VILNIUS UNIVERSITY

FACULTY OF MATHEMATICS AND INFORMATICS

DATA SCIENCE STUDY PROGRAMME

Master's thesis

Comparative Analysis of Time Series Forecasting Models in M Competitions

Laiko eilučių prognozavimo palyginamoji analizė naudojant M konkursų duomenis

Mantas Bagdonas

Supervisor : assoc. prof. dr. Linas Petkevičius

Reviewer : assist. prof. Andrius Buteikis

Vilnius
2025

Summary

This thesis examines modern time series forecasting methods using the complex M5 competition dataset, which includes over 30,000 hierarchical time series of daily Walmart product demand. Lightweight models like DLinear and FITS are compared with traditional methods such as ARIMAX and machine learning models like LightGBM. The results demonstrate that simple, linear models can produce forecasts comparable to resource-intensive methods while maintaining efficiency. It was also found, that the most prominent traditional time series models proposed for the M5 competition as benchmarks were outperformed by DLinear and FITS with wide margins. The study supports the proposition of DLinear as a new baseline for time series forecasting due to its low complexity and competitive accuracy, offering practical recommendations for practitioners to advance forecasting methods effectively.

Keywords: DLinear, FITS, time series, M5 competition, MOFC, demand forecasting, linear regression, frequency domain.

Santrauka

Šiame darbe nagrinėjami modernūs laiko eilučių prognozavimo metodai M5 konkurso duomenų rinkinio, turinčio daugiau nei 30000 Walmart prekių paklausą apibūdinančių hierarchinių laiko eilučių, kontekste. Itin nesudėtingi ir modernūs modeliai DLinear ir FITS lyginami su tradiciniais, statistiniais laiko eilučių modeliais, kaip ARIMAX ir Eksponentinis glodinimas, taip pat su M5 konkurse žibėjusiu LigthGBM. Rezultatai rodo, kad net ir paprasti, linijine regresija ir nedideliais parametrų kiekiais remti modeliai gali sąlyginai tiksliai prognozuoti pardavimus lyginant su kompleksiškais mašininiu mokymusi remtais modeliais. Taip pat, tradiciniai laiko eilučių modeliai, parinkti M5 konkurse kaip lyginimo pagrindas, buvo gerokai pranokti DLinear ir FITS modelių. Šio darbo išvados pritaria DLinear autorių siūlymui, ateityje laikyti DLinear modelį atskaitos tašku analizuojant dar naujesnius laiko eilučių metodus, dėl modelio puikaus kompleksiškumo ir tikslumo santykio.

Tyrimo rezultatai

Raktiniai žodžiai: laiko eilučių prognozavimas, DLinear, FITS, M5 konkursas, MOFC, linijinė regresija, paklausos prognozavimas.

List of Figures

List of Tables

Contents

Introduction

The amount of data storage is entering the period of exponential growth. From 2025, global data storage will double every two to three years (Kez et al. (2022)). This data comes in many formats, including posts on Facebook, YouTube videos, Twitter surveys, and more. By extracting useful information out of this data, patterns can be identified, which can help make forecasts or classify the data in some way. Temporal data is a type of data that represents an event or a state at a specific point in time Mahalakshmi et al. 2016. A large collection of temporal data recorded over a period of time is known as time series data, which will be the primary focus of this thesis.

Time series forecasting is applied in various fields such as finance, healthcare, meteorology (Sidiq (2018)) and energy (Cabreira et al. (2024)). Accurate forecasting methods allow organizations to make informed decisions and predict future trends. Professionals can use these methods to anticipate the amount of product stock they need, plan marketing promotions, make investment decisions and forecast energy consumption. Data-driven decisions help organizations achieve better operational efficiency, and higher efficiency requires a higher standard of accuracy from forecasting models (Makridakis, Spiliotis, et al. (2022)). Forecasting competitions contribute to the advancement of time series forecasting methods by allowing forecasting practitioners to showcase and compare new, state-of-the-art models and methods to solve various problems.

The Makridakis Open Forecasting Center (MOFC) provides a platform for researchers and professionals to compete for prizes, while, more importantly, advancing the state-of-the-art in time series forecasting. MOFC hosts M competitions, where thousands of specialists attempt to identify ways to improve the forecasting methods. The findings obtained in these competitions played an important role in advancing our knowledge of forecasting methods R. K. Hyndman 2020. These competitions also challenge the methods that are considered common practice. Older competitions demonstrated that statistically sophisticated methods like ARIMA and ARARMA can be outperformed by simpler ones, like Gardner's Dampen Trend Exponential Smoothing Makridakis, Hibon 2000. Attention is also brought to less popular or even new, unutilized forecasting methods that are often overlooked. The M competitions try to provide as much reproducibility as possible - the data and forecasts of M3, M4 and M5 competitions are publicly available, allowing researchers to apply and compare newer methods against (arguably) the more accurate forecasting methods of past time.

This thesis explores modern time series forecasting methods on the data provided in the M5 competition (2021). The M5 dataset is complex, hierarchical, exhibits seasonality, intermittency that spans over 30,000 individual time series, proving to be a truly challenging task for time series forecasting specialists. Upon the conclusion of the M5 competition, it was evident that the forecasting community has moved on to Machine Learning based methods to achieve accurate predictions. According to Makridakis, Spiliotis, et al. (2022) it was the first

M competition, where all of the best performing methods relied on Gradient Boosted Decision Tree models and their ensembles.

The aim of this thesis is to conduct a case study of modern, efficient, and interpretable time series forecasting models using data from the M competitions. To achieve this, the following *objectives* were formed:

- Extensive scientific literature review on state-of-the-art time series forecasting models and methodology,
- Investigate and prepare the data of multiple M competitions for experimental benchmarks,
- Do comparative analysis of current-day and winning solutions of M competitions,
- Provide generic results and recommendations for time series forecasting practictitioners.

Time series analysis is an important data mining and analysis field. It is applied for trading, inventory management and planning energy consumption. M competitions take a big part in the advancement of time series forecasting methods. Makridakis Open Forecasting Center periodically hosts competitions, in which time series forecasting specialists compete for prizes by submitting modern solutions to a given problem. At the end of the competition, organizers aggregate the results and draw conclusions about the used methods, highlight new and most effective methods. The aim of this thesis is to revisit the solutions of previous M competitions and apply comparative analysis between them and methods we have today.

1 Literature review

The time series forecasting problem is a widely researched data science field. However, due to the complexity of the time series domain, even old methods have not been completely deprecated. AutoRegressive Integrated Moving Average (ARIMA) is still considered a standard financial time series forecasting model (Paul 2024), proposed over 50 years ago by Box et al. (1970). ARIMA is still considered to be a valuable short-term forecasting model due to its efficiency and interpretability, thus being considered for powerful hybrid, machine learning-empowered models, as proposed by Y. Zhang et al. (2023). Measuring the influence of external factors and what influence they could have on particular trends in time series is particularly important for understanding business behaviors and making decisions (Wang et al. 2021). AutoRegressive Integrated Moving Average with eXogenous variables (ARIMAX) is an extended version of ARIMA that enriches forecasts with external covariates. This model tends to outperform ARIMA in datasets where external factors have a relationship with short and long-term trends, helping to account for the variance not explained by historical data. The importance of external factors in time series forecasting was demonstrated by Umair Mehmood et al. (2024), where significant error loss was recorded with the systematic introduction of variables. ARIMAX and ARIMA preform very well on datasets that exhibit linearity and stationarity characteristics, such as when dealing with more complex, non-linear time series, developing hybrid solutions can increase forecasting accuracy, as found by Yucesan et al. (2018).

As concluded after one of M3 competitions by Makridakis, Hibon (2000), sophisticated statistical models like ARIMA can be outperformed by simpler methods, such as Exponential Smoothing (ES), originally proposed by R. Hyndman, Khandakar (2008). Rabbani et al. (2021) compared the performance of a Seasonal ARIMA (SARIMA) model against a simple ES model. Even when the analyzed time series are seasonal, level and stationary, which are ideal conditions for ARIMA models, they are outperformed by a simple seasonal model. However such simple models that only rely on historical data are very limited, they do not account for an exceptionally wet or cold season, lower infrastructure investments, amount of traffic, day-to-day weather conditions and many other factors. According to findings by Makridakis, Spiliotis, et al. (2022), Exponential Smoothing forecasts were improved by 5.7% on average when using exogenous variables, starting with 25.5% at the top level, but regressing to loss of accuracy on levels 10, 11 and 12.

LSTF is such an in-demand problem, that even models like Random Forest (Breiman 2001), which are known more for their efficiency in classification and regression problems, can be applied in the time series domain, as highlighted by Kane et al. (2014) where it's concluded, that ARIMA model's inability to incorporate non-linear relationships could be reason enough to use an alternative such as Random Forest. In fact, with enough knowledge and resources, methods based on decision trees outperform models that are specifically created for time series forecasting. XGBoost and LightGBM are two popular and modern, robust decision tree algo-

rithms, the latter being the model used by all 50 best performing teams in the M5 competition. These models are considered to be types of Gradient Boosted Decision Trees (GBDT).

GBDTs are powerful, ensemble learning based models that combine multiple weak learners into a strong learner. GBDTs are highly effective for regression tasks like credit scoring (Liu et al. 2022) and classification tasks such as fraud detection (Hancock et al. 2021) due to their ability to model complex, non-linear relationships and handle feature interactions naturally. By iteratively reducing errors through boosting, they achieve high predictive accuracy while being robust to outliers and adaptable to different loss functions. LightGBM was proposed by Ke et al. (2017) to address scalability and efficiency issues when working with large datasets and high feature dimensions. The reason for the scalability issues, is that the data is scanned for each feature to estimate the information gain of all possible split points, which is time and resource consuming. Authors proposed two techniques: Gradient-based One-side Sampling (GOSS) and Exclusive Feature Bundling (EFB). In short, GOSS prioritizes instances (data points) with large gradients (high errors) and may randomly drop instances that have low gradients, to optimize speed, without significantly compromising the accuracy. Despite the undeniably high time series forecasting efficiency, LightGBM has its limitations too, as illustrated by **<empty citation>** LightGBM, much like other time series forecasting methods, struggles to extrapolate beyond observed data, as financial markets undergo structural shifts that weren't observed in the historical data.

The most eye-catching findings of the M4 competition by Makridakis, Spiliotis, et al. (2018) was the dominance of deep learning-based methods. Recurrent Neural Networks (RNNs), such as Long Short-Term Memory (LSTM) or Transformer (Vaswani et al. 2017) models, such as Informer or Temporal Fusion Transformer (TFT), have received much academic attention over the last few years (e.g.). One of the weaknesses hindering traditional time series forecasting models, is the fact that the model can only work with individual time series. This makes optimization methods such as cross-learning, a critical strategy for winners of M5 competition, difficult to apply. LSTMs, first proposed by Hochreiter et al. (1997), have been a reliable solution for the long-term demand forecasting. The key advantage of LSTM is being able to adaptively decide which past information to retain, capturing both short- and long-term dependencies, as opposed to the sliding window models of the traditional statistical methods. LSTMs, while slightly outshined on their own by more robust machine learning or decision tree models, stay relevant in the time series domain, shown by continued progress and inclusion in new hybrid models, such as Hybrid Attention-based Long Short-Term Memory (HA-LSTM) network, proposed by X. Zhang et al. (2023), where HA-LSTM outperformed the vanilla LSTM, as well as traditional statistical models. Such results reinforce the findings of Makridakis, Spiliotis, et al. (2018), that hybrid models generate exceptionally accurate forecasts, by leveraging the strengths of different statistical and machine learning models.

Transformers are deep learning architectures that rely on self-attention mechanisms to model dependencies across sequences, making them highly effective for time series tasks. One advantage Transformers offer is scalability, which is due to their parallel sequence process-

ing, which is sequential in LSTMs. Among their variants, Informers (Haoyi Zhou et al. 2021) optimize transformers by introducing a sparse self-attention mechanism and generative style decoder, collectively responsible for reducing computational complexity and memory consumption, and improving the inference speed of long-sequence predictions by predicting long time-series sequences at one forward operation. As per Haoyi Zhou et al. (2021) findings, Informer models outperform LSTM on common benchmarking datasets, recording a significant decrease in MSE/MAE on all forecasting horizons. The model outperformed other selected baselines as well, such as ARIMA, Reformer (known for struggling with time series forecasting) and DeepAR, generally outperforming them all, however, on the ECL dataset, DeepAR recorded better scores when forecasting shorter horizons (<336). Temporal Fusion Transformers (TFTs) (Lim et al. 2021) focus on interpretable forecasting by combining a self-attention decoder with specialized components such as variable selection networks and sequence-to-sequence layer for locally processing known and observed inputs. The performance of TFTs was found superior to LSTM, interpretable LSTM and TCN models when forecasting energy consumption demand (Nazir et al. 2023).

Due to their efficiency, Transformers have garnered much attention when solving the LSTF problem. However, as pointed out by Zeng et al. (2022), who claim that the way transformers solve their problems is inherently wrong when working in the time domain. Truchan et al. (2024) support this claim, repeating that the temporal information loss due to permutation-invariant nature of self-attention mechanisms is inevitable. According to results provided by Zeng et al. (2022), DLinear and NLinear outperform all Transformer-based models on all forecasting horizons (96, 192, 336 and 720). Transformers were primarily intended for problems in the natural language processing (NLP) domain, therefore these methods are not very interpretable when operating in the time series domain.

2 Data

Sales and demand data are among the most critical yet challenging (Özalp et al. 2021) types of data used by businesses for organizational planning. Unlike evenly distributed datasets, demand data often exhibit a variety of complexities that can make forecasting difficult, such as intermittency, long-tailed distributions, seasonality, trends, external influences, etc. (Ma 2024).

2.1 M5 Dataset

The data set used in this thesis serves as the foundation for the forecasting models and comparative analysis. The data set is provided by the organizers of the M5 competition. The data shows the day-to-day demand of 30490 Walmart items over 5 years. The data spans over:

- 1941 days;
- 3 states;
- 10 stores;
- 3 product categories;
- 7 departments.

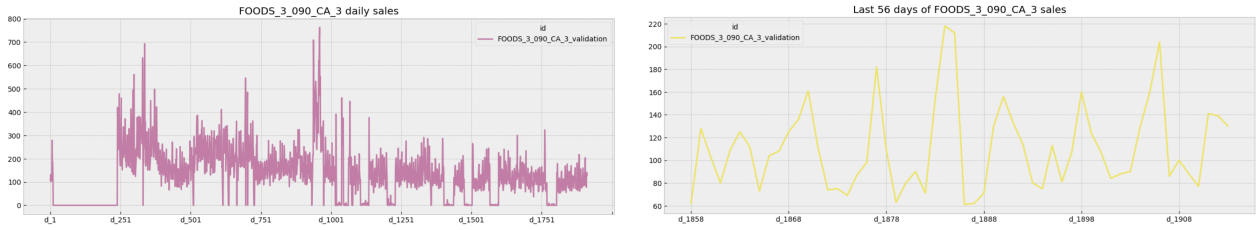
The M5 dataset is known for its hierarchical nature, with forecasts required at multiple levels of aggregation. The data set is organized as a multi-dimensional time series, where each item-store combination forms a distinct time series. The core variables of the data set are as follows:

- Item ID - unique identifier of each product;
- Store ID - unique identifier for each store, e.g. *CA_1*;
- Category - the category a product belongs to, e.g. *HOBBIES*;
- Department ID - a department in a given store, e.g. *HOBBIES_1*;
- Day - day of the observation, between 1 and 1969;
- Sales - the number of units sold on a given day;
- Event indicators - categorical variable indicating the presence of a holiday or a big special event that may influence demand, e.g. NBA finals or Christmas Eve;
- Calendar features - variables that help capture the effect of weekends, months or days of the week/month.

2.2 Exploratory data analysis

Exploratory data analysis is a fundamental step in the data-driven research process allowing us to understand the dataset's characteristics, patterns, anomalies, and complexity. Understanding the potential challenges such as data sparsity (discontinuation of products), and outliers (very low-demand products or departments).

Figure 1a displays all daily sales of a specific food item from store CA_4. The single graph encapsulates many characteristics that increase the difficulty of forecasting. The sales data appears to be unpredictable or at least irregular and quite noisy, however it's difficult to capture the real noisiness using such a zoomed-out graph. There are minimal sales periods that can indicate stock-outs while also exhibiting high sales peaks. Figure 1b plots the sales of the same product, but in the last 56 days. The daily demand is also highly irregular, varying from 60 up to 220 daily sales.



(a) *Food item sales over 5 years* ***(b)*** *Food item sales over last 56 days*

Figure 1 *Zoomed in vs out item sales*

The higher the demand, the more impactful the forecasts are when evaluating the model. Therefore, highly volatile sales data that exhibits so many irregularities can pose difficulties for less sophisticated forecasting models. Figure 2 captures the complexity of this dataset. Three products have relatively regular, but low sales. One product reported only 3 sales in the whole year, while two others have been reporting consistent sales ranging in a wide amplitude. What is also sometimes the case, as seen in the upper right corner, is that sometimes the product may be discontinued and the sales flatline at 0.

The share of total sales for categories or departments is not uniform. The FOODS category, and more specifically, FOODS_3 department consistently report the highest amount of sales across all observed stores. Figure 3 shows how the FOODS_3 department accounts for the majority of the variability in the example store. The seasonal sales of the store are nearly perfectly mirrored by the seasonality of the department, while the rest of the departments stay relatively stagnant. The only other department to record more than 1000 daily sales is HOUSEHOLD_1. It is consistently trending upwards while exhibiting some seasonality as well, while the second HOUSEHOLD department stays flat across all 5 years. Figure 4 confirms that FOODS_3 accounts for the most sales across all stores. HOUSEHOLD_1 comes in second, though it does have competition in WI_1, WI_2 and CA_4.

Seasonal decomposition was applied to the sales data of the largest departments in the CA_3 store - FOODS_3 and HOUSEHOLD_1, using the *seasonal_decompose* function from

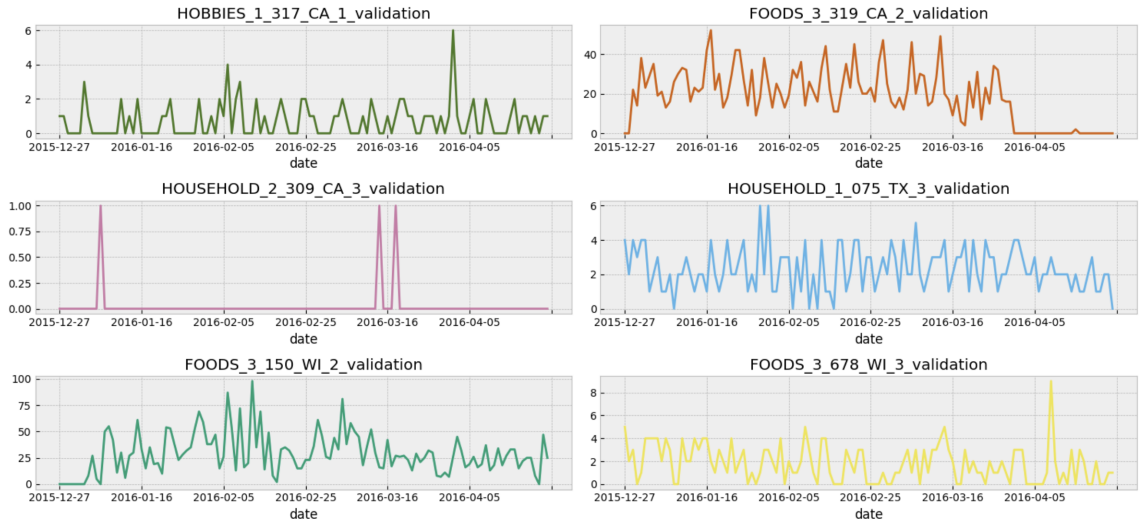


Figure 2 *Six random examples of product sales over the last 365 days*

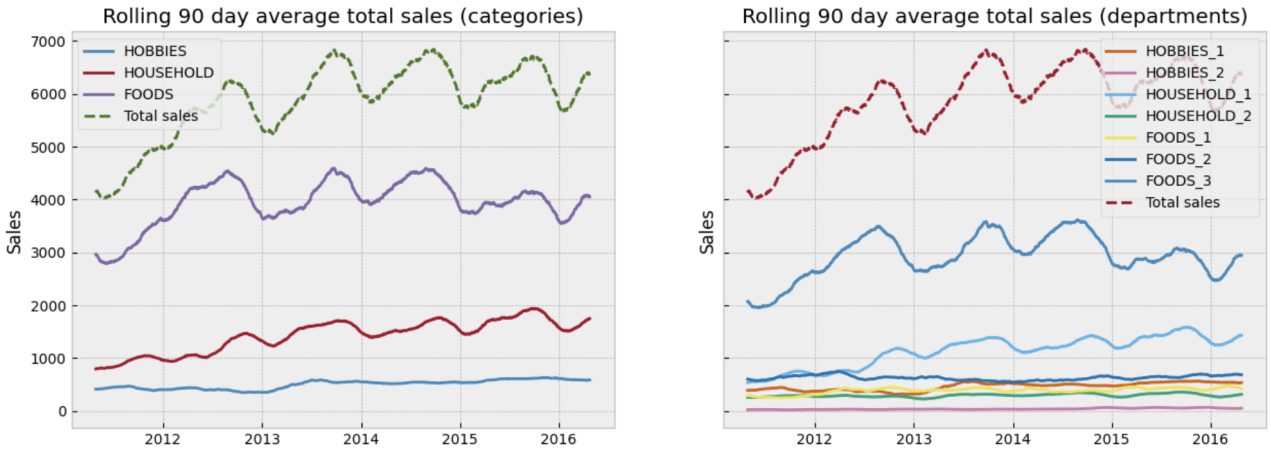


Figure 3 *Category and Department total sales comparison*

the *statsmodels.tsa* Python package. This function separates a time series into three distinct components: **trend, seasonality**, and **residuals (noise)**, enabling a clearer visualiztion of the overall trajectory of the data by isolating the impact of seasonality and noise. For this analysis, the seasonal periodicity was set to 1 year. The decomposed components are displayed in Figure 5 The amplitude of the seasonal component is approximately 600 for FOODS_3 and 400 for HOUSEHOLD_1, indicating a significant seasonal effect. Similarly, the residual component amplitudes, approximately 500 and 200 respectively, highlight a substantial contribution of unexplained noise as well. These findings suggest that both departments are heavily influenced by seasonal patterns and random fluctuations.

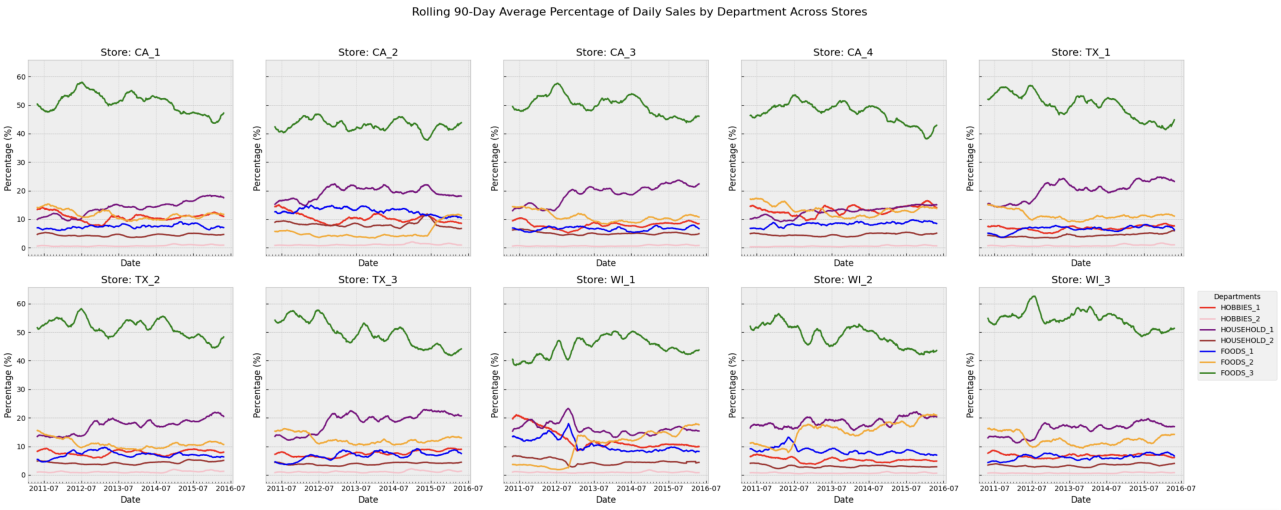


Figure 4 *Percentage share of total sales for all departments across stores*

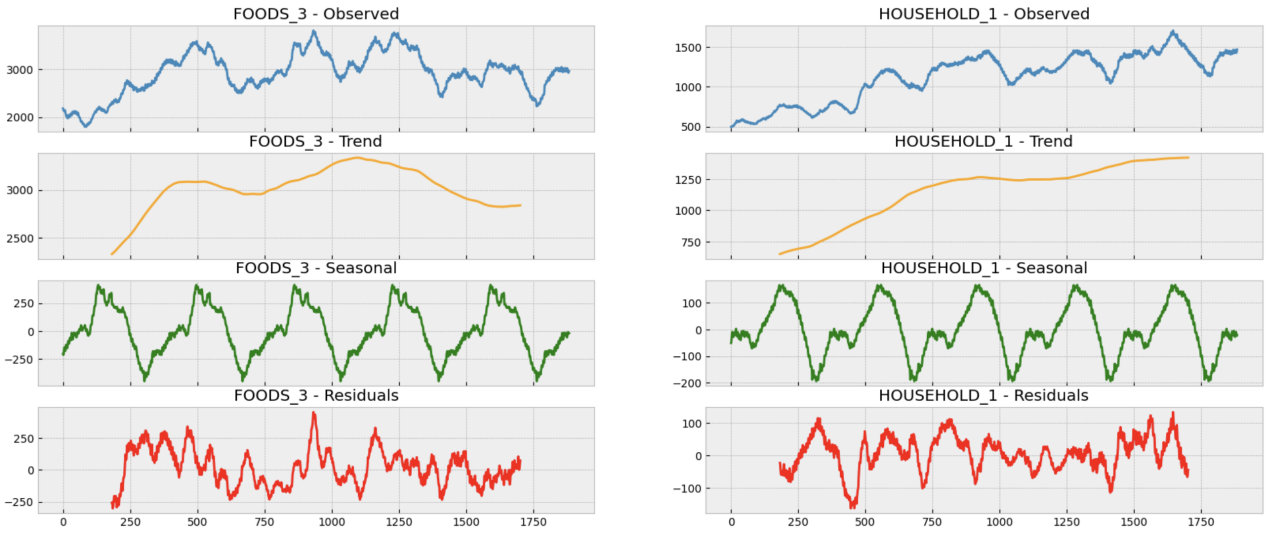


Figure 5 *Seasonal decomposition of FOODS_3 and HOUSEHOLD_1*

3 Analyzed methods

The theoretical overview of forecasting methods provides the foundation of understanding the strengths and limitations of the models under evaluation. This section focuses on two state-of-the-art approaches: **DLinear** and **FITS**.

3.1 DLinear

During a surge of Transformer-based solutions for the long-term time series forecasting (LTSF), a much simpler, lightweight solution was proposed by Zeng et al. (2022) to not only improve the forecasting accuracy, but to act as a lightweight, modern benchmark in LTSF.

Decomposition-based Linear (DLinear) forecasting model was proposed to address the limitations of deep learning-based methods for time series forecasting. The mechanism behind Transformer models is permutation-invariant, which heavily conflicts with the nature of time series domain. While positional encoding preserves some ordering information, the model being inherently uninterested in data ordering inevitably leads to temporal information loss.

Unlike traditional black-box models, DLinear explicitly decomposes a time series into trend and noise components. This decomposition enables the model to isolate systematic patterns and simplify the forecasting process. The model operates on the principle that time series data consists of predictable patterns and noise. By modeling these components separately using linear methods, DLinear achieves competitive performance while being very computationally efficient.

The core idea in DLinear is to decompose Time series x_t into a trend T_t and residual (seasonal) R_t components and model them individually in single-layer linear networks.

The decomposition process is quite simple. First, the input data is padded at the beginning and end to ensure that the moving average computation does not shrink the output size. The padding replicates the first and last values of the time series for half the kernel size on each side. Then the trend component is derived by traversing a moving average kernel over the padded input data. Then, the remainder (seasonal) component is created by subtracting the trend component from the raw input data. First look at the trend estimation method. With selected kernel size k, the time series will be expanded by adding repetitive beginning and ending values y_0 and y_N respectively $(y_{1,-\lfloor k/2 \rfloor}, y_{1,-\lfloor k/2 \rfloor+1}, .., y_{1,-1}, y_1, .., y_N, y_{N,1}, y_{N,2}, .., y_{N,\lfloor k/2 \rfloor})$. Then the trend and residuals are calculated:

$$\begin{aligned} T_t &= \frac{1}{k} \sum_{i=-\lfloor k/2 \rfloor}^{\lfloor k/2 \rfloor} y_{t+i}, \\ R_t &= y_t - T_t, \quad \forall t \in \{1, .., N\} \end{aligned}$$

where w is the lookback window, k is the kernel size, s is the stride, t is the current time step and $\hat{y}_t$ is the padded input data and y_t is the original input data.

The whole structure is captured within Figure 6. Once decomposition is achieved, DLinear models T_t and R_t using simple vector like linear models of trend and residuals:

$$\begin{aligned}
\begin{pmatrix} \hat{T}_{t+1} \\ \hat{T}_{t+2} \\ ... \\ \hat{T}_{t+p} \end{pmatrix}_{P\times 1} &= \begin{pmatrix} w_{1,0}+ & w_{1,t-L+1} & w_{1,t-L+2} & .. & w_{1,t} \\ w_{2,0}+ & w_{2,t-L+1} & w_{2,t-L+2} & .. & w_{2,t} \\ .. & & & & \\ w_{P,0}+ & w_{P,t-L+1} & w_{P,t-L+2} & .. & w_{P,t} \end{pmatrix}_{P\times(L+1)} \begin{pmatrix} 1 \\ T_{t-L+1} \\ T_{t-L+2} \\ ... \\ T_t \end{pmatrix}_{(L+1)\times 1} \\
&= \begin{pmatrix} w_{1,0}+ & w_{1,t-L+1}T_{t-L+1}+ & w_{1,t-L+2}T_{t-L+2}+ & .. & +w_{1,t}T_t \\ w_{2,0}+ & w_{2,t-L+1}T_{t-L+1}+ & w_{2,t-L+2}T_{t-L+2}+ & .. & +w_{2,t}T_t \\ .. & & & & \\ w_{P,0}+ & w_{P,t-L+1}T_{t-L+1}+ & w_{P,t-L+2}T_{t-L+2}+ & .. & +w_{P,t}T_t \end{pmatrix} \\
\mathbf{H}_T &= \mathbf{W}_T\mathbf{T}
\end{aligned}$$

and

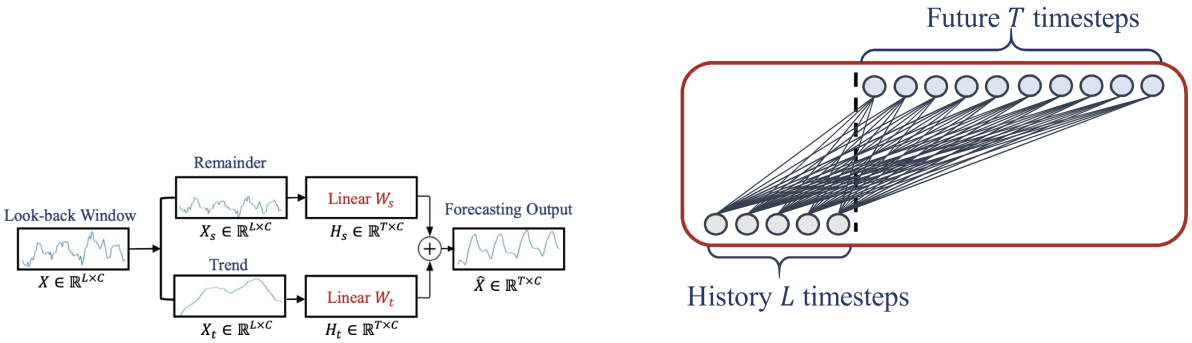
$$\begin{aligned}
\begin{pmatrix} \hat{R}_{t+1} \\ \hat{R}_{t+2} \\ ... \\ \hat{R}_{t+p} \end{pmatrix}_{P\times 1} &= \begin{pmatrix} w_{1,0} & w_{1,t-L+1} & w_{1,t-L+2} & .. & w_{1,t} \\ w_{2,0} & w_{2,t-L+1} & w_{2,t-L+2} & .. & w_{2,t} \\ .. & & & & \\ w_{P,0} & w_{P,t-L+1} & w_{P,t-L+2} & .. & w_{P,t} \end{pmatrix}_{P\times(L+1)} \begin{pmatrix} 1 \\ R_{t-L+1} \\ R_{t-L+2} \\ ... \\ R_t \end{pmatrix}_{(L+1)\times 1} \\
&= \begin{pmatrix} w_{1,0}+ & w_{1,t-L+1}R_{t-L+1}+ & w_{1,t-L+2}R_{t-L+2}+ & .. & +w_{1,t}R_t \\ w_{2,0}+ & w_{2,t-L+1}R_{t-L+1}+ & w_{2,t-L+2}R_{t-L+2}+ & .. & +w_{2,t}R_t \\ .. & & & & \\ w_{P,0}+ & w_{P,t-L+1}R_{t-L+1}+ & w_{P,t-L+2}R_{t-L+2}+ & .. & +w_{P,t}R_t \end{pmatrix} \\
\mathbf{H}_R &= \mathbf{W}_R\mathbf{R}
\end{aligned}$$

where final predictions:

$$\hat{\mathbf{T}} = \mathbf{H}_T + \mathbf{H}_R$$

where W_T and W_R are weight matrices, b_T and b_R are biases and h is the forecasting horizon. The modeled components are then combined:

$$\hat{x}_{t+h} = \hat{T}_{t+h} + \hat{R}_{t+h}$$



(a) *DLinear architecture* ***(b)*** *A single layer neural network*

Figure 6 *The whole structure of DLinear (Zeng et al. 2022)*

3.2 FITS

Unlike existing models that process raw data in the time domain, the Frequency Interpolation Time Series Analysis Baseline (FITS) proposed by Zhijian et al. (2024) operates on the principle that time series can be manipulated through interpolation in the complex frequency domain. Using FITS, forecasting results are obtained by simply extending the given look-back window with frequency interpolation.

The idea of FITS is to treat the time series data as a signal. The raw data can be broken down into a combination of sinusoidal waves (sine and cosine functions). Each wave has its own **frequency**, **amplitude** and **phase**. Such breakdown captures all of the information in the original time series without any informational loss. Instead of predicting the future values of the raw time series, FITS focuses on forecasting each individual wave. Forecasting these waves is simpler because, in order to do so, only the phase needs to be adjusted to project them forward. After the shifted sinusoidal waves are forecasted, they are combined to recreate the full time series forecast. This process ensures that the periodic behaviors in the past data remain consistent in the forecast and the results align with the observed trends. time series data representation in the frequency domain is visualized in Figure 7[1]. Higher amplitude and lower frequency waves represent significant patterns in the original data, while others might represent local noise-like residuals.

A signal existing in the time domain is described by how it changes over time. In the frequency domain, a signal is described by the **amplitude** and **phase** of the **frequency component**, it being represented as a **complex number**. Mathematically, the complex number associated with a frequency component can be represented as:

$$X(f) = |X(f)|e^{j\theta(f)}$$

Here:

- $X(f)$ - The complex number associated with the frequency component at frequency f,

[1]Image take from an article

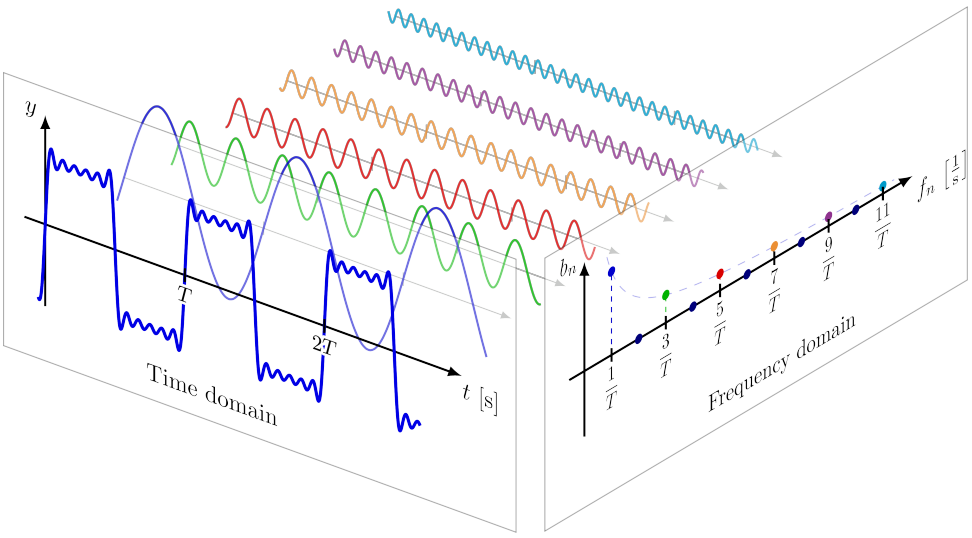


Figure 7 *Data representation in the frequency domain*

- $|X(f)|$ - The amplitude of the frequency component (how strong it is in the signal),
- $\theta(f)$ - The phase of the component (how delayed or shifted it is in time),

Figure 8[2] shows a frequency component represented as a vector on a complex plane. The length of the vector is the amplitude and the angle with the x (Real) axis is the phase.

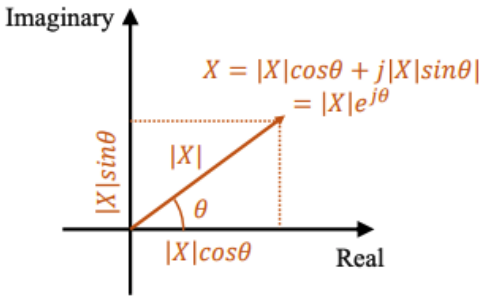


Figure 8 *Complex number on the complex plan*

Forecasting is performed by shifting the phase of the frequency components. If a signal $x(t)$ is shifted forward in time by a constant amount , resulting in the signal $x(t-\tau)$, the Fourier transform is given by:

$$X_\tau(f) = e^{-j2\pi f\tau}X(f) = |X(f)^{j(\theta(f)-2\pi f\tau} = [cos(-2\pi f\tau) + jsin(-2\pi f\tau]X(f)$$

The shifted signal still has the same amplitude $|X(f)|$, while the phase $\theta_\tau(f) - 2\pi f\tau$ shows a shift which is linear to the time shift.

[2]Image taken from Zhijian et al. (2024)

Figure 9[3] illustrates the FITS pipeline. The FITS algorithm steps, when input time series segment: $x_t \in \mathbb{R}^{N \times L_i}$, where N is the batch size and L_i is the input length and interpolation rate: $\eta = \frac{L_o}{L_i}$, where L_o is the desired output length of extended time series (original + predicted window).

1. Apply time series normalization:

$$\bar{y} = \frac{1}{T}\sum_{t=1}^{T} y_t, \quad \text{var}_y = \frac{1}{T}\sum_{t=1}^{T}(y_t - \bar{y})^2 + \epsilon,$$

where T is the time series windon length and $\epsilon = 10^{-5}$ is a small constant for numerical stability. The normalized time series is:

$$\tilde{y}_t = \frac{y_t - \bar{y}}{\sqrt{\text{var}_y}}, \quad \tilde{y}_t \in \mathbb{R}^{N \times L_i}.$$

2. Apply Fast Fourier Transform (rFFT) to the normalized time series to get the complex frequency domain:

$$Y_f = \text{rFFT}(\tilde{y}_t), \quad Y_f \in \mathbb{C}^{N \times L_i/2}, \quad f \in [0, \frac{1}{2T}].$$

3. Apply a low-pass filter to remove high-frequency components:

$$\hat{Y}_f = \begin{cases} \hat{Y}_f, & f \leq f_{\text{cut}} \\ 0, & f > f_{\text{cut}} \end{cases},$$

where f_{cut} is the cutoff frequency. Retain only the components up to f_{cut}:

$$\hat{Y}_{\text{low},f} = \hat{Y}_f, \quad f \in [0, f_{\text{cut}}], \quad Y_{\text{interp},f} \in \mathbb{C}^{N \times (L_i/2)}$$

4. Apply upsample the low-frequency components to match the extended length ηT, where $\eta = \frac{T_{\text{output}}}{T_{\text{input}}}$:

$$\hat{Y}_{\text{interp},f} = W \cdot \hat{Y}_{\text{low},f}, \quad \hat{Y}_{\text{interp},f} \in \mathbb{C}^{N \times \eta(L_i/2)}$$

where W is the weight matrix of the complex-valued linear layer.

5. Apply zero padding to extend the interpolated frequency representation to the original frequency length $T_{\text{output}}/2$ by zero-padding:

$$\hat{Y}_{\text{output},f} = \begin{cases} \hat{Y}_{\text{interp},f}, & f \leq f_{\text{interp}} \\ 0, & f > f_{\text{interp}} \end{cases}, \quad Y_{\text{output},f} \in \mathbb{C}^{N \times (L_o/2)}.$$

[3]Image taken from Zhijian et al. (2024)

6. Apply transform the frequency representation back to the time domain using the inverse FFT:

$$y_{\text{pred},t} = \text{iFFT}(\hat{Y}_{\text{output},f}), \quad y_{\text{pred},t} \in \mathbb{R}^{N \times L_o}$$

7. Apply reconstructed signal to compensation for the change in sequence length:

$$y_{\text{pred},t} = y_{\text{pred},t} \cdot \eta, \quad y_{\text{pred},t} \in \mathbb{R}^{N \times L_o}$$

8. Apply original scale and mean of the time series:

$$y_{\text{reconstructed},t} = y_{\text{pred},t} \cdot \sqrt{\text{var}_y} + \bar{y}, , \quad y_{\text{reconstructed},t} \in \mathbb{R}^{N \times L_o}.$$

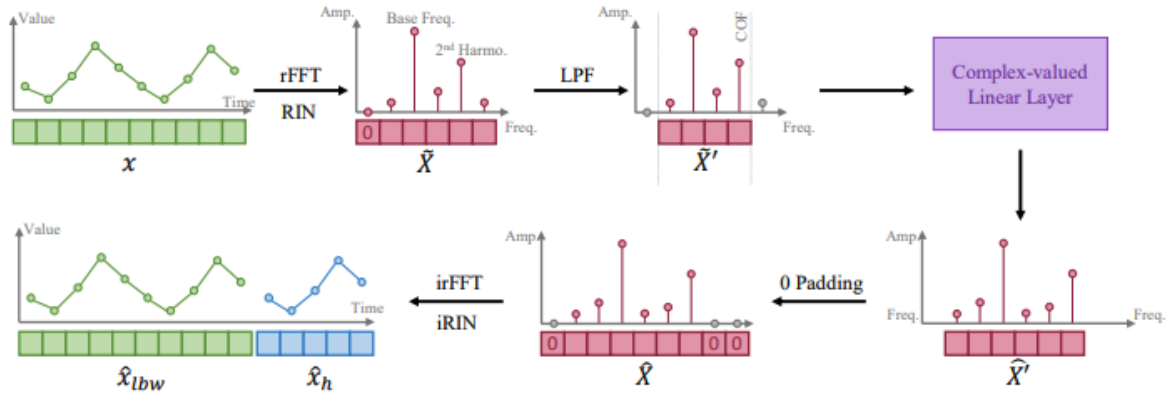


Figure 9 *Pipeline of FITS*

The FITS model offers a unique solution for solving the forecasting task by operating in the frequency domain. It achieves this by utilizing complex number multiplication within a single complex-valued linear layer to interpolate the input (lookback) data. This makes FITS well-suited for generating long-term time series forecasts, as it effectively captures and analyzes both local and long-term patterns.

4 Research

The research presented in this thesis consists of exploratory and comparative analysis of well-established, historically significant time series forecasting models against selected state-of-the-art models. The comparative analysis was conducted within an evaluation framework provided by the organizers of the M5 competition. Each experiment generates a 28-day demand forecast for all 30,490 items. The forecast accuracy is assessed at 12 aggregation levels and the **final** score is drawn by calculating the mean of all aggregate scores. Detailed analysis of benchmarks and selected comparative models and an overview of the evaluation methodology will be discussed in the following subsections. EDA, data preprocessing, submission evaluation was done using Python and R[4].

4.1 Benchmarks

Well-established benchmarks serve the purpose of evaluating modern solutions very well. Benchmarks provide a reference point for how accurately and efficiently the model performs. The organizers of the M5 competition have provided a long list of benchmarks that were used in their comparative analysis. The focus will be on these models:

- Random Forest (right and left) - particularly effective in capturing complex nonlinear relationships. Although primarily used for classification, random forests can also be adapted for regression tasks, thus serving as a robust benchmark (Breiman 2001).
- ARIMA (Autoregressive Integrated Moving Average) - a fundamental benchmark for time-dependent datasets due to its historical significance and flexibility through handling non-stationary data (Box et al. 1970).
- ARIMAX - more comprehensive view of the predictors' impact on the forecasted variable (Box et al. 1970).
- ADIDA (Aggregate-Disaggregate Intermittent Demand Approach) - specifically designed to handle intermittent demand time series by forecasting on different levels of aggregation. Made more relevant by the use of the top-down forecasting strategy during this research (Nikolopoulos et al. 2011).
- Exponential Smoothing, simple and effective in capturing trends, levels, and seasonality of time series data (R. Hyndman, Koehler, et al. 2008).
- Croston's method - tailored for intermittent demand forecasting, where demand occurs at irregular intervals. Separating the forecasting of demand size and demand intervals, enables the method to better handle zero-demand periods, making it a good benchmark for slow-moving products (Croston 1972).

[4]All Python and R code related to research can be found in this repository

- SBA (Syntetos-Boylan Approximation) - a refinement in Croston's method, aimed to correcting the known bias in Croston's forecasts. Adjusts the demand interval component to provide more accurate forecasts for intermittent demand series. Offers improved forecasting accuracy in contexts where inventory control and stock optimization are critical (Syntetos et al. 2001).

Using these benchmarks allows for a more comprehensive comparative analysis of reviewed models due to differences in approach and capabilities. By comparing the results across a diverse list of models, it becomes possible to demonstrate the relative advantage of the chosen models and justify their use in practical applications.

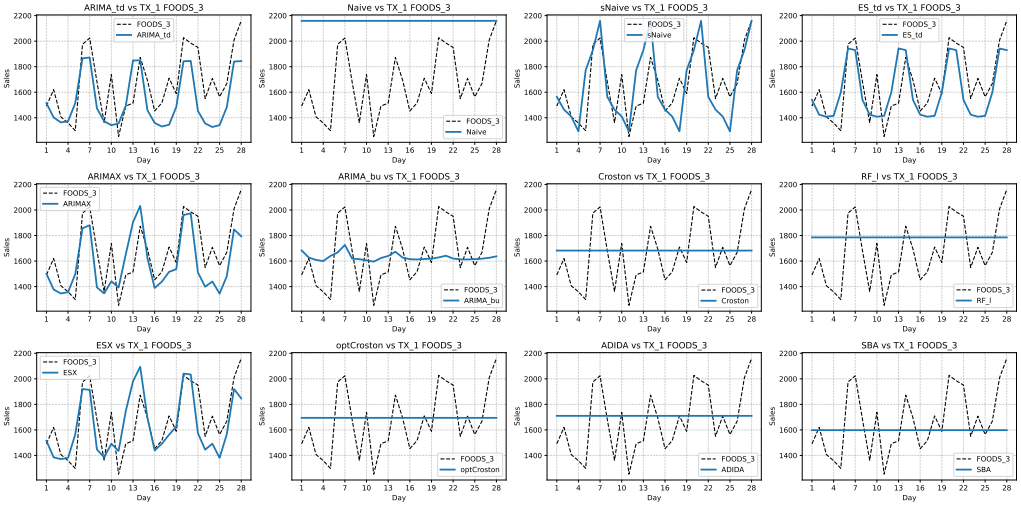


Figure 10 *Benchmark models aggregated forecasts of FOODS_3 department*

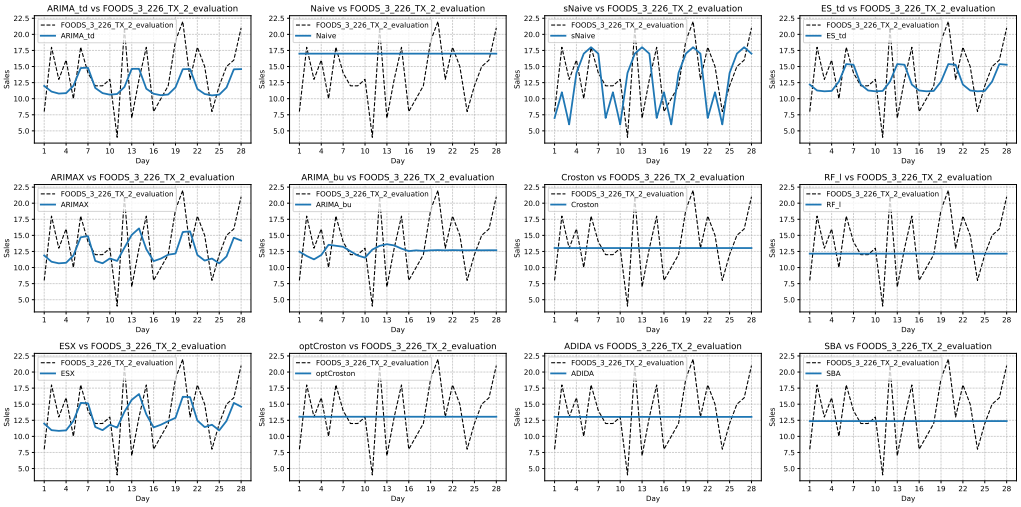


Figure 11 *Benchmark models aggregated forecasts of FOODS_3_226*

Figure 10 shows benchmark model aggregated forecasts plotted against the sales of a specific food department. Models such as ARIMAX or ESX attempt to model noisy day-to-day changes in product sales, while others either expect the sales to continue in a predictable pattern. The disadvantages of SBA, ADIDA and Croston models are made quite apparent, due

to their inability to capture time-dependent patterns. Forecasts these models generate are static and generally estimate an "average" demand rate over a given horizon. Figure 11 illustrates similar patterns. $ARIMA\_td$, $ARIMAX$, ES_td and ESX models were generated using the **top-down** strategy, which is why the department forecasts are just a scaled version of the item-level forecast.

4.2 Methodology

The goal of this thesis is to analyze two state-of-the-art models that aim to provide modern, simple, less resource intensive benchmark models for the LTSF problem. The models will be evaluated under the framework provided by the M5 competition, that computes WRMSSE scores on twelve forecast aggregation levels. The analyzed models will be compared against selected benchmark models, as well as the M5 competition submissions.

4.3 Evaluation

The product-store unit sales can be mapped across either product categories or geographical regions, as shown in Table 1.

ID	Aggregation level	Number of series
1	Unit sales of all products, aggregated for all stores/states	1
2	Unit sales of all products, aggregated for each State	3
3	Unit sales of all products, aggregated for each store	10
4	Unit sales of all products, aggregated for each category	3
5	Unit sales of all products, aggregated for each department	7
6	Unit sales of all products, aggregated for each State and category	9
7	Unit sales of all products, aggregated for each State and department	21
8	Unit sales of all products, aggregated for each store and category	30
9	Unit sales of all products, aggregated for each store and department	70
10	Unit sales of product x, aggregated for all stores/states	3,049
11	Unit sales of product x, aggregated for each State	9,147
12	Unit sales of product x, aggregated for each store	30,490
Total		42,840

Table 1 *M5 sales aggregation levels*

The accuracy score will be computed by separately averaging their scores across the forecasting horizon. Then, the error will be averaged again across the series according to weights assigned to each product.

The accuracy of the point forecasts is evaluated using the Root Mean Squared Scaled Error (RMSSE). This measure is calculated 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 series at point t, and $\hat{T}_t$ is the generated forecast, n the size of the look-back window, and h - the forecasting horizon. Once RMSSE is estimated for all time series, the weighted RMSSE (WRMSSE) is calculated:

$$WRMSSE = \sum_{i=1}^{42{,}840} w_i \times RMSSE_i,$$

where w_i is the weight and $RMSSE_i$ is the score of the ith series in the competition. The weights are based on the amount of dollar sales over the last 28 observations of the final training sample - the sum of sales multiplied by their price. This approach ensures that slow-moving products aren't contributing as much to the overall score of the model. Such model evaluation measure ensures that the forecast accuracy is estimated by focusing most on the series with high importance, i.e., series that are significant in monetary terms (Makridakis, Spiliotis, et al. 2022). Therefore, the best models are expected to perform forecasting with lower errors for the series with more value to the business.

4.4 Top-down forecasting

When working with large time series datasets, such as the M5 Walmart sales dataset, fitting and forecasting tens of thousands of time series separately becomes a very time consuming task. Top-down forecasting approach solves this problem in part, by reducing the amount of time series to model. The M5 Walmart daily sales dataset has 30,490 bottom-level time series to forecast, however, aggregating the sales data by store and department, the amount of time series is reduced to just 70. After forecasting these aggregated time series, forecasts are disaggregated using weights, estimated for each bottom-level time series.

Weights for bottom-level time series are assigned by calculating the share of units sold for each subset.

$$W_i = \frac{\sum_{t=T-h}^{T} X_{i,t}}{\sum_{j=1}^{N}\sum_{t=T-h}^{T} X_{j,t}}$$

where W_i is the weight of the i-th item, N is the number of time series in the subset, T is the observed time period and $X_{i,t}$ is units sold on the i-th time series on day t. The bottom-level forecast is calculated by multiplying the top-level forecast on a given day t by the corresponding bottom-level weight:

$$F_{i,t} = W_i \times A_t$$

where $F_{i,t}$ is the forecast for the i-th bottom level time series for the day t ($0 < t <= 28$) and A_t is the top-level forecast for the selected aggregation level.

The main drawback of the top-down forecasting approach is that the resulting forecasts

heavily depend on the assumption that unit sales will continue on the same constant trajectory. As discussed in 2, the M5 data set exhibits the characteristics of intermittency, and seasonality. The top-down approach forces the resulting bottom-level forecast to follow a group trend, even though there are cases where an item's sales crashed to 0 right before the forecasting horizon, or when an item simply exhibits different weekly demand patterns than other items in the department. However, fitting the model on aggregated data should yield more accurate forecasts on the higher aggregation levels. Since the overall score of the model depends on all 12 aggregation levels equally, an expected lower accuracy in bottom-level forecasts is acceptable if it results in higher scores in higher aggregation levels.

4.5 DLinear

DLinear model was adopted by the *darts* Python package, which provides a configurable interface that acts as a model training black box. The interface to tune certain hyperparameters, such as kernel size, lookback window, optimizer, loss function, learning rate, number of epochs to train the model for. The *darts* implementation[5] of DLinear is a slight improvement of the original model proposed by Zeng et al. 2022. *darts* version includes the ability to use shared weights for the components, supports the use of covariates.

Forecasts were performed using different combinations of hyper-parameters. Additionally, forecast accuracies were evaluated using both parametric and non-parametric versions of the model.

Table 2 shows WRMSSE metrics for forecasts produced using different-sized lookback windows - 112, 56, 28 and 14. DLinear generated the most accurate forecasts using the 56-day lookback window, however, other window sizes tend to produce comparable results. Other hyperparameters, such as learning rate, epoch number did not have any significant effect on the models performance. Without taking into account exogenous covariates, DLinear model tends to generate marginally less accurate results (ass seen in Table 2 **LB56-NP** column.

Different groups of exogenous covariates were experimented with, preprocessed, and the following were selected for the parametric version:

- Event count - derived from variables *event_name_*1 and *event_name_*2 by counting the number of ongoing events on a given day.
- SNAP count - Supplemental nutrition assistance program, derived by counting $snap\_CA$, $snap\_TX$ and $snap\_WI$.
- Weekend - boolean variable that signifies that a given is on the weekend.
- Day of week - Day of week, 1 through 7.

Different kernel sizes influence the balance between the reduction of noise in the trend component and the accuracy of the residual component. A larger kernel effectively smooths

[5]Repository: ¸

out short-term fluctuations, providing a clearer representation of the overall trend. However, this can lead to the residual component capturing less meaningful variation, as some relevant short-term patterns may be absorbed into the trend. Experiments with different kernel sizes were conducted to evaluate their effect on the overall accuracy of the model. Kernel sizes of 7, 14, 21 and 28 were evaluated. Table 3 shows the results of the forecast evaluation by selected kernel size. The models were trained using a lookback window of 56 days according to the best performing run of the previous experiments.

Level	**LB112**	**LB56**	**LB28**	**LB14**	**LB7**	**LB56-NP**
1	0.33536	0.25771	0.27604	0.36395	0.42510	0.27975
2	0.41378	0.35975	0.38257	0.44736	0.53038	0.39386
3	0.48810	0.43347	0.45174	0.50710	0.5774899	0.46404
4	0.38193	0.31914	0.33486	0.43703	0.48290	0.34223
5	0.46598	0.42457	0.41919	0.50555	0.54208	0.43534
6	0.48686	0.45617	0.47734	0.55706	0.60876	0.48442
7	0.57351	0.55362	0.55723	0.62554	0.66606	0.56957
8	0.56374	0.52714	0.54461	0.61082	0.65859	0.55233
9	0.67312	0.64163	0.64712	0.70006	0.74707	0.65689
10	1.00855	1.00336	1.00253	1.01038	1.01376	1.00373
11	0.96078	0.95848	0.95863	0.96379	0.96550	0.95970
12	0.90884	0.90672	0.90759	0.91067	0.91073	0.90820
6.5	0.60505	0.57015	0.57995	0.63661	0.67737	0.58751

Table 2 *DLinear forecast WRMSSE score comparison by lookback window. With default kernel size*

	kernel_size=7	**kernel_size=14**	**kernel_size=21**	**kernel_size=28**
WRMSSE	0.56843	0.56929	0.568815	0.57026

Table 3 *DLinear model WRMSSE scores with different kernel sizes*

4.6 FITS

The FITS forecasting accuracy experiments were performed using the official implementation repository by Zhijian et al. (2024). Quite a few experiments were run with different hyper-parameters to identify the optimal combination. Table 4 shows the best results written in *italic*. 28-day is the minimum required lookback window and is close to being the most performant too. However, a 40-day lookback window gives the model a slight increase in forecast accuracy. After that, the 44-day lookback window and up tend to produce worse performance. Feeding the model with more data per epoch improved the performance quite a bit, while also using a learning rate of at least *0.05*.

FITS still lacks the ability to use covariates in forecasting, therefore, the following experiments were nonparametric. Model training and forecast generation was executed as follows:

1. Time series were aggregated on store-department level;
2. Dataset split into train and validation groups. Selected split was 85/15;
3. Dataset was batched into *lookback window* sized batches. Selected batch size was 32;
4. Random batches were selected over up-to 100 epochs.
5. If validation loss doesn't increase over 3 epochs, the model with the best validation loss is selected as the final model. $MSELoss$ function is used for calculating loss.

Level	**28LB**	**32LB**	**36LB**	**40LB**	**44LB**	**56LB**	**84LB**
1.0	0.31473	0.30884	0.34838	*0.29267*	0.39653	0.40978	0.40494
2.0	0.40819	0.40740	0.43499	*0.39884*	0.47409	0.47780	0.47855
3.0	0.47844	0.47418	0.49890	*0.47324*	0.52526	0.53568	0.54154
4.0	0.37340	0.37107	0.40268	*0.35611*	0.44586	0.46632	0.46084
5.0	0.45271	0.45162	0.48177	*0.44344*	0.52028	0.54956	0.54290
6.0	0.50733	0.50772	0.53096	*0.49231*	0.55566	0.56107	0.56447
7.0	0.58181	0.58099	0.60495	*0.57100*	0.62771	0.63851	0.63707
8.0	0.57090	0.57024	0.58889	*0.56433*	0.60400	0.61460	0.62089
9.0	0.66665	0.66481	0.68293	*0.66174*	0.69345	0.70590	0.70366
10.0	1.00561	1.00502	1.01026	*1.00297*	1.01384	1.01083	1.01134
11.0	0.95973	0.95936	0.96298	*0.95787*	0.96499	0.96201	0.96268
12.0	0.90771	0.90737	0.90970	*0.90637*	0.91094	0.90829	0.90915
6.5	0.60227	0.60072	0.62145	*0.59341*	0.64439	0.65336	0.65317

Table 4 *Forecasting accuracy of FITS given different-sized lookback windows*

4.7 Results

DLinear and FITS models were evaluated against historical benchmarks and against the top M5 kaggle competition submissions. The DLinear submission would rank 38th out of 7,022 and FITS - 86th (Table 5). Overall, forecasts generated by FITS and DLinear models outperformed traditional benchmarks. Furthermore, the forecasts are comparable to the ones submitted by top competitors, who relied on ensembles of robust machine learning models.

Placement	Submission	WRMSSE Score
1	YeonJun IN_STU	0.52043
2	Matthias	0.52816
3	mf	0.53571
...	...	...
38	DLinear 56LB	0.56843
...	...	...
86	FITS 40LB	0.59341
...	...	...
471	ESX	0.67906
...	...	...
520	ARIMAX	0.69061

Table 5 *Submission placements in the M5 competition*

According to Zhijian et al. 2024, FITS model relies on almost 50 times less parameters than even lightweight models such as DLinear. Figure 12 shows a linear relationship between the selected lookback window and the trainable parameter size, parameter count reaching 22012 for the 56 lookback window model. Figure 13 visualizes the complexity graphs for FITS. Here, the model with the 56 lookback window only has 408 trainable parameters, which is ~**53.95** times smaller than its' DLinear counterpart. Graphs also display WRMSSE loss over an increasing lookback window. Figure 12 shows that as lookback window increases, the WRMSSE loss becomes smaller, reaching 0 above the 56 lookback and starting to produce less accurate forecasts. Figure 13 shows a more complicated relationship between WRMSSE loss and window size increase.

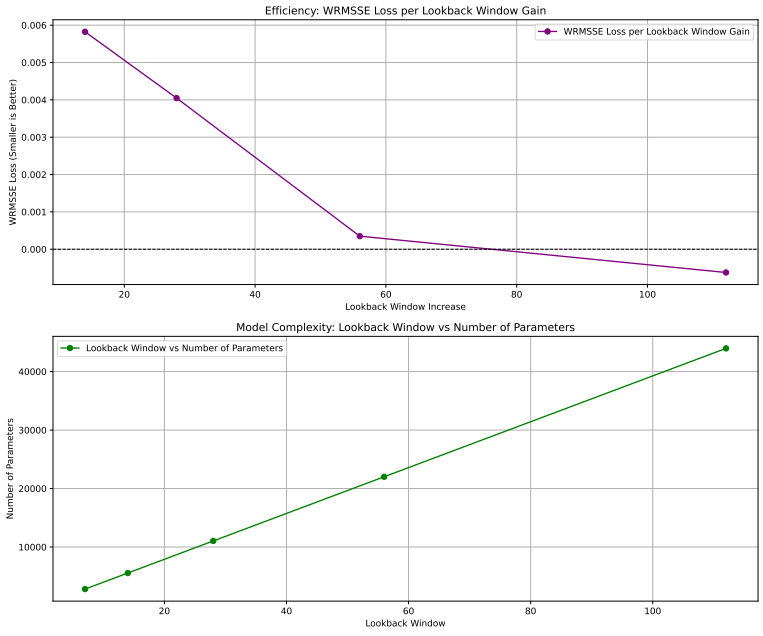


Figure 12 *DLinear complexity and WRMSSE loss vs lookback window increase*

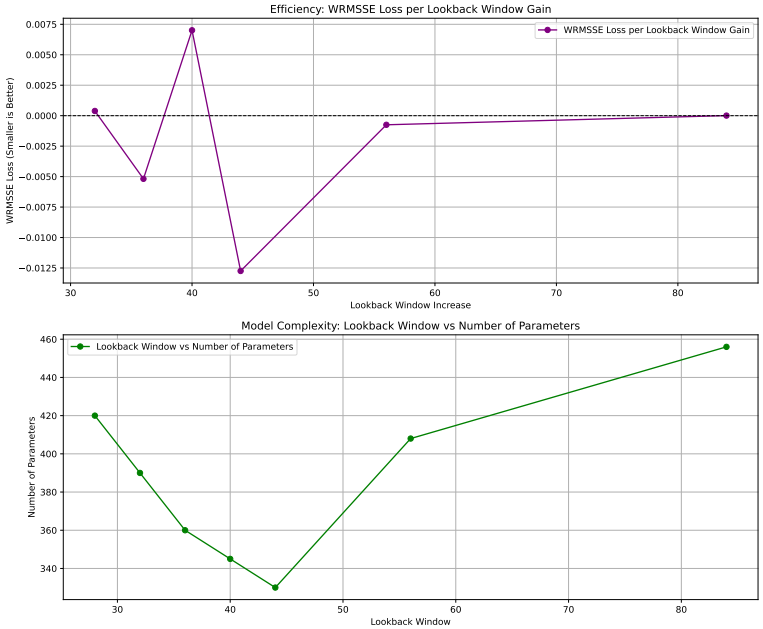


Figure 13 *FITS complexity and WRMSSE loss vs lookback window increase*

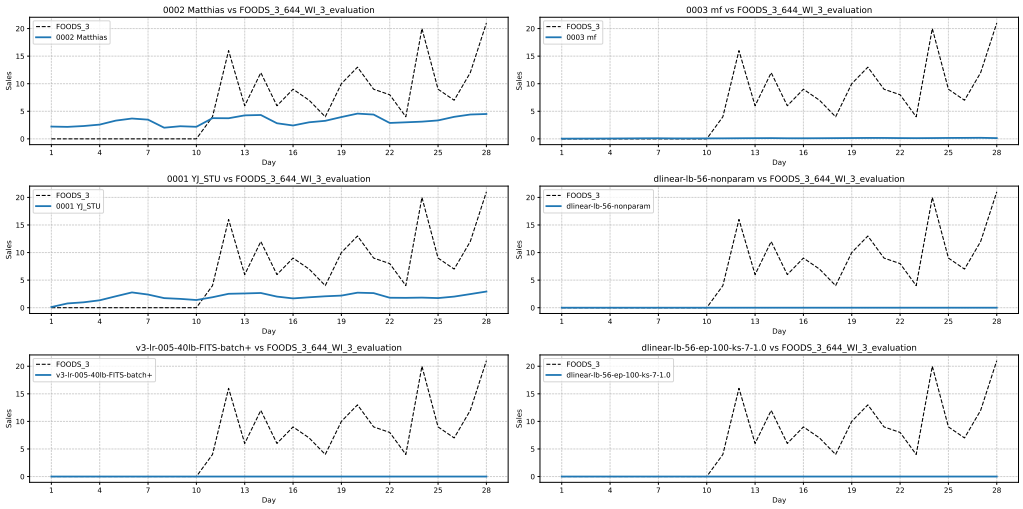


Figure 14 *Final forecasts on an outlier item*

The M5 dataset had **812** instances of bottom-level items that had 0 or close to 0 sales the last 28 days before the forecasting horizon, but started recording sales somewhere during the future 28 days. Figure 14 illustrates the drawback of the top-down forecasting approach and simple item-weight disaggregation. If a certain product doesn't record a high demand over the selected period of time, it is assigned a weight close to zero. Leading to a very low sales forecast, as shown in Figure 14.

Figure 15 and Figure 16 show the aggregated forecasts for specific FOODS and HOBBIES departments. These graphics show how a good top-level forecast may not result in an accurate bottom-level forecast.

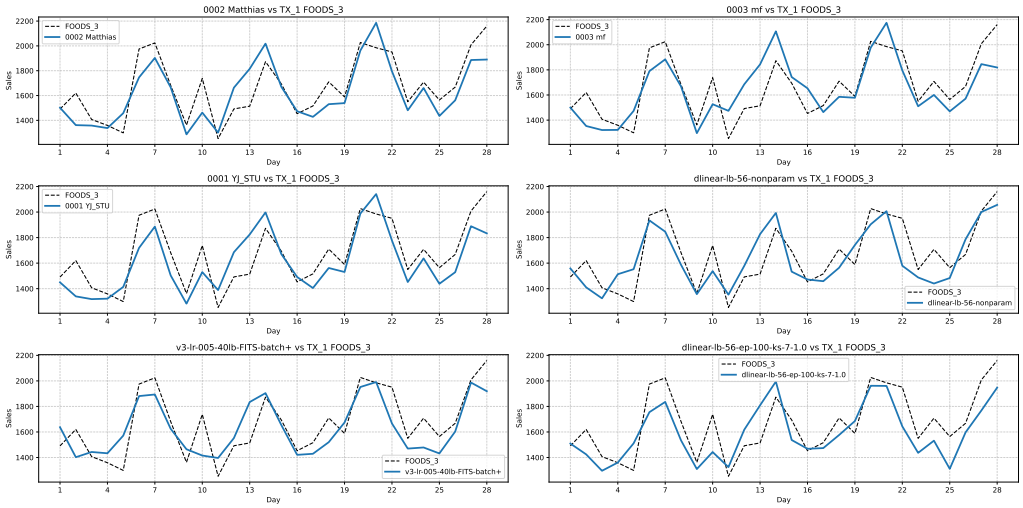


Figure 15 *Final forecasts on an outlier item*

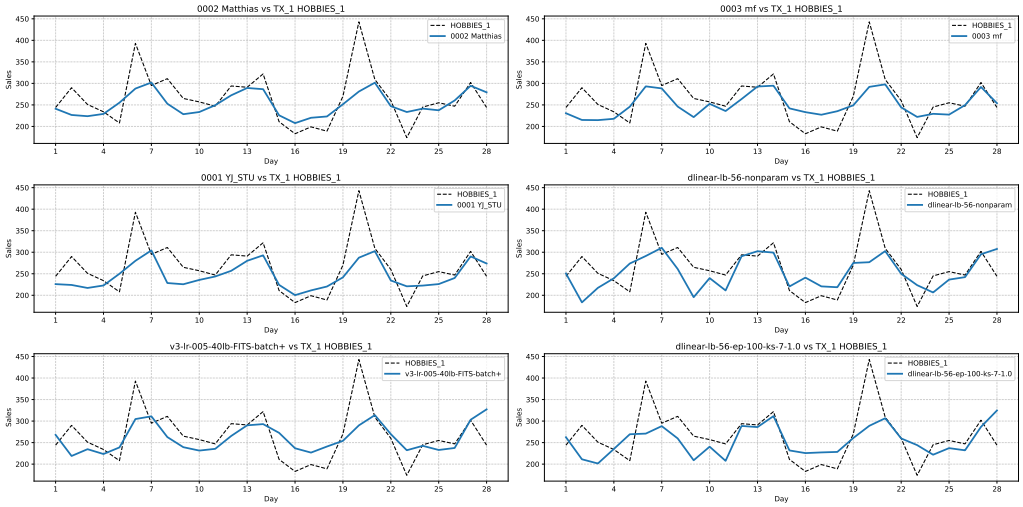


Figure 16 *Final forecasts on an outlier item*

5 Conclusion

Models evaluated in this case study prove to have the ability to generate forecasts with accuracy comparable to robust machine learning models. More specifically, per Makridakis, Spiliotis, et al. 2022 findings, all 50 most accurate forecatsts used LightGBM, a decision tree-based machine learning algorithm. According to the findings of the M5 competition, very few teams tried to build a single model that would generate all 30,490 forecasts. Instead, alternate models were created and results averaged - the winning team developed 220 different models for different aggregation levels, and finally calculated the average forecasts between 6 resulting partial submissions. The M5 dataset is a collection of closely related time series, that's why winning teams were able to leverage cross-learning into training the most flexible models. This strategy is much more complex to implement using DLinear and FITS, as such linear models treat each series independently. Having to rely on high-level forecasting and then making assumptions about item-level demand results in information loss and inaccurate outlier forecasts, as illustrated in Figure 14.

FITS was found to outperform the original DLinear implementation (Zhijian et al. 2024) on various benchmark datasets, however, the *darts* implementation of DLinear shows some improvement and actually generates more accurate forecasts than FITS.

The goal of this thesis was to evaluate two state-of-the-art linear models as they are, therefore micro-optimizations like cross-learning, hybrid or machine-learning based forecast disaggregation and methods were not used to improve final scores. Using the final scores of the selected models and having compared them to traditional benchmark models, following conclusions can be drawn:

- Traditional methods, such as ARIMAX and ESX, were outperformed by a wide margin by both FITS and DLinear, as demonstrated in Table 5;
- Simple and lightweight linear regression-based models produce forecasts comparable to those of robust, resource-demanding, ensemble machine learning models;
- The findings of this thesis support the proposition of Zeng et al. (2022) to consider DLinear a baseline in future research due to comparably accurate forecasts and very low complexity.

References and sources

Box, G., G. Jenkins (1970). *Time Series Analysis: Forecasting and Control*. Holden-Day.

Breiman, L. (2001). "Random Forests". In: *Machine Learning*, pages 5–32.

Cabreira, M. M. L. et al. (2024). "A Hybrid Approach for Hierarchical Forecasting of Industrial Electricity Consumption in Brazil". In: *Energies*.

Croston, J. (1972). "Forecasting and Stock Control for Intermittent Demands". In: *Journal of the Operational Research Society*.

Hancock, J. T., T. M. Khoshgoftaar (2021). "Gradient Boosted Decision Tree Algorithms for Medicare Fraud Detection". In: *SN Computer Science*.

Haoyi Zhou, S. Z. et al. (2021). *Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting*.

Hyndman, R., Y. Khandakar (2008). "Automatic time series for forecasting: the forecast package for R". In: *Journal of Statistical Software*.

Hyndman, R., A. B. Koehler, J. K. Ord (2008). *Forecasting with Exponential Smoothing: The State Space Approach*. Springer Science Business Media.

Hyndman, R. K. (2020). "A brief history of forecasting competitions". In: *International Journal of Forecasting* (1), pages 7–14.

Hochreiter, S., J. Schmidhuber (1997). "Long Short-Term Memory". In: *Neural Computation*.

Yucesan, M., M. Gul, E. Celik (2018). "Performance Comparison between ARIMAX, ANN and ARIMAX-ANN Hybridization in Sales Forecasting for Furniture Industry". In: *Drvna industrija*.

Kane, M. J. et al. (2014). "Comparison of ARIMA and Random Forest time series models for prediction of avian influenza H5N1 outbreaks". In: *BMC Bioinformatics*.

Ke, G. et al. (2017). "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*. NeurIPS.

Kez, D. A. et al. (2022). "Exploring the sustainability challenges facing digitalization and internet data centers". In: *Journal of Cleaner Production*.

Lim, B. et al. (2021). "Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting". In: *International Journal of Forecasting*.

Liu, W., H. Fan, M. Xia (2022). "Credit scoring based on tree-enhanced gradient boosting decision trees". In: *Expert Systems with Applications*.

Ma, L. (2024). "Long and short-term power supply and demand forecasting based on time series analysis under high proportion clean energy integration". In: *Journal of Physics*.

Mahalakshmi, G., D. S. Sridevi, D. S. Rajaram (2016). "A Survey on Forecasting of Time Series Data". In: *2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE'16)*. IEEE.

Makridakis, S., M. Hibon (2000). "The M3-Competition: results, conclusions and implications". In: *International Journal of Forecasting*, pages 451–476.

Makridakis, S., E. Spiliotis, V. Assimakopoulos (2018). "The M4 Competition: Results, findings, conclusion and way forward". In: *International Journal of Forecasting*.

— (2022). "M5 accuracy competition: Results, findings and conclusions". In: *International Journal of Forecasting* (4), pages 1346–1364.

Nazir, A. et al. (2023). "Forecasting energy consumption demand of customers in smart grid using Temporal Fusion Transformer (TFT)". In: *Results in Engineering*.

Nikolopoulos, K. et al. (2011). "An Aggregate - Disaggregate Intermittent Demand Approach (ADIDA) to Forecasting: An Empirical Proposition and Analysis". In: *Journal of the Operational Research Society*.

Özalp, M. M., Ö. Akarsu (2021). "Demand Forecasting in Retail: A Machine Learning Based Approach for Accurate Short-term Forecasts". In: *19th International Logistics and Supply Chain Congress (LMSCM2021)*.

Paul, J. (2024). *Financial Time Series Analysis with Transformer Models*.

Rabbani, M. B. A. et al. (2021). "A Comparison Between Seasonal Autoregressive Integrated Moving Average (SARIMA) and Exponential Smoothing (ES) Based on Time Series Model for Forecasting Road Accidents". In: *Arabian Journal for Science and Engineering*.

Sidiq, M. (2018). *Forecasting Rainfall with Time Series Model*.

Syntetos, A., J. Boylan (2001). "On the bias of intermittent demand estimates". In: *International Journal of Production Economics*.

Truchan, H., C. Kalfar, Z. Ahmadi (2024). "LTBoost: Boosted Hybrids of Ensemble Linear and Gradient Algorithms for the Long-term Time Series Forecasting". In: *CIKM '24: Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 2271–2281.

Umair Mehmood, J. B. et al. (2024). "Machine Learning-based Predictive Inventory for a Vending Machine Warehouse". In: *IEEE Internet of Things Magazine*.

Vaswani, A. et al. (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*. NeurIPS.

Wang, H. et al. (2021). "A Methodology for Calculating the Contribution of Exogenous Variables to ARIMAX Predictions". In: *The 34th Canadian Conference on Artificial Intelligence*. Canadian Artificial Intelligence Association.

Zeng, A. et al. (2022). "Are Transformers Effective for Time Series Forecasting?". In: *Proceedings of the AAAI Conference on Artificial Intelligence*.

Zhang, Y., S. Tang, G. Yu (2023). "An interpretable hybrid predictive model of COVID-19 cases using autoregressive model and LSTM". In: *Scientific Reports*.

Zhang, X. et al. (2023). "Enhancing Time Series Product Demand Forecasting With Hybrid Attention-Based Deep Learning Models". In: *IEEE Access*.

Zhijian, X., Z. Aling, X. Qiang (2024). "FITS: Modeling time series with 10k parameters". In: *Proceedings of the International Conference on Learning Representations (ICLR)*.