\frac{\sum (y - y')^2}{W_1}})$$

where  $w_i$  is the weight of the  $i$ th series of the competition. A lower WRMSSE score is better. Note that the weight of each series will be computed based on the last 28 observations of the training sample of

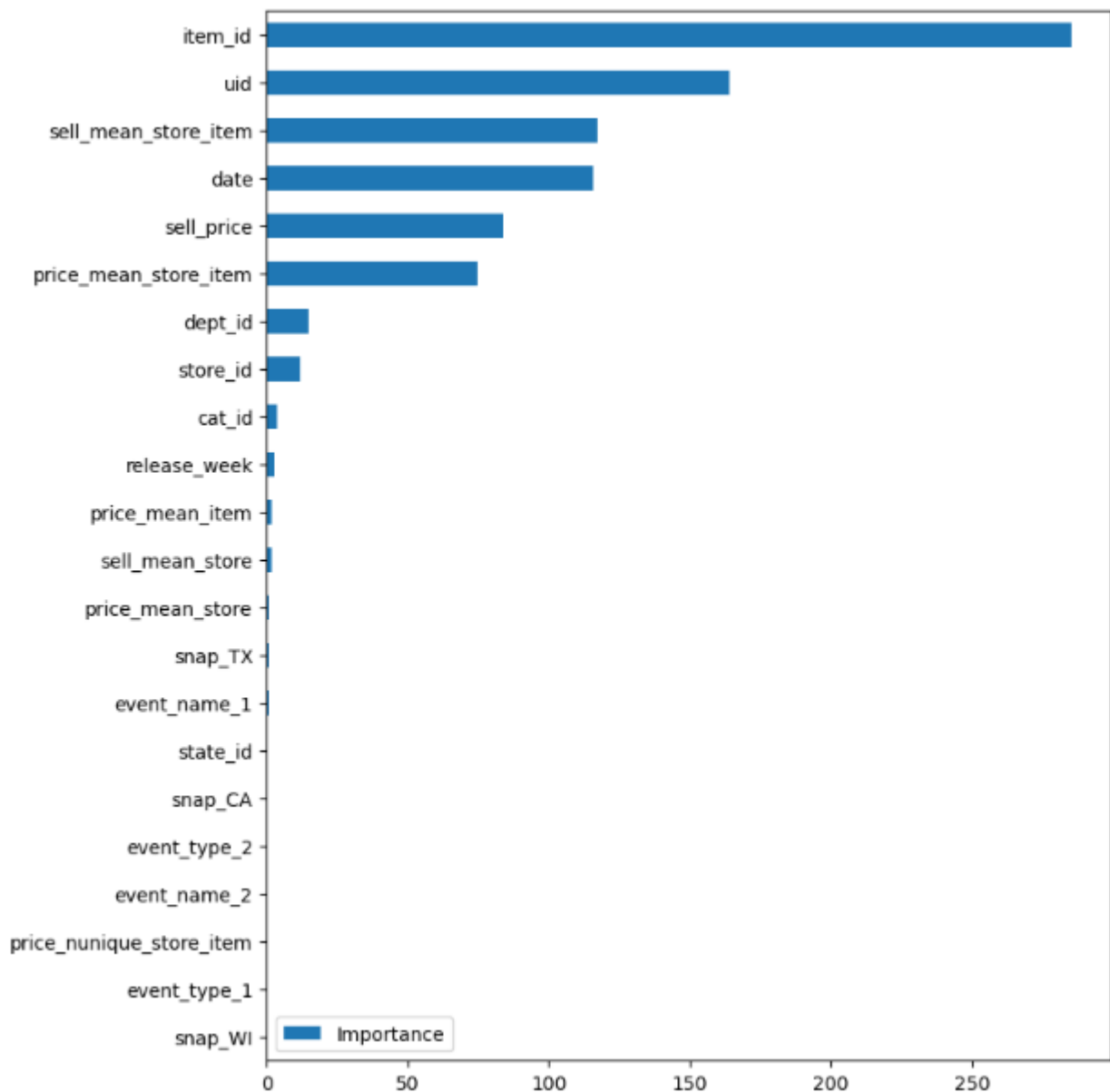


the dataset, i.e., the cumulative actual dollar sales that each series displayed in that particular period (sum of units sold multiplied by their respective price).

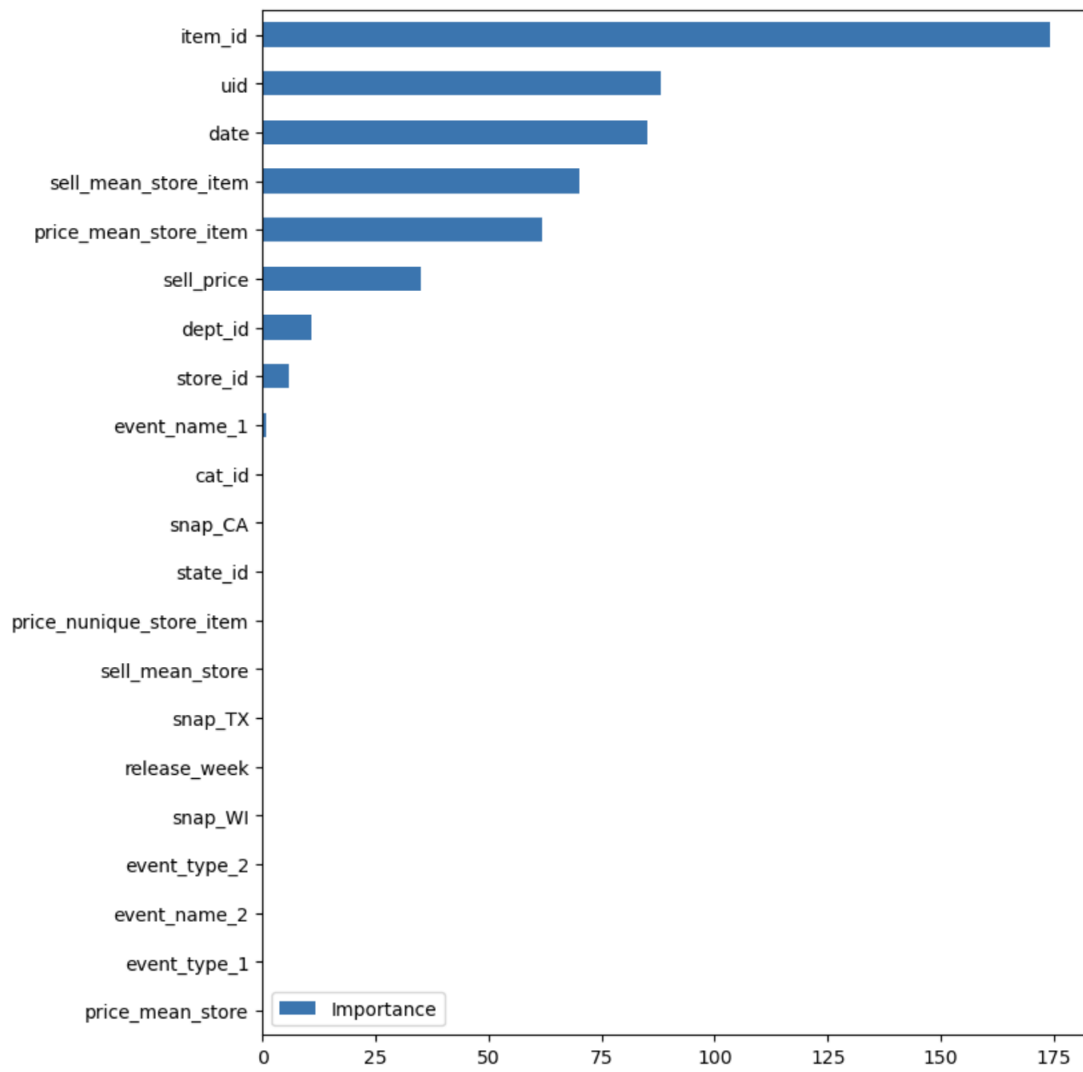
### 5.3. Feature Importance

From the graphs below, we can see that different features matter for various stores. However, some of the feature importance figures do not indicate the performance of our new features. Therefore, we apply the feature importance results from the by-state outcome.

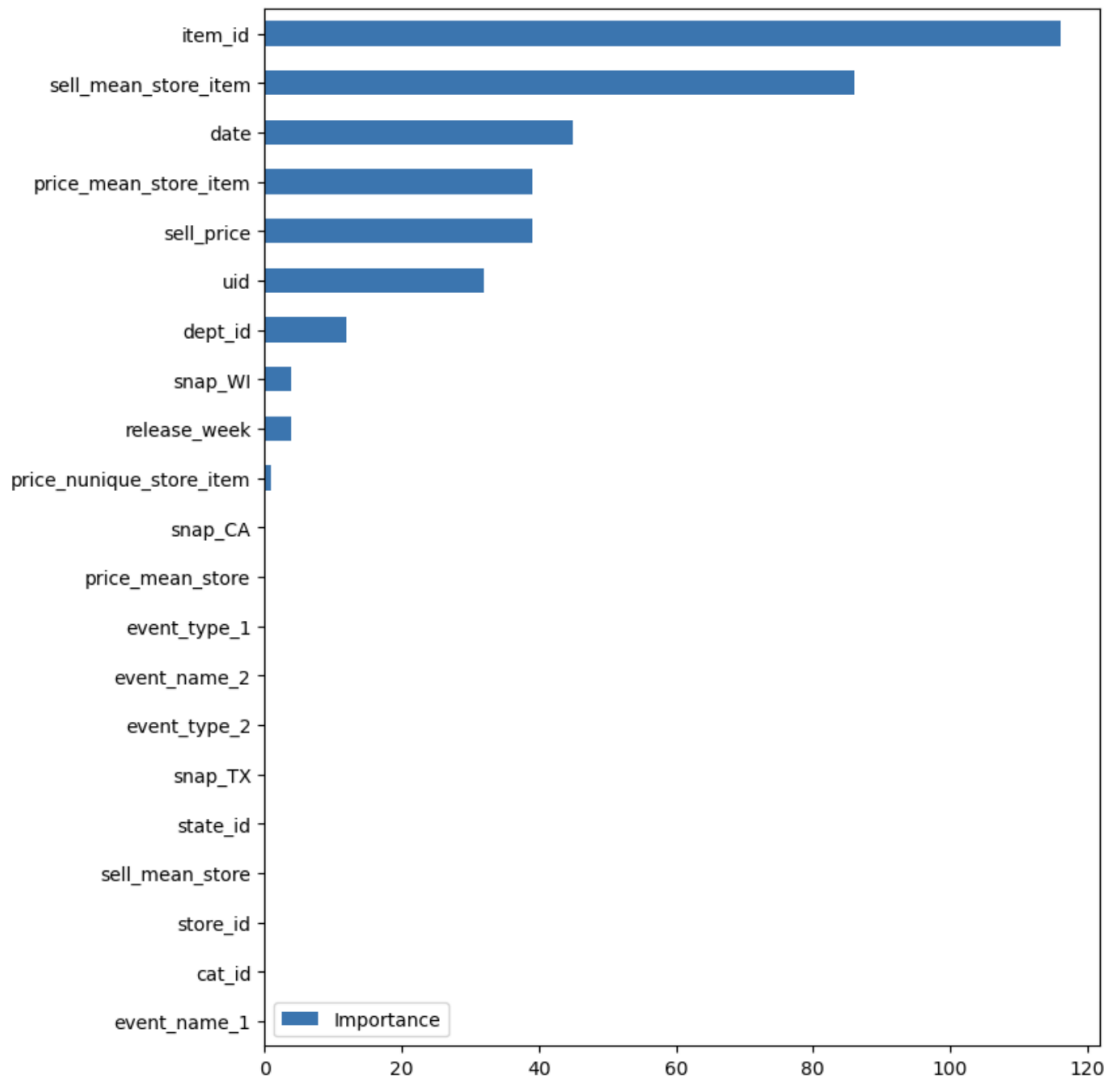
- By State
  - CA



- TX

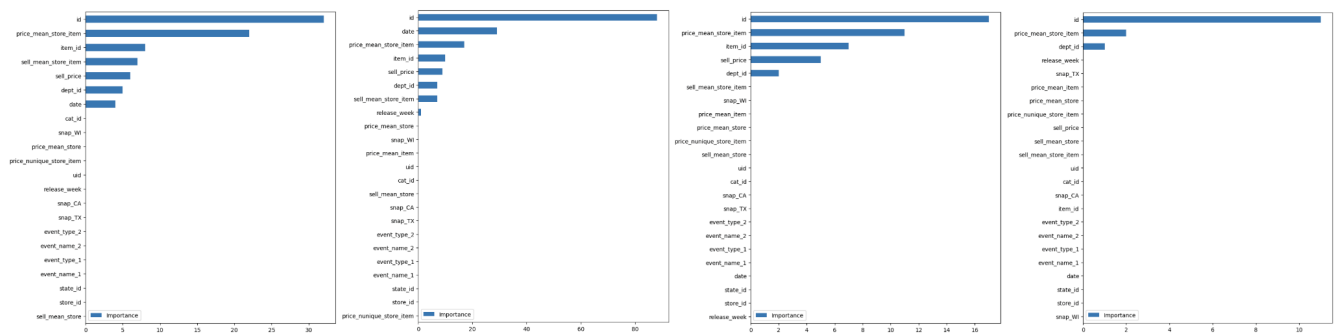


○ WI

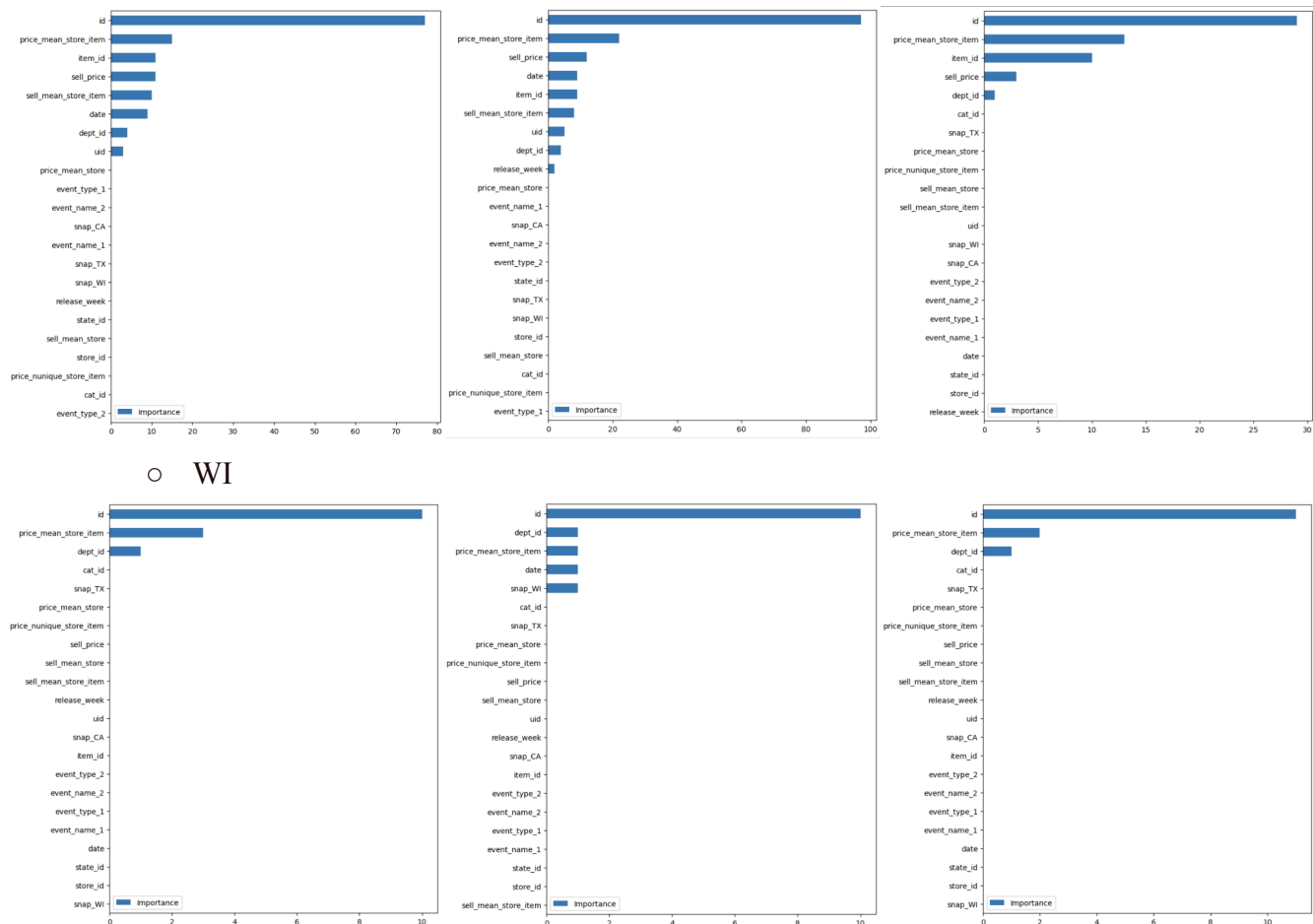


- By store

- CA



- TX



## 5.4. Other References

### 5.4.1. Exploratory Data Analysis

- M5 Competition : EDA + Models  
<https://www.kaggle.com/code/tarunpaparaju/m5-competition-eda-models#EDA->
- M5 : EDA + Basic Forecasting Techniques + Croston  
<https://www.kaggle.com/code/arpitsolanki14/m5-eda-basic-forecasting-techniques-croston>
- EDA and Stores Clustering - Walmart  
<https://www.kaggle.com/code/chanakyavivekkapoor/eda-and-stores-clustering-walmart>
- M5 Forecasting Exhaustive EDA Beginner  
<https://www.kaggle.com/code/anirbansen3027/m5-forecasting-exhaustive-eda-beginner>
- M5: Forecast and EDA Tutorial  
<https://www.kaggle.com/code/tawejsmh/m5-forecast-and-eda-tutorial>
- EDA of Walmart Sales Forecasting  
<https://www.kaggle.com/code/yehyachali/of-walmart-sales-forecasting/notebook>

### 5.4.2. Feature Engineering

- GitHub: Jace-Yang/walmart-sales-forecasting  
<https://github.com/Jace-Yang/walmart-sales-forecasting/blob/master/main.ipynb>

### 5.4.3. Modeling

- M5 Forecasting Kaggle competition: adjusting coefficient seems to be unpredictable due to stochastic cannibalization effect  
<https://lightgbm.readthedocs.io/en/latest/Parameters.html>
  - GitHub: Nixtla/mlforecast  
<https://github.com/Nixtla/mlforecast>
  - m5-mlforecast-eval  
<https://www.kaggle.com/code/lemuz90/m5-mlforecast-eval>
  - (158755)M5 Forecasting  
<https://www.kaggle.com/code/allenljw/158755-m5-forecasting/notebook?scriptVersionId=34274924>
  - YouTube Meetup  
[https://drive.google.com/drive/folders/11\\_PoRJVN6rguFf4EoNkNR3VVfW-WvMn\\_?usp=sharing](https://drive.google.com/drive/folders/11_PoRJVN6rguFf4EoNkNR3VVfW-WvMn_?usp=sharing)
  - M5 Forecasting: LightGBM  
<https://www.kaggle.com/code/bobiboba/m5-forecasting-lightgbm/notebook>
- 5.4.4. Other Topics
- Sales forecasting in retail: what we learned from the M5 competition  
<https://www.artefact.com/blog/sales-forecasting-in-retail-what-we-learned-from-the-m5-competition-published-in-medium-tech-blog/>
  - Cross Validation in Time Series  
<https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4>
  - M Forecasting Competitions  
<https://github.com/Mcompetitions/M5-methods>