ISTANBUL TECHNICAL UNIVERSITY
FACULTY OF ELECTRICAL AND ELECTRONICS ENGINEERING

PRICE PREDICTION WITH MACHINE LEARNING

CONTROL AND AUTOMATION ENGINEERING DESIGN II

Kerem ERCİYEŞ
Oktay KURT
Mustafa SOYDAN

Control and Automation Engineering

Thesis Advisor: Assoc. Prof. Dr. Tufan KUMBASAR

JUNE 2023

ISTANBUL TECHNICAL UNIVERSITY
FACULTY OF ELECTRICAL AND ELECTRONICS ENGINEERING

PRICE PREDICTION WITH MACHINE LEARNING

CONTROL AND AUTOMATION ENGINEERING DESIGN II

Kerem ERCİYEŞ
(040180603)
Oktay KURT
(040180522)
Mustafa SOYDAN
(040190779)

Kontrol ve Otomasyon Mühendisliği

Control and Automation Engineering

JUNE 2023

To our families ,

PREFACE

We would like to express our sincere gratitude to Associate Professor Tufan KUMBASAR, our mentor. He has been incredibly understanding and supportive of our graduation thesis. His advice and assistance have been extremely valuable to us. He has given us a lot to learn and has kept us inspired. We have grown from this experience, and it will undoubtedly aid us in our future professions.

We also want to express our gratitude to our family and friends for their constant support and inspiration.

JUNE 2023

Kerem Erciyeş
Oktay Kurt
Mustafa Soydan

CONTENTS

Sayfa

ABBREVIATIONS

ANN : Artificial Neural Network
PCA : Principal Component Analysis
AR : Autoregressive
MA : Moving average
ARMA : Autoregressive Moving Average
ARIMA : Autoregressive Integrated Moving Average
SARMA : Seasonal Autoregressive Moving Average
SARIMA : Seasonal Autoregressive Integrated Moving Average
SVR : Support Vector Regression
GPs : Gaussian Processes
SVM : Support Vector Machine
MSE : Mean Squared Error
MLP : Multi Layer Perceptron
ReLU : Rectified Linear Unit
RNN : Recurrent Neural Network
LSTM : Long Short-Term Memory
GRU : Gred Recurrent Unit
AIC : Akaike Information Criterion
BIC : Bayesian Information Criterion
RMSE : Root Mean Squared Error
MSE : Mean Squared Error
MAPE : Mean Absolute Percentage Error
NRMSE : Normalized Root Mean Squared Error

SYMBOLS

x	: Historic data features
y,yt	: Actual value
$\hat{y}$	: Predicted value
T	: Total time steps
t	: Arbitrary time steps
f	: Forecasting function
p	: Nonseasonal autoregressive degree
q	: Nonseasonal moving average degree
d	: Nonseasonal integration degree
P	: Seasonal autoregressive degree
Q	: Seasonal moving average degree
D	: Seasonal integration degree
ε,εt	: Random error
c	: Constant
b	: Bias
E	: Cost function
η	: Learning rate
μ	: Mean of time series
ϕi	: Autoregressive model coefficient
θj	: Moving average model coefficient
Tt	: Trend component
St	: Seasonal component
Rt	: Residuals
ωi	: Weights of neural network
σ	: Activation function
$\beta 1$,$\beta 2$	: Exponential decay parameters
ht	: Hidden state
it	: Input gate state
ft	: Forget gate state
ct	: Cell state

LIST OF TABLES

Sayfa

LIST OF FIGURES

Sayfa

NEURAL ORDINARY DIFFERENTIAL EQUATIONS FOR CONTROL AND SYSTEMS

SUMMARY

In this project, our study is to examine time series and make predictions about future data of that series. Time series is a series of data points collected at regular time intervals. It is a sequence of data points in which each data point is associated with a specific time. Time series data is often used to analyze trends and patterns over time. Stock price, sales data, temperature data, and traffic data are some time series examples. Time series data is often analyzed using techniques such as time series analysis, which involves using statistical methods to model and forecast future values based on historical data. Time series analysis can be used to identify trends and patterns in the data, and to make predictions about future values. Time series data is often collected and analyzed to make informed decisions about the future. There are several ways to forecast future prices. Traditional approaches are SMA, ARMA, ARIMA, SARMA, and SARIMA models. A modern approach is Machine Learning which is more complex. Artificial intelligence (AI) has the potential to revolutionize the way we predict prices for goods and services. By leveraging machine learning algorithms, AI can analyze large amounts of data and identify trends and patterns that can be used to make accurate price predictions. There are several approaches to using AI for price prediction, including regression analysis, time series analysis, and neural networks. Each of these approaches has its strengths and limitations, and the most appropriate method will depend on the specific context and the data available. In order to make accurate price predictions using AI, it is important to have a large and diverse dataset to train the model. The model should also be carefully tuned and tested to ensure that it is making accurate predictions. In this project, traditional models and Machine Learning techniques are studied to analyze and make a prediction of different data sets. We will also discuss the importance of properly preparing and preprocessing time series data for machine learning, as well as the evaluation methods used to assess the performance of forecasting models.

KONTROL VE SİSTEMLER İÇİN YAPAY ADİ DİFERANSİYEL DENKLEMLER

ÖZET

Bu projede, zaman serilerini inceleyerek belirli bir serinin gelecekteki verileri hakkında tahminlerde bulunmayı hedefliyoruz. Zaman serisi, belirli zaman aralıklarında toplanan bir veri noktaları dizisidir. Her veri noktası belirli bir zamanla ilişkilendirilmiş bir dizi veridir. Zaman serisi verileri genellikle zaman içindeki trendleri ve kalıpları analiz etmek için kullanılır. Hisse senedi fiyatları, satış verileri, sıcaklık verileri ve trafik verileri, zaman serisi örneklerinden birkaçıdır. Zaman serisi verileri, genellikle tarihsel verilere dayanarak gelecekteki değerleri modellemek ve tahmin etmek için istatistiksel yöntemler kullanan zaman serisi analizi gibi tekniklerle analiz edilir. Zaman serisi analizi, verilerdeki trendleri ve kalıpları belirlemek ve gelecekteki değerler hakkında tahminlerde bulunmak için kullanılabilir. Zaman serisi verileri genellikle geleceğe yönelik bilinçli kararlar almak için toplanır ve analiz edilir. Gelecekteki fiyatları tahmin etmek için birkaç yöntem vardır. Geleneksel yaklaşımlar arasında SMA, ARMA, ARIMA, SARMA ve SARIMA modelleri bulunmaktadır. Daha karmaşık bir modern yaklaşım ise Makine Öğrenmesi'dir. Yapay Zeka (AI), malların ve hizmetlerin fiyatlarını tahmin etme şeklimizi devrim niteliğinde değiştirme potansiyeline sahiptir. Makine öğrenmesi algoritmalarını kullanarak, AI büyük miktarda veriyi analiz edebilir ve doğru fiyat tahminleri yapmak için kullanılabilecek trendleri ve kalıpları belirleyebilir. Fiyat tahmininde AI kullanmak için birkaç yaklaşım vardır. Bunlar arasında regresyon analizi, zaman serisi analizi ve sinir ağları bulunmaktadır. Her bir yaklaşımın kendi güçlü ve zayıf yanları vardır ve en uygun yöntem, belirli bağlam ve mevcut verilere bağlı olacaktır. AI kullanarak doğru fiyat tahminleri yapmak için, modeli eğitmek üzere büyük ve çeşitli bir veri setine sahip olmak önemlidir. Ayrıca modelin doğru tahminlerde bulunduğundan emin olmak için dikkatle ayarlanmalı ve test edilmelidir. Bu projede, farklı veri setlerini analiz etmek ve tahminde bulunmak için geleneksel modeller ve Makine Öğrenmesi teknikleri üzerinde çalışıyoruz. Ayrıca, zaman serisi verilerini makine öğrenmesi için doğru bir şekilde hazırlamanın ve ön işlemenin önemini, ve tahmin modellerinin performansını değerlendirmek için kullanılan yöntemleri de tartışacağız.

1. INTRODUCTION

The digital age continues to influence various sectors by offering the capabilities to interpret vast amounts of data for strategic decision-making. This has redefined many disciplines, including finance and investment, where data-driven insights are instrumental. One critical aspect in this regard is the ability to predict future data from a given series of data points, or, in other words, time series forecasting. Time series forecasting encompasses the utilization of machine learning models and traditional statistical methods to infer patterns from historical data and subsequently predict future trends. These trends can be found in a wide variety of data, such as stock prices, sales data, temperature readings, or even traffic data. This thesis focuses on understanding, analyzing, and predicting time series data, with a spotlight on Bitcoin price prediction. One of the primary objectives of this study is to evaluate and compare traditional forecasting models like SMA, ARMA, ARIMA, SARMA, and SARIMA, with modern Machine Learning techniques. Machine Learning, a subset of Artificial Intelligence, is an intricate but extremely powerful approach that has the potential to revolutionize how we predict future data. Different AI techniques such as regression analysis, time series analysis, and neural networks are used for price prediction, each with its strengths, limitations, and context-specific utility.

A significant element of this study is data handling. Data preparation forms the foundation of any successful forecasting, which involves cleaning, preprocessing, scaling, and filling in missing values in the data. Subsequent steps include splitting the data into training and test sets and transforming the data to make it suitable for machine learning models. Visualizing data plays a crucial role in understanding trends and patterns within the data.

The application of these techniques to time series forecasting can have profound implications for businesses, investors, and decision-makers who rely heavily on accurate predictions for strategic planning. For instance, Bitcoin price prediction, the specific case study of this thesis, is of utmost interest to stakeholders in the digital currency market.

The research plan includes a comprehensive exploration of time series analysis, machine learning algorithms, their suitability for different data sets, and the implementation of these techniques using popular tools such as Python and MATLAB.

The study will incorporate a feedback mechanism with an iterative process of testing and refining the model based on its performance with sample data.

This project aims not just at understanding the technical and fundamental aspects of time series analysis but also at exploring the transformative potential of artificial intelligence in the field of price prediction. By examining these complex methodologies and tools, this thesis aspires to contribute to the growing body of knowledge in this fascinating intersection of data science and finance.

1.1 Literature Search

Time Series Analysis: Time series analysis is an essential aspect of statistical studies and involves analyzing data points collected at consistent time intervals to understand the underlying structures and patterns (Chatfield, 2004). It is a popular tool used for forecasting in various sectors such as finance, weather, sales, and traffic, among others (Brockwell & Davis, 2016).

Forecasting Models: Several statistical methods like Simple Moving Average (SMA), AutoRegressive Moving Average (ARMA), AutoRegressive Integrated Moving Average (ARIMA), Seasonal AutoRegressive Moving Average (SARMA), and Seasonal AutoRegressive Integrated Moving Average (SARIMA) have traditionally been used for time series analysis (Box et al., 2015). These models have shown significant efficacy in predicting future values based on past data.

Machine Learning & AI: Recent advancements in artificial intelligence and machine learning have led to the development of new techniques for time series analysis. These include regression analysis, time series analysis, and neural networks, each with unique strengths and limitations (Brownlee, 2018). These techniques have shown promise in enhancing the accuracy of predictions, particularly in complex datasets.

Bitcoin & Cryptocurrency: The volatile nature of Bitcoin and other cryptocurrencies has made them a prime subject for predictive analysis. Various studies have been conducted utilizing different models, such as LSTM (Long Short Term Memory), to predict Bitcoin prices (McNally, 2018). Sentiment analysis of social media data and search trends have also been used as inputs to enhance prediction accuracy.

Data Preparation & Preprocessing: Effective data handling and preprocessing are crucial steps in time series analysis, involving cleaning, scaling, and filling in missing values in the data (García et al., 2016). The transformation of data to suit the ML models and visualization for a better understanding of the data are also important aspects.

2. TIME SERIES ANALYSIS

Time series analysis is a statistical technique used to analyze and forecast the future values of a series of data points that are spaced at uniform time intervals. It involves analyzing the patterns and trends in the data and using this information to make predictions about future values. Time series analysis is often used in fields such as economics, finance, and meteorology, where data is collected over time and there is a need to forecast future values.

Time series analysis is the statistical analysis of data collected over time, to identify patterns and trends in the data and make forecasts about future values. Time series analysis can be divided into parts:

- Collecting and organizing the data: This involves collecting data at regular intervals over a while and organizing it in a time series format.

- Identifying patterns and trends: This involves examining the data to identify any patterns or trends that may exist. This can be done through visual inspection of the data or by using statistical techniques such as autocorrelation or spectral analysis.

- Modeling the data: This involves fitting a statistical model to the data to capture the underlying patterns and trends. Many different types of models can be used for this purpose, including linear models, autoregressive models, and moving average models.

- Forecasting future values: Once a model has been fit to the data, it can be used to forecast future values. This involves using the model to make predictions about what the data will look like in the future based on the patterns and trends identified in the data.

Overall, time series analysis is a powerful tool for analyzing and forecasting data that is collected over time. It is widely used in a variety of fields and has many applications, including predicting stock prices, forecasting weather patterns, and analyzing economic trends.

2.1 Stationarity

In time series analysis, a stationary time series is one whose statistical properties do not change over time. In other words, the mean, variance, and autocorrelation structure of a stationary time series are constant over time.

There are several formal definitions of a stationary time series, but one common one is a time series that has a constant mean and a constant variance, and an autocovariance that does not depend on time. More formally, a stochastic process X(t) is said to be weakly stationary if the following conditions are met:

- The mean of the process is constant over time.
- The variance of the process is also constant over time.
- The covariance between X(t) and X(t+h) is a function only of the time lag h and not of t.

2.2 Trend

The trend in time series analysis refers to the underlying pattern or direction of a time series data over a period of time. This pattern can be either upward (increasing), downward (decreasing), or flat (no change).

According to Hyndman and Athanasopoulos (2018), the trend is an important component of time series data as it represents the long-term evolution of the data. The trend can be either deterministic (i.e., follows a specific pattern) or stochastic (i.e., random). Deterministic trends can be modeled using mathematical functions such as linear, exponential, or polynomial functions. Stochastic trends, on the other hand, cannot be accurately predicted and are often modeled using statistical techniques.

Understanding and modeling trends are crucial in time series analysis as it allows for the prediction and forecasting of future data values. Trends can also be seasonal, meaning that they follow a specific pattern that repeats itself over a fixed period (e.g., monthly, quarterly, annually). Seasonal trends are often removed from time series data before analyzing the trend component.

2.3 Seasonality

Seasonality refers to the repeating patterns or trends that occur within a specific time frame, such as monthly or yearly. These patterns may be caused by factors such as weather, holidays, or societal trends. For example, a retail store may experience increased sales during the holiday season or a hotel may experience increased occupancy rates during the summer months.

Seasonality can have a significant impact on time series analysis and forecasting, as it can distort the underlying trend or level of the data. Therefore, it is important to identify and adjust for seasonality to accurately analyze and forecast future data.

2.4 Residue

Residue usually refers to the difference between the observed value of a time series at a particular point in time and the predicted value of the time series at that point in time, based on a statistical model. This difference is also known as the "residual" or "error" term in the model. In order to obtain residual function, trend and seasonal components should be removed. Splitting a time series into these three components is called time series decomposition.

For example, suppose you have a time series of daily stock prices, and you want to model the trend in the stock prices over time. You could fit a linear regression model to the time series, with the predicted value of the stock price at the time being a function of the time t and the slope and intercept of the regression line. The residue at time t would then be the observed stock price at the time, minus the predicted stock price.

In addition, if the model is able to accurately capture the underlying trend and pattern in the data, then the residuals should be random noise. This means that the residuals should be uncorrelated and have constant variance over time. If the residuals exhibit systematic patterns or trends, it could indicate that the model is not capturing all of the important features of the data, and the model might need to be revised or improved. A residual example of the Air Passengers data set is given below when trend and seasonality are decomposed.

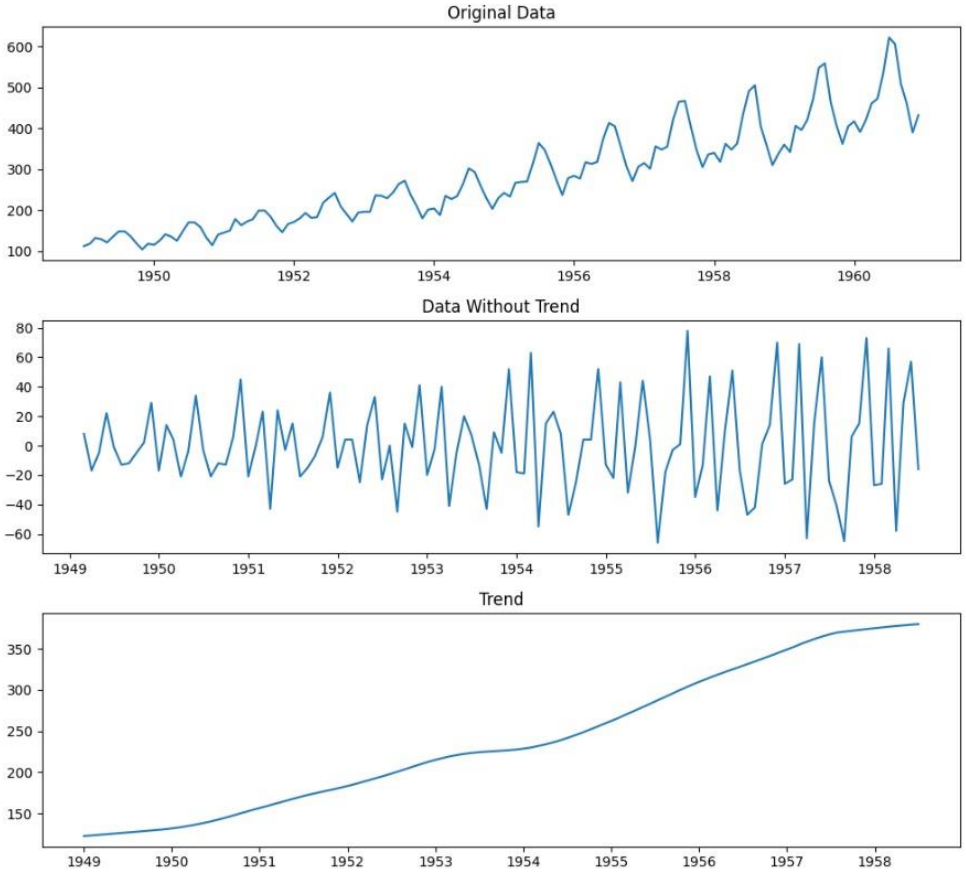


Figure 1: Trend is Decomposed in Air Passengers Data

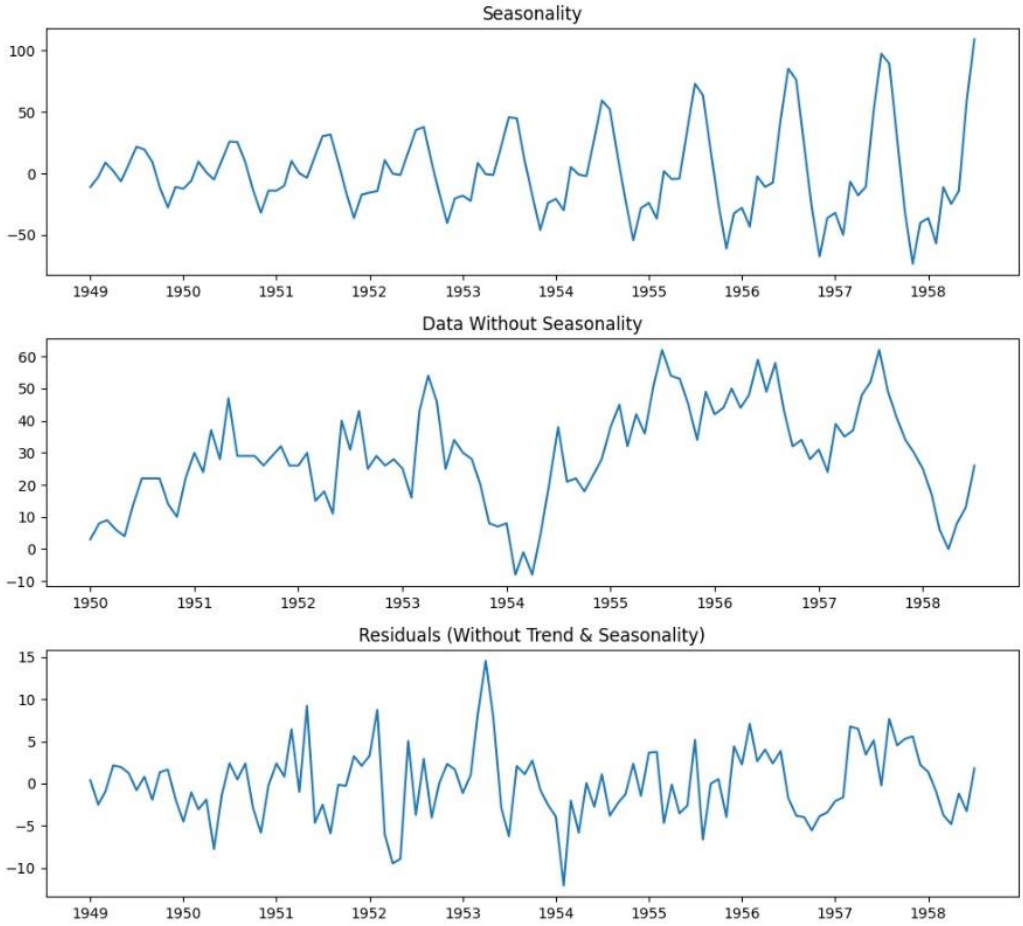


Figure 2: : Seasonality is Decomposed in Air Passengers Data

2.5 The Least Squares Method

The least squares method is a statistical technique that aims to minimize the sum of the squared residuals between a fitted model and the observed data (Hinkley, 1981). Essentially, this means that the method tries to find the line of best fit (also known as the "regression line") that best represents the relationship between the dependent and independent variables in the data.

2.6 Confidence Interval

A confidence interval is a range of values that is likely to contain the true value of a population parameter. It is usually based on a sample statistic and is used to estimate the value of the population parameter. A confidence interval is typically constructed at a predetermined confidence level, such as 95%. This means that if the same sampling process were repeated many times, the confidence interval would contain the true population parameter in about 95% of the samples.

2.7 Prediction Interval

A prediction interval is a range of values that is likely to contain the value of a future observation. It is used to estimate the range within which a future observation will fall, based on the uncertainty in the prediction. Like a confidence interval, a prediction interval is constructed at a predetermined confidence level, such as 95%. However, a prediction interval is typically wider than a confidence interval because it includes the uncertainty in the prediction of a single future observation, rather than the average of many observations.

Prediction intervals predict the distribution of individual future points, whereas confidence intervals and credible intervals of parameters predict the distribution of estimates of the true population mean or other quantity of interest that cannot be observed. A prediction interval predicts an individual number, whereas a confidence interval predicts the mean value. A prediction interval focuses on future events, whereas a confidence interval focuses on past or current events. In summary, a confidence interval is used to estimate a population parameter, while a prediction interval is used to predict a future observation.

In short, a prediction interval is an interval associated with a random variable yet to be observed (forecasting) however a confidence interval is an interval associated with a parameter and is a frequentist concept.

2.8 Distance Metrics

Measuring distance is one of the key parts of Machine Learning algorithms. There are 5 fundamental distance metrics in Machine Learning.

- Euclidean distance: This is the most commonly used distance metric and is based on the Pythagorean theorem. It is the straight-line distance between two points in Euclidean space.
- Manhattan distance: This distance metric is based on the idea of measuring distance by only moving horizontally or vertically. It is also known as the "taxicab" distance because it is the distance a taxi cab would need to travel to go from one point to another.

- Cosine similarity: This distance metric is used to measure the similarity between two vectors. It is commonly used in information retrieval and natural language processing tasks.
- Hamming distance: Hamming distance is another type of distance metric that is used to measure the difference between two binary strings. It is defined as the number of positions at which the two strings differ. Hamming distance is often used in error detection and correction, particularly in communication systems.
- Minkowski distance: This distance metric is a generalization of the Euclidean distance and the Manhattan distance. It is defined as the sum of the absolute differences between the coordinates of the points, raised to the power of p and then taking the p^{th} root.

2.9 Data Preprocessing

2.9.1 PCA

Principal Component Analysis, commonly referred to as PCA, is a statistical procedure used in machine learning and data analysis to reduce the dimensionality of a dataset while preserving as much of the data's original variance as possible. It achieves this by transforming the dataset into a new set of orthogonal variables, known as principal components, which are uncorrelated and ordered such that they capture the largest amount of variance in the data (Jolliffe and Cadima, 2016).

In time series examples, correlations of the time series components (past and future values) can be calculated via eigenvalues and eigenvectors of the covariance matrix. Eigenvectors corresponding to the largest eigenvalues are the principal components that capture the most variance in the data.

Therefore, it can also be used for stock market predictions by data visualization, noise filtering, feature extraction.

2.9.2 Standardization

Standardization is a crucial preprocessing step in many machine learning algorithms. It transforms the features in the data to have zero mean and unit variance, bringing them to the same scale. This technique is especially important when dealing with features that have different units or scales, as it allows for more uniform comparison and prevents features with larger scales from dominating others (Hastie et al., 2001).

The standardization process involves calculating the mean and standard deviation for a feature, and then using these to scale the feature such that the transformed feature has a mean of 0 and a standard deviation of 1. This transformation results in a distribution with a bell curve shape centered at zero, also known as a standard normal distribution.

The advantage of standardization is that it makes the algorithm less sensitive to the scale of features. This can be crucial for machine learning models like linear regression, logistic regression, and support vector machines, which assume that all features are centered around zero and have variance in the same order. Moreover, it improves the numerical stability and convergence speed of gradient-based optimization algorithms.

However, standardization does not normalize the distribution of the data; it only changes the scale. The data might still be skewed or follow a non-normal distribution, so other preprocessing steps might be necessary depending on the nature of the data and the requirements of the model.

References:

Hastie, T., Tibshirani, R., & Friedman, J. (2001). The Elements of Statistical Learning. Springer.

3. PREDICTION MODELS

3.1 Stochastic Prediction Models

Stochastic forecasting models are a class of statistical models that are used to predict future events or outcomes. These models are based on the idea that there is a degree of randomness or uncertainty in the future, and that this uncertainty can be quantified and incorporated into the forecasting process.

Autoregressive (AR) and moving average (MA) models are two commonly used techniques for modeling and predicting time series data. Time series data refers to data that is collected over time. These models are used to identify patterns in the data and make forecasts about future values.

3.1.1 Stochastic prediction models

The Autoregressive (AR) model is a statistical model that describes the current value of a time series as a linear combination of its past values. Mathematically, an autoregressive model of order p can be written as:

$$Y(t) = c + a(1).y(t-1) + a(2).Y(t-2) + \cdots + a(p).Y\,(t-p) + e(t) \quad (3.1)$$

where Y(t) is the value of the time series at time t, c is a constant term, a(1), a(2), ..., a(p) are coefficients that represent the strength of the influence of each past value on the current value, and e(t) is a white noise error term.

Autoregressive models are commonly used to model time series data because they can capture patterns in the data that are not present in a white noise process. For example, an autoregressive model of order 1 (AR(1)) can capture a trend in the data, while an autoregressive model of order 2 (AR(2)) can capture both a trend and periodic fluctuations in the data.

Autoregressive models have been widely studied in the field of time series analysis. One of the earliest references to autoregressive models is the work of Box and Jenkins (1970), who developed a method for estimating the parameters of an autoregressive model using a technique called maximum likelihood estimation.

3.1.2 Moving Average Model

The moving average (MA) model is a statistical technique used to analyze and forecast time series data. It does this by taking the average of a set of data points over a specified time and then using this average to predict future values.

Mathematically, the moving average model can be represented as follows:

$$MA(t) = \left(\frac{1}{n}\right) \sum_{i=t-n+1}^{t} X(i) \tag{3.2}$$

Where MA(t) is the moving average at time t, $X(i)$is the data point at the time i and n is the number of data points used in the average.

For example, if we wanted to smooth out the daily fluctuations in a stock price and focus on the long-term trend, we could use a moving average model with a window of 30 days (n=30). This would take the average of the past 30 days of stock price data and use that average to predict the stock price at time t.

Moving average models have been widely used in statistical analysis and have been shown to be effective at smoothing out short-term fluctuations and highlighting long-term trends (Hyndman & Athanasopoulos, 2018). They have been applied in a variety of fields, including finance, economics, and meteorology, to predict future values and make informed decisions.

3.1.2.1 XOM Price Prediction with MA

This experimental study determines a time series prediction with Moving Average (MA) algorithm. The data named XOM is the value of Exxon Mobile Corporation in the stock market for five years. The window size was chosen as 20. So, each 20 data set is used to predict the 21st data that comes after it.

Figure 3: The Stock Price of XOM Actual (Blue) and Prediction (Green) (from 10.05.2017 to 10.04.2022)

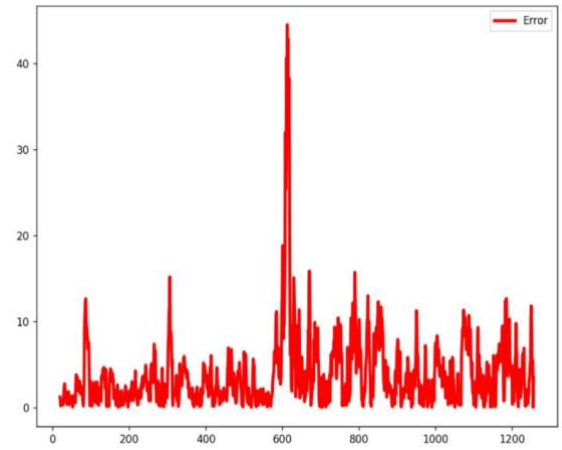


Figure 3: Percentage of Error in XOM Prediction

The percentage of error is calculated by dividing the difference between the actual data and predicted data, and actual data then multiplying the result by one hundred.

3.1.3 Autocorrelation (ACF)

Autocorrelation is a statistical concept that describes the correlation between a variable and a lagged version of itself. It is often used in time series analysis to identify patterns and trends in data and to help forecast future values.

In the context of time series analysis, autocorrelation is a measure of the correlation between a variable and a lagged version of itself. For example, the autocorrelation

between a series X and a lag-1 version of itself ($X(t)$and $X(t-1)\}$) is a measure of the correlation between $X(t)$and $X(t-1)$.ACF is a plot of the autocorrelation of a time series with different lag values. It helps to identify the type of autoregressive (AR) model that can be used to predict the time series.

3.1.4 Partial Autocorrelation (PACF)

Partial autocorrelation is a statistical concept that describes the correlation between a variable and a lagged version of itself while controlling for the effects of intermediate lags.

In the context of time series analysis, autocorrelation refers to the correlation between a variable and a lagged version of itself. Partial autocorrelation extends this concept by controlling for the effects of intermediate lags. For example, the partial autocorrelation between a series X and a lag-2 version of itself ($X(t)$ and $X(t-2)$) is a measure of the correlation between $X(t)$ and $X(t-2)$., controlling for the effects of $X(t-1)$).

PACF is a plot of the partial autocorrelation of a time series with different lag values. It helps to identify the type of moving average (MA) model that can be used to predict the time series.

3.1.5 Information Criteria

Information criteria are used to assess the quality of a statistical model based on the amount of information it contains. They are used to assess the quality of statistical models based on the trade-off between goodness of fit and complexity. The most commonly used information criteria are the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC).

3.1.5.1 Akaike Information Criterion (AIC)

According to Burnham and Anderson (2004), AIC is defined as a measure of the relative quality of a statistical model for a given data set. It deals with the trade-off between the goodness of fit of the model and the complexity of the model. In other words, AIC helps to balance the fit of the model to the data with the number of parameters in the model. The lower the AIC value, the better the model. The formula of AIC is:

$$AIC = 2.k - 2\ln(L) \tag{3.3}$$

where k is the number of parameters in the model and L is the likelihood of the model. The AIC measures the trade-off between the goodness of fit of the model and the complexity of the model. A lower AIC value indicates a better-fitting model.

3.1.5.2 Akaike Information Criterion (AIC)

BIC, on the other hand, is similar to AIC but places a stronger emphasis on the complexity of the model (Schwarz, 1978). BIC is defined as a criterion for model selection among a finite set of models; the model with the lowest BIC is preferred. BIC is generally more conservative than AIC, meaning that it is more likely to select a simpler model with fewer parameters. The formula of BIC is:

$$BIC = kln(n) - 2\ln(L) \tag{3.4}$$

where n is the number of observations in the dataset and k and L is defined as in the AIC. The BIC penalizes model complexity more heavily than the AIC, so it tends to favor simpler models.

3.1.6 Performance Metrics

Several performance metrics can be used to evaluate the accuracy of a regression model. Here are a few common ones below.

3.1.6.1 Mean Absolute Error (MAE)

Mean of the absolute differences between the predicted values and the true values. It gives an idea of the magnitude of the error, but it does not provide information about the direction (i.e., whether the model is under or over-estimating the true values).

$$MAE = \frac{1}{N}\sum_{j=1}^{N}|y_j - \tilde{y}_j| \tag{3.5}$$

3.1.6.2 Mean Squared Error (MSE)

Mean of the squared differences between the predicted values and the true values. It is more sensitive to outliers than MAE, as the squared errors of large errors will be much larger than the errors themselves.

$$MSE = \frac{1}{N}\sum_{j=1}^{N}(y_j - \tilde{y}_j)^2 \tag{3.6}$$

3.1.6.3 Root Mean Squared Error (RMSE)

The root of the MSE. It is interpretable in the same units as the original data, which can be helpful for comparison.

$$RMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(y_j - \tilde{y}_j)^2} \quad (3.7)$$

3.1.6.4 Normalized Root Mean Squared Error (NRMSE)

The Normalized Root Mean Squared Error (NRMSE) is a standard way of quantifying the prediction error of a model. It's an extension of the Root Mean Squared Error (RMSE), a commonly used metric in regression problems, but with normalization to make the scale of the error more interpretable.

$$NRMSE = \sqrt{\frac{1}{N}\sum_{j=1}^{N}\frac{(y_j - \tilde{y}_j)^2}{\sigma}} \quad (3.8)$$

3.1.6.5 Coefficient of Determination (R^2)

R square measures the proportion of the variance in the true values that is explained by the model. It ranges from 0 to 1, with 1 indicating a perfect fit.

$$SE(\bar{Y}) = \sum_{j=1}^{N}(y_j - \bar{y})^2 \quad (3.9)$$

3.1.7 One Step Ahead vs N-step Ahead

In the field of time series analysis, forecasting refers to the process of making predictions about future events based on past data. One-step-ahead forecasting involves making a prediction for a single time point in the future, while n-step-ahead forecasting involves predicting n-time points in the future.

One-step-ahead forecasting is useful when you are interested in predicting a specific time point and do not need to know the future values beyond that point. This type of forecasting can be useful in real-time applications where you need to make a prediction as soon as possible.

On the other hand, n-step-ahead forecasting is useful when you are interested in making predictions for multiple future time points. This type of forecasting can be useful in planning and decision-making, as it allows you to anticipate future events and take action accordingly.

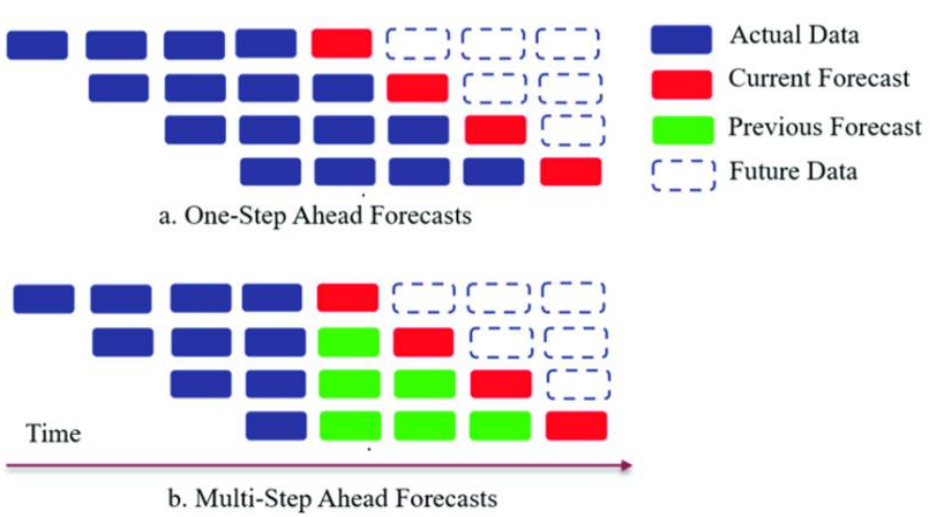


Figure 5: 3 One-Step Ahead vs. Multi-Step Ahead

- One-step ahead forecasting where at each step forecast = 1 and window size = w.
- Multi-step Ahead Forecasting where at each step forecast = h, window size = w.

3.1.8 ARMA (Autoregressive Moving Average) Model

Autoregressive moving average (ARMA) models are statistical models that describe time series in terms of two components: an autoregressive (AR) component and a moving average (MA) component (Box, Jenkins, and Reinsel, 1994). The AR component captures the autocorrelation of a time series at different time lag when the MA component indicates the effect of past errors or innovations on the current value of the time series.

The equation of the ARMA model is following:

$$y(t) = c + \emptyset_1 . y(t-1) + \cdots + \emptyset_p . y(t-p) + \varepsilon(t) + \theta_1 . \varepsilon(t-1) + \cdots + \theta_q . \varepsilon(t-q) \quad (3.10)$$

Where:

- y(t) is the time series data at time t
- C is a constant term
- $\emptyset_1$,...,$\emptyset_n$ are the autoregressive coefficients

- $\varepsilon(t)$is the error term at time t
- θ_1,…, θ_q are the moving average coefficients

To determine p and q values information criteria techniques such as AIC and BIC are being used.

3.1.8.1 Female Birth Value Prediction Using ARMA

This experimental study determines a time series prediction with Autoregressive Moving Average (ARMA) algorithm. The data named Female Birth is the number of births from 01.01.1959 to 12.31.1959 in a district in California.

The autocorrelation graph below is to determine the q parameter of the Arma(p,q) function.

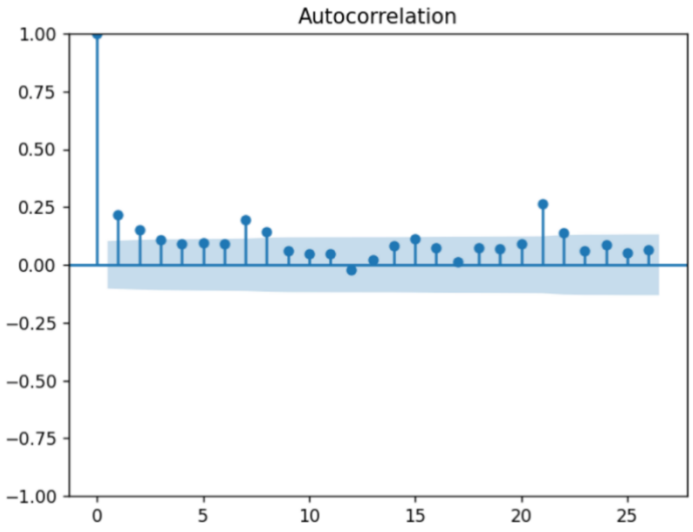


Figure 6: ACF Graph for Female Birth Data

Q parameter is selected as 2 based on Figure 4.

The partial autocorrelation graph below is to determine the p parameter of the Arma(p,q) function.

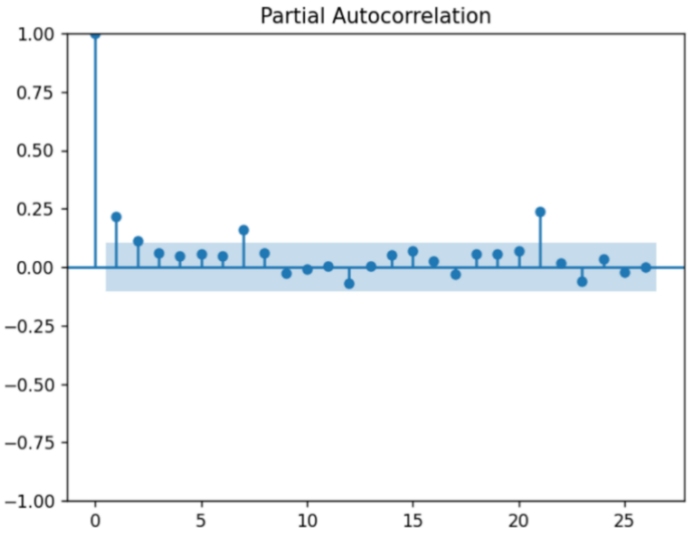


Figure 7: PACF Graph for Female Birth Data

The p parameter is selected as 2 based on Figure 5.

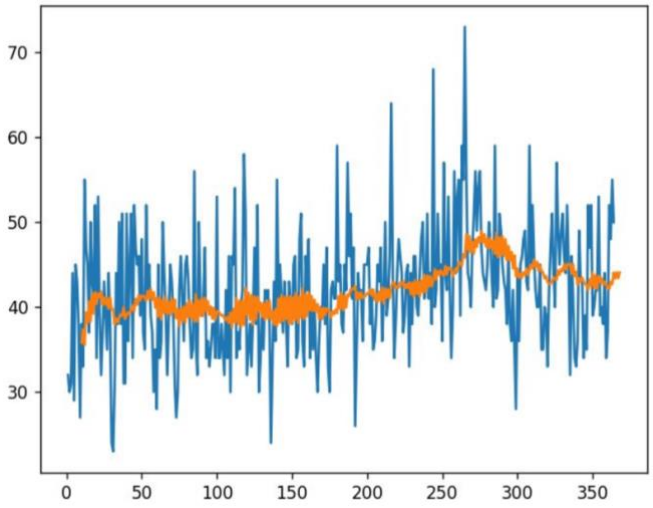


Figure 4: Actual Data (Blue) and Predicted Data (Orange) for Female Birth

By looking at Figure 6 it can be understood that ARMA(2,2) model captured the general behavior of this time series, but it was not enough to capture the sudden changes.

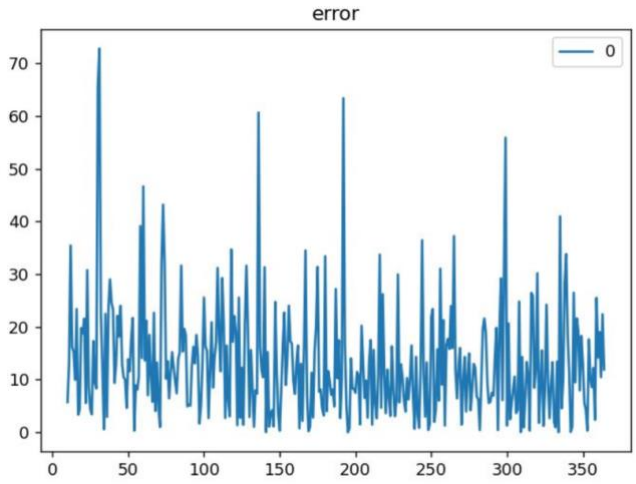


Figure 9: Percentage Error of Female Birth Prediction

The percentage of error is calculated by dividing the difference between the actual data and predicted data, and actual data then multiplying the result by one hundred.

The residual and density graphs are given below

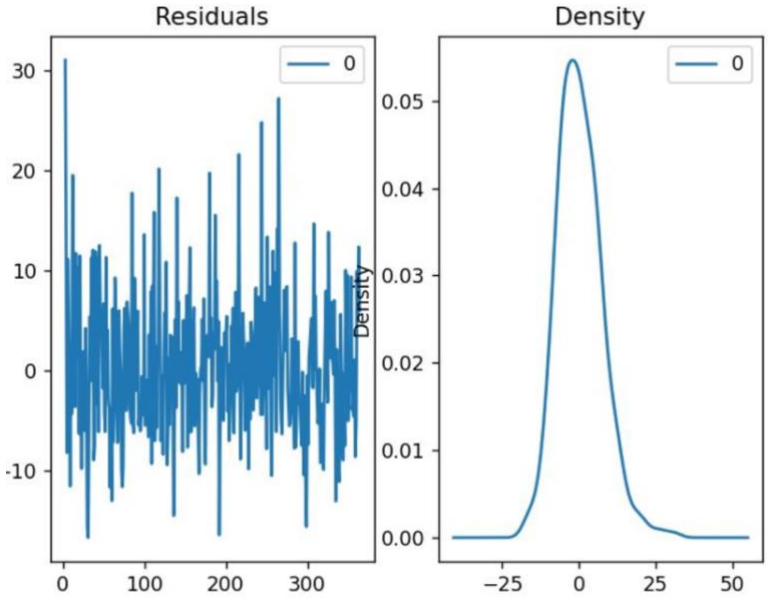


Figure 10: Residual and Density Graphs

The density function of the residual has random noise characteristics and zero means. This means the prediction is accurate.

3.1.9 ARIMA (Autoregressive Integrated Moving Average) Model

According to Hyndman and Athanasopoulos (2018), the ARIMA model is a combination of autoregressive (AR) and moving average (MA) models and is specified

by three parameters: p, d, and q. The parameter p represents the order of the autoregressive component, which captures the dependence of the current value of the time series on past values. The parameter d represents the degree of differencing required to make the time series stationary, and the parameter q represents the order of the moving average component, which captures the dependence of the current value of the time series on past errors. The ARIMA model can be written as:

$$ARIMA(p,d,q) = \left(1 - \sum_{i=1}^{P} \emptyset_i L^i\right)(1-L)^d y_t = \left(1 + \sum_{j=1}^{q} Q_j L^j\right) \quad (3.11)$$

Thus, if a trend is determined in the time series data set, the ARIMA models use parameter d to eliminate the trend and accomplish stationary data.

3.1.9.1 Shampoo Sales Prediction with ARIMA

This experimental study determines a time series prediction with Autoregressive Integrated Moving Average (ARIMA) algorithm. The data named Shampoo Sales is shampoo sales from January 1949 to November 1960 monthly. The mean value of data is changing over a period of time which means data has a trend.

To decide the d value of ARIMA(p,d,q), differentiated graphs of the original series should be considered.

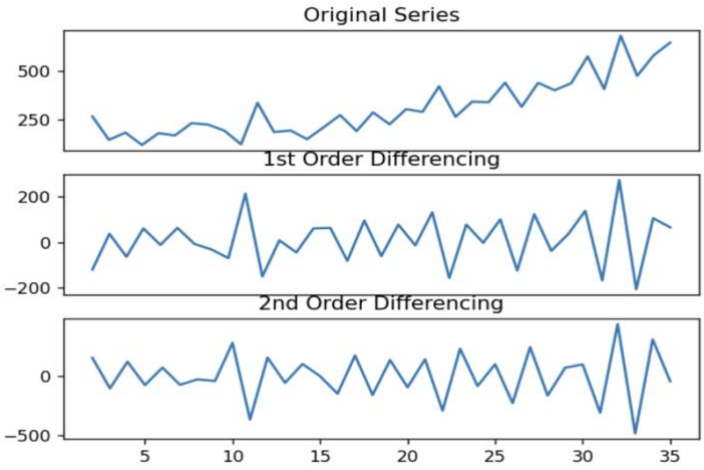


Figure 5: Original Series, First and Second Differencing Graphs

The first order differencing is stationary so d is chosen as 1. The second-order differencing is unnecessary since it brings more noise.

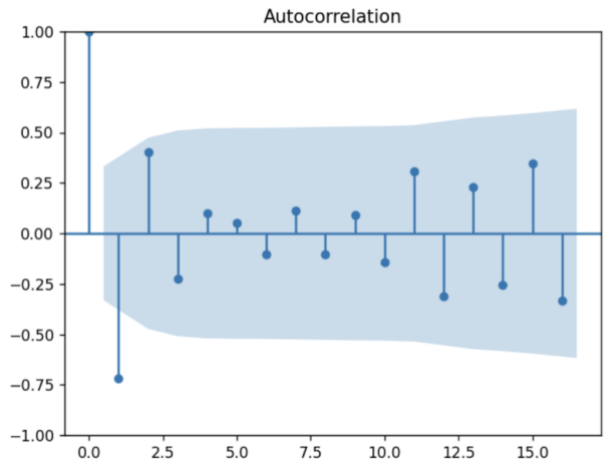


Figure 12: ACF Graph for Shampoo Sales Data

The q parameter is selected as 1 based on Figure 10.

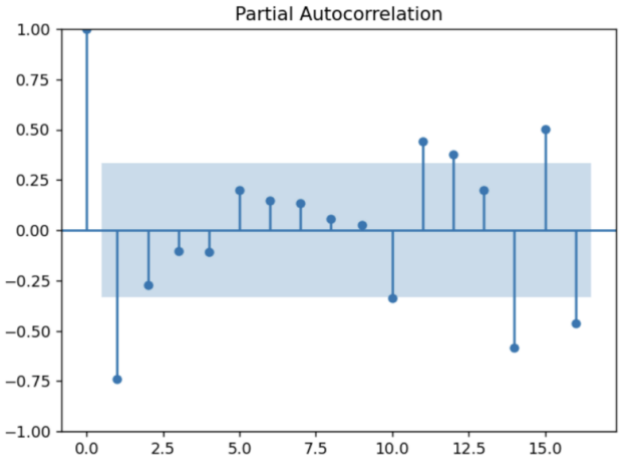


Figure 13: PACF Graph for Shampoo Sales Data

The p parameter is selected as 2 based on Figure 11.

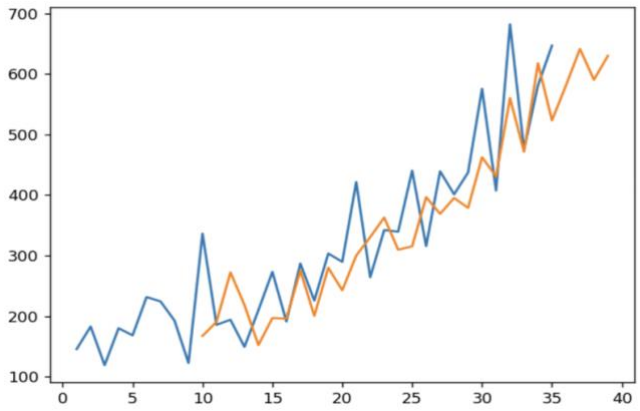


Figure 14: Real (Blue) and Predicted (Orange) Shampoo Sales Data

The ARIMA(2,1,1) model captured the trend and the prediction is accurate.

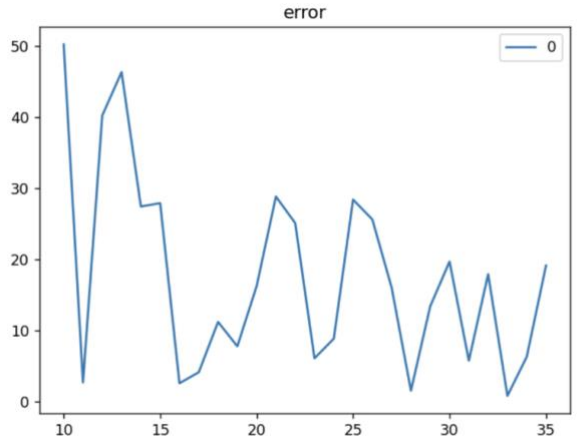


Figure 6: Percentage Error of Shampoo Sales Prediction

The percentage of error is calculated by dividing the difference between the actual data and predicted data, and actual data then multiplying the result by one hundred.

3.1.9.2 BTC Price Prediction with ARIMA

This experimental study determines a time series prediction with Autoregressive Integrated Moving Average (ARIMA) algorithm. The BTC data is the price of Bitcoin last three years daily.

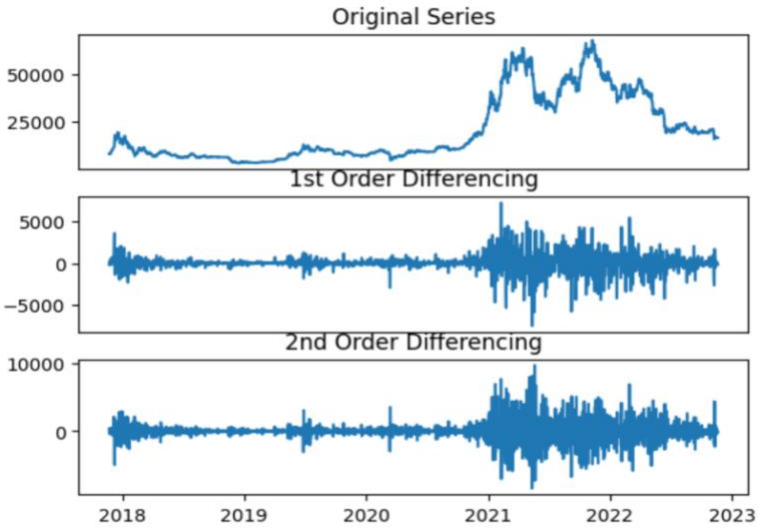


Figure 7: Original, First, and Second Differentiation Graphs for BTC

The first order differencing is stationary so d is chosen as 1. The second-order differencing is noisier.

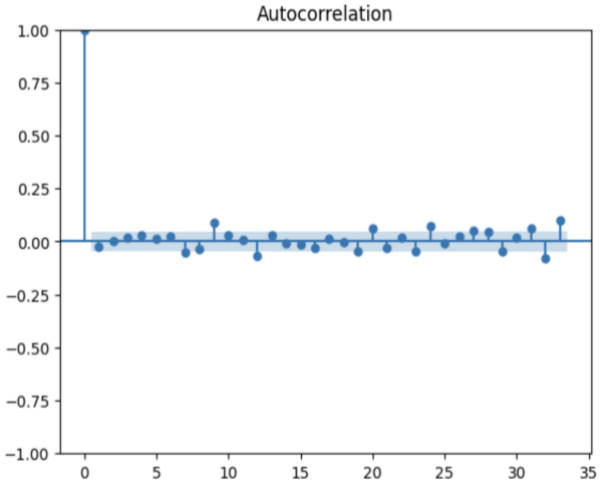


Figure 8: ACF and PACF Graph For BTC

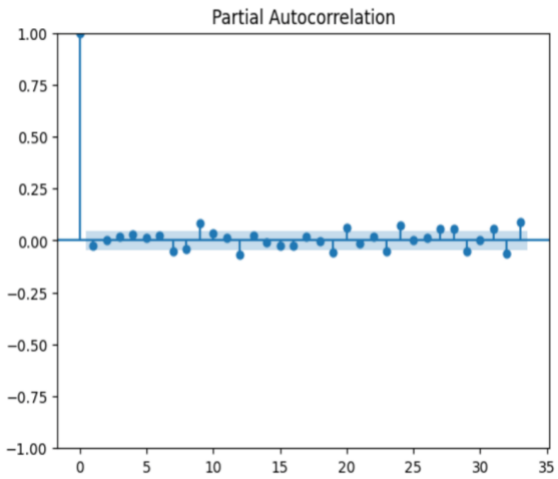


Figure 9: ACF and PACF Graph For BTC

According to the two graphs in Figure 15, there is no correlation between any data.

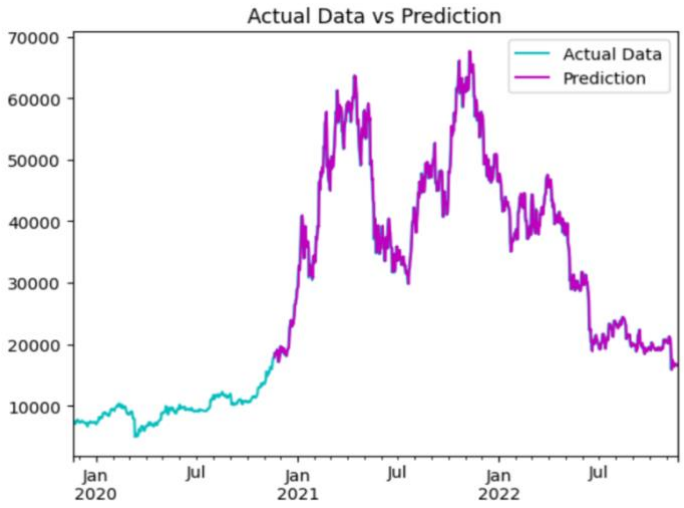


Figure 10: Actual (Green) and Predicted (Purple) Values for BTC

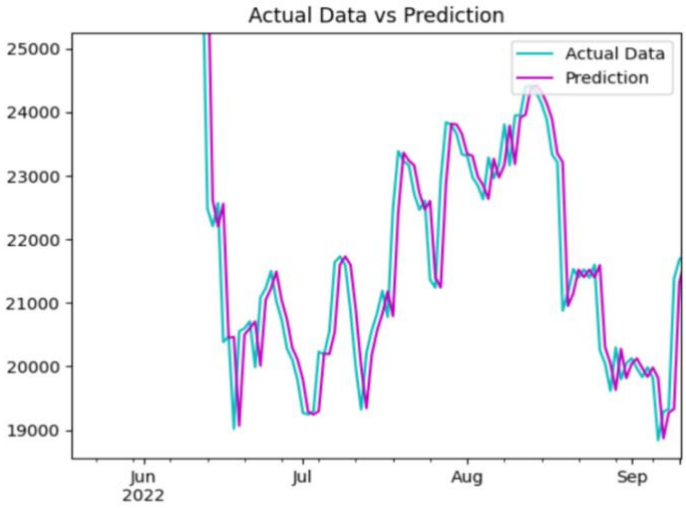


Figure 11: Zoomed Actual (Green) and Predicted (Purple) Values for BTC

The model's prediction for future values is the same values as previous ones since BTC data cannot be captured by using only traditional models.

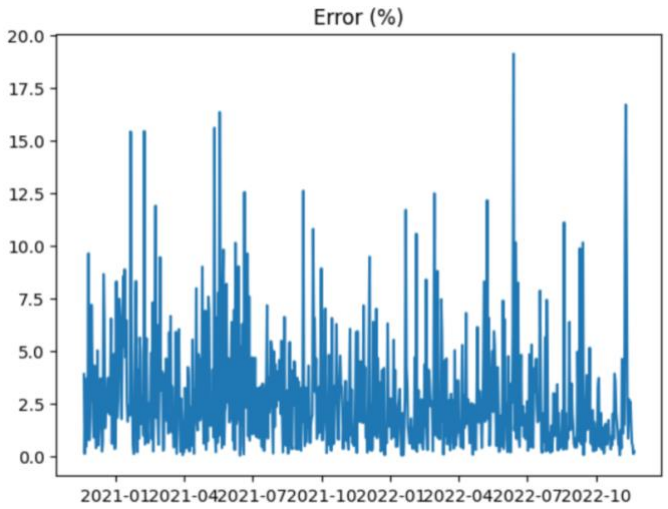


Figure 12: Percentage Error of BTC Prediction

The percentage of error is calculated by dividing the difference between the actual data and predicted data, and actual data then multiplying the result by one hundred.

3.1.10 SARMA (Seasonal Autoregressive Moving Average)

The SARMA model is a generalization of the ARMA model with a seasonality component. When seasonality is realized in a data set then, the SARMA model can be used to eliminate seasonality and gather the stationary data.

3.1.10.1 SARMA (Seasonal Autoregressive Moving Average)

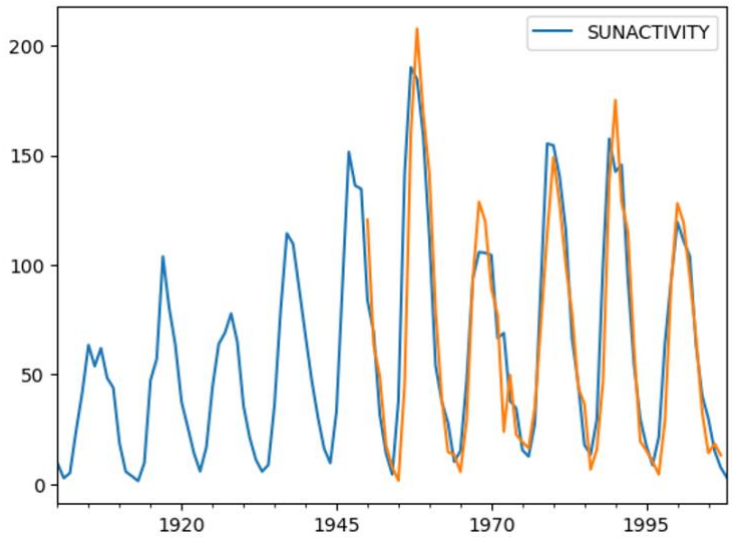


Figure 22: Sun activity Prediction

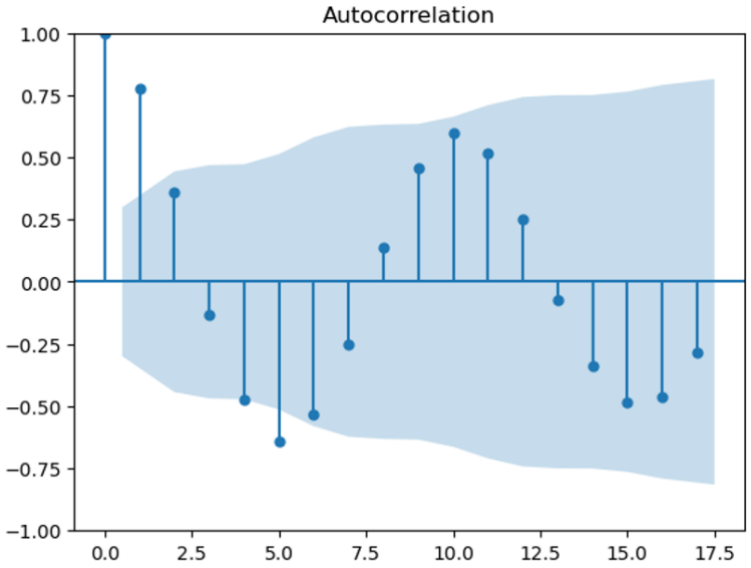


Figure 23: ACF Graph for Sunactivity Data

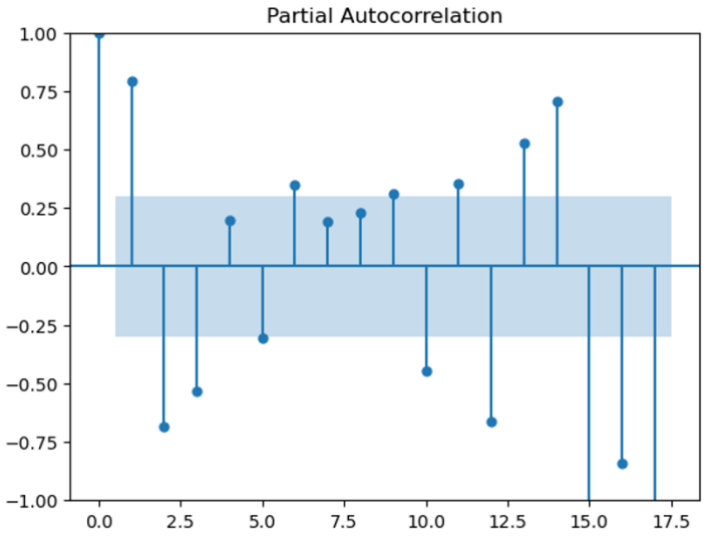


Figure 24: ACF Graph for Sunactivity Data

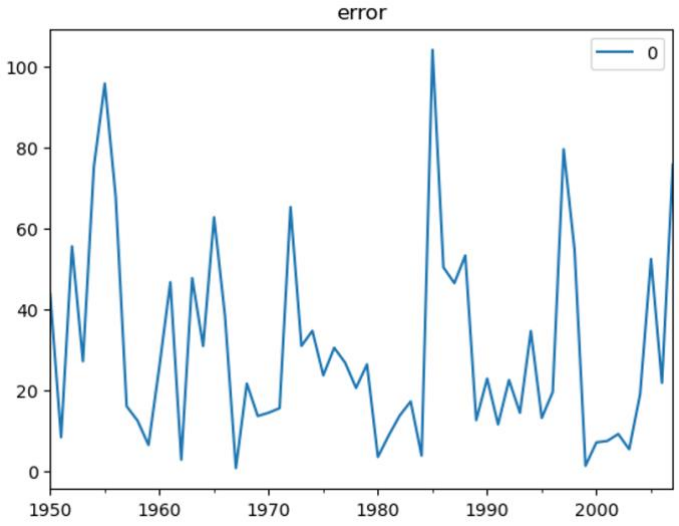


Figure 13: Error for Sun activity

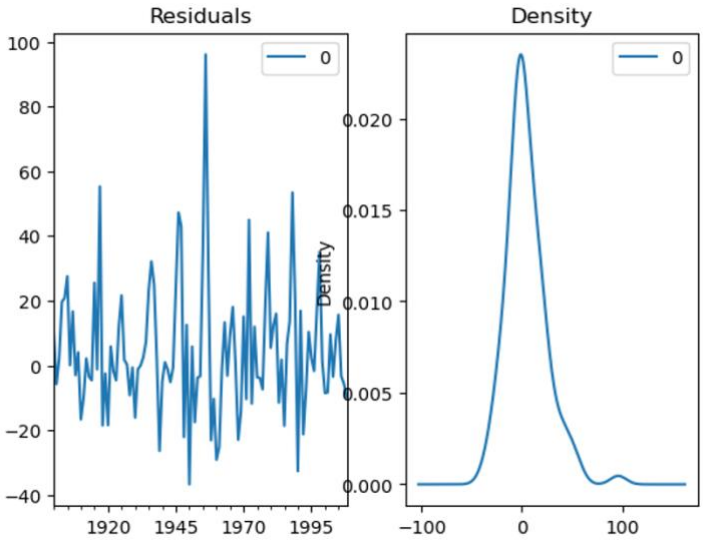


Figure 26: Residual and Density for Sun activity

3.1.11 SARIMA (Seasonal Autoregressive Integrated Moving Average)

The SARIMA model is a generalization of the ARIMA model with a seasonality component. When seasonality and trend are determined in a data set, the SARIMA model is used to eliminate both seasonality and trend for the data set and gather the stationary data.

3.1.11.1 Amount of air passengers prediction using SARIMA

This experimental study determines a time series prediction with Seasonal Autoregressive Integrated Moving Average (SARIMA) algorithm. The data named Air Passengers is the number of passengers from January 1949 to December 1960 monthly. The data is seasonal since it has similar repetitive patterns and it has a trend.

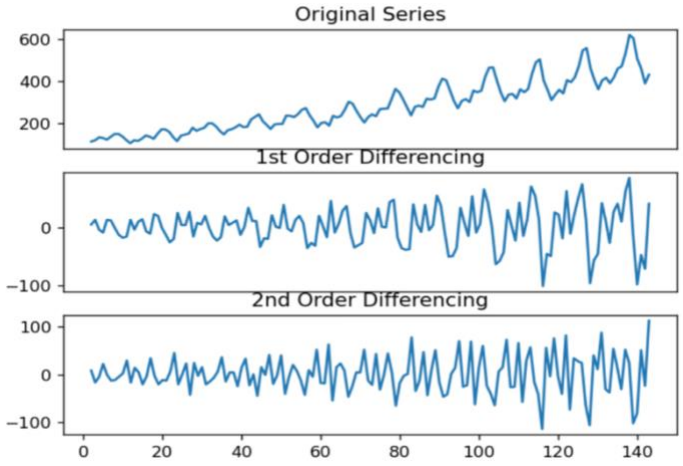


Figure 27: Original, First, and Second Differentiation Graphs for Air Passengers Data

The first order differencing is stationary so d is chosen as 1.

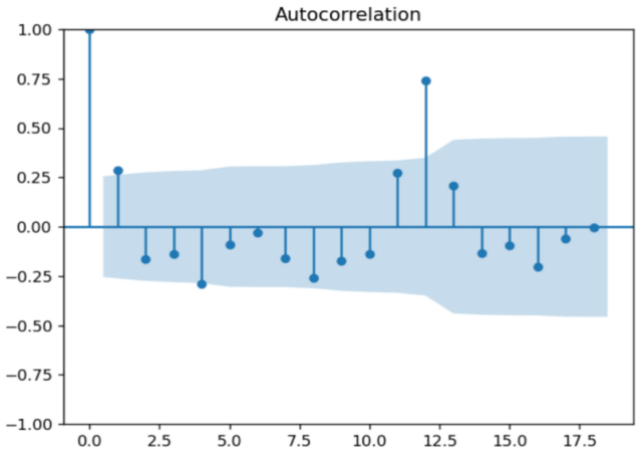


Figure 14: *ACF Graph for Air Passengers Data*

The q parameter is selected as 2 based on Figure 25.

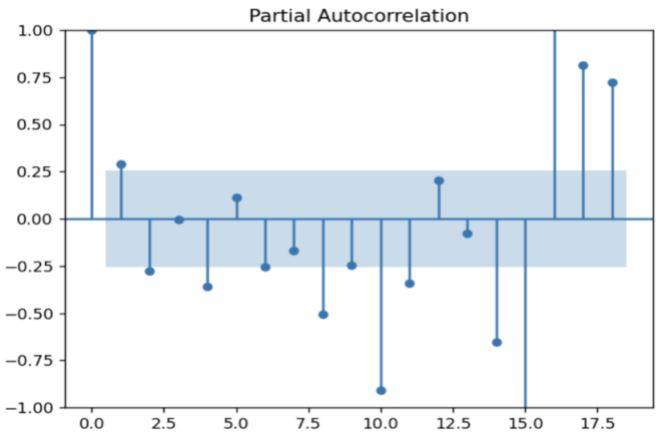


Figure 15: *PACF Graph for Air Passengers Data*

The p parameter is selected as 2 based on Figure 29.

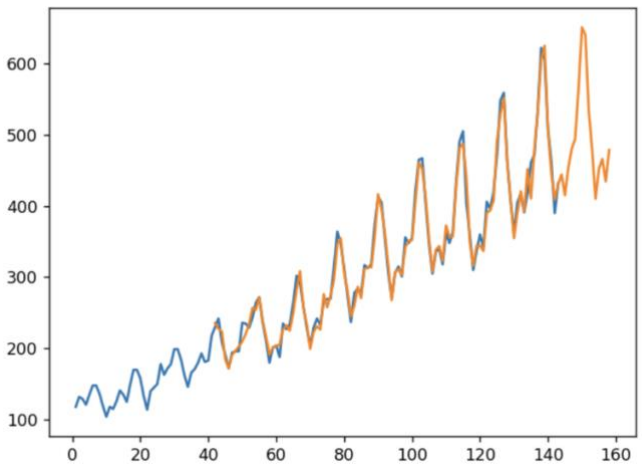


Figure 16: *The Actual (Blue) and Predicted (Orange) Graphs of Air Passengers Data from January 1949 to December 1960*

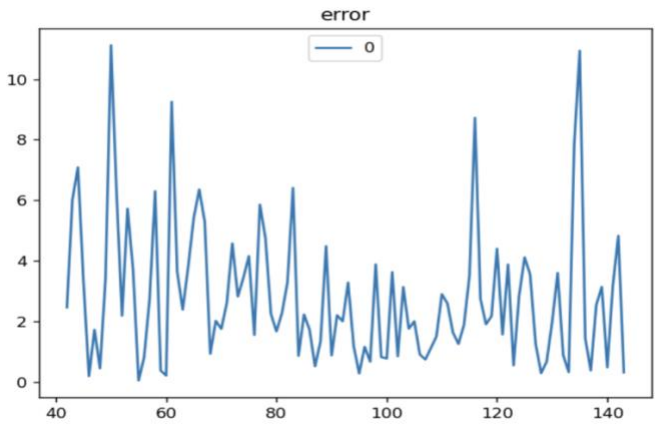


Figure 17: *Percentage Error of Air Passengers Prediction*

The percentage of error is calculated by dividing the difference between the actual data and predicted data, and actual data then multiplying the result by one hundred in Figure 31.

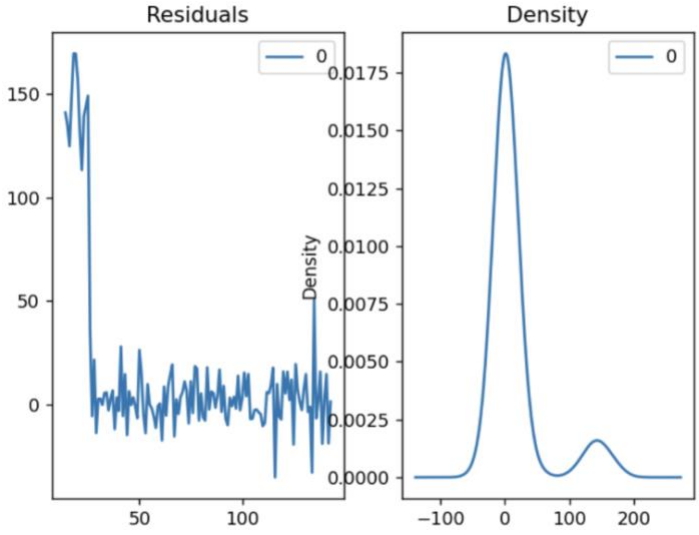


Figure 18: *Residual and Density Graphs for Air Passengers Data*

3.2 Machine Learning Prediction Models

3.2.1 Training and learning

In machine learning, "training" is the process where a model “learns” the underlying patterns of the data through exposure to a training dataset. This is an iterative process where, at each iteration, the model makes a prediction, compares it to the actual output, and then adjusts the model parameters to minimize the prediction error. This

optimization process is accomplished via techniques like gradient descent and is commonly measured by a loss function. In short, the model generalizes unseen data.

3.2.2 Hyperparameters

Hyperparameters are parameters that are set before training a machine learning model. Unlike model parameters, they cannot be learned from the data during training and must be predefined. They represent higher-level properties of the model such as complexity, speed of learning, or the capacity to prevent overfitting. The correct choice of hyperparameters can significantly improve the performance of a model. (Goodfellow, Bengio & Courville, 2016)

In the context of regression models used in time series analysis, hyperparameters can include:

- Learning rate or step size in gradient-based optimization algorithms.
- Weight Decay or Regularization is a regularization technique applied to the weights of a neural network.
- Regularization parameters, such as the regularization strength in Ridge and Lasso Regression, or dropout rates in Neural Networks.
- Number of layers and number of units per layer in Neural Networks.
- Initialization parameters such as the initial weights in Neural Networks.
- Tree-specific parameters in decision tree-based models like Random Forests or Gradient Boosting Machines, e.g., maximum depth of the trees, minimum samples per leaf, and so forth.
- The number of components in models like PCA.
- Kernel parameters in Kernel methods like the SVR.
- ARIMA/SARIMA parameters (p, d, q) and (P, D, Q, m), which respectively define the order of the autoregressive part, the degree of differencing, and the order of the moving-average part of the model.
- Size of the sliding window in window-based models.
- Batch size is the number of training samples to work through before the model's internal parameters (weights and biases) are updated.

- The number of epochs is a hyperparameter that determines how many times the learning algorithm will work through the entire dataset.

The best set of hyperparameters is usually found by means of techniques such as grid search, random search, Bayesian optimization, or genetic algorithms (Bergstra, Bengio, 2012)

It's important to note that each model has its own set of hyperparameters, and there's not a one-size-fits-all approach for setting them. The choice of hyperparameters depends heavily on the characteristics of the data and the problem to be solved.

3.2.2.1 Bayesian optimization

Bayesian optimization is a sequential design strategy for the global optimization of black-box functions that does not assume any functional forms. Instead, it builds a posterior distribution of functions to efficiently update the surrogate model as new information becomes available (Shahriari et al., 2016)1.

The Bayesian optimization process works in the following way:

Surrogate Model: Bayesian optimization begins by defining a surrogate probabilistic model for the objective function. Gaussian Processes (GPs) are often used for this purpose due to their ability to provide uncertainty estimates along with predictions. The surrogate model gives us a belief about the objective function, which is updated with each new observation.
Acquisition Function: An acquisition function is then defined over the surrogate model, which determines the utility of evaluating the objective function at any given point. The acquisition function trades off between exploiting areas of the input space where the surrogate model predicts high performance and exploring areas where the predictions are uncertain. There are several choices for the acquisition function, such as “Expected Improvement”, “Probability of Improvement”, and “Upper Confidence

Bound". Once a point is selected and evaluated, the surrogate model is updated to incorporate the new observation.

This procedure is repeated, choosing points that maximize the acquisition function until a stopping criterion is met, like reaching a maximum number of iterations or achieving a satisfactory performance level.
In the context of hyperparameter optimization in machine learning, Bayesian optimization provides an efficient methodology to navigate the high-dimensional space and find the optimal hyperparameters.

3.2.3 Decision tree

A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Decision trees are commonly used in operations research, specifically in decision analysis, to help identify a strategy most likely to reach a goal, but are also a popular tool in machine learning.

There are several factors that can influence the number of branches that are created in a decision tree. These include:

- The complexity of the data: If the data is complex and has many features, then the decision tree may need more branches to accurately classify the data.
- The level of detail desired: If a more detailed analysis is needed, then the decision tree may need more branches to provide a more granular understanding of the data.
- The size of the data set: A larger data set may require more branches to accurately classify the data.
- The accuracy desired: If a high level of accuracy is desired, then the decision tree may need more branches to ensure that the data is accurately classified.
- The goal of the analysis: The desired outcome of the analysis may also influence the number of branches needed in the decision tree. For example, if the goal is to identify patterns in the data, then a simpler decision tree with

fewer branches may be sufficient. However, if the goal is to make accurate predictions, then a more complex decision tree with more branches may be necessary.

3.2.4 Random forest regressor

The random forest algorithm was originally introduced by Breiman (Breiman, L.,2001) and it is a robust and versatile ensemble learning method. The time series regression analysis for an algorithm is defined as each decision tree is constructed using a bootstrap sample of the data. Each node is separated using the best among a subset of predictors randomly chosen at that node. A population of uncorrelated trees that operate as a committee to make decisions is created.

The major benefits of using Random Forest for time series regression include its ability to handle complex nonlinear relationships, its robustness against overfitting due to ensemble learning, and its capacity to provide variable importance measures. However, one must bear in mind that the algorithm is essentially "blind" to the temporal order of observations unless explicitly provided with lagged variables (Granger et al., 1976).

3.2.4.1 The hyperparameters of random forest regressor

- The number of estimators hyperparameter determines the number of decision trees in the forest.
- The criterion hyperparameter defines the function that is used for evaluating the split quality in the tree.
- The minimum samples split hyperparameter determines the minimum number of samples that are required in a node for a split to be attempted in the tree.
- The minimum sample leaf hyperparameter determines the minimum number of samples required at a leaf node.
- The maximum features hyperparameter indicates the number of features to consider while searching for the best split.
- The maximum depth hyperparameter determines the maximum depth which is the maximum distance between the root and any leaf of each tree.

3.2.5 Linear regression

Linear regression is a statistical method used to model the linear relationship between a dependent variable and one or more independent variables. It is based on the idea of finding a line that best fits the data points, using the least squares method.

The general form of a linear regression model can be written as:

$$Y = \sum_{i=0}^{n} \beta_i X^i + \varepsilon \quad (3.12)$$

Where:

- Y is the dependent variable (also called the outcome or response variable)
- X_1, X_2,..., X_n are the independent variables (also called explanatory variables or predictors)
- β_0 is the intercept, the expected value of Y when all X variables are equal to 0
- β_1, β_2,..., β_n are the coefficients, representing the change in Y for a unit change in X_1, X_2,..., X_n , respectively
- ε is the error term, representing the difference between the actual value of Y and the predicted value of Y

3.2.6 Hyperparameters of linear regression

- The fit intercept hyperparameter determines if the intercept is used for this model or not. The intercept indicates the value of a dependent variable when all independent variables are zero
- Copy X hyperparameter determines if X is copied or not before fitting the best linear regression model.

3.2.7 Non-Linear Regression

Non-linear regression is a type of regression analysis in which the response variable is modeled as a nonlinear function of one or more predictor variables. It is a method used to model a relationship between a dependent variable and one or more independent variables, where the relationship is not a straight line.

Examples of non-linear relationships include exponential, logarithmic, and polynomial functions. Non-linear regression can be more difficult to fit than linear regression, and

it may require more data and more complex statistical techniques to obtain an accurate model. However, non-linear regression can provide a better fit to the data in certain situations, such as when the relationship between the dependent and independent variables is non-linear.

One example of non-linear regression is the use of polynomial regression, in which the response variable is modeled as a polynomial function of one or more predictor variables. Polynomial regression can be used to model relationships between variables that are not well-represented by a straight line.

3.2.8 Support vector regression (SVR)

The Support Vector Regression (SVR) algorithm is a supervised learning algorithm that uses the principles of Support Vector Machine (SVM). The support vectors are the training examples that are closest to the hyperplane. Because these data points impact the orientation and position of the hyperplane they are named as "support vector".

3.2.8.1 The hyperparameteters used for SVR algorithm

- Kernel indicates the kernel function used in prediction. The SVR algorithm is binary, linear, and non-probabilistic but the modeling of non-linear relationships is possible using a kernel. The kernel options are stated below:
    - Linear creates SVM linear. It is used for linearly separable data.
    - Poly allows SVM to use a polynomial function of the specified degree as the kernel. This can fit more complex data, but it can also fit more than necessary.
    - Radial Basis Function. It creates a boundary around the data points based on their distance from the origin.
    - Sigmoid introduces non-linearities in decision boundaries.
- C is the penalty parameter for the error term.
- Epsilon is the Epsilon-tube within no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value.

3.2.9 Artificial neural networks (ANN)

For linear data analysis, the stochastic estimation methods in the previous section are suitable. However, to analyze non-linear time series data, these models are poor and insufficient.

Deep learning is a subset of machine learning and relies on deep artificial neural networks (ANNs). In nonlinear models, artificial neural networks occupy a very important place, and deep learning has received a lot of attention lately. An artificial neural network is a layered structure of connected neurons, inspired by biological neural networks. It is not one algorithm but a combination of various algorithms which allows us to do complex operations on data. Each neuron receives input from other neurons or external sources and applies a transformation to this input to produce an output. The output of one neuron is then passed as input to other neurons in the next layer, and so on until the final output is produced. The structure of an ANN is designed to recognize patterns and relationships in input data, allowing it to make intelligent decisions and predictions. One key aspect of ANNs is that they are able to learn from data, rather than being explicitly programmed to perform a specific task. This is done by adjusting the weights of the connections between neurons based on the input data and the desired output. Through this process, the network can learn to recognize patterns and relationships in the data that it was not initially programmed to recognize.

In our study, all neurons in ANNs will hold different functions with different weights.

ANN Architecture is given below.

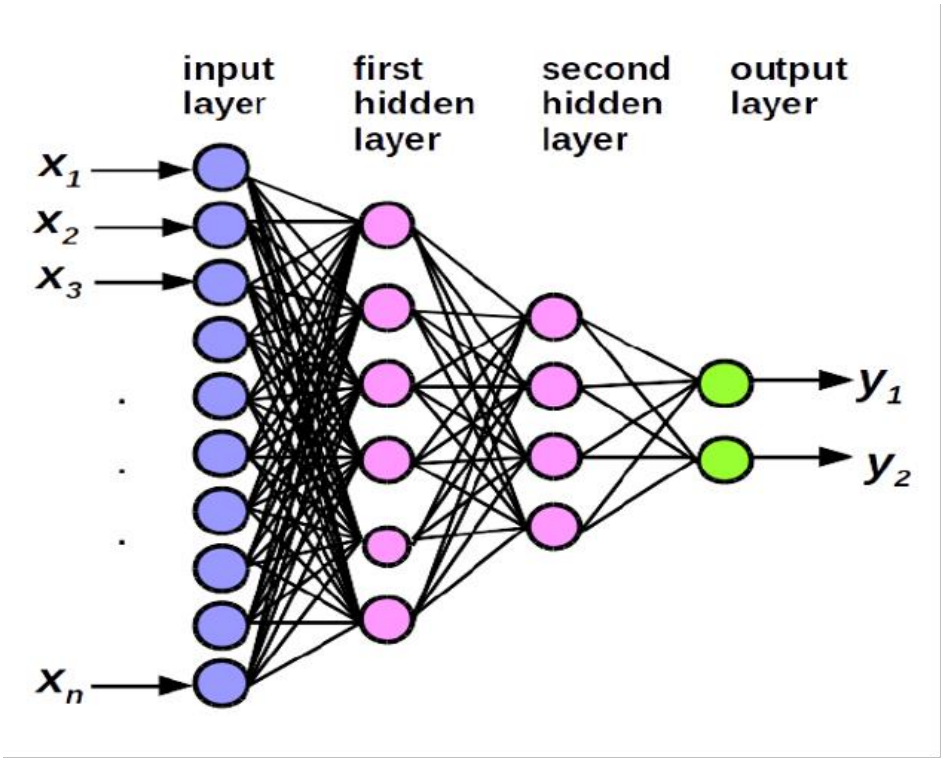


Figure 19: ANN Structure

3.2.9.1 Learning

Learning in artificial neural networks is the process by which the network's internal parameters, or weights, are adjusted to minimize the difference between the predicted output and the true output, with the goal of accurately predicting future inputs. This process is typically done through the use of an optimization algorithm, such as stochastic gradient descent, which iteratively adjusts the weights in a way that reduces the loss function, a measure of the difference between the predicted and true outputs.

3.2.9.1.1 Cost Function

The cost function associates the cost value and the differentiation between actual values and predicted values. It leads to minimizing the parameter values which are used to measure and increase the accuracy of the model. There are several types of cost functions where the most common type is Mean Squared Error (MSE) which means the average of the squared differences between the predicted and actual values.

The Formula of the MSE Cost Function

$$E = \frac{1}{m} \sum_{l=1}^{m} (y_l - \hat{y}_l)^2 \qquad (3.13)$$

The other types of the cost function are as follows, Mean Absolute Error (MAE) and Huber Loss Function.

3.2.9.1.2 Back Propagation

Backpropagation is an algorithm used to train artificial neural networks, which are a type of regression model. It is a method of training a neural network by adjusting the weights and biases of the network to minimize the error between the predicted output of the network and the true output.

The backpropagation algorithm works by propagating the error backward over the network, starting with the output layer and working its way back over the hidden layers. For each layer, the algorithm calculates the gradient of the error with respect to the weights and biases of that layer and adjusts the weights and biases accordingly to minimize the error.

3.2.10 Multi-layer perceptron (MLP)

The Multi-Layer Perceptron (MLP) is a class of artificial neural network that consists of multiple layers of neurons in a directed graph. Each layer is fully connected to the next one, meaning that every node in a layer is connected to every node in the next layer. The nodes of these layers perform computation and transfer information from the input layer to the output layer, often via one or more "hidden" layers in between. (Bishop, 2006).

For each neuron, an MLP takes a weighted sum of inputs, adds a bias, and then passes the result through a nonlinear activation function.

The learning in MLPs is trained on a dataset where the true output is known for each input. Training involves iteratively adjusting the “weights” and “biases” of the MLP to minimize the discrepancy between the predicted and actual output. This is typically done using a technique called backpropagation in conjunction with an optimization algorithm such as stochastic gradient descent. (Rumelhart, Hinton, & Williams, 1986).

Backpropagation involves computing the gradient of a loss function with respect to the weights and biases in the MLP, then updating the parameters in a way that reduces the loss.

References

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. Nature, 323(6088), 533-536.

3.2.10.1 weights and biases

In the domain of machine learning, particularly in the context of artificial neural networks, weights signify the strength of the connections between neurons in adjacent layers of a neural network and biases are additional parameters in the network that enable the model to fit the data more flexibly. Weight determines the amount of influence the input will have on the output. A larger weight implies that the input has a more impact on the output, while a smaller weight suggests a less significant influence.

The process of learning in a neural network involves finding the optimal set of weights and biases that minimizes the discrepancy between the network's predictions and the actual data. This process is typically performed using an iterative optimization algorithm such as backpropagation for computing gradients. (LeCun et al., 2015)

3.2.11 The training process of an ANN model

The training process of a machine learning algorithm involves the concepts of "batches" and "epochs".

3.2.11.1 Epoch

An "epoch" is completed when the entire dataset is presented to the model. The number of epochs is the number of how many times the learning algorithm will work through the complete dataset. The number of epochs is a hyperparameter that determines how many times the learning algorithm will cycle through the entire dataset. It is noteworthy that too few epochs can result in underfitting of the model, while too many epochs can lead to overfitting.

The training process is repeated for a specified “number of epochs". There is a trade-off in determining the number of epochs. If the number of epochs is too few, it may leave the model “underfit” and not adequately trained. When there are too many numbers of epochs, could lead the model to become “overfit” to the training data, thereby performing poorly on unseen data.

This cycle of processing batches for numerous epochs constitutes the core of the training process in machine learning, with the ultimate goal of obtaining a model that can generalize well to unseen data.

3.2.11.2 Batch

In the field of machine learning and, more specifically, in the process of training deep neural networks, a "batch" refers to the subset of the dataset that is used for a single update of the model parameters during the learning process. The batch size, thus, dictates the number of training instances used in one forward and backward pass of the specified model. The selection of an appropriate batch size is often a balance between computational efficiency and the stability and generalization of the learning process. Larger batch sizes can leverage the benefits of hardware accelerations and perform more efficient matrix computations. However, they can also lead to less noisy estimations of the gradient and potentially poorer generalization performance.

3.2.11.3 Optimization methods

Optimization, in the context of machine learning, refers to the process of fine-tuning a model to improve its performance on a given task. More broadly, optimization is a branch of mathematics focused on finding the best solution from a set of possible choices, given certain constraints (Bertsekas, 1999).

In machine learning, the objective of the optimization is usually to minimize a loss function, which quantifies the difference between the model's predictions and the true data (Goodfellow, Bengio & Courville, 2016). The optimal solution is the one that makes this loss as small as possible.

3.2.12 Activation Functions

Activation Functions are mathematical equations that activate the output of a neuron. They play a crucial role in the neural network's ability to capture complex patterns and make accurate predictions.

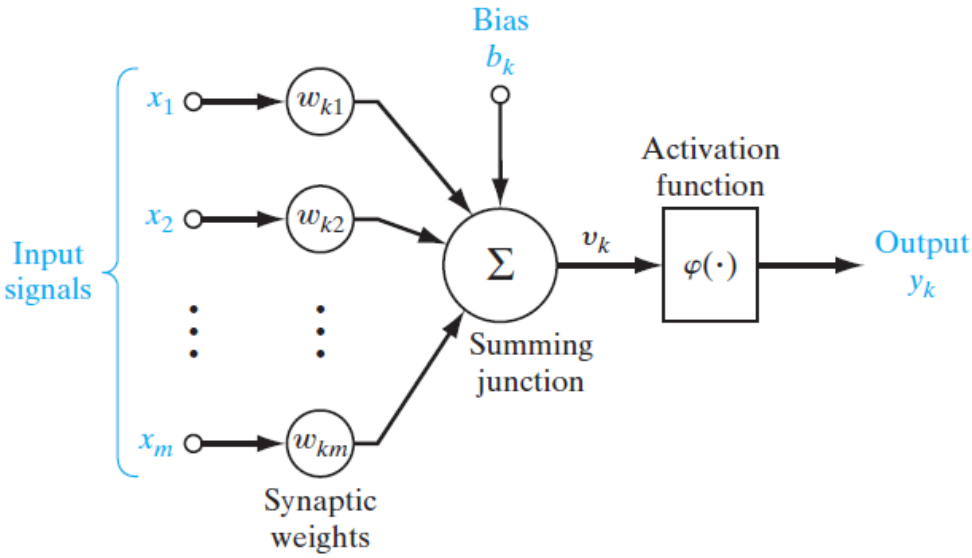


Figure 20: Neuron Schematic with Activation Function

An activation function for limiting the amplitude of the output of a neuron. The activation
function is also referred to as a squashing function, in that it squashes
(limits) the permissible amplitude range of the output signal to some finite value.

Activation functions introduce non-linearity into the output of a neuron. This nonlinearity transforms the input signal into an output signal that is used as input for the next layer in the network. This characteristic allows neural networks to model complex patterns and relationships within the data that are not easily captured by linear transformations alone.

Different activation functions are used in different circumstances based on the properties that they exhibit and the requirements of the specific task at hand. Here are a few commonly used activation functions:

Sigmoid Function: The sigmoid function takes a real-valued number and squashes it into a range between 0 and 1. It is often used in the output layer of a binary classification neural network, where the goal is to achieve a binary output.

Tanh Function: The hyperbolic tangent (tanh) function also takes a real-valued number and squashes it into the range between -1 and 1. Like the sigmoid function, tanh is also sigmoidal, but it provides stronger negative inputs due to its larger output range.

ReLU Function: The Rectified Linear Unit (ReLU) function takes a real-valued number and thresholds it at zero to remove negative values. ReLU has gained popularity in recent years because it tends to perform well in practice and promotes sparsity within the neural network, thus leading to a less computationally intensive model. (Haykin page 42) -> Refer de koy

The choice of activation function can significantly impact the performance of a neural network, influencing both the efficiency of training and the quality of results. Therefore, selecting an appropriate activation function is an essential step in the design of a neural network.

3.2.13 Cross Validation

Cross-validation is a resampling procedure used in machine learning to evaluate the generalizability of a model on an independent data set. In standard k-fold cross-validation, the original dataset is partitioned into k equal-sized subsets. Then, the model is trained on k-1 subsets together and the remaining subset is used as the validation set to assess the model's performance. This process is repeated k times, with each of the k subsets serving as the validation set once. The final model performance is taken as the average of the performance.

Loss-Epoch Graph is a visualization tool that plots the value of the loss function against the number of epochs. Each epoch represents a complete pass of the entire training dataset through the model. This graph is crucial in understanding how quickly or effectively the model is learning and identifying issues like “underfitting” or “overfitting”.

3.2.13.1 Overfitting and Underfitting

Overfitting and underfitting are common phenomena in machine learning that refer to the performance of a model on new, unseen data. Overfitting occurs when a model learns the training data too well, capturing not only the underlying patterns but also the noise or outliers in the data. An overfit model exhibits low training error but high validation error and generally performs poorly on unseen data because it has essentially memorized the training dataset instead of learning to generalize from it. Underfitting, on the other hand, occurs when a model is too simple to capture the underlying structure of the data. An underfit model will have high training error and will also perform poorly on unseen data. These phenomena can often be identified using the loss-epoch graph: overfitting is typically indicated by a large gap between

training and validation loss as training progresses, while underfitting is suggested by a persistently high training and validation loss.

3.2.14 Recurrent Neural Network (RNN)

A recurrent neural network (RNN) is a type of neural network that allows previous outputs to be used as inputs while generating the current output. This creates a temporal dynamic in the network, allowing it to model temporal dependencies and make predictions based on time series data or natural language processing tasks (Elman, 1990).

In an RNN, each unit in the network has a hidden state, which is updated at each time step based on the input at that time step and the previous hidden state. The hidden state can be thought of as a memory of the past inputs and outputs, and it allows the network to use this information to inform its current output. RNNs can be unfolded in time, creating a temporal sequence of inputs and outputs. This allows the network to process a sequence of inputs and make a prediction for each element in the sequence. There are several types of RNNs, including the long short-term memory (LSTM) network and the gated recurrent unit (GRU) network, which are designed to address some of the challenges of training standard RNNs.

3.2.14.1 Long short-term memory (LSTM)

Long Short-Term Memory (LSTM) networks are a type of recurrent neural network (RNN) designed to learn long-term dependencies in sequence prediction tasks. LSTM networks were introduced by Hochreiter and Schmidhuber in 1997 as a solution to the "vanishing gradient" problem that can occur when training traditional RNNs.

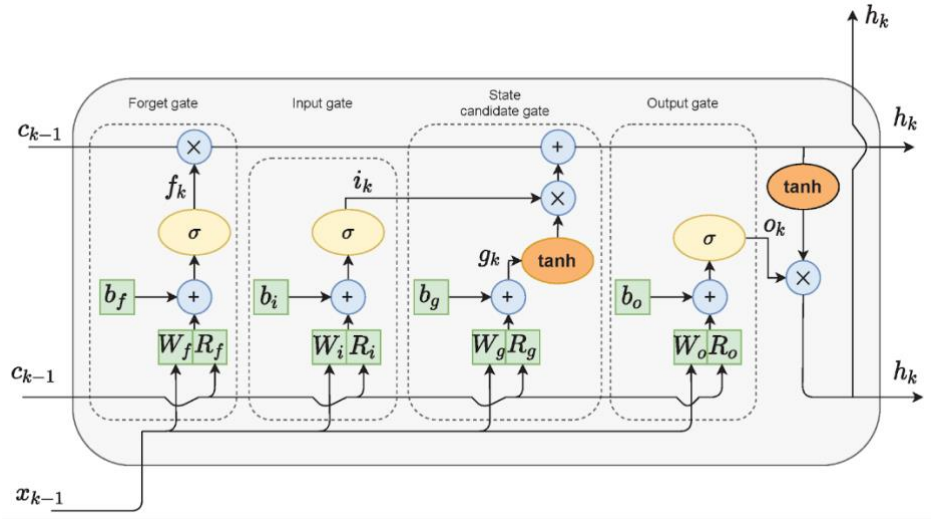


Figure 21: LSTM Structure

- The architecture of LSTM networks is composed of LSTM cells, which, unlike standard feedforward neurons, are designed to remember values over arbitrary time intervals. Each LSTM cell contains an input gate, a forget gate, and an output gate. The cell also contains a memory cell that can store information for either short or long periods, and a self-connected cell state, which allows information to be carried across many time steps without suffering from the vanishing or exploding gradient problem.
- The input gate determines how much of the incoming information from the network input should be stored in the cell state.
- Forget gate controls how much of the information currently in the cell state should be forgotten or kept.
- The output gate decides what the next hidden state should be.
- Training is like other types of neural networks. LSTM networks are trained with a variant of the backpropagation algorithm called backpropagation through time. When applied to LSTMs, this algorithm calculates gradients for the loss function concerning the weights of the LSTM network, which are then used to update the weights. The unique architecture of LSTM cells helps mitigate the vanishing gradient problem, as the cell design allows gradients to flow unchanged if the forget gate is set to remember.

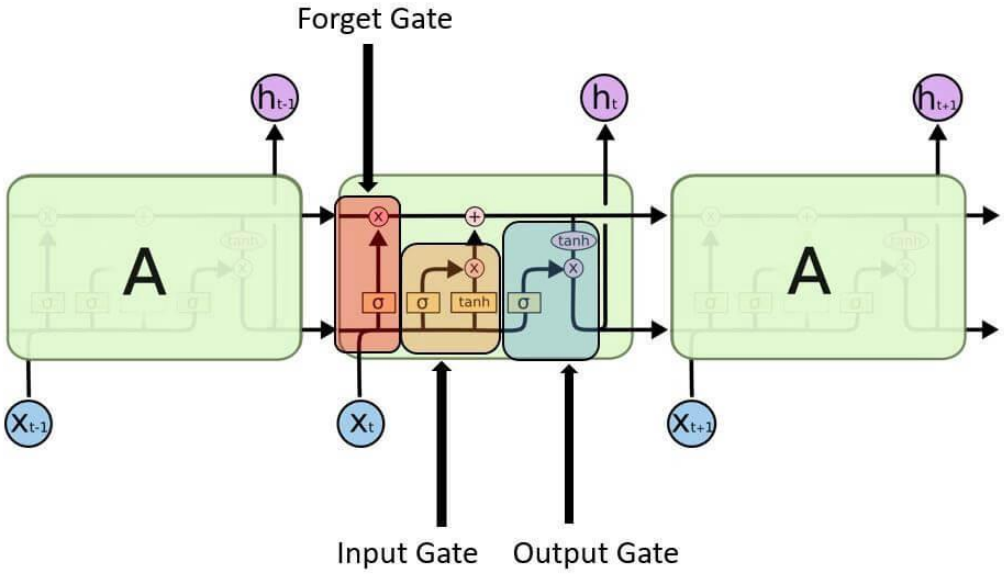


Figure 22: LSTM Sequence

LSTMs have demonstrated state-of-the-art performance on a variety of sequence prediction problems, including time series prediction, handwriting recognition, and machine translation. Their ability to learn long-term dependencies in sequence data makes them highly effective for these types of tasks.

4. EXPERIMENTAL STUDIES

This experimental study covers five different time-series datasets. Stochastic prediction models and machine learning models are applied and compared for time series regression analysis performance. The Bayesian search method is used to tune hyperparameters for all machine learning models. 80% of the datasets are used for training and 20% of the datasets are used for testing. The metrics used for analyzing the performance of the predictions are MSE, RMSE, NRMSE, MAPE, and R squared error.

The experimental result is obtained by using Python packages such as PyTorch, sklearn, statsmodels, pandas, numpy, matplotlib. Visual Studio Code and Jupyter code editors, pip, and conda package managers have been used for coding.

4.1 Female Birth Dataset

The data named Female Birth is the number of births from 01.01.1959 to 12.31.1959 in a district in California. The data has neither trend nor seasonality. Because of that, ARMA is used. In addition, to compare the results of ARMA, linear regression, random forest, and SVR is also used for regression analysis. The input of the machine learning prediction models is one-step lagged data and the mean value of the last three values of the data.

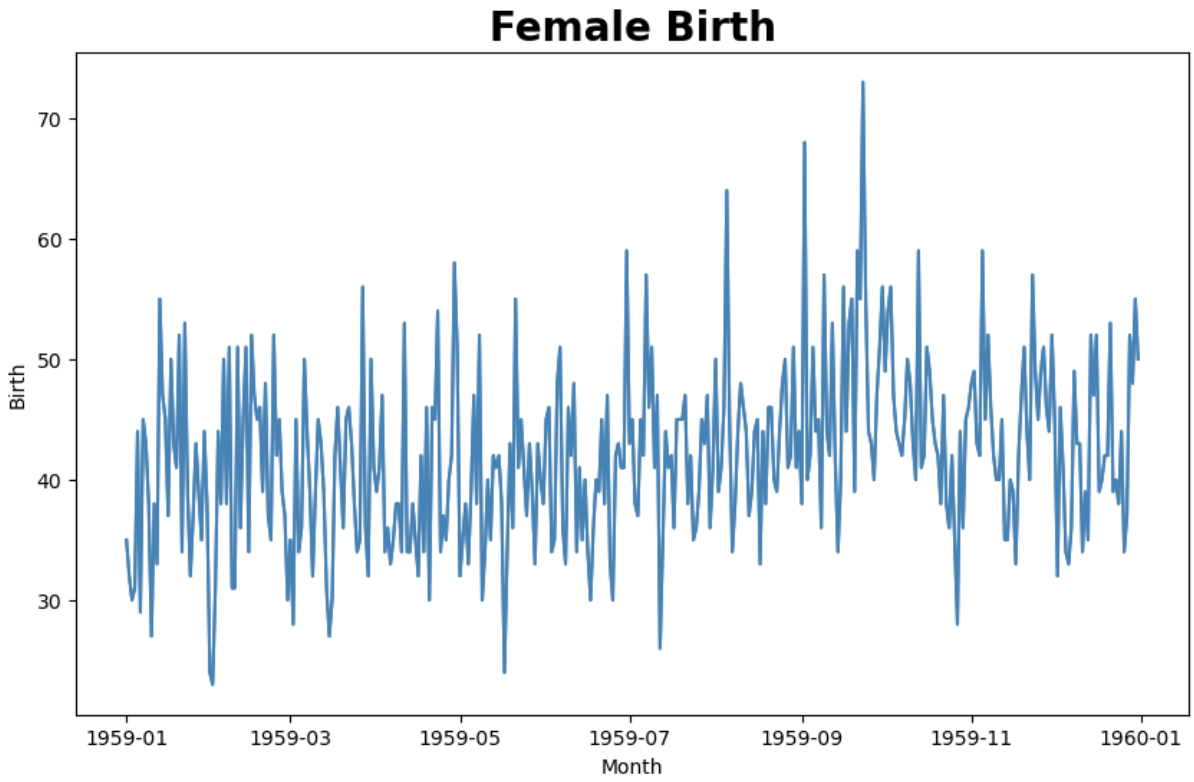


Figure 23

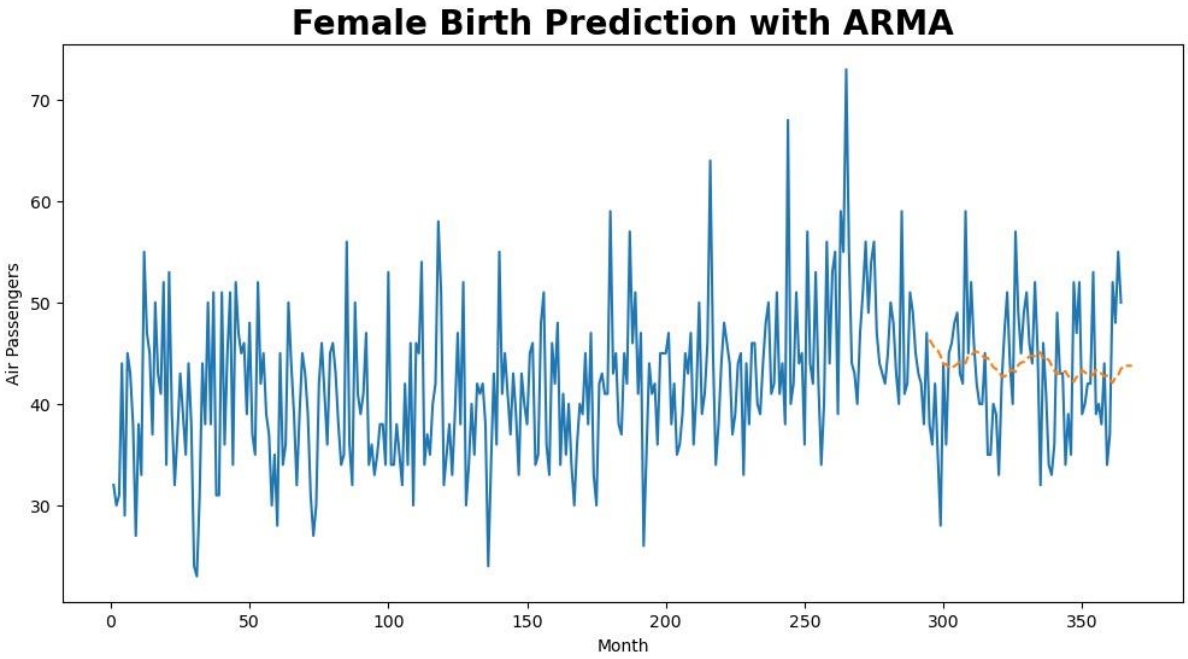


Figure 38

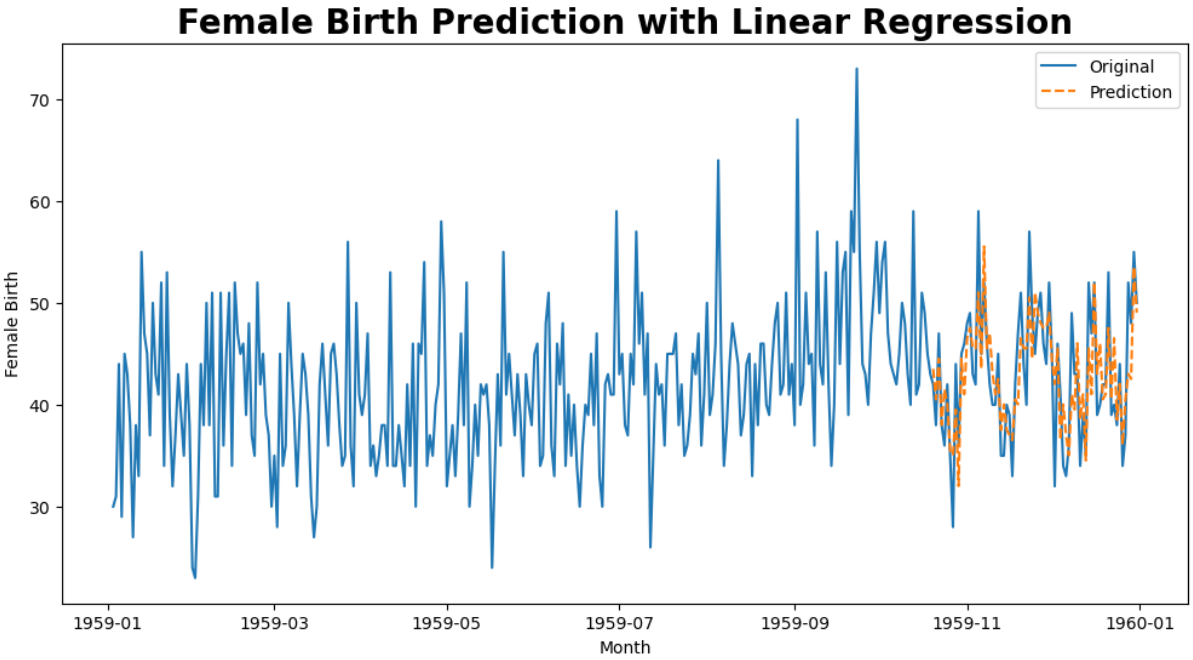


Figure 24

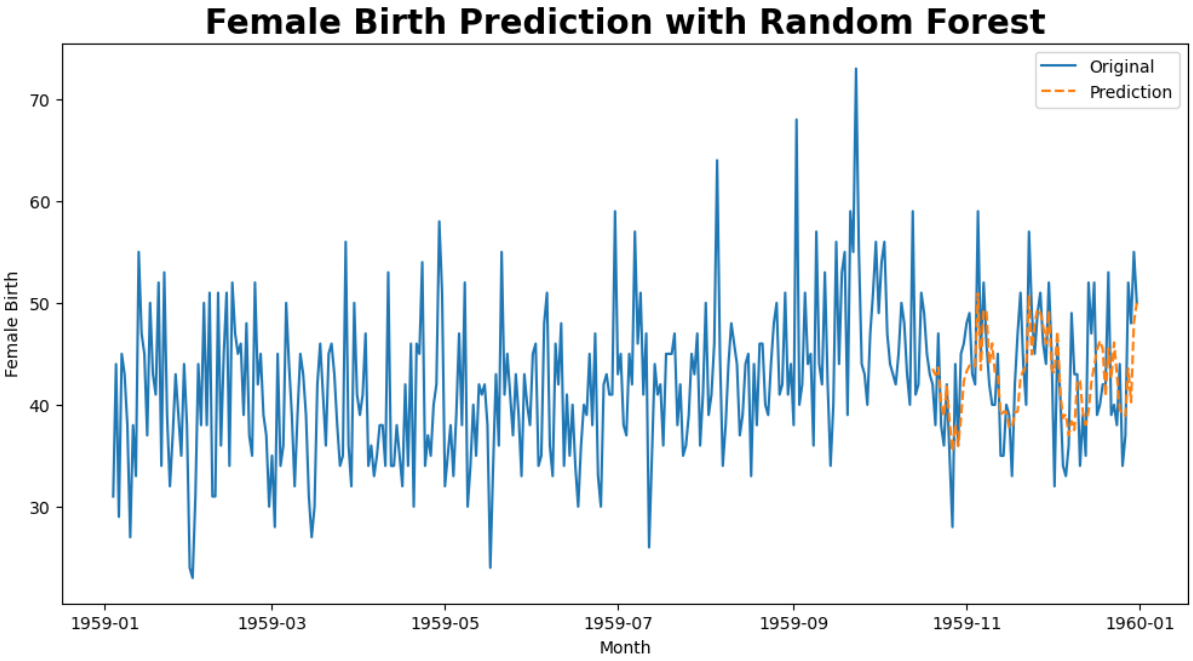


Figure 25

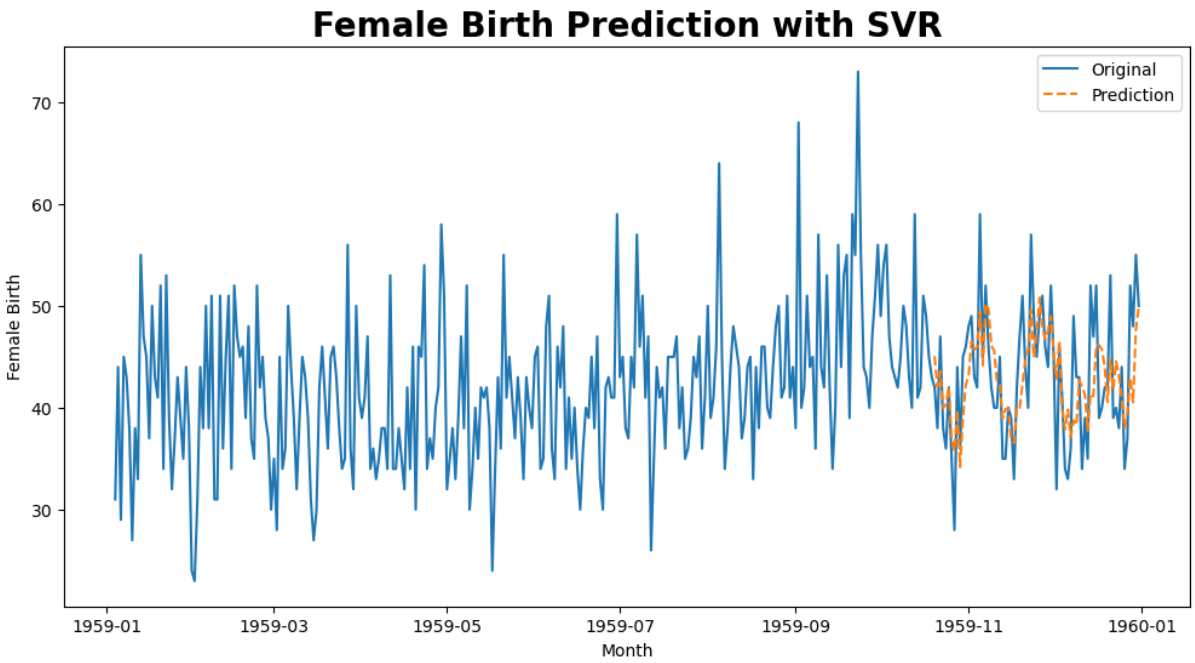


Figure 26

The errors are shown in Table 1.

Models	MSE	RMSE	NRMSE	MAPE	R^2 Error
ARMA	46.2365	6.799	0.2193	0.1337	-0.0898
Linear Regression	16.680	4.0841	0.13174	0.07979	0.595
Random Forest	19.62790	4.430	0.1429	0.08776	0.524
SVR	21.816	4.670	0.1506	0.0875	0.4712

Table 1: Female Birth Prediction Errors

4.2 Shampoo Sales Dataset

This dataset indicated shampoo sales in an unknown area from January 1949 to November 1960 monthly. The dataset has a trend. Because of that ARIMA is used. In addition, to compare the results of ARIMA, linear regression, random forest, and SVR is also used for regression analysis. Before applying the machine learning models, the first differentiation of the data is obtained and removed to eliminate the trend. After the prediction is done the trend is added to the predicted residual. The input of the machine learning prediction models is one and two-step lagged data and the mean value of the last four values of the data.

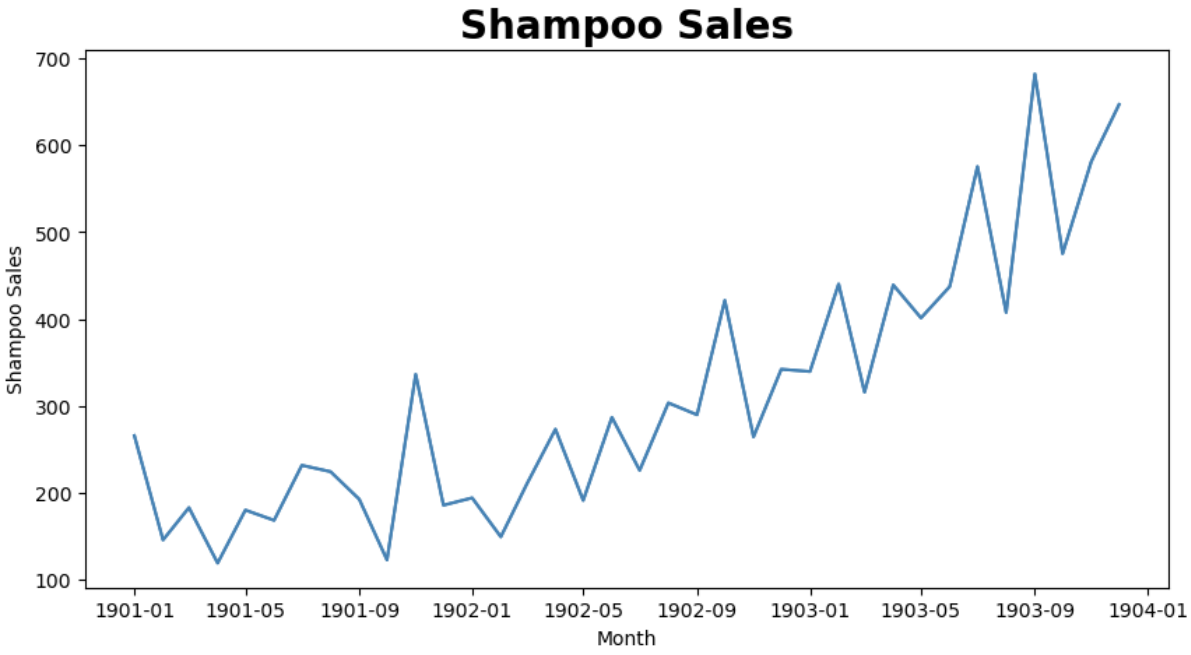


Figure 27

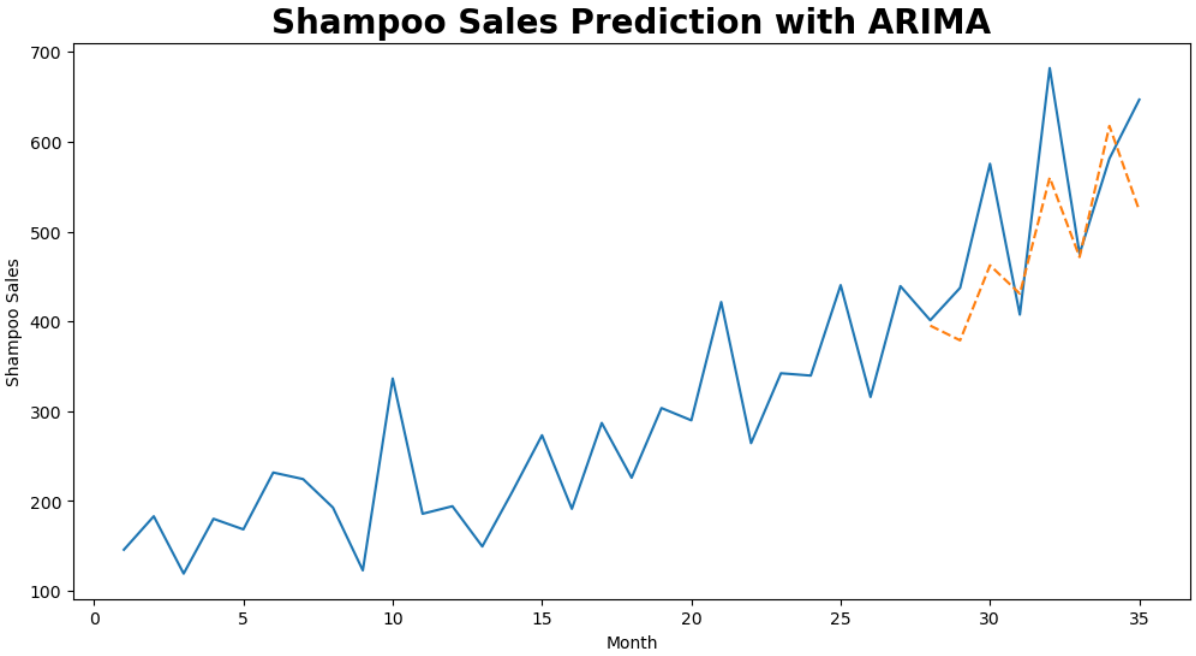


Figure 28

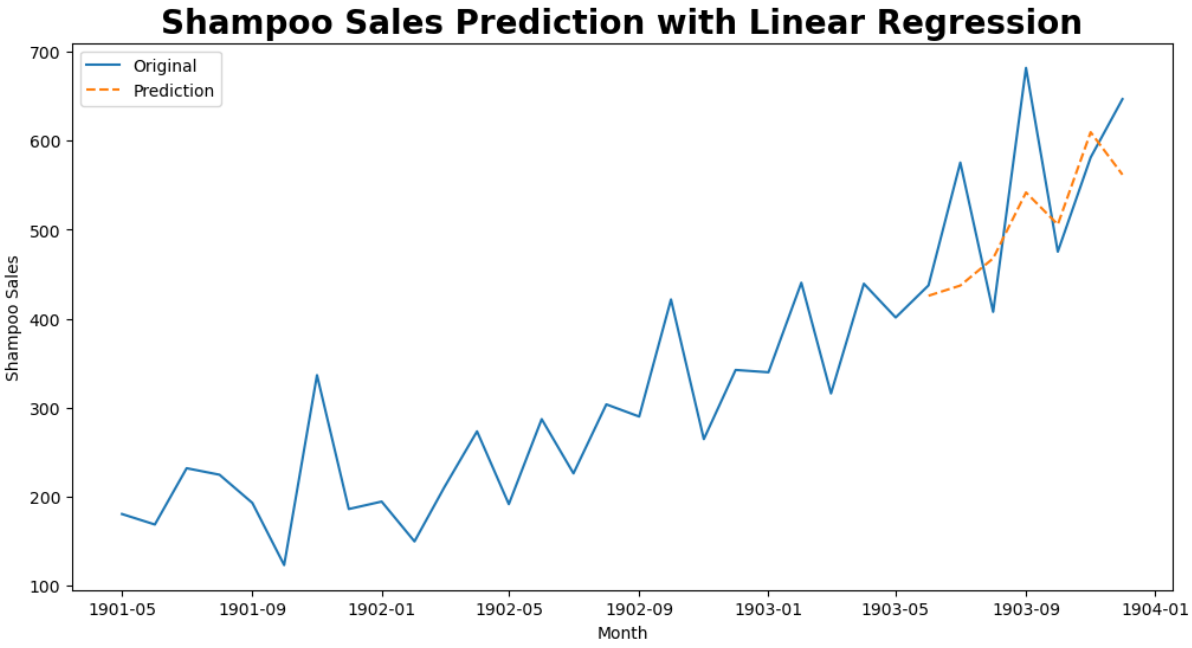


Figure 44

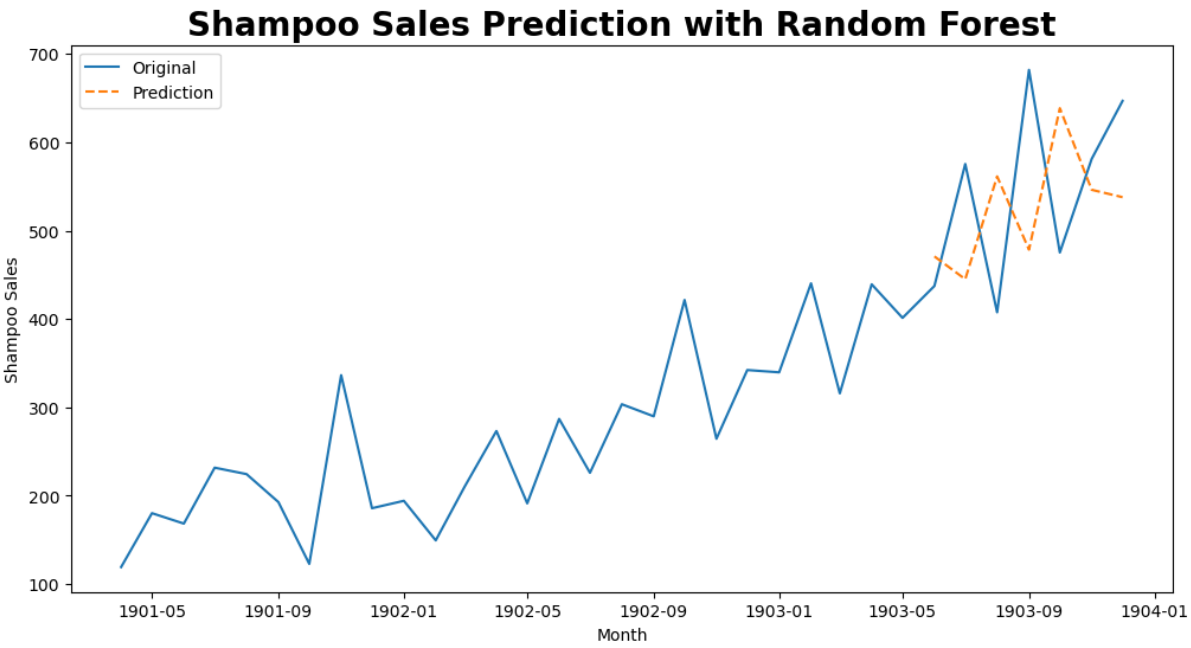


Figure 45

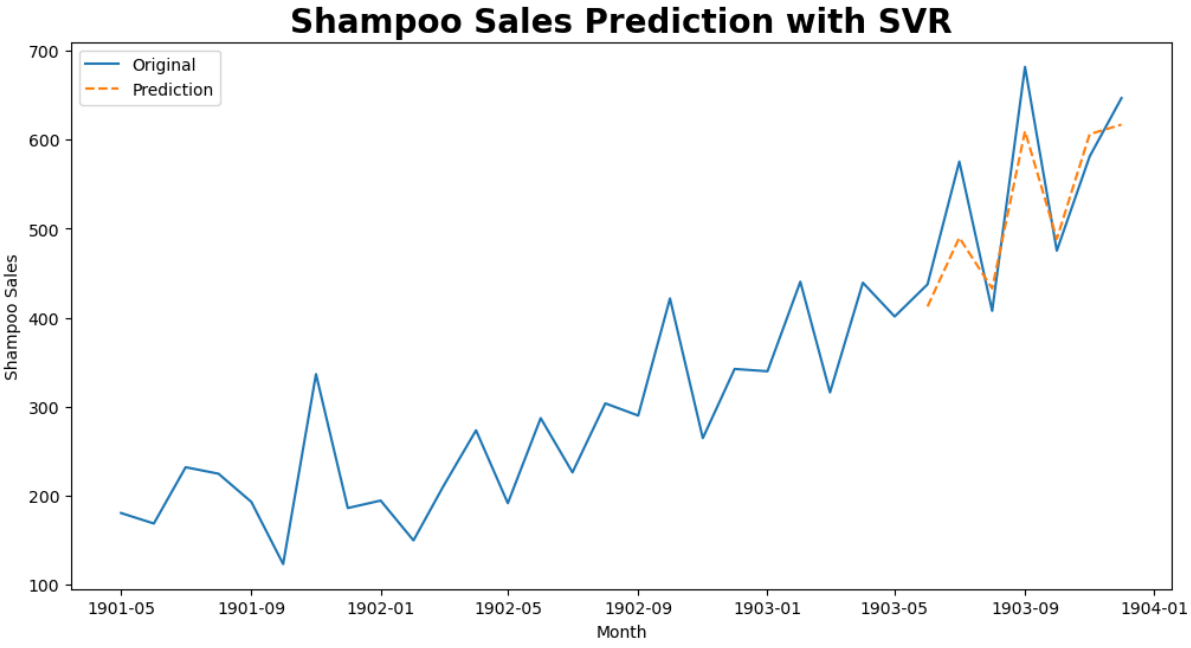


Figure 46

The errors are shown in Table 2.

Models	MSE	RMSE	NRMSE	MAPE	R^2 Error
ARIMA	6027.79	77.638	0.2766	0.1052	0.42898
Linear Regression	7351.522	85.741	0.312	0.1234	0.228
Random Forest	17566.319	132.538	0.483	0.2216	-0.8434
SVR	2214.412	47.0575	0.1715	0.0700	0.7676

Table 2: Shampoo Sales Prediction Errors

4.3 Sun Activity Dataset

The sun activity dataset has seasonality. Because of that SARMA is used. In addition, to compare the results of SARMA, linear regression, random forest, and SVR is also used for regression analysis. Before applying the machine learning models, the seasonality of the dataset is removed. After the prediction is done the seasonality is added to the residual. The input of the machine learning prediction models is one and two-step lagged data and the mean value of the last five values of the data.

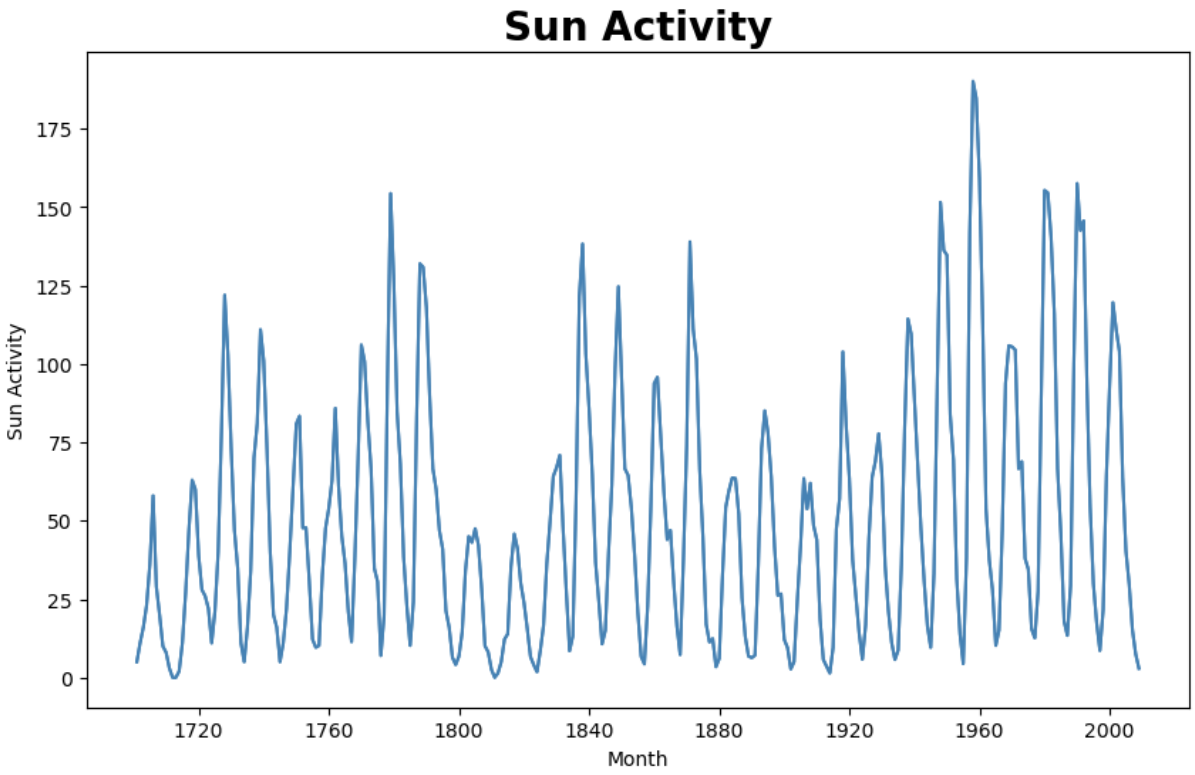


Figure 29

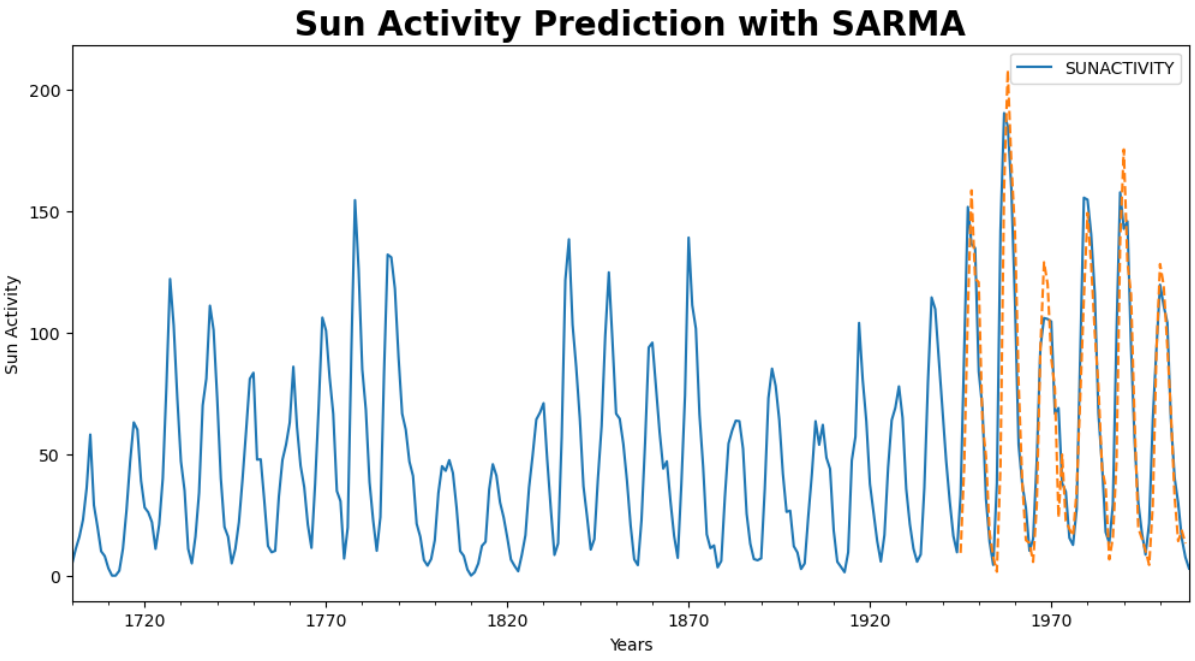


Figure 30

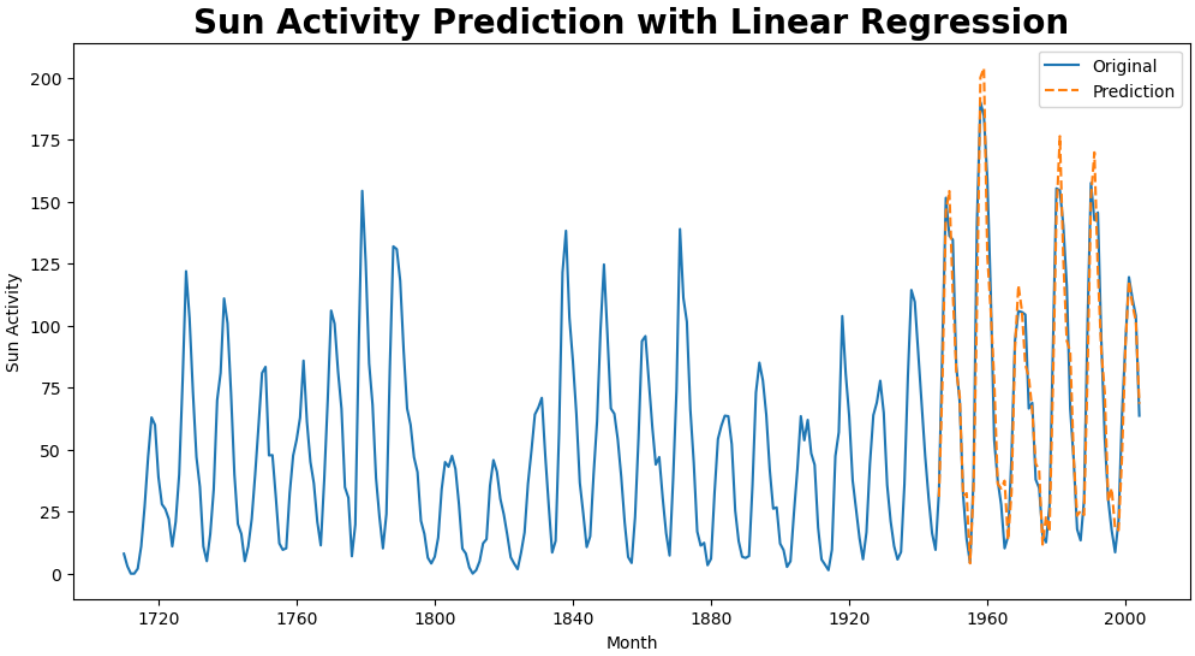


Figure 31

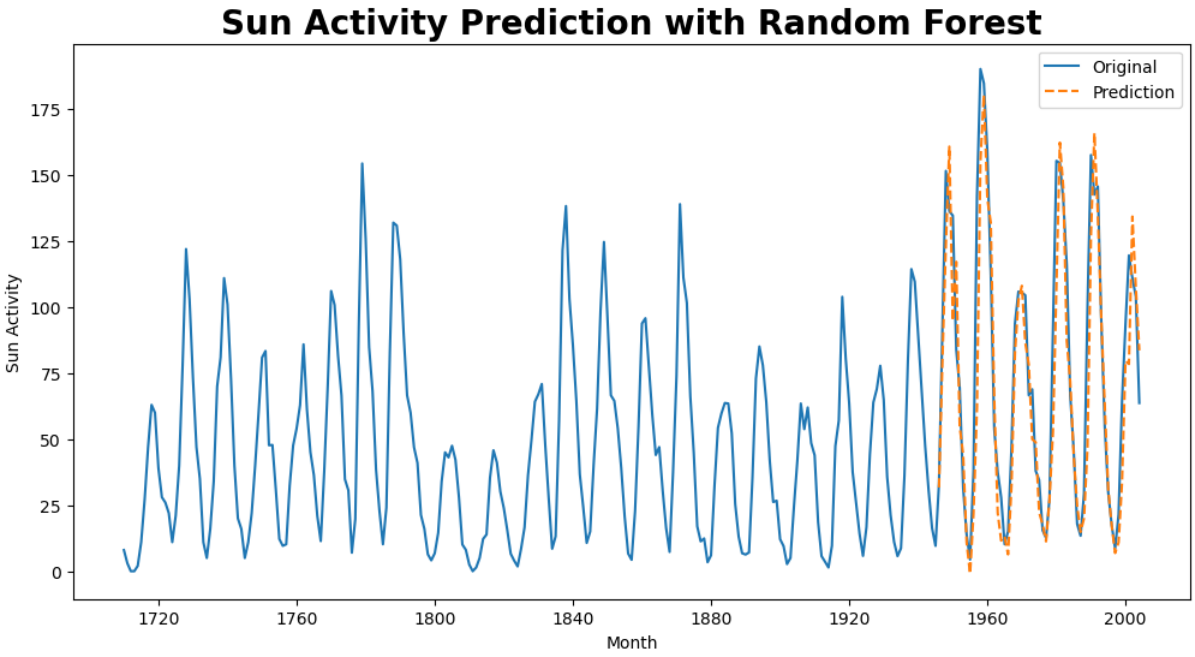


Figure 32

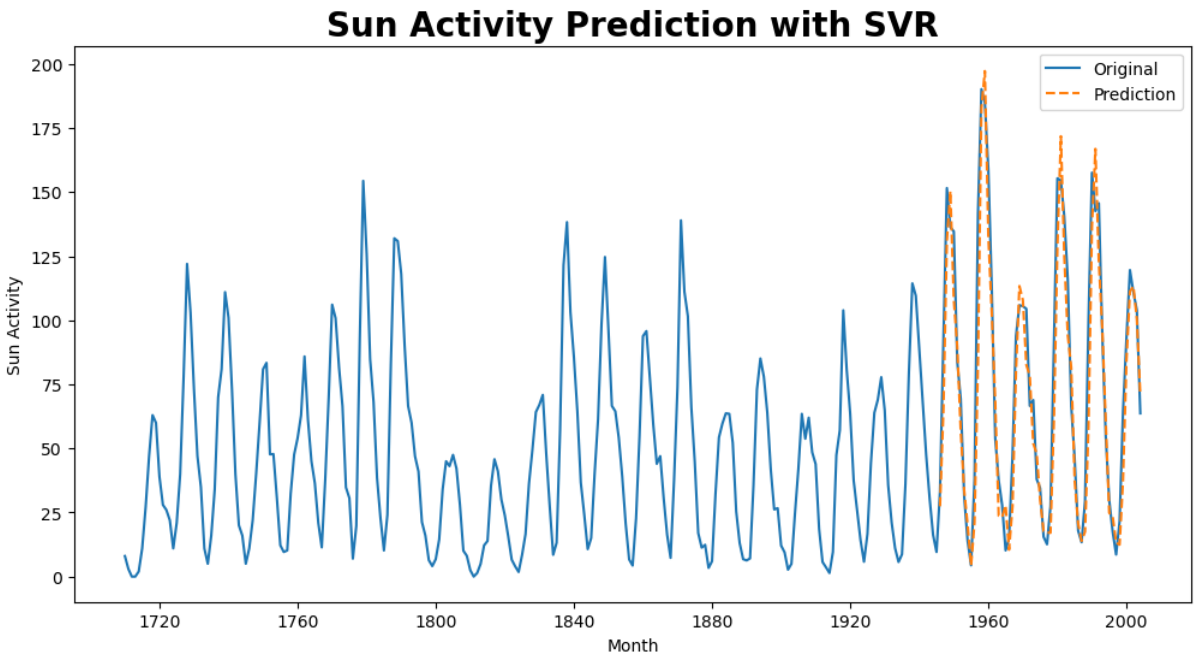


Figure 33

The errors are shown in Table 3.

Models	MSE	RMSE	NRMSE	MAPE	R^2 Error
SARMA	559.233	23.648	0.1272	0.3099	0.7923
Linear Regression	197.139	14.04	0.0755	0.254	0.926
Random Forest	485.6872	22.038	0.118	0.241	0.818
SVR	290.302	17.038	0.0917	0.228	0.8914

Table 3: Sun Activity Prediction Errors

4.4 Air Passengers Dataset

Air passengers are a commonly used dataset for time series regression analysis. The Air Passengers dataset indicates the number of air passengers from January 1949 to December 1960 monthly. Data has obvious trend and seasonality components. Because of that SARIMA is used and machine learning models such as linear regression, random forest, SVR, MLP, and LSTM are applied to the dataset. Figure 52 shows the time series plot of the dataset. The input of the machine learning prediction models are one, two, three-step lagged data and the mean value of last 10 values of the data.

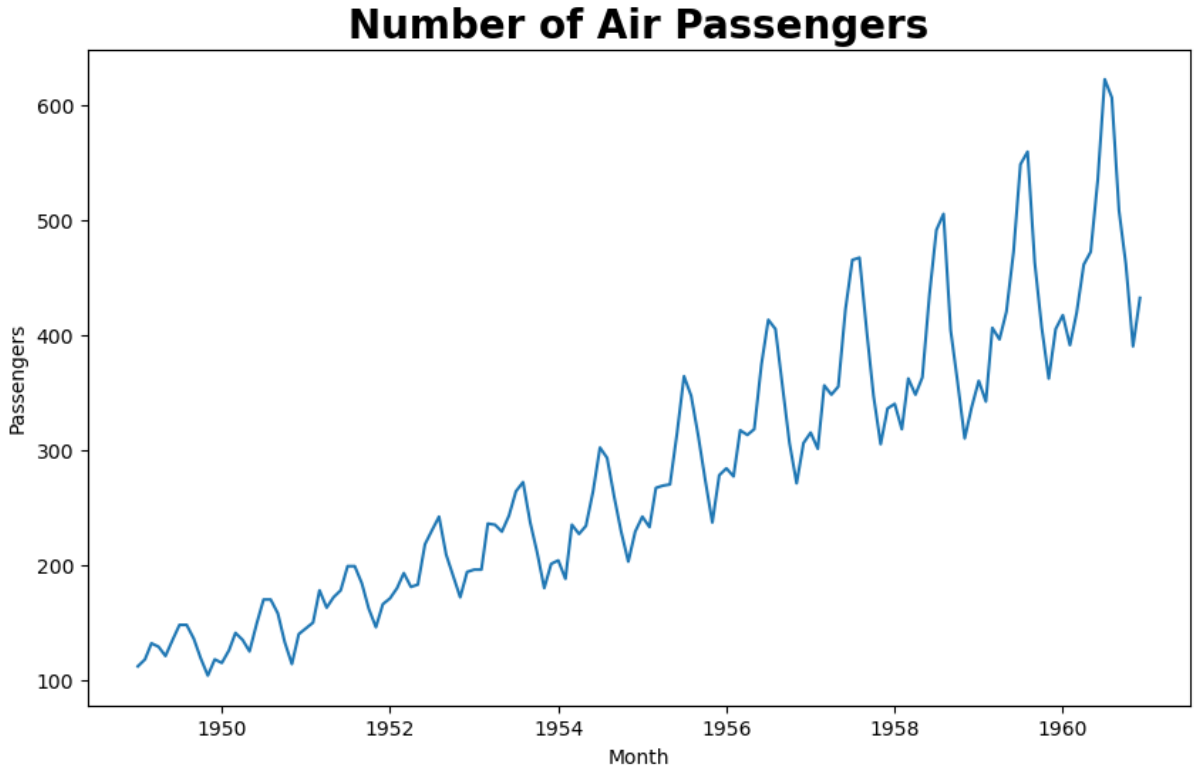


Figure 34

According to Figure 53, yearly seasonality is clearly shown. Since the dataset is monthly there are 12 data in a season. Because of that seasonality is 12. Before implementing machine learning models the trend and seasonality components of the dataset are removed and residual data obtained. Linear regression, random forest, and SVR algorithms were applied to the residuals. The removed trend and seasonality components are added to the predicted residual.

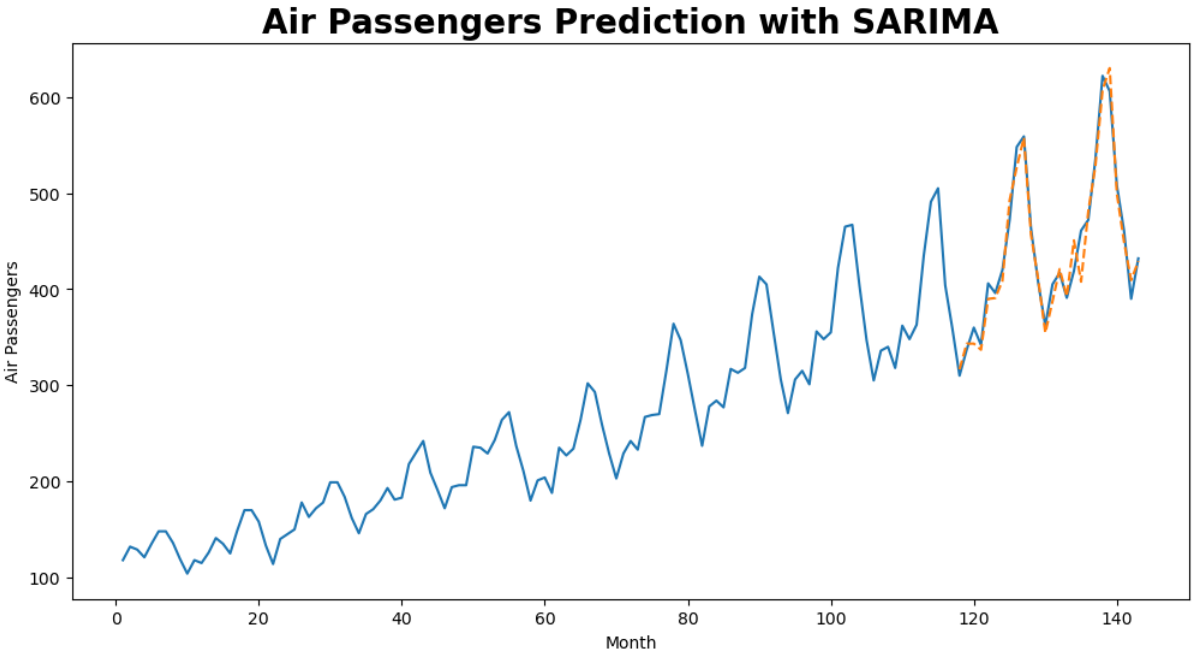


Figure 35

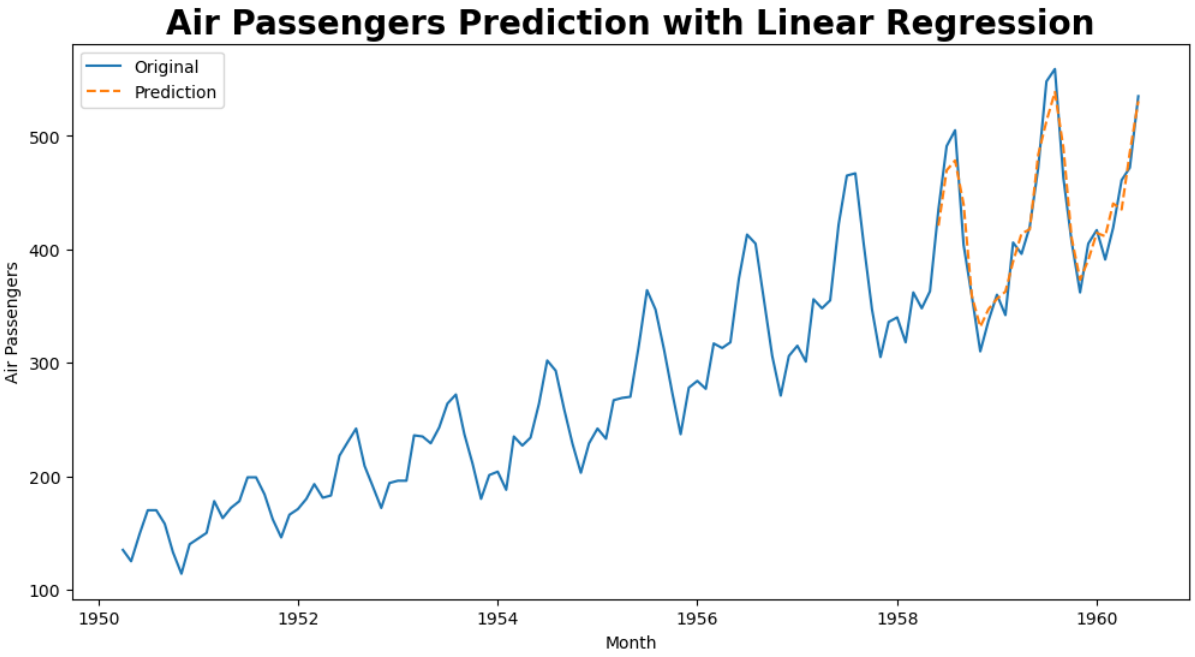


Figure 36

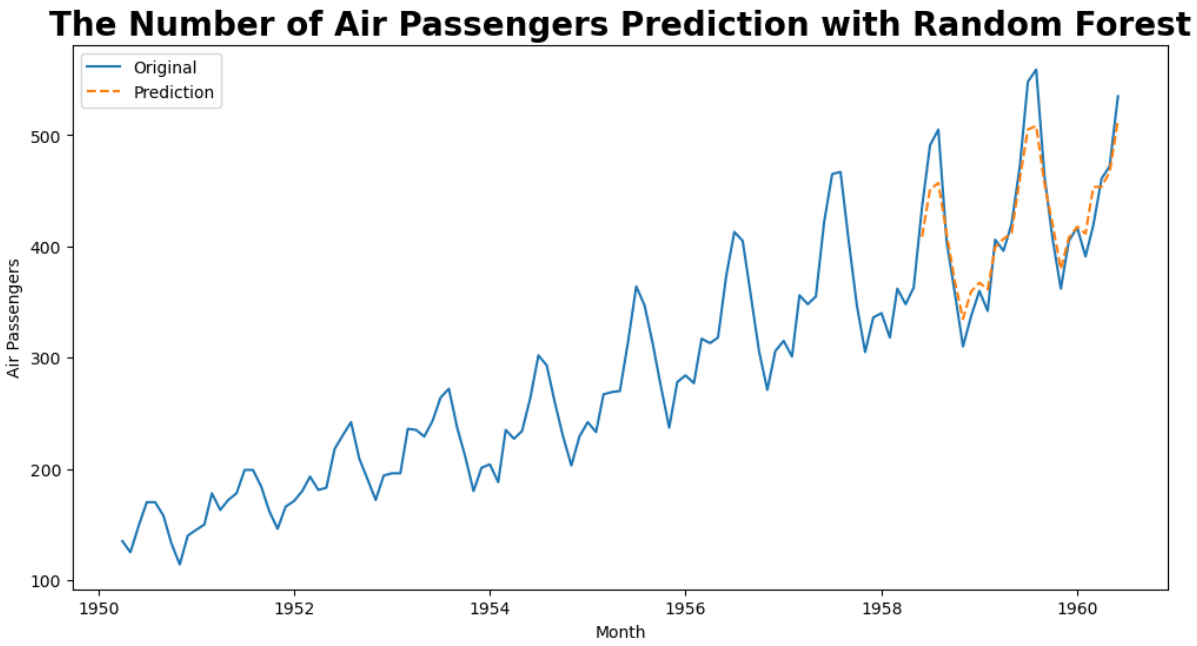


Figure 37

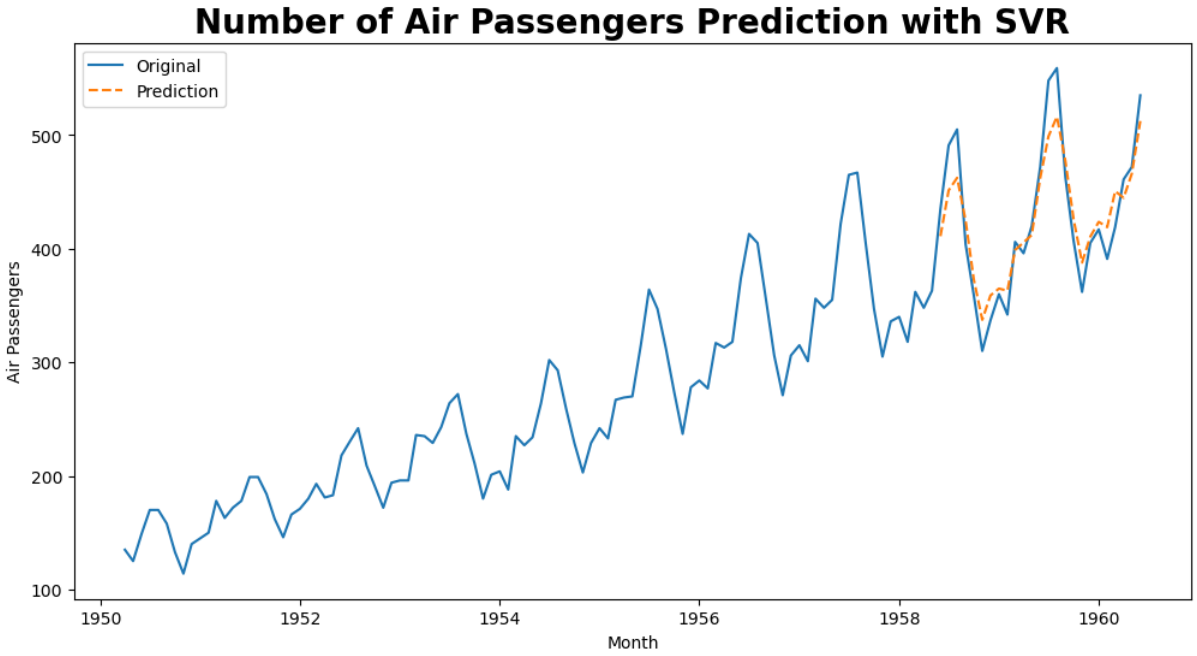


Figure 38

The errors are shown in Table 4.

Models	MSE	RMSE	NRMSE	MAPE	R^2 Error
SARIMA	286.7074	16.9324	0.0542	0.0288	0.9547
Linear Regression	353.983	18.8144	0.0755	0.0375	0.9174
Random Forest	548.9433	23.42954	0.0940	0.04274	0.87192
SVR	589.3219	24.2759	0.097493	0.0480	0.862
MLP	2779.55	52.721	0.1689	0.1033	0.544
LSTM	2241.203	47.341	0.1517	0.087	0.645

Table 4: Air Passengers Prediction Errors

4.5 Bitcoin Price Dataset

4.5.1 Twitter sentiment analysis

4.5.1.1 Understanding bitcoin trends through Twitter Sentiment Analysis

The research study presented here utilizes an interesting method called Twitter Sentiment Analysis for forecasting Bitcoin prices. This strategy involves deciphering the mood or sentiment towards Bitcoin on Twitter, which can be indicative of market movements. Here's a detailed look at the process.

4.5.1.2 Choosing the Right Data

The research study presented here utilizes an interesting method called Twitter Sentiment Analysis for forecasting Bitcoin prices. This strategy involves deciphering

the mood or sentiment towards Bitcoin on Twitter, which can be indicative of market movements. Here's a detailed look at the process.

4.5.1.3 Cleaning up the data

After curating the desired tweets, it was necessary to tidy up the data. This involves eliminating columns that were not needed for the analysis. With a neat and streamlined dataset, the focus was maintained on the tweet content and their respective timestamps.

4.5.1.4 Analyzing sentiment

With the cleaned-up data, the central task of sentiment analysis was undertaken. A tool called “sentiment_analysis_twitter_roBERTa-base” was used for this purpose. This tool excels at discerning the underlying sentiment in each tweet. It yielded a score between 0 and 1 for every tweet, where 0 signified a negative sentiment and 1 pointed to a positive sentiment.

date	text	sentiment
2021-02-10	#Bitcoin’s Rally Signals the Rise of Digital A...	1.0
2021-02-10	If Apple Inc ever makes a move for #Bitcoin as...	0.5
2021-02-10	.@LFDodds has more on Twitter saying it could ...	0.5
2021-02-10	Crypto update:\n#Bitcoin 44726.30 -5.5%\n#Ethe...	0.5
2021-02-10	#Bitcoin’s Rally Signals the Rise of Digital A...	1.0
...	...	...
2021-02-07	Here's your Crypto update:\n #BITCOIN 38057.70...	0.5
2021-02-07	#Bitcoin mining is horrible for the environment...	0.0
2021-02-07	Just keeping printing the money 💷, #inflation ...	0.5
2021-02-07	Online Sleuths Believe Satoshi Nakamoto’s #Bit...	0.5
2021-02-07	#BREAKING: #Bitcoin has fallen from $41k/#BTC...	0.0

Table 5: Sentiment Scores for Each Tweet

4.5.1.5 Averaging out daily sentiment scores

The fast-paced nature of Twitter means thousands of Bitcoin-related tweets get posted every day. To make sense of this vast amount of information, individual tweet scores were not analyzed separately. Instead, the scores of all tweets posted on the same day were averaged. This approach provided an overall sentiment score for each day, which could then be tracked over time.

This process of conducting Twitter Sentiment Analysis provides a unique perspective on predicting Bitcoin prices. As will be discussed further in the article, these daily sentiment scores significantly contribute to the Bitcoin price prediction model.

date	**sentiment-score**
2021-02-05	0.714286
2021-02-06	0.625000
2021-02-07	0.428571
2021-02-08	0.685185
2021-02-09	0.675676
...	...
2022-03-17	0.692308
2022-03-18	0.617424
2022-03-19	0.733333
2022-03-20	0.560976
2022-03-21	0.597938

Table 6: Sentiment Scores Day by Day

4.5.1.6 Filling in the gaps: utilizing the fear and greed index

There were instances of missing data in the selected tweet dataset. This missing data, if left unaddressed, could lead to inaccuracies in the sentiment analysis. To fill in these

gaps, an interesting metric is known as the 'Fear and Greed Index' was used from another dataset that takes a similar approach to sentiment analysis.

The Fear and Greed Index is a commonly used tool in the world of cryptocurrency and traditional finance markets. This index essentially represents the primary emotional drivers in the market – fear and greed. When investors get too greedy, that means the market is due for a correction. On the other hand, when fear takes over, it can often be a buying opportunity.

date	**fear_and_greed_index**
2018-06-01	24
2018-06-02	27
2018-06-03	40
2018-06-04	41
2018-06-05	26
...	...
2023-05-28	50
2023-05-29	52
2023-05-30	51
2023-05-31	51
2023-06-01	52

Table 7: Fear and Greed Index

4.5.2 Google trends analysis

Another valuable resource used in this research study is Google Trends, Google provides insights into the popularity of search terms over time. Specifically, for this study, the search term of interest is 'Bitcoin'.

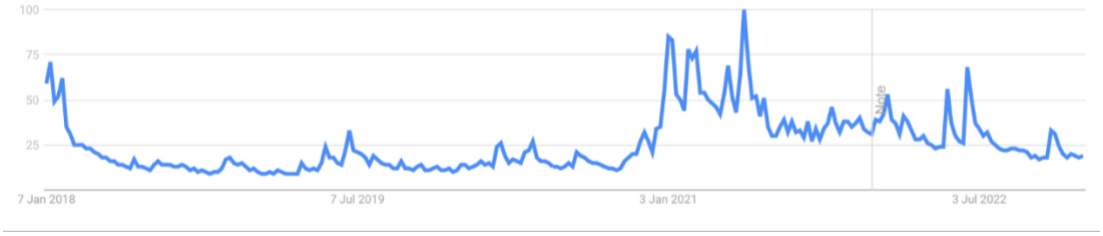


Figure 39: Google Trends Index for 5 years (2018 – 2023)

To gain a holistic perspective on Bitcoin's search popularity, a long-term time frame was chosen. The data spans five years, from June 1, 2018, to June 1, 2023. This broad time frame allows for observing both short-term fluctuations and long-term trends in Bitcoin's search interests.

date	google_trends_value
2018-06-01	7.0
2018-06-02	8.0
2018-06-03	7.0
2018-06-04	7.0
2018-06-05	8.0
...	...
2023-05-28	10.0
2023-05-29	11.0
2023-05-30	11.0
2023-05-31	11.0
2023-06-01	11.0

Table 8: Google Trends

4.5.3 Prediction results

We have conducted a thorough analysis of Bitcoin data, segmented into time lags up to 10 days prior. To support this analysis, we extracted data from a Bitcoin-related tweet dataset from Twitter and prepared it for sentiment analysis through necessary classifications. Following the sentiment analysis, we further bolstered our data with search volume results from Google Trends. This allowed us to understand when Bitcoin garnered more interest and was searched more frequently on Google over the past five years. We utilized this information as an input in our prediction model. In addition to these three inputs, we also included the daily volume of Bitcoin, as well as its daily highest and lowest values, as inputs in our model. We updated the hyperparameters in our LSTM model using the Bayesian Optimization method. We experimented with training our model with different epoch values and determined the weights of our model based on the most suitable results. We performed our price prediction with this model and found that it demonstrated better prediction performance than another model we used, ARIMA, for Bitcoin.

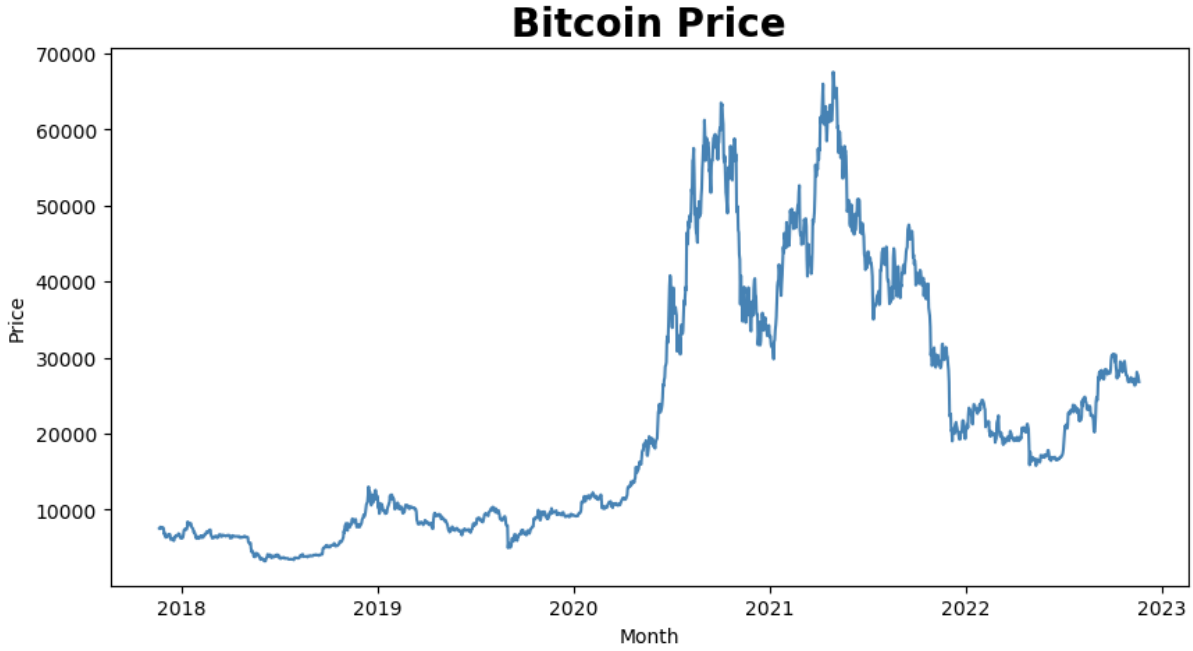


Figure 58

Figure 40: Bitcoin Price Prediction with LSTM

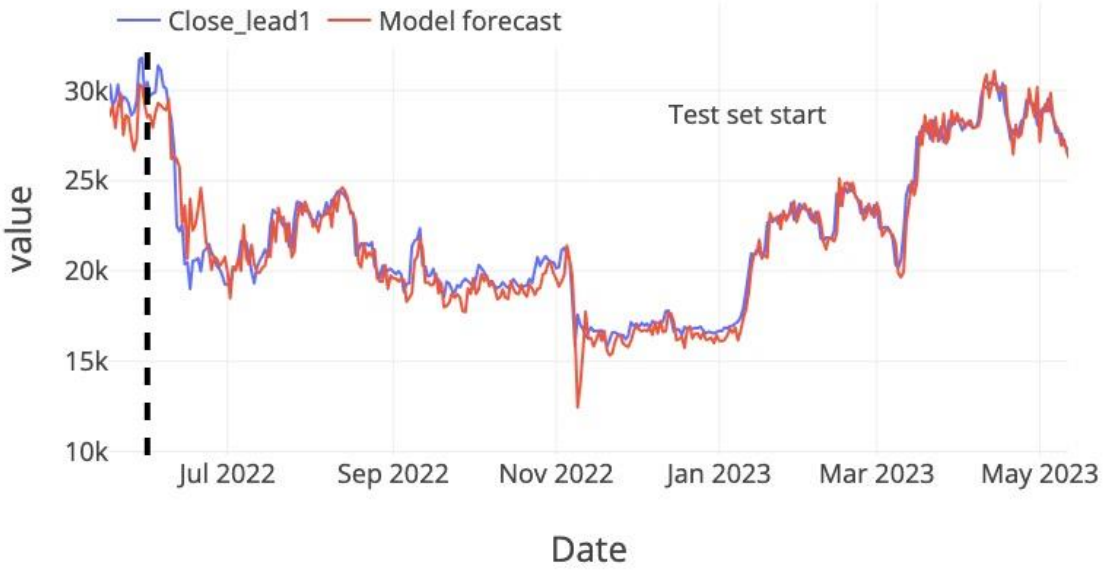


Figure 41: Bitcoin Price Prediction with LSTM (Test Data)

Models	MSE	RMSE	NRMSE	MAPE	R^2 Error
LSTM	999379.4101	999.6896	0.064150	0.0323548	0.939691

Table 9: LSTM Errors

5. CONCLUSIONS AND DISCUSSIONS

This study has meticulously explored the intricacies of time series analysis, demonstrating its utility in predicting future data trends. By examining traditional models like SMA, ARMA, ARIMA, SARMA, and SARIMA, alongside Machine Learning techniques, we have gleaned valuable insights into the strengths and limitations of these methods. Additionally, the study underscores the revolutionary potential of artificial intelligence in enhancing prediction accuracy, specifically in terms of price prediction for various goods and services. The integration of regression analysis, time series analysis, and neural networks illustrate the diversity of approaches in AI for price prediction.

Our empirical work, centered on Bitcoin price prediction, further reinforces the value of this research. We extracted and analyzed Bitcoin data, Twitter sentiment data, and Google Trends data, fusing these distinct data sets into a robust model. The incorporation of additional parameters, such as Bitcoin's daily volume and daily highest and lowest values, contributed to enhancing the depth and precision of our predictions. We deployed an LSTM model, leveraging Bayesian Optimization for hyperparameter tuning, and demonstrated superior prediction performance compared to the ARIMA model. These findings underline the importance of proper data preparation, preprocessing, and the prudent selection of machine learning models. Moreover, they highlight the capacity of machine learning techniques to not only learn from past data but also efficiently forecast future values.

In conclusion, this study illustrates the profound potential of machine learning in time series forecasting. It encourages the continued exploration and refinement of these techniques to bolster their efficacy in various predictive tasks, further empowering data-driven decision-making across myriad domains.

6. REFERENCES

Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time series analysis: forecasting and control. John Wiley & Sons.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). Classification and Regression Trees. Wadsworth.

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

Brownlee, J. (2018). Introduction to time series forecasting with Python: how to prepare data and develop models to predict the future. Machine Learning Mastery.

Burnham, K. P., & Anderson, D. R. (2004). Multimodel inference: Understanding AIC and BIC in model selection. Sociological Methods & Research, 33(2), 261-304.
.

Chatfield, C. (2014). The analysis of time series: an introduction. Chapman and Hall/CRC.

Dama, F., & Kozlíková, B. (2021). Time Series Analysis and Modeling to Forecast: a Survey. ArXiv: Learning. Retrieved from https://arxiv.org/pdf/2104.00164.pdf

Elman, J. L. (1990). Finding structure in time. Cognitive Science, 14(2), 179-211.

Encyclopedia of Statistics in Behavioral Science. (2005). Time series analysis. John Wiley & Sons.

García, S., Luengo, J., & Herrera, F. (2016). Data preprocessing in data mining. Springer.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. Retrieved from http://www.deeplearningbook.org

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.

Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: principles and practice (2nd ed.). OTexts.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112). New York: Springer.

Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: a review and recent developments. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 374(2065), 20150202.

Kelleher, J. D., Mac Namee, B., & D'Arcy, A. (2015). Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies. MIT Press.

Kristoufek, L. (2013). Bitcoin meets Google Trends and Wikipedia: Quantifying the relationship between phenomena of the Internet era. Scientific reports, 3, 3415.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

McNally, S. (2018). Predicting the price of Bitcoin using Machine Learning. arXiv preprint arXiv:1802.04065.

Montgomery, D. C., & Runger, G. C. (2003). Applied statistics and probability for engineers (Vol. 3). John Wiley & Sons.

Nogueira, F. (2014). Bayesian Optimization: Open source constrained global optimization tool for Python. Retrieved from https://github.com/fmfn/BayesianOptimization

Rathan, K., Sai, S. V., & Manikanta, T. S. (2019). Crypto-Currency price prediction using Decision Tree and Regression techniques. 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). https://doi.org/10.1109/icoei.2019.8862585

Schwarz, G. (1978). Estimating the dimension of a model. The Annals of Statistics, 6(2), 461-464

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE, 104(1), 148-175.

Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding machine learning: From theory to algorithms. Cambridge University Press.

Suradhaniwar, S., Kar, S., Durbha, S. S., & Jagarlapudi, A. (2021). Time series forecasting of univariate agrometeorological data: A comparative performance evaluation via one-step and multi-step ahead forecasting strategies. Sensors, 21(7), 2430. https://doi.org/10.3390/s21072430

Time Series Analysis - MATLAB & Simulink. (n.d.). Retrieved from https://www.mathworks.com/help/ident/time-series-model-identification.html

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13, 281-305.

CURRICULUM VITAE

Ad-Soyad **: Kerem ERCİYEŞ**

Doğum Tarihi ve Yeri **: 10.01.2000 İZMİR**

E-posta **:** erciyes18@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2023, İstanbul Teknik Üniversitesi, Eletrik Elektronik Fakültesi, Kontrol ve Otomasyon Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2021 yılında Pavotek firmasında çalıştı.

Ad-Soyad **: Oktay KURT**

Doğum Tarihi ve Yeri **: 14.02.2000 İZMİR**

E-posta **:** kurto18@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2023, İstanbul Teknik Üniversitesi, Eletrik Elektronik Fakültesi, Kontrol ve Otomasyon Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2023 yılında İstanbul Teknik Üniversitesi Artificial Intelligence and Intelligence Systems (AI2S) labında çalıştı.

Ad-Soyad **: Mustafa SOYDAN**

Doğum Tarihi ve Yeri **: 22.06.2000 ORDU**

E-posta **:** soydan19@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2023, İstanbul Teknik Üniversitesi, Eletrik Elektronik Fakültesi, Kontrol ve Otomasyon Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2022-2023 yılları arasında BOSH firmasında çalıştı.

IEEE CODE OF ETHICS: It will be added to the report as it is.

IEEE Code of Ethics

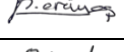
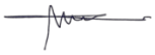
We, the members of the IEEE, in recognition of the importance of our technologies in affecting the quality of life throughout the world, and in accepting a personal obligation to our profession, its members and the communities we serve, do hereby commit ourselves to the highest ethical and professional conduct and agree:

1. **to accept responsibility in making engineering decisions consistent with the safety, health and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;**
2. **to avoid real or perceived conflicts of interest whenever possible, and to disclose them to affected parties when they do exist;**
3. **to be honest and realistic in stating claims or estimates based on available data;**
4. **to reject bribery in all its forms;**
5. **to improve the understanding of technology, its appropriate application, and potential consequences;**
6. **to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;**
7. **to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;**
8. **to treat fairly all persons regardless of such factors as race, religion, gender, disability, age, or national origin;**
9. **to avoid injuring others, their property, reputation, or employment by false or malicious action;**
10. **to assist colleagues and co-workers in their professional development and to support them in following this code of ethics.**

Approved by the IEEE Board of Directors
August 1990.

ETHICAL RULES COMPLIANCE STATEMENT

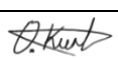
I recognize and accept the basic principles of engineering which stated below.

Kerem	**Erciyeş**	**Signature/Date:**  **10.01.2023**
Oktay	**Kurt**	**Signature/Date:**  **10.01.2023**
Mustafa	**Soydan**	**Signature/Date:**  **10.01.2023**

Engineers; they glorify and develop the integrity, honor and value of the engineering profession by using their own knowledge and skills to increase the welfare of humanity, by serving honestly and impartially to the public, their employers and customers, by striving to increase the ability and prestige of the engineering profession, by supporting the professional and technical unity of their disciplines.

- Engineers will prioritize the safety, health and comfort of the society while performing their professional duties.
- Engineers will only provide service in areas where they are authorized.
- Engineers will only issue objective and realistic reports.
- Engineers will act as reliable attorneys or assistants to the employer or client in professional matters and avoid conflicts of interest.
- Engineers will establish their professional reputation according to the requirements of their services and will not enter into unfair competition with other colleagues.
- Engineers will work to promote and develop integrity, honor and value of profession.
- Engineers will continue their professional development through their own careers and will provide opportunities for the professional development of engineers under their control.

I declare that the parst quoted from any source in this report are less than 15%, and the number of one-to-one quotations in paragraphs is zero.

Kerem	**Erciyeş**	**Signature/Date:**  **10.01.2023**
Oktay	**Kurt**	**Signature/Date:**  **10.01.2023**
Mustafa	**Soydan**	**Signature/Date:**  **10.01.2023**

STANDARDS AND CONSTRAINTS : Add to your report what you are asked below.

Answer the questions about preparing the design project.

1. **What is the design aspect of your project? Explain.**
   This is a repetition of an existing project with developments. In this project, our aspect is to improve existing solutions with extra beneficial features which make some specific data easy to analyze.

2. **Briefly explain what the engineering problem that you have solved in your project and what is your solution on this problem.**
   There is uncertainty in time series data and it is difficult to make predictions. Our solution is analyzing time series and making a prediction with machine learning approach.

3. **Which knowledge that you have learnt and which experiences that you have gained throughout the university education have you used when preparing your project?**

   Probability and Statistics
   Numerical Methods
   Economics
   Programming Technique in Control
   Computer Controlled Systems
   Object Oriented Programming
   State Space Methods in Control Systems

4. **Which modern tools/softwares/programming languages and packages etc. did you use? Briefly explain for what purpose you used them.**

   Python programming language for analyzing time series and machine learning toolbox.
   MATLAB for machine learning tutorials and time series documents.
   statsmodels, Matplotlib, pandas, NumPy libraries.

5. **Do you have any certificate on any other disciplines/topics in addition to the department curriculum? (For example, using online platforms such as CUDA, Udemy, Coursera)**

MATLAB Machine Learning Onramp
MATLAB Deep Learning Onramp
MATLAB Reinforcement Learning Onramp
3Blue1Brown YouTube Channel
freeCodeCamp.org YouTube Channel

6. **What was the engineering standards/norms that you have used and taken into account?**

Institute of Electrical and Electronics Engineers (IEEE)3652.1-2020
Institute of Electrical and Electronics Engineers (IEEE)3333.1.3-2022
International Organization for Standardization ISO/IEC 23053:2022

7. What are the realistic limitations that you have used or taken into account?

a) Economy
Some of the data and indicators are overpriced.

b) Environmental Issues

.....

c) Sustainability
Since there is a huge amount of data, algorithms should be updated regularly.

d) Producibility

.....

e) Ethical Issues
The algorithm can be abused to manipulate the market

f) Health

.....

g) Safety
Data could be manipulated and predictions could be wrong.

h) Social Issues
Because predictions may be wrong, it can cause a loss of real money.

Project Team(Project Executive/Team Leader): Kerem Erciyeş

Assoc. Prof. Dr. Tufan Kumbasar

Metni buraya yazın

Project Topic: Price Prediction With Machine Learning

This project is approved by .....*Project Advisor*..... (.....)

Note : This page can be expanded for the desired constraints if it is necessary