Norwegian School of Economics

Bergen, Spring 2022

Sales forecasting in fashion retail chain

A case study in Vietnam

Giang Phan & Thuong Au

Supervisor: Maximilian Rohrer

Master's thesis, Economics and Business Administration

Business Analytics

NORWEGIAN SCHOOL OF ECONOMICS

Acknowledgments

This thesis is the final requirement for completing the Master in Business Analytics at the Norwegian School of Economics (NHH). We have had the privilege of applying our knowledge gained in Norway in analyzing a significant industry, textile, in the Vietnamese economy.

First and foremost, we would like to thank our supervisor, Assistant Professor Maximilian Rohrer, for his devotion. He has guided us with his exceptional advice and instant support when needed throughout the time.

We would not have completed this thesis without valued data from Couple TX provided by Ms. Nga Tang. She did not only introduce us to real-world business problems but also assisted us with industry insights so that we could explore and construct proper models.

Finally, we wish to send our gratitude and appreciation to our family members and friends, who have always been great emotional supporters in walking us through all the hardships during this pandemic time.

Norwegian School of Economics

Bergen, June 2022

Giang Phan Thuong Au

Abstract

Sales forecasting is the key to the success of a supply chain, especially in the fashion industry. Vietnam is a major apparel supplier for international brands and has a dynamic, fast-growing market. The lack of complete forecasting systems in such a market motivates our development of sales forecast procedures featuring quantifiable results and practical implementation. Both statistical and machine learning methods were tested using actual data. We found that Random Forest consistently yields the lowest error metrics, followed closely by XGBoost. Meanwhile, clustering did not provide conclusive evidence of improved accuracy. As the study's data sources come entirely from one company, these procedures are applicable for other firms to deploy their data without external mining sources.

Keywords – NHH, Master Thesis, Time Series, Seasonal Naïve, ARIMA, Facebook Prophet, XGBoost, Random Forests, Fashion, Multistore, Retail, Sales Forecasting

Contents

List of Figures

List of Tables

1. Introduction

Vietnam’s fashion industry is an important contributor to its economy. Despite being a major apparel manufacturer worldwide, the domestic fashion market is underdeveloped. Proper exploitation of available data will help clothing companies in the competition. Thus, we aim to develop a well-tuned, reliable sales forecasting process using historical sales data to plan their operations more precisely and efficiently. This thesis section begins with detailed background information about Vietnam's fashion market, the company whose data is used in forecasting model development, and current forecasting practices. It continues by stating the focus of the entire thesis and ends with how it is organized into different sections.

1.1. Background

1.1.1. Vietnam’s apparel market landscape

Contributing largely to Vietnam’s Gross Domestic Product (GDP), the textile industry has continuously increased. According to Vietnam's General Statistics Office, the textile industry's export in the first six-month of 2021 is 31,4 million USD, accounting for 19,82% of the GDP. Despite being a significant player in the global textile industry, the domestic industry is still considered small and underdeveloped. There are many reasons. First, Vietnamese garment enterprises are mostly export-oriented or outsourcing partners. Hence, few firms focus on the local market. Second, they are not used to defining the market needs for their products at a specific time in a particular place. Thus, overstock inventory discourages companies from producing trendy clothes. However, with a young population, Vietnam is expected to be promising land for the “human beauty” industry. It is understandable since youngsters seem to care about their appearance and are willing to spend a significant amount of money on it. Local enterprises can exploit the price advantage to maintain their share in the market.

1.1.2. Company background

Founded in 2009, Couple TX is one of the most popular fashion brands for young adults in Vietnam (Pham, 2019). Their target customers are young males, females, and kids, with couple-matching clothes as their most famous products. They have 45 official stores concentrated in the South (coupletx.com, 2019). Even though having run a retail business since 2009, Couple TX only started collecting data properly through a point of sale (POS) system in 2017. It records all customer purchases with transaction information. Couple TX uses this data to plan production.

1.1.3. Demand forecast in the industry: role, current practices, and challenges

Optimizing the inventory effectively and efficiently is one of the keys to success in any manufacturing business. Hence, demand forecasting has long become one of the crucial yet challenging practices in supply chain management. Predicting the number of products for each store at a particular time is even more challenging. An accurate demand forecast minimizes the risk in all the operational activities related. It improves supply chain performance by having a better budget, inventory, and distribution planning (Frank et al., 2003). Meanwhile, a poor forecast could lead to consequences, such as the Bullwhip Effect (Norrman & Naslund, 2019).

However, the rapidity and instability of the fashion industry have made it harder to have a precise demand prediction. It is because of the high level of uncertainty. Their products have a short life cycle, highly variable with trends and seasons. Thus, different methods and variables are also exploited to improve forecasting accuracy. Sales data is the most common and valuable information for the retail industry. From them, time-series forecasting methods are applied to forecast future sales. Within the last ten years, the number of machine learning applications in sales forecasting research has increased by almost tenfolds (Aamer et al., 2020).

On the other hand, according to Rostami-Tabar (2021), business forecasting practice in developing countries is far behind developed countries. Rostami-Tabar states that there has been no significant effort recorded to shorten the gap in the last two decades. Developing countries lack the adequate capacity for statistical analysis, including reliable data, skillful practitioners, technology updates, and funds (Bhattacharyya & Timilsina, 2010). Therefore, retail businesses in developing countries should highly consider simple techniques in this case despite their less commonly recommended by academic literature.

Couple TX, or other Vietnamese enterprises, does not apply any proper forecasting methods for sales in their business. Usually, specialists prepare and randomly buffer the production quantity based on experience. Hence, it is crucial now to implement a sophisticated method. The aiming model needs to be easy to use yet much more accurate than the number purely put by the company's employees.

1.1.4. Machine learning applications in the fashion industry

In the industrial revolution 4.0 era, machine learning more and more contributes to the development of technology in humankind. They promise better solutions than traditional methods. Hence, the fashion industry is not excluded. Machine learning helps garment manufacturers with the following seasonal design themes prediction and inventory management. It advises us with the complex data-driven decision in the period, target customers, place, and demand of a particular trend.

Meanwhile, in Vietnam, the application to the fashion industry is still limited despite government policies and investment in digitalization, industrial revolution 4.0. Vietnamese fashion enterprises only focus on machinery automation in manufacturing but have not developed their business scope for the retailing market. There are few publications on fashion using machine learning in Vietnam, and neither is there a leader in technology development in this industry. However, machine learning gradually captures more attention as it becomes critical to business success. A highly prioritized need of fashion companies is more precise manufacturing planning.

1.2. Topic statement

This thesis aims to evaluate the performance of various forecasting methods using Couple TX data. Both traditional and advanced models are included and compared against each other to find the best-performing model. Additionally, since the chain retailers consist of various stores, we also consider the options of choosing between a "one-for-all" model, individual models for each store, or models for each group of stores. Therefore, we also try clustering based on store variables along the way to maximize forecasting accuracy.

1.3. Thesis structure

The first section states an overview of the Vietnamese apparel industry and Couple TX. We also discuss the sales forecast practice and machine learning applications in Vietnam.

Section 2 finds and studies research that discusses either three topics. The first topic is forecasting methods in the fashion industry. Next, we search for clustering practice and its impacts on forecasting results. Lastly, we find studies of store characteristics on store performance. We also identify gaps in those research to orient our later work.

Section 3 through 5 provide theoretical background, experiment setup, and results. We introduce various forecasting methods and evaluation metrics and explain why we choose them. We also present how we preprocess the data, feed them into different models, and evaluate the results. We pick a specific store to display the effect more easily and clearly.

Section 6 follows up and answers research questions. Finally, section 7 concludes the paper by summarizing the findings and offering extensions for future researchers.

2. Literature review

Having understood the purpose of the thesis, the following section is divided into three parts. Previous time-series forecasting methods, which contain both traditional statistical approaches and advanced artificial intelligence methods, are conducted to investigate and form a grounding knowledge for this master thesis. From that, we try to identify any limitations and gaps. Then, improvement ideas that can later be focused on are proposed in the last subsection.

2.1. Forecasting methods

2.1.1. Traditional statistic approach

Time-series forecasting is widely employed when predicting sales data. Those methods are simple and easy to implement with available historical data. The results are computed fast accordingly. This practice assumes that previous patterns in the past, whether seasonal, trend, cyclical, or irregular, will reappear in the future (Chopra & Meindl, 2016, p. 528). Hence, it becomes even more suitable if the demand is relatively stable. These statistical techniques include well-known models, such as naive, moving average, exponential smoothing, or Autoregressive Integrated Moving Average (ARIMA).

Nowadays, practitioners usually use statistical approaches as baseline models to compare performance with other more advanced methods. For example, a Naive forecast can be a benchmark to forecast new item demand (Singh et al., 2019) and daily SKU demand (Spiliotis et al., 2020). Moreover, due to the high fluctuation of the fashion industry, ARIMA and Seasonal ARIMA (SARIMA) approaches are popular among other methods for quick and intuitive forecast results. Both ARIMA and SARIMA-based models are well-examined by Liu et al. (2013), Box et al. (2015), or, more recently, Ren et al. (2016).

Traditional time-series methods are undeniably still in good use up to now. However, at the same time, there are some situations where they do not detect and perform well. They are insufficient to capture non-linear patterns. Additional and random factors that influence data, such as promotions, holidays, weather, etc., are also noise and reduce the accuracy of the models (Brownlee, 2017; Hyndman & Athanasopoulos, 2021).

2.1.2. Machine learning methods

Motivated to improve the drawbacks of old statistical methods, various machine leads are proposed and developed along with the speed of technology. They identify underlying patterns and derive "arbitrary nonlinearity" from the data (Wang & Liu, 2021). Much research has been done on fundamental to extreme machine learning and artificial intelligence models (Aksoy et al., 2012, Ren et al., 2016, Ren et al., 2020)). However, aiming for the not-so-complex models for Couple TX, we focus only on fundamental machine learning approaches.

Of all machine learning methods, regression trees are those mainly used algorithms. These supervised learning algorithms predict data by dividing it into smaller groups and fitting a simple for each subgroup (Breiman et al., 1984). The collective complex techniques, namely Random Forests and Extreme gradient boosting, have been applied in sales forecasting, including garment retailers (Singh et al., 2019; van Steenbergen & Mes, 2020; Guven et al., 2021; Spiliotis et al., 2020).

In recent years, Facebook Prophet, a library developed by Meta in 2017, has been known as a time-series forecast that can ease the special days added to the data. It is used to predict the demand of various industries, like city electricity (Reppucci, 2020), retail (Žunić et al., 2020), luxury fashion (Alfaro, 2019), and COVID-19 case trend (Gaur, 2020).

Although machine learning models are more effective and precise in non-linear data, they inevitably bear drawbacks. First, they require money and proper knowledge to implement a proper model. Moreover, it also takes a certain amount of time to run a forecasting task. Hence, it might be redundant in some cases.

2.2. Identifying Research Gaps

Even though much effort has been put into finding an accurate sales forecast for fashion retailers, the authors have found that few researchers focus on the difference between each store in its chain.

2.2.1. "One-size-fits-all" model

It is common knowledge that sales patterns may vary between locations and stores, even in the same brand. This situation is explained by consumer buying power, promotion strategies, etc. For example, customers in big cities tend to spend more than those in small cities, resulting in much higher sales. At the same time, sales may differ if the promotions are not applied in all stores but specific locations according to business plans. Hence, applying only one forecasting model can significantly different results amongst stores (Song, 2015). However, building one model for each specific store requires an enormous amount of time and money. Instead of applying complex techniques to improve accuracy for one model only, not many researchers pay attention to finding solutions for retail sales forecasts. One way to solve this problem is to use a cluster-based hybrid to reduce the degree of data heterogeneity (Chen & Lu, 2021). However, there is not much work done in the fashion industry.

2.2.2. Only focus on customer and product characteristics

When predicting sales of a store, it is easy to notice that more effort is put into analyzing and segmenting customer or product characteristics. In 2005, Thomassey & Fiordaliso cluster current retail sales data into groups of similar pattern items. Van Steenbergen & Mes (2020) apply same procedure. In 2021, Chen & Lu propose a clustered method for physical and non-physical channels. However, many other factors should be considered to evaluate store performance. They can be demographic aspects like store size, location (Lusch et al., 2015), level of competition (Ghosh, 1984), and promotions (Gauri et al., 2017).

2.3. Filling the Gaps

Having studied previous research and found the potential area to improve sales forecasting for a retail fashion chain, this thesis aims to tackle those limitations. Therefore, we will incorporate two approaches to improve model accuracy. In both approaches, we use grid-search to find out the best hyperparameters for each model.

First, we try to answer which model gives the most accurate predicted results, either statistical or machine learning. Here, we put the total sales of the whole chain and tune on. Then, with the conducted parameters, we will fit stores individually. We will also use and compare two statistical methods and three advanced techniques.

This thesis aims to improve forecasting accuracy by detecting similar patterns with clustering techniques in the second approach. We choose store characteristics such as store level and locations to manually group by as they are already recorded in the data. We also use the Dynamic Time Warping method, a machine learning technique, to do the clustering for a more objective result.

3. Theory

This section presents the theoretical background of the models in use, how they are estimated, and the process of evaluating their performance. It begins with a time-series forecast where we discuss the particularities of time-series data and the forecasting problem. Next, a brief introduction of various forecasting methods and models is presented. Later, we will discuss time-series clustering methods and their potential to improve forecast accuracy. Then, performance evaluation metrics and validation techniques to compare model performance are described. Finally, software packages and their authors employed in the study are listed for reference.

3.1. Time-series forecasting

A time series is a sequence of observations recorded at successive equally spaced points in time (Hyndman & Athanasopoulos, 2021). The data may include demand samples, temperature, earnings, profits, shipments, accidents, productivity, etc. Forecasting techniques developed for time-series data are based on the assumption that past values of the series are a good indicator of future values. This characteristic of time series makes it a common problem attracting significant interest in research (Parmezan et al., 2019; Makridakis et al., 2020; Makridakis et al., 2018). Time-series analysis has been used for various applications such as econometric forecasting (Ahmed et al., 2010), quality and process control (Naim & Mahara, 2018), sales forecasting (Brownlee, 2017; Papacharalampous et al., 2018; Pavlyshenko, 2019; Van Belle et al., 2021), health surveillance (Shih & Rajendran, 2019; Zhang et al., 2014), energy demand forecasting (Lai et al., 2020; Wang et al., 2018).

While analyzing a time series, it is helpful to decompose the data into components to see the underlying patterns more quickly: a trend-cycle component (sometimes called the trend for simplicity), a seasonal component, and a remainder component (containing anything else in the time series) (Hyndman & Athanasopoulos, 2021). According to these authors, additive and multiplicative decomposition are two common ways to decompose time-series data.

- Additive decomposition can be written as:

  $y_t = S_t + T_t + R_t$

  y_t: the data

  S_t: the seasonal component

  T_t: the trend-cycle component

  R_t: the remainder component

- Similarly, for multiplicative decomposition

  $y_t = S_t \times T_t \times R_t$

The additive decomposition is most appropriate if the magnitude of the seasonal fluctuations, or the variation around the trend-cycle, does not vary with the level of the time-series. When the variation in the seasonal pattern, or the variation around the trend cycle, appears to be proportional to the time series level, then a multiplicative decomposition is more appropriate.

3.2. Forecasting models

Generally, forecasting models can be categorized into two groups:

1. Traditional forecasting methods include Naive, MA (moving average), ETS (exponential smoothing), ARIMA (Autoregressive Integrated Moving Average), and their variants.
2. Advanced forecasting methods are based on machine learning and artificial intelligence principles such as Facebook Prophet, Random Forests, XGB (Extreme Gradient Boosting), Support Vector Regressor, and Neural Networks.

Following is a brief discussion of the models used in our analysis and their formulations.

3.2.1. Seasonal naive (SNAIVE)

This method is a variant of the naive method, also known as a random walk, the most basic and simplest forecasting method. To use this method, one simply repeats the last seen data. There is no need for further understanding of the data or other variables.

For seasonality in data, the seasonal naive forecasting method obtains the forecast for time $T + h$ by the formula:

m: the seasonal period

k: the integer part of

(Hyndman & Athanasopoulos, 2021)

As an example, forecasts of the next n Mondays are the data seen last Monday, forecasts of the next n Januaries are the data seen last January and so on.

3.2.2. Auto-Regressive Integrated Moving Average

As the name suggests, ARIMA models focus on the correlation between the lagged time-series observations (Nau, 2020). They try to capture the correlation between the current and past values that often appear in the time-series data. This thesis employs its variant, Seasonal ARIMA with Exogenous variables (SARIMAX), to include seasonality and external information during the modeling process.

An ARIMA model of order (p, d, q), ARIMA(p, d, q), is a combination of three operations:

1. The autoregressive, AR(p), captures the autocorrelation between the present and the past values.
2. The moving average, MA(q), captures the past forecast errors of the model.
3. The integration (d) stands for the degree of differencing. Differencing operation comprises taking the difference between consecutive observations.

ARIMA models alone cannot capture the seasonal effect if a time series exhibits it. In that case, we use an extension of the ARIMA model called seasonal ARIMA (SARIMA). Four additional parameters (P, D, Q)m are added to the model. While (p, d, q) describes the non-seasonal part of a time series, (P, D, Q)m describes the seasonal part. The parameters “P” and “Q”, similar to “p” and “q”, capture the seasonal autoregressive and seasonal moving average behavior. “D” is the seasonal differencing, “m” indicates the seasonal period. For example, m = 7 for weekly seasonality. SARIMA can also be extended to include external information and become SARIMAX (X stands for exogenous variable).

An essential prerequisite for ARIMA models is that the time series should be stationary (De Gooijer & Hyndman, 2006; Hyndman & Athanasopoulos, 2021; Nau, 2020). A time series is stationary when its statistical properties (mean, variance and covariance) do not change over time (Seabold & Josef, 2010). It is recommended to use the Augmented Dickey-Fuller (ADF), and the Kwiatkowski-Phillips-Schmidt-Shin (KPSS) tests to check for stationarity. We can make a non-stationary series stationary series by differencing and seasonal differencing.

More details about manual order selection for SARIMAX models can be found in “Statistical forecasting: notes on regression and time series analysis” by Nau, 2020. In this study, the stepwise algorithm suggested in (Hyndman & Khandakar, 2008) is applied using the pmdarima package in Python (Smith et al., 2017) to find the optimal order. The next step is to estimate the model’s parameters using maximum likelihood estimation, which is adopted from (Brockwell & Davis, 2016).

The parameters of the ARMA(p,q) model can be estimated by maximizing the Gaussian likelihood:

$$L\left(\boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{\theta}, \sigma^2\right) = (2\pi)^{-n/2}(\det \Gamma_n)^{-1/2} \exp\left\{-\frac{1}{2}\left(\mathbf{Y} - X\boldsymbol{\beta}\right)' \Gamma_n^{-1}\left(\mathbf{Y} - X\boldsymbol{\beta}\right)\right\}$$

where $\Gamma_n(\phi, \theta, \sigma^2)$ is the covariance matrix of **W** = **Y** − Xβ. Since $\{W_t\}$ is an ARMA(p, q) process with parameters (φ, θ, σ^2), the maximum likelihood estimators $\hat{\beta}$, $\hat{\Phi}$, and $\hat{\theta}$ are found by minimizing

$$\ell(\boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{\theta}) = \ln\left(n^{-1} S(\boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{\theta})\right) + n^{-1} \sum_{t=1}^{n} \ln r_{t-1}$$

where

$$S(\boldsymbol{\beta}, \boldsymbol{\phi}, \boldsymbol{\theta}) = \sum_{t=1}^{n} \left(W_t - \hat{W}_t\right)^2 / r_{t-1}$$

$\{\widehat{W}_t\}$ is the best one-step predictor of $\{W_t\}$, and $r_{t-1}\sigma^2$ is its mean squared error. The function $l(\beta, \Phi, \theta)$ can be expressed in terms of the observations $\{Y_t\}$ and the parameters β, φ, and θ using the innovations algorithm and minimized numerically to give the maximum likelihood estimators, $\hat{\beta}$, $\hat{\phi}$ and $\hat{\theta}$. The maximum likelihood estimator of σ^2 is then provided by $\hat{\sigma}^2 = S(\hat{\beta}, \hat{\phi}, \hat{\theta})/n$.

An extension of an iterative scheme, proposed by Cochrane and Orcutt (1949) for the case q = 0, simplifies the minimization considerably. It is based on the observation that for fixed φ and θ, the value of β that minimizes $l(\beta, \Phi, \theta)$ is $\hat{\beta}_{GLS}(\Phi, \theta)$, which can be computed algebraically instead of by searching numerically for the minimizing value. The scheme is as follows.

(i) Compute $\hat{\beta}_{OLS}$ and the estimated residuals $Y_t - x_t'\hat{\beta}_{OLS}, t = 1, \ldots, n$.

(ii) Fit an ARMA(p, q) model by maximum Gaussian likelihood to the estimated residuals.

(iii) For the fitted ARMA model compute the corresponding estimator $\hat{\beta}_{GLS}$.

(iv) Compute the residuals $Y_t - x_t'\hat{\beta}_{GLS}, t = 1, \ldots, n$, and return to (ii), stopping when the estimators have stabilized.

If $\{W_t\}$ is a causal and invertible ARMA process, then under mild conditions on the explanatory variables X_t, the maximum likelihood estimates are asymptotically multivariate normal. In addition, the estimated regression coefficients are asymptotically independent of the estimated ARMA parameters.

The large-sample covariance matrix of the ARMA parameter estimators, suitably normalized, has a complicated form involving both the regression variables and X_t the covariance function of $\{W_t\}$. Therefore, it is convenient to estimate the covariance matrix as $-H^{-1}$, where H is the Hessian matrix of the empirical log-likelihood evaluated at its maximum.

3.2.3. Facebook Prophet

Facebook's open-source forecasting tool, Prophet, is an additive regression model with interpretable parameters. It uses a decomposable time-series model comprising three main components: trend, seasonality, and holidays. It allows the user to intuitively pick all the components related to the forecasting problem and effortlessly make the required changes by tuning the parameters. It works well on a time series showing multiple seasonal effects in the historical data, is robust to missing data and shifts in trend, and typically handles outliers well (Taylor & Letham, 2017).

According to its authors (2017, p.7), "while we give up some critical inferential advantages of using a generative model such as an ARIMA, this formulation provides several practical advantages:

- Flexibility: We can easily accommodate seasonality with multiple periods and let the analyst make different assumptions about trends.
- Unlike with ARIMA models, the measurements do not need to be regularly spaced, and we do not need to interpolate missing values, e.g., from removing outliers.
- Fitting is very fast, allowing the analyst to explore many model specifications interactively, for example, in a Shiny application (Chang et al., 2015).

- The forecasting model has easily interpretable parameters that the analyst can change to impose assumptions on the forecast. Moreover, analysts typically do have experience with regression and can easily extend the model to include new components."

Prophet's formulation and estimation process follow, as adopted from Taylor & Letham, 2017:

$g(t)$: the growth function modeling the trend

: seasonality term

: holiday effects

ϵ_t: error term

The Trend Model

One can choose between Prophet's two options for modeling the trend: logistic growth and piecewise linear trends. In the former case, suppose there are S change points at times s_j, $j = 1, \cdots, S$, $\boldsymbol{\delta} \in \mathbb{R}^S$ is a vector of rate adjustments where σ_j is the change in rate occurs at time s_j, k is the base growth rate. The rate at any time t is then $k + \sum_{j:t>s_j} \delta_j$. This formula is represented more cleanly by defining a vector $\mathbf{a}(t) \in \{0, 1\}^S$ such that

$$a_j(t) = \begin{cases} 1, & \text{if } t \geq s_j, \\ 0, & \text{otherwise.} \end{cases}$$

The rate at time t is then $+a(t)^{T\delta}$. When the rate k is adjusted, the offset parameter m must also be adjusted to connect the endpoints of the segments. The correct adjustment at change point j is easily computed as

$$\gamma_j = \left(s_j - m - \sum_{l<j} \gamma_l \right) \left(1 - \frac{k + \sum_{l<j} \delta_l}{k + \sum_{l\leq j} \delta_l} \right)$$

The piecewise logistic growth model is then

$$g(t) = \frac{C(t)}{1 + \exp(-(k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})(t - (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})))}$$

Analysts can adjust the expected capacities of the system at any point in time C(t) to suit their prior knowledge.

Suppose the data do not exhibit saturating growth. In that case, a piecewise linear trend model is more appropriate

$$g(t) = (k + \mathbf{a}(t)^\intercal \boldsymbol{\delta})t + (m + \mathbf{a}(t)^\intercal \boldsymbol{\gamma})$$

All terms are similar to those in the logistic growth model, except for the adjustment term to make the function continuous:

$$\gamma_j = -s_j \delta_j$$

The Seasonality Term

In Prophet, Fourier series is applied to model multi-period seasonality. Let P be the regular period expected in the data. Arbitrary smooth seasonal effects can be approximated with a standard Fourier series (without an intercept because a trend component is simultaneously fitted):

$$s(t) = \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi nt}{P}\right) + b_n \sin\left(\frac{2\pi nt}{P}\right) \right)$$

Fitting seasonality requires estimating the 2N parameters $\beta = [a_1, b_1, \cdots, a_N, b_N]^\intercal$. This fitting is done by constructing a matrix of seasonality vectors for each value of t in our historical and future data, for example, with yearly seasonality and N = 10,

$$X(t) = \left[\cos\left(\frac{2\pi(1)t}{365.25}\right), \ldots, \sin\left(\frac{2\pi(10)t}{365.25}\right) \right]$$

The seasonal component is then $s(t) = X(t)\beta$ and it takes $\beta \sim Normal(0, \sigma^2)$ to impose a smoothing prior on the seasonality.

Holidays and Events

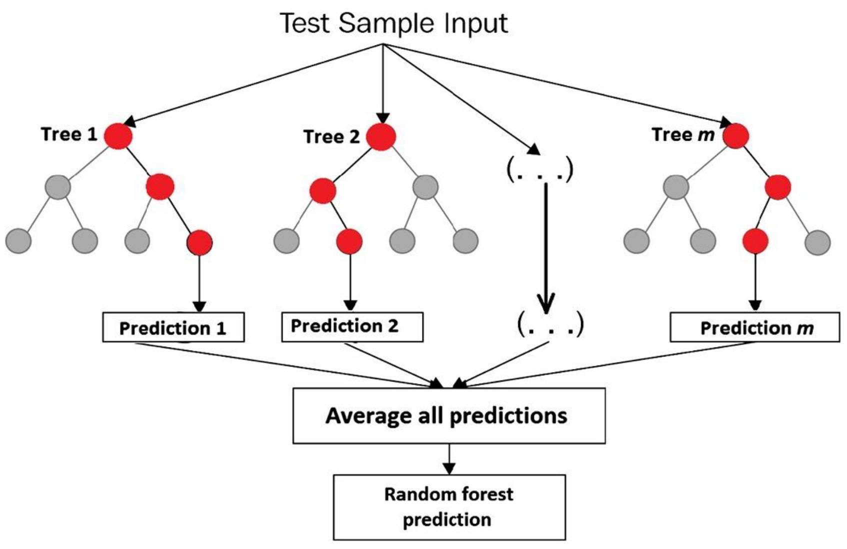
Holidays and events often cause disruptions in business time series and are not always fixed to particular calendar days (e.g. Lunar New Year). It is important to model them correctly, but this is not an easy task. The user of Prophet is given the option of defining their list of events during the time they are modeling. For each holiday i, let D_i be the set of past and future dates for that holiday. An indicator function is added to represent whether time t is during holiday i, and assign each holiday a parameter κ_i which is the corresponding change in the forecast. This practice is done similarly as seasonality by generating a matrix of regressors $Z(t) = [1(t \epsilon D_1), \ldots, 1(t \epsilon D_L)]$ and taking $h(t) = Z(t)k$. As with seasonality, we use a prior $K \sim Normal(0, \sigma^2)$. We include additional parameters to treat each of the days in the window around the holiday as a holiday itself to account for the days surrounding the holiday.

3.2.4. Random Forests (RF)

Random Forests regression is an implementation of Regression Trees (a decision tree where the target variable's values are continuous) where it combines multiple trees, each one depending on the values of a random vector sampled independently and with the same distribution (Breiman, 2001). The method's accuracy depends on the size of the forest and the strength and correlation of the individual trees. Given that Random Forests averages the predictions of multiple regression trees, it is more robust to noise and less likely to over-fit on the training data. An illustration of the model's operations (Afzal et al., 2020) is given in Figure 3.1.

Figure 3.1

The Random Forests Prediction Algorithm



Note. Random Forests relies on the bagging mechanism. Every decision tree is trained on separate data training sets (prepared by replacement), and their predictions are merged to obtain a single optimal solution. From "Response surface analysis, clustering, and Random Forests regression of pressure in suddenly expanded high-speed aerodynamic flows" Afzal et al., 2020.

A Random Forests operates by constructing several binary decision trees fitted using bootstrap samples during training time. The output is obtained by aggregating over the ensemble, which is the mean of the prediction of all the trees (Dudek, 2015). This method is called bagging, which reduces the variance and overfitting issue, improving accuracy. It:

- Applied to observations to improve model performance because it decreases the variance of the model without increasing the bias: while the predictions of a single tree are highly sensitive to noise in its training set, the average of many trees is not, as long as the trees are not correlated.
- Applied to features to reduce the correlation of the trees in an ordinary bootstrap sample: if one or a few features are robust predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated.

To understand how RF learns patterns from data and makes predictions in a regression problem, we introduce the model's elements: regression trees and bagging. The following explanations and formulas are adopted from Hastie et al., 2009.

Regression trees

Tree-based methods partition the feature space into a set of regions and then fit a simple model (like a constant) in each one. A variable and split-point need to be specified to achieve the best fit. Let p be the number of input variables, N be the sample size, M be the number of regions to partition, and we model the response variable Y as a constant c_m in each region R_m:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m)$$

If our objective is to minimize the sum of squares loss function, $\sum(y_i - f(x_i))^2$, the best $\hat{c}_m$ is just the average of $y_i$in region R_m:

$$\hat{c}_m = \text{ave}(y_i|x_i \in R_m)$$

Because finding the best binary partition in terms of minimum sum of squares is generally computationally infeasible, a greedy algorithm is used instead. Starting with all of the data, consider a splitting variable j and split point s, and define the pair of half-planes

$$R_1(j, s) = \{X|X_j \leq s\} \text{ and } R_2(j, s) = \{X|X_j > s\}$$

Then, we seek the splitting variable j and split point s that solve

$$\min_{j,\ s}\left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2\right].$$

For any j and s, the inner minimization is solved by

$$\hat{c}_1 = \text{ave}(y_i|x_i \in R_1(j, s)) \text{ and } \hat{c}_2 = \text{ave}(y_i|x_i \in R_2(j, s))$$

Having found the best split, we partition the data into the two resulting regions and repeat the splitting process on each of the two regions. Then this process is repeated in all of the resulting regions.

Bootstrap aggregating

The bootstrap is a general tool for assessing and improving statistical accuracy. Let $\mathbf{Z} = (z_1, z_2, \cdots, z_N)$ be a training set of size N where $z_i = (x_i, y_i)$. The basic idea is to draw randomly N-sized datasets with replacements from **Z** B times, producing B bootstrap datasets. A model is fitted to each bootstrap dataset, obtaining the prediction

$\hat{f}(x)$ at input x. Bootstrap aggregation or bagging averages this prediction on all bootstrap samples, reducing its variance.

Random Forests algorithm

Random Forests (Breiman, 2001) is a substantial modification of bagging that builds an extensive collection of de-correlated trees and then averages them (Hastie et al., 2009). The main goal is to improve the variance reduction of bagging by reducing the correlation between the trees without increasing the variance too much. This goal is achieved in the tree-growing process through random selection of the input variables as shown in Algorithm 3.1.

Algorithm 3.1

Random Forests for Regression

```
1. For b = 1 to B:
    a. Draw a bootstrap sample Z∗ of size N from the training data.
    b. Grow a random-forest tree $T_b$ to the bootstrapped data, by recursively
       repeating the following steps for each terminal node of the tree, until the
       minimum node size $n_{min}$ is reached.
          i. Select m variables at random from the p variables.
         ii. Pick the best variable/split-point among the m.
        iii. Split the node into two daughter nodes.
2. Output the ensemble of trees $\{T_b\}_1^B$
```

To predict a new point x: $\hat{f}_{\mathrm{rf}}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$

The typical value for m is $p/3$ as recommended by Breiman, 2001. The smaller m is, the more reduction is achieved in the correlation between any pair of trees, decreasing the variance of the average.

3.2.5. Extreme gradient boosting (XGB)

Extreme gradient boosting (XGB) is another implementation of regression trees where the Gradient Boosting framework is applied. Designed to be highly efficient, flexible, and portable, XGB provides a parallel tree boosting that solves many data science problems quickly and accurately (Chen & Guestrin, 2016). In XGB, trees are built one at a time when each new tree in the sequence tries to correct errors made by the previously built tree. In the original paper, Chen & Guestrin listed its features as follows

- XGB is almost ten times faster than the other boosting techniques because of parallel learning and distributed computing.
- It also includes a variety of regularization (L1 and L2), which reduces overfitting and improves overall performance. Therefore, it is also called a "regularized boosting" technique.
- It can manage sparse data by using a sparsity-aware split finding algorithm.
- It can handle weighted data efficiently by using distributed weighted quantile sketch algorithm.
- It uses the hardware resources optimally by system cache optimization.
- It employs out-of-core computation, enabling a primary computer to process massive datasets that do not fit in the memory.

Besides, XGB is remarkably flexible for tuning. Many hyperparameters can be configured to increase the performance of the model. That has helped it become the winning solution in many machine learning competitions like Kaggle (Chen & Guestrin, 2016).

To better understand how XGB builds trees and makes predictions, Chen & Guestrin (2016) suggest calling it regularized gradient boosting. While the algorithm's "extreme" component has more to do with managing computing resources, which is not the focus of this study, we attempt to clarify how gradient boosting is applied uniquely. It is a machine learning technique for regression and classification problems that optimizes a collection of weak prediction models, usually CARTs (classification and regression trees), to build an accurate and reliable predictor. XGB further improves upon the base gradient boosting framework through systems optimization (parallelization, tree pruning, hardware optimization) and algorithmic enhancements (regularization, sparsity awareness, weighted quantile sketch, cross-validation) (Morde, 2019). Among these, tree pruning, regularization and cross-validation are most relevant in our regression problem.

XGB as Regularized Gradient Boosting

Gradient boosting gives a prediction model in the form of an ensemble of weak learners, typically decision trees (Hastie et al., 2009). XGB uses a more regularized model formalization to control over-fitting, which gives it better performance. The following is how the model is estimated, as adopted from the original documentation by the author, Tianqi Chen (2014).

The ensemble model is formularized as

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), f_k \in \mathcal{F}$$

where K is the number of trees, f_k is a function in the functional space F, and F is the set of all possible CARTs.

The trees are trained by optimizing the objective function:

$$\text{obj} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \omega(f_i)$$

Where $\sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)})$ is the differentiable loss function, and $\sum_{i=1}^{t} \omega(f_i)$ is the regularization term.

The functions f_i are learned by applying an additive strategy: fix what we have learned, and add one new tree at a time. Let the prediction value at step t be $\hat{y}_i^{(t)}$, we have

$$\begin{aligned}
\hat{y}_i^{(0)} &= 0\\
\hat{y}_i^{(1)} &= f_1(x_i) = \hat{y}_i^{(0)} + f_1(x_i)\\
\hat{y}_i^{(2)} &= f_1(x_i) + f_2(x_i) = \hat{y}_i^{(1)} + f_2(x_i)\\
&\dots\\
\hat{y}_i^{(t)} &= \sum_{k=1}^{t} f_k(x_i) = \hat{y}_i^{(t-1)} + f_t(x_i)
\end{aligned}$$

At each step, the tree that optimizes the objective function is added:

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^{n}(y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)))^2 + \sum_{i=1}^{t}\omega(f_i)\\
&= \sum_{i=1}^{n}[2(\hat{y}_i^{(t-1)} - y_i)f_t(x_i) + f_t(x_i)^2] + \omega(f_t) + \text{constant}
\end{aligned}$$

If the loss function is mean squared error (MSE), the objective becomes

$$\begin{aligned}
\text{obj}^{(t)} &= \sum_{i=1}^{n}(y_i - (\hat{y}_i^{(t-1)} + f_t(x_i)))^2 + \sum_{i=1}^{t}\omega(f_i)\\
&= \sum_{i=1}^{n}[2(\hat{y}_i^{(t-1)} - y_i)f_t(x_i) + f_t(x_i)^2] + \omega(f_t) + \text{constant}
\end{aligned}$$

In this case, we take the Taylor expansion of the loss function up to the second-order:

$$\text{obj}^{(t)} = \sum_{i=1}^{n}[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \omega(f_t) + \text{constant}$$

Where g_i, the gradients, and h_i, the hessians, are defined as

$$\begin{aligned}
g_i &= \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})\\
h_i &= \partial^2_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})
\end{aligned}$$

After we remove all the constants, the specific objective at step t becomes

$$\sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \omega(f_t)$$

To define the regularization term, first we need to refine the definition of the tree $f(x)$ as

$$f_t(x) = w_{q(x)}, w \in R^T, q : R^d \to \{1, 2, \cdots, T\}$$

Here w is the vector of scores on leaves, q is a function assigning each data point to the corresponding leaf, and T is the number of leaves. Then, the regularization term is

$$\omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

After reformulating the tree model, we can write the objective value with the t-th tree as

$$\begin{aligned}\text{obj}^{(t)} &\approx \sum_{i=1}^{n}[g_i w_{q(x_i)} + \frac{1}{2}h_i w_{q(x_i)}^2] + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2 \\ &= \sum_{j=1}^{T}[(\sum_{i\in I_j} g_i)w_j + \frac{1}{2}(\sum_{i\in I_j} h_i + \lambda)w_j^2] + \gamma T\end{aligned}$$

Where $I_j = \{i|q(x_i) = j\}$ is the set of indices of data points assigned to the j-th leaf. By defining $G_j = \sum_{i\in I_j} g_i$ and $H_j = \sum_{i\in I_j} h_i$, the objective could be further compressed as

$$\text{obj}^{(t)} = \sum_{j=1}^{T}[G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2] + \gamma T$$

In this equation, w_j are independent concerning each other. The form $G_j w_j + \frac{1}{2}(H_j + \lambda)w_j^2$ is quadratic. The best w_j for a given structure $q(x)$, and the best objective reduction we can get are

$$\begin{aligned}w_j^* &= -\frac{G_j}{H_j + \lambda} \\ \text{obj}^* &= -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j + \lambda} + \gamma T\end{aligned}$$

The last equation measures how good a tree structure $q(x)$ is. Although it is ideal for enumerating all possible trees and picking the best one, optimizing one level of the tree at a time is more practical. Specifically, we try to split a leaf into two leaves, and the score it gains is

$$Gain = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

This formula can be decomposed as 1) the score on the new left leaf, 2) the score on the new right leaf, 3) the score on the original leaf 4) regularization on the additional leaf. If the gain is smaller than γ, it would be better not to add that branch. This technique is what the pruning technique in tree-based models is. To efficiently search for an optimal split, we sort all instances first. Then, the left to right scan is sufficient to calculate the structure score of all possible split solutions and the best split can be found efficiently.

XGB penalizes more complex models through regularization terms to prevent overfitting. In addition to λ and γ as presented above, α (L1 regularization on weights) can also be included in the regularization term $\sum_{i=1}^{t} \omega(f_i)$ to control shrinkage and enable feature selection. Adjusting the rate a model learns patterns in data, η, is yet another way to improve XGB's performance.

3.2.6. Hyperparameters optimization

In machine learning model training, we first need to define the learning process by setting the model's hyperparameters. They control several learning aspects such as speed, complexity, and quality. Then, we feed the training data to the model and let it compute model parameters. These will directly determine the values of the predictions.

Different algorithms have different sets of hyperparameters to tune. It is essential to understand the effect of each parameter to tune it accordingly. Interested readers can find more information in the respective documentation of the models (Chen & Guestrin, 2016; Breiman, 2001; Taylor & Letham, 2017). For instance, some of XGB's most influential hyperparameters include min_child_weight, learning_rate, n_estimator, reg_alpha, and subsample, to name a few.

To find the best set of hyperparameters, we can apply two methods

1. Random search: this method evaluates a given number of random combinations by selecting a random value for each hyperparameter at every iteration. It is suitable at the early stage when we want to try large numbers of hyperparameters (Pedregosa et al., 2011).
2. Grid search: the result of a random search informs us about a narrower space to search for the optimal values of hyperparameters. To improve further, we can carry out an exhaustive (grid) search of all hyperparameter combinations in that search space to pinpoint the final values (Pedregosa et al., 2011).

It is noteworthy that hyperparameter tuning can lead to overfitting where a model fits the training set well but performs poorly on a test set. Hence, its generalizability is compromised. One way to avoid this problem is using the time-series cross-validation technique described in subsection 3.4.6.

3.3. Clustering method

Clustering is a data mining technique where similar data are placed into related or homogeneous groups without advanced knowledge of the groups' definitions (Rai & Singh, 2010). In other words, clustering is trying to group data points that share similar patterns into one group while attempting to classify other data points that do not share similarities into different groups. Time-series clustering is a particular type of clustering, as the data to classify is dynamic and temporal data. One of the main applications of time-series clustering is pattern discovery (Aghabozorgi et al., 2015). Our research interest is the analysis of the patterns of sales of different stores within a company.

In theory, there are three different ways to cluster time series, namely shape-based, feature-based, and model-based (Aghabozorgi et al., 2015). In our research, with limited time and capability, we will only test the effectiveness of the shape-based algorithms. These algorithms are adapted from conventional clustering methods by modifying the distance or similarity with an appropriate one for the time series. Specifically, one popular distance measurement used to compare time series is Dynamic Time Warping (DTW) (Chu et al., 2002). DTW measures the similarity between two temporal sequences that do not align precisely in the index. This research will use the K-means clustering algorithm with DTW mentioned in a global averaging method for dynamic time warping, with applications to clustering (Petitjean et al., 2011). The DTW process calculates the distance between the time series with the cluster centroids. A centroid is an average sequence from a group of time series in DTW space. Then, the DTW Barycenter Averaging (DBA) algorithm minimizes the sum of squared DTW distance between the centroid and the series in the cluster to determine the final centroid location in the semantic space.

In this thesis, we want to test the hypothesis that grouping stores into clusters where the stores would share similar sales patterns and trends would benefit the performance of the forecasting model. In section 2.3, size and location are significant factors for store performance. After having the clusters of all stores using the K-Means clustering with DTW for the time series algorithm provided by tslearn (Tavenard et al., 2020), we will tune the best model for each cluster and then use it to predict the sales of the stores in the same cluster.

3.4. Performance evaluation

We will use time-series cross-validation to compute error metrics to ensure the best model is applicable for future observations. A forecast "error" is the difference between an observed value and its forecast. Here "error" does not mean a mistake; it means the unpredictable part of an observation (Hyndman & Athanasopoulos, 2021). It can be written as:

y_{T+h}: observed value

$\hat{y}_{T+h|T}$: forecast

In-sample error is calculated using training data, given by, $\{y_1, \cdots, y_T\}$ and out-of-sample error is calculated based on test data, given by

It is usually necessary to employ various metrics to quantify different aspects of model performance. Each metric has its advantages and disadvantages and is appropriate for different contexts. The metrics used in this thesis include

- Root mean square error (RMSE)
- Mean absolute error (MAE)
- Mean absolute percentage error (MAPE)
- Mean absolute scaled error (MASE)

3.4.1. RMSE

RMSE is a standard performance measure for regression problems. It gives an idea of how much error the system typically makes in its predictions, with a higher weight for large errors (Géron, 2019).

It is given by

(Hyndman & Athanasopoulos, 2021)

3.4.2. MAE

MAE is very similar to RMSE, except it is less sensitive to outliers. Its formula is

(Hyndman & Athanasopoulos, 2021)

MAE and RMSE are scale-dependent measures, meaning that the series in question needs to have the same units as in our case. They reveal the absolute magnitude of forecast errors.

3.4.3. MAPE

An advantage of MAPE is that it is unit-free. Thus, it allows meaningful comparison when error magnitudes do not matter. To compute MAPE, we use the following formula

$$\text{MAPE} = \text{mean}(|100e_t/y_t|)$$

(Hyndman & Athanasopoulos, 2021)

The fact that MAPE is a percentage error measure makes it easy to understand and communicate. In addition, it provides context to model performance. Thus, it is used as a primary objective in various comparisons in our analysis.

3.4.4. MASE

Hyndman & Koehler (2006) propose scaled errors as an alternative to using percentage errors when comparing forecast accuracy across series with different units. They propose scaling the errors based on the training MAE from a simple forecast method. For seasonal time series, a scaled error can be defined using seasonal naïve forecasts:

$$q_j = \frac{e_j}{\frac{1}{T-m}\sum_{t=m+1}^{T}|y_t - y_{t-m}|}$$

(Hyndman & Athanasopoulos, 2021)

To calculate MASE, we use the formula

$$\text{MASE} = \text{mean}(|q_j|)$$

(Hyndman & Athanasopoulos, 2021)

MASE is helpful to comprehend the degree of improvement (if any) over a simple method. A quick look at it will tell how much value a method is adding on top of, for example, repeating past values.

3.4.5. Temporal train/test split

According to Hyndman & Athanasopoulos (2021), when choosing models, it is common practice to separate the available data into two portions, training and test data, where the training data is used to estimate any parameters of a forecasting method and the test data is used to evaluate its accuracy.

For example, our dataset includes daily observations from 2017-09-01 to 2021-01-31, and we want to make forecasts for 7 days ahead. Our train set will consist of data points from the first day to 2021-01-24, and the test set is the final 7 days (2021-01-25 to 31).

3.4.6. Time-series cross-validation

This is a more sophisticated version of the train/test split where we keep several continuous test sets (Hyndman & Athanasopoulos, 2021). We continue the previous example and extend it to include 4 test sets, also folds, with 7-days "jumps." Table 3.1 shows four partitions. Subsequently, a model will be fitted on the train set and evaluated on the test set 4 times. The final score is the average score of all evaluations. As a result, the scoring process is more robust than the simple train/test split.

Table 3.1

Datetime index of 4-Folds time-series cross-validation

Fold	Train set	Test set
1	2017-09-01 to 2021-01-03	2021-01-04 to 2021-01-10
2	2017-09-08 to 2021-01-10	2021-01-11 to 2021-01-17
3	2017-09-15 to 2021-01-17	2021-01-18 to 2021-01-24
4	2017-09-22 to 2021-01-24	2021-01-25 to 2021-01-31

3.5. Software packages

All analysis is carried out in Python programming language with the help from various packages:

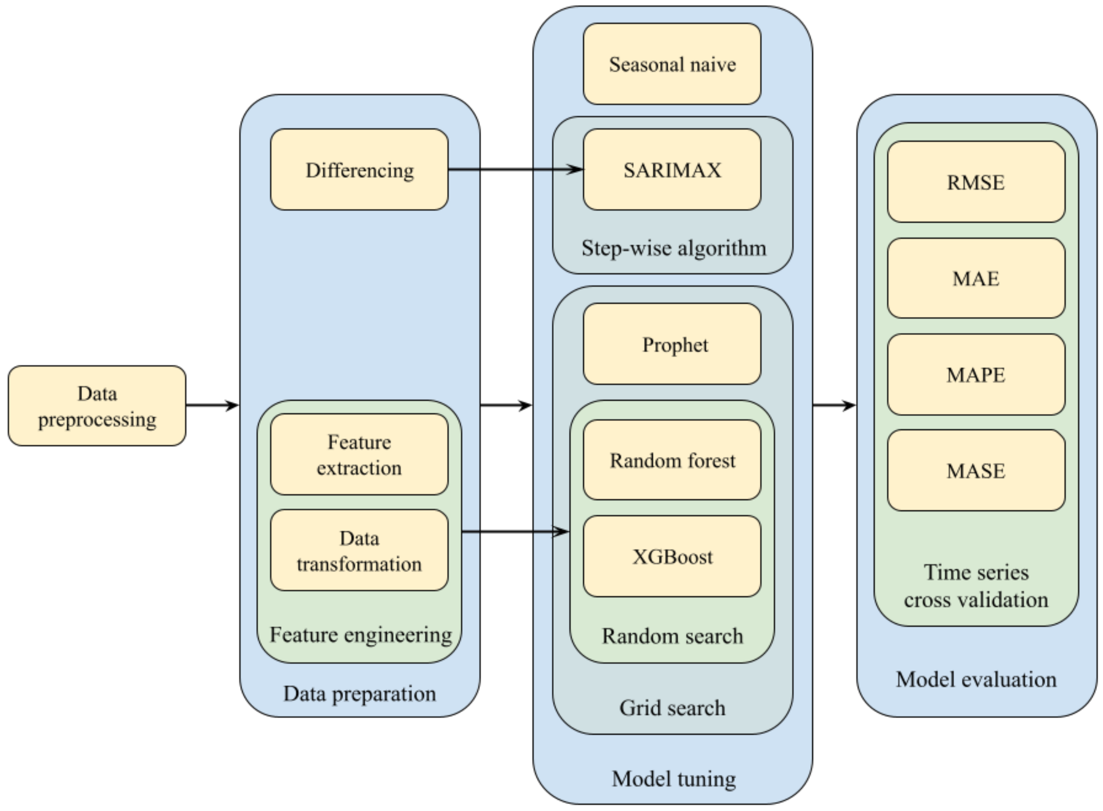
- pandas (Reback et al., 2022)
- numpy (Harris et al., 2020)
- statsmodels (Seabold & Josef, 2010)
- pmdarima (Smith & others, 2017)
- plotly (Plotly Technologies Inc., 2015)
- sktime (Löning et al., 2019; Löning et al., 2022)
- sklearn (Pedregosa et al., 2011)
- xgboost (Chen & Guestrin, 2016)
- tslearn (Tavenard, Romain, et al., 2020)

4. Experimental setup

This section describes how we set up our experiment to find answers to the research questions. The first subsection contains information about the dataset and preparation steps for modeling. Secondly, we specify how the sales forecasting task is executed using different procedures for different types of models. Finally, time-series cross-validation techniques are employed to compute error metrics, which serve as a basis for model performance assessment. Figure 4.1 shows the summarized process.

Figure 4.1

Schematic diagram describing the forecasting process



4.1. Data description, preprocessing, and preparation

4.1.1. Endogenous variable

The original dataset consists of entries from August 2017. We choose to cut off at 2021-01-31 when the COVID pandemic's impacts started manifesting because those disruptions are out of the scope of our study. Raw data is extracted from the company's SQL database. It comes in the form of a table in which each row represents order, and each column contains details such as the order's value, time, store, customer, and channel. Relevant filtered to retain information about the actual order value (after subtracting promotions, discounts, etc.), the store from which the order originated, and the timestamp of the order. From our discussion with the company representative, orders whose value exceeds 20 million VND belong to the wholesale channel and should be excluded from our analysis. Then, we specify the correct data types for each column to ensure proper treatment.

The dataset contains 121 individual stores in total. Only 38 of those are selected for modeling based on a few criteria:

- Active until the last day of the specified date range (2021-01-31)
- Active for at least two years before 2021-01-31 (to ensure sufficient data for robust model fitting)
- Being a physical store (sales from online channels are not included in this study)

We then group this filtered data by store_id and resample to daily frequency by taking the sum. As a result, we obtain a long-form dataset with no missing values, with each row representing the sales value of a unique store on a day.

shows examples of this dataset. This dataset will be reshaped during the analysis to suit the task. Next, sales of the whole company is computed by summing up sales across all stores daily. Table 4.1 shows examples of this dataset. We will reshape this dataset during the analysis to suit the task.

Table 4.1

Examples from the Dataset of store daily sales

	date	store_id	sales
0	2017-08-07	307222	1.52765
1	2017-08-08	307222	0.00000
…	…	…	…
44632	2021-01-30	566792	1.62220
44633	2021-01-31	566792	9.35250

Next, sales of the whole company is computed by summing up sales across all stores daily. Figure 4.2 plotted sales value in the form of a time series. Its descriptive statistics are presented in Table 4.2.

Figure 4.2

Time plot of company sales

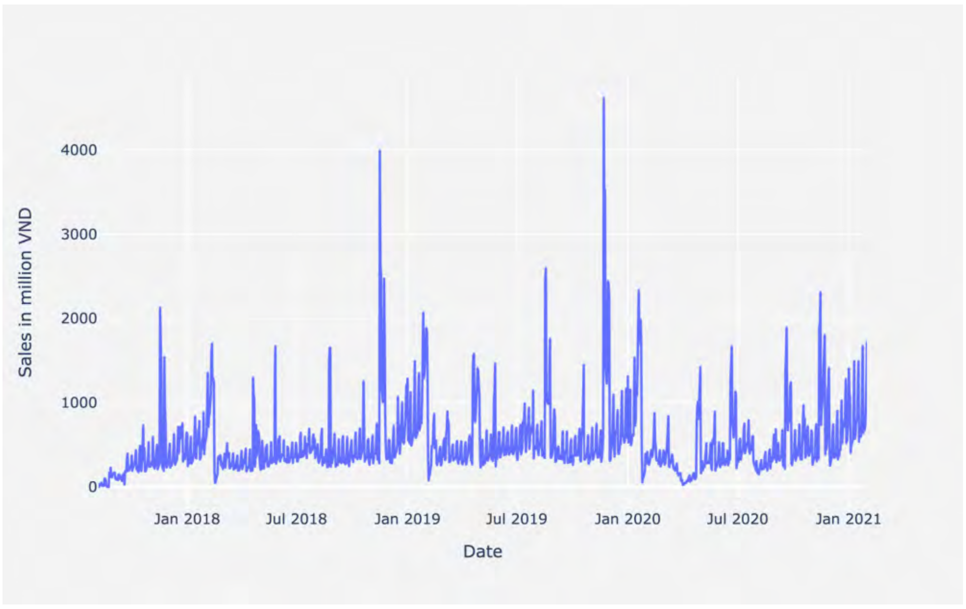


Table 4.2

Descriptive statistics of company Sales

count	1274.00
mean	524.92
std	428.62
min	0.00
25%	294.56
50%	391.87
75%	609.97
max	4617.47

4.1.2. Exogenous variables

From the interviews with the company's representative, their website, Facebook page, and the data itself, the following events are identified to have substantial impacts on sales:

- Lunar New Year holidays and International Labor Day (May 1): most stores are closed, and their sales are zero. Those that remain open have meager sales.
- Major sales-boosting promotions often precede holidays:
    - Vietnam Reunification day (April 30)
    - End of the school year (end of May)
    - New school year (end of August)
    - Vietnamese Women's Day (October 20)
    - Black Friday (4th Friday in November; 5th Friday if November 1st is a Friday)
    - End of year (Lunar calendar)

These days are treated as exogenous variables whose data type is boolean. In particular, the corresponding variables off_day and promo_day have values True on applicable days and False otherwise.

4.1.3. Feature engineering

Feature engineering "is a crucial step in the machine learning pipeline because the right features can ease the difficulty of modeling and enable the pipeline to output higher quality results" (Casari & Zheng, 2017). This subsection explains the process of engineering machine learning features from the endogenous variable.

Date-related features are generated based on the index of the dataset. These include:

- dayofweek: 0…6 representing Monday-Sunday
- dayofmonth: 1…31 representing days in a month
- dayofyear: 1…366 representing 366 days in a year (365 if not a leap year)
- weekofyear: 1…53 representing weeks in a year
- month: 1…12 representing 12 months in a year
- quarter: 1…4 representing four quarters in a year
- year: 1…5 representing the years 2017-2021

Lag features are derived from the previous data points of the target variable. For example, lag_1 is the sales figure from yesterday; lag_2 is the sales figure from the day before yesterday, and so on. Lag features are not available for the first few observations, and they will be treated as missing data. In addition, minimum lag features should be kept less than the corresponding forecasting horizons to avoid the problem of looking ahead. For example, if we plan to make forecasts for the next seven days, the latest data point available is lag_7 of day T+7.

Rolling mean is the historical average of the target variable. For instance, rolling_mean_2 is the average of 2 observations and is "rolled" forward in time. This feature is based on lag features to avoid the looking-ahead issue.

4.2. Model tuning

This subsection describes techniques particular to each model to find its optimal configuration (either order for SARIMAX or hyperparameters for Prophet, Random Forests, and XGB).

4.2.1. Seasonal Naïve (SNAIVE)

Seasonal naïve is a simple method and can be used to make forecasts directly from the cleaned data of each store. For example, the forecasted sales of a store on 2021-01-04 (Monday) is the store's mean of daily sales on all Mondays before 2021-01-04. Hence, seasonal naïve can be evaluated directly in the next subsection.

4.2.2. SARIMAX

Similar to seasonal naïve, SARIMAX does not require any data transformation in advance. However, the model's order, (p,d,q)(P,D,Q)m, must be specified. The function auto_arima from the package pmdarima (Smith & others, 2017) is applied to the whole company dataset to search for the optimal order. It uses the stepwise algorithm outlined in Hyndman & Khandakar (2008), which can be significantly faster than fitting all (or a random subset of) hyper-parameter combinations and is less likely to over-fit the model. The process includes the following steps:

- Use the KPSS test to determine d, the order of differencing
- Use the OCSB test to determine D, the order of seasonal differencing
- Fit models within predefined ranges of p, q, P, Q
- Return AICc of all fitted models
- Model with the lowest AICc is selected
- off_day and promo_day are also included as additional features in the regression operation

The optimal model is ARIMA(1,0,1)(0,1,1)[7] with intercept. Its AICc is 17593.089, as reported in Table 4.3, and all valid configurations. The order indicates a seasonal difference, non-seasonal AR(1) and MA(1) components, and a seasonal MA(1) component.

Table 4.3

Various SARIMAX Configurations And Corresponding AICc

Model	AICc
ARIMA(1,0,1)(0,1,1)[7] intercept	17593.089
ARIMA(0,0,2)(0,1,1)[7] intercept	17644.607
ARIMA(0,0,1)(0,1,1)[7] intercept	17868.152
ARIMA(1,0,1)(1,1,0)[7] intercept	17876.624
ARIMA(1,0,0)(1,1,0)[7] intercept	17876.735
ARIMA(1,0,1)(0,1,0)[7] intercept	18047.180
ARIMA(0,0,1)(1,1,0)[7] intercept	18127.536
ARIMA(0,0,1)(0,1,0)[7] intercept	18282.897
ARIMA(0,0,0)(0,1,1)[7] intercept	18303.364
ARIMA(0,0,0)(0,1,0)[7]	18663.766
ARIMA(0,0,0)(0,1,0)[7] intercept	18665.600

To illustrate the estimation process, historical daily sales of store 320264 are plotted in Figure 4.3 along with the series' ACF and PACF plots. The significant spikes at lags 1 and 7 in the ACF plot suggest non-seasonal MA(1) and seasonal MA(1) components, while the spike at lag 1 in the PACF plot points to an AR(1), hence the optimized model. We then fit the model to the training set using maximum likelihood estimation to estimate the parameters.

After the model is fitted, its residuals resemble white noise, as shown in Figure 4.4. The Ljung-Box test applied to the residuals yields a p-value of 0.126, which confirms that the residuals are similar to white noise, and we are ready to make forecasts on the test set

Figure 4.3

Historical sales of Store 320264

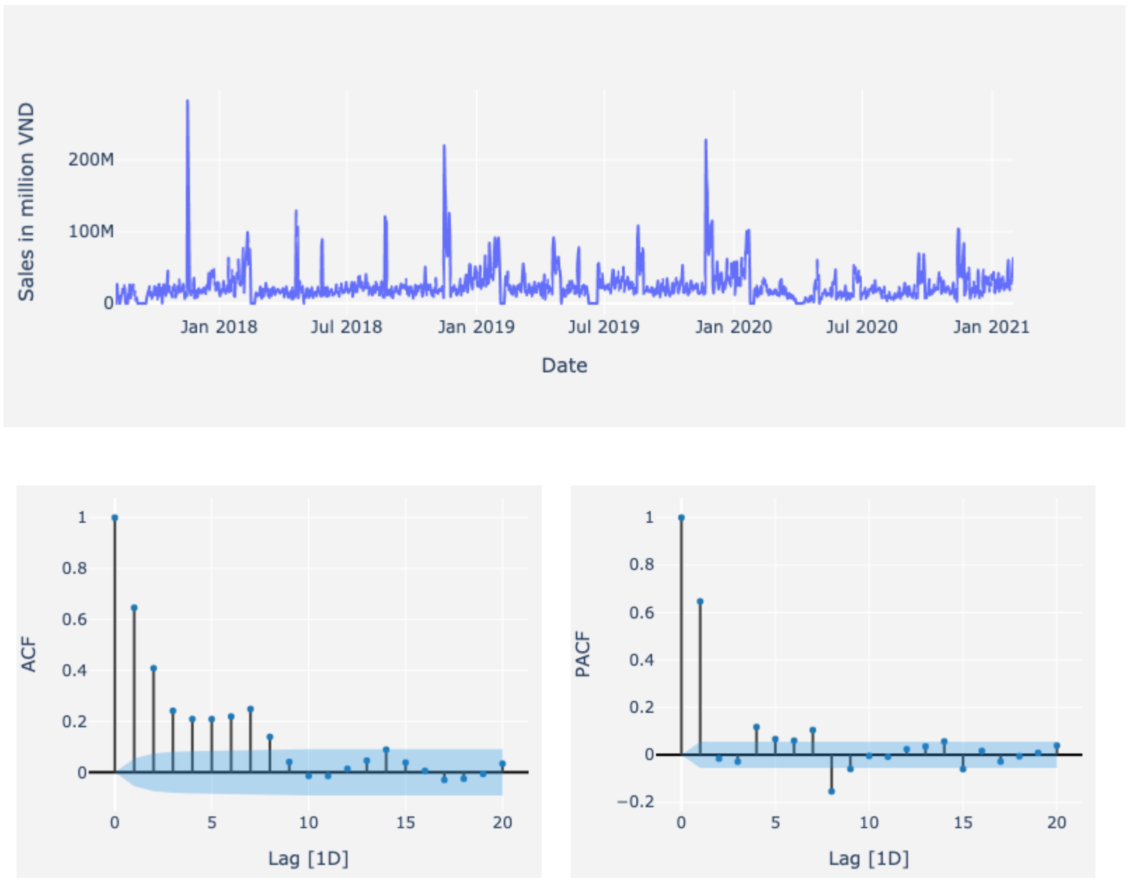
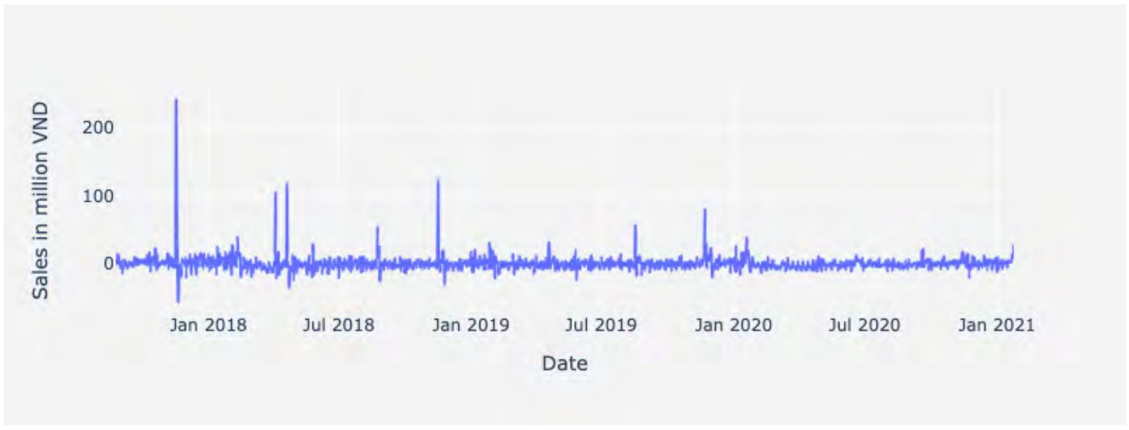
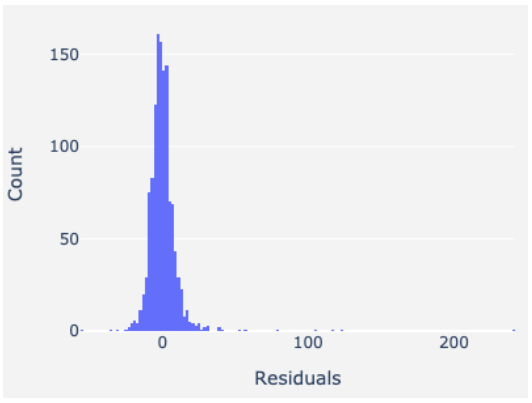
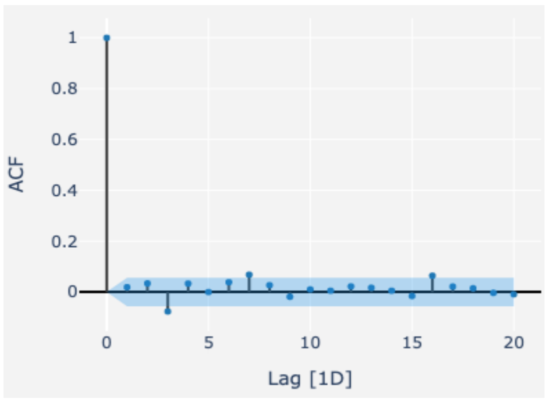


Figure 4.4

Residuals of model ARIMA(1,0,1)(0,1,1)[7] fitted on data of store 320264







4.2.3. *Facebook Prophet*

Facebook's Prophet, as a specialized forecast package, only requires input data to follow its format specification. Hence, we convert our dataset into a data frame with two columns, ds, and y, representing the date and the target variable, as shown in Table 4.4.

Table 4.4

Data formatted to PROPHET's requirements

	ds	y
0	2017-08-07	1.528
1	2017-08-08	0.000
2	2017-08-09	34.942
3	2017-08-10	20.152
4	2017-08-11	24.305
...	...	...
1262	2021-01-20	653.033
1263	2021-01-21	595.027
1264	2021-01-22	745.328
1265	2021-01-23	1207.594
1266	2021-01-24	1672.900

As recommended in (Taylor & Letham, 2017), the following hyperparameters are tuned to find their optimal values:

- changepoint_prior_scale: determines the flexibility of the trend, in particular, how much the trend changes at the trend changepoints.
- seasonality_prior_scale: controls the flexibility of the seasonality: a large value allows the seasonality to fit large fluctuations, and a small value shrinks the magnitude of the seasonality.
- holidays_prior_scale: controls flexibility to fit holiday effects.

A few other hyperparameters are also important but do not need tuning because they can be specified based on the characteristics of the data:

- seasonality_mode is set additive because the magnitude of seasonal fluctuations does not change over time.
- weekly_seasonality and yearly_seasonality are set to True as the data exhibits both types of seasonality.

A baseline score can be established by fitting a model with the default set of hyperparameters (seasonality_prior_scale = 10.0, holidays_prior_scale = 10.0, changepoint_prior_scale = 0.05). This setup results in a MAPE of 0.148 on the whole company dataset.

Next, the same dataset is fed to the grid-search algorithm to find the optimal hyperparameters. A random search is unnecessary since the total number of Prophet models to compare is only 64. The top 4 configurations have the same MAPE of 0.1 under changepoint_prior_scale of 0.1 and holidays_prior_scale of 0.01. This result is some improvement over the default settings. With further assistance from RMSE, we can select a seasonality_prior_scale of 10.

4.2.4. Random Forests (RF)

RF regressor was not designed initially as a forecaster, but the function *make_reduction* in Python package *sktime* can adapt it (Löning et al., 2019; Löning et al., 2022). In addition, feature engineering is necessary to realize the model's strengths.

The augmented dataset of the whole company is input into the Random Forests forecaster to begin the tuning process, consisting of two steps as described in 3.2.6: random search and grid search. While the main parameters to adjust are n_estimators and max_features, additional parameters can also be included to control the model's behavior further:

- n_estimators: dictates the number of trees in the forest. A larger value is better but longer to compute.
- max_features: controls the size of the random subsets of features to consider when splitting a node. Lower values reduce variance but increase bias.
- max_depth: is the longest path between the root node and the leaf node.
- min_samples_split: describes the minimum required number of observations in any given node to split.
- max_leaf_nodes: limits the maximum number of leaf nodes after splitting.
- min_samples_leaf: the minimum number of samples that should be present in the leaf node after splitting a node.

The possible values range is too large for an exhaustive grid search algorithm. Hence, we need to apply a random search for which we can specify the number of models to sample. Based on its result, more granular values are identified for grid searching to pinpoint the optimal hyperparameters. The whole process reduces MAPE from 0.4 of the default set of hyperparameters to 0.21 of the optimal hyperparameters.

4.2.5. Extreme gradient boosting (XGB)

Similar to RF, the input data for XGB needs to go through feature engineering before the model can be tuned or fitted. Random and grid searches are performed to find the optimal hyperparameters:

- n_estimators: the number of decision trees used in the ensemble. A larger value is better but takes longer to compute.
- learning_rate: controls each model's contribution to the ensemble prediction. Lower rates may require more decision trees in the ensemble.
- max_depth: represents the longest path between the root node and the leaf node.
- subsample: is the number of samples used to fit each tree.
- colsample_bytree: is the number of features used to fit each decision tree.

Early stopping is also used for XGB: more trees are added to the ensemble until the evaluation metric in the validation set stops improving. The tuning process applied to the whole company dataset decreases MAPE from 0.38 to 0.19.

4.3. Model evaluation

In the previous section, to find the optimal configuration of each model, they have gone through the hyperparameters optimization process applied to the whole company dataset. This section presents the evaluation of these optimal models.

For each model, time-series cross-validation is applied to each store's dataset to fit repeatedly, make forecasts and compute error metrics (RMSE, MAE, MAPE, MASE). A model's optimal configuration stays unchanged during this process, and only the coefficients are re-estimated for each store and cross-validation fold. The pseudo-code in Figure 4.5 illustrates this process more clearly. This nested loop is applied to all models, resulting in 760 sets of error metrics. These are further summarized and reported in section 5 to understand the models' overall performance better.

Figure 4.5

Pseudo code of the model evaluation process

```
Input: dataset of all stores, dataset of exogenous variables
Output: error metrics
for each store in all stores:
        Extract and preprocess the store dataset
        Merge the preprocessed store dataset with the exogenous dataset
        for each fold in 4 folds:
                Extract train set and test set from the merged dataset
                Fit the model
                Produce forecasts
                Compute error metrics
```

4.4. Clustering setup

To answer the second research question, we will set up an experiment for comparing the performance of SARIMAX, PROPHET, XBG, and RF models between baseline and different grouping methods.

The baseline is determined by the performance of the models tuned on the whole company's data and then fit-predict for each store. Next, we will aggregate the sales data into groups. The models will then be tuned on the aggregated data, giving us the best model's hyperparameters of each class in the group. The tuned models of each class are used to fit and predict the stores' sales data belonging to that particular class.

Figure 4.6

Pseudo code of the grouping evaluation process

```
Input: dataset of all stores with exogenous variables, variables of groups: store_level
as SL, province as P, cluster as C
Output: Models' MAPE score for each group
for each model in (ARIMA, PROPHET, XBF, RF):
    for each class in (SL, P, C):
        Aggregate Sales data of stores into one time series
        Preprocess the aggregated time series and merge with exog variables
        Model's hyperparameters tuning with grid-search (autoarima for
        ARIMA) and cross-validation
        for each store in class:
            Extract and preprocess the store data
            Fit the model with best hyperparameters on store's data
            Produce forecasts
            Compute error metrics (MAPE)
```

We use two methods to define groups

1. The first method is a conventional one in which we use predefined store categories such as store_level, province. These variables are usually considered demographic data of the stores determined by the company's business strategy.
2. The other one is to use the K-means clustering algorithm with DTW distance to find the clusters of the store's sales time-series

5. Results

The performance evaluation results from section 4.3 are presented in this section. These figures are supposed to serve as the final model assessment. In addition, example forecasts of every model on the last cross-validation set are included. Although these examples are not representative and should not be used for model evaluation, they can illustrate models' predictions and help clarify the differences among error metrics.

Table 5.1

Results metrics

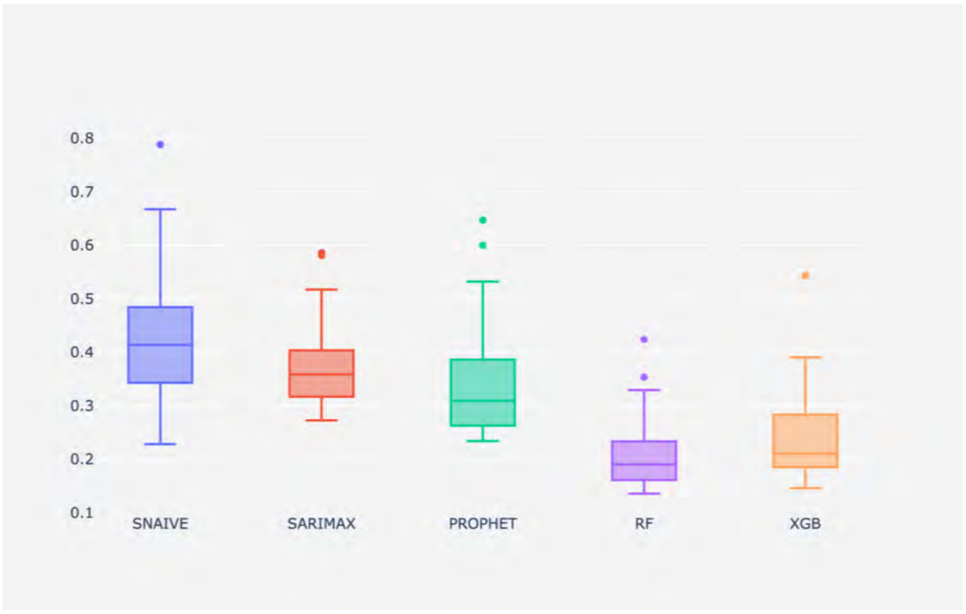
	Model	SNAIVE	ARIMA	PROPHET	RF	XGB
Horizon	Metrics					
7-day	RMSE	11.243	9.322	8.871	**4.500**	5.384
	MAE	8.908	7.471	7.011	**3.642**	4.361
	MAPE	0.419	0.378	0.338	**0.209**	0.243
14-day	RMSE	10.204	9.331	8.708	**4.547**	4.673
	MAE	7.875	7.297	6.642	**3.542**	3.648
	MAPE	0.414	0.416	0.367	**0.223**	0.238
21-day	RMSE	12.529	12.437	11.378	**5.651**	5.746
	MAE	9.524	9.586	8.826	**4.153**	4.228
	MAPE	0.463	0.493	0.445	**0.241**	0.253
28-day	RMSE	11.432	11.067	10.423	**5.321**	7.125
	MAE	8.404	8.253	7.992	**3.830**	5.169
	MAPE	0.456	0.478	0.459	**0.251**	0.308

5.1. Mean absolute percentage error (MAPE)

Figure 5.1 displays the box plots of all five models. It shows that RF archives the lowest MAPE with a median of 0.19. XGB, whose MAPE is 0.21, comes second. RF is also the most stable model with the narrowest interquartile range in terms of variation. SARIMAX and PROPHET's MAPEs are both lower than SNAIVE's. In addition, they fluctuate less, as seen in shorter whiskers.

Figure 5.1

MAPE over 7-days horizon



MAPE over longer forecast horizons can be found in Error metrics box plots over different horizons.

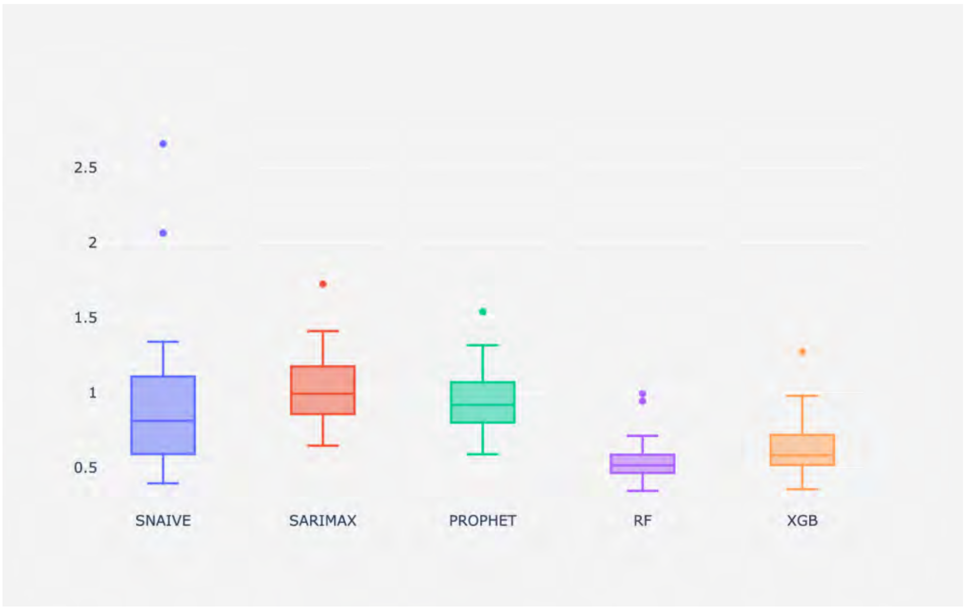
5.2. Mean absolute scaled error (MASE)

The MASE box plots below paint a slightly different picture. While RF and XGB show similar performance as their scores remain the lowest among all applied models (median MASEs are 0.51 and 0.58, respectively), SARIMAX and PROPHET's MASEs are higher than that of SNAIVE. In other words, these two models are less accurate than

SNAIVE when errors are scaled based on the in-sample MAE from the naïve forecast method (Hyndman & Koehler, 2006).

Figure 5.2

MASE over 7-days horizon



One can refer to Error metrics box plots over different horizons for MASEs over other horizons.

5.3. Mean absolute error (MAE), and Root mean square error (RMSE)

Results in terms of MAE and RMSE, plotted below, are consistent with MASE's. Overall, RMSEs are larger in magnitude than MAE across all models. RF's accuracy is the lowest and most stable of all. It is followed by XGB, PROPHET, SNAIVE, and SARIMAX, respectively. These metrics identify more outliers than MAPE and MASE do. However, because these values are on the original data scale, their deviation needs added context for a fair interpretation. For example, in the case of PROPHET, store_328165's MAE and RMSE of 27.24 and 36.93, respectively, are the highest.

However, its MAPE of 0.357 is slightly larger than the mean of 0.338. Thus, we should not treat it as an outlier.

Figure 5.3

MAE over 7-days horizon

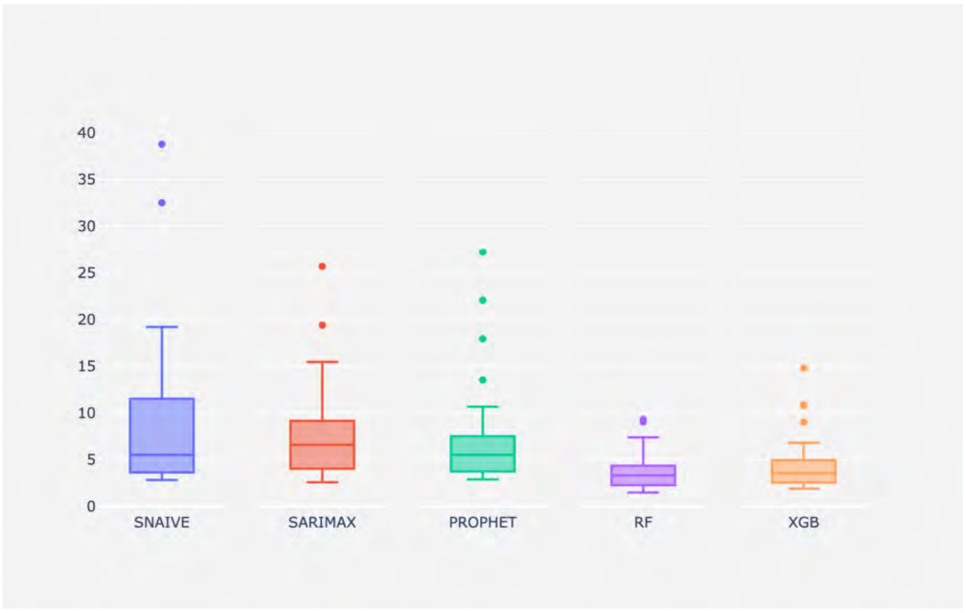
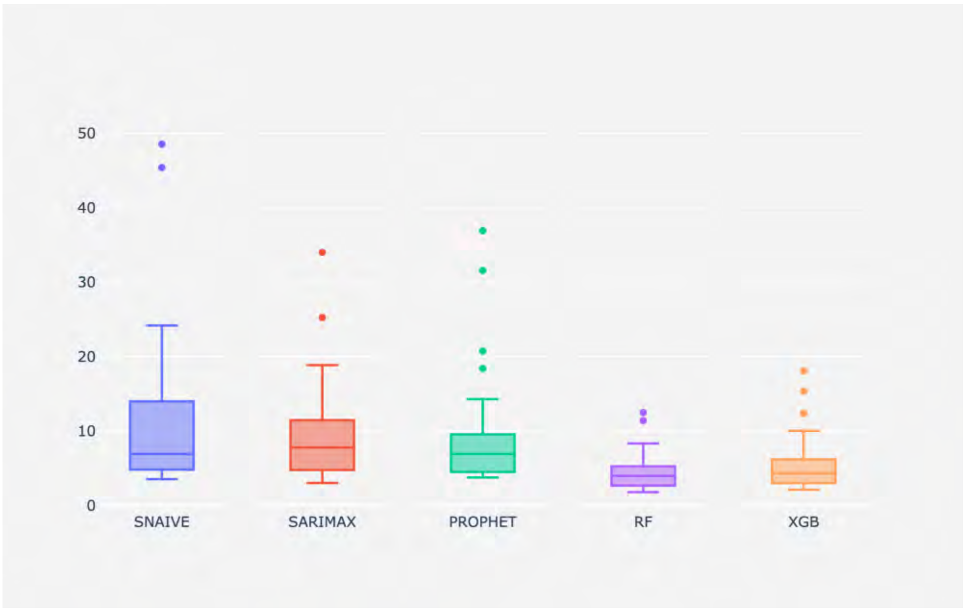


Figure 5.4

RMSE over 7-days horizon



5.4. Forecast results

Figure 5.2 is an example store's sales forecasts made by the models over the horizon of one season (7 days starting from Monday 2021-01-25). Generally, all methods model weekly seasonality in data reasonably well except for SNAIVE, which underestimates the sales increase towards the weekend.

Meanwhile, SARIMAX and PROPHET exhibit both under- and over-estimations. XGB makes high errors on Monday and Tuesday but comes closer to real values over time. The closest assembly to observed data are forecasts made by RF with 2 out of 7 days of nearly exact predictions. **Error! Reference source not found.** visualizes forecasts for the same store over a longer period, 14 days. RF and XGB track the real sales figures much closer than the rest. Their predictions are more accurate on most days. It is also noticeable PROPHET, SARIMAX, and SNAIVE produce more similar forecasts.

Figure 5.2

Sales forecast for store 307222 over 7 days

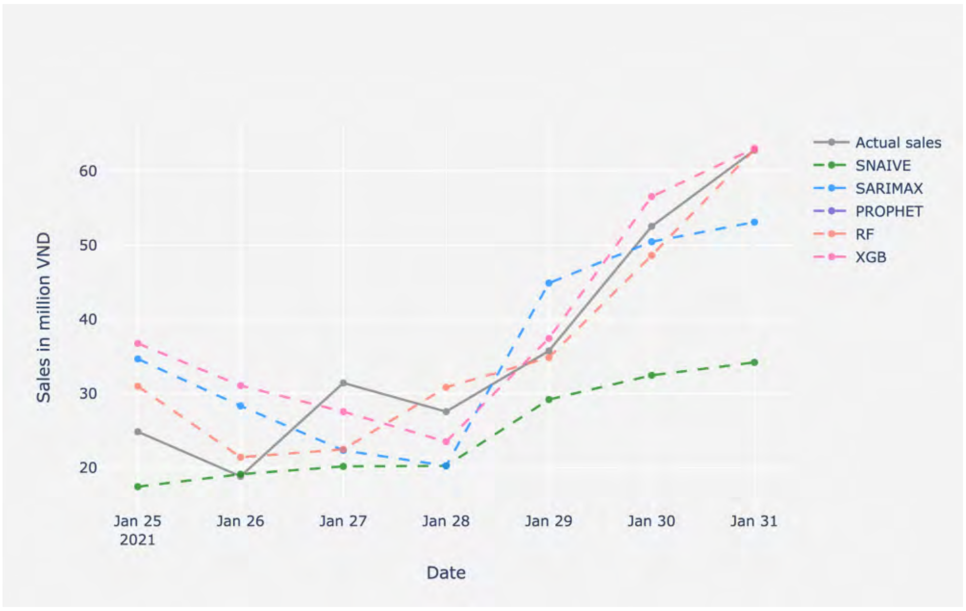
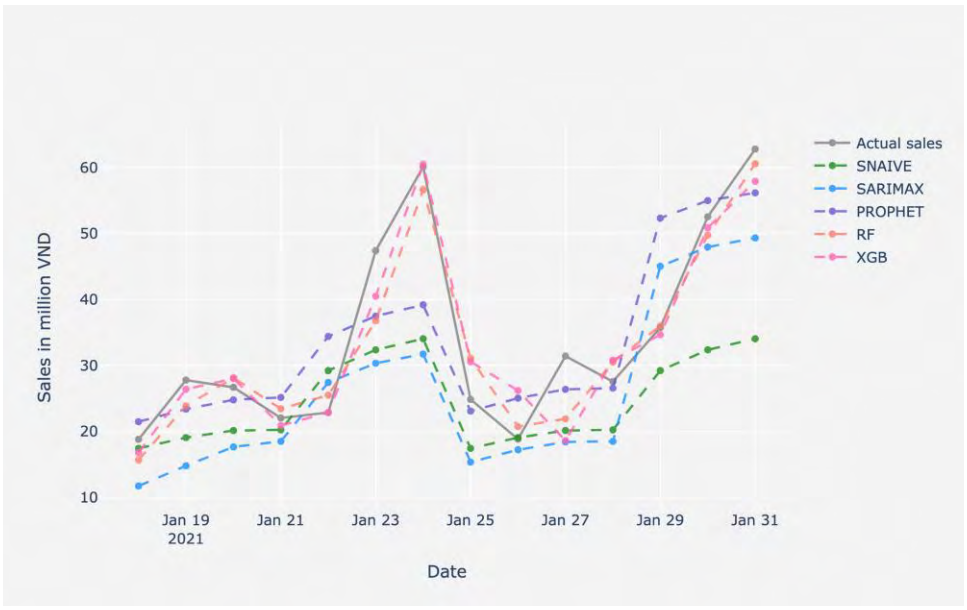


Figure 5.3

Sales forecast for store 307222 over 14 days



5.5. Store clustering

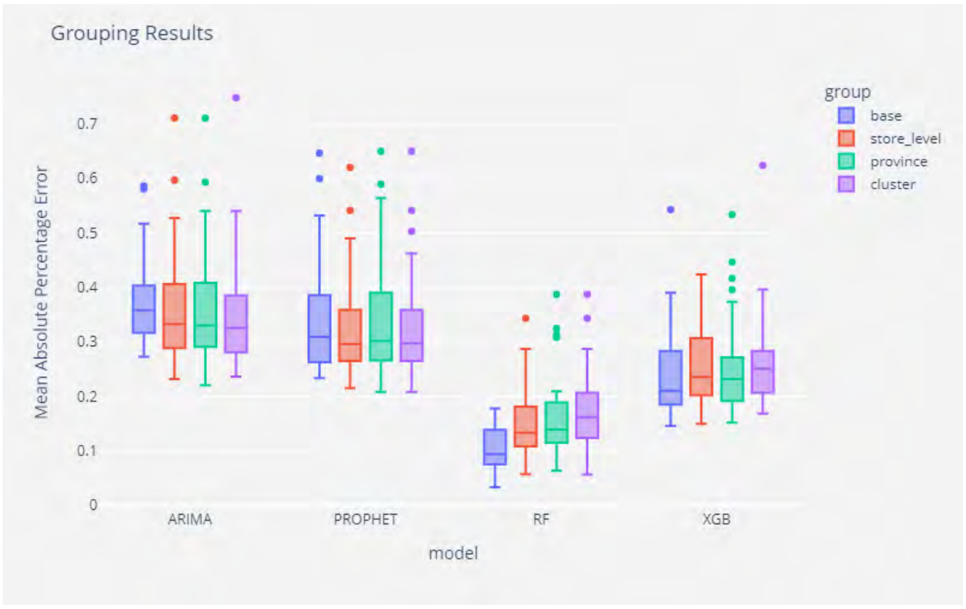
Table 5.2

Grouping results - Mean MAPE

Group \ Model	ARIMA	PROPHET	RF	XGB
Base	0.377	0.338	**0.104**	**0.243**
Store Level	0.360	**0.325**	0.150	0.251
Province	0.358	0.341	0.159	0.252
Clustering	**0.353**	0.328	0.173	0.264

Figure 5.4

MAPE of clustering over different models



The MAPE scores of the grouping technique show slightly better results in ARIMA and PROPHET models. However, in RF models, the baseline tuned on the whole company sales for the RF model performed significantly better than the grouping ones. Thus, to have a better score and compute efficiency, we believe staying with the baseline method is better when deploying the model into real-world scenarios.

6. Discussion

This section aims at discussing the results of the forecasting models derived above. Many interesting insights have been found. Through the evaluation and comparison of different techniques, the authors identify the best method for Couple TX. Further, we try to answer our research questions. Then, certain limitations of the study will be presented and discussed, along with some potential ideas for future research.

6.1. Research questions answer

Overall, machine learning techniques yield better results for forecasting retail store sales than statistical methods. To be specific, Random Forests consistently outperforms all others, then XGB. Regarding different performance evaluation metrics, the ranking for the other three is slightly divergent. In MAPE measurement, Facebook Prophet has the lowest, then SARIMA, and SNAIVE as last. However, in contrast, the order has changed to SNAIVE, Facebook Prophet, then SARIMA in terms of the error measures MASE, MAE, and RMSE. It is a common situation of model ranking for different evaluation metrics, which also happens to Makridakis & Hibon (2000), and Aanes & Gullien (2018).

The forecast accuracy fluctuates and is less precise in longer forecast results. Details for each method's errors in horizon 14-day, 21-day, and 28-day are visualized in Appendix A. In general, the ranking of five methods in each error metric can be considered consistent. However, forecasting on a longer horizon consumes more time and is less incorrect, which is against the purpose of finding a fast and intuitive model for the fashion industry.

Random Forests is the most well-performing model despite the horizon or performance evaluation metric.

On the other hand, the hierarchical forecasts have resulted in contrasting results among three models, ARIMA, Prophet, RF, and XGB. There is a clear improvement in accuracy in ARIMA and Prophet methods when clustering the sales data. Even though the performance between segmenting by store level, location, or KNN is merely the same, they indicate promising future research and prove the impact of store elements on sales performance.

Nevertheless, Random Forests and XGB somewhat show the opposite. The baseline performs much better than any others do. Although, in the worst scenario, it is still surprising that the clustering does not outperform benchmark results. Still, KNN yields the lowest result in the Random Forests and XGB model.

6.2. Strengths, Limitations, and Future Research

Our thesis has some significant outcomes. It is remarkable to introduce a company with no experience with proper forecast techniques, ranging from basic to advance. The forecast has been made not only on a particular horizon but expanded to the longer term. Moreover, we also consider clustering to improve model accuracy and solve the problem of multi-stores in a retail chain. Many evaluation metrics have given a general yet subjective perspective to choosing the best forecasting model.

Forecasting is a difficult task, and we are just beginners in the field. Thus, it is inevitable for us to avoid limitations. First, only one data preprocessing procedure was considered, indicated by preliminary analysis, instead of experimenting with different techniques. For example, Box-Cox transformations (Box & Cox, 1964), min-max scaling, and standard scaling (Löning et al., 2019; Löning et al., 2022) are some popular techniques to help improve model performance. Second, we acknowledge our lack of domain expertise in Vietnam's fashion retail market. Channeling in the domain knowledge of someone with a deep understanding of the ebbs and flows in the fashion industry will contribute to a better interpretation of variations in the data. Third, it is likely that more and better data leads to better performance. In particular, detailed information about future promotion programs can be indicative and used as an additional exogenous variable. Finally, we did not manage to explore deeply the reasons why the models differ, especially when clustering.

On the other hand, this thesis upholds many opportunities for future research. Several additional preprocess procedures could be considered. For example, practitioners should integrate more data from multiple sources to have the overall picture rather than a flat file of sales transactions. Moreover, it would be interesting to engage more closely with a domain expert, like a common practice as always. From that, they could give more opinions when choosing different clustering techniques in the future. Predicting the sales when promotions are not run on the whole chain but in some particular stores is also a good topic for retail business. Other than using only store characteristics as variables, more factors that affect sales performance can be experimented with, such as weather, store manager, store service, or customer characteristics related. Especially in the Internet of Things era, it would be a vast topic to compare online and physical store sales forecasts. Later studies could try other model types like neural networks and hybrid models.

7. Conclusion

Textile industry is one of the most significant contributors to Vietnam's economy. However, the local market is not attractive to customers. It is because of the lack of proper control over the demand amount. Having a quick and easy sales forecast is one of the solutions to this problem. Thus, we set out a suitable forecasting model for every store of a chain retailer. Here we use data from a Vietnamese fashion brand, Couple TX.

Even though using quantitative prediction is still a new practice to Couple TX in particular and any company in developing countries in general, we aimed to formulate a thorough procedure, from data exploration, then model tuning, to model evaluation. We chose Seasonal naïve and ARIMA models as simple benchmark methods and advanced techniques from machine learning like Facebook Prophet, Random Forests, and XGB. The horizon forecast was conducted variously, ranging from 7-day to 28-day. Moreover, in this study, we examine some store characteristics to see whether they help improve store performance. To evaluate those models, we calculate metrics, namely MAPE, MASE, RMSE, and MAE.

Overall, we find that machine-learning methods yield better results. Random Forests is the most accurate one. Depending on the length of the horizon and which performance metric is chosen, the ranking of other methods might differ. In general, XGB comes second, then Facebook Prophet, ARIMA, and seasonal naïve, as expected. The thesis has seen a slight reduction in the error of models when clustering sales data. However, it happens only on ARIMA, and Facebook Prophet, while Random Forests and XGB work best with whole data.

Due to time constraints and data confidentiality, we could not investigate some limitations further. However, it also gives opportunities and potential for future research. We recommend including expert opinions, trying neural networks, and other clustering methods.

References

Aamer, A., Eka Yani, L. P., & Alan Priyatna, I. M. (2020). Data Analytics in the Supply Chain Management: Review of Machine Learning Applications in Demand Forecasting. Operations and Supply Chain Management: An International Journal, 14(1), 1–13. https://doi.org/10.31387/oscm0440281

Aanes, B., & Gullien, M. (2018). Forecasting Norwegian inflation with deep neural networks: the application and comparison of different feedforward architectures [MSc. Thesis]. In openaccess.nhh.no.

Afzal, A., Aabid, A., Khan, A., Afghan Khan, S., Rajak, U., Nath Verma, T., & Kumar, R. (2020). Response surface analysis, clustering, and random forest regression of pressure in suddenly expanded high-speed aerodynamic flows. Aerospace Science and Technology, 107, 106318. https://doi.org/10.1016/j.ast.2020.106318

Afzal, A., Aabid, A., Khan, A., Afghan Khan, S., Rajak, U., Nath Verma, T., & Kumar, R. (2020). Response surface analysis, clustering, and random forest regression of pressure in suddenly expanded high-speed aerodynamic flows. Aerospace Science and Technology, 107, 106318. https://doi.org/10.1016/j.ast.2020.106318

Aghabozorgi, S., Seyed Shirkhorshidi, A., & Ying Wah, T. (2015). Time-series clustering – A decade review. Information Systems, 53, 16–38. https://doi.org/10.1016/j.is.2015.04.007

Ahmed, N. K., Atiya, A. F., Gayar, N. E., & El-Shishiny, H. (2010). An Empirical Comparison of Machine Learning Models for Time Series Forecasting. Econometric Reviews, 29(5–6), 594–621. https://doi.org/10.1080/07474938.2010.481556

Aksoy, A., Öztürk, N., & Sucky, E. (2012). Demand forecasting for apparel manufacturers by using neurofuzzy techniques. Journal of Modelling in Management, 9(1), 18–35. https://doi.org/10.1108/JM21020110045

Alfaro, L. (2019). Automated Time Series Demand Forecast for Luxury Fashion Online Retail Company Internship report presented as partial requirement for obtaining the Master’s degree in Advanced Analytics [MSc Thesis].

Bhattacharyya, S. C., & Timilsina, G. R. (2010). Modelling energy demand of developing countries: Are the specific features adequately captured? Energy Policy, 38(4), 1979–1990. https://doi.org/10.1016/j.enpol.2009.11.079

Box, G. E. P., & Cox, D. R. (1964). An Analysis of Transformations. Journal of the Royal Statistical Society: Series B (Methodological), 26(2), 211–243. https://doi.org/10.1111/j.2517-6161.1964.tb00553.x

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time Series Analysis: Forecasting and Control, 5th Edition | Wiley. In Wiley.com (5th ed.). New York: Wiley.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and Regression Trees (Wadsworth Statistics/Probability). In undefined. Chapman and Hall.

Brockwell, P. J., & Davis, R. A. (2016). Introduction to time series and forecasting. Springer.

Brownlee, J. (2017). Introduction to Time Series Forecasting With Python: How to Prepare Data and Develop Models to Predict the Future. Machine Learning Mastery.

Brownlee, J. (2018). Deep Learning for Time Series Forecasting: Predict the Future with MLPs, CNNs and LSTMs in Python. Machine Learning Mastery.

Chang, W., Cheng, J., Allaire, J. J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., Borges, B., RStudio, library), jQuery F. (jQuery library and jQuery U., inst/www/shared/jquery-AUTHORS.txt), jQuery contributors (jQuery library; authors listed in, inst/www/shared/jqueryui/AUTHORS.txt), jQuery U. contributors (jQuery U. library; authors listed in, library), M. O. (Bootstrap, library), J. T. (Bootstrap, library), B. contributors (Bootstrap, Twitter, library), I. (Bootstrap, … R), R. C.

T. (tar implementation from. (2015). shiny: Web Application Framework for R (1.7.1) [Computer software]. https://CRAN.R-project.org/package=shiny

Chen, I-Fei., & Lu, C.-J. (2021). Demand Forecasting for Multichannel Fashion Retailers by Integrating Clustering and Machine Learning Algorithms. Processes, 9(9), 1578. https://doi.org/10.3390/pr9091578

Chen, T. (2014). Introduction to Boosted Trees. https://web.njit.edu/~usman/courses/cs675_spring20/BoostedTree.pdf

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794. https://doi.org/10.1145/2939672.2939785

Chopra, S., & Meindl, P. (2016). Supply Chain Management: Strategy, Planning, and Operation. In Northwestern Scholars (6th ed., p. 528). Pearson Education. https://www.scholars.northwestern.edu/en/publications/supply-chain-management-strategy-planning-and-operation-2

Chu, S., Keogh, E., Hart, D., & Pazzani, M. (2002). Iterative deepening dynamic time warping for time series. 195–212. https://doi.org/10.1137/1.9781611972726.12

coupletx.com. (2019). COUPLE TX. https://coupletx.com/pages/danh-sach-cua-hang?province=all

Cochran, D., & Orcutt, G. H. (1949). Applications of least squares regression to relationships containing autocorrelated errors. Journal of the American Statistical Association, 44, 32–61.

De Gooijer, J. G., & Hyndman, R. J. (2006). 25 years of time series forecasting. International Journal of Forecasting, 22(3), 443–473. https://doi.org/10.1016/j.ijforecast.2006.01.001

Don-Alvin Adegeest. (2021, August 2). Global trade: Vietnam overtakes Bangladesh in apparel exports. FashionUnited; FashionUnited. https://fashionunited.com/news/business/global-trade-vietnam-overtakes-bangladesh-in-apparel-exports/2021080241334

Dudek, G. (2015). Short-Term Load Forecasting Using Random Forests. In D. Filev, J. Jabłkowski, J. Kacprzyk, M. Krawczak, I. Popchev, L. Rutkowski, V. Sgurev, E. Sotirova, P. Szynkarczyk, & S. Zadrozny (Eds.), Intelligent Systems'2014 (pp. 821–828). Springer International Publishing. https://doi.org/10.1007/978-3-319-11310-4_71

Frank, C., Garg, A., Sztandera, L., & Raheja, A. (2003). Forecasting women's apparel sales using mathematical modeling. International Journal of Clothing Science and Technology, 15(2), 107–125. https://doi.org/10.1108/09556220310470097

Gauri, D. K., Ratchford, B., Pancras, J., & Talukdar, D. (2017). An Empirical Analysis of the Impact of Promotional Discounts on Store Performance. Journal of Retailing, 93(3), 283–303. https://doi.org/10.1016/j.jretai.2017.06.001

Gaur, S. (2020). Global forecasting of Covid-19 using ARIMA based FB-Prophet. International Journal of Engineering Applied Sciences and Technology, 5(2), 463–467.

Géron, A. (2019). Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems (Second edition). O'Reilly Media, Inc.

Ghosh, A. (1984). Parameter nonstationarity in retail choice models. Journal of Business Research, 12(4), 425–436. https://doi.org/10.1016/0148-2963(84)90023-7

Gonzalo, A., Holger Harreis, Carlos Sanchez Altable, & Cyrielle Villepelet. (2020, May 6). Fashion's digital transformation: Now or never. McKinsey & Company; McKinsey & Company. https://www.mckinsey.com/industries/retail/our-insights/fashions-digital-transformation-now-or-never

GÜVEN, İ., UYGUN, Ö., & ŞİMŞİR, F. (2021). Machine Learning Algorithms with Intermittent Demand Forecasting: An Application in Retail Apparel with Plenty of Predictors. TEKSTİL ve KONFEKSİYON, 31(2). https://doi.org/10.32710/tekstilvekonfeksiyon.809867

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. Nature, 585(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning, second edition : data mining, inference, and prediction. Springer.

Hayes, A. (2021, April 29). How Fast Fashion Works. Investopedia. https://www.investopedia.com/terms/f/fast-fashion.asp

Hyndman, R. J., & Athanasopoulos, G. (2021). Forecasting: principles and practice.

Hyndman, R. J., & Khandakar, Y. (2008). Automatic Time Series Forecasting: The forecast Package for R. Journal of Statistical Software, 27(3). https://doi.org/10.18637/jss.v027.i03

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. International Journal of Forecasting, 22(4), 679–688. https://doi.org/10.1016/j.ijforecast.2006.03.001

Kimball, R., & Ross, M. (2013). The data warehouse toolkit : the definitive guide to dimensional modeling. John Wiley & Sons, Inc.

Lai, J.-P., Chang, Y.-M., Chen, C.-H., & Pai, P.-F. (2020). A Survey of Machine Learning Models in Renewable Energy Predictions. Applied Sciences, 10(17), 5975. https://doi.org/10.3390/app10175975

Liu, N., Ren, S., Choi, T.-M., Hui, C.-L., & Ng, S.-F. (2013). Sales Forecasting for Fashion Retailing Service Industry: A Review. Mathematical Problems in Engineering, 2013, 1–9. https://doi.org/10.1155/2013/738675

Local fashion brands struggle to compete in domestic market. (2021). Vietnamnews.vn. https://vietnamnews.vn/economy/950800/local-fashion-brands-struggle-to-compete-in-domestic-market.html

Löning, M., Bagnall, A., Ganesh, S., & Kazakov, V. (2019). sktime: A Unified Interface for Machine Learning with Time Series. 10.

Löning, M., Bagnall, T., Király, F., Middlehurst, M., Ganesh, S., Oastler, G., & Lines, J. (2022). alan-turing-institute/sktime: v0.11.2 hotfix. Zenodo. https://doi.org/10.5281/zenodo.6450102

Lusch, R. F., Serpkenci, R. R., & Orvis, B. T. (2015). Determinants of Retail Store Performance: A Partial Examination of Selected Elements of Retailer Conduct. Proceedings of the 1995 World Marketing Congress, 495–504. https://doi.org/10.1007/978-3-319-17311-5_69

Makridakis, S., & Hibon, M. (2000). The M3-Competition: results, conclusions and implications. International Journal of Forecasting, 16(4), 451–476. https://doi.org/10.1016/s0169-2070(00)00057-1

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). Statistical and Machine Learning forecasting methods: Concerns and ways forward. PLOS ONE, 13(3), e0194889. https://doi.org/10.1371/journal.pone.0194889

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2020). The M4 Competition: 100,000 time series and 61 forecasting methods. International Journal of Forecasting, 36(1), 54–74. https://doi.org/10.1016/j.ijforecast.2019.04.014

Morde, V. (2019, April 8). XGBoost Algorithm: Long May She Reign! Medium; Towards Data Science. https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

Naim, I., & Mahara, T. (2018). Comparative Analysis of Univariate Forecasting Techniques for Industrial Natural Gas Consumption. International Journal of Image, Graphics and Signal Processing, 10(5), 33–44. https://doi.org/10.5815/ijigsp.2018.05.04

Nau, R. (2020). Statistical forecasting: notes on regression and time series analysis. https://people.duke.edu/~rnau/411home.htm

Parmezan, A. R. S., Souza, V. M. A., & Batista, G. E. A. P. A. (2019). Evaluation of statistical and machine learning models for time series prediction: Identifying the state-of-the-art and the best conditions for the use of each model. Information Sciences, 484, 302–337. https://doi.org/10.1016/j.ins.2019.01.076

Norrman, A., & Naslund, D. (2019). Supply Chain Incentive Alignment: The Gap between Perceived Importance and Actual Practice. Operations and Supply Chain Management: An International Journal, 12(3), 129–142. https://doi.org/10.31387/oscm0380237

Papacharalampous, G., Tyralis, H., & Koutsoyiannis, D. (2018). Univariate Time Series Forecasting of Temperature and Precipitation with a Focus on Machine Learning Algorithms: a Multiple-Case Study from Greece. Water Resources Management, 32(15), 5207–5239. https://doi.org/10.1007/s11269-018-2155-6

Pavlyshenko, B. (2019). Machine-Learning Models for Sales Time Series Forecasting. Data, 4(1), 15. https://doi.org/10.3390/data4010015

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12(85), 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Petitjean, F., Ketterlin, A., & Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. Pattern Recognition, 44(3), 678–693. https://doi.org/10.1016/j.patcog.2010.09.013

Pham, L. (2019, November 13). The most popular Vietnamese local brands | - PHILIAGROUP Official Website. Philiagroup.com. https://www.philiagroup.com/the-most-popular-vietnamese-local-brands/

Plotly Technologies Inc. (2015). Collaborative data science. https://plot.ly

Rai, P., & Singh, S. (2010). A Survey of Clustering Techniques. International Journal of Computer Applications, 7(12), 1–5.

Reback, J., Jbrockmendel, McKinney, W., Van Den Bossche, J., Augspurger, T., Roeschke, M., Hawkins, S., Cloud, P., Gfyoung, Sinhrks, Hoefler, P., Klein, A., Terji Petersen, Tratner, J., She, C., Ayd, W., Naveh, S., JHM Darbyshire, Garcia, M., … Battiston, P. (2022). pandas-dev/pandas: Pandas 1.4.2 (v1.4.2) [Computer software]. Zenodo. https://doi.org/10.5281/ZENODO.3509134

Ren, S., Chan, H., & Siqin, T. (2020). Demand forecasting in retail operations for fashionable products: methods, practices, and real case study. Annals of Operations Research, 291(1), 761–777. https://doi.org/10.1007/s10479019031488

Ren, S., Chan, H.-L., & Ram, P. (2016). A Comparative Study on Fashion Demand Forecasting Models with Multiple Sources of Uncertainty. Annals of Operations Research, 257(1-2), 335–355. https://doi.org/10.1007/s10479-016-2204-6

Reppucci, F. (2020). A Clustering-Base Approach For City Electricity Demand Forecasting [Master of Science in Computer Science and Engineering].

Rostami-Tabar, B. (2021). Business Forecasting in Developing Countries. Business Forecasting: The Emerging Role of Artificial Intelligence and Machine Learning, 382.

Seabold, S., & Josef, P. (2010). statsmodels: Econometric and statistical modeling with python. 9th Python in Science Conference

Shih, H., & Rajendran, S. (2019). Comparison of Time Series Methods and Machine Learning Algorithms for Forecasting Taiwan Blood Services Foundation's Blood Supply. Journal of Healthcare Engineering, 2019, e6123745. https://doi.org/10.1155/2019/6123745

Singh, P., Gupta, Y., Jha, N., & Rajan, A. (2019). Fashion Retail: Forecasting Demand for New Items. https://arxiv.org/pdf/1907.01960.pdf

Smith, T. G., & others. (2017). pmdarima: ARIMA estimators for Python. http://alkaline-ml.com/pmdarima/

Song, Q. (2015). Lessons Learned and Challenges Encountered in Retail Sales Forecast. Industrial Engineering and Management Systems, 14(2), 196–209. https://doi.org/10.7232/iems.2015.14.2.196

Spiliotis, E., Makridakis, S., Semenoglou, A.-A., & Assimakopoulos, V. (2020). Comparison of statistical and machine learning methods for daily SKU demand forecasting. Operational Research. https://doi.org/10.1007/s12351-020-00605-2

Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., & Kolar, K. (2020). A Machine Learning Toolkit for Time Series Data. Journal of Machine Learning Research, 21, 1–6.

Taylor, S. J., & Letham, B. (2017). Forecasting at scale (e3190v2). PeerJ Preprints. https://doi.org/10.7287/peerj.preprints.3190v2

Thomassey, S., & Fiordaliso, A. (2006). A hybrid sales forecasting system based on clustering and decision trees. Decision Support Systems, 42(1), 408–421. https://doi.org/10.1016/j.dss.2005.01.008

Van Belle, J., Guns, T., & Verbeke, W. (2021). Using shared sell-through data to forecast wholesaler demand in multi-echelon supply chains. European Journal of Operational Research, 288(2), 466–479. https://doi.org/10.1016/j.ejor.2020.05.059

van Steenbergen, R. M., & Mes, M. R. K. (2020). Forecasting demand profiles of new products. Decision Support Systems, 139, [113401]. https://doi.org/10.1016/j.dss.2020.113401

Vietnam Briefing, & Nguyen, T. (2020, August 7). Seizing Investment Opportunities in Vietnam’s Garment and Textile Sector. Vietnam Briefing News. https://www.vietnam-briefing.com/news/seizing-investment-opportunities-vietnams-textile-garment-industry.html/

Vietnam Investment Review. (2018, September 21). Fast fashion set to transform market. Vietnam Investment Review - VIR; https://vir.com.vn/. https://vir.com.vn/fast-fashion-set-to-transform-market-62517.html

Wang, Q., Li, S., & Li, R. (2018). Forecasting energy demand in China and India: Using single-linear, hybrid-linear, and non-linear time series forecast techniques. Energy, 161, 821–831. https://doi.org/10.1016/j.energy.2018.07.168

Wang, Z., & Liu, X. (2021). Statistical Analysis and Big Data Based Intelligent Fashion Prediction Model. IEEE, 878–882. https://doi.org/10.1109/TOCS53301.2021.9688919

Zhang, X., Zhang, T., Young, A. A., & Li, X. (2014). Applications and Comparisons of Four Time Series Models in Epidemiological Surveillance Data. PLoS ONE, 9(2), e88075. https://doi.org/10.1371/journal.pone.0088075

Žunić, E., Korjenić, K., Hodžić, K., & Đonko, D. (2020). Application of Facebook's Prophet Algorithm for Successful Sales Forecasting Based on Real-world Data. International Journal of Computer Science and Information Technology, 12(2), 23–36. https://doi.org/10.5121/ijcsit.2020.12203

Appendix A - Error metrics box plots over different horizons

Figure A.1

MAPE over 14-days horizon

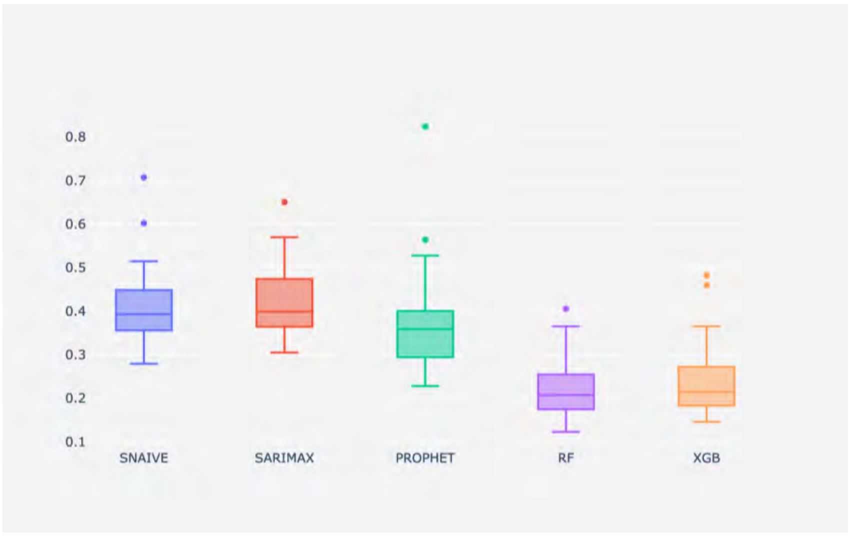


Figure A.2

MAPE over 21-days horizon

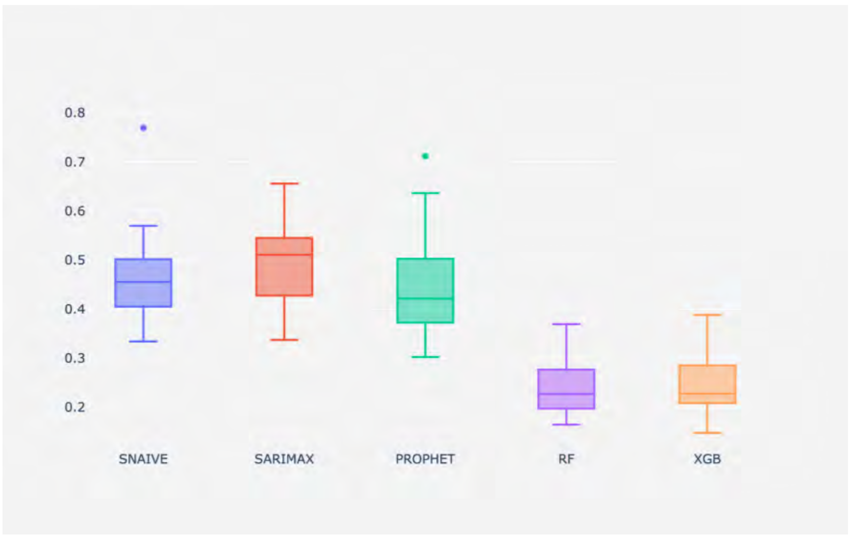


Figure A.3

MAPE over 28-days horizon

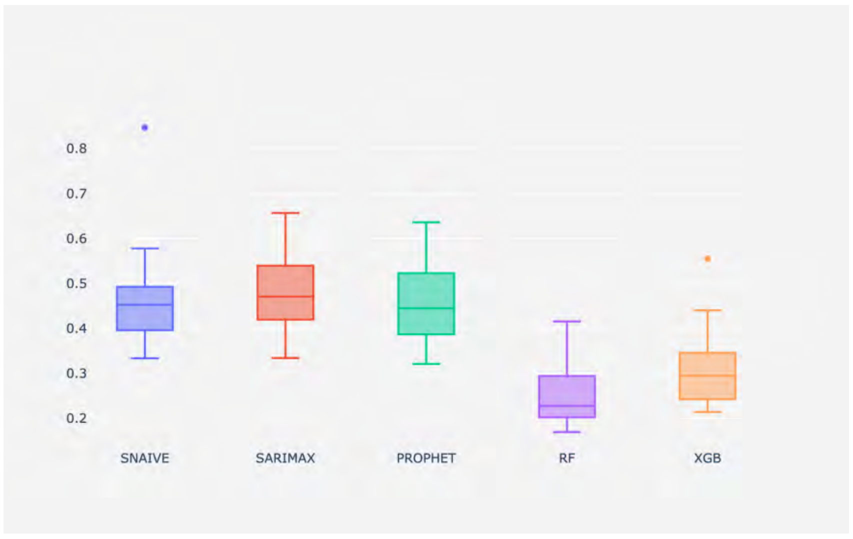


Figure A.4

MASE over 14-days horizon

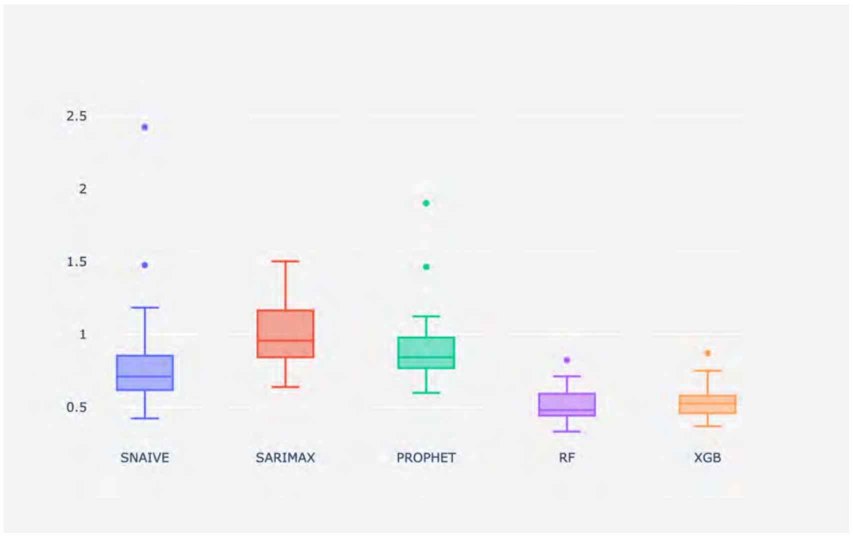


Figure A.5

MASE over 21-days horizon

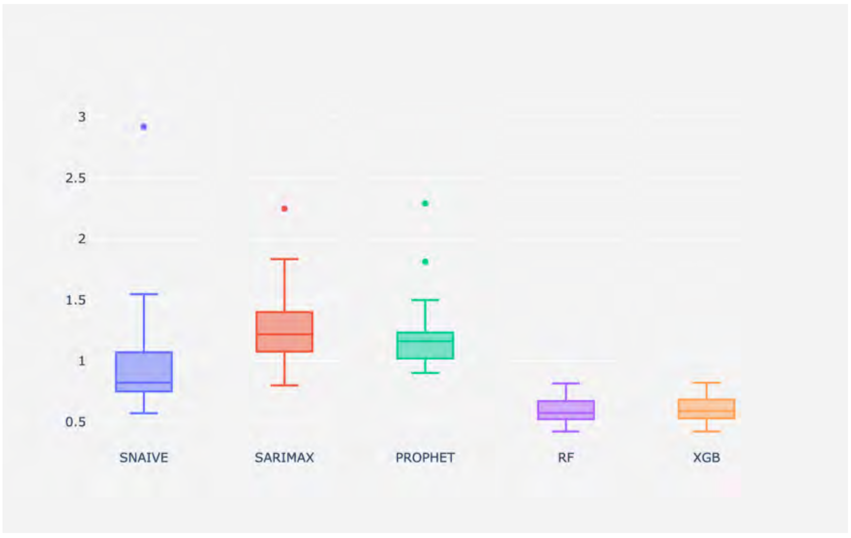


Figure A.6

MASE over 28-days horizon

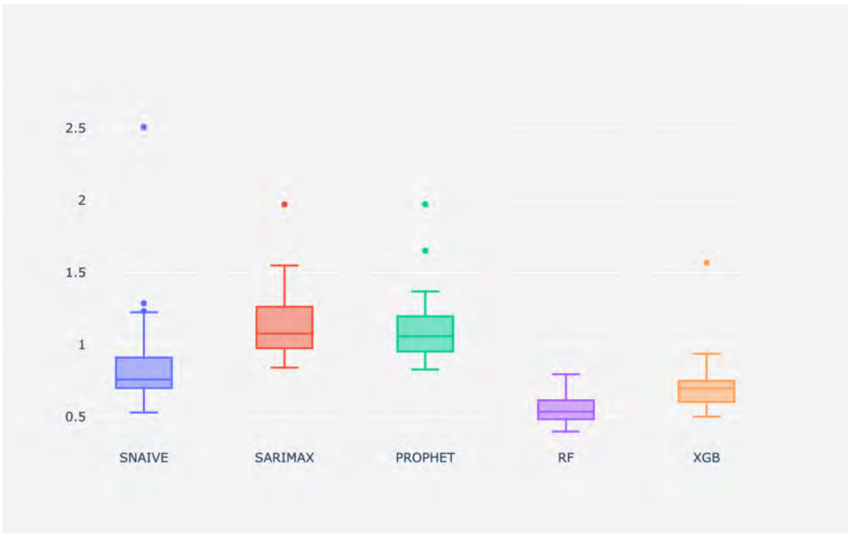


Figure A.7

MAE over 14-days horizon

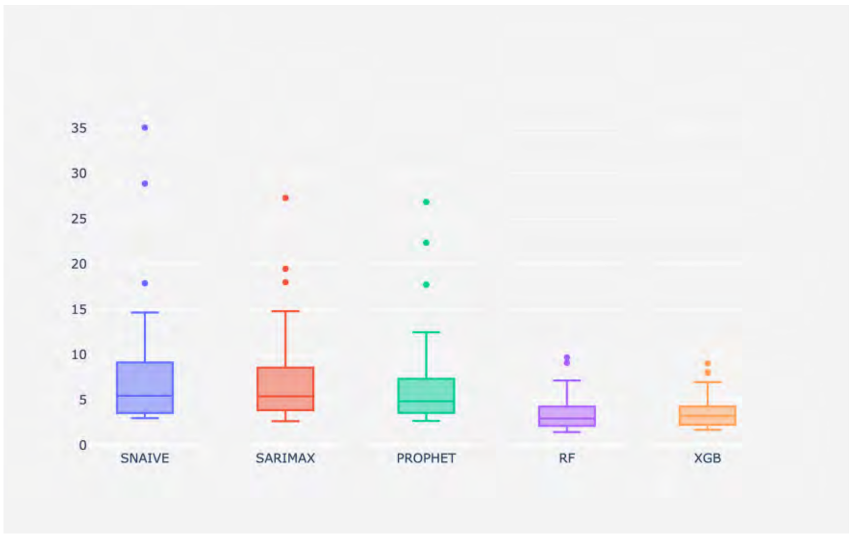


Figure A.8

MAE over 21-days horizon

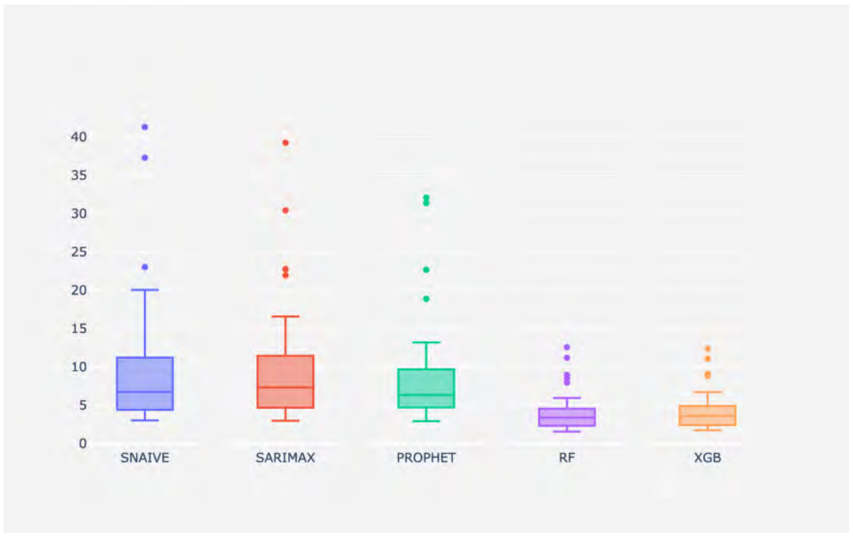


Figure A.9

MAE over 28-days horizon

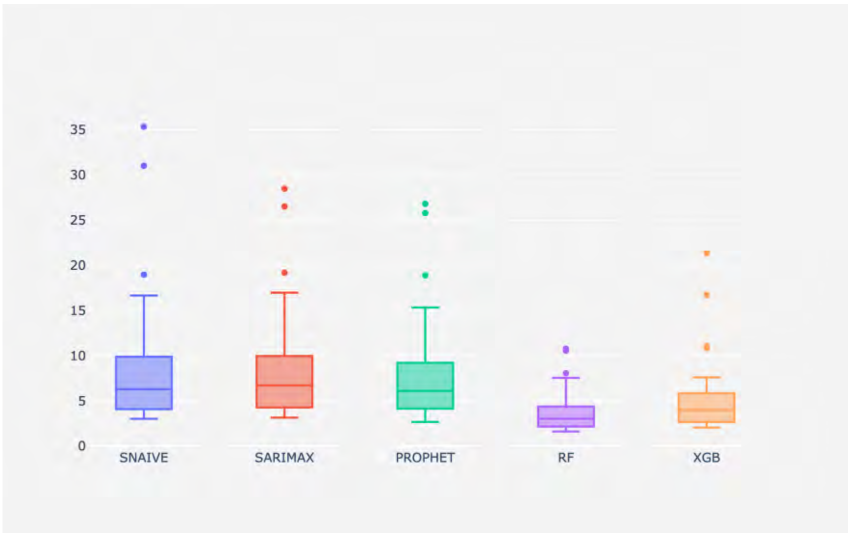


Figure A.10

RMSE over 14-days horizon

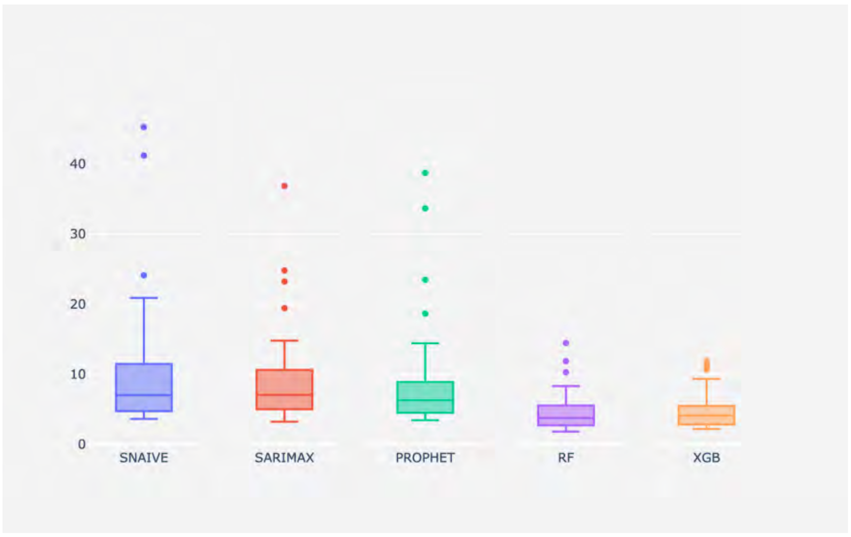


Figure A.11

RMSE over 21-days horizon

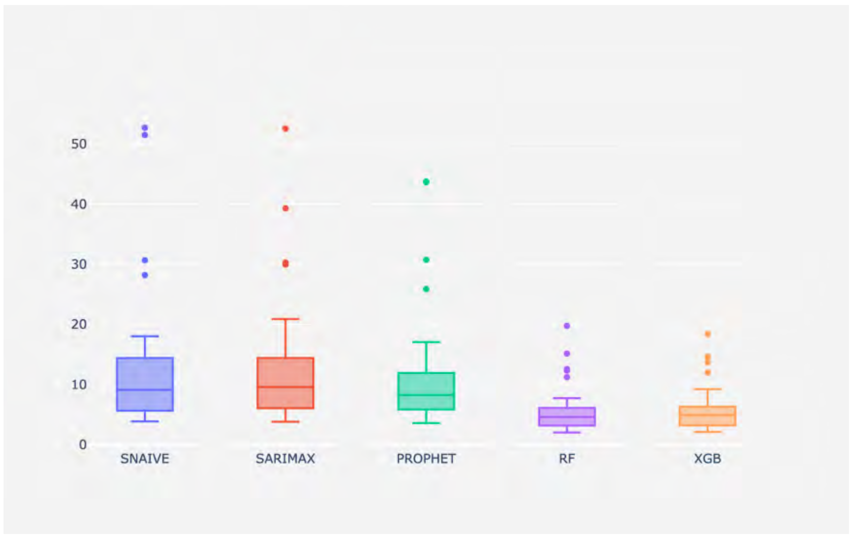
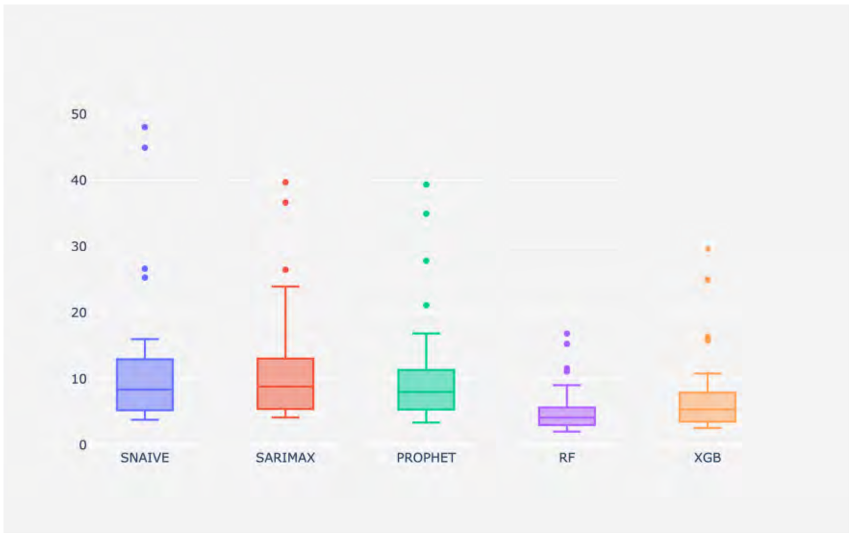


Figure A.12

RMSE over 28-days horizon



Appendix B - Forecast plots over different horizons

Figure B.1

Sales forecast for store 307222 over 21 days

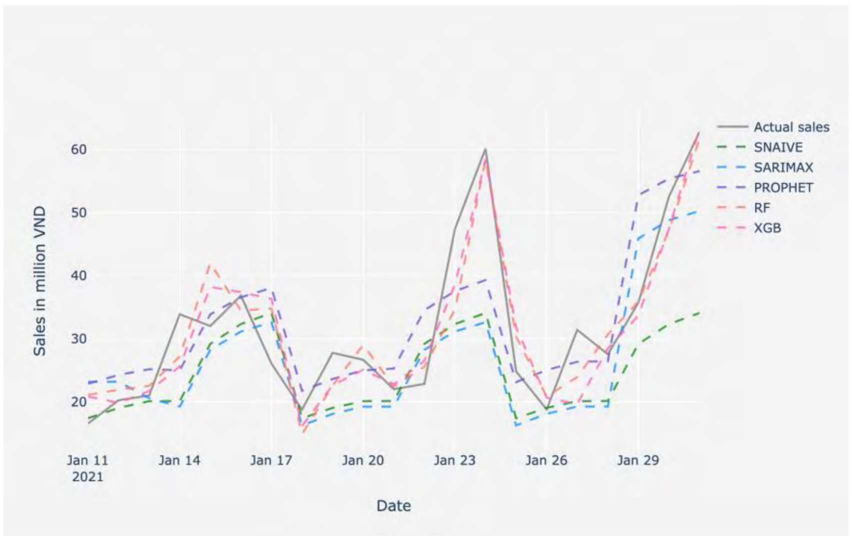
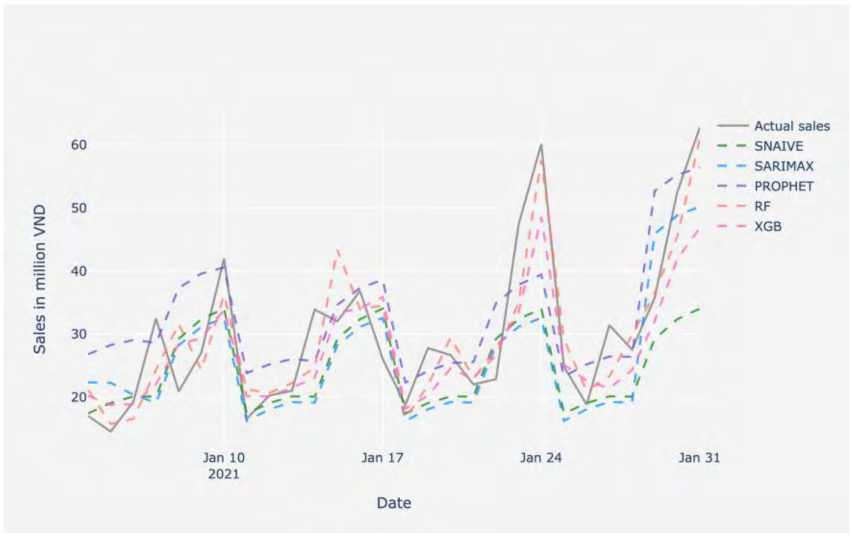


Figure B.2

Sales forecast for store 307222 over 28 days


Actual sales
SNAIVE
SARIMAX
PROPHET
RF
XGB
Sales in million VND
Jan 10
Jan 17
Jan 24
Jan 31
Date