MULTIOBJECTIVE TREES FOR FORECASTING

by

İrem Arıca

B.S., Management Engineering, Istanbul Technical University, 2019

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2023

ACKNOWLEDGEMENTS

Firstly, I am sincerely grateful to my advisor, Assoc. Prof. Mustafa Gökçe Baydoğan, for his invaluable guidance, support, and mentorship throughout this journey. His expertise, patience, and continuous encouragement have played a crucial role in shaping the direction and outcome of my research.

I would also like to extend my gratitude to my supervisor, Prof. Mustafa Necati Aras, for his valuable insights and support.

Additionally, I would like to express my appreciation to Assoc. Prof. Gönenç Yücel and Assist. Prof. Mert Edalı for their participation as members of my thesis committee and for their insightful feedback and suggestions.

I am deeply thankful to my mother, Sabiha Arıca, for her unwavering love and support. Her encouragement and belief in me have been my constant motivation. Furthermore, I am grateful to Özgün Mert Yüksek for his support, understanding, and patience throughout this journey.

I would like to dedicate this thesis to the memories of my father and my uncle, whose wisdom and guidance have profoundly influenced my life and academic pursuits.

Finally, I would like to acknowledge and express my appreciation to all my family members and friends who have supported and encouraged me throughout this journey.

ABSTRACT

MULTIOBJECTIVE TREES FOR FORECASTING

Time series forecasting is a significant task, and it can be used in various fields such as education, finance, medicine, and manufacturing. The wide use of temporal data has resulted in many researches and studies in the field of data mining. Besides time series analysis models, tree-based methods are commonly used for time series forecasting. Tree-based models provide accurate results in complex datasets, and they can explain nonlinear relationships. However, time series data is obtained by measuring a variable at certain time intervals and often contains predictable fluctuations. In the use of tree-based modeling, the temporal relationship and seasonality in time series data should be included in the modeling process. Regression trees consider all data points independently of each other. Thus, preprocessing or postprocessing methods are needed for their use in time series forecasting. However, temporal and seasonality effects can be incorporated into learning process of trees. In this thesis, Multiobjective Trees for Forecasting (MTF) Model is proposed, which is a tree-based ensemble approach that makes changes in the learning phase of decision trees. In addition to objective of decision trees, two new considerations are added to be taken into account during partition. These are temporal closeness in nodes and the seasonal distance between observations. A single decision metric is created by combining these three objectives in different weights, and splitting is done according to this decision metric. In the experiments, five different synthetic datasets and three time series datasets are used. Adding these objectives yields better results than the traditional approach, when the weights are selected with parameter tuning. Finally, multiobjective trees are tested in randomized setting similar to random forest models and compared with the traditional time series forecasting models. More competitive results are obtained when only one decision tree is fitted compared to randomized setting.

ÖZET

TAHMİNLEME İÇİN ÇOK AMAÇLI AĞAÇLAR

Zaman serisi tahmini önemli bir görevdir ve eğitim, finans, tıp ve üretim gibi çeşitli alanlarda kullanılabilmektedir. Zamansal verilerin yaygın olarak kullanılması, veri madenciliği alanında birçok araştırma ve çalışmanın yapılmasına neden olmuştur. Zaman serisi analizi modellerinin yanı sıra, ağaç tabanlı yöntemler de zaman serisi analizi için yaygın olarak kullanılan yöntemlerdir. Ağaç tabanlı modeller karmaşık veri kümelerinde doğru sonuçlar sağlamaktadır ve doğrusal olmayan ilişkileri açıklayabilirler. Ancak zaman serisi verileri, bir değişkenin belirli zaman aralıklarında ölçülmesiyle elde edilir ve çoğu zaman öngörülebilir dalgalanmalar içerir. Ağaç tabanlı modellerin kullanımında, zamansal ilişki ve mevsimsellik modelleme sürecine dahil edilmelidir. Regresyon ağaçları, tüm veri noktalarını birbirinden bağımsız olarak dikkate almaktadır. Bu nedenle, zaman serisi tahmininde kullanımları için önceden işleme veya sonradan işleme yöntemlerine ihtiyaç vardır. Diğer yandan, ağaçların öğrenme algoritmasındaki değişiklikler ile zamansal etki ve mevsimsellik sağlanabilir. Çok amaçlı ağaçlar, karar ağaçlarının öğrenme sürecinde değişiklikler yapan ağaç tabanlı bir topluluk yaklaşımıdır. Karar ağaçlarının amacına ek olarak, bölme kararı sırasında dikkate alınacak iki yeni amaç daha eklenmiştir. Bunlar düğüm içerisindeki gözlemlerin zamansal yakınlığı ve gözlemler arasındaki mevsimsel mesafedir. Bu üç hedef farklı ağırlıklarda birleştirilerek tek bir karar metriği oluşturulur ve bu karar metriğine göre bölme işlemi yapılır. Deneylerde beş farklı sentetik veri seti ve üç adet zaman serisi veri seti kullanılmıştır. Bu hedefleri eklemek, parametre seçimi sonucunda belirlenen ağırlıklarla geleneksel yaklaşımdan daha iyi sonuçlar vermektedir. Son olarak, çok amaçlı ağaçlar rastgele orman modellerine benzer şekilde rastgele ortamda test edilmiştir ve geleneksel zaman serisi tahmin modelleri ile karşılaştırılmıştır. Tek bir karar ağacı uygulandığında rastgele ortama kıyasla daha rekabetçi sonuçlar elde edilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

a	Weight of sum of squares error objective
b	Weight of temporal closeness in nodes objective
c	Weight of seasonal distance between observations objective
DT_i	i^{th} decision tree model
k	Time difference threshold
P_i	Prediction value obtained from i^{th} decision tree
R_i	i^{th} hyperrectangle
S_i	i^{th} sample created from train set
T_{in}	Time difference in n^{th} terminal node of i^{th} decision tree
X_t	Observation at time t
$_{R_j}$	Mean response value of observations in hyperrectangle j

LIST OF ACRONYMS/ABBREVIATIONS

ACF	Autocorrelation Function
AR	Autoregression
ARIMA	Autoregressive Integrated Moving Average
DIST	Seasonal Distance Between Observations
EBLR	Explainable Boosted Linear Regression
MA	Moving Average
MTCE	Mean Temporal Closeness Error
MTF	Multiobjective Trees for Forecasting
MSDE	Mean Seasonal Distance Error
RMSE	Root Mean Square Error
RSS	Residual Sum of Square
SARIMA	Seasonal Autoregressive Integrated Moving Average
SSE	Sum of Squares Error
TBMA	Tree-Based Moving Average
TDT	Traditional Decision Tree
TIME	Temporal Closeness in Nodes

1. INTRODUCTION

Time series forecasting is a crucial task, and a significant number of methods are proposed with the increasing use of temporal data over time. Time series data is a sequence of data points obtained by measuring a variable with certain time intervals. Predicting the future data points using past observations is the basis of time series forecasting. Since time series data can be created for all variables changing over time, the application areas are wide and can be used in various sectors such as finance, medicine, production and education [1]. Predicting stock prices [2], weekly sales [3], need for medical resources, the probability of weather events are some of the application areas. One of the important points to obtain accurate results is to consider the temporal relationship in the data during modelling. Since time series forecasting is a dynamic problem by its structure and affected by many factors that change over time, it contains some factors such as weather conditions, macroeconomic conditions, seasonality and periodicity. Modeling these factors has some difficulties. In addition to these, the features that are not in the form of a time series can also affect time series data, and the relationships between variables can be nonlinear. When the data is high dimensional, efficiently discovering useful information becomes very difficult due to curse of dimensionality and increased probability of overfitting. All of these cause the prediction to be complex and increases the computational load.

In traditional models such as autoregression (AR), moving average (MA), autoregressive integrated moving average (ARIMA) and seasonal autoregressive integrated moving average (SARIMA) are used for modeling of autocorrelations. Autocorrelation refers to the correlation of the same variables between consecutive time intervals [4]. The basis of MA models is creating a new series by taking the average of previous observations in the original time series data. Considering the ease of application and low computational load, it is a highly effective method. However, it is not appropriate to use in complex datasets as it cannot evaluate the relationships between variables. Autoregression model depends on lagged versions of response variable. ARIMA and SARIMA models are the combination of both autoregression and moving average mod-

els. The only difference between ARIMA and SARIMA models is that SARIMA model is used when the time series data is seasonal. They are more advantageous than MA model as it can model the trend, noise or seasonality components of time series data. Thus, the temporal relationship and seasonality in the data are examined. However, since these are univariate models and assume linear relationships between observations, they may not be sufficient for prediction problems that involves many features and nonlinear correlations.

The use of tree-based models is also very common for time series forecasting, as they can give good results in complex datasets and also explain nonlinear relationships. Decision trees are used for both regression and classification tasks. Regression trees recursively partitions the feature space and create certain rules that defines target variable. The mean of the observations in each rule is presented as a prediction. Since decision trees can also be expressed visually, explainability of the model is more simple than other machine learning models. Regression trees are greedy because of their structure and result in high variance predictions. Tree-based ensemble models are used to solve this problem. Random forest model is a tree-based ensemble model fitted by the use of bagging method. Instead of learning a single decision tree, random forest model uses multiple decision trees [5]. The aim of the bagging is creating subsets from original data chosen randomly with replacement. Each subset of data is used for the decision tree learning. Tree-based ensemble models are beneficial for time series forecasting, since they consider the relationship between target variable and predictors and handle nonlinear relationship [6]. However, there are some shortcomings in considering the temporal relationship between observations because it considers the observations as independent points. Feature engineering can be applied to take into account the temporal effect. However, it becomes complex to determine the variables suitable for each time series when working with multiple time series. If the learner is greedy, the variables created with preprocessing may prevent to capture the actual relationships between variables and can not model as in the time series analysis models.

On the other hand, neural networks and deep learning algorithms are also used for time series prediction. Some application areas of deep learning models used for time series forecasting are stock market movements estimation [7], forecasting aggregate retail sales prediction [8] and temperature forecasting [9]. The main models are recurrent neural network (RNN), convolutional neural networks (CNN) and long short term memory (LSTM). There are many studies involving the combination of these models and these algorithms are advanced for learning complex structures. For example, hybrid techniques that combine linear and nonlinear models such as ARIMA and ANN can split a time series into linear and nonlinear components and give better results than a single time series model [10]. However, explanation of learning structure is harder than other models. Also, creating a model requires a long processing time [11]. In the study that compares a tree-based machine learning model with deep learning models, it is seen that results are very similar to each other in terms of prediction accuracy [12]. Therefore, the findings suggest that tree-based models are more prominent than deep learning models for time series forecasting, and significant improvement in model accuracy can be achieved with the changes in tree-based models [13].

Postprocessing methods can be used to handle with the problems encountered in tree-based models. Tree-Based Moving Average (TBMA) model [14] can be given as an example of postprocessing models. It is a model created for the problems of tendency to overfit and the curse of dimensionality when using datasets with many variables. In TBMA approach, multiple decision trees are fitted using a set of rules based on grouping parameters. Thus, multiple candidate forecasts are created that considers the similarities between the train set and test set observations. The prediction value is calculated by determining recent observations and taking their median. In this respect, the temporal difference between the observations is taken into account. However, seasonality is represented with a categorical feature, and the seasonal distance between observations are not taken into account. TBMA is a successful postprocessing model and gives good results in terms of model accuracy. On the other hand, it is possible to add temporal relation and seasonality with the changes that can be made during the learning phase of trees.

Since tree-based models consider observations as independent data points, the temporal relationship cannot be expressed. Time index, which indicates the temporal order of the observations, can be added to the feature set as a new variable to express the temporal relationship. However, the decision tree algorithm aims to reduce the sum of squares error during the split and does not use the time index feature in the partition. For example, predictions for temporal nodes in a decision tree are calculated using both future and past data. In cases where the time difference is high, prediction of target variable may not be calculated as accurately as desired. In Figure 1.1, the changes in average hourly electricity consumption and temperature of Turkey over 4 years are visualized.

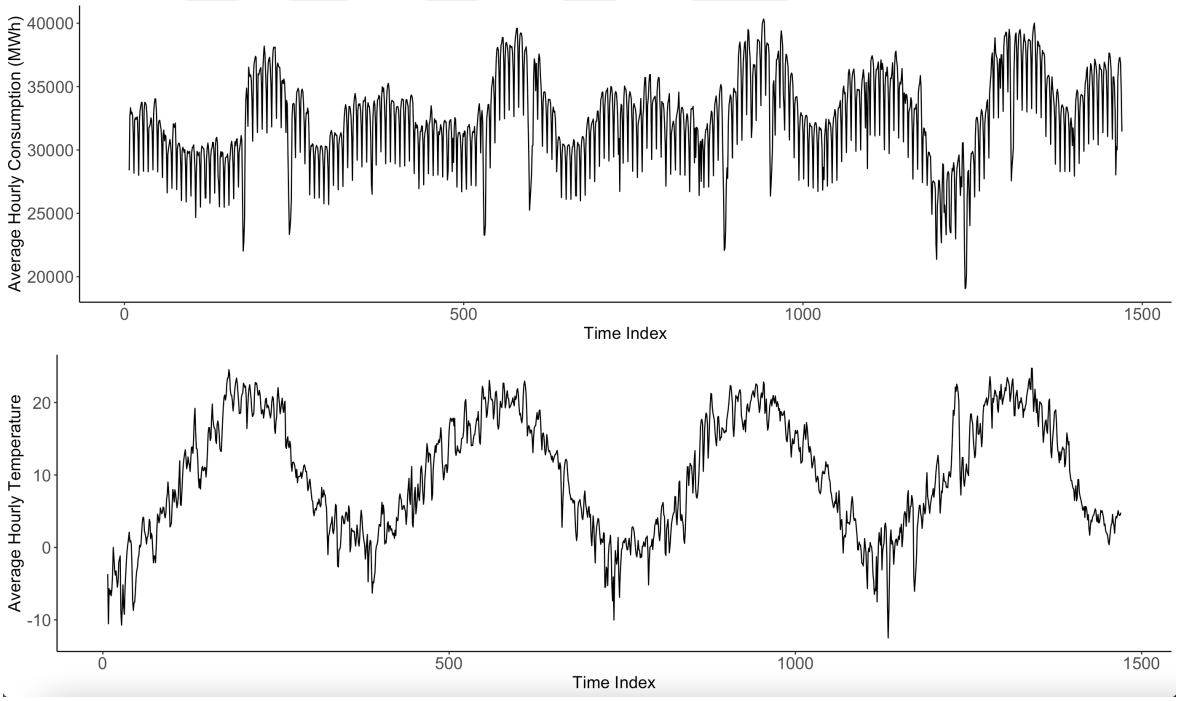

Figure 1.1. Average hourly electricity consumption and temperature in Turkey.

Simple regression tree fitted with the use of day of week and temperature features from electricity consumption dataset is shown in Figure 1.2. For each node, the prediction value and the ratio of the number of observations in that node to the total number of observations are given.

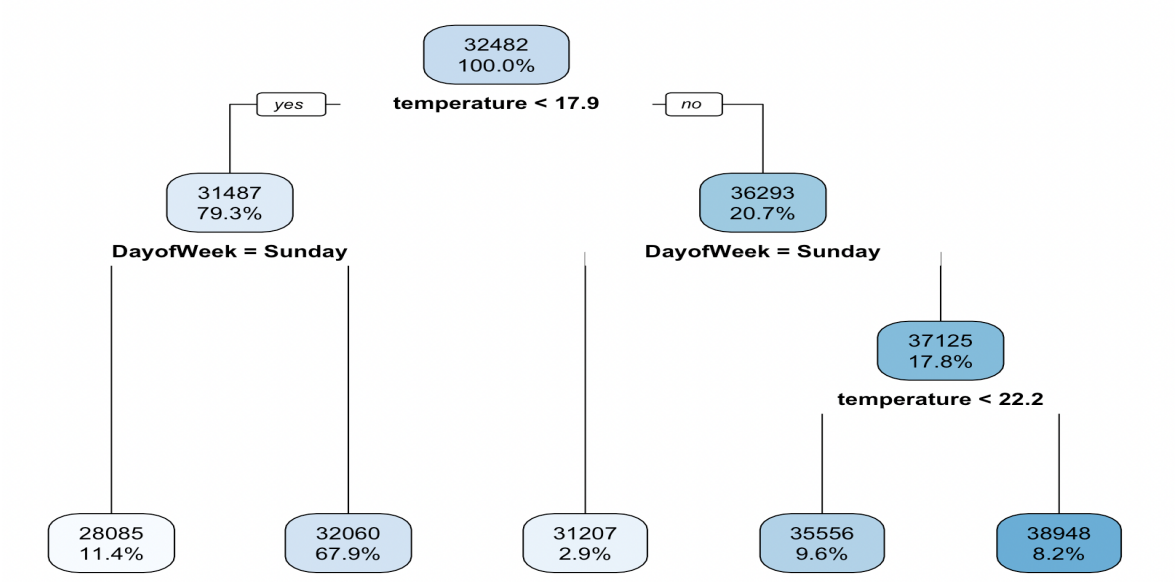

Figure 1.2. Regression tree example that uses day of week and temperature features.

Splitting is made over the temperature value and a single day of the week. The consumption values of the observations for each node are given in Figure 1.3. It can be observed from this figure that taking the average value for each node results in inaccurate predictions. In cases where the temperature value is less than 17.9, the same value is predicted for all days except Sunday. Node 3 and 4, in Figure 1.3, represent these observations. When the actual values of target variable are examined, it is seen that there is seasonal relationship depending on the days of the week. So, it can be said that the seasonality information is not sufficiently explained. Also, the target variable is not distributed around the average value. For example, when the temperature decreases, consumption increases and rises above the average value. Since time index is not used in the partition, this relationship cannot be expressed sufficiently. This shows that the temporal relationship and seasonality should be taken into account in tree-based models. When these are not considered, the probability of inaccurate predictions is higher. Postprocessing can be used to solve this problem. For the observation at time t, the prediction can be made by taking the average of the recent data points rather than taking the average of all the observations in the same node. However, considering temporality and seasonality after learning may lead to

inaccurate inferences in learning.

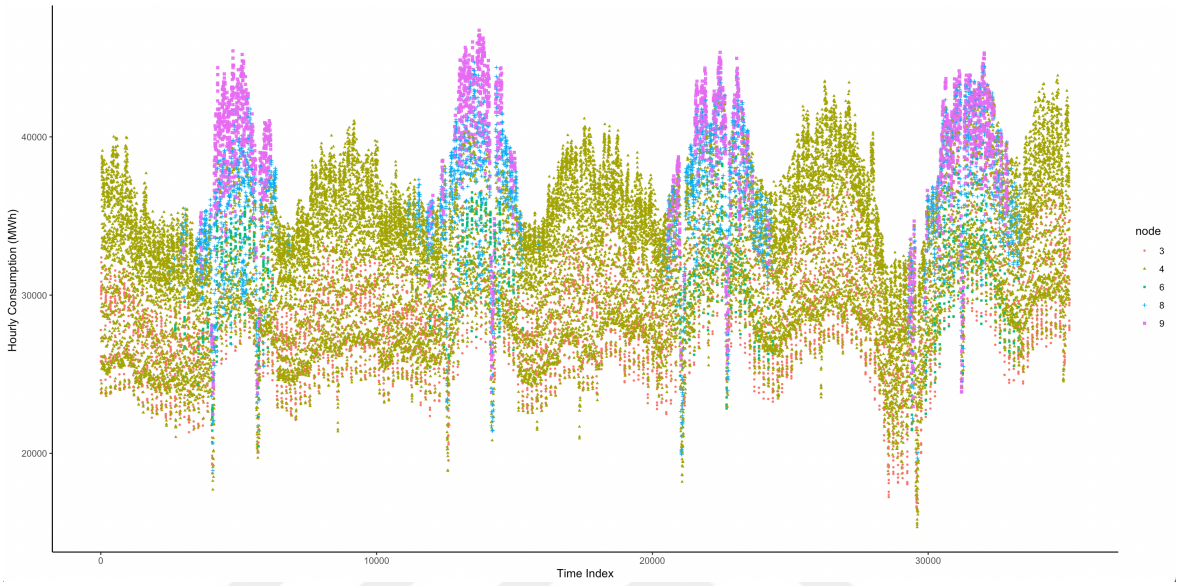

Figure 1.3. Decision tree nodes of electricity consumption values.

In this study, temporal closeness and seasonality are considered in the splitting process of the decision trees. Since there is no example that considers these relationship in the learning process, this is a novel study working on the learning phase of the trees. Temporal closeness is expressed by the time difference of the observations in the node. For example, if a node formed as a result of a decision tree partition has the 10^{th} and 90^{th} days of the year, it can be said that the temporal closeness for that node is 80 days. Seasonality is represented with the seasonal distance between the observations. Adding these as factors to consider during splitting makes decision trees multiobjective trees. It is observed that trees, which uses this splitting process, give better results in some test problems. In addition, random versions of multiobjective trees are tested.

This thesis is organized as follows. In the next chapter, background information is provided. In Chapter 3, literature review about similar studies is given. The methodology of the model proposed in this study is explained in Chapter 4. The experiments and results are presented in Chapter 5. Finally, the main conclusions are summarized in Chapter 6.

2. BACKGROUND

In this chapter, statistical time series forecasting methods are introduced first. Afterwards, the principals of regression trees and tree-based ensemble models are explained.

2.1. Statistical Time Series Forecasting Methods

Frequently used methods for time series forecasting are autoregression (AR) and moving average (MA) models. In AR model, the inputs of the regression equation are the previous observations. It assumes that the response value is a linear combination of its past values and random error term. On the other hand, moving average model uses forecasting errors of past in order to predict future data points. Simple moving average (SMA) is given by

$$SMA_k = \frac{1}{k} \sum_{t=n-k+1}^{n} y_t. \tag{2.1}$$

y_t is the actual value of t_{th} data point. n is equal to the total number of observations and k denotes the the number of periods. In SMA, the sum of the observation values of the previous n periods is divided by the number of periods. For instance, to calculate the 3-day moving average, the prediction value can be found by adding the sales values of the previous 3 days and dividing them by 3.

Autoregressive integrated moving average (ARIMA) model is a statistical time series forecasting model generated by the combination of AR and MA models. Thus, it uses both lags of time series data and the errors of past predictions for forecasting. Seasonal autoregressive integrated moving average (SARIMA) is an extension of ARIMA model and it also contains a seasonal lags. Therefore, while ARIMA is successful in explaining the temporal relationships in the time series, SARIMA model can capture both temporal and seasonal relationships as it includes seasonal component.

2.2. Regression Trees

Machine learning based models are popular for time series forecasting in addition to traditional models from time series analysis domain such as MA, ARIMA and SARIMA. Tree-based learning strategies are widely used machine learning models. Regression trees are non-parametric machine learning method used for both regression and classification. Response variable is predicted by learning simple decision rules on features of the training dataset. According to these rules, feature space is divided into hyperrectangles and target variable is calculated by taking the mean of response variables in those hyperrectangles. Therefore, decision trees can be classified as piecewise constant approximation. The regression tree building process can be explained in two steps:

(i) Divide the feature space into J distinct hyperrectangles R_1, R_2, ..., R_J.
It is a crucial criterion that none of these hyperrectangles overlap with each other.

(ii) Make a prediction for each terminal node by averaging the response values of the observations in that hyperrectangle.

In Figure 2.1, an illustration of a decision tree partition of the feature space is given.

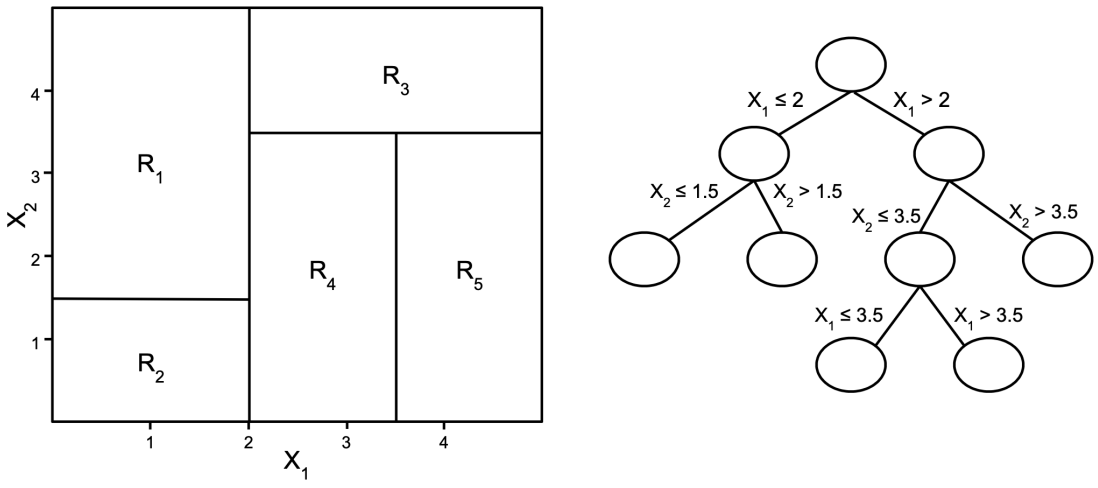

Figure 2.1. Illustration of a decision tree partition of the feature space.

In the decision tree in Figure 2.1, the feature space is split from the variable X_1 at first. The observations where the value of the variable X_1 is less than or equal to 2 is split again over the variable X_2. Thus, feature space is divided into R_1 and R_2 hyperrectangles. R_3, R_4 and R_5 hyperrectangles are created as a result of the partitions made in cases where the X_1 variable is greater than 0.4.

The aim of the partitioning of feature space is finding the hyperrectangles R_1, R_2, ..., R_J that minimizes residual sum of square (RSS). In the case where the response values of the observations in hyperrectangle J are y_1, y_2, ..., y_n, RSS can be calculated as

$$RSS = \sum_{j=1}^{J} \sum_{i \epsilon R_J} (y_i - \hat{y}_{R_j})^2 \tag{2.2}$$

and $\hat{y}_{R_j}$ can be calculated as

$$\hat{y}_{R_j} = \frac{\sum_{n=1}^{n} y_n}{n}. \tag{2.3}$$

In Equation 2.1, $\hat{y}_{R_j}$ represents the mean response value of the observations in hyperractangle j. However, it is computationally inefficient to split every possible value of feature space into J regions. For this reason, recursive binary splitting approach, a top-down greedy approach, is used in regression trees [15]. It begins from the top of the tree and all observations are in the same region at this point, then new branches are created further down on the tree by splits the feature space. Since it only considers current split for making a decision instead of looking for all future possible splits, it is a greedy approach.

Decision tree includes root node, decision nodes and terminal nodes. Parent and children nodes are also referred to as decision nodes and terminal nodes, respectively. Root node consists of all observations in the training set. The nodes formed by the partition of the root node are called decision nodes if they also have any sub-nodes. Nodes that do not split are called terminal nodes or leaf node. Simple example for the structure of decision trees can be seen in the Figure 2.2.

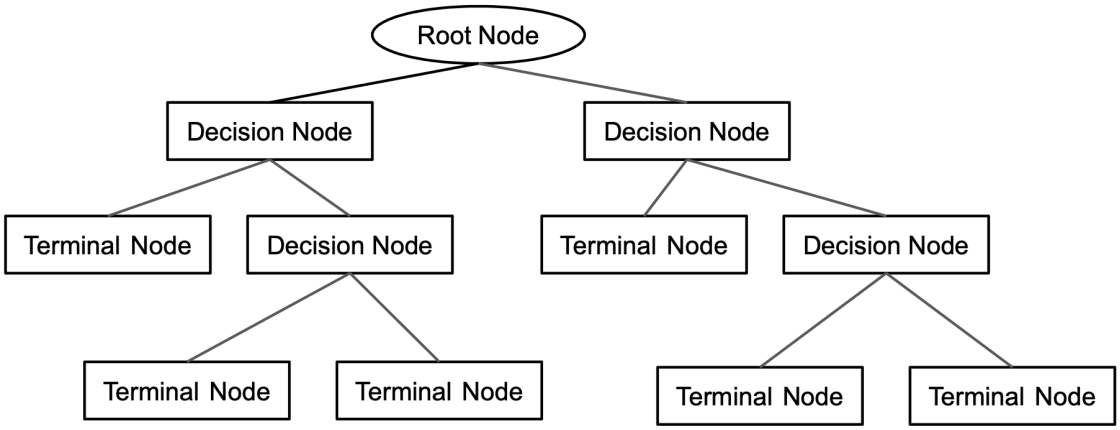

Figure 2.2. Simple example of regression tree structure.

Decision tree method is easier to interpret since it has more similarity to people's decision making process compared to other prediction algorithms. Being able to be expressed visually also contributes to this and can be shown as a great advantage. In addition, regression trees can handle both numerical and categorical variables easily. Even if irrelevant variables are included in the model, they can make accurate predictions. It is an important advantage that the learning process is faster than other models. On the other hand, their prediction accuracy is not better than other prediction methods and they have high variance. The use of tree-based ensemble models, which is created by the combination of high number of decision trees, can be beneficial in terms of increasing the prediction accuracy. However, constant prediction is made for each observation in a terminal node. When the data is limited, it is not possible to capture more complex patterns and this can be problematic in terms of prediction performance.

Simple regression tree example is given in Figure 2.3. Target variable in this regression tree is median value of house prices. Target variable is explained with crime rate in the town, number of rooms in the residence, percentage of lower status of the population and distance to employment centers. Prediction value is calculated as the mean of the observation in the given terminal node. Also, the percentage values in

the nodes indicates the percentage of observations that belongs to that node. This regression tree is fitted from a total of 500 observations (days).

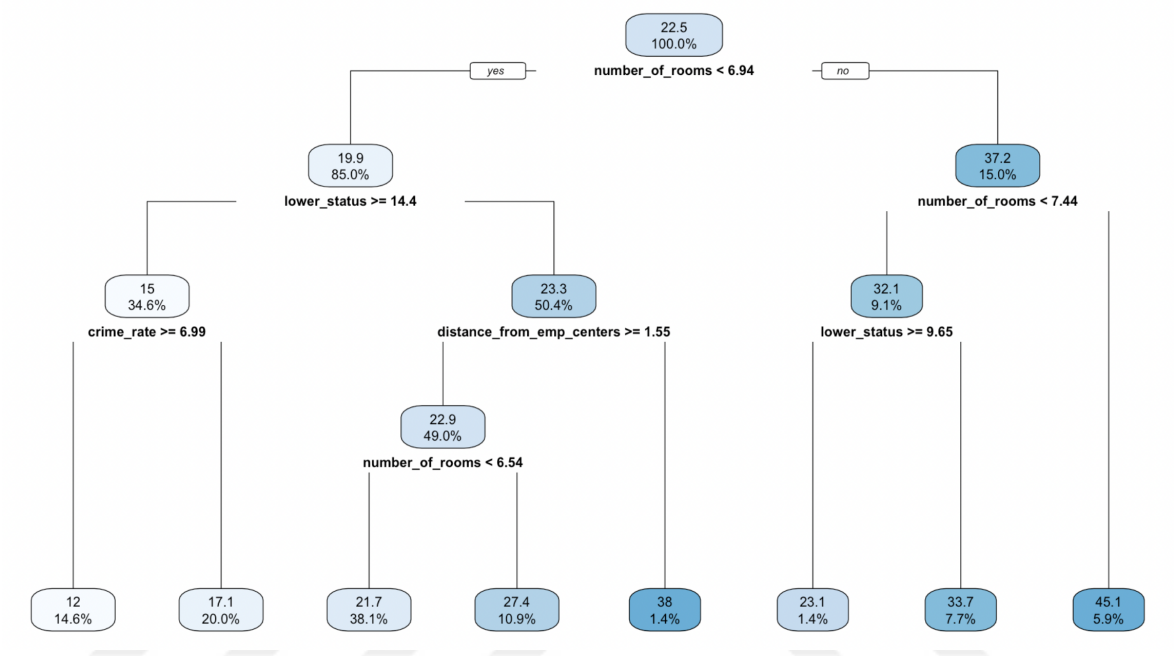

Figure 2.3. Simple example of regression tree.

2.3. Tree-based Ensemble Models

Tree-based ensemble models are highly reliable models that can be applied to a wide range of problems and real-world applications [16]. They are developed to increase the prediction accuracy by reducing the variance in decision making systems and currently show successful results in numerous machine learning problems such as feature selection, class-imbalanced data and incremental learning [17]. Tree-based ensemble models are mainly bagging and boosting. Random forest is one of the used bagging models.

Bagging is a general approach that means to create multiple versions of a prediction model and use these to obtain an aggregated estimator [18]. As tree-based ensemble model, bagging aims to reduce the variance by generating large number of decision trees. Decision trees are fitted with random samples with replacement from

the training set and their predictions are used for calculating the overall prediction. This random sample selection step is called as bootstrapping. The number of instances used in each decision tree is equal to the number of features in the original training set. The process of bagging can be seen in the following figure.

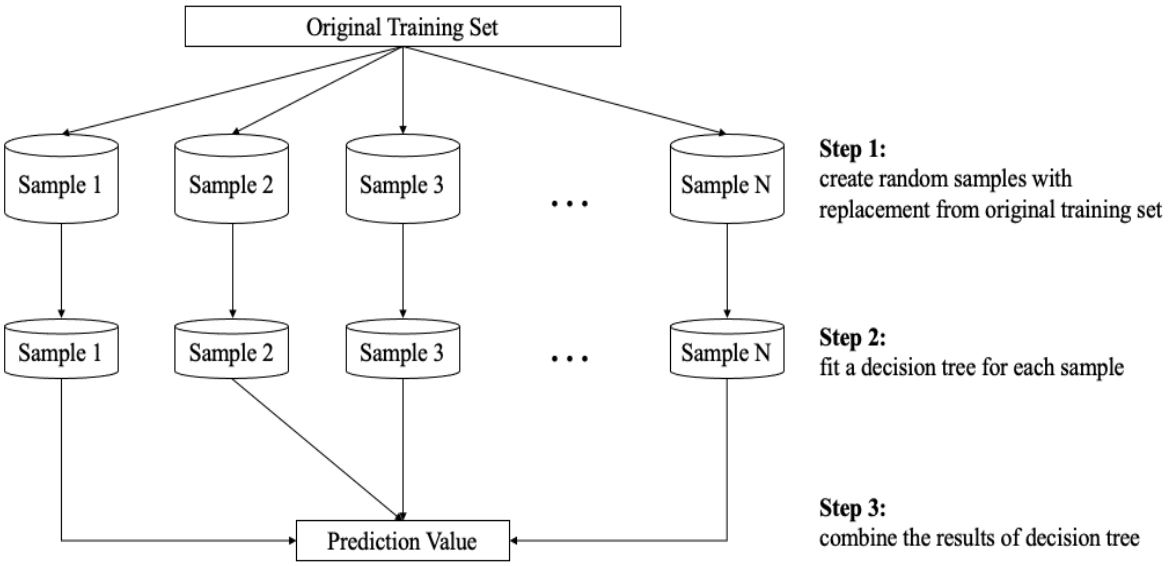

Figure 2.4. Bagging process.

Random forest is basically bagged trees in a specific way and it can be used for both regression and classification. It is said that random forest, which is a data-driven statistical method, is quite simple in terms of explainability and the bagging algorithm is used to obtain a *strong learner* [19]. In the training process, a large number of decision trees are generated similar to bagging process. As opposed to bagging algorithm, random forest model selects m predictors randomly from p predictors in the splitting phase. Therefore, random forest is successful for modeling high dimensional complex datasets.

2.4. Performance Metrics

In the experiments of this study, the main performance metrics used to measure the accuracy of the models are root mean square error (RMSE), mean temporal closeness error (MTCE), and mean seasonal distance error (MSDE). The aim is to select the model that minimizes RMSE. RMSE is given by

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(y_i - \hat{y}_i)^2}{n}}. \tag{2.4}$$

MTCE and MSDE is used for comparing temporal and seasonal error of predictions. MTCE is calculated as

$$MTCE = \frac{\sum_{i=1}^{n} T_{max} - T_{min}}{n} \tag{2.5}$$

and MSDE is calculated as

$$MSDE = \frac{\sum_{i=1}^{n} DIST_n}{n}. \tag{2.6}$$

While y_i represents the actual value of i^{th} data point, $\hat{y}_i$ denotes the predicted value of i^{th} observation. n is equal to the total number of observations. When calculating RMSE, the difference between the actual and predicted value is squared and divided by the total number of observations. This calculation is made for each data point and the square root of their sum is taken. When calculating MTCE, T_{max} represents the maximum time index in the node where each observation belongs, while T_{min} represents the minimum time index. The difference between T_{max} and T_{min} for each observation is calculated and averaged. When calculating MSDE, $DIST_n$ represents the seasonal distance in each node where each observation belongs. The average of $DIST_n$ values is calculated. Detailed calculations are explained in Chapter 3.

3. LITERATURE REVIEW

This chapter introduces a literature review involving the use of tree-based models for time series forecasting. Also, the approaches used to deal with the temporal and seasonal relationships in time series forecasting are mentioned.

Regression trees and tree-based models are commonly used models for time series forecasting [20]. Regression trees are advantageous as they can handle non-linear relationships and provide interpretable results. However, in order to create model that can explain the change over time, it is necessary to create new variables by feature engineering. The temporal relationship between observations and seasonality can be expressed with these new variables.

For example, future months can be predicted for time series data at the monthly level. A categorical variable that will represent each month or the actual target value in the same month of the previous year can be given as examples of these variables [21]. With the addition of these features, time series data can be analyzed in more detail. Regression tree parameters should be determined well. Trees that are too shallow may be insufficient to learn the data, while trees that are too deep can cause overfitting. In M5 forecasting competition [22], tree-based algorithms are the winning models and their popularity is increased over time [23]. Sales demand forecasting [24], wind power forecasting [25], cryptocurrency price trend forecasting [26] are some examples of real-world applications. Temporal relationship and seasonality are taken into account with feature engineering, but there is no change in the learning algorithm of the regression trees.

In the random sample selection phase of tree-based methods, all observations have the equal probability for selection. However, recent observations contain more up-to-date information for prediction, as changes occur over time in time series data. For dealing with this problem, a new time-based ensemble model, *temporal bagging (T-bagging)*, is proposed [27]. This model increases the change of selection of recent

observation by using weighted random sampling selection strategy. Another study uses bagging algorithm is *temporal sampling forest (TS-F)* [28]. For temporal classification tasks, this method combines bagging and temporal randomization procedures and increases the capability of handling temporal data. In addition, there are many models developed for increasing the ability of the bagging algorithm in terms of prediction accuracy. For instance, A-Bagging algorithm applies clustering and weighted majority voting [29]. Also, different predictor subsets produce good prediction results and increases the performance [5]. For example, random forest based risk adjustment model [30] uses high dimensional healthcare data to predict health costs according to relative risks of a patient and gives accurate results. In addition, random forest is an effective method for predicting missing data and it can provide high accuracy even if the missing data rate is high [31].

One of the methods that can be applied to improve the model accuracy of regression trees is postprocessing. Since tree-based models are simpler and easier to explain than deep learning models, improving model accuracy with preprocessing or postprocessing can be an effective method. Tree-Based Moving Average (TBMA) [14] and Explainable Boosted Linear Regression (EBLR) [32] models are good examples for this. TBMA approach fits large number of decision trees and uses set of rules based on grouping parameters. In this way, it creates multiple candidate forecasts and calculates the prediction value by taking their median. Unlike other models, it determines the similarities between the observations used in the training set of the decision tree and the test observations, and ensures that the similarity information is more prominently included in the model. Also, probabilistic estimation can also be made using the distribution of candidate estimates instead of taking their median. It has been stated that another usage area of the TBMA model is feature extraction. In the study, TBMA model is proposed for forecasting problems with a large number of predictors and it is used to predict the short-term electricity load forecasting (STLF) problems. According to the results, model provides accurate results and the best performing model is extreme gradient boosting (XGB) developed with TBMA approach. However, TBMA model can be insufficient for capturing the dynamic system of time series datasets, since it uses the linearity assumption of moving average. Although it considers tem-

poral relationship, it does not take into account the seasonality. Seasonality is only expressed as a categorical variable.

EBLR approach, used for time series forecasting, starts with a base model and explains the model's errors with regression trees, so it can be said that it is an iterative technique. Since the path with the highest errors is included in each iteration as a new variable in the new decision tree, nonlinear relationships are also effective in the model. Adding these paths as a new variable also minimizes the loss of information during learning process. As a result, it can be said that the EBLR approach is an effective method for both point and probabilistic forecasts. Despite the fact that the learning process is carried out in a similar way to gradient boosting and random forest models, this model is less complex and easier to interpret compared to the other models.

Regression trees consider a single objective in the learning process, which aims to maximize the improvement on the sum of squares error. In this thesis, temporal closeness and seasonal distance objectives are added to the learning algorithm of regression trees. There is no similar study to this, however there are multiobjective trees which are used in multi-target datasets. The single objective models are insufficient for dealing with complex issues in the real world. Decision trees are an effective model for solving complex problems and they can be adapted to handle multiple objectives [33]. So, multiobjective decision trees can be fitted to optimize multiple objectives. In [34], multiobjective trees are introduced as the decision trees that predict multiple target response at the same time. Bagging and random forest are applied to multiobjective decision trees and the performances of all models are compared with each other. According to the results of the article, the accuracy of multiobjective trees is improved with the use of bagging or random forest. This shows that including randomness to the learning process can be beneficial for prediction performance. It is also an important output that multiobjective decision trees give at least as good results as single objective decision trees. In addition, multiobjective random forest gives significantly better results from multiobjective bagging similar to single objective models. Multiobjective decision trees are not used for time series forecasting. Since they can capture complex relationships and deal with imprecise data, they are advantageous to use in time series

forecasting problems.

Multiobjective Trees for Forecasting (MTF) model changes the learning algorithm of decision trees to consider temporal relationships and seasonality. In each split, traditional decision tree learning algorithm aims to maximize the improvement in sum of squares error. In the learning process of MTF model, minimizing temporal closeness in nodes and seasonal distance between observations are also added as new objectives. Therefore, temporal relationships between data points are more considered than traditional decision tree learning algorithm. As an another important factor, seasonal difference between observations is added as new objective to understand the trend in time series data and give importance in the partitioning. It has similar structure as multiobjective trees that make predictions for multi targets, but time series data has single response variable. New considerations are combined and expressed as a single objective and a single response variable is predicted. It is not a preprocessing model, as the methodology does not include the seasonal decomposition or the addition of extra features to the learning. Also, it is not a postprocessing model that will manipulate the trees to learn temporality or seasonality. This is a novel study since there is no similar study that considers temporal closeness and seasonal distance in the learning phase of trees. The performance of the MTF model is evaluated with using different real-world time series datasets and compared with time series models and traditional decision tree algorithm. It is expected that the new learning algorithm will provide more accurate results while taking temporal relationships and seasonal patterns into account.

4. METHODOLOGY

This chapter explains our tree-based learning approach for time series forecasting. The proposed model, which includes the changes made in the learning process of regression trees, is called Multiobjective Trees for Forecasting (MTF).

Regression trees are trained by recursively split of feature space based on some rules. While partitioning of the feature space, all possible splits are evaluated and sum of squares error (SSE) is calculated for each possible split. In regression trees, the aim of split is to select the highest decrease in SSE compared to parent node. Thus, SSE will be minimized as a result of all splits. However, splits evaluated in this way do not consider the temporality between data points. One disadvantage is that the split which works best for SSE minimization may contain data that are distinct in time. In terms of time series forecasting, recency of the data points are crucial. Nevertheless, containing greater number of recent data points compared to other nodes is not taken into consideration in traditional regression tree algorithm. In addition, seasonality is an important factor for time series forecasting. Seasonality indicates the predictable fluctuations in time series data and it can be beneficial to understand the trend. Almost all time series datasets have similar behavior at certain time intervals such as hourly, daily, weekly or monthly. For example, electricity consumption may be higher in summer than the other seasons due to the use of air conditioners. Adding seasonality effect to the model can improve the prediction accuracy, as it provides to capture periodic pattern in the dataset. Since the sine and cosine functions are periodic, it can be said that using these functions is one of the ways to measure the seasonality effect.

Simple regression tree learned from day of week and temperature features from an electricity consumption dataset is shown in Figure 4.1. In the terminal nodes, the percentage of observations that comply with the rule and the estimated value as the mean of the observations is given. This regression tree is fitted from a total of 1470 observations (days) and separates the data based on different temperature values and day information.

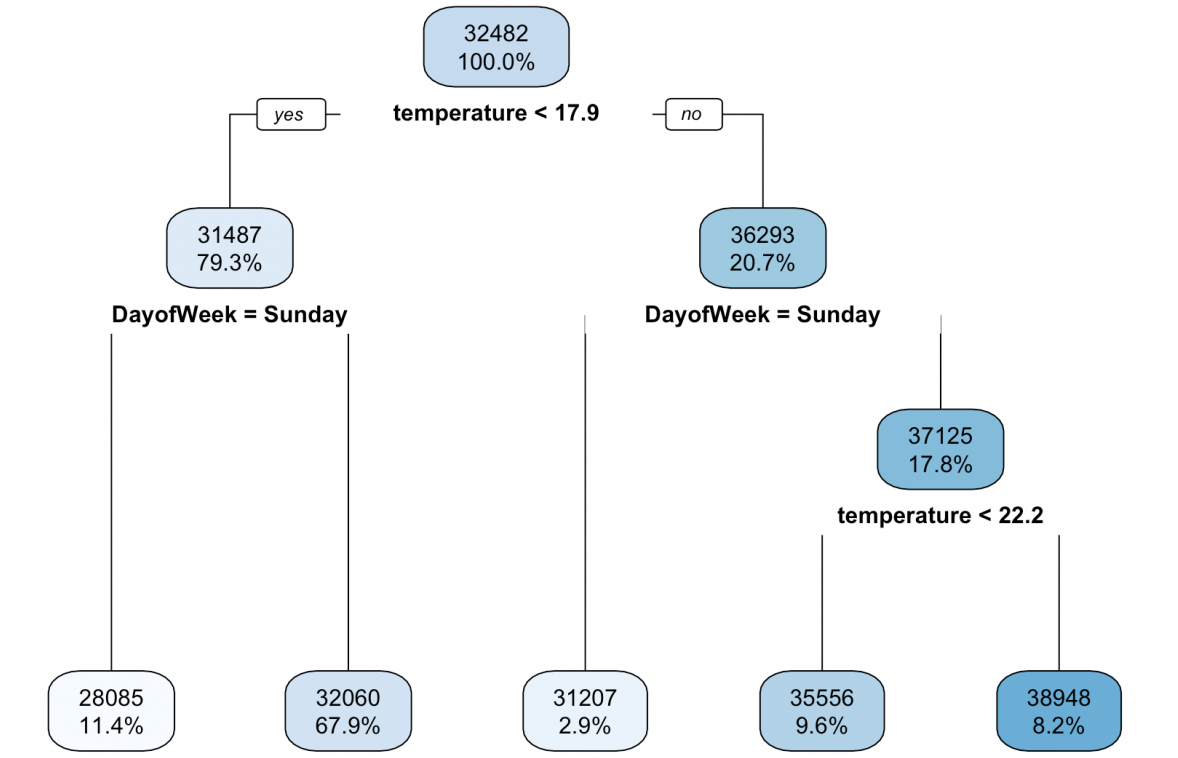

Figure 4.1. Regression tree model fitted with temperature and day of week features.

One way to deal with the temporality problem of regression trees in forecasting tasks is postprocessing the trees to account for recency of the observations. After creating a large number of decision trees, the effect of the recent data point in estimation can be increased with the certain rules to be applied on these trees. Tree-based moving average (TBMA) model [25] is one of the examples of this approach. Another way to handle with temporality problem is changing the learning process of regression trees. Regression trees divide the feature space into various regions by splitting based on the certain rules and make predictions accordingly. Split evaluation can be manipulated by using alternative criteria. Two alternative forecasting models including these methods are explained in details.

4.1. Dataset Definition

Consider that a time series dataset X with length T. Let X_t stands for the observation at time t. Each X_t has a feature vector *(1 x f)* which provides information about X_t. In the case of f features, the features of X_t are expressed as F_1, F_2, ..., F_{ft}. This feature set is used for prediction of X_t. Dataset D is formed from time series dataset X and feature vector F, and turns into a matrix of length T. The features used can be external or time series variables. Promotion days and temperature values can be given as examples of external features, while time series features can be lag variables. For example, lagged values of the time series observations can be used as a regressor as in autoregressive models. X_{t-1} refers to Lag-1 and indicates the response value of previous data point.

4.2. Multiobjective Trees for Forecasting

Decision metric used in the partitioning process of our approach consist of a combination of three criteria. These objectives are decreasing SSE in each split, reducing the time difference within the node and minimizing the seasonal distance between the observations. These are calculated for parent node and all possible child nodes at each split. Their calculation methods and combination as a decision metric are as follows.

4.2.1. Sum of Squares Error (SSE)

Suppose dataset D is the parent node. Prediction value of this parent node is calculated as the average of all observations and represented by $\hat{y}$. SSE of parent node, SSE_0, can be calculated as

$$SSE_0 = \sum_{t=1}^{T} (y_t - \hat{y})^2. \tag{4.1}$$

In order to perform partitioning, we first select the variable X_1 and assume that splitting point is s. The first region which is formed as a result of this split is given by

$$R_1 = \{X \mid X_1 \leq s\} \tag{4.2}$$

and the second region is given by

$$R_2 = \{X \mid X_1 > s\}. \tag{4.3}$$

The notation of R_1 represents the region in the feature space where the value of X_1 is less than or equal to s. As a result of the split, SSE of the first child node can be calculated as

$$SSE_1 = \sum_{t=1}^{T_1} (y_t - \hat{y}_{R_1})^2 \tag{4.4}$$

and SSE of the second child node is calculated as

$$SSE_2 = \sum_{t=1}^{T_2} (y_t - \hat{y}_{R_2})^2. \tag{4.5}$$

Also, the total of T_1 and T_2 is computed as

$$T_1 + T_2 = T. \tag{4.6}$$

After calculation of the SSE of child nodes, the reduction in SSE obtained by this split can be found by subtracting the sum of the SSE values of the child nodes from the SSE of parent node. ΔSSE denotes the reduction in SSE and it is computed as

$$\Delta SSE = SSE_0 - (SSE_1 + SSE_2). \tag{4.7}$$

This calculation is done for all possible splitting points of all X_1, X_2, ..., X_F variables, where the feature vector is 1, 2, ..., F. After calculating all values, normalization is applied to avoid scaling problems between the values of objectives. Thus, the value of SSE to be used in split evaluation, ΔSSE_{norm}, is obtained as

$$\Delta SSE_{norm} = \frac{\Delta SSE - \Delta SSE_{min}}{\Delta SSE_{max} - \Delta SSE_{min}}. \tag{4.8}$$

4.2.2. Temporal Closeness in Nodes

Assume dataset D is the parent node. Time index in parent node is *1, 2, . . . ,* T and time difference in parent node, $TIME_0$, can be calculated as

$$TIME_0 = T_{max} - T_{min}. \tag{4.9}$$

In the calculation of $TIME_0$, T_{max} and T_{min} are used. T_{max} represents the maximum value of time index at the specified node, and it is calculated as

$$T_{max} = max(1, 2, \ldots, T). \tag{4.10}$$

T_{min} represents the value of the minimum time index, and it is calculated as

$$T_{min} = min(1, 2, \ldots, T). \tag{4.11}$$

We first select the variable X_1 and assume that splitting point is s. The first region which is formed as a result of this split is given by

$$R_1 = \{X \mid X_1 \leq s\} \tag{4.12}$$

and the second region is given by

$$R_2 = \{X \mid X_1 > s\}. \tag{4.13}$$

$T_{R_{1max}}$ and $T_{R_{2max}}$ represents the maximum time index in regions R_1 and R_2. $T_{R_{1min}}$ and $T_{R_{2min}}$ shows the minimum time index in regions R_1 and R_2. Time difference in the first child node of this split is calculated as

$$TIME_1 = T_{R_{1max}} - T_{R_{1min}} \tag{4.14}$$

and time difference in the second child node of this split is calculated as

$$TIME_2 = T_{R_{2max}} - T_{R_{2min}}. \tag{4.15}$$

The decrease in time difference which is obtained by this split can be found by subtracting the weighted average time difference in child nodes from the time difference in parent node. $\Delta TIME$ denotes the decrease in time difference. n_1 and n_2 represent the number of data points in the R_1 and R_2 regions, respectively. $\Delta TIME$ is computed

as

$$\Delta TIME = TIME_0 - \frac{n_1.TIME_1 + n_2.TIME_2}{n_1 + n_2}. \tag{4.16}$$

All possible splitting points of all X_1, X_2, ..., X_F variables are selected and this calculation is completed. Then, normalization is applied same as in the SSE calculations. Thus, the value of time difference to be used in decision metric, $\Delta TIME_{norm}$, is calculated as

$$\Delta TIME_{norm} = \frac{\Delta TIME - \Delta TIME_{min}}{\Delta TIME_{max} - \Delta TIME_{min}}. \tag{4.17}$$

4.2.3. The Seasonal Distance Between The Observations

Suppose dataset D is the parent node. The example of variables that can represent frequency or seasonality in the dataset can be given as day, week and month. The values of seasonal distance variable, S, is between *1, 2, ..., f* and seasonal distance between the observations in the parent node is $DIST_0$. For instance, while S vary between 1 and 7 for a dataset with weekly seasonality, its values are between 1 and 365 for a dataset with daily seasonality. Since seasonal variables such as hours of the day and days of the week have cyclical patterns, each of them are converted into two cyclical variables. For this transformation, the values of seasonal variable are normalized at first, S_{norm} is calculated as

$$S_{norm} = norm(1, 2, ..., f). \tag{4.18}$$

After calculating S_{norm}, two different vectors, S_{cos} and S_{sin}, are obtained by taking the cosine and sine of the normalized values. S_{cos} is computed as

$$S_{cos} = cos(S_{norm}) \tag{4.19}$$

and S_{sin} is computed as

$$S_{sin} = sin(S_{norm}). \tag{4.20}$$

If the dataset has weekly seasonality, the values of S is between 1 and 7. S_{norm} can be calculated as

$$S_{norm} = \frac{2\pi(1, 2, ..., 7)}{max(1, 2, ..., 7)} = (\frac{2\pi}{7}, \frac{4\pi}{7}, ..., \frac{14\pi}{7}). \tag{4.21}$$

After calculating S_{norm}, the cosine vector, S_{cos}, can be calculated as

$$S_{cos} = cos(0.898, 1.795, ..., 6.283) = (0.623, -0.222, ..., 0.999) \tag{4.22}$$

and the sine vector, S_{sin}, can be calculated as

$$S_{sin} = sin(0.898, 1.795, ..., 6.283) = (0.782, 0.975, ..., -0.0001). \tag{4.23}$$

If all (S_{cos}, S_{sin}) pairs are drawn, it can be said that all of them are located on a circle. Sample visualization for sine and cosine pairs of weekly seasonality is represented in Figure 4.2.

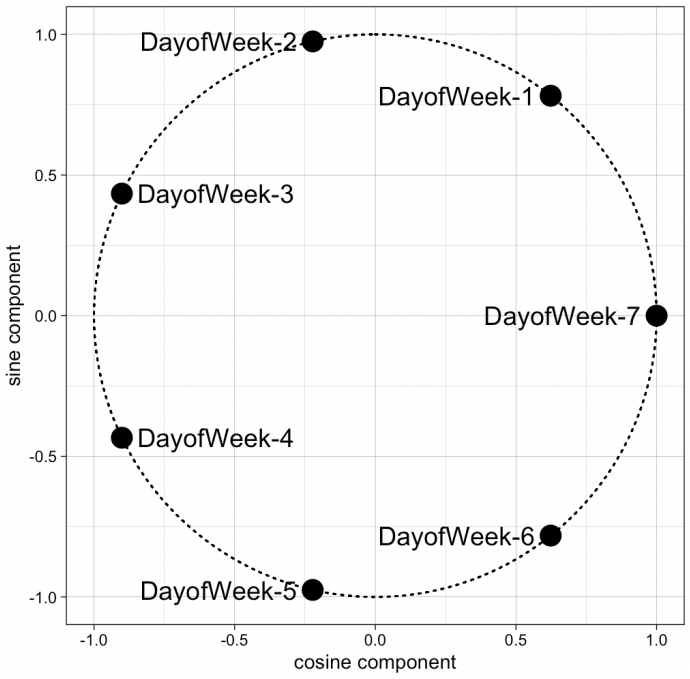

Figure 4.2. Visualization of sine and cosine pairs.

Seasonal distance in parent node, $DIST_0$, is obtained by summing the distances of all points from the center. $DIST_0$ is calculated as

$$DIST_0 = (S_{cos} - mean(S_{cos}))^2 + (S_{sin} - mean(S_{sin}))^2. \tag{4.24}$$

We first select the variable X_1 and suppose that splitting point is s. The first region which is formed as a result of this split is given by

$$R_1 = \{X \mid X_1 \leq s\} \tag{4.25}$$

and the second region is given by

$$R_2 = \{X \mid X_1 > s\}. \tag{4.26}$$

Seasonal distance in the first child node of this split is calculated as

$$DIST_1 = (S_{cos_{R_1}} - mean(S_{cos_{R_1}}))^2 + (S_{sin_{R_1}} - mean(S_{sin_{R_1}}))^2 \tag{4.27}$$

and the seasonal distance in the second child node of this split is computed as

$$DIST_2 = (S_{cos_{R_2}} - mean(S_{cos_{R_2}}))^2 + (S_{sin_{R_2}} - mean(S_{sin_{R_2}}))^2. \tag{4.28}$$

Seasonal distance difference which is obtained by this split can be found by subtracting the sum of the distance values in child nodes from the distance value in parent node. $\Delta DIST$ denotes the reduction in seasonal distance, and it is calculated as

$$\Delta DIST = DIST_0 - (DIST_1 + DIST_2). \tag{4.29}$$

Similar as the other objectives, this calculation is completed for all possible splitting points of all X_1, X_2, ..., X_F variables. Then, normalization is applied in the same way. Thus, the value of seasonal distance value to be used in decision metric, $\Delta DIST_{norm}$, is calculated as

$$\Delta DIST_{norm} = \frac{\Delta DIST - \Delta DIST_{min}}{\Delta DIST_{max} - \Delta DIST_{min}}. \tag{4.30}$$

4.2.4. Combination of Objectives

The decision metric includes the reduction in SSE, improvement in temporal closeness of data points and minimization of seasonal distance between the observations. By including each of them in the equation with various weights, this decision metric can be calculated. The aim of a split is finding the best split that minimizes our decision metric. Our decision metric, $DecisionMetric$, is calculated as

$$DecisionMetric = a * \Delta SSE_{norm} + b * \Delta TIME_{norm} + c * \Delta DIST_{norm}. \quad (4.31)$$

The values of ΔSSE_{norm}, $\Delta TIME_{norm}$ and $\Delta DIST_{norm}$ between 0 and 1 to deal with scaling problem. a, b and c are the weights of each criterion and their sum is equal to 1. By taking each of these parameters as the same, it is ensured that all criteria have an equal effect. Instead, a certain criterion can be made more effective in splitting process by increasing its weight. Also, weights that will increase the prediction accuracy can be determined.

4.3. Illustration

For a better explanation of our approach, five different synthetic datasets are created, where temporal and seasonal components are represented separately and together. The details of the datasets can be found in Table 4.1. *Temporal Dataset* is only dependent on time index and does not contain any seasonality. Also, it can be explained by autoregressive terms. *Weekly Seasonal Dataset* contains weekly seasonality and the increase in response value over time is not significant. Similarly, *Monthly Seasonal Dataset* is a time series data that only includes monthly seasonality. Finally, two datasets are created which has both temporal and seasonal components. These datasets are referred as *Weekly Seasonal Temporal Dataset* and *Monthly Seasonal Temporal Dataset*.

Table 4.1. Explanation of synthetic datasets.

Dataset Name	Explanation
Temporal Dataset	dependent to time index
Weekly Seasonal Dataset	has weekly seasonality
Monthly Seasonal Dataset	has monthly seasonality
Weekly Seasonal Temporal Dataset	dependent to time index and has weekly seasonality
Monthly Seasonal Temporal Dataset	dependent to time index and has monthly seasonality

Consider a time series data as follows, y_0 and y_1 initialized as zero. m denotes cumulative month index and it is used for explaining temporal effect. S is used for representing seasonal distance variable. It indicates day of the week for weekly datasets and month of the year for monthly seasonal datasets. S_{cos} and S_{sin} pairs represent the seasonal relationship in the dataset. *Temporal Dataset* is given by

$$y_t = -0.15y_{t-1} + 0.25y_{t-2} + (1 + \frac{m}{24}) * \mathcal{N}(3000, 100), \tag{4.32}$$

Weekly Seasonal Dataset is given by

$$y_t = -0.15y_{t-1} + 0.25y_{t-2} + 1200 * S_{cos} + 750 * S_{sin} + \mathcal{N}(3000, 100), \tag{4.33}$$

Monthly Seasonal Dataset is given by

$$y_t = -0.15y_{t-1} + 0.25y_{t-2} + 1200 * S_{cos} + 1200 * S_{sin} + \mathcal{N}(3000, 100), \tag{4.34}$$

Weekly Seasonal Temporal Dataset is given by

$$y_t = -0.15y_{t-1} + 0.25y_{t-2} + 1200 * S_{cos} + 750 * S_{sin} + (1 + \frac{m}{24}) * \mathcal{N}(3000, 100) \tag{4.35}$$

and *Monthly Seasonal Temporal Dataset* is given by

$$y_t = -0.15y_{t-1} + 0.25y_{t-2} + 1200 * S_{cos} + 1200 * S_{sin} + (1 + \frac{m}{24}) * \mathcal{N}(3000, 100). \tag{4.36}$$

In our example datasets, response variable is explained by time index, autoregressive terms and cyclical components of day of week variables. Response values of synthetic datasets can be seen in Figure 4.3.

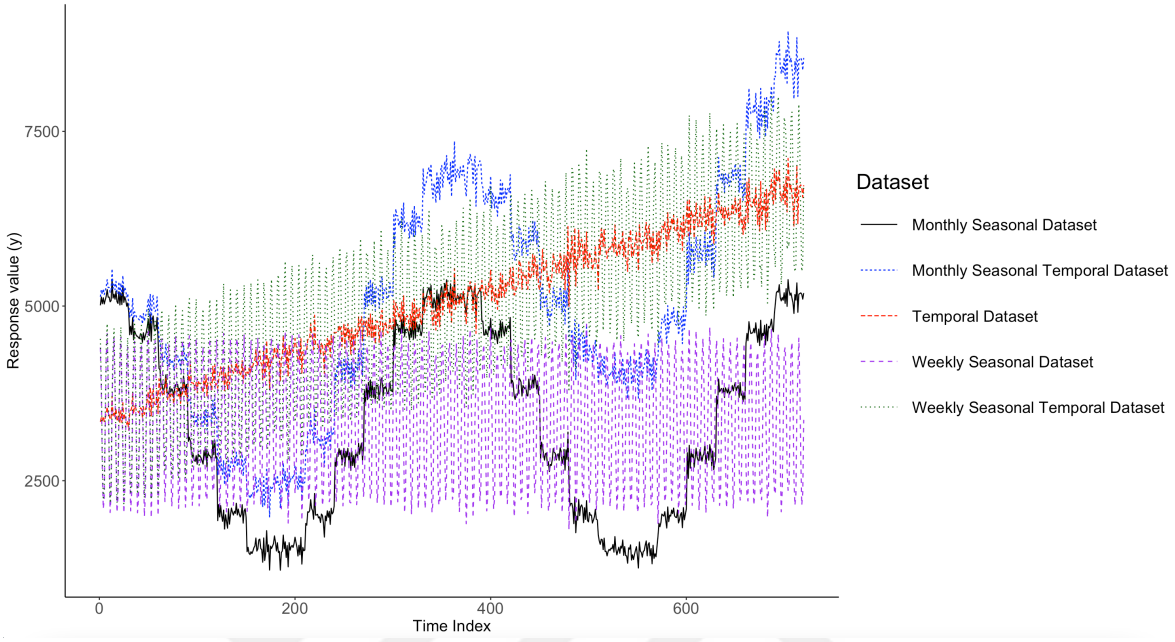

Figure 4.3. Synthetic datasets.

4.4. Objective Weight Sensitivity

In this section, the model's output is analysed with the change of objective weights and the results of the decision tree model built using the classical method and our approach are compared. As a first step, performance metrics are investigated for all synthetic datasets when TIME and DIST weights are changed between 0 and 1. In this case, SSE weight is kept constant and equal to 1. Performance metrics used are root mean square error (RMSE), mean temporal closeness error (MTCE) and mean seasonal distance error (MSDE). Then, the traditional decision tree (TDT) algorithm that aims only SSE minimization and the multiobjective tree model (MTF) that uses the split method we created are compared for all synthetic datasets. While using our approach, parameter tuning is made for obtaining more accurate results. In this case, root mean square error (RMSE) is used to make a comparison between the two models. The minimum number of observations to be able to split is equal to 20 for all decision trees.

For *Temporal Dataset*, the weight of TIME differs between 0 and 1, and DIST component has no effect on splitting. Accordingly, the change in performance metrics for train and test sets can be seen in Figure 4.4. When the weight of TIME is increased, MTCE decreases for both train and test set observations. However, although RMSE value for the train set increases with the increasing weight of TIME, RMSE of test set decreases. Since this dataset is highly dependent on time index, giving importance to TIME component has a positive effect. Also, this dataset has no seasonality. So, a significant difference in MSDE value is not expected.

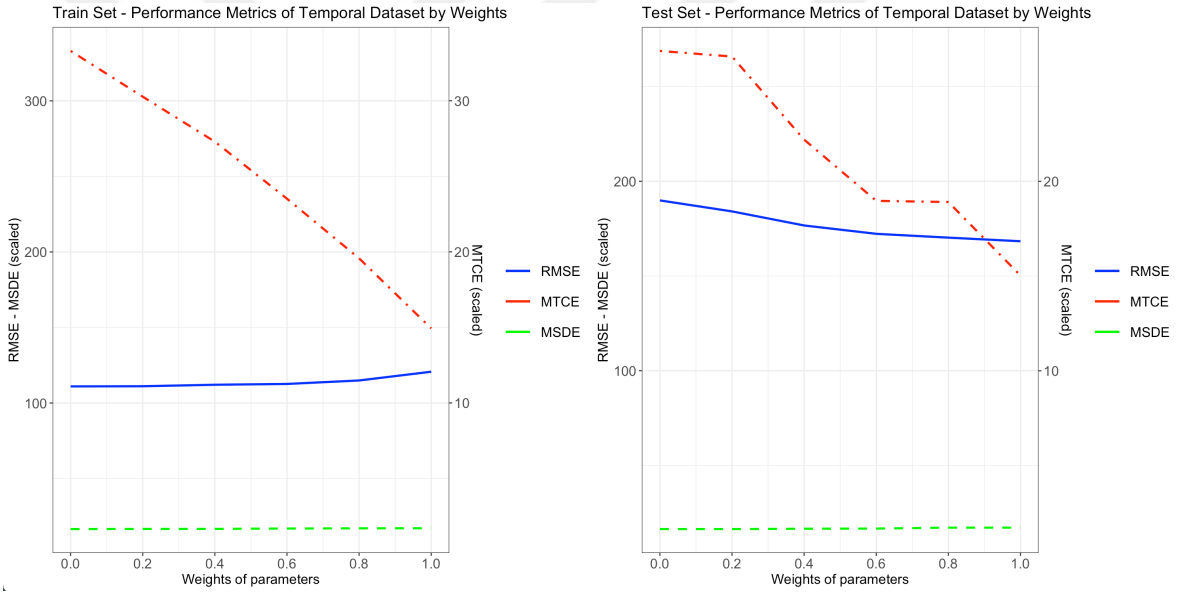

Figure 4.4. Performance metrics of temporal dataset by objective weights.

In cases where the weight of TIME component increased, smaller RMSE values can be observed. Therefore, traditional decision tree approach and multiobjective tree model with parameter tuning are compared. When two different methods are used, no significant difference is observed between RMSE values of the train set observations. However, there is a significant difference between RMSE values of test set observations. When only SSE considered, RMSE value is equal to 219.89. As a result of MTF model, which parameter tuning is applied, RMSE value decreases to 199.96. Since temporality is taken into account, the average time difference also decreases from 27.3 days to 23.7 days for train set and 26.2 days to 12.4 days for test set. There is no difference between

the two models in terms of seasonality. It can be said that MTF model gives more reliable results than TDT model for this dataset.

For *Weekly Seasonal Dataset*, the weight of DIST changes between 0 and 1, and TIME component has no effect on splitting. Figure 4.5 shows the resulting changes in performance metrics. For train set observations, RMSE value is not adversely affected if DIST objective is included in the model with a small ratio. However, RMSE value decreases with the increasing weight of DIST component for test set observations. Therefore, optimum value can be obtained with parameter tuning. Also, when the weight of DIST is increased, average seasonal distance in nodes decreases as expected. So, it can be said that data points that are similar to each other in terms of seasonality are located in the same nodes. Since TIME objective has no effect on splitting, average time difference in nodes increased over a year.

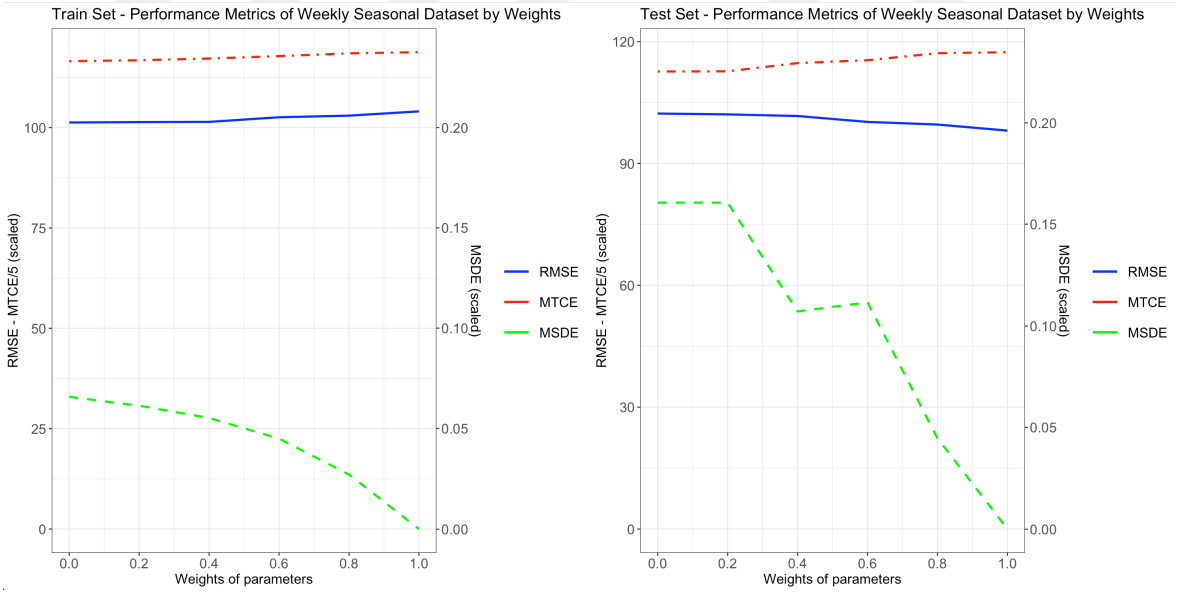

Figure 4.5. Performance metrics of weekly seasonal dataset by objective weights.

Parameter tuning is applied to compare TDT and MTF models. There is no significant difference between RMSE values of compared models for train set. When there is weekly seasonality in the dataset, RMSE value of the tree that considers only SSE, is 98.10 for test set. However, adding TIME and DIST objectives with applying

parameter tuning, RMSE value of MTF model slightly decreases to 97.85. It is observed that the mean time difference in temporal nodes is also considerably reduced for both train and test set observations. Thus, it can be said that the MTF model provides better predictions for this dataset.

For *Monthly Seasonal Dataset*, the weight of DIST changes between 0 and 1. Figure 4.6 shows the values of performance metrics with the change in DIST weight. Similar to previous dataset, average seasonal distance in nodes decreases when the weight of DIST is increased for both train and test sets. Thus, higher weight of DIST component increases the probability of seasonally similar points to be in the same terminal node. Also, RMSE value decreases and provides more accurate estimations for test set observations. Parameter tuning can be beneficial for obtaining more accurate predictions.

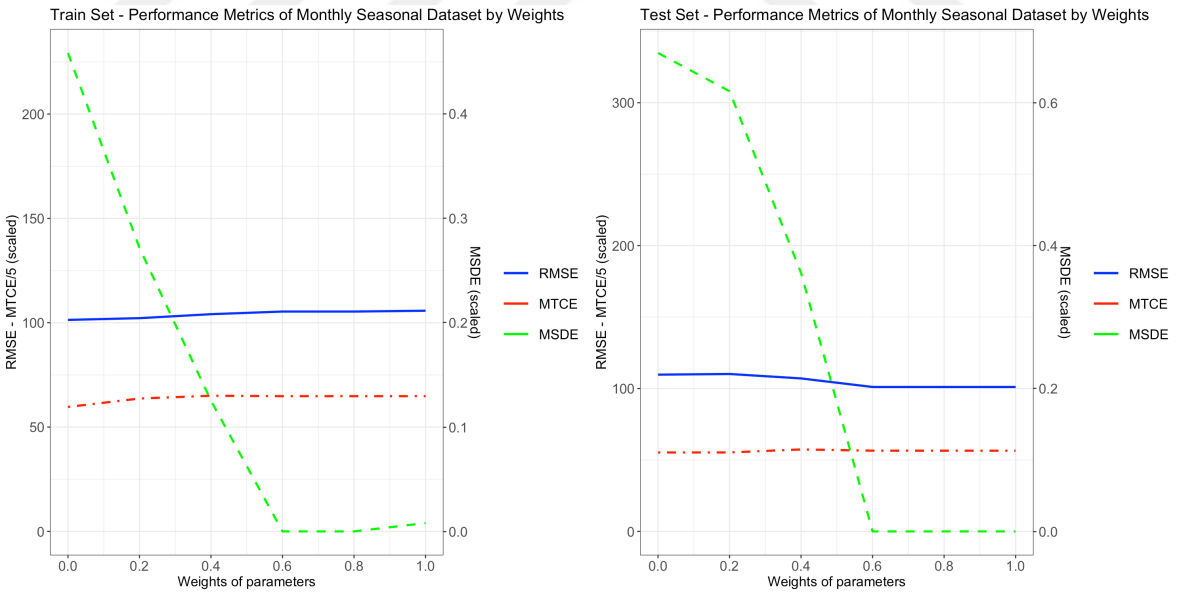

Figure 4.6. Performance metrics of monthly seasonal dataset by objective weights.

When there is monthly seasonality in the dataset, RMSE value of TDT model is 101.36 for train set and 109.70 for test set. After applying parameter tuning on TIME AND DIST objective, RMSE value of MTF model is 104.36 for train set and 104.79 for test set. Despite a slight increase in train set results, test set results are improved

considerably. In addition, observations in terminal nodes are closer to each other in time and similar points in terms of seasonality coexist with a higher rate. Thus, it can be concluded that MTF model provides more accurate predictions for this dataset.

The same analysis is performed for datasets that has seasonal relation and dependent to time index. For *Weekly Seasonal Temporal Dataset* and *Monthly Seasonal Temporal Dataset*, the weight of TIME and DIST changes between 0 and 1, while the weight of SSE is constant. The changes in performance metrics is represented with heatmaps. RMSE, MTCE and MSDE values is visualized in Figure 4.7 for train set and in Figure 4.8 for test set of *Weekly Seasonal Temporal Dataset*.

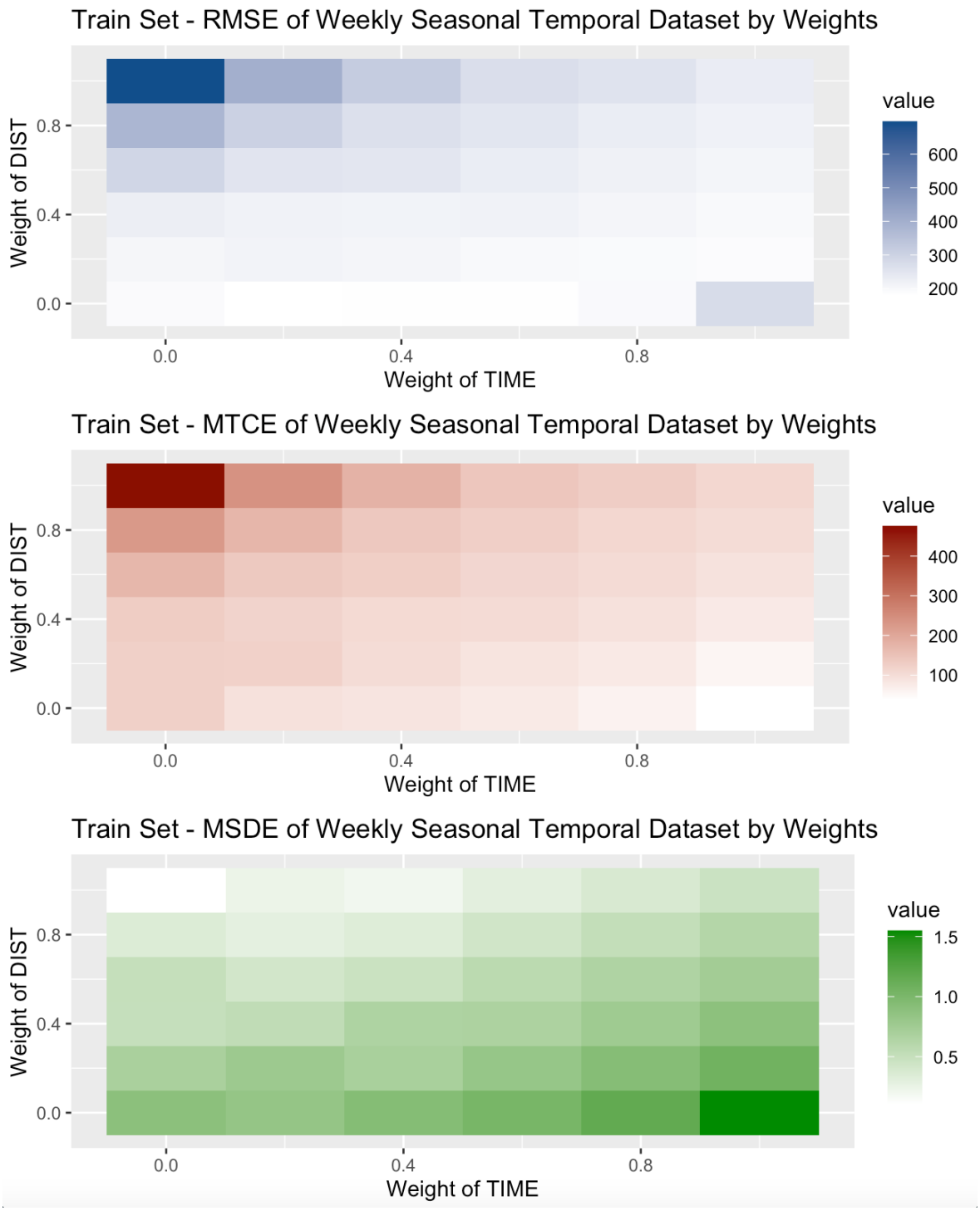

Figure 4.7. Performance metrics of weekly seasonal temporal dataset by objective weights for train set.

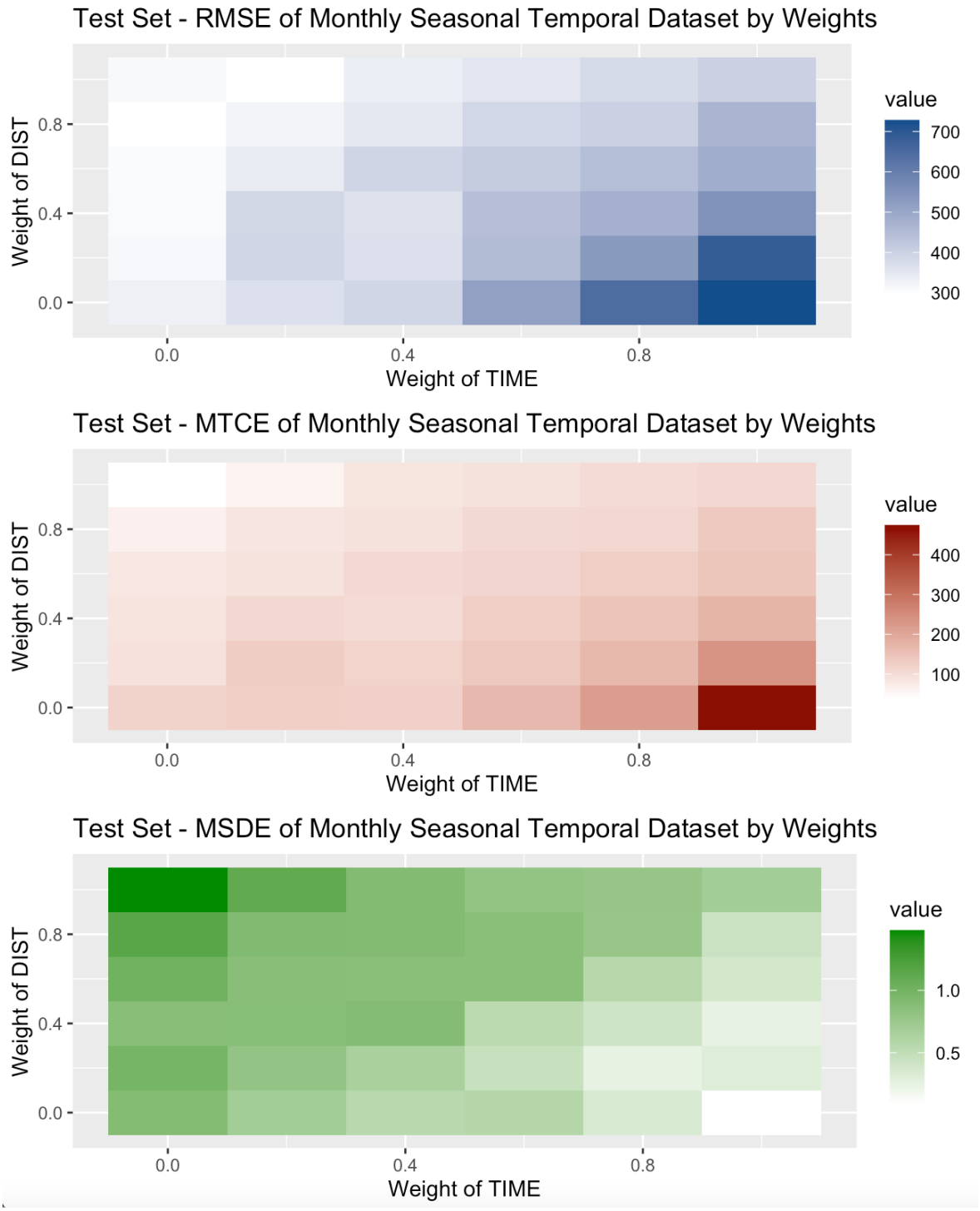

Figure 4.8. Performance metrics of weekly seasonal temporal dataset by objective weights for test set.

According to train set results of *Weekly Seasonal Temporal Dataset*, using many weights of TIME and DIST objectives are resulted in more accurate results than the TDT model. Especially, the best results are observed when these values are between 0.2 and 0.6. In addition, temporal and seasonal errors also decrease. As expected, when the weight of TIME is equal to 1, MTCE is minimized, while when the weight of DIST is equal to 1, MSDE is minimized.

When test set results in Figure 4.8 are examined, there are similar inferences to train set results. Taking into account the temporal closeness and seasonal distance metrics during splitting is also decreased RMSE values when their weight is small and resulted in more accurate predictions. Optimal objective weights that reduce both temporal and seasonal errors as well as RMSE value can be found.

For train set results of *Weekly Seasonal Temporal Dataset*, there is a slightly decrease in RMSE value with the use of MTF model. Train set RMSE value of TDT model is 428.15 and this value is 419.10 with MTF model. In addition, test set results in terms of RMSE is decreased. While RMSE value of TDT model's predictions is 331.93, it decreased to 291.84 with MTF model. It can be said that when the dataset is both weekly seasonal and dependent to time index, MTF model with parameter tuning can yield better results. Also, terminal nodes of decision trees can contain close data points together.

Finally, RMSE, MTCE and MSDE values is visualized in Figure 4.9 for train set and in Figure 4.10 for test set of *Monthly Seasonal Temporal Dataset*.

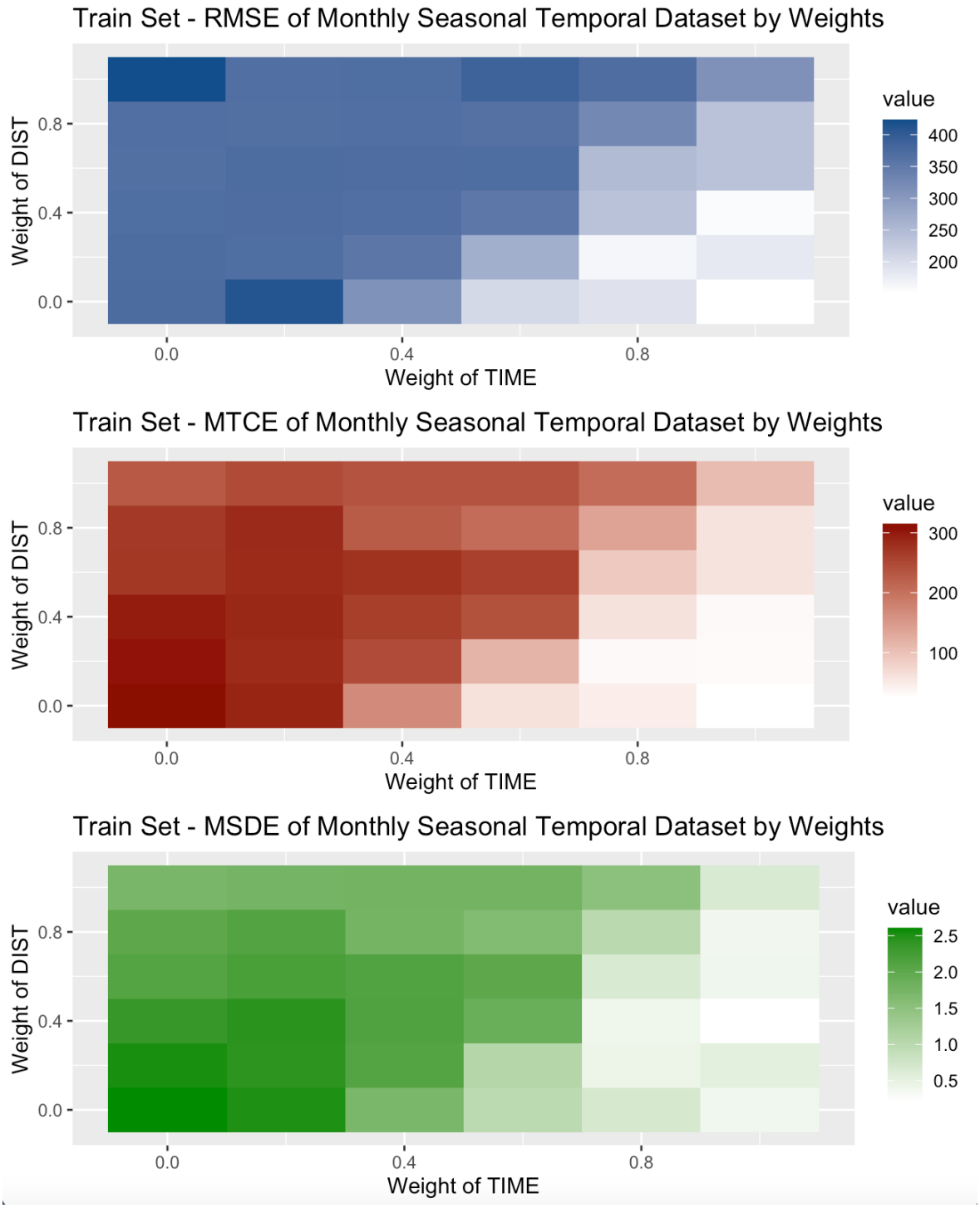

Figure 4.9. Performance metrics of monthly seasonal temporal dataset by objective weights for train set.

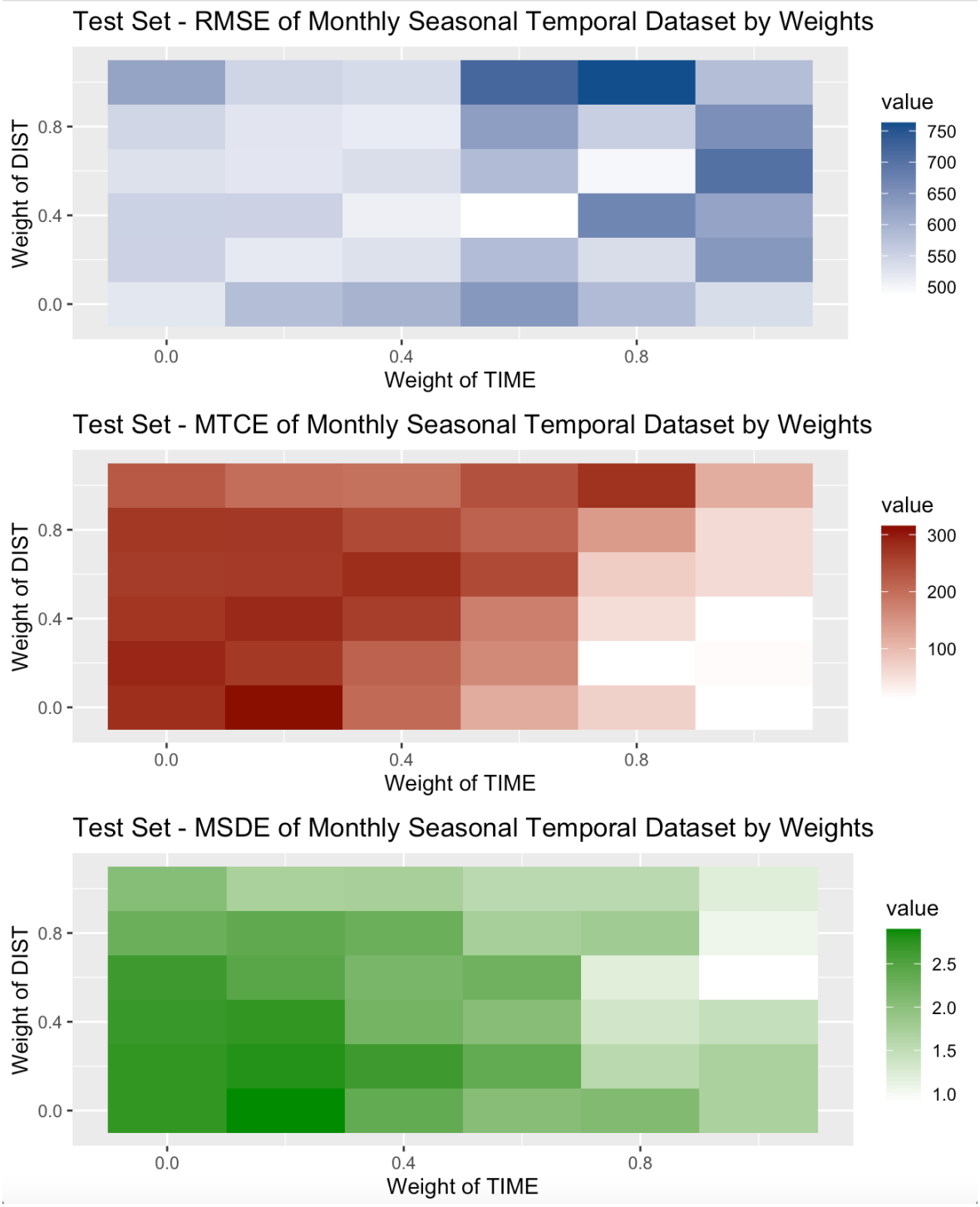

Figure 4.10. Performance metrics of monthly seasonal temporal dataset by objective weights for test set.

When train set results of *Monthly Seasonal Temporal Dataset* are investigated, smaller RMSE values are obtained with different TIME and DIST weight pairs. For example, the weight pairs (1.0, 0), (0.8, 0.2) or (1.0, 0.4) provides more accurate predictions than using only SSE in splitting process. In addition to more accurate results, average of temporal error and seasonal error is smaller than other options when the weight of DIST component is equal to 1.

When test set results in Figure 4.10 are examined, using TIME and DIST objectives with a smaller weight than 0.4 provides similar results to SSE model. However, for some weight pairs, smaller RMSE values than TDT model can be obtained with MTF model. These results indicate that it is possible to make more reliable predictions by considering both temporality and seasonality through parameter tuning.

Similar to the other synthetic datasets, there is a decrease in RMSE values of test set observations after parameter tuning. Even though RMSE value of train set is increased from 384.12 to 396.19, RMSE values of test set is decreased from 459.22 to 445.67. Also, it is observed that both temporal and seasonal errors have decreased. It can be said that weighting temporality and seasonality provides beneficial results in terms of prediction accuracy in this dataset.

All results mentioned in this section is summarized in Table 4.2.

Table 4.2. RMSE values of TDT and MTF models for synthetic datasets.

		Train Set			Test Set		
Dateset	Model	RMSE	MTDE	MSDE	RMSE	MTDE	MSDE
Temporal Dataset	TDT	128.06	27.29	1.68	219.89	26.17	1.63
Temporal Dataset	MTF	130.39	23.72	1.68	199.96	12.36	1.64
Weekly Seasonal Dataset	TDT	100.18	593.94	0.14	98.10	587.00	0.11
Weekly Seasonal Dataset	MTF	100.10	554.99	0.07	97.85	554.87	0.06
Monthly Seasonal Dataset	TDT	101.36	298.31	1.13	109.70	276.19	1.53
Monthly Seasonal Dataset	MTF	104.36	323.38	0.33	104.79	298.3	0.94
Weekly Seasonal Temporal Dataset	TDT	428.15	83.15	1.67	331.93	50.39	1.71
Weekly Seasonal Temporal Dataset	MTF	419.10	62.00	1.56	291.84	49.01	1.51
Monthly Seasonal Temporal Dataset	TDT	384.12	47.69	1.44	459.22	245.63	2.72
Monthly Seasonal Temporal Dataset	MTF	386.19	43.11	1.05	445.67	161.68	2.33

5. EXPERIMENTS

This section explains the prediction performance of postprocessing and multi-objective trees for time series forecasting. In the beginning, the datasets used are introduced. Then the implementation of the models is given in detail and effect of weights is analysed. At the end of the section, multiobjective tree method used in randomized setting.

5.1. Datasets

In addition to five different synthetic datasets, mentioned in Chapter 4, Google, electricity consumption and solar power datasets are used in the experiments. Summary information of these datasets can be seen in the Table 5.1.

Table 5.1. Summary information of datasets.

	Number of features	Number of observations	Granularity	Seasonality
Temporal	4	720	Daily	-
Weekly Seasonal	4	720	Daily	Weekly
Monthly Seasonal	4	720	Daily	Monthly
Weekly Seasonal Temporal	4	720	Daily	Weekly
Monthly Seasonal Temporal	4	720	Daily	Monthly
Google	4	995	Daily	Weekly
Electricity Consumption	12	35135	Hourly	Weekly
Solar Power	41	10896	Hourly	Hourly

5.1.1. Synthetic Datasets

The generation process of synthetic datasets is detailed in Section 4.3. All synthetic datasets contains 2-years of data and response variable, y, is dependent to four features. These features are date (time index), day of week and response values of the previous days (lag1 and lag2). The response values of these datasets are visualized in Figure 5.1.

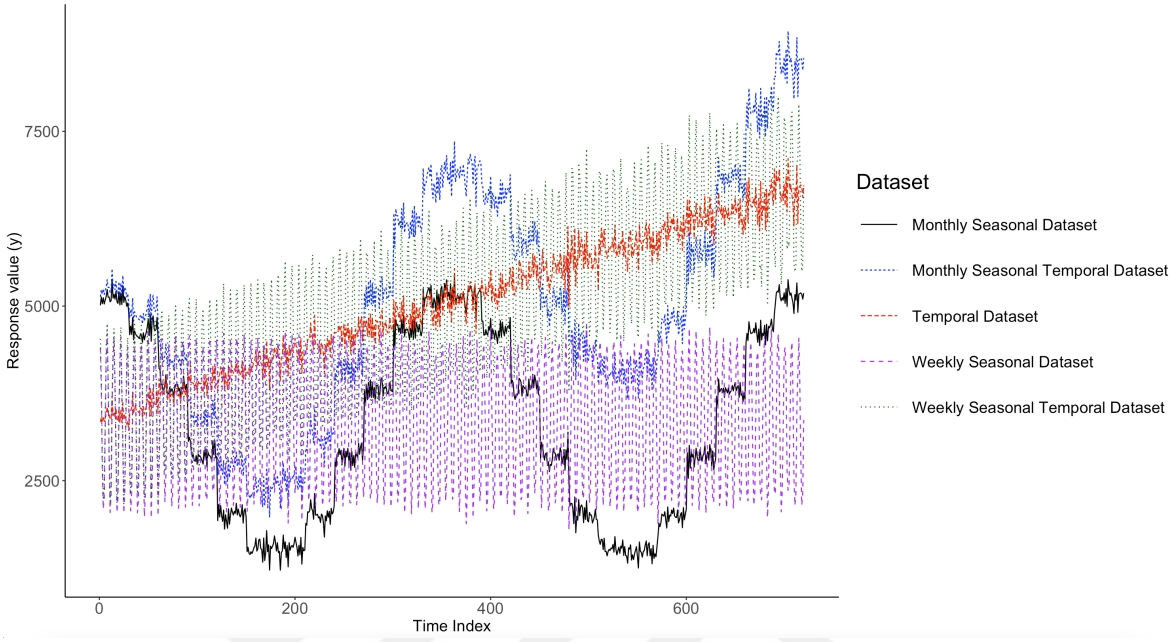

Figure 5.1. Synthetic datasets.

Weekly Seasonal Dataset and *Weekly Seasonal Temporal Dataset* are highly dependent on the days of the week. In order to observe seasonality, autocorrelation (ACF) plot is examined and it is given in Figure 5.2. When the ACF plot is analysed, significant lags are seen in every 7 time periods. This indicates that there is a 7-days seasonal cycle in the synthetic dataset.

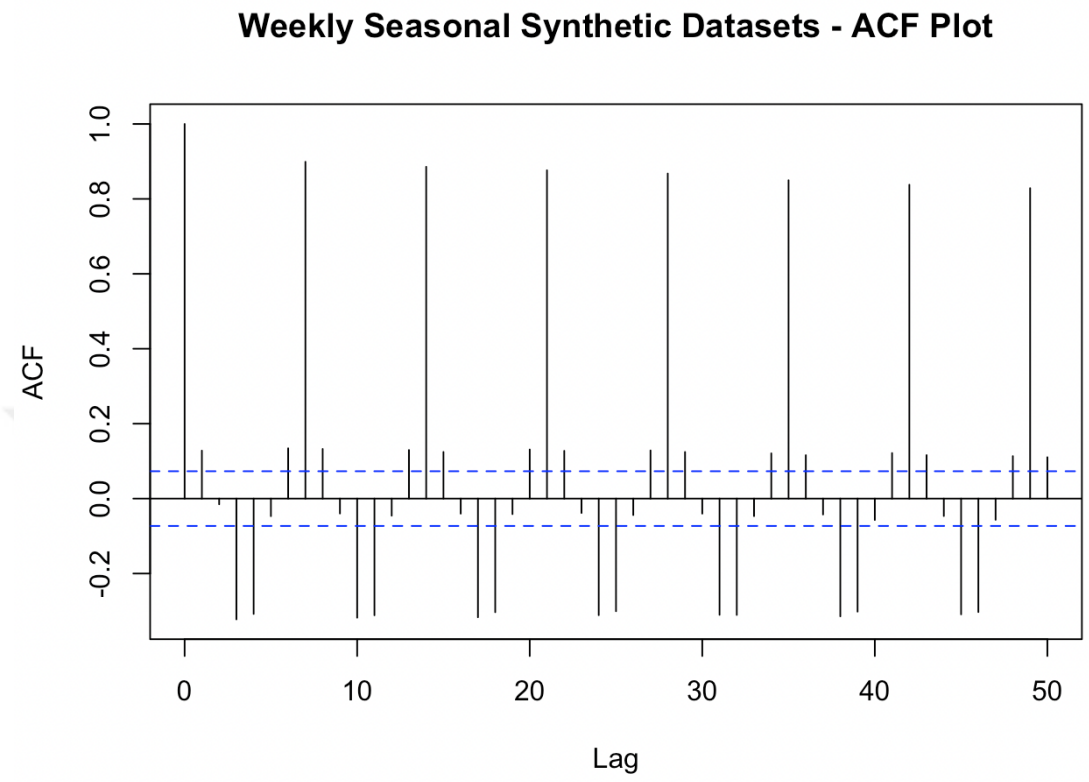

Figure 5.2. ACF plot of Weekly Seasonal synthetic datasets.

Monthly Seasonal Dataset and *Monthly Seasonal Temporal Dataset* are highly dependent on the month of the year. Autocorrelation plot is examined to observe seasonality and it is given in Figure 5.3. Autocorrelation is higher in the first lags and there is a seasonality according to month of the year.

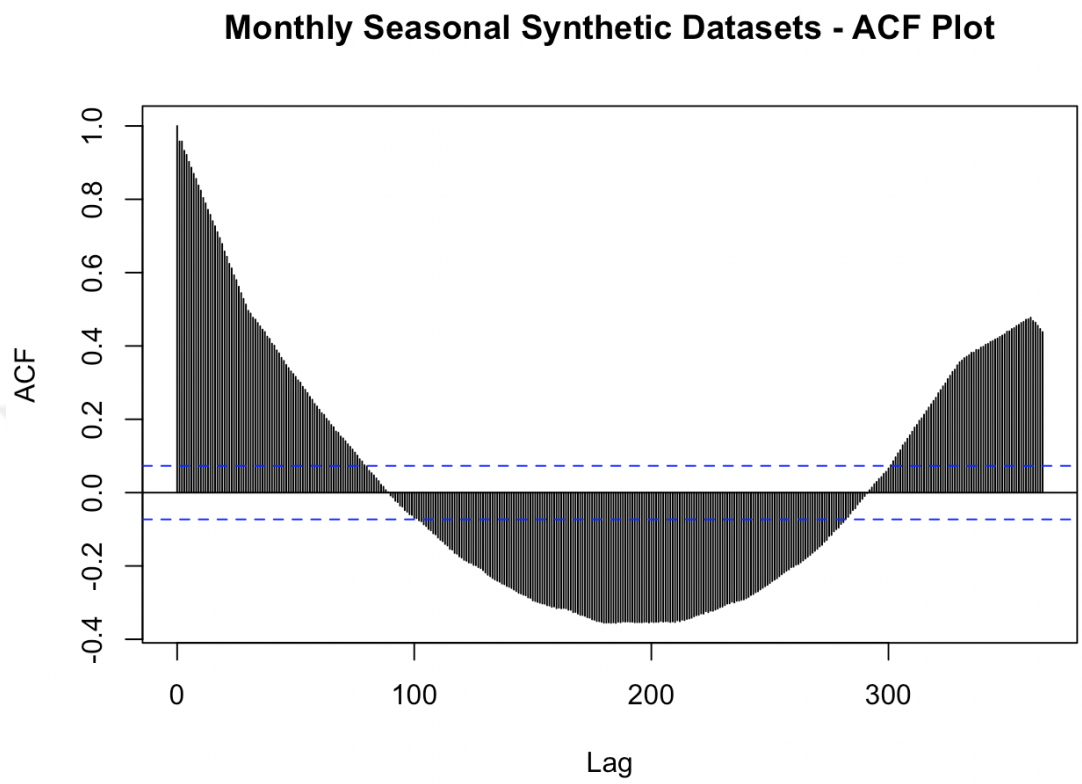

Figure 5.3. ACF plot of Monthly Seasonal synthetic datasets.

5.1.2. Google Dataset

Google dataset contains Google's daily closing stock prices for 1000 consecutive trading days between Feb 2013 and Feb 2017 and represented in Figure 5.4. Response value is explained with four variables and most of them is similar to the synthetic dataset. These variables are date (time index), day of week, lag1 and lag2.

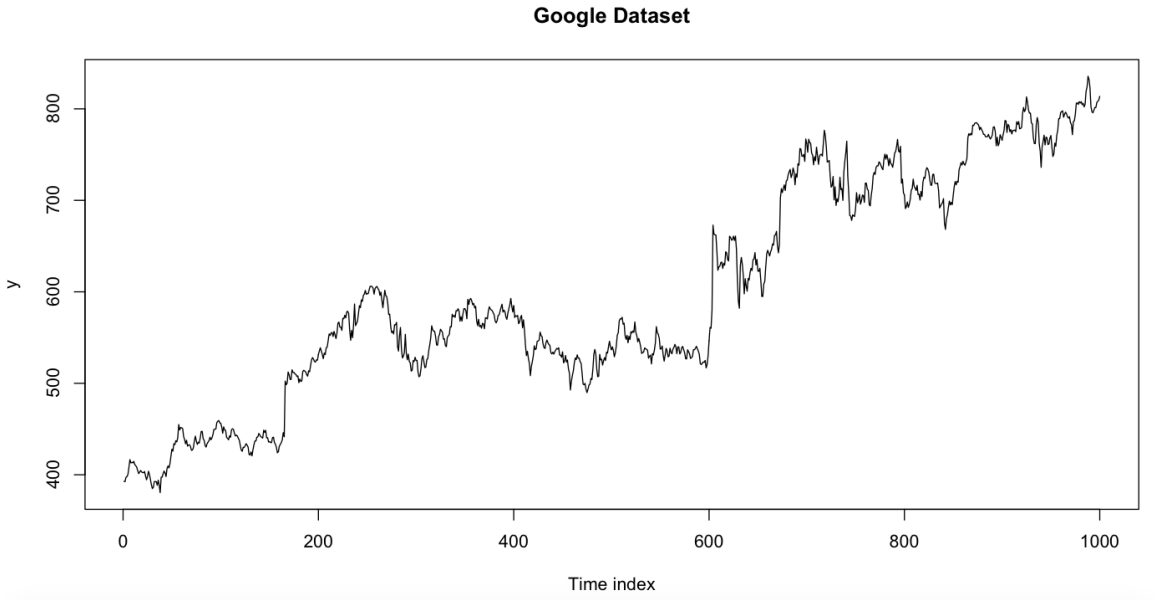

Figure 5.4. Google dataset.

Stock price values are highly dependent on the previous days by its nature. Especially the effect of the latest days is higher compared to older days. In Figure 5.5, autocorrelation plot of Google dataset is visualized. It is seen that the ACF value decreases geometrically as the lag value increases.

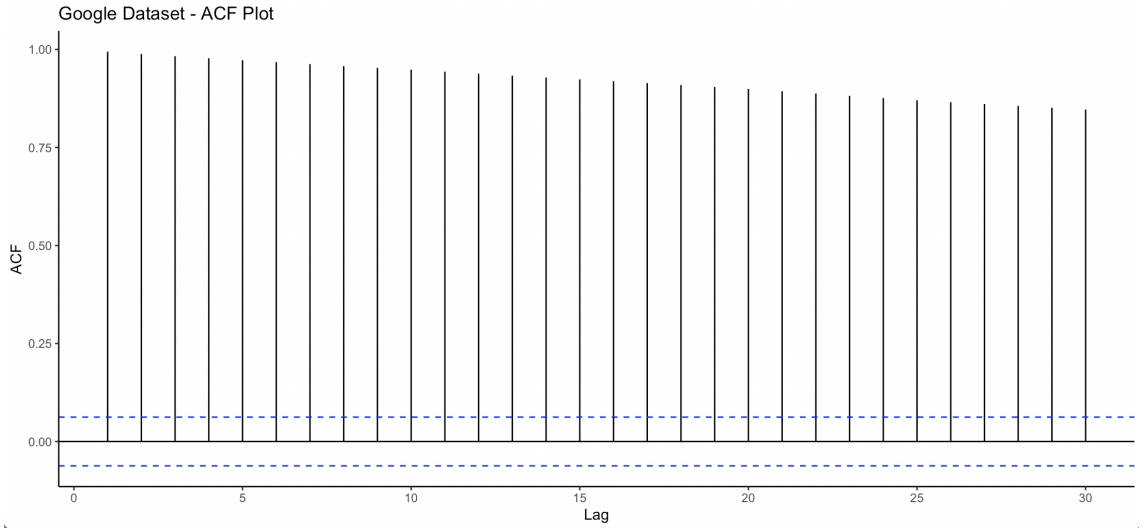

Figure 5.5. ACF plot of Google dataset.

5.1.3. Electricity Consumption Dataset

This dataset contains electricity consumption of Turkey between Jan 2017 and Jan 2021 and can be seen in Figure 5.6. Apart from the others, this dataset contains hourly information and consist of 35135 data points. It is a suitable example in terms of seasonality. In addition to date (time index), day of week and hour of day features, the temperature values in the regions are also used as variables. Moreover, holidays are added as two separate variables, religious holidays and national holidays. As a result, a total of 12 variables are used to explain the consumption value.

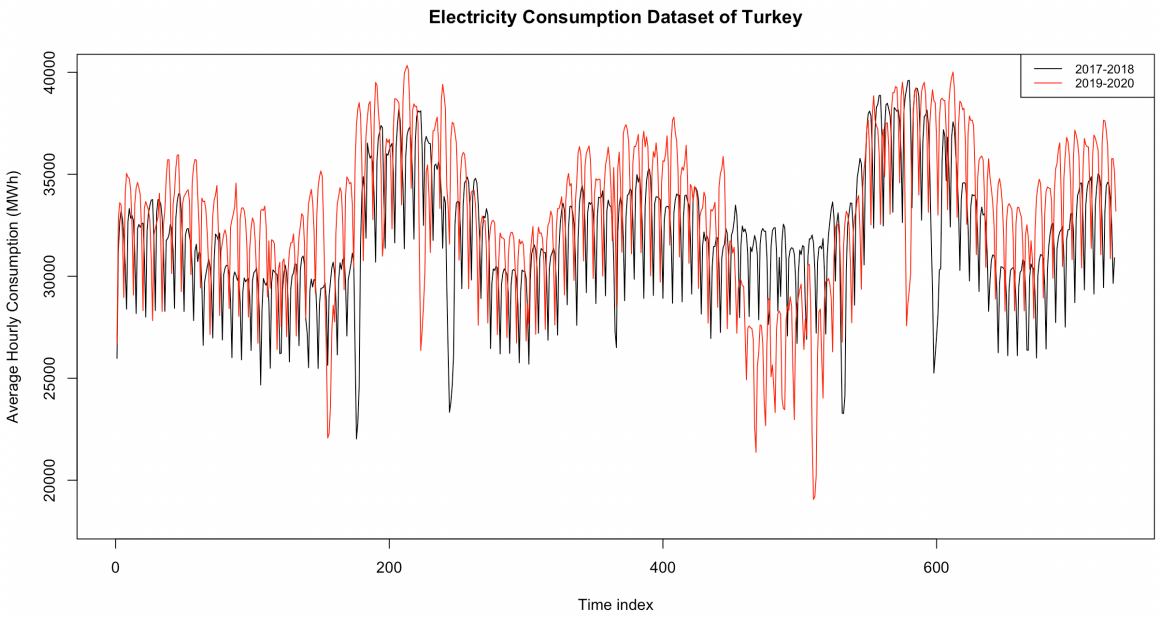

Figure 5.6. Electricity Consumption dataset of Turkey.

Electricity consumption values varies according to the day of the week. Therefore, Electricity Consumption dataset has weekly seasonality. Also, significant lags are seen every 7 days in the autocorrelation plot in Figure 5.7.

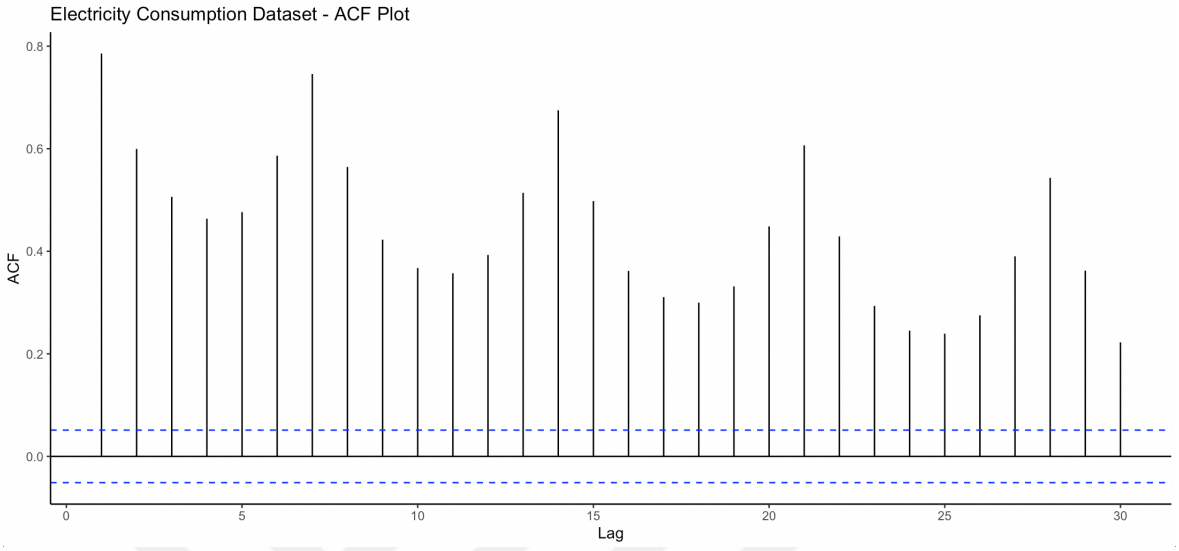

Figure 5.7. ACF plot of Electricity Consumption dataset.

5.1.4. Solar Power Dataset

Solar power dataset provides the solar power production of KIVANC 2 GES (solar energy power plant) between Feb 2021 and May 2022 and can be seen in Figure 5.8. This dataset consist of 10896 observations and similar to electricity consumption dataset, it contains hourly information. In addition to time series features such as date, day of week, hour of day features and holidays, it includes variables related to temperature, relative humidity, downward shortwave radiation flux (DSWRF) and cloud cover data in the given region. Total number of features that can explain the production value is equal to 41 features. Since high temperature values affects the efficiency of solar powers, temperature variable is an important feature that represents seasonality.

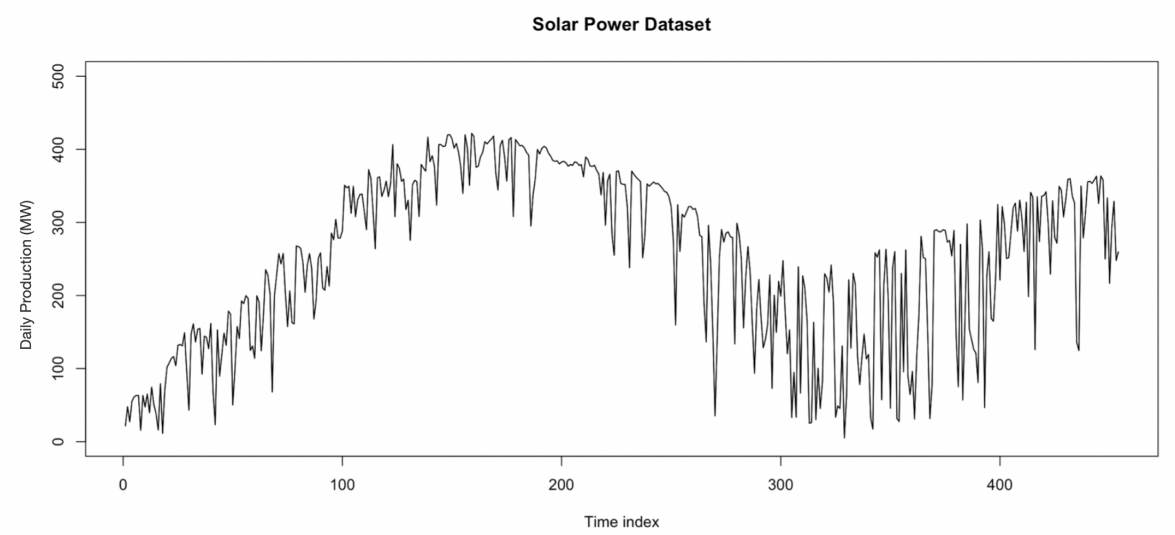

Figure 5.8. Solar Power dataset.

Solar power production varies according to the day of the year. For this dataset, predictable changes occur over a one-year period, so it can be said that Solar Power dataset has annual seasonality. The autocorrelation plot is given in Figure 5.9.

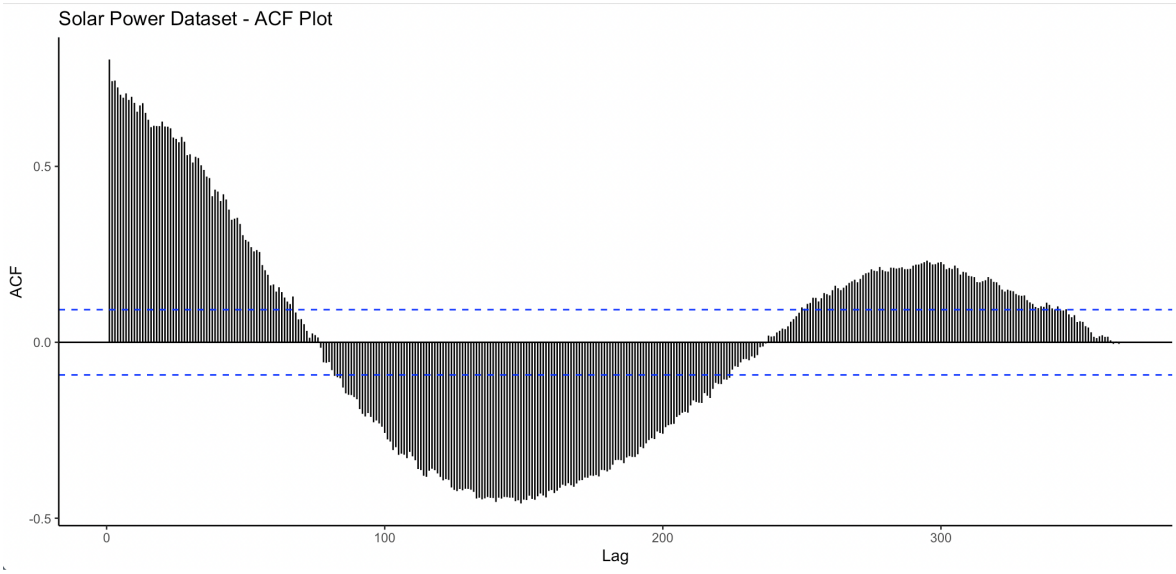

Figure 5.9. ACF plot of Solar Power dataset.

5.2. Implementation of Multiobjective Trees

Multiobjective Trees for Forecasting model is implemented with R programming language and the source code can be found in Github repository [35]. Experiments are done with R version 4.2.1 and the computer used has Apple M1 chip, 8 cores and 16 GB memory. In the implementation process of multiobjective trees, rolling approach is used to determine train and test sets. In the first set, 70% of the total number of observations are assigned as training data and testing period is determined as 1-month. Afterwards, the training data is increased as much as the testing period and a new test data was created by shifting the test set. In the experiments of this study, 8 different train and test set pairs are used. Representation of the rolling approach to determine train and test data can be seen in Figure 5.10. In this section, only one decision tree is fitted for each set of train and test data to understand the effect of objective weights. In the random forest setting of multiobjective trees, 50 decision trees are generated. The minimum number of observations required in a node to be able to split is taken as 20 in each tree. The depth of decision trees is unrestricted.

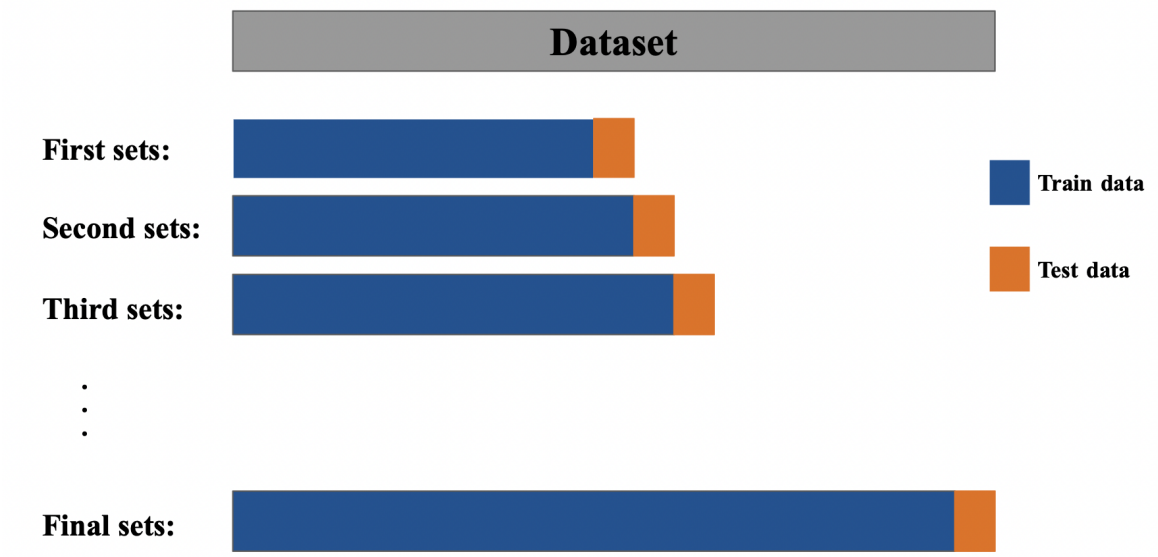

Figure 5.10. Rolling approach to determine train and test data.

Weights of the objectives used in splitting process are crucial. In order to use optimum weights, parameter tuning is applied for all train and test set pairs. While the weight of SSE is equal to 1, TIME and DIST parameters differ between 0 and 1. For all weight settings, prediction errors are analysed and the weights that provides the most accurate prediction results are selected. The traditional decision tree calculation where the weight of SSE is equal to 1 and other objectives are not included in the model, is used as the base model. Then, the results of MTF model, after applying a parameter tuning, is compared with base model.

Firstly, in order to understand the change in results by the properties of datasets, RMSE values of synthetic datasets are examined. The detailed analysis is given in Chapter 3 and RMSE values can be seen in Figure 5.11.

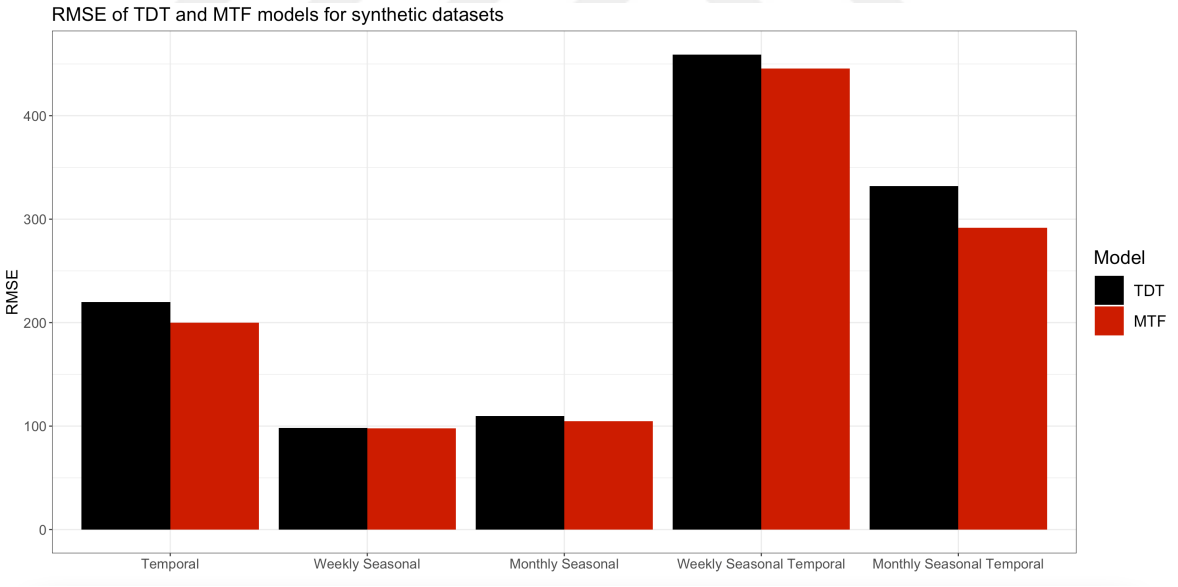

Figure 5.11. Test set RMSE results of TDT and MTF models for all synthetic datasets.

For *Temporal Dataset*, which response value is highly effected by the time index, better predictions are obtained with MTF model. RMSE value is decreased to 199.97 from 219.89 and it can be said that results are improved by 9%. For the synthetic datasets with seasonality, improvement rate in test set RMSE values are smaller than *Temporal Dataset*. RMSE value decreased from 98.10 to 97.85 for *Weekly Seasonal Dataset* and from 109.70 to 104.79 for *Monthly Seasonal Dataset*. However, higher improvement rate is observed when both the response value increased depending on time and the dataset included seasonality. There is a 12% decrease in the RMSE value from 331.93 to 291.84 for *Weekly Seasonal Temporal Dataset* and 3% decrease from 459.22 to 445.67 for *Monthly Seasonal Temporal Dataset*.

Test set RMSE results for Google, Electricity Consumption and Solar Power datasets is represented in Figure 5.12, Figure 5.13 and Figure 5.14.

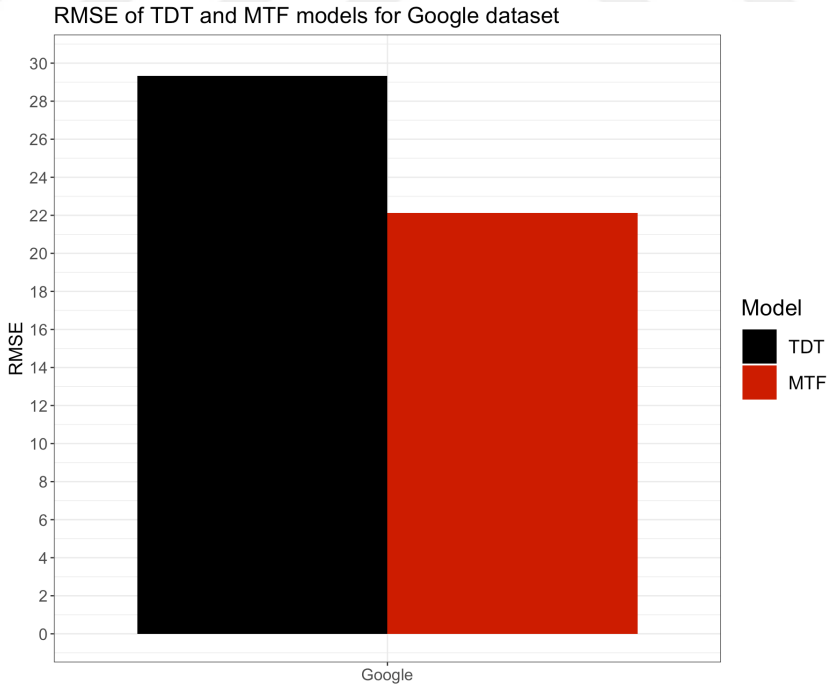

Figure 5.12. Test set RMSE results of TDT and MTF models for Google dataset.

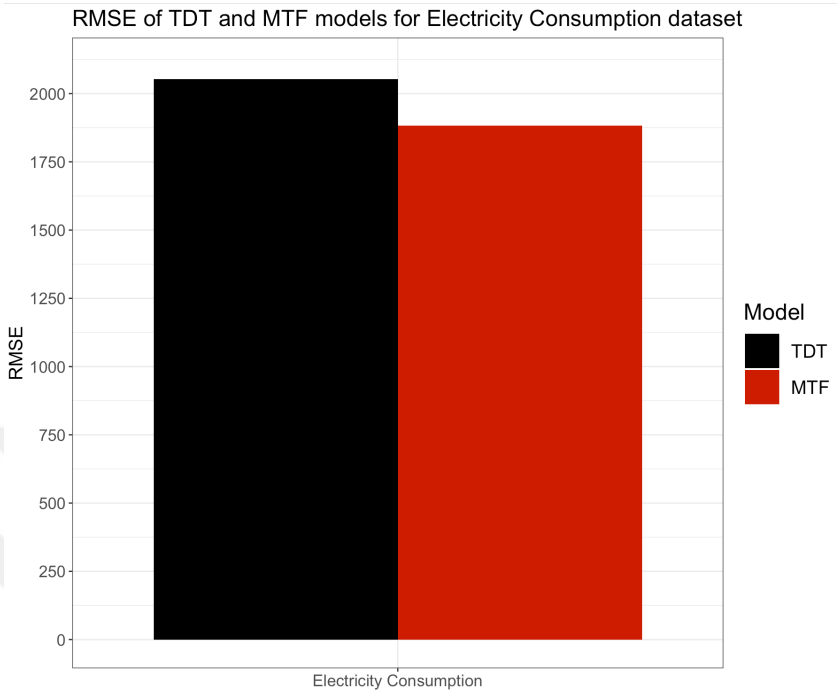

Figure 5.13. Test set RMSE results of TDT and MTF models for Electricity Consumption dataset.

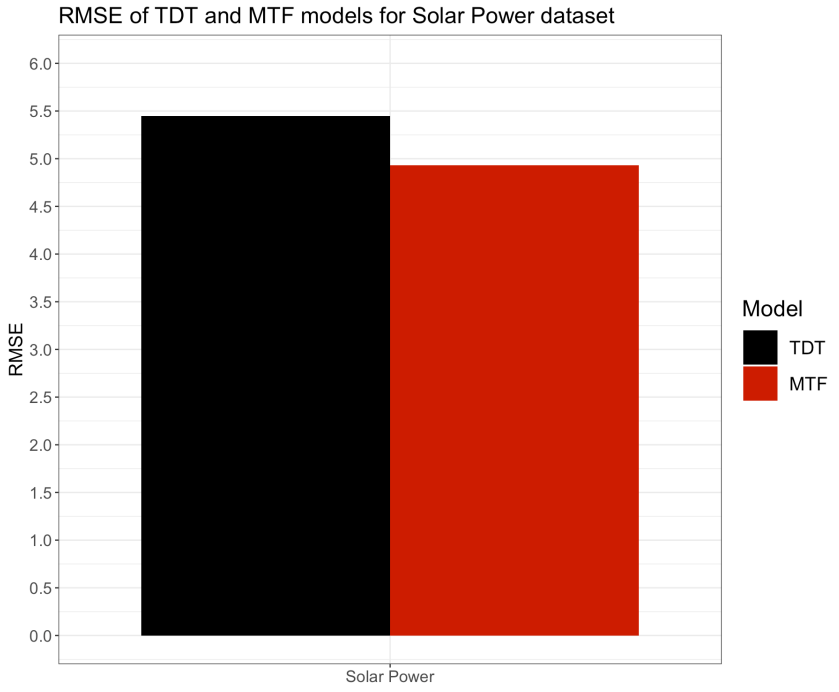

Figure 5.14. Test set RMSE results of TDT and MTF models for Solar Power dataset.

When the results of datasets with different properties are examined, the highest difference is observed in Google dataset. RMSE value of test set decreased from 29.32 to 22.12. With a 25% decrease in the RMSE value, it can be said that the predictions made with MTF model are more accurate than TDT model. When MTF model is used for the test set observations of Electricity Consumption and Solar Power datasets, a decrease of approximately 10% compared to the base model is observed in RMSE values. RMSE value for the Electricity Consumption dataset decreased from 2053.14 to 1882.66, and for the Solar Power dataset, it decreased from 5.45 to 4.93. Based on these results, it can be said that MTF model provides better predictions than the base model, which only considers SSE minimization. In addition, decision tree experiments can be repeated in a randomized setting to improve the results.

5.3. Multiobjective Trees in Randomized Setting

In the randomized setting of multiobjective trees, rolling approach is used to determine train and test sets. The number of trees fitted for each train and test set pairs is equal to 50. Similar to random forest model, random samples with replacement are taken from the train dataset for each multiobjective tree model. Also, random weights in a specific range are assigned to model objectives for each tree. Similar to previous section, hyperparameter tuning is applied for weight selection. At first, parameter tuning is done for each train and test set pairs, then, random samples are created.

The results of synthetic datasets are analysed to make an inference according to the properties of dataset. RMSE values is visualized in Figure 5.15.

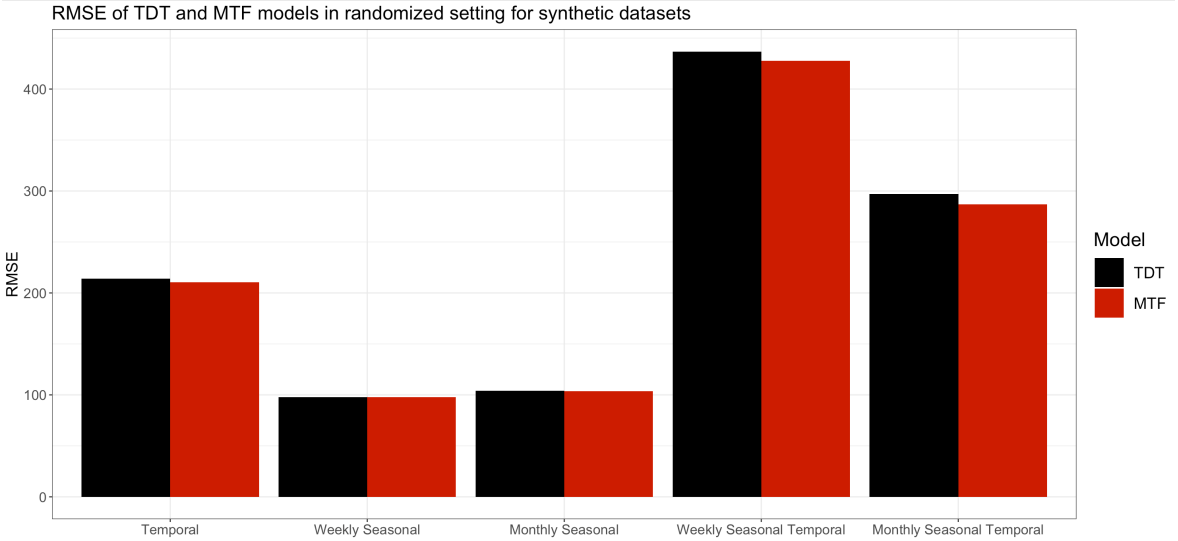

Figure 5.15. Test set RMSE results of TDT and MTF models in randomized setting for all synthetic datasets.

When experiments are conducted with decision trees in a randomized setting, there is a slight improvement in test set RMSE values if the dataset has temporal relationships or contains seasonality. However, the improvement rate is higher if the dataset has both temporal relationships or contains seasonality. On the other hand, the improvement rate is significantly lower than that obtained with a single decision tree. Therefore, it can be said that the use of MTF model in a randomized setting does not create a significant improvement compared to a single decision tree.

Test set RMSE results in randomized setting for Google, Electricity Consumption and Solar Power datasets is represented in Figure 5.16, Figure 5.17 and Figure 5.18.

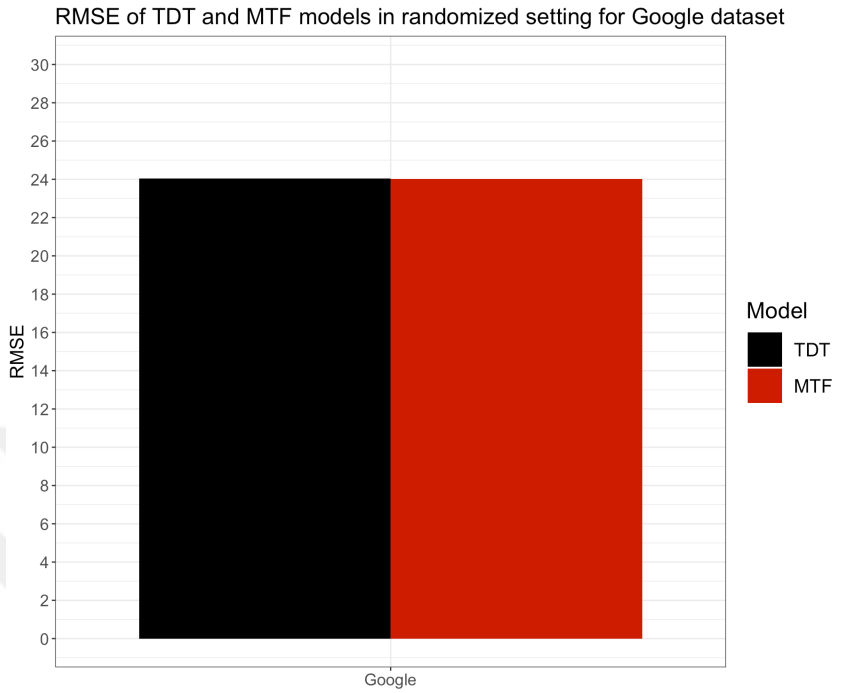

Figure 5.16. Test set RMSE results of TDT and MTF models in randomized setting for Google dataset.

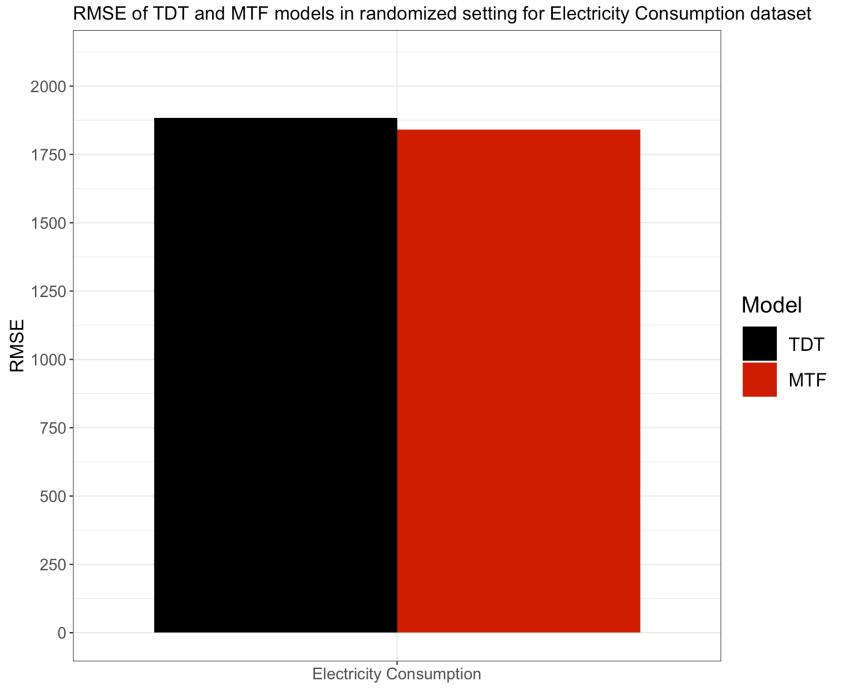

Figure 5.17. Test set RMSE results of TDT and MTF models in randomized setting for Electricity Consumption dataset.

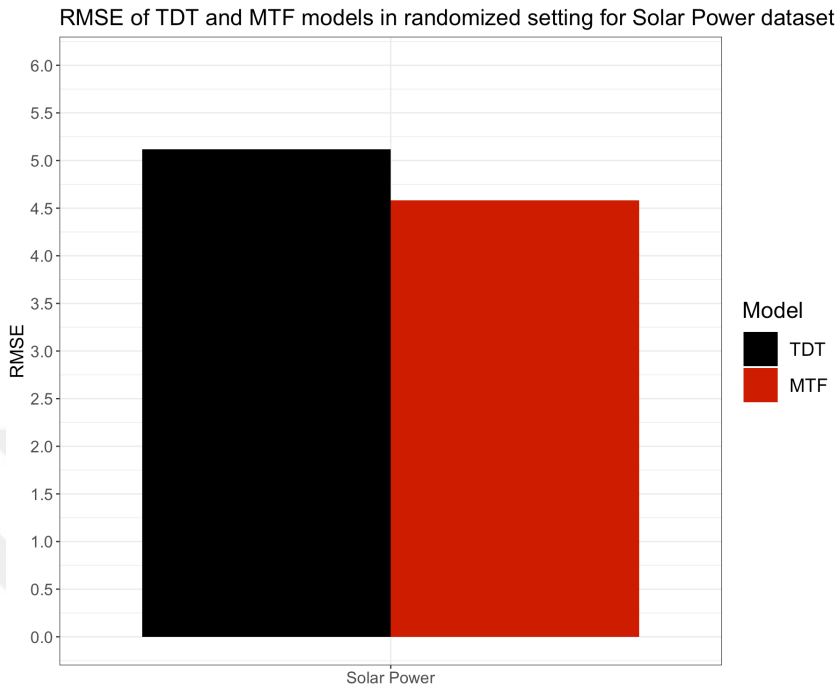

Figure 5.18. Test set RMSE results of TDT and MTF models in randomized setting for Solar Power dataset.

In the random forest setting, there is not as much improvement observed compared to experiments conducted with a single decision tree. However, the highest improvement in RMSE values is observed in Solar Power dataset. For Google dataset, RMSE value of the random forest version of the trees that aimed only for SSE minimization during the splitting process is 24.05. In the same setting, RMSE value obtained with the MTF model is 24.01. This indicates that although there is a slight improvement in predictions, it is not significant. For Electricity consumption dataset, the trees using the SSE model have RMSE value of 1884.11 for test set observations, while this value is reduced to 1841.52 with MTF model, indicating only a 3% improvement. However, for Solar Power dataset, there is a significant decrease in RMSE values. RMSE value of the SSE model has decreased from 5.12 to 4.58, resulting in an 11% improvement.

5.4. Results of Models

RMSE results of all models for each dataset is given in Table 5.2. This table provides a comparison for the performance of time series analysis models, decision tree, random forest and MTF models on different datasets. MODEL-LAG1, MODEL-LAG2 and MODEL-LAG7 models use lagged values of the observation as prediction. As time series analysis models, ARIMA and SARIMA models are used. TDT-DT and MTF-DT models are the results of the implementation of traditional decision tree and multiobjective tree models. Finally, TDT-RF and MDF-RF models are their implementation in randomized setting.

Table 5.2. RMSE Results of all models and datasets.

Dataset	MODEL-LAG1	MODEL-LAG2	MODEL-LAG7	ARIMA	SARIMA	TDT-DT	MTF-DT	TDT-RF	MTF-RF
Temporal	283.46	229.03	281.31	217.03	217.03	219.89	199.96	213.81	210.32
Weekly Seasonal	763.33	1354.73	144.43	295.90	113.22	98.10	97.85	97.94	97.85
Monthly Seasonal	160.55	144.39	222.07	128.40	124.01	109.70	104.79	104.20	103.63
Weekly Seasonal Temporal	798.22	1365.14	295.37	374.07	215.93	331.93	291.84	297.15	286.98
Monthly Seasonal Temporal	693.73	579.99	776.61	498.82	491.92	459.22	445.67	436.79	427.52
Google	9.44	13.46	23.46	9.46	9.46	29.32	22.12	24.05	24.01
Electricity Consumption	2925.96	3848.23	2707.43	4830.16	4637.71	2053.14	1882.66	1884.11	1841.52
Solar Power	6.46	7.04	7.18	21.09	18.36	5.45	4.93	5.12	4.58

For synthetic datasets, MTF-DT and MTF-RF models provides more accurate predictions with the smallest RMSE values. Also, the results show that MTF-DT and MTF-RF models outperforms all other models on Electricity Consumption and Solar Power datasets. These datasets are characterized by an increasing mean response over time and weekly or hourly relationships among observations. In addition, they contain variables such as temperature and relative humidity that provide information about the response. Different from the other datasets, ARIMA and SARIMA models perform well on datasets with high autocorrelation, such as Google Dataset. Overall, MTF-DT and MTF-RF models are promising approaches for time series forecasting, especially for datasets with seasonal and temporal relationships.

6. CONCLUSION

Multiobjective Trees for Forecasting (MTF) model is a tree-based ensemble model. MTF model changes the learning algorithm of decision trees to capture temporal relationships and seasonality during the learning process. There are three criteria that are taken into account in each partition. The first objective is decreasing the sum of squares error (SSE) as in the traditional decision tree (TDT) algorithm. In order to capture the temporal relationship, temporal closeness in each node is calculated. Thus, nodes that contains observations close to each other, gain importance during learning. Also, seasonality is represented with sine and cosine components and minimizing seasonal distance between the observations is added as a new objective.

Rolling approach is used to create train and test sets. Then, parameter tuning is applied to determine the weights of the objectives in the MTF model. The results are tested using both decision tree and random forest settings. When only one decision tree is fitted, it is observed that MTF model led to significantly better predictions than the traditional decision tree approach. By tuning the parameters, objective weights that reduce both the temporal and seasonal errors and improve prediction accuracy by reducing RMSE value can be found. However, when multiobjective trees are observed in the randomized setting, they provide smaller improvement rate compared to using a single decision tree. Therefore, there is no significant improvement in terms of prediction accuracy compared to the classic random forest model.

In addition to the MTF model, using the response value of the previous week as a prediction value also yield positive results in synthetic datasets with weekly seasonality. It can be said that MTF model performs better predictions than TDT model by taking into account the seasonal relationship. Similarly, ARIMA and SARIMA models produce accurate predictions in datasets with high autocorrelation. Especially in Electricity Consumption and Solar Power datasets, MTF model performs better than all other models in terms of accuracy. The common features of these datasets are the increase in mean response over time and the relationships between observations

at weekly or hourly level. In addition, unlike other datasets, they contain additional variables that can provide information about the response variable such as temperature and relative humidity. In other datasets, only day of the week, time index, and lag variables are available.

In summary, MTF model presents a significant advancement in the field of time series forecasting, as it effectively combines the strengths of decision tree learning with the ability to capture both temporal relationships and seasonality. MTF model is promising for improving the accuracy by capturing temporality and seasonality. Also, it can be used for a wide range of applications and industries since it performs good results with high dimensional datasets. Future research may focus on investigating the conditions under which the random forest version of the MTF model performs best.

REFERENCES

1. Tanizaki, T., T. Hoshino, T. Shimmura and T. Takenaka, "Demand Forecasting in Restaurants Using Machine Learning and Statistical Analysis", *12th CIRP Conference on Intelligent Computation in Manufacturing Engineering*, Vol. 79, pp. 679–683, Gulf of Naples, Italy, 2019.

2. Vijh, M., D. Chandola, V. A. Tikkiwal and A. Kumar, "Stock Closing Price Prediction Using Machine Learning Techniques", *International Conference on Computational Intelligence and Data Science*, Vol. 167, pp. 599–606, Gurgaon, India, 2020.

3. Kohli, S., G. T. Godwin and S. Urolagin, *Advances in Machine Learning and Computational Intelligence*, chap. Sales Prediction Using Linear and KNN Regression, pp. 321–329, Springer, Singapore, 2021.

4. Box, G. E., G. Jenkins, G. C. Reinsel and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, John Wiley & Sons, Inc., Hoboken, New Jersey, 2015.

5. Breiman, L., "Random Forests", *Machine Learning*, Vol. 45, pp. 5–32, 2020.

6. Cutler, A., D. R. Cutler and J. R. Stevens, *High-Dimensional Data Analysis in Cancer Research*, chap. Tree-Based Methods, pp. 1–19, Springer, New York, NY, 2009.

7. Alon, I., M. Qi and R. Sadowski, "Forecasting Aggregate Retail Sales: A Comparison of Artifcial Neural Networks and Traditional Methods", *Journal of Retailing and Consumer Services*, Vol. 8, pp. 147–156, 2001.

8. Baboo, D. S. S. and I. K. Shereef, "An Efficient Weather Forecasting System Using Artificial Neural Network", *International Journal of Environmental Science and Development*, Vol. 1, pp. 321–326, 2010.

9. Khashei, M. and M. Bijari, "Which Methodology is Better for Combining Linear and Nonlinear Models for Time Series Forecasting?", *Journal of Industrial and Systems Engineering*, Vol. 4, pp. 265–285, 2011.

10. Taskaya Temizel, T. and M. Casey, "A Comparative Study of Autoregressive Neural Network Hybrids", *Neural Networks*, Vol. 18, pp. 781–789, 2005.

11. Niu, D., Y. Wang and D. D. Wu, "Power Load Forecasting Using Support Vector Machine and Ant Colony Optimization", *Expert Systems with Applications*, Vol. 37, No. 3, pp. 2531–2539, 2010.

12. Yu, H.-F., N. Rao and I. S. Dhillon, "Temporal Regularized Matrix Factorization for High-Dimensional Time Series Prediction", *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems*, pp. 847–855, Barcelona, Spain, 2016.

13. Januschowski, T., Y. Wang, K. Torkkola, T. Erkkilä, H. Hasson and J. Gasthaus, "Forecasting With Trees", *International Journal of Forecasting*, Vol. 38, No. 4, pp. 1473–1481, 2022.

14. Sel, B., *Forecasting With A Large Number Of Predictors: A Tree Based Moving Average Approach*, Master's Thesis, Boğaziçi University, 2020.

15. James, G., D. Witten, T. Hastie and R. Tibshirani, *An Introduction to Statistical Learning With Applications in R*, chap. Tree-Based Methods, pp. 301–335, Springer, New York, NY, 2021.

16. Papadopoulos, S., E. Azar, W.-L. Woon and C. E. Kontokosta, "Evaluation of Tree-Based Ensemble Learning Algorithms for Building Energy Performance Estimation", *Journal of Building Performance Simulation*, Vol. 11, No. 3, pp. 322–332, 2018.

17. Polikar, R., *Ensemble Machine Learning: Methods and Applications*, chap. Ensem-

ble Learning, pp. 1–34, Springer, New York, NY, 2012.

18. Breiman, L., "Bagging Predictors", *Machine Learning*, Vol. 24, pp. 123–140, 2004.

19. Hengl, T., M. Nussbaum, M. Wright, G. Heuvelink and B. Graeler, "Random Forest as a Generic Framework for Predictive Modeling of Spatial and Spatio-Temporal Variables", *PeerJ*, Vol. 6, p. e5518, 2018.

20. Loh, W.-Y., "Classification and Regression Trees", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 1, pp. 14–23, 2011.

21. Spiliotis, E., "Decision Trees for Time-Series Forecasting", *Foresight: The International Journal of Applied Forecasting*, , No. 64, pp. 30–44, 2022.

22. "M5 Accuracy Competition: Results, Findings, and Conclusions", *International Journal of Forecasting*, Vol. 38, No. 4, pp. 1346–1364, 2022.

23. Godahewa, R., G. Webb, D. Schmidt and C. Bergmeir, "SETAR-Tree: A Novel and Accurate Tree Algorithm for Global Time Series Forecasting", *Machine Learning*, pp. 1–37, 2023.

24. Weng, T., W. Liu and J. Xiao, "Supply Chain Sales Forecasting Based on LightGBM and LSTM Combination Model", *Industrial Management Data Systems*, 2019.

25. Ju, Y., G. Sun, Q. Chen, M. Zhang, H. Zhu and M. ur Rehman, "A Model Combining Convolutional Neural Network and LightGBM Algorithm for Ultra-Short-Term Wind Power Forecasting", *IEEE Access*, Vol. 7, pp. 28309–28318, 2019.

26. Sun, X., M. Liu and Z. Sima, "A Novel Cryptocurrency Price Trend Forecasting Model Based on LightGBM", *Finance Research Letters*, Vol. 32, 2018.

27. Tüysüzoğlu, G., D. Birant and V. Kıranoğlu, "Temporal Bagging: A New Method for Time-Based Ensemble Learning", *Turkish Journal of Electrical Engineering and*

Computer Sciences, Vol. 30, pp. 279–294, 2022.

28. Ooi, T. S., S. Y. Ong and W. Cheah, "Temporal Sampling Forest (TS-FTS-F): An Ensemble Temporal Learner", *Soft Computing*, Vol. 21, pp. 7039–7052, 2017.

29. Agarwal, S. and C. R. Chowdary, "A-Stacking and A-Bagging: Adaptive Versions of Ensemble Learning Algorithms for Spoof Fingerprint Detection", *Expert Systems with Applications*, Vol. 146, p. 113160, 2020.

30. Li, L., S. Bagheri, H. Goote, A. Hasan and G. Hazard, "Risk Adjustment of Patient Expenditures: A Big Data Analytics Approach", *IEEE International Conference on Big Data*, pp. 12–14, Silicon Valley, CA, USA, 2013.

31. Galicia, A., R. Talavera-Llames, A. Troncoso, I. Koprinska and F. Martínez-Álvarez, "Multi-step Forecasting for Big Data Time Series Based on Ensemble Learning", *Knowledge-Based Systems*, Vol. 163, pp. 830–841, 2019.

32. Ilic, I., B. Görgülü, M. Cevik and M. G. Baydoğan, "Explainable Boosted Linear Regression for Time Series Forecasting", *Pattern Recognition*, Vol. 120, p. 108144, 2021.

33. Haimes, Y., *Risk Modeling, Assessment, and Management*, chap. Multiobjective Decision-Tree Analysis, pp. 375–427, John Wiley & Sons, Inc., Hoboken, New Jersey, 2008.

34. Adıyeke, E. and M. G. Baydoğan, "The Benefits of Target Relations: A Comparison of Multitask Extensions and Classifier Chains", *Pattern Recognition*, Vol. 107, p. 107507, 2020.

35. Arica, I., "Multiobjective Trees for Forecasting Model Github Repository", `https://github.com/iremarica/mtf-model`, 2023.