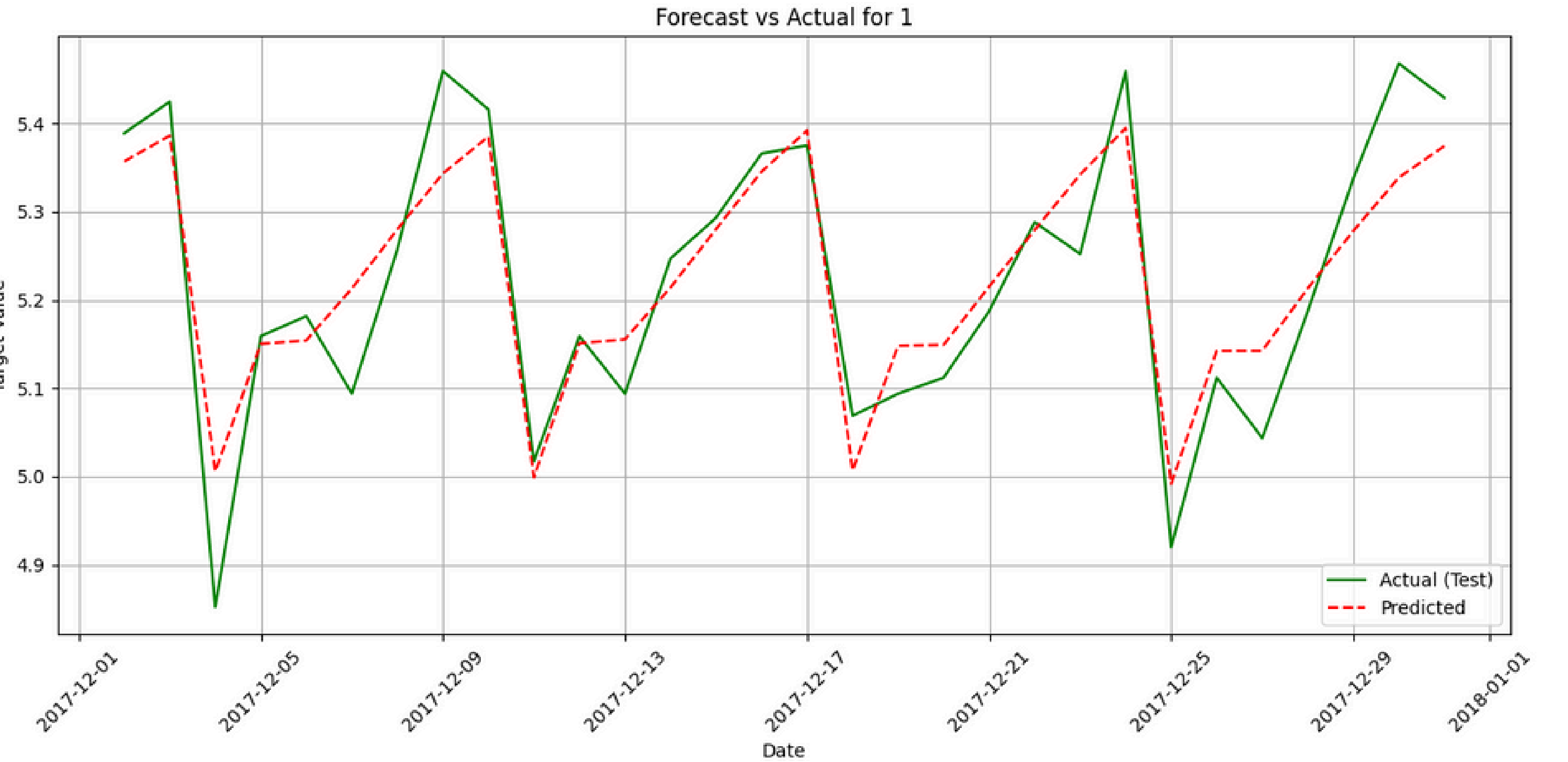
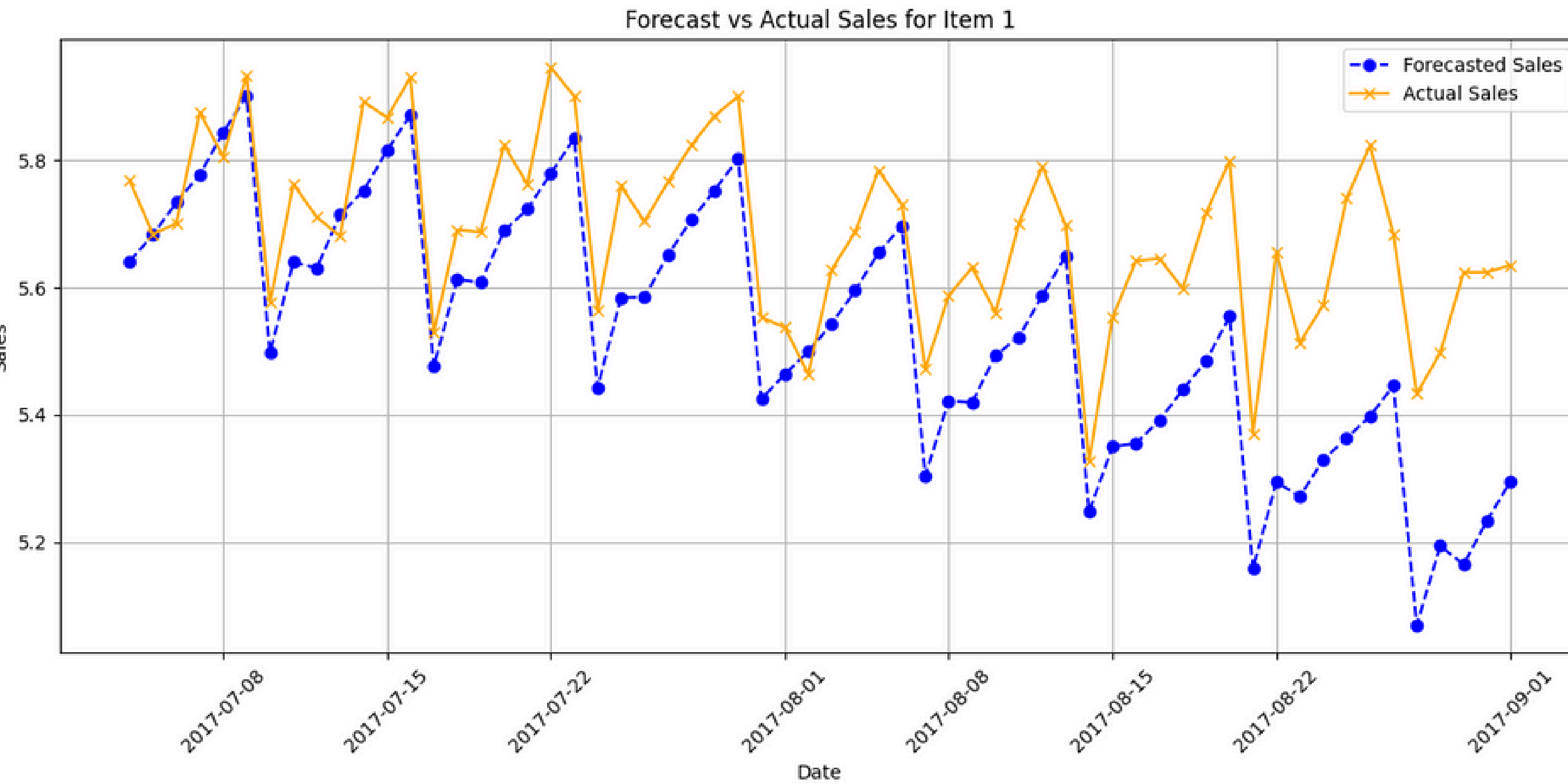
BOUCHEMLA CHERIF
OSMANE

16/06/2025

PROJECT REPORT
SALES FORECASTING


Forecast vs Actual for 1
Target Value
Date
Actual (Test)
Predicted
5.4
5.3
5.2
5.1
5.0
4.9
2017-12-01
2017-12-05
2017-12-09
2017-12-13
2017-12-17
2017-12-21
2017-12-25
2017-12-29
2018-01-01


Forecast vs Actual Sales for Item 1
Sales
Date
Forecasted Sales
Actual Sales
5.8
5.6
5.4
5.2
2017-07-08
2017-07-15
2017-07-22
2017-08-01
2017-08-08
2017-08-15
2017-08-22
2017-09-01

outline :

- intro
- fundamentals
- model 1
- model 2
- mistakes i made
- discussion and conclusion

introduction

introduction

Predicting future sales is critical for every business to optimize inventory levels, reduce holding costs, and improve cash-flow management . Using AI to analyze a company’s historical sales data delivers more accurate and timely forecasts than traditional methods, giving managers clear insights to make proactive stocking and budgeting decisions . By automating demand prediction, AI frees teams to focus on strategy and growth, turning past sales history into a powerful tool for smarter operations and healthier finances .

the data

	item	date	sales
0	1	2013-01-01	133
1	1	2013-01-02	99
2	1	2013-01-03	127
3	1	2013-01-04	145
4	1	2013-01-05	149

the data i used is basically 3 columns/features for now, item id, date of sale, and the total sales of that item on that date

fundamentals

data preprocess : log(x+1)

we transform the sales collumn by applying this function, why ?

- reduce skewness and outliers effect
- Transforms Multiplicative Relationships into Additive Ones
- Enhances Interpretability : Small differences on the log scale approximate percentage changes in the original units. This means that a unit change in log(sales + 1) roughly corresponds to a 100 % change in sales when changes are small, making insights easier to communicate

feature engineering

feature engineering here is basically just adding new features (columns) to our table
in our case we can add time related features such as holidays, day of year, month, is_weekend.
these features are self explanatory but the statistical features are not as simple.

rolling statistics :

These are especially useful in time series problems because they help the model recognize patterns, trends, and variability over time. Below is a breakdown of the features and their purpose.

for example the rolling_mean_7
calculates the mean for past 7 total sales
from this you can easily guess what the rolling std. median... do

Lag

Lag features explicitly give the model prior values of the target variable:

- lag_1, lag_7:
- The sales value 1 day and 7 days ago, respectively.

percentage changes

pct_change_1, pct_change_7:
The percent change in sales compared to 1 and 7 days ago, respectively, and shifted to avoid using current data.

auto correlation 7

- y_t be the value at time t
- $\bar{y}$ be the mean of the values in the window
- k be the lag (e.g., 7)
- N be the window size (e.g., 14)

Then the **autocorrelation at lag** k is:

$$\text{autocorr}(k) = \frac{\sum_{t=1}^{N-k}(y_t - \bar{y})(y_{t+k} - \bar{y})}{\sum_{t=1}^{N}(y_t - \bar{y})^2}$$

this feature basically detects if the weekly pattern that happened last week happened this week

exponentially moving average

Vt= current avg
Vt-1 = previous avg

theta = current value of sales

beta = constand between 0 and 1

$$V_t = \beta * V_{t-1} + (1-\beta) * \theta_t$$

$$
\begin{aligned}
V_0 &= 0 \\
V_1 &= \beta * V_0 + (1-\beta) * \theta_1 \\
V_2 &= \beta * V_1 + (1-\beta) * \theta_2 \\
V_3 &= \beta * V_2 + (1-\beta) * \theta_3 \\
V_4 &= \beta * V_3 + (1-\beta) * \theta_4 \\
V_5 &= \beta * V_4 + (1-\beta) * \theta_5 \\
V_6 &= \beta * V_5 + (1-\beta) * \theta_6 \\
V_7 &= \beta * V_6 + (1-\beta) * \theta_7 \\
V_8 &= \beta * V_7 + (1-\beta) * \theta_8 \\
V_9 &= \beta * V_8 + (1-\beta) * \theta_9
\end{aligned}
$$

this feature is helpful in noise reduction finding and finding short term trends

percentage change

pct_change_1, pct_change_7:
The percent change in sales compared to 1 and 7 days ago, respectively, and shifted to avoid using current data. Captures sales growth or decline trends.

ensembling :

Ensembling in machine learning and AI is a technique where multiple models are combined to solve the same problem and produce better predictive performance than any individual model on its own.

- Bagging : Train multiple models (usually of the same type) on random subsets of the data (with replacement), then average or vote their outputs.
- Boosting : Models are trained sequentially, and each new model focuses on fixing errors made by previous ones.
- Stacking : Train several models (could be of different types), then train a **meta-model** on their outputs to make the final prediction.

model 1

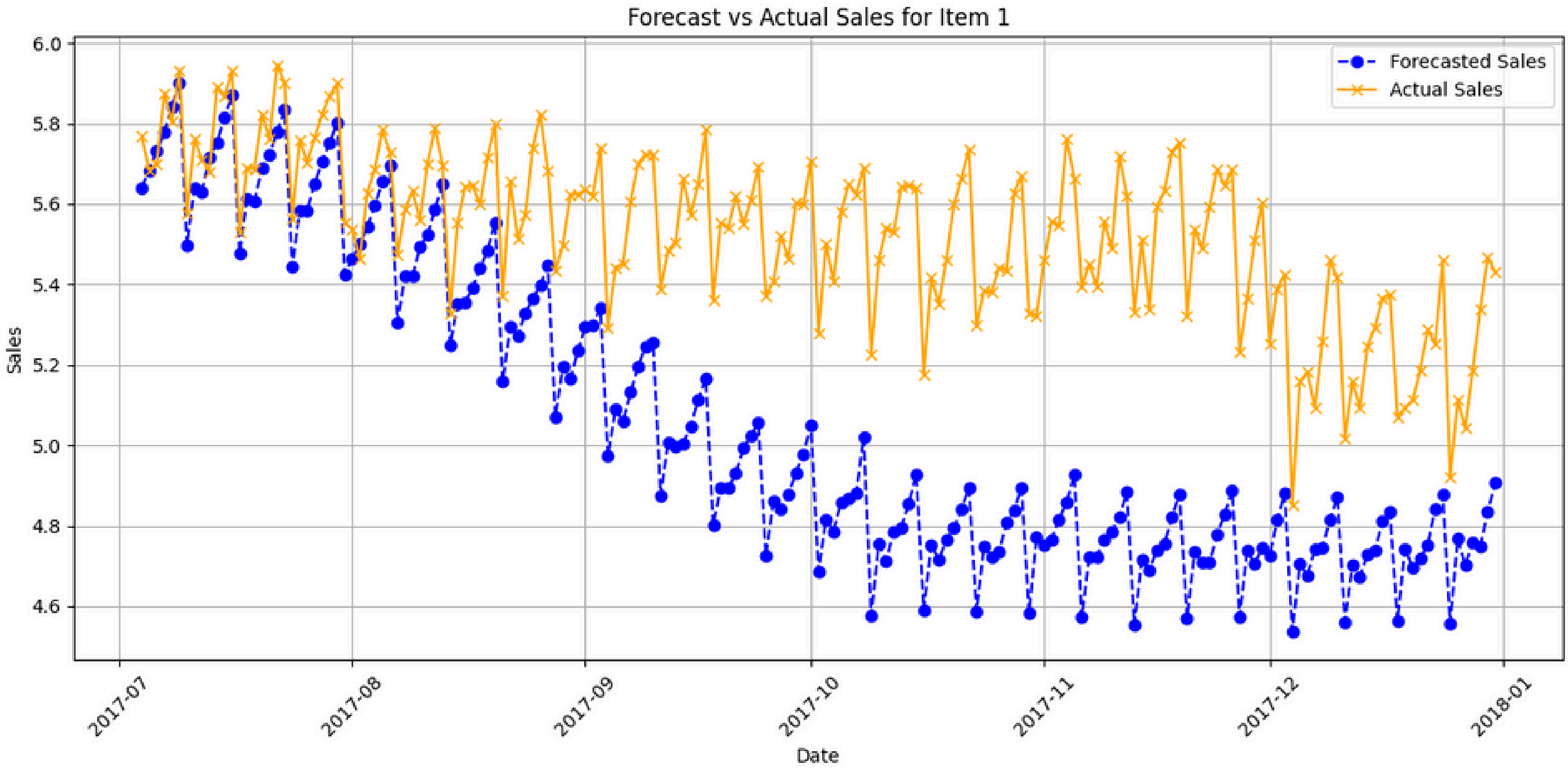
xgboost

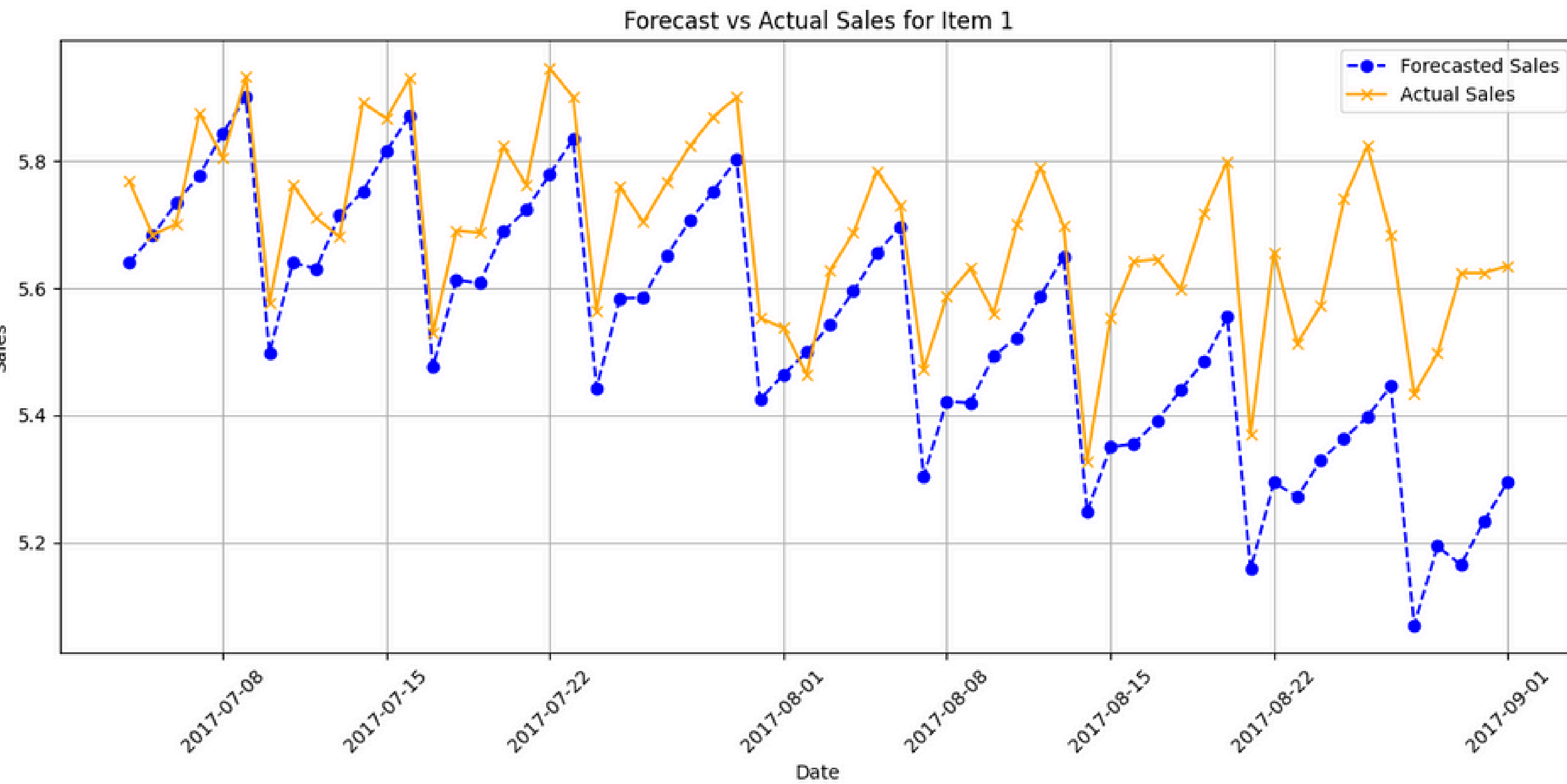
XGBoost (eXtreme Gradient Boosting) is an efficient and scalable implementation of gradient boosting that builds an ensemble of decision trees to solve regression, classification, and ranking problems.
XGBoost builds decision trees one after another. Each new tree is trained to correct the errors (residuals) made by the combined predictions of all previous trees.
The final prediction is a weighted sum of the outputs from all the individual trees. The idea is to gradually improve predictions by adding small, corrective steps.

recursive forecasting

for the first model i tried to implement
a recursive forecasting approach
basically because we don’t have future
feature values for times like t+2 and so
on
we our current data which allows us to
predict the sales for t+1
and we use that prediction to make
the prediction for t+n

results


Forecast vs Actual Sales for Item 1
Forecasted Sales
Actual Sales
Sales
Date
6.0
5.8
5.6
5.4
5.2
5.0
4.8
4.6
2017-07
2017-08
2017-09
2017-10
2017-11
2017-12
2018-01


Forecast vs Actual Sales for Item 1
Forecasted Sales
Actual Sales
Date
5.8
5.6
5.4
5.2
2017-07-08
2017-07-15
2017-07-22
2017-08-01
2017-08-08
2017-08-15
2017-08-22
2017-09-01

model 2

Autogluon library

The AutoGluon library in Python is an open-source AutoML (Automated Machine Learning) toolkit developed by Amazon Web Services (AWS). It is designed to automate the process of training and tuning machine learning models, making it easier and faster for both beginners and experienced practitioners to build high-performance ML models.
for our purposes well use autogluon for time series predicton
here is a link to the documentation for the library :

https://auto.gluon.ai/stable/tutorials/timeseries/forecasting-indepth.html

models used

Models Used

- Chronos: Transformer-based model, fine-tuned to capture global sales patterns and long-range dependencies.
- RecursiveTabular: Machine learning model trained on tabular features including rolling statistics, expanding means, and holiday effects.
- ETS (Exponential Smoothing): Captures trend and seasonality using a statistical approach

.

models used

Ensembling

AutoGluon to intelligently combine the predictions of multiple models. Instead of boosting or stacking explicitly, AutoGluon assigns learned weights based on validation performance and blends the forecasts accordingly

chronos

chronos is a state of the art time series forecasting model, it is based on the transformer

1. Tokenization of Time-Series Data
    - Instead of processing raw numerical values directly, Chronos discretizes time-series data into bins/ranges (similar to how words are tokenized in NLP).
    - Each range is mapped to a token, converting continuous time-series data into a discrete sequence (e.g., "bin_12", "bin_5", etc.).
    - This allows the model to treat forecasting like a language modeling task, predicting the next "token" (value range) in the sequence.

this is a good video that details how chrnos works :
https://youtu.be/_gFycwbfS0g?si=bJ1jr_jXjSQNz0ah

chronos

In this project, Chronos is used within the AutoGluon TimeSeries framework, which enables training across multiple item-level time series by treating each item (via item_id) as a separate time series instance. The model learns shared patterns across all items' sales histories but only sees the target variable.

To utilize additional information (e.g., holiday indicators, rolling statistics), we combine Chronos with other models like RecursiveTabular that can handle multivariate input. When ensembling is enabled in AutoGluon, it trains multiple models in parallel—including Chronos—and learns how to best combine their predictions to improve accuracy.

This setup allows us to harness the strengths of Transformer-based models for capturing long-term trends, while also leveraging rich feature sets via models that can process covariates.

recursive tabular model

this model basically does the same as model one that i made
but it does the feature engineering on its own
This recursive process introduces some risk of error accumulation over time, but it allows the model to take advantage of powerful feature engineering and model flexibility.
Why Use RecursiveTabular ?

- Handles multivariate input: Unlike Chronos, this model can use additional features (known covariates) during both training and prediction.
- Flexible and interpretable: Since it relies on standard tabular regressors like LightGBM, the resulting models can be inspected for feature importance.
- Easy to incorporate domain knowledge: You can add domain-specific features like "is_holiday" or "rolling_mean_7", and the model will learn how to use them effectively.

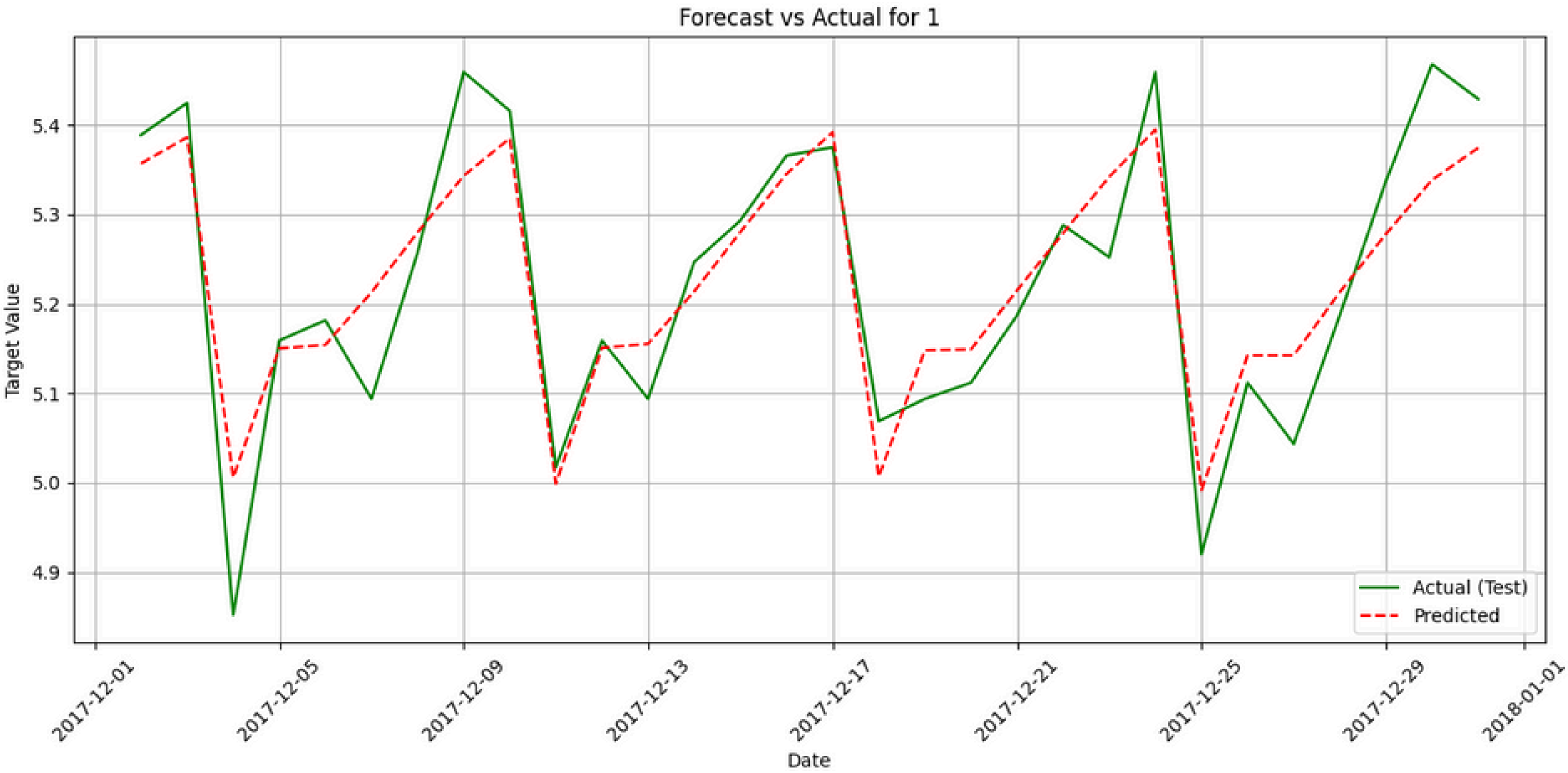
AutoEts

The AutoETS model in AutoGluon is a classic statistical forecasting method based on the ETS (Error, Trend, Seasonality) framework. It is particularly effective for univariate time series that exhibit clear trends and/or seasonal patterns, such as monthly sales, website traffic, or electricity demand.
How ETS Works
ETS models break down a time series into three components:

- Error (E): The type of noise (additive or multiplicative)
- Trend (T): How the series is changing over time (increasing, decreasing, or none)
- Seasonality (S): Repeating patterns over time (weekly, yearly, etc.)

There are many combinations of these components (e.g., additive trend + multiplicative seasonality), and AutoETS automatically searches across them to find the best one for your data using maximum likelihood estimation.

results


Forecast vs Actual for 1
Target Value
5.4
5.3
5.2
5.1
5.0
4.9
Actual (Test)
Predicted
2017-12-01
2017-12-05
2017-12-09
2017-12-13
2017-12-17
2017-12-21
2017-12-25
2017-12-29
2018-01-01
Date

conclusion :

Although the models I developed for time series forecasting rely primarily on sales and date features (excluding holiday-related factors), they still deliver acceptable performance. However, their reliability is limited because they do not account for external factors such as imports/exports, supply chain dynamics, or key economic indicators like GDP and inflation rates. For businesses seeking accurate sales forecasts, a tailored model—one that incorporates industry-specific variables and macroeconomic trends—would be essential to achieve optimal results.