ISTANBUL TECHNICAL UNIVERSITY ★ FACULTY OF MANAGEMENT

INTRADAY ELECTRICITY MARKET PRICE FORECAST

B.Sc. THESIS

Berk KAPUCU
Ömer Seyfeddin KOÇ
Hakan GÜNAY

Department of Industrial Engineering

JUNE 2021

ISTANBUL TECHNICAL UNIVERSITY ★ FACULTY OF MANAGEMENT

INTRADAY ELECTRICITY MARKET PRICE FORECAST

B.Sc. THESIS

Berk KAPUCU
(070160022)
Ömer Seyfeddin KOÇ
(070160029)
Hakan GÜNAY
(070160069)

Department of Industrial Engineering

Thesis Advisor: Res. Asst. PhD Hüseyin Kutay Tinç

JUNE 2021

ISTANBUL TEKNİK ÜNİVERSİTESİ ★ İŞLETME FAKÜLTESİ

GÜN İÇİ ENERJİ PİYASALARINDA FİYAT TAHMİNİ

LİSANS TEZİ

Berk KAPUCU
(070160022)
Ömer Seyfeddin KOÇ
(070160029)
Hakan GÜNAY
(070160069)

Endüstri Mühendisliği Anabilim Dalı

Tez Danışmanı: Arş.Gör.Dr. Hüseyin Kutay Tinç

HAZİRAN 2021

To our families and friends,

FOREWORD

The fact that data science and machine learning concepts are already being used and developing today and the need for these concepts is increasing day by day, has made the concept of Data Science one of the most popular technologies of today. As three industrial engineering students who are aware of the importance of data science, we decided to move forward in this field.

We would like to express our sincere thanks to Hüseyin Kutaç Tinç, who supported us in every field during this process and our university education. He has made great efforts for us to come to this point and gain awareness.

June 2021

Berk KAPUCU
Ömer Seyfeddin KOÇ
Hakan GÜNAY

TABLE OF CONTENTS

Page

ABBREVIATIONS

ANN	: Artificial Neural Networks
ARIMA	: Autoregressive Integrated Moving Average
EIA	: United States Energy Information Administration
EMRA	: Energy Market Regulatory Authority
EPEX	: European Power Exchange
EU	: European Union
EXIST	: Energy Exchange Istanbul
GDP	: Gross Domestic Product
GRU	: Gated Recurrent Units
KGUP	: Finalized Daily Generation Programs
KOPI	: Unpaid Market Transactions
LASSO	: Least Absolute Shrinkage and Selection Operator
LSTM	: Long-Short Term Memory
MLP	: Multilayer Perception
MLR	: Multiple Linear Regression
OECD	: Organization for Economic Co-operation and Development
OEYE	: Match-Eliminate Proposal
RNN	: Recurrent Neural Networks
RSS	: Residual Sum of Squares
SAE	: Stacked Autoencoders
SLR	: Simple Linear Regression
TEIAS	: Turkey Electricity Transmission Company
TEYE	: Match or Eliminate All
USA	: United States of America

SYMBOLS

$\widehat{\beta}_0$	: Predicted intercept of the regression line
$\widehat{\beta}_1$	: Predicted coefficient of the 1^{st} variable
$\widehat{\beta}_j$	: Predicted coefficient of the j^{th} variable
λ	: Tuning parameter in LASSO and Ridge regression
$\sigma(x)$	: Sigmoid function
β_j	: j^{th} beta coefficient
a_j	: Net input of the neuron j
b_0	: Intercept of the regression line
b_j	: Bias term of the neuron j
b_j	: Regression coefficients
e_i	: Difference between i^{th} observed target value and the i^{th} target value
$f(a_j)$	: Activation function of the neuron j
h_t	: Hidden state of the network at time t
w_{ji}	: i^{th} weight of the neuron j
x_{ij}	: i_{th} observation value of the j_{th} variable
X_{ip}	: Values of example i on the j = 1, 2, … p predictors
x_j	: Output value of the neuron j
y_i	: Observed target value of the i^{th} observation
$\hat{Y}_i$	: Predicted value of ith observation in a regression

LIST OF TABLES

Page

LIST OF FIGURES

Page

INTRADAY ELECTRICITY MARKET PRICE FORECAST

SUMMARY

Energy has made a great contribution to humanity in the last century, facilitated human life and changed the future of the world. Countries add issues related to energy security and energy independence to their strategies in the 21st century. At this point, energy has a critical importance for both people and countries. Recently, electricity has become a tool and its production is made by private companies rather than the state, so this market has also become an open trade area. With the large volume of electricity, whose consumption is rapidly increasing and of great importance, there is a very high flow of money in this area. Technology, which has an impact on every aspect of our lives, also affects the energy markets. In recent years, the use of "machine learning" and "artificial neural networks" algorithms in many areas has been increasing very rapidly. It is an inevitable consequence that these algorithms will appear on a global scale in electricity markets as a field of use. We have dealt with balancing a high level of activity carried out on a Turkey intraday electricity market like global scale like provides an appropriate environment for the realization of this study. However, studies conducted under this topic head generally focus on non-deep issues such as descriptive statistics and the importance of variables. From this perspective when we consider the missing thick point, the forecasting work in this area in Turkey is not more specific enough.

Our study starts with the introduction part, which includes the place of electricity in Turkey and in the world from past to present, and then includes why electricity is important and the points that can be related to electricity production in the coming years. In the next stage, there is a part that includes the structure of the Turkish Energy Market. It is important to understand the workings of the market in order to interpret forecasting models. In the last part, using the data obtained by EPIAS, more than 10 simple and advanced models were processed and an forecasting algorithm was created.

In this application part, one hour delayed forecast was made using "moving average", "weighted moving average", "single exponential smoothing", "double exponential smoothing", "auto regressive", and "auto regressive moving-average" models, respectively. The outputs of these simple models are meaningless to the market because models must be forecasted after 2 hours or more to be applicable. At this point, more advanced models were tested in our study and forecasting models were developed for 2 hours and more. Before applying to the models, anomalies were detected with Facebook's Prophet library, and then these extreme data were replaced with the predictive values of the same library. In the next step, 5 different feature selection algorithms were applied since there are more than 50 features in our main data source. For the models, 9 target features, which are the common selection of these 5 different feature selection algorithms, were determined. In the last stage, Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Gated Recurrent Unit (GRU), Convolutional Neural Network (CNN) and CNN-LSTM models were

developed, respectively. Among these models, the CNN-LSTM model gave the best result with an absolute mean error of 20.79. Python software language was used in the development and implementation of all these models.

GÜN İÇİ ENERJİ PİYASALARINDA FİYAT TAHMİNİ

ÖZET

Enerji, son yüzyılda insanlığa çok büyük katkı sağlamış, insan yaşamını kolaylaştırmış ve dünyanın geleceğini değiştirmiştir. Ülkeler 21. Yüzyılda stratejilerine enerji güvenliği ve enerji bağımsızlığı ile ilgili konuları eklemektedir. Bu noktada enerji hem insanlar hem de ülkeler için çok kritik bir önem taşımaktadır. Son zamanlarda elektiriğin alınıp satılabilen bir araç haline gelmesi ve üretiminin devlet eliyle değil de daha çok özel iştirakler tarafından üstlenilmesi sebebiyle bu piyasa da açık bir ticaret alanı haline gelmiştir. Tüketimi hızla artan ve çok büyük bir önem taşıyan elektriğin bu denli büyük bir hacme sahip olması ile bu alanda çok yüksek meblalarda bir para akışı gerçekleşmektedir. Teknoloji, ticaretin olduğu her alanda etkin bir geliştiricilik misyonunu üstlenmesinden elektrik piyasaları da nasibini almıştır. Son yıllarda "makine ögrenmesi" ve "yapay sinir aglari" algoritmalarinin birçok alandaki kullanımı çok hızlı bir şekilde artmaktadır. Bu algoritmaların bir kullanım alanı olarak elektrik piyasalarında da global çapta karşımıza çıkmaları kaçınılmaz bir sonuctur. Ele aldığımız Türkiye gün-içi elektrik piyasasında global çaptaki benzerleri gibi gerçekleştirilen yüksek düzeyde dengeleme faaliyeti bu çalışmanın gerçekleşebilmesi için uygun bir ortam sağlamaktadır. Fakat bu konu başlığı altında yapılan çalışmalarda genel olarak tanımlayıcı istatistik ve değişkenlerin önemi gibi yüzeysel konular üzerinde yoğunlaşılmaktadır. Bu açıdan bakıldığında eksik kalındığını düşündüğümüz nokta, bu alanda tahminleme çalışmalarının Türkiye özelinde yeterince fazla olmamasıdır.

Çalışmamız öncelikle elektriğin Türkiye'deki ve dünyadaki geçmişten günümüze yerini içeren giriş kısmı ile başlamakta ve sonrasında elektriğin neden önemli olduğunu, önümüzdeki yıllarda elektirik üretimiyle alakalı gelinebilecek noktaları içermektedir. Bir sonraki aşamada Türkiye Enerji Piyasası'nın yapısını içeren bir kısım bulunmaktadır. Tahmin modellerini yorumlamak için piyasanın işleyişini kavramak önemlidir. Son kısımda ise EPIAS tarafından elde edilen veriler kullanılarak 10'dan fazla basit ve gelişmiş model ile bu verileri işleyip bir tahminleme algoritması oluşturuldu.

Bu uygulama kısmında sırasıyla "moving average", "weighted moving average", "single exponential smoothing", "double exponential smoothing", "auto regressive", ve "auto regressive moving-average" modelleri kullanılarak bir saat gecikmeli tahminleme yapılmıştır. Bu basit modellerin çıktıları piyasa için bir anlamı yoktur çünkü modellerin uygulanabilir olması için 2 saat veya daha sonrası tahmin edilmelidir. Bu noktada çalışmamızda daha gelişmiş modeller denenerek 2 saat ve daha ötesi için tahmin modelleri geliştirilmişitr. Modellere uygulanmdan önce Facebook'ın Prophet kütüphanesi ile anomaliler tespit edildi daha sonra ise bu uç veriler yine aynı kütüphanenin tahmin değerleri ile değiştirildi. Sonraki aşamada, ana

veri kaynağımızda 50'den fazla özellik olmasından dolayı 5 farklı özellik seçim algoritması uygulandı. Modeller için, uygulanan bu 5 farklı özellik seçim algoritmalarının ortak seçimi olan 9 tane hedef özellik belirlendi. Son aşamada ise sırasıyla Long Short-Term Memory (LSTM), Bidirectional LSTM (BiLSTM), Gated Recurrent Unit (GRU), Convolutional Neural Network (CNN) ve CNN-LSTM modelleri geliştirildi. Bu modeller arasından en iyi sonucu 20,79 mutlak ortalama hata ile CNN-LSTM modeli verdi. Tüm bu modellerin geliştirilmesi ve uygulaması kısmında ise Python yazılım dili kullanıldı.

1. INTRODUCTION

1.1 Purpose and Motivation

Intraday energy market in Turkey was founded in 2015, shows an activity intended to reduce the imbalance problems that occurred on the day-ahead energy market. Due to this activity, it reaches a high trading volume with an increasing trend by market participants. In addition, since the data set with historical clearing prices in this market is too complex to be used in any algorithmic activity, more work is required in this area. In such an environment, the need for market participants need a price estimation tool that can be put forward on this market will enable Turkey's prestige has contributed in this area is our main source of motivation.

2. LITERATURE REVIEW

2.1 Energy and Resources

2.1.1 Worldwide Trends

In the 21st century, many new concepts have emerged showing the development levels of countries. In addition to the national income per citizen used since the past, energy consumption per capita has been added today. In other words, the amount of energy produced by a country, the amount of energy per capita or the amount of energy it consumes gives us information about the social / economic development level of that country. In the article published by Lu (2016), energy consumption of developed sectors in Taiwan between 1998-2014 was examined. In parallel with these consumptions, growth rates in the economy were recorded and as a result, there was a 1% increase in electricity consumption and a parallel increase of 1.7% in gross domestic product (GDP).

The rapid rise of the world economy in the 21st century has been parallel to the energy sector. Energy, which is the raw material of economic development and independence, continues to be at the top with its strategic importance for countries. The demand for energy continues to increase along with global growth and population increases. Although this rise in economic and energy in the 21st century increases production and consumption and stimulates trade in the world, it creates another security problem that needs attention for countries. Therefore, countries have added energy independence and security to their strategies in the 21st century.

However, according to Terzic (2013), energy security and independence should be considered differently. Every president after Richard Nixon emphasizes "energy independence" or "energy security" without exception and directs his strategies by including this. However, it should be considered whether energy independence is appropriate for politicians to achieve these goals. At this point, the main issue is energy security. To achieve this, instead of reducing energy imports from foreign countries, it is to reduce energy imports from enemy countries. In addition, it is certain that a much safer energy policy will be created by diversifying energy sources. In this way, it will be difficult to manipulate the global energy markets and a much safer energy market will occur.

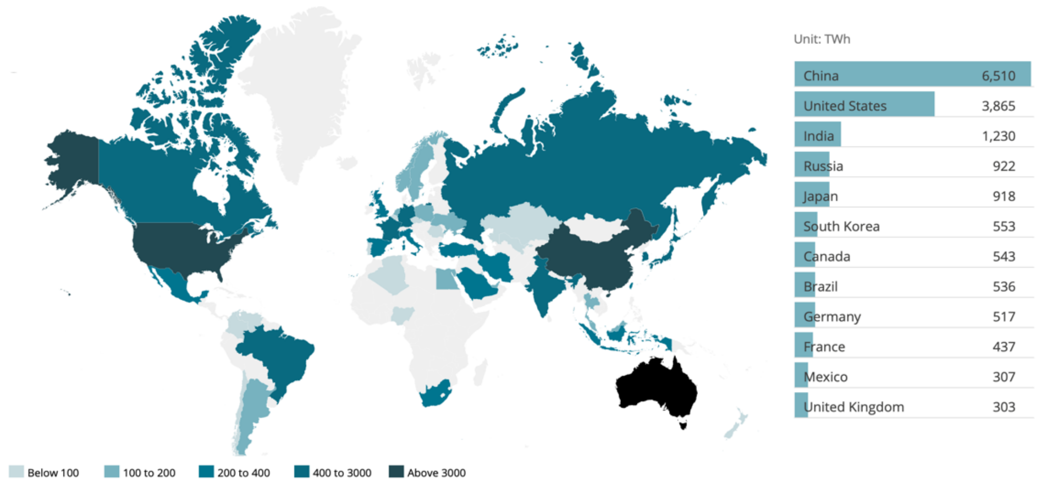


Figure 2.1 : Electricity consumption by country.

It should not be forgotten that fossil fuels are finite and will run out one day. This is a very important issue that should not be ignored besides energy security and independence. Fossil fuels that entered our lives in the 20th century made our lives easier and offered a much more comfortable life, but at the same time alarms for future generations. According to the research conducted by Tertzakian et al. (2007), 1000 barrels of oil per second is consumed in the world. Developed countries such as America and China make up a large part of this consumption pie. While countries are making plans to ensure energy security and impose restrictions on fossil fuels, they accelerate and support their transformation to energy with technology and laws. Technological developments reduce the production costs of renewable energy sources such as solar and wind energy, and contribute to the increase in energy efficiency and installed power capacity. In addition, governments have passed laws supporting renewable energy, accelerating the transformation to renewable energy. However, there are some difficulties in terms of sustainability of renewable energy sources and prevention of climate change around the world. The most prominent of these challenges are market failures, lack of knowledge in people and management, problems with access to raw materials for future green energy sources, and most importantly, our inefficient use of energy by people (Owusu & Asumadu-Sarkodie, 2016).

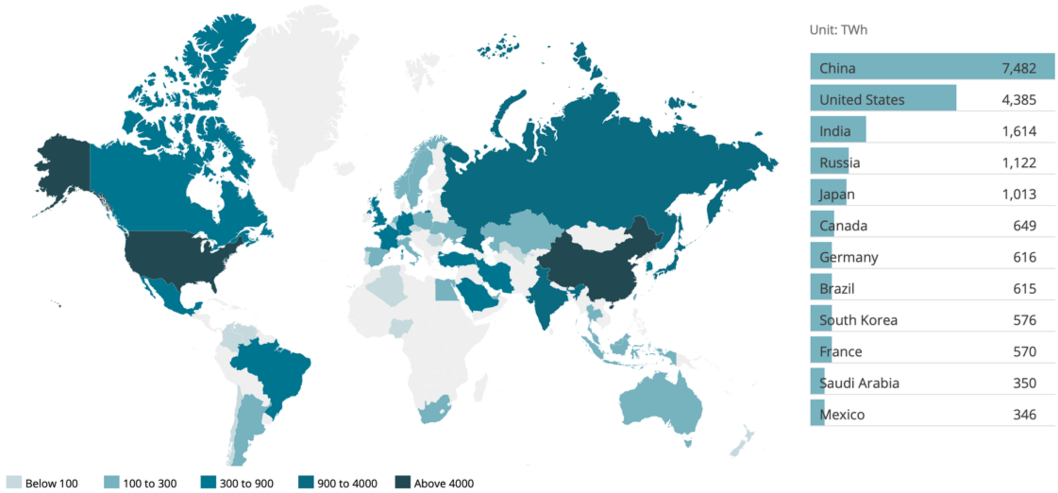


Figure 2.2 : Electricity production by country.

Despite all this, the figures 2.3. show that energy production with fossil fuels, which has been used worldwide for years, seems to maintain the leadership for many years. Political developments in countries with fossil fuel reserves, especially oil, have a great impact on the world energy market. This situation disrupts the forecast balances on the supply side. Therefore, countries dependent on foreign countries in energy make a great effort to produce their own energy. For example, the 1973 oil crisis in the world and the resulting increasing socio-economic problems pushed the countries that are largely foreign-dependent in the field of energy to very difficult conditions.

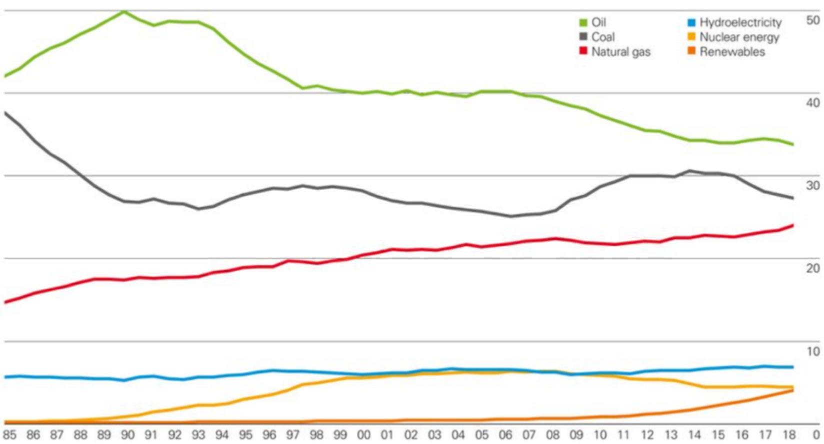


Figure 2.3 : Electricity consumption by country.

Global energy demand will increase in half by the time we come from 2020 to 2050 (United States Energy Information Administration, 2020). The biggest reason for this global energy demand will be developing countries. The main source of this increase is expected to be countries other than the Organization for Economic Co-operation and Development (OECD) members (Bilirgen, 2020). With the increasing population in these countries and the acceleration of economic growth, how to meet the increasing energy demand is a situation that needs to be planned today. China, United States of America (USA) and European Union (EU) countries meet more than half of the energy demand in the world. As of the end of 2020, the weight of China, which constitutes almost half of the total demand of the USA and EU countries group, is increasing every year. China, which has become a global factory, is the source of almost a quarter of the world's energy demand.

2.1.2 Trends in Turkey

Turkey is a country of growing faster than the world average. It supports this growth with population and natural resources. Energy consumption also increases regularly in parallel with this trend.

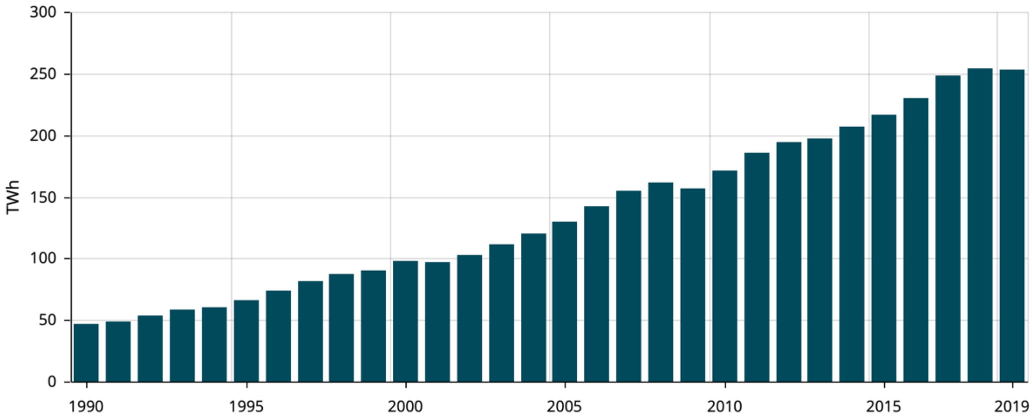


Figure 2.4 : Energy consumption of Turkey by years.

On the other hand, Turkey is having difficulty producing its own energy. Therefore, foreign dependency in energy has been at high levels throughout history. Rapid growth trend as a result of increased energy demand, creating large loads to Turkey in terms of imports. This poses a major problem for Turkey, which is 75% dependent on foreign energy consumption (Acaravci & Yıldız, 2018). For this reason, the risks affecting global energy prices have a significant impact on the financial assets and overall vulnerability of the country. Therefore, Turkey great of effort for the energy nationalization. While energy consumption is increasing year by year, important investments are made in the field of energy. With these investments supported by the desire to reduce foreign dependency. The power, which was below 10,000 MW in the mid-1980s, was increased to 91,300 MW by the end of 2019. The power increase in the 10-year period between 2009 and 2019 is more than doubled (Turkey Electricity Transmission Company, 2020). Turkey Electricity Transmission Company (TEIAS) also aims to increase the installed capacity of up to 109.500 MW in 2 years.

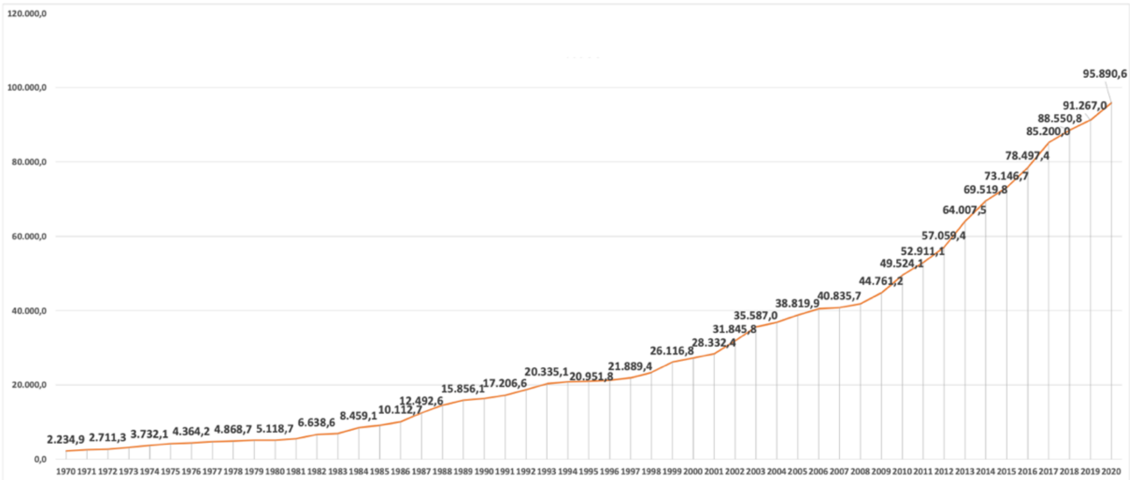


Figure 2.5 : Energy production of Turkey by years.

As a result of the incentives given to domestic and renewable energy investments, the installed power capacity is also in an increasing trend. Turkey reached 95.890 GW of installed power at the end of 2020. This total installed power is met from more than 8500 plants (Turkey Electricity Transmission Company, 2020). Solar power plants take the biggest share among these plants with 7,518 units. Looking at the sources of the power table, the largest generation of power comes from hydroelectric and natural gas power plants. In addition, it is noteworthy that the number of solar power plants is high.

Table 2.1 : Energy distribution for each source.

	Number of Plants	Installed Power (MW)
Stream	577	8059
Asphaltite Coal	1	405
Waste Heat	83	369
Dam	133	22925
Biomass	275	1115
Natural Gas	343	25672
Fuel Oil	11	306
Sun	7518	6667
Imported Coal	15	8986
Geothermal	60	1613
Lignite	47	10119
LNG	1	2

Motorin	1	1
Nafta	1	5
Wind	332	8832
Hard Coal	4	810

The investor structure of the installed power changes within the framework of state policies. The private sector's support for investments caused the capital structure to reverse. At the end of 2019, 25% of the total installed power belongs to the public capital. This rate was 68% in 2002.

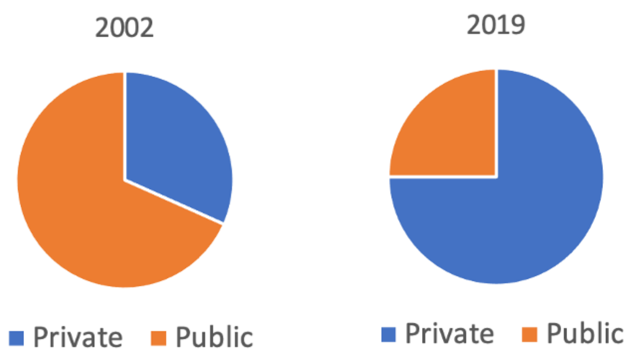


Figure 2.6 : Public and private capital comparison.

2.2 Energy Markets in Turkey

The functioning of existing energy markets in Turkey, transparency in market operations and the execution of existing Energy Exchange Istanbul (EXIST) is responsible. EXIST was officially established on March 14, 2013 with the binding of the necessary laws and conditions in this field and then started its activities actively by obtaining the market operation license from the Energy Market Regulatory Authority (EMRA) on September 1, 2015. The main field of activity specified by the institution and the general framework that it has established as a goal: “The planning, establishment, development and operation of the energy markets included in the market operating license in an efficient, transparent, reliable way to meet the needs of the energy market”. It is to be an energy market operator where liquidity reaches the highest level with the increasing number of market participants, product variety and transaction volume, and enables trade through market mergers to ensure reliable reference price formation without discrimination between equal parties. It is defined as. These institutions in September 2015, with license to operate in Turkey by the electricity sector began to have a serious evolution. After the license process, "Transparency Platform" was put into use on 14 March 2016, "New Day-Ahead Market" was activated on June 1, 2016, and a structure suitable for the existence of the private sector in the electricity markets was designed with a number of activities like this. The private sector share in electricity generation increased to 77.7% at the end of September 2020 (Industrial Development Bank of Turkey: Energy Outlook, 2020). There are 3 different market types available regarding the pricing of electricity. Each type of market is a structure that operates under various rules. EXIST is responsible for the transparency of these structures. These markets are day-ahead market, intraday market and balancing market.

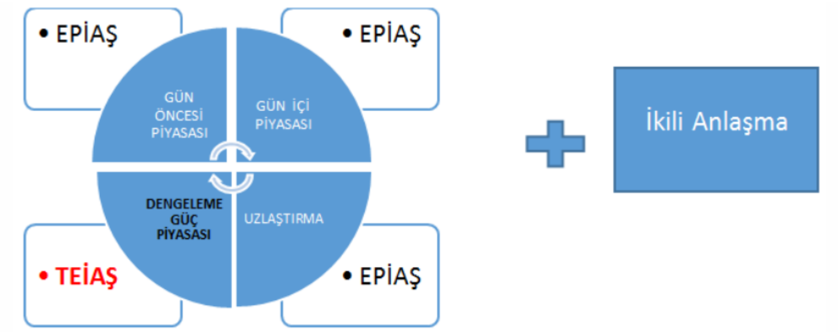


Figure 2.7 : Members of energy market.

2.2.1 Day-Ahead Market

After the market operation license of EXIST in September 2015, it has implemented many innovations in this context with the aim of regulating the existing electricity markets in this license in a more transparent, reliable, effective and meeting the needs of the sector. One of the aims of these innovations is to bring the current electricity markets to the same efficiency level as their counterparts abroad.

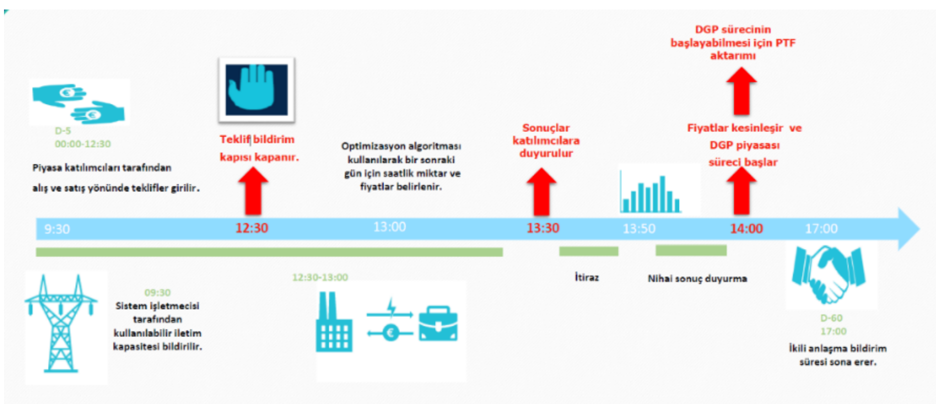


Figure 2.8 : Flow of energy market.

The day-ahead market revealed on December 1, 2011 was revised with domestic resources (during this revision process, it was examined in detail in equivalent sources in Europe). The system took its final form on June 1, 2016. In case of to be a part of this market, the “Day-Ahead Market Participation Agreement”, which contains the obligations of the participants, must be signed.

The day-ahead market is an active market, which is used for the balancing of the electricity trade and the purchase and sale one day before the delivery of the electricity, which is run and supervised by the market operator. The day-ahead market accepts transactions on an hourly basis on a daily basis. It is accepted as the process starting from 00:00 in each day and until 00:00 the next day. This interval is divided into hourly time slots. Bids in the market are arranged in such a way that they can bid within the 5-day period starting from the next day of the day. Settlement prices determined in this market are determined separately for each hour on a daily basis. The transaction, which is called the daily advance payment notification, regarding the money to be received and given with the announcement of the settlement prices, is announced to the participants by the market operator (EXIST) the next day. The central settlement bank mediates the announcement process. In case of payment arising regarding these receivables and receivables, the market participant submits the payment to the market operator until 10.30, if the payment is made as a letter of guarantee, and until 11.00 for other guarantees.

Market transactions start until 12.30, when the market participants submit their bids for the next day to the market operator through the new day-ahead market system. The bids submitted are checked and approved by the market operator between 12.30- 13.00 hours. Using the optimization tools available in the system with the approved offers, market clearing prices and market clearing loads are created for each hour of the relevant day. After the market clearing prices and loads are announced at 13.30, the current situation is notified to the participants and any objections that may arise are accepted until 13.50. Objections are evaluated between 13.50-14.00 and the last table will appear at 14.00. The table created here contains individual clearing loads and prices for 24 hours of the next day. In addition, bilateral agreements between market participants are notified through the new day-ahead market system between 00.00-17.00 hours. These agreements are controlled from 17.00 to 17.05. During this control process, the "Unpaid Market Transactions (KOPI)" method is used. KOPI is generally based on the issues created by the market operator, the market transaction amounts of the market participants; It refers to the transactions that are determined to exceed the amount calculated using the data on sales, purchase, import, export and production. The process regarding the agreements canceled by KOPI method takes place between 17.05- 17.15.

The market allows participants to submit 3 different types of bids as hourly, block and flexible bids for the specified time intervals. These offers may have different pricing for different time zones. While submitting bids, only Turkish lira can be used as the currency and the reported bid has a unit of "₺ / MWh" and the bid can have a maximum price sensitivity of 1 percent (such as 49.98₺ / MWh). It is mandatory that the bids made can be expressed as an integer in "LOT". LOT unit corresponds to 0.1MWh in expression. While making offers, offers can be made for both buying and selling. Positive expressions such as +100 LOT are used for the buy-side orders, while negative expressions such as -50 LOT are used for the sell-side orders. When making royalty, the price range is limited to between $ 0 and $ 2000 per MWh. This arrangement has been made by the market operator and the right to change belongs to the market operator. In the bids submitted by the same participant on the same day, the latest state is based on the calculations.

2.2.1.1 Hourly Bid

These bid types consist of bids placed for 1 hour interval. The pricing levels formed include 64 levels, 32 for the buy side and 32 for the sell side. It ranks from the lowest price to the highest price at these levels. There are no both buy and sell offers for the same price level in this bid scale. When the supply-demand curve is created for calculations on the offers, the gaps between the bids are filled with the "Linear Interpolation" technique. The general purpose of this type of offer for the participants is to make a quick profit on electricity, mostly by buying energy at low price and selling it at high price.

Saat	Fiyat (TL/MWh)									
	0	49,99	50	79,99	80	109,99	110	199,99	200	2000
0 – 1 (Lot)	600	600	400	400	0	0	-200	-200	-1000	-1000
1 – 2 (Lot)	300	300	300	300	200	200	0	0	-200	-2000

Figure 2.9 : Example of Hourly Bid.

2.2.1.2 Block Bid

Bid types covering a certain consecutive time period are called block bids. The most basic rule of these offer types is that they contain at least 3 consecutive hours bids. The offers made in this method are either completely accepted or completely rejected, that is, the bid block does not have a chance to be partially accepted by the market operator. In this bid type, the participant is given the right to bid for 50 blocks in a day. A maximum of 6 blocks of block bids can be interconnected, bid blocks can have a maximum of 3 levels, and there can be a maximum of 3 blocks at the second and third levels. Interrelated blocks must be the same type of bid, that is, an entire block must have a buy or sell offer.

Saat	Fiyat (TL/MWh)	Miktar (Lot)
2 - 6	110	-1800
6 - 14	60	1500

Figure 2.10 : Example of block bid.

2.2.1.3 Flexible Offer

These offers are in the direction of buying and selling, in a certain time interval, which is at least 8 hours and a maximum of 24 hours. These offers are within a certain offer period, this period is maximum 4 hours, the amount to be determined by the bidder and the type of offer laid out in the price. A participant can make up to 6 flexible royalties within a delivery day. A participant cannot make both a buy and a sell offer to be valid during the same flexible bidding period.

PK	Fiyat (TL/MWh)	Miktar (Lot)
A	180	**-18**
B	160	**-150**

Figure 2.11 : Example of flexible offer.

One of the most important opportunities offered by the day-ahead market to the participants is bilateral agreements. These agreements provide an important opportunity for the participants in the market to cover their shortcomings and to make the best use of their surplus production. The most basic rule in the realization of these agreements is that the participants who will realize the agreement must enter the same values. These agreements can be notified up to 60 days later, according to the rules set by the market operator. In these agreements, participants use positive values for buying and negative values for selling, while making offers from their own angle.

While determining the market clearing price for the market before the day, a heavy mathematical modeling is created with the demand and supply surpluses that may occur by taking into account block bids, hourly bids, flexible bids and bilateral agreements. The conditions of the proposals, KOPI standards and various periodic factors are added to the model as constraints and the model is tried to be solved. For the solution of this model, four modules are used as "pre-processing", "heuristic", "optimization", "post-model algorithms". Along with the different algorithms in these modules, there are also algorithms that contain the effects of multiple modules, which are not specific to a single module, for verification and repair operations for use in various situations. In this solution system, all algorithms except reporting in the post-transaction part generate the clearing price. With the reporting part, this clearing price is announced to the participants.

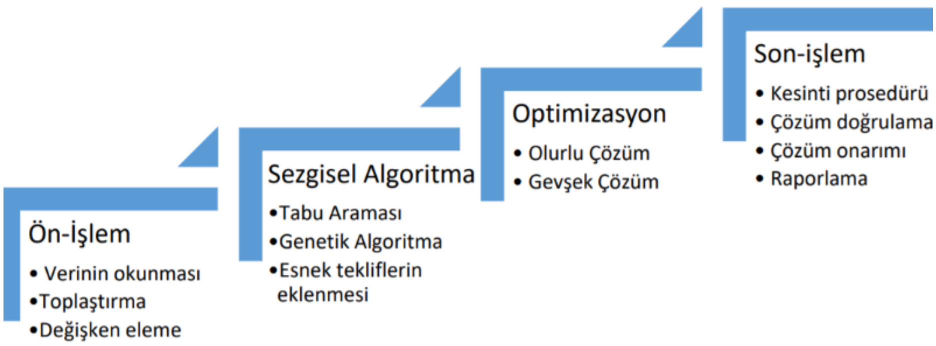


Figure 2.12 : Four modules.

In general, the day ahead market appears as a system that allows the participants in the market to balance themselves for the next day with bilateral agreements and determine the reference price of the electricity, and aims to balance the next day's system from the previous day. In this way, it facilitates the participants' work and prevents confusion in the market. According to the year of Turkey's energy consumption is constantly increasing. Therefore, the fact that we have a growing energy sector increases the importance of this market.

2.2.2 Intraday Market

The most important feature of this market is that it contributes to balancing by being a binding between the day-ahead market and the balancing market. In the day-ahead market, participants can bid up to 36 hours later. Participants can get rid of their current troubles with this market, especially in case of supply or demand differences that may occur due to various failures that may occur in renewable resources or unexpected changes in consumption. In addition, the intraday market, which emerges as a market where the participants can evaluate their capacity that they cannot utilize in the day-ahead market, creates an additional trading area for the participants. The intraday market started operating in July 2015 to provide the above mentioned contributions. The software of this market has been developed by Turkish engineers with completely domestic resources. The current system used in our country has no shortcomings from its counterparts.

Figure 2.13 : Flow of intraday market.

Unlike the day ahead market, this market is a continuous market where matches take place at any time of the day, not at a specific time of day. The bids valid for the next day are realized from 18:00 to the intraday market gate closing time after the end of the day-ahead market and bilateral agreements. Stated closing time is 1 hour before the physical delivery. During this period, the bid can be submitted, the bid can be updated or canceled. In this market, like the day-ahead market, the sales lower limit is $ 0, but there is no upper limit and no limit is applied to the participants for the number of offers. The only condition to bid in the intraday market is the collateral checks performed at 11:00 and 17:00 before the intraday market opening time. Participants who are out of the market during the controls carried out at 11.00 o'clock must complete their deposits until 17.00.

Offers made in this market are characterized by a variety of situations. These are: active, inactive, cancellation, timeout, realized and partial realized. The currency and unit terms in the offers to be made are the same as the day ahead market.

- Active: It is the status used for pending offers that have not reached a match status yet.
- Passive: Inactive by the participant who made the offer. It cannot be matched and visible to other participants until it is reactivated.
- Cancellation: These offer types are those that are canceled by the market operator or the bidder who made the bid. In addition to this, due to the structure of the offer, the offers can automatically turn into a canceled offer. These offers cannot be reactivated.
- Timeout: Unmatched offers until the time of extinction. It is removed from the pending proposal list without any change or evaluation.
- Realized: Offers that are realized in accordance with the specified conditions.
- Partially Realized: It is the hourly bids in which the “whole” bid does not match under the specified conditions. These offers wait until they expire. After it expires, the offer disappears and the new offer is created. According to the change made in the newly created proposal, several relevant situations occur.
    - Loss of Priority: If there is a change in price, the amount of electricity in the offer increases, or if the offer is activated from passive, priority is lost.
    - Maintaining Priority: If the offered amount decreases, priority is preserved.

In the Intraday Market, there are 2 types of bids for the participants. These are hourly bids and block bids.

1. Hourly Offer: They are available for both full and partial matching as a match. There is a contract format "PH14012018" created for this offer. Here, "PH" means that the offer is hourly offer, "140120" means that the offer was made on the 20th day of January in 2014, and "18" means that it belongs to 18:00..

    - Active Bid: This type of bid waits on the pending bid list until 60 minutes before physical delivery. If the offer does not match or does not match completely, it will be removed from the pending proposal list.

- Temporary Offer: In this type of offer, the participant is asked to determine the validity period of the offer. This period can be chosen up to 60 minutes before delivery.
- Match-Eliminate proposal (OEYE): This type of offer does not enter the list like the pending proposal. If the bid given exactly matches the current bids, it will result as the actual bid. If it doesn't match, the offer disappears. If the part is matched, the matching part is considered as the actual offer and the remaining part disappears.
- Match All-Eliminate Offer (TEYE): The difference from OEYE offers is that they can only match exactly. So there is no partial match for this offer. Other than that, other features are the same as OEYE.

2. Block Offer: These offer types are accepted as a whole as in the day-ahead market, and cannot be broken down in any way. For this reason, these offers do not have the option of realization like hourly offers. The scope of block bids here is a minimum of 1 hour and a maximum of 24 hours. In addition, in this type of offer, there is no chance to make an offer between the hours of 2 different days. There are 2 different types of offers that can be made within these rules. While creating these offers, a contract form as "PB14012019-04" is created. In this format, "PB" indicates that the offer is a block offer, "1411200" indicates that the royalty belongs to the 20th day of the 11th month of 2014, and "19-04" indicates that the proposal was formed starting from 19.00 and covering 4 hours.
   - Active Bid: Bids that are in the pending bid list until 60 minutes before the start of the bid These offers are either fulfilled or canceled and removed from the list.
   - Timed Offer: Unlike the active offer, in this type of offer, the bidder is asked to determine for how many hours the bid can be actively included in the list. This period can be defined up to 90 minutes before the start of the physical delivery at most.

In this market, when the offers are matched, there are 4 different qualities such as whether the offer is an hourly or block bid, as well as a buy or a sell offer.

The rule in the matching applied in hourly purchase offers is that the price specified in the purchase offer is greater than or equal to the current best selling offer price. If the bid entered is lower than the best selling price, this offer is added directly to the buy list and waits to match. When this rule is fulfilled, the matching offers are executed over the previously entered and waiting list sales price. If the LOT quantities in the match are equal, both offers are considered as actual offers and the offers disappear from the list. If the LOT amount specified in the purchase is greater, the sales offer is qualified as the actual offer, while the purchase order is considered as partially realized and a new match is searched for the remaining part. If such a match is not achieved, the offer is included in the list and waits for its customer. If the amount of LOT specified in the sale is greater, the transactions we mentioned this time take place for the sales offer.

The rule of hourly sale offers is that the best sell offer is less than or equal to the current best bid offer. If this rule is not met, the offer will be included in the direct sales offer list and wait for a match. Matching occurs when this rule is fulfilled. If the LOT quantities of the matched bids are equal, both bids are considered fulfilled and the matched bid is deleted in the buy list. If the LOT specified in the purchase offer is more than specified in the sales offer, the sales offer is qualified as realized, while the purchase order is considered as partially realized. In this case, the unmatched LOT amount in the purchase order is updated in the list and waits for a re-match, and the sales offer is qualified as the actual transaction. If the LOT amount specified in the sales offer is more than the purchase offer, this time the purchase order is considered as fulfilled and is deleted from the offer list and the sales offer is partially realized. Unrealized waits to match in the sales list.

When a new bid is received in block buy orders, this offer is compared with the appropriate block sell offers. When matching bids, the first criterion considered is the start time and coverage time of the bid and the LOT amount offered. When equality is achieved in these, the other factor, price, is passed. If the bid offer price is equal to the selling offer price, a direct match takes place and both offers are qualified as realized. If the purchase price is higher than the selling price, the transaction takes place over the previously available selling price in the table. There is no other matching method other than this. If there is no match in the first place, the bid offer is included in the list.

The first rule in block sales offers is to achieve equality in the starting time, coverage period and LOT amounts as in the purchase offers. Once equality is achieved in these criteria, prices are examined. If the selling price is equal to the purchase price, a match takes place. Both offers are considered as realized and the purchase offer is deleted from the list. If the selling price is less than the purchase price, the transaction is carried out over the previously listed purchase price. There is no other matching method. If there is no match in the first place, the offered sales offer is included in the list.

2.2.3 Balancing Power Market

Although a balanced market is attempted to be created with the Day-Ahead Market and the Intraday Market, various imbalances may occur due to various accidents (e.g. malfunctions in production facilities etc.) or instantaneous demand increases (e.g. instant activation of large production sites, etc.). Balancing power market operates in order to prevent imbalance that may occur in this case and any interruptions that may occur in this context. With this market, the system operator is provided with a backup power that can be activated within a maximum of 15 minutes. The system operator uses the offers offered on this market to maintain the balance when necessary. In general, any organization that can generate or receive 10MW of electricity within 15 minutes from operating organizations must be in this market.

Balancing Power Market starts operating at 14:00 every day. Participants are asked to submit their "Available Capacity" and "Finalized Daily Generation Programs" to the System Operator until 16:00. These offers are listed by the system operator in order of price separately for each hour. After this stage, the System operator informs the participants the necessary instructions for the offers that they find suitable to use when necessary.

KGUP is notified on an hourly basis for 24 hours between 14:00 and 16:00 for the next day on the basis of the "Settlement Withdrawal Unit".

Market participants can bid at the level of a total of 15 in the form of loading and shedding while making their offers. At the same time, the offer prices must be equal to the market clearing price or higher than the market clearing price. Load shedding prices, on the other hand, must be less than or equal to the market clearing price for the same time period. The difference between the offer prices is limited to a maximum of 20%. Also, freight prices must be higher than the previous level or the same as the previous level. Load shedding prices, on the other hand, must be less than the previous level or equal to the previous level. The minimum load amount used in the offers is 10MW.

The instructions mentioned in relation to this market are usually expressed with the labels "0" and "1". While the instruction expressed with "0" is a label given for the purpose of system balancing, the instruction expressed with "1" aims to eliminate the restrictions of the system. Instructions are expressed with only one type of tag, and the tag is preferred for whatever reason the instruction is more relevant. The instructions given in the balancing power market are transmitted to the participant either through the system or in written ways, except in case of emergency. After this transmission, the participant is obliged to apply the relevant instruction within 4 hours after the end of each hour. If not, participants must make an excuse. In emergency instructions, compliance with any offer, etc. Regardless of the criteria, the relevant order must be fulfilled. Such emergency instructions can be given not only to participants in the balancing power market but to all electricity producers when necessary. If the given emergency instructions are not followed, the system operator must be informed of the situation quickly.

2.3 Models and Forecasting Techniques

In this section, we introduce Multiple Linear Regression (MLR), variable significancy methods (Least Absolute Shrinkage and Selection Operator (LASSO) and Ridge), econometric, time series models, Artificial Neural Networks (ANN), Recurrent Neural Networks (RNN), Long-Short Term Memory (LSTM), Gated Recurrent Units (GRU), and some hybrid models are described in various markets for forecasting of intraday electricity price in Turkish intraday market. Especially Tensorflow, Keras and some other Python programming language libraries are used to forecasting the Turkish intraday electricity prices data, which are far from linear structure. After that, the performance of these selected models on the data are measured and compared with each other.

2.3.1 Forecasting Intraday Electricity Price

It is difficult to predict hourly spot prices in day-ahead and intraday electricity markets. Especially in systems where renewable energy systems are used at a high rate, these price predictions become more difficult. Since historical data include limited amounts of data, periods with low and price levels are difficult to predict, resulting in the problem of prediction bias (Andrade, Filipe, Reis & Bessa, 2017). In addition, a sensitive prediction could not be made due to the wide range of the estimation made. Moreover, these markets require more than one prediction variables to be included in the model, resulting in increased complexity of the model before the improvement in the forecast result can be guaranteed. For many reasons explained here, it is very difficult and complex to predict prices in domestic electricity markets.

In the intraday electricity markets, there are some insights that have been taken from the estimation results. Peak hours usually have a higher average price value than off-peak hours. Some conclusions have also been drawn regarding seasonality. For example, Furio (2011) observed in his study on the Spanish electricity market that daily seasonalities occurred as a result of low demand on weekends, and also observed that the prices of the last-time-negotiated hours tend to be higher than the same hourly period that was traded in previous sessions. This market is consistent with players in the market willing to pay higher prices when they have one last chance to buy electricity. Finally, it has been revealed that there is a positive relationship between energy and price in the majority of hourly periods trading in this market.

2.3.2 Forecasting Models

In this section the forecasting models that we use are explained in detail. These models we use are multiple linear regression, LASSO regression, ANN, RNN, LSTM, and GRU as mentioned.

2.3.2.1 Multiple Linear Regression

Simple Linear Regression (SLR) is a practical method for predicting a response variable based on a single predictor variable. However we have more than one predictor for forecasting intraday electricity price. Therefore MLR which is a statistical method in which more than one independent variable is used to predict the outcome of a dependent variable. Janke and Steinke (2019) have created a series of MLR models using different sub-sets of inputs to predict volume-weighted price distributions in the German continuous intraday electricity market. In addition, again for the German EPEX (European Power Exchange) intraday market, Ziel (2017) proposed a regression model to evaluate the impact of wind and solar energy-based errors when estimating intraday electricity spot prices. The purpose of multiple linear regression is to model the linear relationship between explanatory (independent) variables and target (dependent) variable. MLR analysis includes determining the multiple regression equation that shows the relation of the series of predictor variables to the response variable being studied. The basic design of the regression equation (2.3) as follows:

$$\widehat{Y_i} = b_0 + b_1 X_{i1} + b_2 X_{i2} + \ldots\ldots + b_p X_{ip} \tag{2.1}$$

where b_1, b_2, … b_j, … bp are regression coefficients and b_0 is the regression intercept. In addition X_{i1}, X_{i2}, … X_{ip} are the values of example i on the j = 1, 2, … p predictors. This equation is interpreted as when the value of one Xj independent variable is increased by one unit while the other independent variables are kept constant, the value of the dependent variable increases by the coefficient of the independent variable whose value is increased by one unit. Apart from the subject we mentioned, it is important to focus on the other issue is how the value intercept and coefficients of predictors are calculated. Let $\hat{y}_i = b_0 + b_1 * x_i$ be the prediction for Y based on the ith value of X. $e_i = y_i - \hat{y}_i$ then corresponds the ith residual which is the difference between the i^{th} observed target value and the i^{th} target value which predicted by our linear multiple regression model. We describe the Residual Sum of Squares (RSS) (2.2) as:

$$\mathrm{RSS} = e_1^2 + e_2^2 + \cdots + e_n^2 \tag{2.2}$$

or correspondingly

$$\mathrm{RSS} = (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \ldots + (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2 \tag{2.3}$$

The least squares method selects β_0 and β_1 to minimize RSS (2.3).

2.3.2.2 Variable Significancy Studies (LASSO and Ridge Regression)

In the intraday electricity price forecasting it is determined which variables are significant in the model by using the results of the models or the variable selection techniques, which will be mentioned shortly. A similar feature in these studies is that only lagged load and price values are included in the model as explanatory variables, usually together with calendar variables. However, some other variables can also affect the price and uncertainty in the intraday electricity market. For example, as a result of their studies, Dillig, Jung and Karl (2016) have revealed that the renewable energy sources generation has an undeniable effect on the spot prices of countries with high production and integration levels in this area, such as Germany, Portugal, and Denmark. Again in a similar study, a study was conducted for the six intraday market, and in the results, it was revealed that the best variables were the calendar variables, the daily session and the hourly prices of the previous day sessions (Monteiro, Ramirez-Rosado, Fernandez-Pedro Conde, 2016). As a different finding, Carmona and Coulon (2014) found that a significant majority of the information explaining intraday prices was related to the latest intraday trade for hourly products. Oksuz and Ugurlu 2019) think that this result might be due to the fact that Germany is the most mature and liquid intraday electricity market. In some problems, ANNs can also be used to select variables. For example, Keles et al. (2016) used an advanced ANN model to select and prepare appropriate data to estimate hourly electricity prices.

The LASSO regression method is broadly applied in electricity price forecasting researches because it has a potential to shrink the number of predictor variables on which the given solution depends. For example, LASSO regression has been applied in the German EPEX market to obtain statistical insights on variable selection in intraday electricity price and to make a very short-term electricity price estimation (Uniejewski, Marcjasz & Weron, 2019).

Before talking about LASSO regression, it should be mentioned a little Ridge regression, then comparing the two and emphasizing how LASSO regression is used in the selection of variables. The Ridge regression, which is very similar to least squares method with its attempt to make coefficient estimates that serve to minimize the RSS, differs in estimating the coefficients of independent variables by minimizing them. In Ridge regression, the coefficient estimates are made in a manner that minimizes the equality when the tuning parameter $\lambda \geq 0$ in the equation (2.4).

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2 = \mathrm{RSS} + \lambda\sum_{j=1}^{p}\beta_j^2 \tag{2.4}$$

The second term $\lambda \sum \beta_j^2$ in the equation, which causes the ridge regression to differ from the least squares method, is named shrinkage penalty. Also, it should be mentioned that the λ parameter in this equality is very critical because it serves to control the influence of predictive variables. For instance, as the λ tuning parameter value goes towards infinity, the regression coefficients estimate approach towards zero. However, when the value of the λ parameter approaches zero, the effect of the penalty term decreases and ridge regression returns to the least squares when the λ is equal to zero. It is an important drawback of ridge regression that it does not select a certain set among all the predictor variables and includes all it includes all the variables in the final model. Penalty term $\lambda \sum \beta_j^2$ entry will pull all coefficients towards zero, but will not pull any of them exactly zero unless the tuning parameter lambda is infinite. This situation may not cause a problem regarding the accuracy of estimates, but having too many predictor variables cause difficulties in interpreting the models. LASSO regression is a very crucial alternative to ridge regression that eliminates this disadvantage. LASSO regression equation is given equation 2.5.

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS} + \lambda\sum_{j=1}^{p}|\beta_j| \tag{2.5}$$

The only difference between them is that the term β_j^2 in the ridge regression penalty term is changed by $|\beta_j|$ penalty in lasso regression. L_1 penalty is used for lasso regression, while L_2 penalty is used in the ridge regression. Lasso regression, like ridge regression, brings the coefficients of predictor variables closer to zero, in addition, the L_1 penalty in lasso regression pushes these coefficients to be equal to zero when the λ is large enough. For this reason, lasso regression could be used for variable selection. As a result of all, interpreting models in lasso regression is much simpler than ridge regression.

2.3.2.3 Time Series, Probabilistic and Econometric Models

Narajewski and Ziel (2019) proposed an econometric time series model for the ID-3 price prediction in the German intraday electricity market. ARIMA techniques are used to analyze time series and are techniques that have proven successful in many different areas. While the Autoregressive Integrated Moving Average (ARIMA) technique is used in the intraday electricity markets, it is also used in the day-ahead electricity markets. Arroyo and Jose (2002) suggested predicting day-ahead electricity prices based on the ARIMA technique, and obtained their results from the Spanish and California markets.

2.3.2.4 Artificial Neural Networks (Multilayer Perceptron)

While deep learning models are used in estimating the prices of the intraday electricity market, in addition to this, deep learning models are also used successfully in the day-ahead electricity markets apart from the examined intraday electricity market. While deep learning models are used in estimating the prices of the intraday electricity market, in addition to this, deep learning models are also used successfully in the day-ahead electricity markets apart from the examined domestic electricity market. Lago, Ridder, and Schutter (2018) concluded that deep learning models are more successful in predicting day-ahead market electricity price than traditional machine learning methods and autoregressive time series models, with a statistically significant difference in accuracy. The human brain, which made up of 10^{10} neurons, serves to communicate via connection network system. ANNs, which have a structure similar to biological nervous systems, function as computational networks distributed in a parallel manner. If we talk about the Multilayer Perception (MLP) architecture, MLP architecture is a feed forward neural network in which non-linear neurons come together in successive layers and information on these layers flows from the input layer to the output layer in a unidirectional way through hidden layers in between (Figure 2.14). While the nodes in a layer have no connection between themselves and with the layers behind them, ther are connected to all nodes in the adjacent layer. The number of hidden layers is a very significant parameter for the network. Basically, in the MLP algorithm, first a forward propagation step takes place and then a backward propagation step comes.

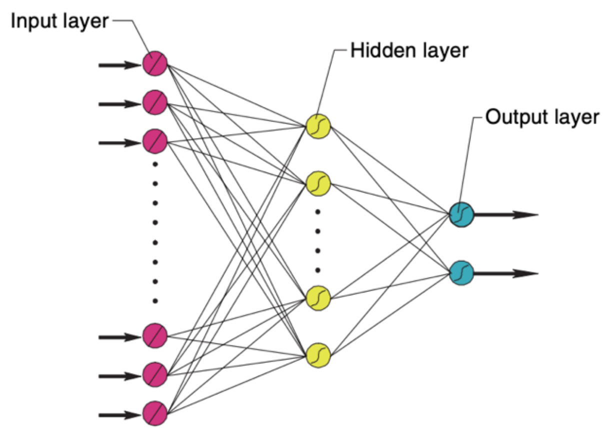


Figure 2.14 : Scheme of three-layered neural network, with one input,one hidden, and one output layer.

2.3.2.5 Forward-propagation Step

Shown in figure 2.15 with a single neuron connections. The strength of each connection between i and j neurons is indicated by weight (w_{ji}).

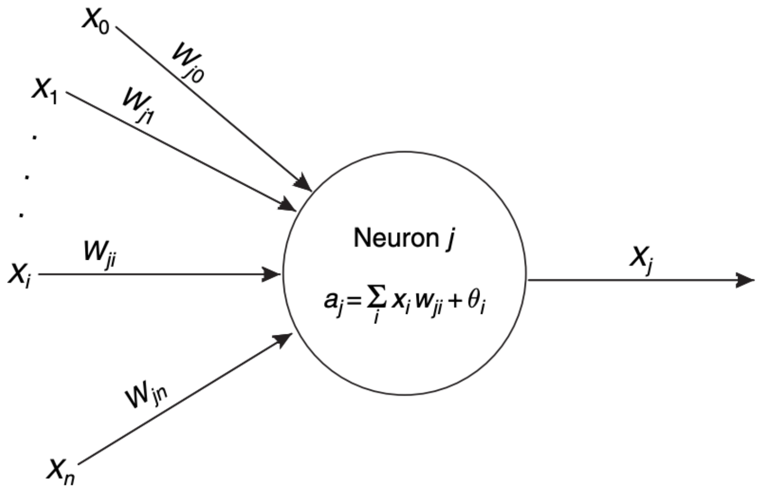


Figure 2.15 : Single neuron in a network.

The net input, called activation, is obtained from the product of its inputs for each neuron with their respective link weights. As seen in the equation 2.6, i indicates the total number of neurons in the background layer, while b_j is called bias term.

$$a_j = \sum_i x_i \; w_{ji} + b_j \tag{2.6}$$

Immediately after the activation of the relevant neuron is calculated as above, the output value which is a response, can be obtained by inserting one of the activation functions:

$$x_j = f(a_j) \tag{2.7}$$

Many different function could be used as the activation function. The most widely used of these are the relu, sigmoid and softmax function since they have nonlinear structure. The sigmoid function, which is one of these functions, is shown in equation 2.8. However, graphs of other functions are also indicated (Figure 2.16.).

$$x_j = f(a_j) = \frac{1}{1+e^{-a_j}} \tag{2.8}$$

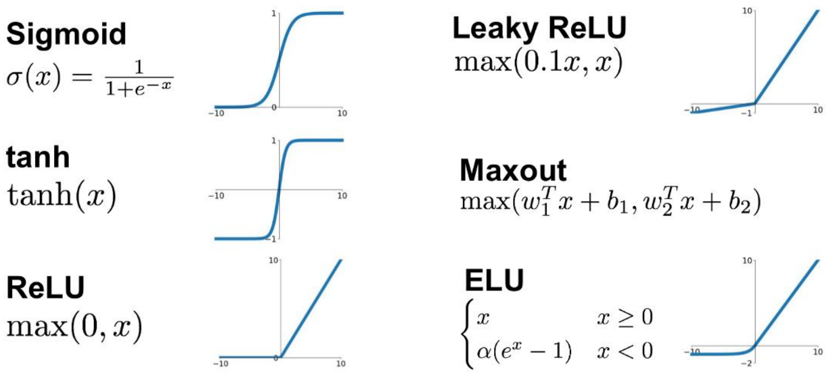


Figure 2.16 : Different activation functions

To summarize this forward propagation step in general, which is mentioned in details, it starts with the delivery of a data input to the input layer in the model, and this forward propagation step continues while the output layer after activation calculations is propagated forward through the hidden layers. At each consecutive layer in this network structure, all neurons collect their inputs and then use one of the activation functions (relu, sigmoid, tanh, etc.) to calculate their output. The output layer of this network then produces the result that predicts the target.

2.3.2.6 Output Layer

It is the most extreme layer of the neural network. It processes the data received from the hidden layer with the function used by the network and outputs it. The number of neurons in the output layer is equal to the output number of each data presented to the network. The values obtained from this layer are the output values of the artificial neural network for the problem in question. In the feed-forward phase, neurons in the input layer transmit data values directly to the hidden layer. Each neuron in the hidden layer calculates the total value by weighing its input values and transmits them to an forward layer or directly to the output layer by processing them with a transport function. The weights between the layers are randomly chosen from small numbers.

2.3.2.7 Backpropagation Step

A neural network is characterized by the network's architecture, the activation function in a neuron, and its training algorithm. The training algorithm generally used for training multilayer neural networks is the error back propagation algorithm. It can be used for any feed forward neural networks with a derivative activation function. If we define an error function for each of the network outputs like the total error function, the error function becomes the differential function of the weights. Therefore, we can find the derivative value of the error with the weights. These derivatives can be used to find the weights of the minimum error function by the slope reduction or optimization method. The algorithm used for the derivative value of the error function is known as the "back propagation" algorithm because it propagates the error backward through the network. The most used method in the literature for updating the parameters of artificial neural networks is the back propagation method. This method, which is successfully used in many areas where solutions are produced with artificial neural networks, from voice recognition problems to nonlinear system identification and control problems, is based on the minimization of a quadratic cost function over time by adapting the network parameters.

2.3.2.8 ANN for Intraday Electricity Price Forecasting

Various combinations of artificial neural network architectures can be used to address the intraday electricity price prediction problem. Oksuz and Ugurlu (2019) have proposed an architecture which consists of input, 3 hidden, and final fully connected layer with 1 neuron for final regression.

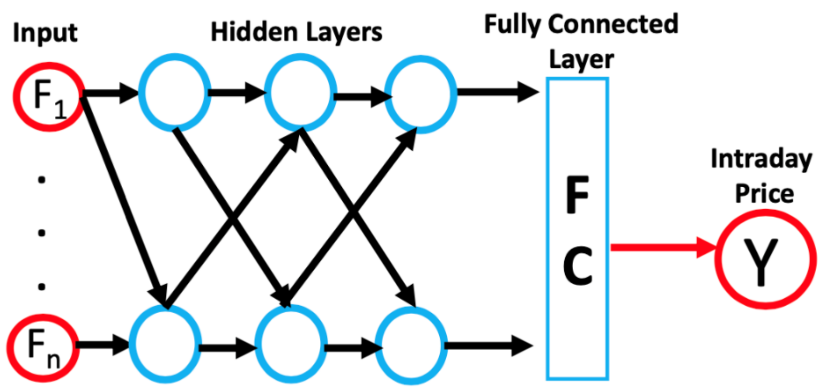


Figure 2.17 : ANN Architecture in Intraday Price Forecasting.

2.3.2.9 Recurrent Neural Networks (RNNs)

RNNs, a type of artificial neural networks, are used to model sequential data such as text, voice and time series data as input. For example, it has been demonstrated that RNN and Stacked Autoencoders (SAE) can be used to solve time series problems (Zhou et al., 2017). They are called recurrents because they perform the same task over and over for each element of an array. All inputs and outputs in a traditional neural network are generally independent of each other, but RNNs are networks with links between units and in which these links form a routed loop. Unlike feed forward networks, RNNs to process inputs it has its own input memories that it can use. They use their own memory to process data. With its cyclical structure, RNNs can make sense of the data that progress through the time series by providing the use of past information. The cycle is shown in figure 2.18.

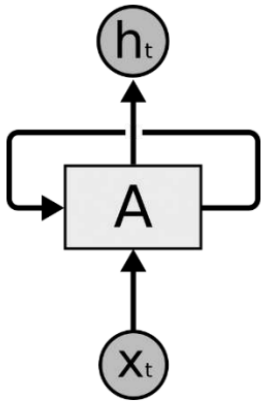


Figure 2.18 : RNN Cyclical Structure.

A simple recurrent neural network structure is given in the above figure. The rectangle labeled 'A' represents the cell in an artificial neural network. The network's input value is X and its output value is h. Every value that comes out of the cell comes to itself, forming a loop structure. Thanks to this cyclical structure, the neural network can also use the information of the previous time and make sense of new information using old information. In traditional artificial neural networks, results from cells do not come back to them as inputs. In RNN, the result coming out of the cell comes back to itself as an input. The RNN structure can be thought of as multiple copies of the same network in a time slot. When the expansion of RNN structure is examined, an architecture like the following figure 2.19. emerges.

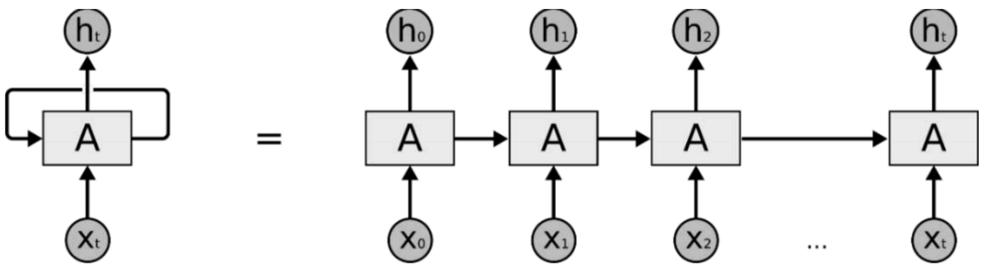


Figure 2.19 : Expansion of RNN Structure : RNN Architecture.

Backpropagation is required in the RNN training process as in many neural networks. This situation poses a big problem for the RNN structure. Each of the weights of the neural network is updated in proportion to the partial derivative value of the error function according to the current weight in each repetition of the training. The gradient is the value that allows all weights to be adjusted and is also dependent on the previous layer. If the back propagation process is continuously renewed in more than one time interval, the result becomes smaller and approaches zero, effectively preventing the weight value from being updated. This situation reveals the vanishing gradient problem. In the worst case, it can stop the neural network from getting further training. Conversely, there is a risk of encountering an exploding gradient problem when activation functions whose derivatives can take large values are used.

Although RNNs provide successful results in time-based problems due to their ability to connect with the past and make sense, they are weak in remembering the long past. All information is kept within the neural network, but some of this information is important, while some is unnecessary. Some transactions do not need to keep all history. LSTM networks with different architectures, a variety of RNNs, have been developed to solve this problem and gradient problems. These LSTM networks will be discussed in detail in the next topic.

2.3.2.10 RNN for Intraday Electricity Price Forecasting

Oksuz and Ugurlu (2019) input the properties and use 50 blocks for training tied to a fully connected layer with 1 node for estimating the intra-day electricity price. (Figure 2.20.)

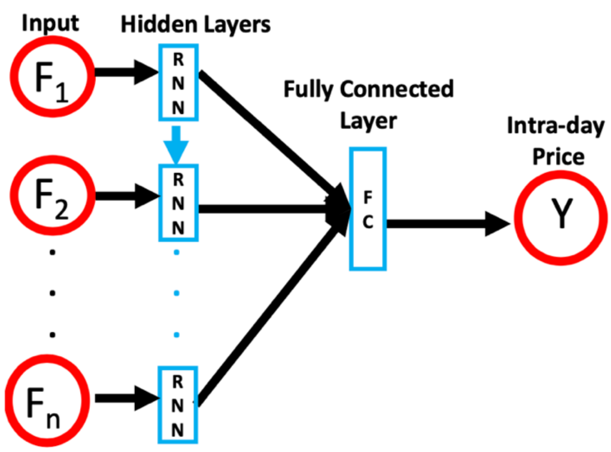


Figure 2.20 : RNN Architecture in Intraday Price Forecasting.

2.3.2.11 Long-Short Term Memory (LSTM) and Gated Recurrent Units (GRU)

LSTM networks are the type of RNN developed in 1997 by Schmidhuber and Hochreiter (1997) to solve the vanishing gradient and long-term dependency problems. In order to overcome this problem, Oksuz and Ugurlu (2019) proposed two different RNN architectures, LSTM and GRU, in their experiments. In addition to the intraday electricity market in the previous study, it was revealed that the use of LSTM gave significant successful results in the day-ahead electricity price estimates (Peng, Liu, Liu, & Wang, 2018). The theoretical architecture and working principles of LSTMs will be discussed starting from this here. LSTM have special interior gates that provide better performance and are more consistent than RNNs. Compared to other neural networks structure, LSTMs have been found to be quite successful in solving problems involving time series such as text processing, speech recognition and prediction. Its network design provides easy transfer of gradients. LSTM cells have the ability to keep input data in memory for a long time. The memories of these cells use the previous state data and the current input data for the next state input. They decide which data to keep in their memory, and then they create a network structure that produces more effective results by combining previous state data, data currently in memory and input data.

LSTM networks are sequential networks interconnected. In the LSTM architecture, there are four different layers in the form of a repeating chain instead of a single neural layer.

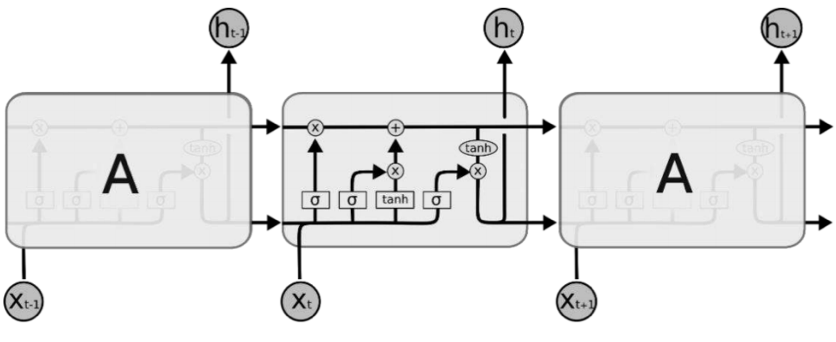


Figure 2.21 : LSTM network architecture.

LSTM neural networks have different activation functions connected to different gates and gates. Sigmoid and tanh activation functions are available in the figure above. The gates in the LSTM are the input gate, forget gate and output gate. The forget gate is the first stage of the LSTM network and decides whether to keep the information. The input gate consists of sigmoid neural network and tangent layer structures. In the Sigmoid layer, it is decided which values will be updated, and a new data structure that must be stored in the tangent layer is created. There are tangent and sigmoid layers in the output layer as well. In the tangent layer, it is decided how much of the stored information will be used, while in the sigmoid layer, it is determined whether the new information will be used or not. The value obtained by multiplying the values from the layers is determined as the output value.

Another RNN architecture, GRU, is built with two gates: the first is an update gate and the other is a reset gate. The previous memory is the update gate that is designed to be retained, while the reset pass allows memory combinations. Unlike LSTM architecture, GRU architecture has the ability to receive information from all hidden content. In this way, GRUs have a simpler and easier to train structure.

2.3.2.12 Hybrid Models

Artificial neural networks and models derived from it; They are very popular due to their non-linear nature, complexity, efficiency and flexibility. However, if the disadvantage is examined, a single neural network model causes certain limitations. For this reason, the use of hybrid models that use and combine multiple algorithms together in order to avoid and balance the disadvantages of a single model has become widespread recently. Kuo and Huang (2018) propose a deep neural network model (EPnet) that combines CNN and LSTM models using the previous 24-hour electricity prices to predict the next hour's electricity price. A hybrid model combining wavelet transform, ARIMA and Radial Basis Function Neural Networks (RBFN) was used to predict the price of electricity in the other market, day-ahead electricity markets (Shafie-khah, Moghaddam & Sheikh-El-Eslami, 2010). Again, in the day-ahead market, there is a study in which predictions are made using the Box-Jenkins methodology, which is one of the Wavelet-ARIMA models, but the non-linearity in the data could not be captured because the model was based on linearity approach (Conejo, Plazas, Espinola & Molina, 2005). Pousinho, Mendes and Catalão (2012) proposed a model based on the combination of adaptive network-based fuzzy inference and particle swarm optimization and system, which is a hybrid approach when making short-term electricity price forecasting in the Spanish market. The prediction of electricity prices is very difficult due to the nonlinear nature of the data and the intermittent increases in time series, regardless of the market for the day or ahead. At this stage, hybrid models come into play in these prediction models. For example, Sharma and Srinivasan (2013) proposed a hybrid model by combining the ability of Recurrent Neural Networks to match dynamic functions and the ability of connected stimulable systems to skip spiky time series. Kim and Won (2018) used a hybrid model that uses LSTM by integrating the price volatility in stocks, which is a prediction problem of a similar nature with intraday electricity price, and produces important results.

3. METHODOLOGY

3.1 Data Preprocessing

A data set considered is usually not created directly for our purposes. For this reason, it has to go through a series of processes in order to serve our purpose. A series of operations such as correcting the missing parts in a generally considered data set, various reduction operations performed on various inputs, discarding dependent variables obtained using various algorithms, if necessary, according to the dependency level, are called "Data Preprocessing." This process is carried out before the model is established and contributes significantly to the more effective operation of the installed model. This part is the part where half of the effort is spent in an average project when other steps are taken into account (García et al, 2017). According to Pintelas et al. (2006), this step has been examined under five headings. These headings are “Detection of Outliers”, “Missing-Data Treatments”, “Discretization”, “Normalization”, “Feature Selection”.

3.1.1 Detection of Outliers

This process is generally based on detection of outlier data. The presence of contrary data is a situation that prevents the algorithm to be established from giving proper results. These data do not contain meaningful information for our purposes (Luengo el al, 2019). These outlier data are commonly referred to as noise.

3.1.2 Missing-data Treatments

One of the main problems encountered during data preprocessing is filling in missing values (Alexandropoulos et al, 2019). This step is carried out with two different methods, such as automatically filling the empty parts of the available data with a certain method or manually filling the remaining empty parts after extracting the contrary data in a retrieved dataset. The difference of the data here from the outlier data is that it serves our purpose in terms of content, but it contains deficiencies in certain parts due to various reasons.

3.1.3 Discretization

The continuous numerical information we have at this step is brought into a separate and limited form, this data becomes more understandable to humans and is more suitable for machine learning algorithms (Herrera et al, 2020). This process, as he said, is performed when we have continuous numerical data. There are many methods that can be used while performing this operation, and the simplest of them is equal size discretization (Pintelas al., 2006).

3.1.4 Normalization

As a general meaning, it is defined as the reduction of input values to a certain range. As a result of the wide range of available values in the data we have, it is important to perform this operation for the algorithm we have set up to work more properly. There are many methods such as Z-score normalization or standardization, min – max normalization, unit length scaling (Alexandropoulos et al, 2019).

3.1.5 Feature Selection

As long as the data set we have is not prepared directly for our purposes, it is likely that there will be many irrelevant features in the data set. In this case, the feature selection process is applied to ensure that these unrelated features do not tire and slow down the algorithm, and to obtain a faster result. This step is also called Dimensional Reduction. The aim here is to create an effective and small data group by preserving the current distribution and quality of the classes and to take an action on this group (García et al, 2016).

3.2 Python

It was first created in 1991 by Guido Von Rossum. It is a very simple programming language in terms of usage and learning. Python is more effective and simpler than C++ and Java programming languages. Nowadays, most of the studies related to data science use the ython software language (Huang et al, 2019). Due to the opportunities it offers such as "Broadcasting", Python has increased its usability in fields such as data science and artificial intelligence, which actively use matrix structures.

3.3 NumPy

NumPy is an array programming library that provides users with a powerful and effective solution for matrix structures and high-dimensional array structures (Walt et al, 2020). This library, which is used with the "import" command in the Python programming language, has many commands and structures that will provide data manipulation.

3.4 Sci-kit Learn

It is a library related to machine learning in Python programming language. The purpose of this library is that, with its high-level content, even non-experts can easily set up both supervised and unsupervised machine-learning algorithms on Python. For this reason, it has been designed in such a way that a rich content as possible can easily reach the user (Pedregosa et. Al, 2011). Each type of algorithm is classified separately in terms of content, and an expression such as "from … import …" is used to access these algorithms. For example, "sklearn.linear_model import LinearRegression" statement provides linear model setup.

3.5 TensorFlow

Tensorflow is an open source distributed numerical framework and created by Google in 2015. It is primarily designed to alleviate the heavy operations performed on the "neural network" (Ganegedara, 2018). Having a flexible structure that can be differentiated within the data flow has an important effect in its use for neural networks. In this way, it mediates the establishment of comprehensive neural networks with multiple nodes and layers. It has been developing since the day it was first released, that is, it has a dynamic structure.

3.6 Keras

It is a Python library designed for deep learning algorithms that can run on TensorFlow. It is designed on the request of networking in a simpler way that arises due to the complex structure of TensorFlow (Manaswi, 2018). Keras; it can work well with Microsoft Cognitive, TensorFlow, PlaidML, Theano or R. Since this module does not tire the device while working, it contributes to the processing of higher content.

4. DATA

4.1 Data Sources

In this study, the data to be used while training the model was taken from the website of Energy Markets Management Inc. (https://seffaflik.epias.com.tr/transparency/). The date range between 01.01.2015 00:00 and 30.04.2021 23:00 has been chosen to ensure that the data in the models are consistent with each other. In this date range, the Intraday Market and Day Ahead Market values are the input values for the model. Through this platform established for transparency, data can be easily accessed on the website. Figure 4.1 shows this platform.

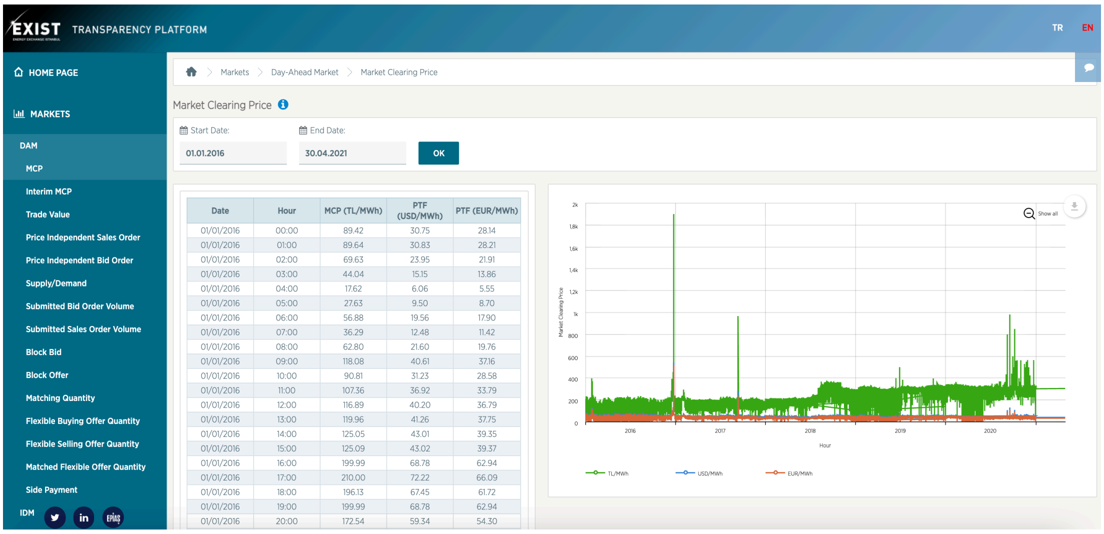


Figure 4.1 : EXIST website.

In this study, many market data such as Day Ahead Market, Intraday Market and production volumes are used. All collected data is divided hourly, so estimations are made hourly in the model.

4.2 Data Analysis

4.2.1 Intraday Market

In the period from 2016 to 2021, the intraday market shows an increasing trend. In the figure 4.2, the monthly average values of the intraday market by years are shown. Looking at the last 5 years, the lowest monthly average price was 113.1, while the highest average price was 326.2 TL in September 2018.

Month of D..	2016	2017	2018	2019	2020	2021
January	164.9	178.8	181.6	221.2	310.3	295.8
February	117.7	170.2	173.8	251.0	294.2	286.8
March	113.1	143.0	158.1	242.5	242.1	310.0
April	123.0	143.2	192.9	174.1	180.4	309.8
May	116.8	151.0	177.9	184.9	201.7	
June	146.3	149.2	185.1	208.8	293.0	
July	139.4	174.0	207.9	281.5	297.5	
August	163.1	171.2	298.4	289.2	299.1	
September	138.3	176.5	326.6	295.4	310.6	
October	139.5	163.1	315.8	290.0	319.8	
November	148.5	172.8	291.0	298.4	293.9	
December	217.6	155.3	256.2	290.5	294.0	

Date

Avg. WAP (TL/MWh)

113.1 326.6

Figure 4.2 : Average of WAP (TL/MWH).

The reason for the serious increase in prices is associated with the exchange rate problem Turkey had in the summer of 2018. It is seen that the serious increase in foreign currency is reflected in energy prices. In the energy markets, a price increase of almost 50-70% was experienced within a month.

Figure 4.3 : Turkish Lira / USD.

Only the data in figure 4.4 shows how dynamic and rapidly changing the Energy Market is. In such a dynamic and volatile market, price estimation provides companies with an important advantage. Finally, although the average prices showed a downward trend from the end of 2018 to July 2019, these high levels began to persist after June 2019.

Hour of Hour	2016	2017	2018	2019	2020	2021
0	150.8	155.8	239.4	230.0	270.7	301.6
1	131.6	142.6	226.9	249.3	290.2	293.0
2	116.1	130.2	204.1	236.5	263.7	281.1
3	96.0	121.9	194.6	206.1	246.2	269.4
4	90.7	117.9	187.1	191.2	237.3	267.1
5	84.3	117.2	199.9	191.9	228.5	272.8
6	83.0	133.5	194.1	202.3	223.5	274.4
7	111.9	150.7	207.6	232.1	241.4	278.7
8	139.7	174.4	233.9	256.6	278.1	301.4
9	173.7	191.1	247.6	251.7	263.3	293.6
10	184.7	193.7	244.2	263.4	273.4	304.4
11	193.7	196.9	248.4	276.0	283.4	309.7
12	162.0	181.1	228.8	243.8	261.6	293.0
13	167.4	186.4	240.1	242.0	272.6	297.1
14	179.9	190.9	247.6	261.9	287.5	307.6
15	172.6	184.4	247.2	256.6	288.2	305.7
16	166.9	180.2	251.2	266.3	297.2	311.4
17	160.4	169.3	247.4	281.3	309.6	325.1
18	152.7	169.1	246.7	286.9	313.8	331.4
19	152.5	173.7	247.7	295.0	324.4	337.9
20	158.8	170.6	251.9	297.2	322.2	330.1
21	150.5	167.4	250.0	293.5	312.6	322.3
22	142.3	151.5	232.4	283.8	301.3	314.7
23	123.7	143.5	217.6	260.0	282.5	297.1

Date

Avg. WAP (TL/MWh): 83.0 – 337.9

Figure 4.4 : Hourly average price.

When the intraday market is analyzed hourly, it is seen that the increase by years shapes the general trend here as well. The average hourly prices increase as we move towards the right in the table. The first hours of the day have lower pricing compared to 9:00 and later. To see what the hourly distribution is like in general, figure 4.5 provides a more descriptive picture.

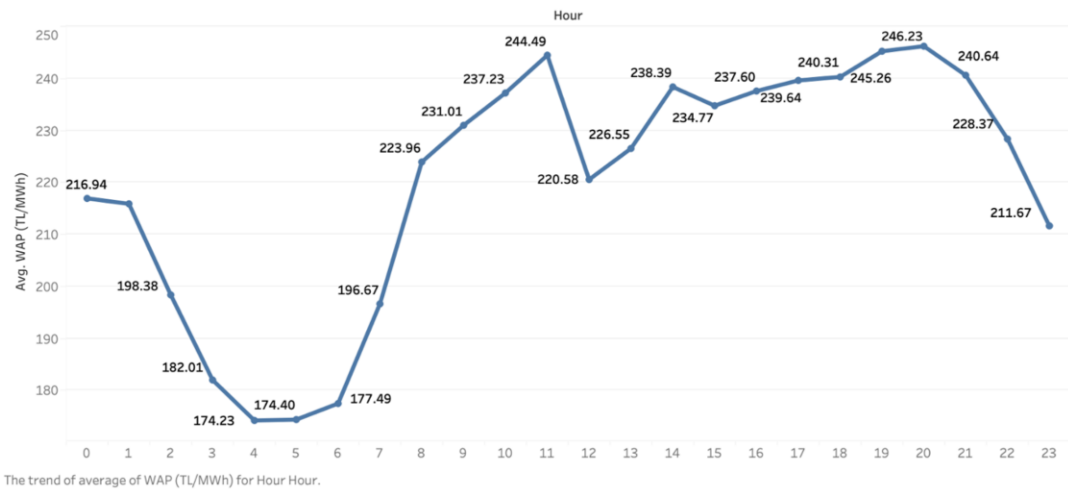


Figure 4.5 : Average hourly prices with line graph.

As can be seen in figure 4.5, it is the interval where the intraday market average prices are minimum between 02:00 and 7:00. The market goes into an upward trend from 8:00 until 11:00. Afterwards, it is seen that it decreases at 12:00 and follows a stable pricing until 20:00-21:00. Lastly, the size of the Intraday market by years is shown in figure 4.6.

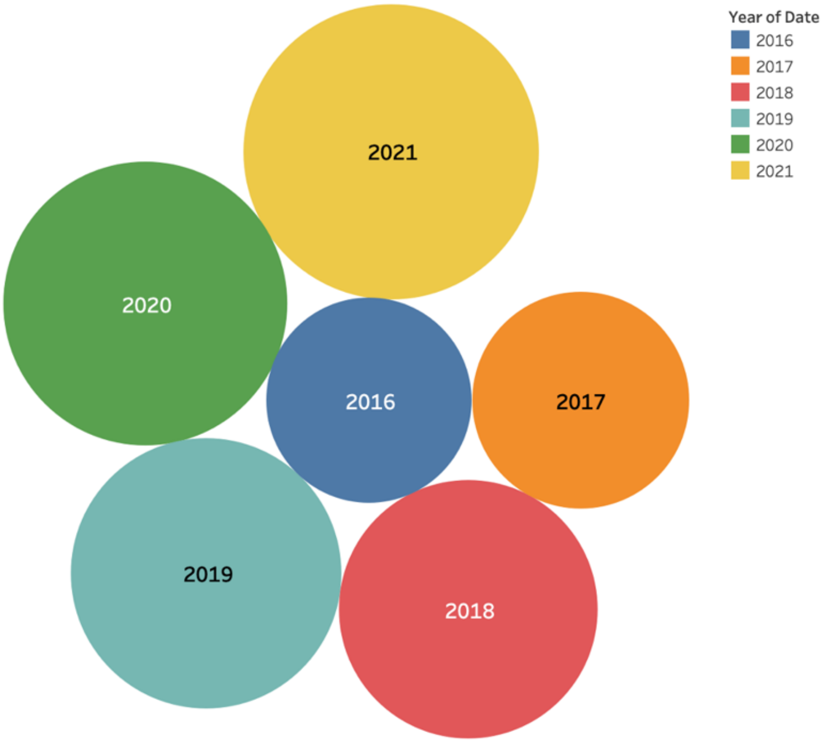


Figure 4.6 : Intra Day Market size by years.

4.2.2 Day Ahead Market

Electricity consumption and production forecasts / plans are made for the next day in the Day-Ahead Market. Electricity producers report how much electricity they will generate from how many TL. Likewise, distribution companies indicate their purchase amount and how much they are willing to pay per MW. EXIST institution tries to establish a supply-demand balance in order to avoid surplus / deficit with these offers. At the point where the purchase and sale quantities balance each other, a certain price or price range is also balanced (supply-demand balance is made every hour of the day). The price determined as a result is called the market clearing price (PTF). In the intraday market, the day ahead market prices also provide us with reference while estimating the price. Therefore, day-ahead market values are also used in the model. The day ahead market has an almost complementary role with the intraday market.

Hour of Hour	2016	2017	2018	2019	2020	2021
0	149.9	158.2	241.8	239.5	272.8	306.8
1	127.8	145.3	229.0	259.7	291.6	298.1
2	110.1	132.2	205.3	245.6	264.5	285.4
3	85.5	124.0	195.6	214.0	246.9	273.4
4	79.2	119.5	187.6	197.8	238.7	271.4
5	74.7	118.9	200.9	197.8	229.4	277.6
6	72.4	135.6	194.7	208.1	223.9	278.8
7	107.4	153.2	208.9	240.5	242.3	282.5
8	137.2	176.5	235.6	268.6	279.4	303.5
9	171.2	192.4	249.6	262.8	264.3	295.6
10	183.2	195.3	245.6	273.1	274.8	306.1
11	193.0	198.0	248.9	285.0	284.1	310.3
12	160.2	182.1	228.5	249.0	260.6	293.8
13	167.5	186.2	240.4	246.9	272.0	298.6
14	179.2	193.5	248.4	268.6	287.9	306.3
15	171.8	185.4	247.5	262.3	288.5	303.8
16	167.0	181.1	251.9	272.5	297.2	309.1
17	159.6	170.6	248.0	288.6	310.5	323.6
18	152.3	170.4	247.3	295.3	314.5	330.2
19	152.3	175.0	248.5	304.6	325.9	337.5
20	158.6	172.6	253.5	306.3	323.5	329.8
21	150.4	169.6	251.3	302.9	312.9	322.4
22	141.9	152.9	232.9	292.0	300.9	314.4
23	121.2	143.6	217.4	266.3	282.0	297.0

Date

Avg. MCP (TL/MWh)

72.4 337.5

Figure 4.7 : Day Ahead Market hourly prices (TL).

In the day ahead market, as in the intraday market, it is observed that there is an increase in prices as the years progress. Figure 4.7 shows the distribution of the average prices of the day ahead market by hours. Looking at the last 5 years, the lowest hourly price is 72.4 TL, while the highest price is 337.5 TL.

Hour of Hour	2016	2017	2018	2019	2020	2021
0	49.62	43.45	49.63	42.25	39.30	40.70
1	42.29	39.90	46.87	45.92	42.08	39.43
2	36.40	36.31	42.00	43.30	38.10	37.72
3	28.05	34.08	40.11	37.72	35.45	36.13
4	25.93	32.85	38.47	34.87	34.21	35.85
5	24.44	32.64	41.03	35.06	32.94	36.69
6	23.52	37.11	40.34	37.03	32.30	36.99
7	35.23	42.05	43.07	42.70	34.93	37.46
8	45.20	48.47	48.30	47.52	40.31	40.24
9	56.54	52.84	51.21	46.48	38.05	39.24
10	60.49	53.62	50.43	48.34	39.61	40.58
11	63.74	54.40	51.05	50.44	40.96	41.12
12	52.74	50.03	46.69	44.09	37.51	38.95
13	55.26	51.19	49.01	43.64	39.12	39.57
14	59.02	53.21	50.76	47.47	41.33	40.59
15	56.62	50.96	50.44	46.33	41.37	40.27
16	55.09	49.79	51.43	48.15	42.61	40.98
17	52.41	46.90	50.51	51.08	44.52	42.90
18	49.97	46.76	50.51	52.28	45.18	43.73
19	50.08	48.02	50.90	53.88	46.90	44.68
20	52.30	47.41	51.97	54.12	46.66	43.65
21	49.68	46.59	51.43	53.49	45.17	42.66
22	46.80	42.00	47.68	51.58	43.41	41.61
23	39.84	39.47	44.31	47.05	40.67	39.33

Date

Avg. PTF (USD/MWh)

23.52 63.74

Figure 4.8 : Day-ahead market prices (USD).

In the scenario where the market data is analyzed in dollar terms, a completely opposite picture is formed compared to the TL. As seen in figure 4.9, while the intraday market took its highest value in dollar-based pricing in 2016, there was a decrease in prices as the years progressed.

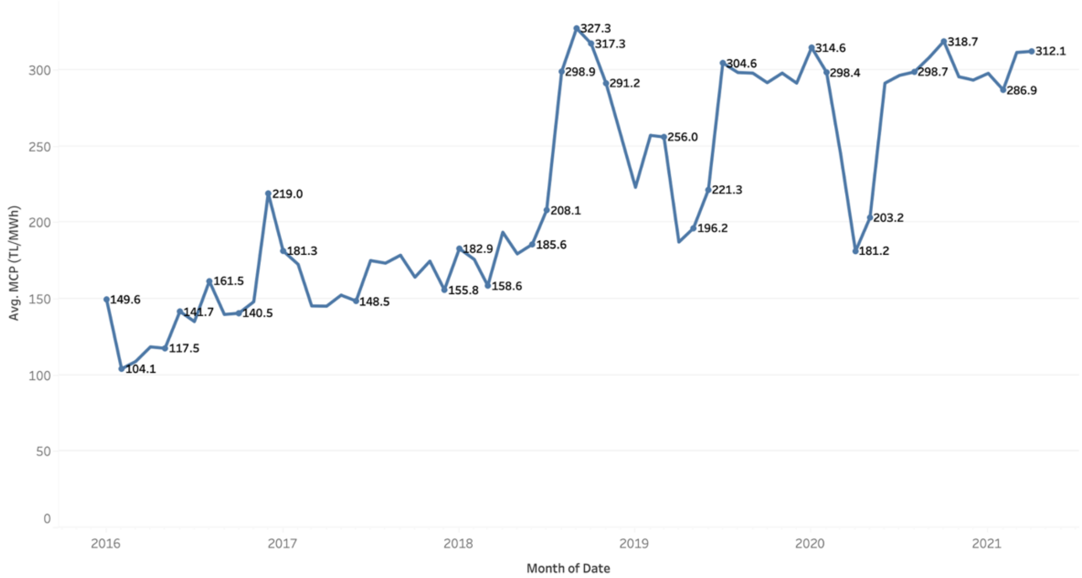


Figure 4.9 : Average PTF prices by month.

Finally, when the monthly distribution for the Day Ahead market is analyzed, Figure 4.10 is formed. As can be seen here, there are sudden decreases in average PTF values in certain periods. The values in this graph are also TL based.

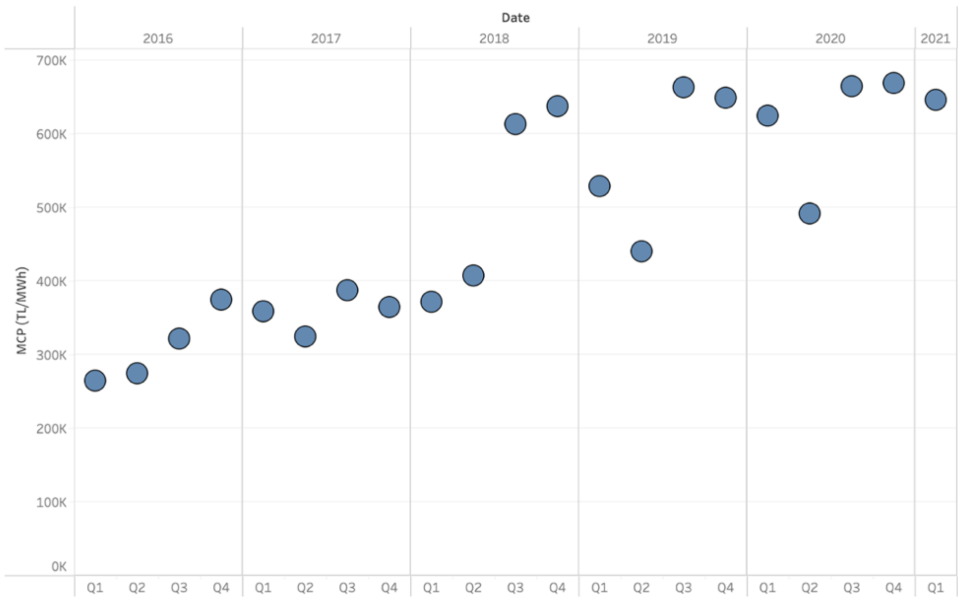


Figure 4.220 : Average PTF prices by quarters.

In the figure 4.10, it is seen that these decreases correspond to the first two quarters of the year. Towards the end of the year, there is an increase in prices and the last quarter of the year is closing at the highest values.

4.2.3 Electricity Production

Electricity generation in Turkey is currently actively carried out using both renewable and non-renewable resources. Electricity production continues with an increasing trend. When the generation data obtained from EPIAS are examined, it is seen that electricity generation is discussed under two main headings as "Planned" and "Actual" generation.

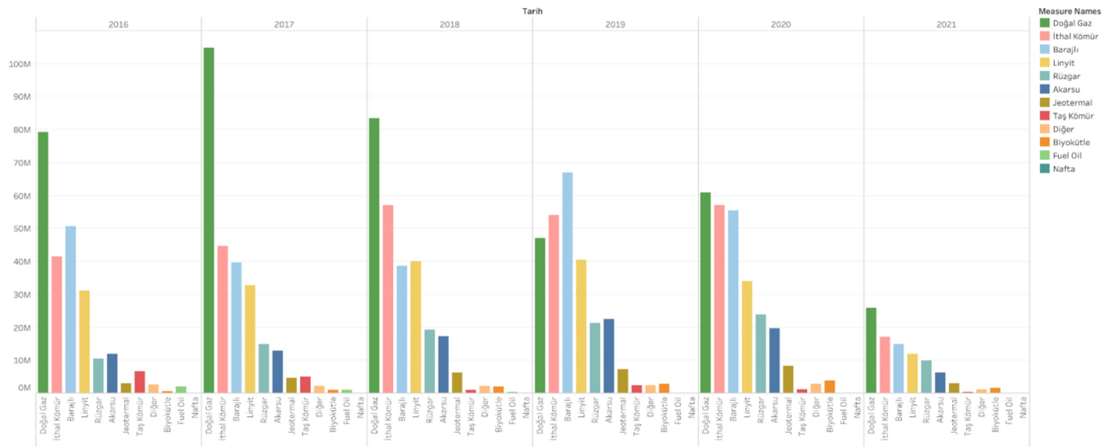


Figure 4.11 : Electricity Generation Planned According to Energy Resources.

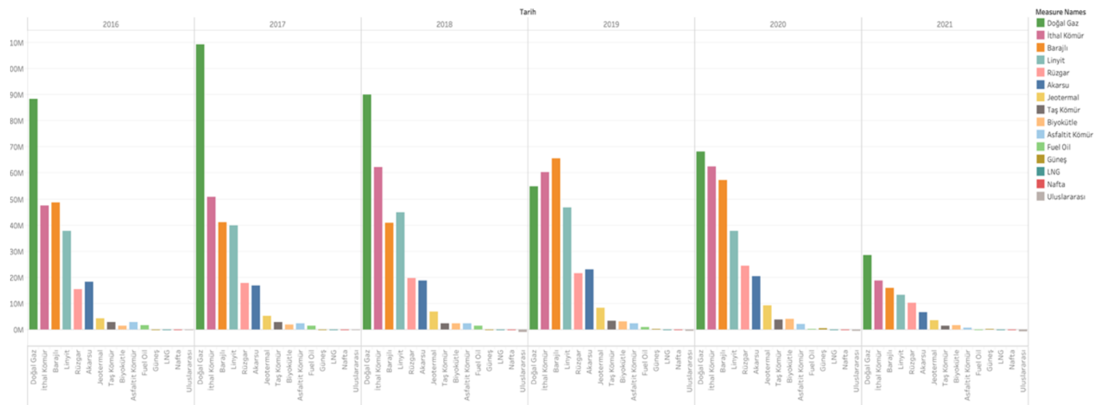


Figure 4.12 : Actual Electricity Generation According to Energy Sources.

First of all, when the distribution of Planned and Realized Production according to energy resources (Figure 4.11-12) is examined, it is seen that non-renewable resources occupy a larger place in electricity generation than renewable resources. In addition, it is obvious that the majority of the electricity supply in our country is realized by natural gas and imported coal-based power plants. With the effect of the recent pandemic, the production realized in these two sources has been reduced. Hydroelectric power plants stand out as an energy source during the pandemic period. In this case, the increasing exchange rates with the pandemic also have an effect. When we look at the production resources mentioned in the graphics, there are almost all production types available in our country, but there is no equal distribution among these production types. Particularly, the share of environmentally friendly renewable resources in the electricity supply of the country is very low. Wind Turbines attract the most attention among renewable resources.

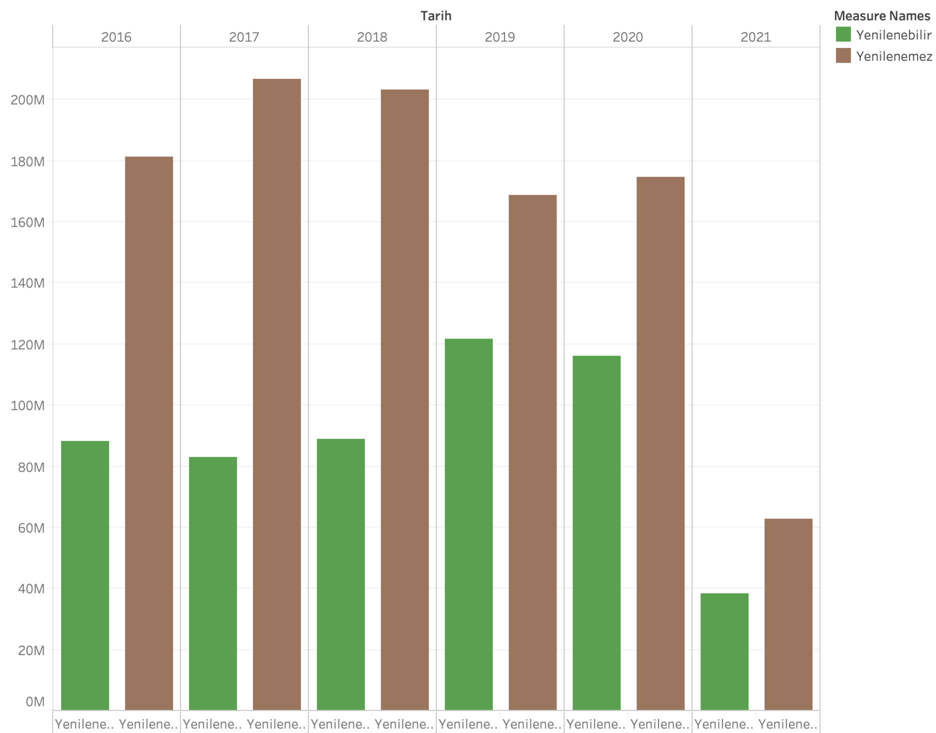


Figure 4.13. Energy Production According to Renewable & Non-Renewable Sources.

When renewable and non-renewable resources are examined more comprehensively, as can be seen in figure 4.13, the share of renewable resources in production has increased over the years. While the share of renewable resources in production was approximately 32% in 2016, it increased to approximately 43% in 2020. However, this increase is not only the increase in the use of renewable resources, but also the decrease in the use of non-renewable resources.

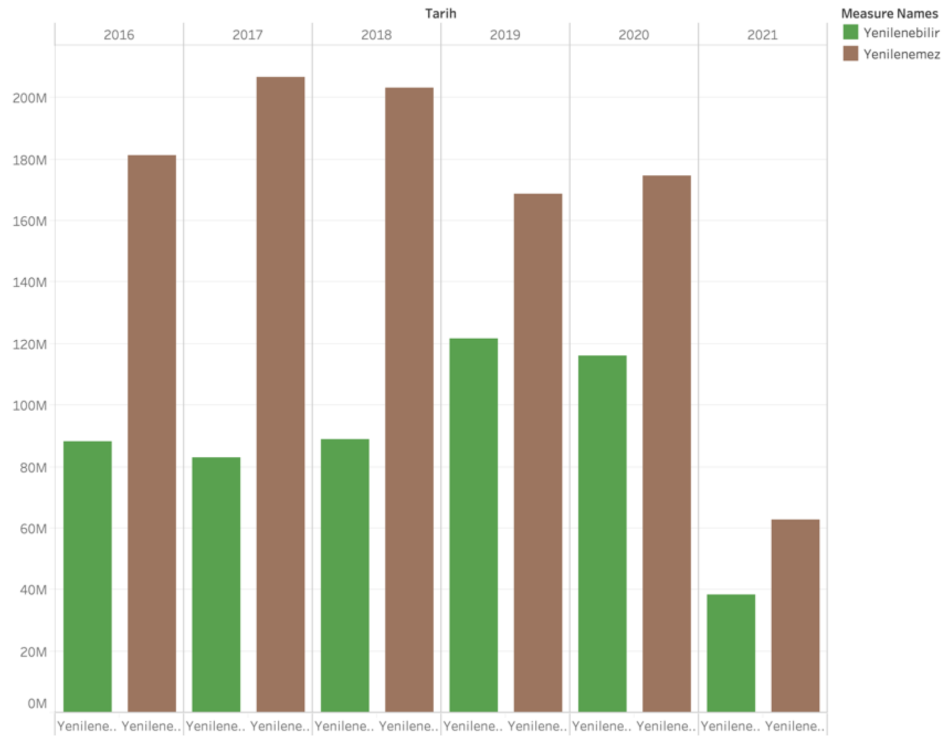


Figure 4.14 : Planned Energy Production (Renewable & Non-Renewable Sources).

The low share of renewable resources in production enables the amount of electricity generated to be realized in a more planned manner. This is a proposition that explains the overlap of the planned and produced energy, considering that the electricity supply in our country is generally based on non-renewable resources. When the graphs (Figure 4.13-14) are examined, there is a slight difference between the planned and produced electricity amounts.

	Tarih					
Quarter of ..	2016	2017	2018	2019	2020	2021
Q1	65,539,178	70,852,714	72,771,440	71,645,321	73,974,387	75,854,601
Q2	65,051,163	67,234,335	68,483,576	69,076,533	61,124,088	24,535,234
Q3	70,283,577	78,902,798	78,909,961	77,788,384	80,570,952	
Q4	68,416,589	72,865,411	71,005,759	71,305,065	74,557,960	

Figure 4.15 : Electricity Generation According to the Quarters.

When the actual electricity generation is examined on a seasonal basis, the time intervals in which the electricity production increases and decreases can be determined and a seasonality can be achieved through this determination. When the electricity production in our country is analyzed, although electricity generation is generally distributed almost equally, there is an increase in electricity production in the third quarter, that is, in the summer months. In addition, the detection of the first pandemic case in the second quarter of 2020 and the subsequent shutdown process seriously affected electricity generation. This effect can be easily detected from the data we have. In addition, the general increase in electricity generation in our country is clearly seen in the table above. When the distribution of each quarter by years is examined, year-on-year increases are detected, except for the second quarter of 2020. Considering the pandemic conditions we are in, this shows the confidence of the domestic market in the energy sector.

5. APPLICATIONS

5.1 Introduction to Application and Data

The data to be used in the application part is taken from the EPIAS system. Initially, the date, GIP AOF (TL/MWh), PTF (TL/MWh), positive/negative imbalance, production and consumption values were taken over the system. Then, using these data, the historical values for (h-1), (h-2) and other times were derived as separate features. Finally, the values found in the created database are listed under the following headings:

Table 5.1 : Features of data.

Features Name	h	h-1	h-2	h-3	h-8	h-24	h-48	h-72	h-168
Date	+								
Hour	+								
Day	+								
Weekday	+								
Month	+								
Year	+								
GIP AOF (TL/MWh)		+	+	+	+	+	+	+	+
PTF (TL/MWh)	+	+	+	+	+	+	+	+	+
GOP Transaction Volume	+								
DGP System Direction		+							
Total Production	+								
Positive Imbalance		+	+	+	+	+	+	+	+
Negative Imbalance		+	+	+	+	+	+	+	+

In the next sections, the number of features has been reduced by removing features that have little effect with feature selection techniques. Simple and advanced models were tried in practice. In simple models, the next hour is predicted, but the next GIP (TL) forecast has no effect on applicability. Therefore, advanced models have been developed to predict for 2 hours and beyond.

5.2 Basic Models

Before giving the data to the models, "Facebook prophet anomaly detection" was performed to detect exceptional values in the data. Then the previous week values of the same time of the same day were assigned as values to these free hours. The anomaly detection process will be discussed in more detail in the next sections. In basic models, related operations were performed via excel.

5.2.1 Moving Average

In the moving average method, 4 different scenarios were applied by taking the averages of the previous 2, 3, 4 and 5 hours.

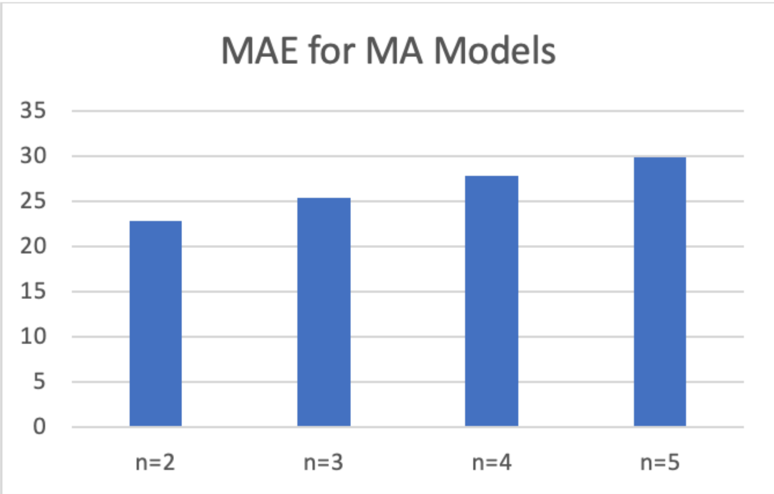


Figure 5.1 : MAE Values of MA Models

The model with the best MAE value is the 2-hour average model. As the past hours are included in the average, the predictive capacity of the model decreases. This shows that moving away from the reference time increases the deviation.

5.2.2 Weighted Moving Average

5.2.2.1 Scenario 1: Last 24 Hours.

Unlike the Moving Average method, estimation was made with the Weighted Moving Average method by assigning a certain weight to the values of the last 24 hours. The MAE value was chosen as the objective function and minimized. The weight of each hour was determined in the equation solved with the Excel solver. The weight of each hour in the past 24 hours is shown in table 5.2.

Table 5.2 : Weights for Weighted Moving Average method (Last 24 hours).

w1	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12
0,12	0,11	0,09	0,06	0,05	0,03	0,01	0,00	0,00	0,00	0,00	0,00
w13	**w14**	**w15**	**w16**	**w17**	**w18**	**w19**	**w20**	**w21**	**w22**	**w23**	**w24**
0,00	0,00	0,00	0,00	0,00	0,01	0,02	0,04	0,06	0,09	0,14	0,19

As in the MA, it is seen in the WMA model that the time closest to the reference time has a large coefficient. In addition, it is seen that the first value to be looked at to estimate the GIP value based on 24 hours is 24 hours ago (Coefficient: 0.19). In this model, the MAE value is 24.13.

5.2.2.2 Scenario 2: Last 2 Hours.

In the weighted moving average method, a much better result was obtained compared to the first model, based on the last 2 hours instead of the last 24 hours. The MAE value in this scenario is 19.56.

5.2.2.3 Scenario 3: Last 1-2 and 23-24 Hours.

The value (h-24) according to the weights in the first scenario also constitutes a great reference in price estimation. At this point, (h-1), (h-2), (h-23) and (h-24) values were used in the estimation in the 3rd scenario of the weighted average method. The MAE value in this scenario is 18.79.

5.2.3 Single Exponential Smoothing

The weight value was 0.88 in the single exponential smoothing method estimation. The MAE value is 18.83.

5.2.4 Double Exponential Smoothing

In the double exponential smoothing method estimation, the alpha value was 1.013 and the beta value was 0.001. The MAE score of the estimation made using these values is 19.61.

5.2.5 Auto Regressive

In the autoregressive model, 3 hours were looked at retrospectively and weights were assigned to each hour value. After solving the equation with Excel solver, the (h-1) coefficient has the highest value, while the (h-2) and (h-3) coefficients have lower values. The MAE value of the model is 19.55.

5.2.6 ARMA

The errors of the moving averages for the 3-hour time frame are weighted. In the result, the MAE coefficient of (h-1) was 0.45. The MAE of the model is 23.19.

5.2.7 Result

All of the models established in this section are simple models used to predict one hour ahead. The MAE value of each model is given in table 4, and these values are in Turkish Lira. However, these model outputs are not applicable in real life. In the following sections, it is tried to predict 2 hours later by using more complex models.

Table 5.3 : Basic Models Summary

Model	MAE
Moving Average	22.87
Weighted Moving Average (24h)	24.13
Weighted Moving Average (2h)	19.56
Weighted Moving Average (2h+24h)	18.79
Single Exponential Smoothing	18.83
Double Exponential Smoothing	19.61
Auto Regressive	19.55
ARMA	23.19

5.3 Feature Selection Methods

Since the size of the market data and the number of features are high, the data elimination process was carried out using feature selection methods. Feature selection was performed by using Exhaustive Feature Selection, Sequential Feature Selector (Forward), Sequential Feature Selector (Backward), Ridge and Lasso techniques separately. Data with common features found in all 5 methods were used in the models.

5.3.1 Exhaustive Feature Selection

The outputs of the EFS operation performed by specifying the maximum 9 features limit are as follows. According to this technique, the most important features that should be used in the model are: PTF (TL/MWh), PTF (h-1), GIP (h-1), PTF (h-2), GIP (h-2), PTF (h-24), GIP (h-24), PTF (h-168), and GIP (h-168).

```
Features: 14912/14912

Best subset (indices): (0, 3, 4, 5, 6, 10, 11, 12, 13)
Best subset (corresponding names): ('PTF (TL/MWh)', 'PTF (h-1)', 'GIP (h-1)', 'PTF (h-2)', 'G
IP (h-2)', 'PTF (h-24)', 'GIP (h-24)', 'PTF (h-168)', 'GIP (h-168)')
```

Figure 5.2 : EFS Results.

5.3.2 Sequential Forward Selection

The best 9 features were selected with the SFS technique, the results are as follows: PTF (TL/MWh), PTF (h-1), GIP (h-1), PTF (h-3), GIP (h-3), PTF (h-24), GIP (h-24), PTF (h-168), and GIP (h-168).

5.3.3 Sequential Backward Selection

The best 9 features were selected with the SBS technique, the results are as follows: PTF (TL/MWh), PTF (h-1), GIP (h-1), PTF (h-2), GIP (h-2), PTF (h-24), GIP (h-24), PTF (h-168), and GIP (h-168). Unlike the SFS technique, features have (h-2) values instead of (h-3).

5.3.4 Ridge Regression

Ridge Regression technique was also used for the selection of features. The coefficients resulting from the Ridge are as follows:

```
PTF (TL/MWh)             107.600822
GIP (h-1)                 65.405057
PTF (h-1)                 61.729332
GIP (h-24)                 9.468693
PTF (h-24)                 9.073299
PTF (h-2)                  6.856381
GIP (h-2)                  6.346050
GIP (h-168)                5.205890
PTF (h-168)                5.153173
GIP (h-3)                  1.501206
PTF (h-3)                  1.396933
GOP İşlem Hacmi (TL)       0.969204
Toplam Uretim (MWh)        0.474761
Demand                     0.025477
```

Figure 5.3 : Ridge Regression Coefficients

These coefficients also show that the most important feature when estimating the GIP value is the PTF value. Afterwards, (h-1), (h-24) and (h-2) values for GIP and PTF are significant for the model.

5.3.5 Lasso Regression

In addition to Ridge regression, the most important features were determined using Lasso regression. The coefficients created by Lasso regression are shown in figure 40.

```
PTF (TL/MWh)              75.693093
GIP (h-1)                 49.385248
PTF (h-1)                 44.869518
GIP (h-24)                 5.803321
PTF (h-24)                 5.322795
PTF (h-2)                  5.197321
GIP (h-2)                  5.117718
GIP (h-168)                3.376681
PTF (h-168)                3.056000
GIP (h-3)                  1.394135
PTF (h-3)                  1.318046
GOP İşlem Hacmi (TL)       0.939949
Toplam Uretim (MWh)        0.224147
Demand                     0.000000
```

Figure 5.4 : Lasso Regression Coefficients

As with all feature selection techniques, the most important features in Lasso were the PTF value. Then (h-1), (h-24) and (h-2) GIP&PTF values come, respectively.

5.3.6 Overview and Final Feature Selection

The results are very close to each other in the 5 different feature selection techniques used. The features to be used in the models after this step will consist of the intersections of these 5 techniques. The 9 features that stand out in each technique are given in the table below:

Table 2.4 : Model Comparison

EFS	SFS	SBS	Ridge	Lasso
PTF (TL/MWh)	PTF (TL/MWh)	PTF (TL/MWh)	PTF (TL/MWh)	PTF (TL/MWh)
PTF (h-1)	PTF (h-1)	PTF (h-1)	PTF (h-1)	PTF (h-1)
GIP (h-1)	GIP (h-1)	GIP (h-1)	GIP (h-1)	GIP (h-1)
PTF (h-2)	PTF (h-3)	PTF (h-2)	PTF (h-2)	PTF (h-2)
GIP (h-2)	GIP (h-3)	GIP (h-2)	GIP (h-2)	GIP (h-2)
PTF (h-24)	PTF (h-24)	PTF (h-24)	PTF (h-24)	PTF (h-24)
GIP (h-24)	GIP (h-24)	GIP (h-24)	GIP (h-24)	GIP (h-24)
PTF (h-168)	PTF (h-168)	PTF (h-168)	PTF (h-168)	PTF (h-168)
GIP (h-168)	GIP (h-168)	GIP (h-168)	GIP (h-168)	GIP (h-168)

5.4 Anomaly Detection and Filling

The price data in the intraday electricity market we have had very high volatility and anomalies due to the situation of the market and especially the increasing dollar rate. For example, the price that was 250 TL 1 hour ago could exceed 1000 TL after one hour due to the variability in the market, or there were no significant changes in the data 5 years ago due to the dollar rate, but after 2018, a significant upward trend was also seen in the TL based. The view of our data with 48 and 168 hour moving averages for both time periods is given below.

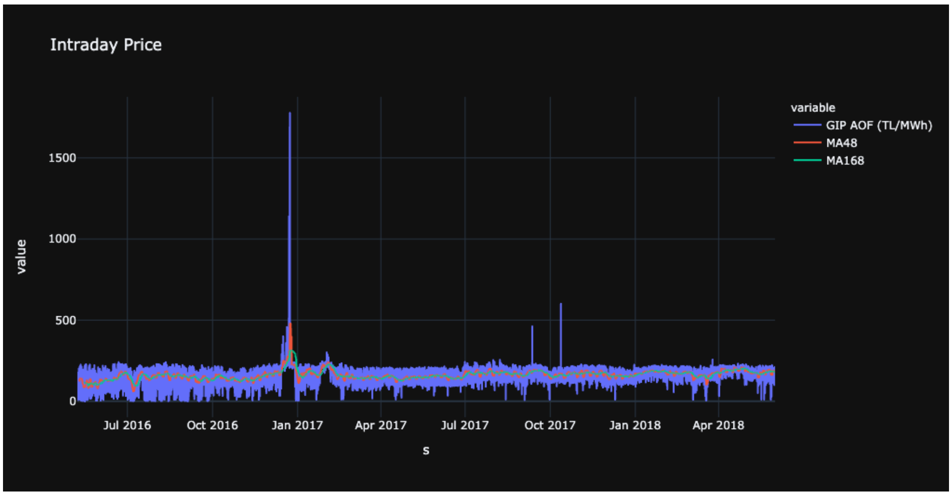


Figure 5.5 : Trend of data with moving averages between 2016-2018.

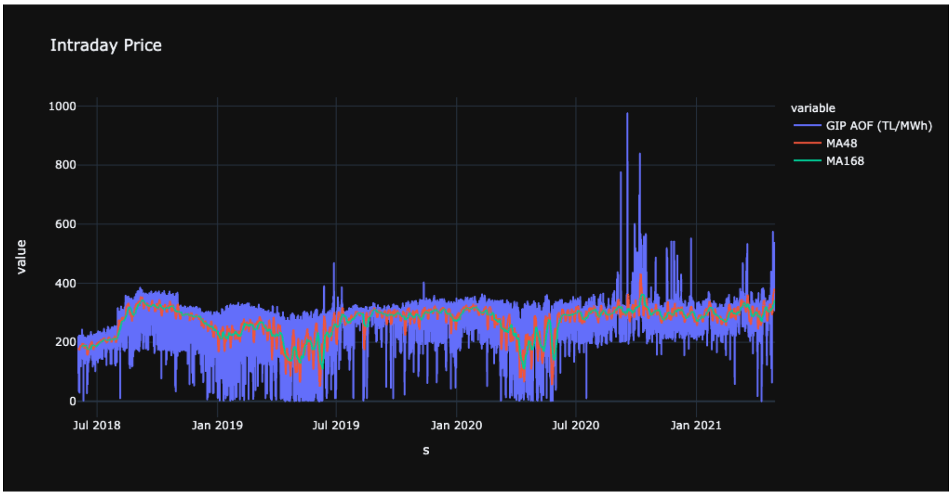


Figure 5.6 : Trend of data with moving averages for post 2018.

Due to the stated situations, we had to accurately detect the anomalies in our data and replace them with a good method while forecasting the price in the intraday market. In addition, since our anomaly data between 2016 and 2018 could not be detected as anomaly due to the very high price increase after 2018, we divided our data into two parts. We made anomaly detection by evaluating our data between 2016-2018 and our data after 2018 separately. Using the advantages provided by Facebook's Prophet library for anomaly detection in time series, we performed anomaly detection as shown below.

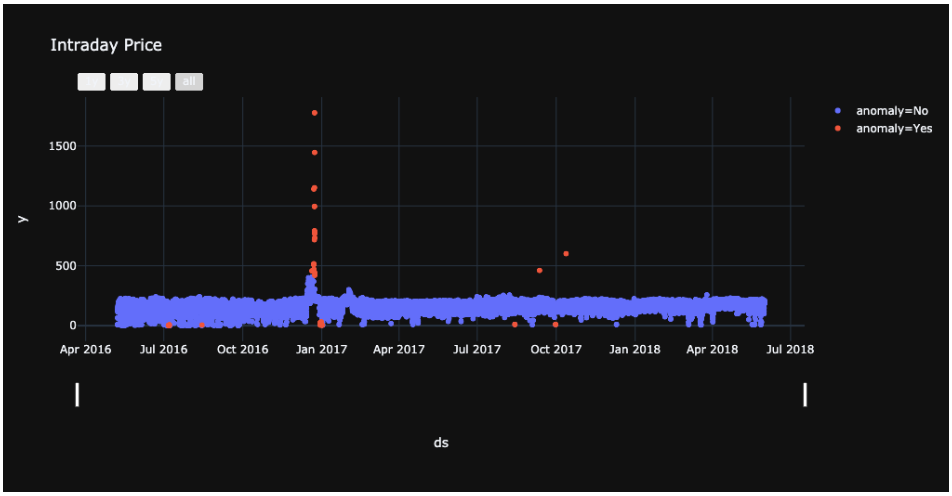


Figure 5.7 : Anomaly detection between 2016-2018.

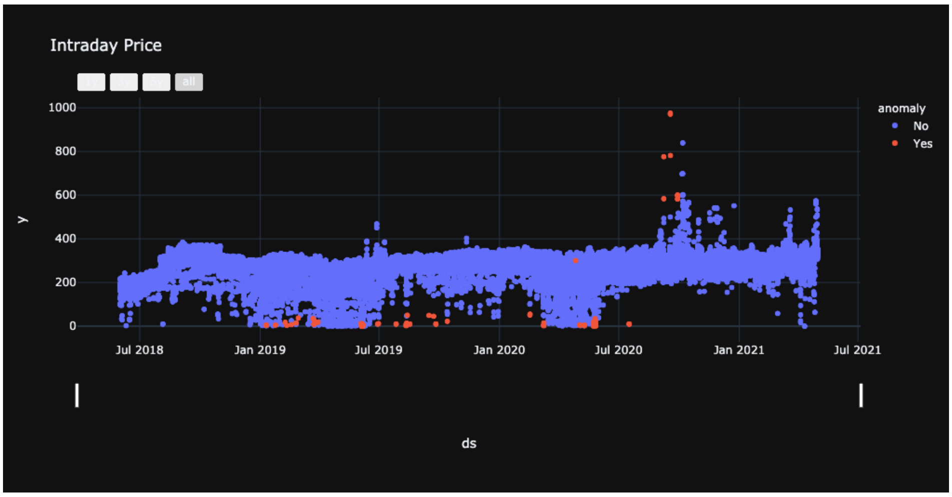


Figure 5.8 : Anomaly detection between 2018-2021.

Since the Prophet library is actually used to make predictions in time series, we also performed anomaly detection and also estimated the price for each hour with this library. We changed the data we detected as anomaly with the predictions (yhat) made by Prophet and fed the model. As a result, we see the real values of some data that we detected as anomaly in tabular form and the values that have been changed as follows.

Table 5.5 : Some anomaly data seen in tabular form and changed Prophet prediction values (yhat).

ds	y	yhat	yhat_lower	yhat_upper	error	uncertainity	anomaly
2016-07-06 09:00:00	2.54	172.086566	124.180957	224.710491	-169.546566	100.529534	Yes
2016-07-07 08:00:00	1.17	154.318013	102.875615	204.043760	-153.148013	101.168145	Yes
2016-07-07 09:00:00	1.01	174.838703	122.913332	221.424596	-173.828703	98.511264	Yes
2016-07-07 10:00:00	7.45	183.000881	131.705859	231.377286	-175.550881	99.671427	Yes
2016-08-14 09:00:00	5.08	164.061940	111.578918	213.271104	-158.981940	101.692187	Yes
2016-12-20 11:00:00	457.75	269.227981	219.995455	318.397911	188.522019	98.402456	Yes
2016-12-22 09:00:00	458.83	268.447267	215.144096	318.673632	190.382733	103.529536	Yes
2016-12-22 10:00:00	461.98	276.715175	229.085092	329.291313	185.264825	100.206221	Yes
2016-12-22 11:00:00	510.86	275.709238	222.400840	325.925620	235.150762	103.524780	Yes
2016-12-22 13:00:00	509.81	271.069659	222.394962	319.264250	238.740341	96.869287	Yes
2016-12-22 14:00:00	514.01	270.912024	220.090386	322.614338	243.097976	102.523953	Yes
2016-12-22 15:00:00	515.04	268.477812	220.240327	318.111483	246.562188	97.871156	Yes
2016-12-22 16:00:00	517.10	262.517919	213.594632	311.985720	254.582081	98.391088	Yes
2016-12-22 17:00:00	1141.38	255.615866	199.364320	309.259625	885.764134	109.895305	Yes
2016-12-22 18:00:00	511.02	251.422467	206.743222	301.363638	259.597533	94.620416	Yes

5.5 Advanced Models

After the selection of the best variables described in the previous sections, anomaly detection and finalization of the data, 80% of our data was splitted for training and 20% for testing. In other words, our models were trained for the first 34.754 hours and tested using the last 8.562 hours. Different methods of deep learning were used in this forecasting phase. Since our intraday price forecasting problem here is time series, we tried the deep learning methods that work best in this subject.

In the intraday electricity market, it is not meaningful to make an hourly forecast because, for example, at 14:30, the intraday electricity price offer cannot be given for 15:00. For this reason, all the models we set up tried to predict the intraday electricity price in 2 hours. While LSTM, BiLSTM, GRU, CNN, CNN-LSTM hybrid models are used in this forecasting part, mean absolute error (MAE) is preferred in comparison and interpretation of these models. For these 5 models, 168 hours are looked back as the lagged time parameter in the data.

5.5.1 Long Short-Term Memory (LSTM)

In our first model, LSTM was fit using the following code example.

```
model = tf.keras.models.Sequential()

model.add(tf.keras.layers.LSTM(64,input_shape = (lag, n_features)))
model.add(tf.keras.layers.Dense(64, activation = 'relu'))
model.add(tf.keras.layers.Dense(32, activation = 'relu'))
model.add(tf.keras.layers.Dense(16, activation = 'relu'))
model.add(tf.keras.layers.Dense(8, activation = 'relu'))
model.add(tf.keras.layers.Dense(4, activation = 'relu'))
model.add(tf.keras.layers.Dense(period))
model.compile(optimizer = 'adam', metrics = ['mae'])
history = model.fit(X_train,y_train, epochs = 10,  validation_data=(X_test, y_test))
```

Figure 5.9 : Code for LSTM network.

In the summary of the LSTM architecture in this code sample, we see the number of layers in each layer and the total number of parameters optimized in the model. Just below, we see in the output obtained via TensorBoard how the layers in this architecture are connected to each other in order.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 64)                19200
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_3 (Dense)              (None, 32)                2080
_________________________________________________________________
dense_4 (Dense)              (None, 16)                528
_________________________________________________________________
dense_5 (Dense)              (None, 8)                 136
_________________________________________________________________
dense_6 (Dense)              (None, 4)                 36
_________________________________________________________________
dense_7 (Dense)              (None, 1)                 5
=================================================================
Total params: 26,145
Trainable params: 26,145
Non-trainable params: 0
```

Figure 5.10 : LSTM Model Summary.

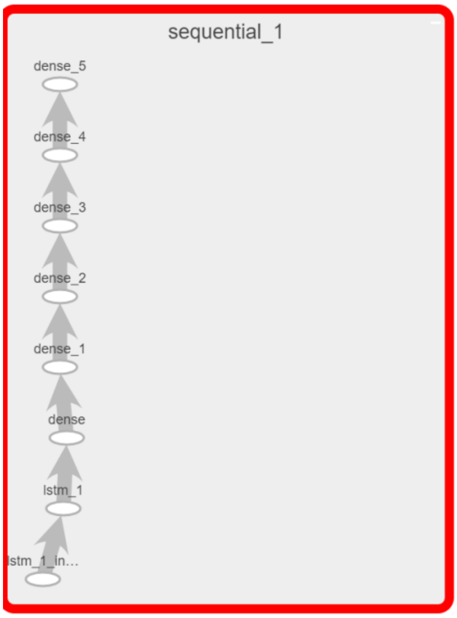


Figure 5.11 : LSTM Architecture.

For MAE, which is our error metric to be used in the comparison of all models, a value of 23.90 was obtained on the test data in the LSTM model. The prediction made by our model for the last 72 hours in our normalized data is given below.

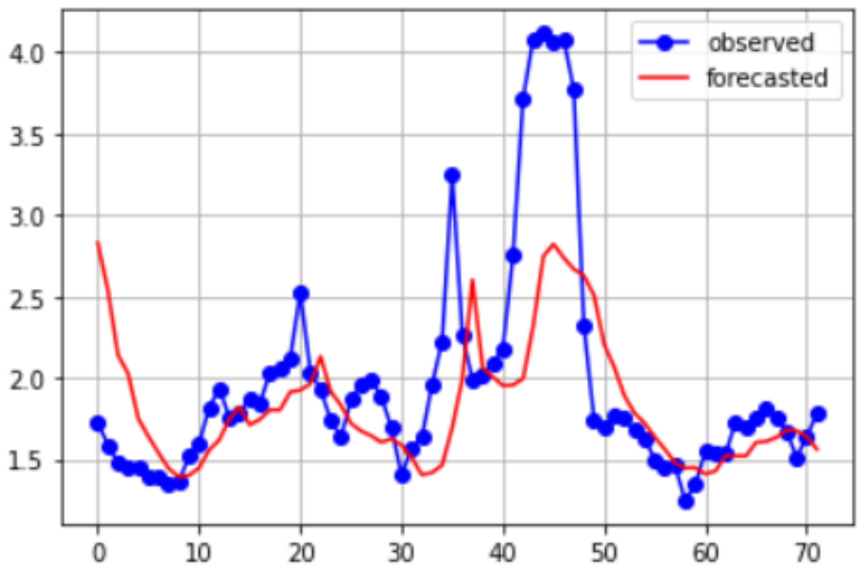


Figure 5.12 : Normalized observed-forecast graph of the LSTM network for the last 72 hours.

5.5.2 Bidirectional LSTM (BiLSTM) and Gated Recurrent Unit (GRU)

Our second and third models, BiLSTM and GRU, are fitted using the code example below.

```
# Create BiLSTM model
def create_bilstm(units):
    model = Sequential()
    # Input layer
    model.add(Bidirectional(
              LSTM(units = units, return_sequences=True),
              input_shape=(X_train.shape[1], X_train.shape[2])))
    # Hidden layer
    model.add(Bidirectional(LSTM(units = units)))
    model.add(Dense(1))
    #Compile model
    model.compile(optimizer="adam",loss="mae")
    return model
model_bilstm = create_bilstm(64)
```

```
# Create GRU model
def create_gru(units):
    model = Sequential()
    # Input layer
    model.add(GRU (units = units, return_sequences = True,
    input_shape = [X_train.shape[1], X_train.shape[2]]))
    model.add(Dropout(0.2))
    # Hidden layer
    model.add(GRU(units = units))
    model.add(Dropout(0.2))
    model.add(Dense(units = 1)) |
    #Compile model
    model.compile(optimizer="adam",loss="mae")
    return model
model_gru = create_gru(64)
```

Figure 5.13 : BiLSTM and GRU Network Code.

```python
def fit_model(model):
    early_stop = keras.callbacks.EarlyStopping(monitor = "val_loss",
                                               patience = 10)
    history = model.fit(X_train, y_train, epochs = 50,
                        validation_split = 0.2,
                        batch_size = 16, shuffle = False,
                        callbacks = [early_stop])
    return history
history_gru = fit_model(model_gru)
history_bilstm = fit_model(model_bilstm)
```

Figure 5.14 : Fitting BiLSTM and GRU networks.

In the summary of the BiLSTM architecture in these codes, we see the number of layers in each layer and the total number of parameters optimized in the model. Just below, we see in the output obtained via TensorBoard how the layers in this architecture are connected to each other in order.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
bidirectional (Bidirectional (None, 168, 128)          38400
_________________________________________________________________
bidirectional_1 (Bidirection (None, 128)               98816
_________________________________________________________________
dense_8 (Dense)              (None, 1)                 129
=================================================================
Total params: 137,345
Trainable params: 137,345
Non-trainable params: 0
```

Figure 5.235 : BiLSTM Model summary.

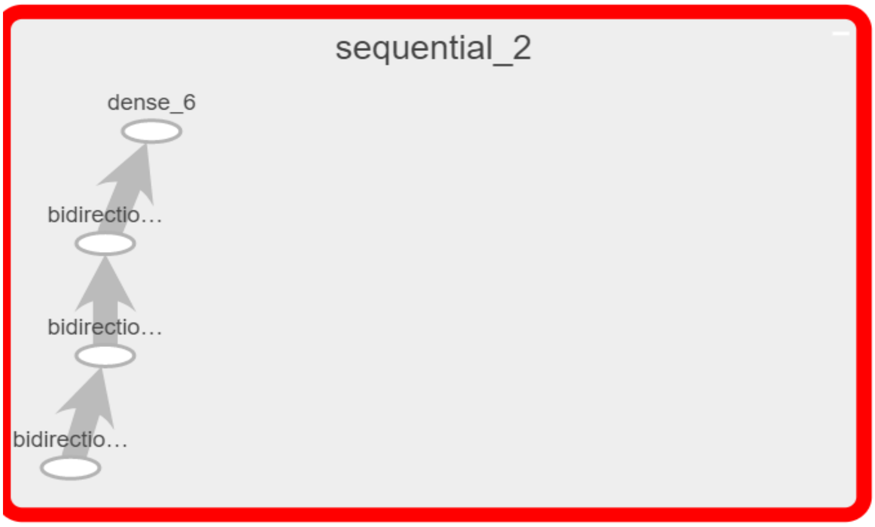


Figure 5.16 : BiLSTM Architecture.

For MAE, which is our error metric to be used in the comparison of all models, a value of 24,138 was obtained on the test data in the BiLSTM model. The prediction made by our model for the last 72 hours in our normalized data is given below.

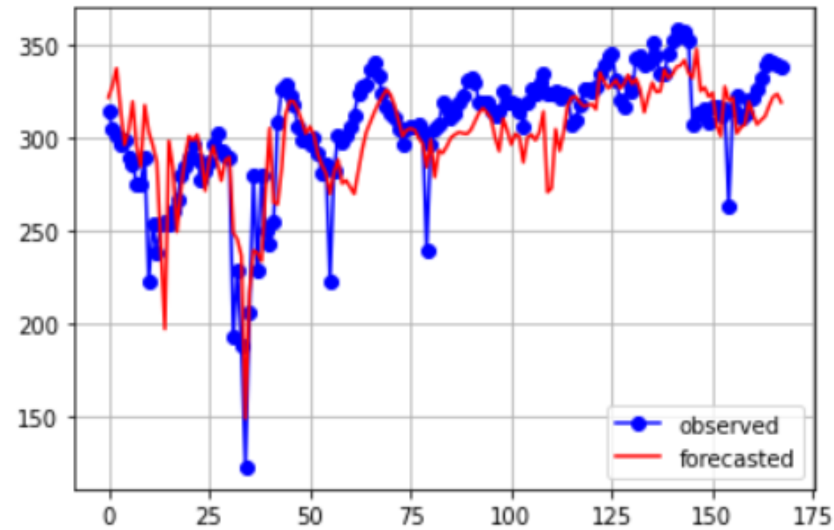


Figure 5.17 : Normalized observed-forecast graph of the BiLSTM network for the last 72 hours.

In the summary of the GRU architecture in these codes, we see the number of layers in each layer and the total number of parameters optimized in the model. Just below, we see in the output obtained via TensorBoard how the layers in this architecture are connected to each other in order.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
gru (GRU)                    (None, 168, 64)           14592
_________________________________________________________________
dropout (Dropout)            (None, 168, 64)           0
_________________________________________________________________
gru_1 (GRU)                  (None, 64)                24960
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense (Dense)                (None, 1)                 65
=================================================================
Total params: 39,617
Trainable params: 39,617
Non-trainable params: 0
```

Figure 5.18 : GRU Model Summary.

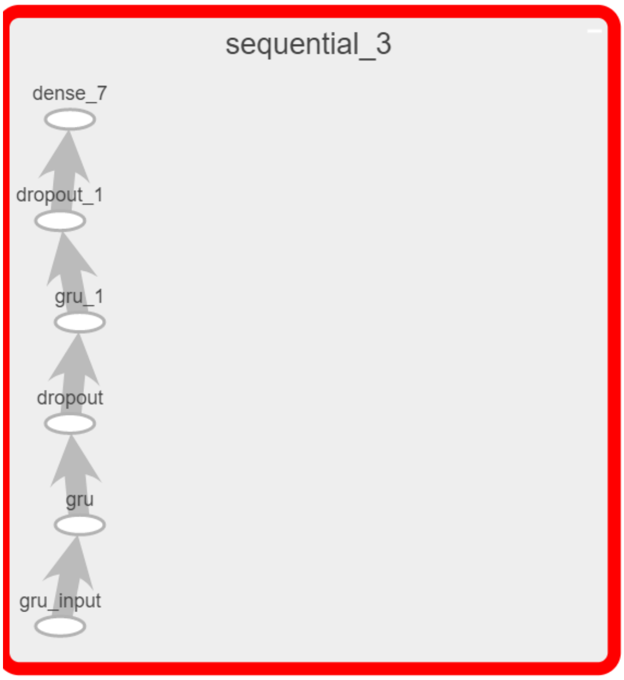


Figure 5.19 : GRU Architecture.

For MAE, which is our error metric to be used in the comparison of all models, a value of 28,43 was obtained on the test data in the GRU model. The prediction made by our model for the last 72 hours in our normalized data is given below.

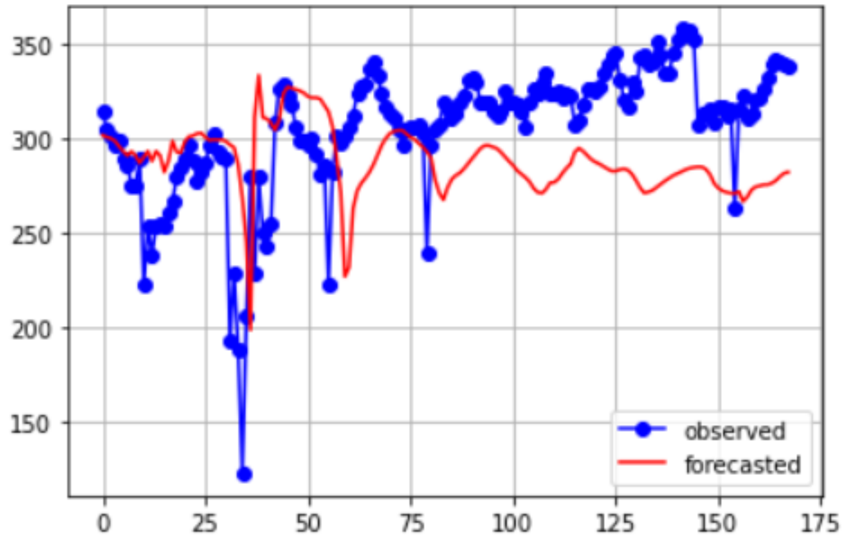


Figure 5.20 : Normalized observed-forecast graph of the GRU network for the last 72 hours.

5.5.3 Convolutional Neural Network (CNN)

In our fourth model, CNN, our model was fit using the following code example.

```
model_cnn = Sequential()

model_cnn.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(lag,n_features)))
model_cnn.add(MaxPooling1D(pool_size=2))
model_cnn.add(Flatten())
model_cnn.add(Dense(50, activation='relu'))
model_cnn.add(Dense(period))
model_cnn.compile(optimizer = 'adam',loss = 'mae', metrics = ['mae'])
history = model_cnn.fit(X_train,y_train, epochs = 10,  validation_data=(X_test, y_test))
```

Figure 5.21 : Code for CNN network.

In the summary of the CNN architecture, we see the number of layers in each layer and the total number of parameters optimized in the model. Underneath, we see the output of the architecture of the network, which is again received via TensorBoard.

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 167, 64)	1344
max_pooling1d (MaxPooling1D)	(None, 83, 64)	0
flatten (Flatten)	(None, 5312)	0
dense_1 (Dense)	(None, 50)	265650
dense_2 (Dense)	(None, 1)	51

```
Total params: 267,045
Trainable params: 267,045
Non-trainable params: 0
```

Figure 5.22 : CNN Model Summary.

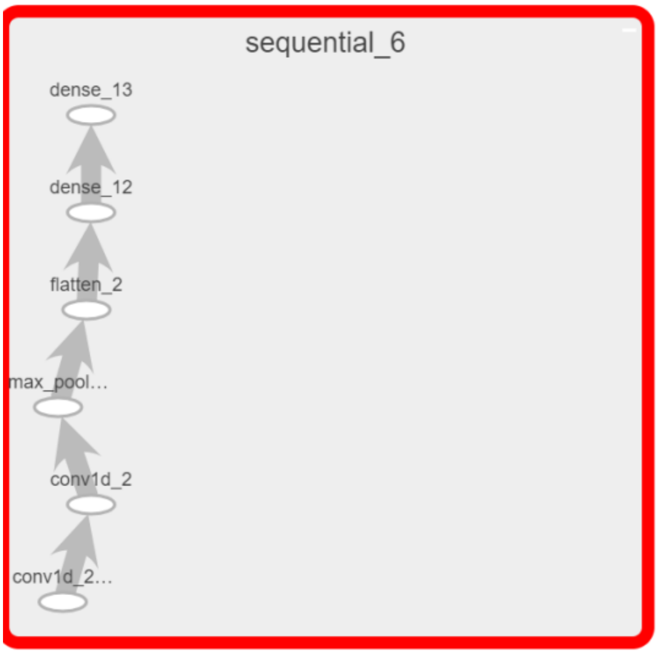


Figure 5.23 : CNN Architecture.

For MAE, which is our error metric to be used in the comparison of all models, a value of 23.90 was obtained on the test data in the CNN model. The prediction made by our model for the last 72 hours in our normalized data is given below.

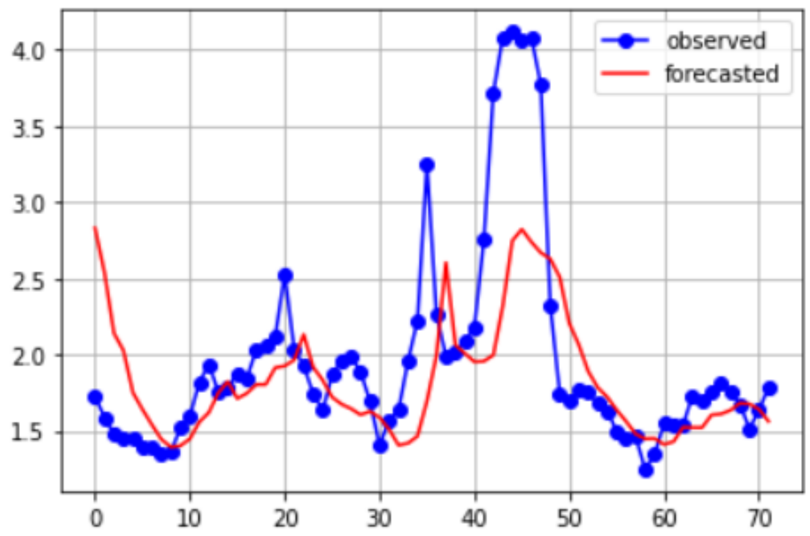


Figure 5.24 : Normalized observed-forecast graph of the CNN network for the last 72 hours.

5.5.4 Hybrid Model: CNN-LSTM

Our model was fit using the following code example in our last and only hybrid model, CNN-LSTM.

```
model_cnn_lstm = Sequential()
model_cnn_lstm.add(TimeDistributed(Conv1D(filters=16, kernel_size=1, activation='relu'), input_shape=(None, X_train.shape[2], X_t
model_cnn_lstm.add(TimeDistributed(MaxPooling1D(pool_size=2)))
model_cnn_lstm.add(TimeDistributed(Flatten()))
model_cnn_lstm.add(LSTM(50, activation='relu'))
model_cnn_lstm.add(Dense(32, activation = 'relu'))
model_cnn_lstm.add(Dropout(0.1))
model_cnn_lstm.add(Dense(1))
model_cnn_lstm.summary()
model_cnn_lstm.compile(optimizer = 'adam',loss = 'mae', metrics = ['mae'])
history = model_cnn_lstm.fit(X_train,y_train, epochs = 10,  validation_data=(X_test, y_test))
```

Figure 5.25 : Code for CNN-LSTM hybrid network.

In the summary of the CNN-LSTM architecture, we see the number of layers in each layer and the total number of parameters optimized in the model. Underneath, we see the output of the architecture of the network, which is again received via TensorBoard.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
time_distributed (TimeDistri (None, None, 84, 16)      176
_________________________________________________________________
time_distributed_1 (TimeDist (None, None, 42, 16)      0
_________________________________________________________________
time_distributed_2 (TimeDist (None, None, 672)         0
_________________________________________________________________
lstm (LSTM)                  (None, 50)                144600
_________________________________________________________________
dense_3 (Dense)              (None, 32)                1632
_________________________________________________________________
dropout_2 (Dropout)          (None, 32)                0
_________________________________________________________________
dense_4 (Dense)              (None, 1)                 33
=================================================================
Total params: 146,441
Trainable params: 146,441
Non-trainable params: 0
```

Figure 5.26 : CNN-LSTM Model Summary.

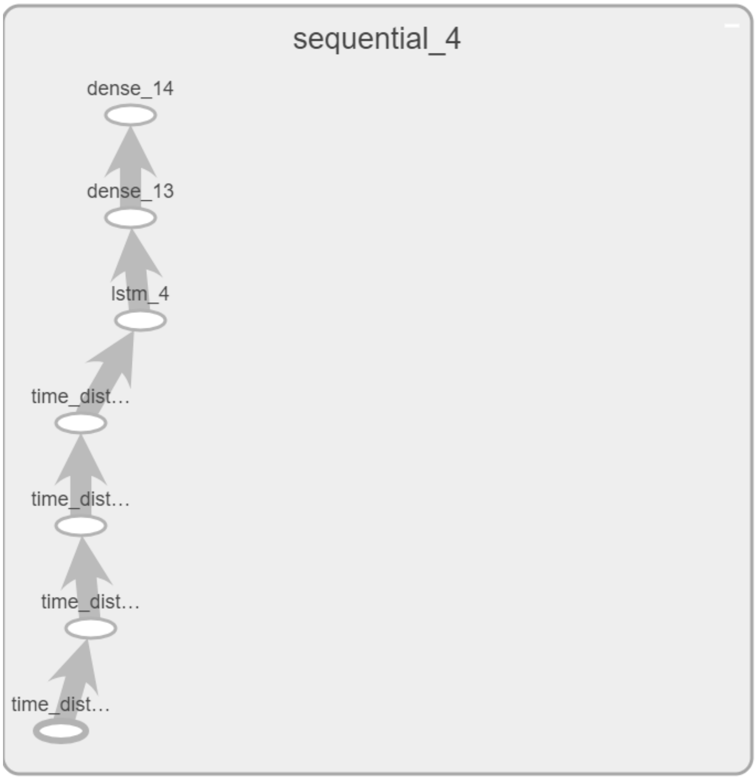


Figure 5.27 : CNN-LSTM Architecture.

For MAE, which is our error metric to be used in the comparison of all models, a value of 22.71 was obtained on the test data in the CNN-LSTM hybrid model. The prediction made by our model for the last 72 hours in our normalized data is given below.

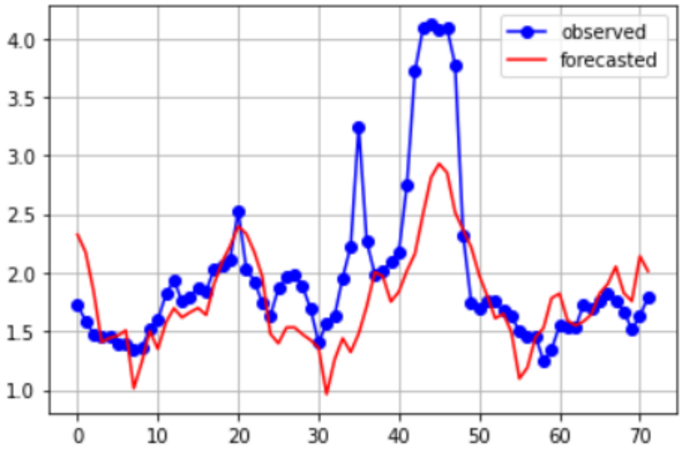


Figure 5.28 : Normalized observed-forecast graph of the CNN-LSTM network for the last 72 hours.

No hyperparameter tuning work has been done up to this stage. When the 5 models used above were compared, it was deemed appropriate to perform hyperparameter tuning on the CNN-LSTM model, which had the lowest MAE result. Here, the combination of hyperparameters that gives the lowest loss on the test data has been tried to be determined. The number of neurons in the dense layer, the dropout rate after the dense layer, the optimizer of the model, the number of epochs and the lagged data were studied by training 72 different combination models.

Trial ID	Show Metrics	num_units	dropout	optimizer	epoch	lag	MAE
d07200f384b5a...	☐	32.000	0.20000	adam	10.000	168.00	0.26909
cf6bba121bcd3...	☐	16.000	0.20000	adam	10.000	168.00	0.27271
a06095109349a...	☐	32.000	0.20000	sgd	20.000	168.00	0.27288
dea079e6a662a...	☐	32.000	0.10000	adam	20.000	168.00	0.27410
2258cb7827ab0...	☐	8.0000	0.20000	sgd	20.000	168.00	0.27482
edf59c94b1fff6...	☐	8.0000	0.10000	adam	10.000	48.000	0.27486
0256ebb3c7ee4...	☐	16.000	0.10000	adam	10.000	168.00	0.27571
947d678b9ef04...	☐	8.0000	0.10000	sgd	20.000	168.00	0.27701
46b03b6f75bf0...	☐	32.000	0.10000	adam	10.000	48.000	0.27765
ff71fd42696759...	☐	32.000	0.20000	sgd	10.000	168.00	0.27866
e30629caa509d...	☐	32.000	0.10000	sgd	10.000	168.00	0.28030
2a7d4a1173aa9...	☐	32.000	0.20000	adam	20.000	168.00	0.28180
bfe101dc5d412...	☐	32.000	0.10000	adam	10.000	168.00	0.28348
766b33cbecba9...	☐	16.000	0.10000	sgd	20.000	48.000	0.28475
d83815391bb70...	☐	8.0000	0.10000	adam	20.000	48.000	0.28486
8ad466352e2ee...	☐	8.0000	0.10000	adam	20.000	168.00	0.28490
3956b260329df...	☐	32.000	0.20000	sgd	20.000	24.000	0.28597

Figure 5.29 : Different combinations of hyperparameters giving the lowest MAE.

As can be seen in the result table that gives the lowest error on the normalized data, it is found that 32 neurons for the dense layer, 0.2 for the dropout rate, "adam" for the optimizer, 10 for the epoch, and 168 for the lagged data. The most important point to note in hyperparameter optimization was the use of 168 hours ago as lagged data in all of the first 5 combinations with the best results. The table that gives the other best results is given above.

All the combinations tried in the output via TensorBoard are given below with their links. Here, the connection path for the combination with the lowest MAE result is shown in green.

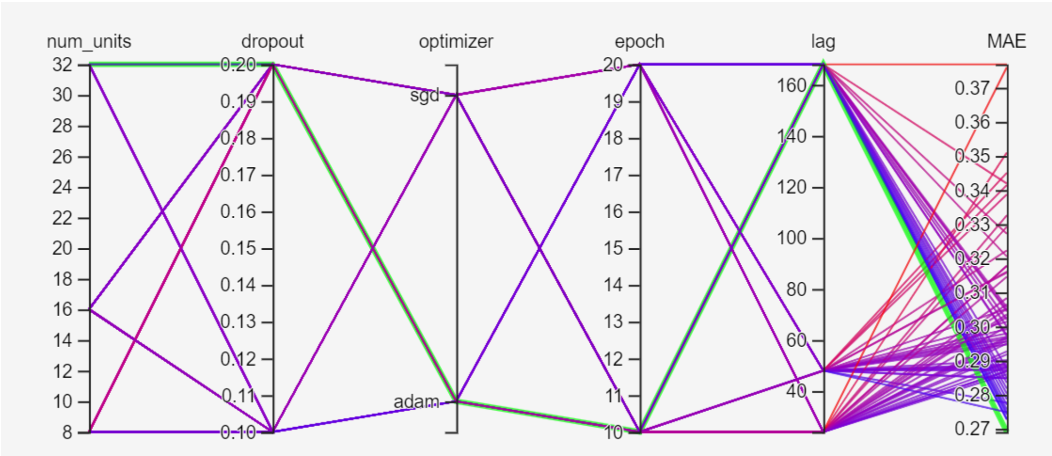


Figure 5.30 : TensorBoard output showing connection paths.

As a final step, our model was trained for the last time in order to find the lowest MAE in the test data with these best tuned parameters. As a result, MAE values as low as 20,89 were reached. In addition the prediction made by our best model for the last 72 hours in our normalized data is given below.

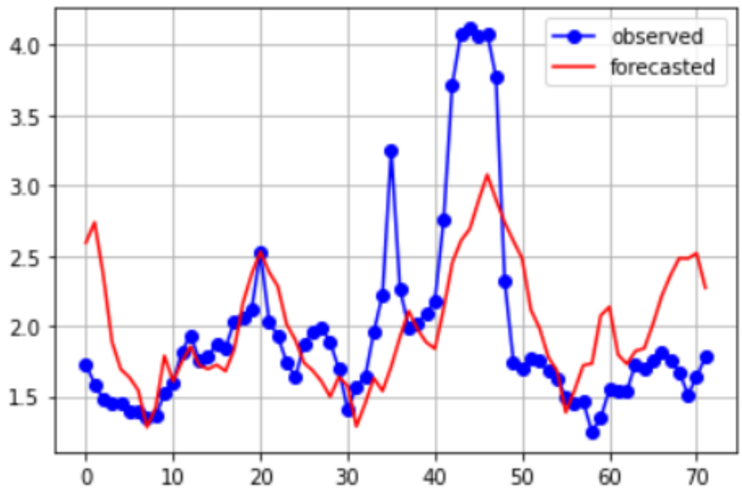


Figure 5.31 : Normalized observed-forecast graph of the our best CNN-LSTM network for the last 72 hours.

6. CONCLUSION

In the first part of our study, the electricity markets in Turkey, the increasing importance of electricity, and the main algorithms used in the intraday electricity market, which is our focus, and related literature research are mentioned. In the second part of our study, firstly the intraday electricity market, other electricity markets, energy production, and consumption data in Turkey were analyzed in detail. As a result of these analyzes, it has been predicted that there may be anomalies in the intraday electricity market data that may adversely affect our prediction results. At this stage, data manipulation was deemed necessary. The detection of these anomalies was made by splitting the data into two parts as 2016-2018 and after 2018, due to the significant increase in the dollar exchange rate in Turkey after 2018. After splitting the data, anomalies in the intraday electricity market clearing price were detected using Facebook's Prophet library, and anomaly data were filled with time-series predictions made with the help of this library. Afterward, simple forecasting methods such as moving averages and exponential smoothing were used by using only the intraday electricity market clearing price to form a basis for our study at a level that can predict the next 1 hour in the market. However, since the actual forecast to be made in the intraday electricity market should be 2 hours later, we have detailed our analyzes from this stage. Five different feature selection methods were used on the data whose anomalies were detected and corrected. 4 of these 5 methods predicted the intraday electricity market clearing price, using the same subset of features (PTF (TL/MWh), PTF (h-1), GIP (h-1), PTF (h-2), GIP (h-2), PTF (h-24), GIP (h-24), PTF (h-168), and GIP (h-168)) were found to be significant variables for the model, so the prediction algorithms were continued with these 9 independent variables. In the next step, our data is divided into two parts, the first 80% train and the remaining 20% test, with a lag of 168 hours. Then, our data was trained on the train data with the LSTM, BiLSTM, GRU, and CNN-LSTM architectures of artificial neural networks, and the results were evaluated on the test data. While the MAE error metric was used for comparison, the CNN-LSTM hybrid model gave the best result from these 4 different architectures with an MAE value of 22.71. After this stage, since CNN-LSTM architecture gave the best results, hyperparameter tuning was applied to some parameters using Tensorboard to further improve this model and the best parameters were determined. Finally, our data was retrained with the best parameters by using the CNN-LSTM model and an

MAE of 20.89 was obtained in the test data. Our study stands out by using a hybrid deep learning model for the first time in the Turkish intraday electricity market and demonstrating its superiority over other deep learning models.

REFERENCES

Acaravci, A., & Yıldız, T. (2018). Türkiye'nin enerji bağımlılığı. *Uluslararası Ekonomi Ve Yenilik Dergisi, 4*(2), 137-152. doi:10.20979/ueyd.457898

Alexandropoulos, S. N., Kotsiantis, S. B., & Vrahatis, M. N. (2019). Data preprocessing in predictive data mining. *The Knowledge Engineering Review, 34*. doi:10.1017/s026988891800036x

Andrade, J., Filipe, J., Reis, M., & Bessa, R. (2017). Probabilistic price forecasting for day-ahead and intraday markets: Beyond the statistical model. *Sustainability, 9*(11), 1990. doi:10.3390/su9111990

Bilirgen, Ü. (2020). *Enerji: Sektörel Bakış* (pp. 3-4, Rep.). KPMG.

Biswas, S., Islam, M. J., Huang, Y., & Rajan, H. (2019). Boa meets Python: A Boa dataset of data Science software in Python language. *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*. doi:10.1109/msr.2019.00086

Conejo, A., Plazas, M., Espinola, R., & Molina, A. (2005). Day-Ahead electricity price forecasting using the wavelet transform and ARIMA Models. *IEEE Transactions on Power Systems, 20*(2), 1035-1042. doi:10.1109/tpwrs.2005.846054

Contreras, J., Espinola, R., Nogales, F. J., & Conejo, A. J. (2002). ARIMA models to predict Next-Day electricity prices. *IEEE Power Engineering Review, 22*(9), 57-57. doi:10.1109/mper.2002.4312577

Cordón, I., Luengo, J., García, S., Herrera, F., & Charte, F. (2019). Smartdata: Data preprocessing to Achieve smart data in R. *Neurocomputing, 360*, 1-13. doi:10.1016/j.neucom.2019.06.006

Dillig, M., Jung, M., & Karl, J. (2016). The impact of renewables on electricity prices in Germany – an estimation based on Historic spot prices in the Years 2011–2013. *Renewable and Sustainable Energy Reviews, 57*, 7-15. doi:10.1016/j.rser.2015.12.003

Furió, D. (2011). A survey on the Spanish electricity intraday market. Estudios de economía aplicada, 29(2), 1-19.

Ganegedara, T. (2018). Natural language processing with Tensorflow: Teach language to machines using Python's deep learning library. In *Natural Language Processing with TensorFlow: Teach language to machines using Python's deep learning library* (pp. 28-29). Birmingham: Packt Publishing.

García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: Methods and prospects. *Big Data Analytics, 1*(1). doi:10.1186/s41044-016-0014-0

Harris, C. R., Millman, K. J., Van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., . . . Oliphant, T. E. (2020). Array programming with numpy. *Nature, 585*(7825), 357-362. doi:10.1038/s41586-020-2649-2

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. doi:10.1162/neco.1997.9.8.1735

Janke, T., & Steinke, F. (2019). Forecasting the price distribution of continuous intraday electricity trading. *Energies, 12*(22), 4262. doi:10.3390/en12224262

Keles, D., Scelle, J., Paraschiv, F., & Fichtner, W. (2016). Extended forecast methods FOR day-ahead electricity spot prices applying artificial neural networks. *Applied Energy, 162*, 218-230. doi:10.1016/j.apenergy.2015.09.087

Kim, H. Y., & Won, C. H. (2018). Forecasting the volatility of stock PRICE INDEX: A hybrid model integrating LSTM with multiple GARCH-type models. *Expert Systems with Applications, 103*, 25-37. doi:10.1016/j.eswa.2018.03.002

Kotsiantis, S. B., Kanellopoulos, D., & Pintelas, P. E. (2006). Data preprocessing for supervised leaning. International Journal of Computer Science, 1(2), 111-117.

Kuo, P., & Huang, C. (2018). An electricity price forecasting model by hybrid structured deep neural networks. *Sustainability, 10*(4), 1280. doi:10.3390/su10041280

Lago, J., De Ridder, F., & De Schutter, B. (2018). Forecasting spot electricity prices: Deep learning approaches and empirical comparison of traditional algorithms. *Applied Energy, 221*, 386-405. doi:10.1016/j.apenergy.2018.02.069

Lu, W. (2016). Electricity Consumption and Economic Growth: Evidence from 17 Taiwanese Industries. *Sustainability, 9*(1), 50. doi:10.3390/su9010050

Luengo, J. (2020). Big data preprocessing: Enabling smart data. In *Big data preprocessing: Enabling smart data* (pp. 121-122). Springer Nature Switzerland AG.

Manaswi, N. K. (2018). Understanding and working with keras. *Deep Learning with Applications Using Python,* 31-43. doi:10.1007/978-1-4842-3516-4_2

Monteiro, C., Ramirez-Rosado, I., Fernandez-Jimenez, L., & Conde, P. (2016). Short-term price forecasting models based on artificial neural networks for intraday sessions in the iberian electricity market. *Energies, 9*(9), 721. doi:10.3390/en9090721

Narajewski, M., & Ziel, F. (2019). Estimation and simulation of the transaction arrival process in intraday electricity markets. *Energies, 12*(23), 4518. doi:10.3390/en12234518

Oksuz, I., & Ugurlu, U. (2019). Neural network based model comparison for Intraday electricity price forecasting. *Energies, 12*(23), 4557. doi:10.3390/en12234557

Owusu, P. A., & Asumadu-Sarkodie, S. (2016). A review of renewable energy sources, sustainability issues and climate change mitigation. *Cogent Engineering, 3*(1), 1167990. doi:10.1080/23311916.2016.1167990

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.

Peng, L., Liu, S., Liu, R., & Wang, L. (2018). Effective long short-term memory with differential evolution algorithm for electricity price prediction. *Energy, 162*, 1301-1314. doi:10.1016/j.energy.2018.05.052

Pousinho, H., Mendes, V., & Catalão, J. (2012). Short-term electricity prices forecasting in a competitive market by a Hybrid PSO–ANFIS approach. *International Journal of Electrical Power & Energy Systems, 39*(1), 29-35. doi:10.1016/j.ijepes.2012.01.001

Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., & Herrera, F. (2017). A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing, 239*, 39-57. doi:10.1016/j.neucom.2017.01.078

Shafie-khah, M., Moghaddam, M. P., & Sheikh-El-Eslami, M. (2011). Price forecasting of day-ahead electricity markets using a hybrid forecast method. *Energy Conversion and Management, 52*(5), 2165-2169. doi:10.1016/j.enconman.2010.10.047

Sharma, V., & Srinivasan, D. (2013). A hybrid intelligent model based on recurrent neural networks and excitable dynamics for price prediction in deregulated electricity market. *Engineering Applications of Artificial Intelligence, 26*(5-6), 1562-1574. doi:10.1016/j.engappai.2012.12.012

Tertzakian, P. (2007). *A thousand barrels a second: The coming oil break point and the challenges facing an energy dependent world*. New York: McGraw-Hill Professional.

Terzic, B. (2013). *Energy Independence and Security: A Reality Check* (pp. 3-4, Rep.). Deloitte University Press.

Uniejewski, B., Marcjasz, G., & Weron, R. (2019). Understanding intraday electricity markets: Variable selection and very short-term price forecasting using lasso. *International Journal of Forecasting, 35*(4), 1533-1547. doi:10.1016/j.ijforecast.2019.02.001

Zhou, T., Han, G., Xu, X., Lin, Z., Han, C., Huang, Y., & Qin, J. (2017). Δ -agree AdaBoost stacked autoencoder for short-term traffic flow forecasting. *Neurocomputing, 247*, 31-38. doi:10.1016/j.neucom.2017.03.049

Ziel, F. (2017). Modeling the impact of wind and solar power forecasting errors on intraday electricity prices. *2017 14th International Conference on the European Energy Market (EEM)*. doi:10.1109/eem.2017.7981900

APPENDICES

APPENDIX A

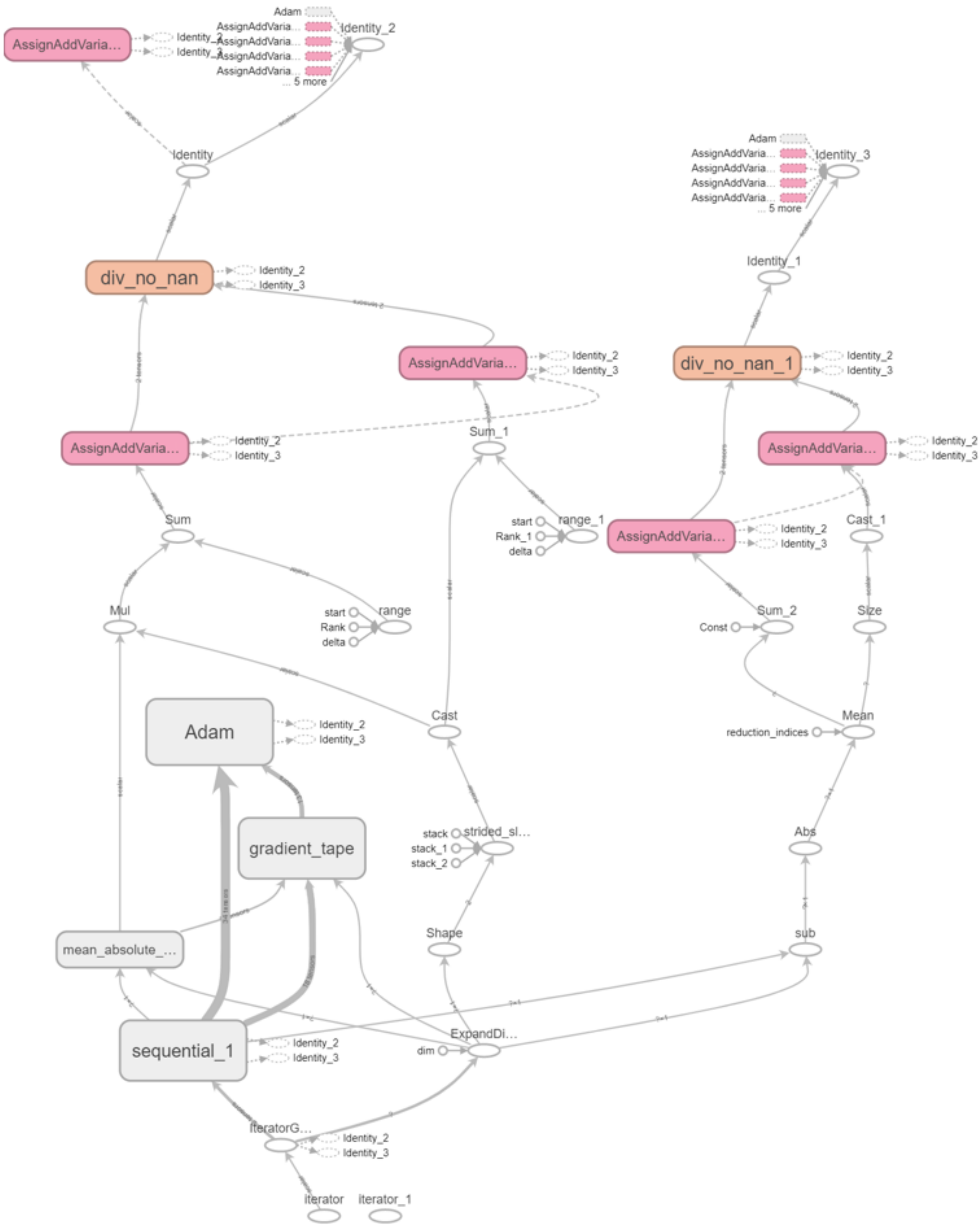


Figure A.1 : Detailed Structure of Long Short-Term Memory.

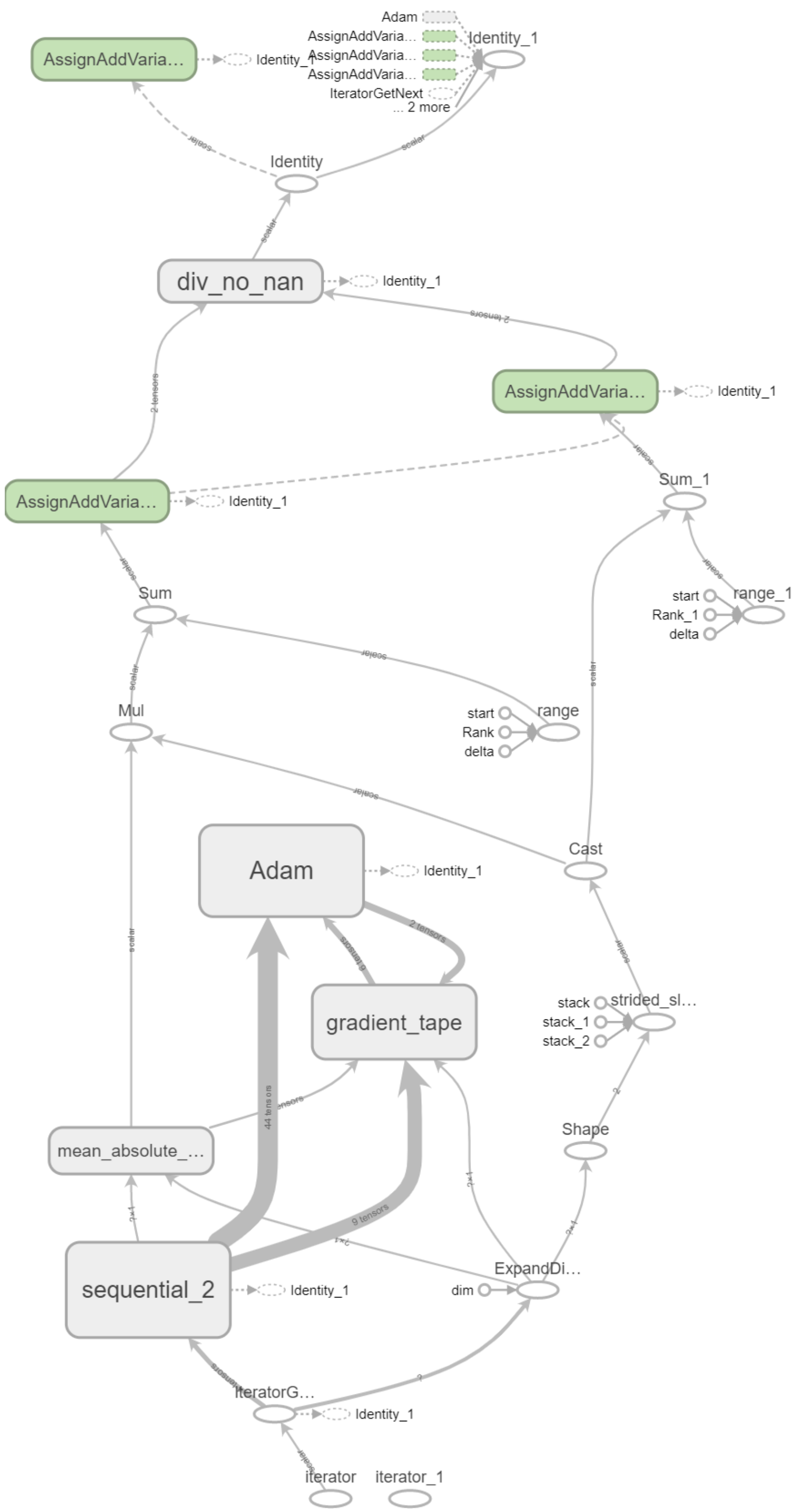


Figure A.2 : Detailed Structure of Bidirectional Long Short-Term Memory (BiLSTM)

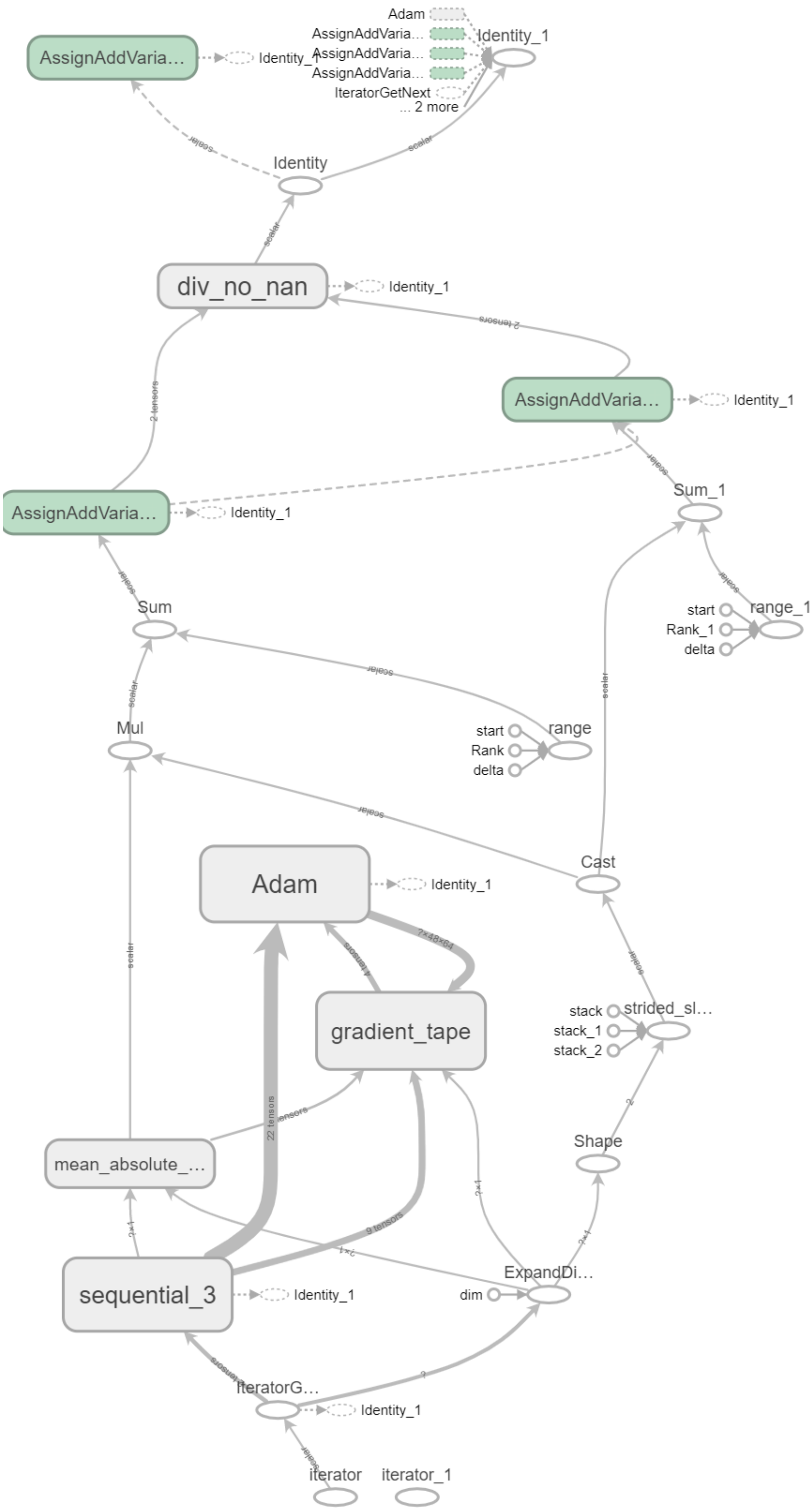


Figure A.3 : Detailed Structure of Gated Recurrent Unit (GRU).

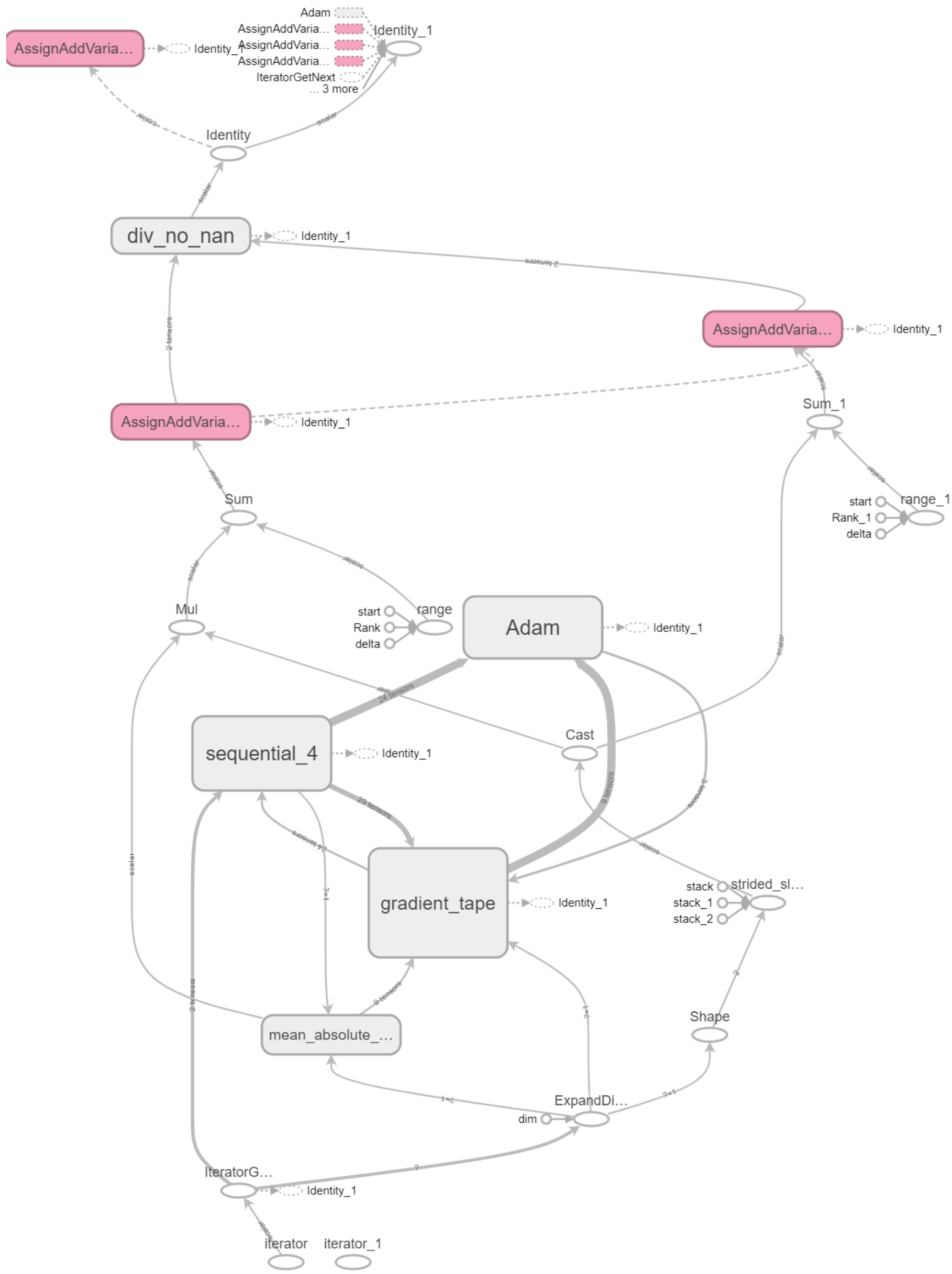


Figure A.4 : Detailed Structure of Convolutional Neural Networks - Long Short-Term Memory

APPENDIX B

In [ ]:

```
import pandas as pd
```

In [ ]:

```
df = pd.read_excel('/content/2016_2018.xlsx',
                   parse_dates={'timestamp' : ['Tarih', 'Saat']})
```

In [ ]:

```
import pandas
import datetime

start = datetime.date(2016, 5, 8)
end = datetime.date(2018, 6, 1)
s = pandas.Series(pandas.date_range(start=start, end=end, freq='H'))
```

In [ ]:

```
df["s"] = s
```

In [ ]:

```
# create moving-averages
df['MA48'] = df['GIP AOF (TL/MWh)'].rolling(48).mean()
df['MA168'] = df['GIP AOF (TL/MWh)'].rolling(168).mean()
# plot
import plotly.express as px
fig = px.line(df, x="s", y=['GIP AOF (TL/MWh)','MA48','MA168',], title='Intraday Pri
fig.show()
```

In [ ]:

```
data = df[["s","GIP AOF (TL/MWh)"]]
```

In [ ]:

```
data = data.reset_index()[['s', 'GIP AOF (TL/MWh)']].rename({'s':'ds',
                                                           'GIP AOF (TL/MWh)':'y'},
                                                          axis='columns')
```

In [ ]:

```
n = len(df)
train = data[:int(0.8*n)]
test = data[int(0.8*n):]
```

In [ ]:

```
model = Prophet(changepoint_range=0.80)
```

In [ ]:

```
model.fit(train)
```

In [ ]:

```
future = model.make_future_dataframe(periods=119, freq='H')
```

In [ ]:

```
future.tail()
```

In [ ]:

```
forecast = model.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

In [ ]:

```
results = pd.concat([data.set_index('ds')['y'], forecast.set_index('ds')[['yhat','yh
                    axis=1)
```

In [ ]:

```
fig1 = model.plot(forecast)
```

In [ ]:

```
comp = model.plot_components(forecast)
```

In [ ]:

```
results['error'] = results['y'] - results['yhat']
```

In [ ]:

```
results['uncertainity'] = results['yhat_upper'] - results['yhat_lower']
```

In [ ]:

```
results[results['error'].abs()>1.5*results['uncertainity']]
```

In [ ]:

```
results['anomaly'] = results.apply(lambda x: 'Yes' if(np.abs(x['error'])>1.5*x['unce
```

In [ ]:

```
fig = px.scatter(results.reset_index(), x='ds', y='y',
                 color='anomaly',title='Intraday Price',template = 'plotly_dark')
fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
                     dict(count=1, label='1y', step='year', stepmode='backward'),
                     dict(count=2, label='3y', step='year', stepmode='backward'),
                     dict(count=3, label='5y', step='year', stepmode='backward'),
                     dict(step='all')
        ])
    )
)
fig.show()
```

Figure B.1 : Anomaly Detection and Filling Codes (Part 1)

APPENDIX C

In [ ]:

```
import pandas as pd
```

In [ ]:

```
df = pd.read_excel('/content/2018_2021.xlsx',
                 parse_dates={'timestamp' : ['Tarih', 'Saat']})
```

In [ ]:

```
import pandas
import datetime

start = datetime.date(2018, 6, 1)
end = datetime.date(2021, 5, 1)
s = pandas.Series(pandas.date_range(start=start, end=end, freq='H'))
```

In [ ]:

```
df["s"] = s
```

In [ ]:

```
data = df[["s","GIP AOF (TL/MWh)"]]
```

In [ ]:

```
data = data.reset_index()[['s', 'GIP AOF (TL/MWh)']].rename({'s':'ds',
                                                          'GIP AOF (TL/MWh)':'y'},
                                                         axis='columns')
```

In [ ]:

```
n = len(data)
train = data[:int(0.8*n)]
test = data[int(0.8*n):]
```

In [ ]:

```
print(train.shape)
print(test.shape)
```

In [ ]:

```
model = Prophet(changepoint_range=0.80)
```

In [ ]:

```
model.fit(train)
```

In [ ]:

```
future = model.make_future_dataframe(periods=119, freq='H')
```

In [ ]:

```
forecast = model.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

In [ ]:

```
results = pd.concat([data.set_index('ds')['y'], forecast.set_index('ds')[['yhat','yh
                     axis=1)
```

In [ ]:

```
fig1 = model.plot(forecast)
```

In [ ]:

```
comp = model.plot_components(forecast)
```

In [ ]:

```
results['error'] = results['y'] - results['yhat']
```

In [ ]:

```
results['uncertainity'] = results['yhat_upper'] - results['yhat_lower']
```

In [ ]:

```
results[results['error'].abs()>1.5*results['uncertainity']]
```

In [ ]:

```
results['anomaly'] = results.apply(lambda x: 'Yes' if(np.abs(x['error'])>1.5*x['unce
```

In [ ]:

```
import plotly.express as px
```

In [ ]:

```
fig = px.scatter(results.reset_index(), x='ds', y='y',
                 color='anomaly',title='Intraday Price',template = 'plotly_dark')
fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
                     dict(count=1, label='1y', step='year', stepmode='backward'),
                     dict(count=2, label='3y', step='year', stepmode='backward'),
                     dict(count=3, label='5y', step='year', stepmode='backward'),
                     dict(step='all')
        ])
    )
)
fig.show()
```

Figure C.1 : Anomaly Detection and Filling Codes (Part 2)

APPENDIX D

In [ ]:

```
import pandas as pd
```

In [ ]:

```
df = pd.read_excel('/content/2018_2021.xlsx',
                 parse_dates={'timestamp' : ['Tarih', 'Saat']})
```

In [ ]:

```
import pandas
import datetime

start = datetime.date(2018, 6, 1)
end = datetime.date(2021, 5, 1)
s = pandas.Series(pandas.date_range(start=start, end=end, freq='H'))
```

In [ ]:

```
df["s"] = s
```

In [ ]:

```
data = df[["s","GIP AOF (TL/MWh)"]]
```

In [ ]:

```
data = data.reset_index()[['s', 'GIP AOF (TL/MWh)']].rename({'s':'ds',
                                                    'GIP AOF (TL/MWh)':'y'},
                                                   axis='columns')
```

In [ ]:

```
n = len(data)
train = data[:int(0.8*n)]
test = data[int(0.8*n):]
```

In [ ]:

```
print(train.shape)
print(test.shape)
```

In [ ]:

```
model = Prophet(changepoint_range=0.80)
```

In [ ]:

```
model.fit(train)
```

In [ ]:

```
future = model.make_future_dataframe(periods=119, freq='H')
```

In [ ]:

```
forecast = model.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

In [ ]:

```
results = pd.concat([data.set_index('ds')['y'], forecast.set_index('ds')[['yhat','yh
                    axis=1)
```

In [ ]:

```
fig1 = model.plot(forecast)
```

In [ ]:

```
comp = model.plot_components(forecast)
```

In [ ]:

```
results['error'] = results['y'] - results['yhat']
```

In [ ]:

```
results['uncertainity'] = results['yhat_upper'] - results['yhat_lower']
```

In [ ]:

```
results[results['error'].abs()>1.5*results['uncertainity']]
```

In [ ]:

```
results['anomaly'] = results.apply(lambda x: 'Yes' if(np.abs(x['error'])>1.5*x['unce
```

In [ ]:

```
import plotly.express as px
```

In [ ]:

```
fig = px.scatter(results.reset_index(), x='ds', y='y',
                 color='anomaly',title='Intraday Price',template = 'plotly_dark')
fig.update_xaxes(
    rangeslider_visible=True,
    rangeselector=dict(
        buttons=list([
                    dict(count=1, label='1y', step='year', stepmode='backward'),
                    dict(count=2, label='3y', step='year', stepmode='backward'),
                    dict(count=3, label='5y', step='year', stepmode='backward'),
                    dict(step='all')
        ])
    )
)
fig.show()
```

In [ ]:

```
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
import os
import pandas as pd
```

In [ ]:

```
df = pd.read_excel('7Mayis_without_anomaly.xlsx',
                   parse_dates={'dt' : ['Tarih', 'Saat']},
                    index_col='dt')
```

In [ ]:

```
df = df[["GIP AOF (TL/MWh)",'PTF (TL/MWh)', 'PTF (h-1)', 'GIP (h-1)', 'PTF (h-2)', '
         'GIP (h-24)', 'PTF (h-168)', 'GIP (h-168)']]
```

In [ ]:

```
n = len(df)
train_df = df[:int(0.8*n)]
test_df = df[int(0.8*n):]
print(train_df.shape)
print(test_df.shape)

train_mean = train_df.mean()
train_std = train_df.std()

train_df = (train_df - train_mean) / train_std
test_df = (test_df - train_mean) / train_std
```

In [ ]:

```
def make_data(data,lag = 48, offset = 2, period = 1, target_col = 0):
    X = []
    y = []
    for i in range(len(data) - lag - offset):
        X.append(data[i:i+lag,:])
        y.append(data[i+lag + offset - 1,target_col])

    return np.array(X), np.array(y)
```

In [ ]:

```
lag = 168
offset = 2
period = 1
target_col = 0
n_features = train_df.shape[1]

X_train, y_train = make_data(train_df.values, lag = lag, offset = offset, period = p
X_test, y_test = make_data(test_df.values, lag = lag, offset = offset, period = peri
print(X_train.shape)
print(X_train[:2])
print(y_train[:2])
```

LSTM

In [ ]:

```
model = tf.keras.models.Sequential()

model.add(tf.keras.layers.LSTM(64,input_shape = (lag, n_features)))
model.add(tf.keras.layers.Dense(64, activation = 'relu'))
model.add(tf.keras.layers.Dense(32, activation = 'relu'))
model.add(tf.keras.layers.Dense(16, activation = 'relu'))
model.add(tf.keras.layers.Dense(8, activation = 'relu'))
model.add(tf.keras.layers.Dense(4, activation = 'relu'))
model.add(tf.keras.layers.Dense(period))
```

In [ ]:

```
model.summary()
```

In [ ]:

```
model.compile(optimizer = 'adam',loss = 'mae', metrics = ['mae'])
history = model.fit(X_train,y_train, epochs = 10,  validation_data=(X_test, y_test))
```

In [ ]:

```
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.show()
```

In [ ]:

```
ypred = model.predict(X_test)

plt.plot(y_test[-72:], '-ob', label =  'observed')
plt.plot(ypred[-72:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

In [ ]:

```
ypred_old = (ypred * train_std[0]) + train_mean[0]
```

In [ ]:

```
y_test_old = (y_test*train_std[0]) + train_mean[0]
```

In [ ]:

```
mae(ypred_old,y_test_old)
```

In [ ]:

```
ypred = model.predict(X_test)

plt.plot(y_test_old[-168:], '-ob', label =  'observed')
plt.plot(ypred_old[-168:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

BiLSTM - GRU

In [ ]:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import Sequential, layers, callbacks
from tensorflow.keras.layers import Dense, LSTM, Dropout, GRU, Bidirectional,TimeDis
```

In [ ]:

```
# Create BiLSTM model
def create_bilstm(units):
    model = Sequential()
    # Input layer
    model.add(Bidirectional(
              LSTM(units = units, return_sequences=True),
              input_shape=(X_train.shape[1], X_train.shape[2])))
    # Hidden layer
    model.add(Bidirectional(LSTM(units = units)))
    model.add(Dense(1))
    #Compile model
    model.compile(optimizer="adam",loss="mae")
    return model
model_bilstm = create_bilstm(64)
```

In [ ]:

```
# Create GRU model
def create_gru(units):
    model = Sequential()
    # Input layer
    model.add(GRU (units = units, return_sequences = True,
    input_shape = [X_train.shape[1], X_train.shape[2]]))
    model.add(Dropout(0.2))
    # Hidden layer
    model.add(GRU(units = units))
    model.add(Dropout(0.2))
    model.add(Dense(units = 1))
    #Compile model
    model.compile(optimizer="adam",loss="mae")
    return model
model_gru = create_gru(64)
```

In [ ]:

```
def fit_model(model):
    early_stop = keras.callbacks.EarlyStopping(monitor = "val_loss",
                                               patience = 10)
    history = model.fit(X_train, y_train, epochs = 15,
                        validation_split = 0.2,
                        batch_size = 16, shuffle = False,
                        callbacks = [early_stop])
    return history
history_gru = fit_model(model_gru)
history_bilstm = fit_model(model_bilstm)
```

In [ ]:

```
def plot_loss (history, model_name):
    plt.figure(figsize = (10, 6))
    plt.plot(history.history["loss"])
    plt.plot(history.history["val_loss"])
    plt.title("Model Train vs Validation Loss for " + model_name)
    plt.ylabel("Loss")
    plt.xlabel("epoch")
    plt.legend(["Train loss", "Validation loss"], loc="upper right")

plot_loss(history_gru, "GRU")
plot_loss (history_bilstm, "Bidirectional LSTM")
```

In [ ]:

```
# Make prediction
def prediction(model):
    prediction = model.predict(X_test)
    prediction = (prediction * train_std[1]) + train_mean[1]
    return prediction
prediction_gru = prediction(model_gru)
prediction_bilstm = prediction(model_bilstm)
# Plot test data vs prediction
def plot_future(prediction, model_name, y_test_old):
    plt.figure(figsize=(10, 6))
    range_future = len(prediction)
    plt.plot(np.arange(range_future), np.array(y_test_old),
             label="Test data")
    plt.plot(np.arange(range_future),
             np.array(prediction),label="Prediction")
    plt.title("Test data vs prediction for " + model_name)
    plt.legend(loc="upper left")
    plt.xlabel("Time (hour)")
    plt.ylabel("GIP TL")

plot_future(prediction_gru, "GRU", y_test_old)
plot_future(prediction_bilstm, "Bidirectional LSTM", y_test_old)
```

In [ ]:

```
ypred = prediction_gru

plt.plot(y_test_old[-336:-168], '-ob', label =  'observed')
plt.plot(ypred[-336:-168], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

In [ ]:

```
ypred = prediction_bilstm

plt.plot(y_test_old[-336:-168], '-ob', label =  'observed')
plt.plot(ypred[-336:-168], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

In [ ]:

```
mae(prediction_gru,y_test_old)
```

In [ ]:

```
mae(prediction_bilstm,y_test_old)
```

In [ ]:

```
# Make prediction for new data
def prediction(model):
    prediction = model.predict(X_30)
    prediction = (prediction * train_std[1]) + train_mean[1]
    return prediction
prediction_gru = prediction(model_gru)
prediction_bilstm = prediction(model_bilstm)
# Plot history and future
def plot_multi_step(history, prediction1, prediction2):

    plt.figure(figsize=(15, 6))

    range_history = len(history)
    range_future = list(range(range_history, range_history +
                        len(prediction1)))
    plt.plot(np.arange(range_history), np.array(history),
             label='History')
    plt.plot(range_future, np.array(prediction1),
             label='Forecasted for GRU')
    plt.plot(range_future, np.array(prediction2),
             label='Forecasted for BiLSTM')

    plt.legend(loc='upper right')
    plt.xlabel('Time step (Hours)')
    plt.ylabel('GIP TL')

plot_multi_step(new_data, prediction_gru, prediction_bilstm)
```

CNN

In [ ]:

```
from keras.layers.convolutional import Conv1D
from keras.layers.convolutional import MaxPooling1D
from keras.layers import Input, Flatten
```

In [ ]:

```
model_cnn = Sequential()

model_cnn.add(Conv1D(filters=64, kernel_size=2, activation='relu', input_shape=(lag,
model_cnn.add(MaxPooling1D(pool_size=2))
model_cnn.add(Flatten())
model_cnn.add(Dense(50, activation='relu'))
model_cnn.add(Dense(period))
```

In [ ]:

```
model_cnn.summary()
```

In [ ]:

```
model_cnn.compile(optimizer = 'adam',loss = 'mae', metrics = ['mae'])
history = model_cnn.fit(X_train,y_train, epochs = 10,  validation_data=(X_test, y_te
```

In [ ]:

```
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.show()
```

In [ ]:

```
ypred = model.predict(X_test)

plt.plot(y_test[-72:], '-ob', label =  'observed')
plt.plot(ypred[-72:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

In [ ]:

```
ypred_old = (ypred * train_std[0]) + train_mean[0]
```

In [ ]:

```
y_test_old = (y_test*train_std[0]) + train_mean[0]
```

In [ ]:

```
mae(ypred_old,y_test_old)
```

In [ ]:

```
plt.plot(y_test_old[-168:], '-ob', label =  'observed')
plt.plot(ypred_old[-168:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

CNN-LSTM

In [ ]:

```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import Sequential, layers, callbacks
from tensorflow.keras.layers import Dense, LSTM, Dropout, GRU, Bidirectional,TimeDis
```

In [ ]:

```
subsequences = 2
timesteps = X_train.shape[1]//subsequences
X_train = X_train.reshape((X_train.shape[0], subsequences, timesteps , n_features))
X_test = X_test.reshape((X_test.shape[0], subsequences, timesteps, n_features))
```

In [ ]:

```
model_cnn_lstm = Sequential()
model_cnn_lstm.add(TimeDistributed(Conv1D(filters=16, kernel_size=1, activation='rel
model_cnn_lstm.add(TimeDistributed(MaxPooling1D(pool_size=2)))
model_cnn_lstm.add(TimeDistributed(Flatten()))
model_cnn_lstm.add(LSTM(50, activation='relu'))
model_cnn_lstm.add(Dense(32, activation = 'relu'))
model_cnn_lstm.add(Dropout(0.2))
model_cnn_lstm.add(Dense(1))
model_cnn_lstm.compile(loss='mae', optimizer="adam")
```

In [ ]:

```
model_cnn_lstm.compile(optimizer = 'adam',loss = 'mae', metrics = ['mae'])
history = model_cnn_lstm.fit(X_train,y_train, epochs = 10,  validation_data=(X_test,
```

In [ ]:

```
plt.plot(history.history['loss'], label='train')
plt.plot(history.history['val_loss'], label='test')
plt.legend()
plt.show()
```

In [ ]:

```
ypred = model_cnn_lstm.predict(X_test)

plt.plot(y_test[-72:], '-ob', label =  'observed')
plt.plot(ypred[-72:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

In [ ]:

```
ypred_old = (ypred * train_std[0]) + train_mean[0]
```

In [ ]:

```
y_test_old = (y_test*train_std[0]) + train_mean[0]
```

In [ ]:

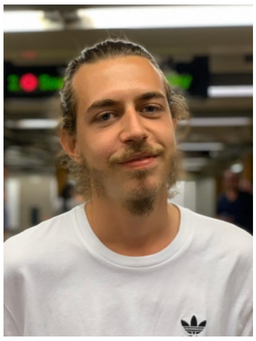
```
mae(ypred_old,y_test_old)
```

In [ ]:

```
plt.plot(y_test_old[-168:], '-ob', label =  'observed')
plt.plot(ypred_old[-168:], '-r', label = 'forecasted')
plt.grid()
plt.legend()
```

Figure D.1 : Other Models Code

CURRICULUM VITAE



Name Surname : Berk Kapucu

Place and Date of Birth : Çanakkale / 1997

E-Mail : beerkapucu@gmail.com

EDUCATION :

- **B.Sc.** : 2021, Istanbul Technical University, Faculty of Management, Industrial Engineering Department

CURRICULUM VITAE

Name Surname : Ömer Seyfeddin Koç

Place and Date of Birth : Istanbul / 1999

E-Mail : omerseyfeddinkoc@gmail.com

EDUCATION :

- **B.Sc.** : 2021, Istanbul Technical University, Faculty of Management, Industrial Engineering Department

CURRICULUM VITAE

Name Surname : Hakan Günay

Place and Date of Birth : Istanbul / 1998

E-Mail : hakan.gunay.itu@gmail.com

EDUCATION :

- **B.Sc.** : 2021, Istanbul Technical University, Faculty of Management, Industrial Engineering Department