January 2015

Nearest Neighbor Foreign Exchange Rate Forecasting with Mahalanobis Distance

Vindya Kumari Pathirana
University of South Florida, vkumari@mail.usf.edu

Follow this and additional works at: https://digitalcommons.usf.edu/etd

Part of the Geology Commons, and the Mathematics Commons

Scholar Commons Citation
Pathirana, Vindya Kumari, "Nearest Neighbor Foreign Exchange Rate Forecasting with Mahalanobis Distance" (2015). USF Tampa Graduate Theses and Dissertations.
https://digitalcommons.usf.edu/etd/5757

This Dissertation is brought to you for free and open access by the USF Graduate Theses and Dissertations at Digital Commons @ University of South Florida. It has been accepted for inclusion in USF Tampa Graduate Theses and Dissertations by an authorized administrator of Digital Commons @ University of South Florida. For more information, please contact digitalcommons@usf.edu.
Nearest Neighbor Foreign Exchange Rate Forecasting with Mahalanobis Distance

by

Vindya I. Kumari Pathirana

A dissertation submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
Mathematics & Statistics
College of Arts and Sciences
University of South Florida

Major Professor: Kandethody M. Ramachandran, Ph.D.
Yuncheng You, Ph.D.
Gangaram S. Ladde, Ph.D.
Manoug Manougian, Ph.D.

Date of Approval:
July 8, 2015

Keywords: Foreign Exchange Trading, $k$-Nearest Neighbor Algorithm, Mahalanobis Distance, ARIMA, Multi-step Ahead Time Series Forecasting

Copyright © 2015, Vindya I. Kumari Pathirana
Dedication

To my loving parents who always believed in me, who taught me to stay positive, look for good in people, appreciate others for what they can do, and help people without expecting anything back. Where I’m today is all because of them. Thanks for being the best parents in the world, and thanks for always being there for me.
Acknowledgments

My greatest thanks goes to my major professor Dr. Kandethody Ramachandran, for his unlimited guidance, generous support, and countless hours of personal time assisting me throughout my dissertation. Without his guidance this dissertation is impossible.

I also like to thankful to my dissertation committee members, Dr. Yuncheng You, Dr. Gangaram Ladde, Dr. Marcus McWaters, and Dr. Manoug Manougian. My thanks also goes to Dr. Getachew Dagne, chairperson of defense.

Moreover, I would like to thank my family and friends who were always there for me when I needed them the most. Specially Hansi & Manoj for being the best room-mates & friends and Rasika for being a good friend. You guys were always there for me when ever I needed some one at least to talk while I was alone working on my dissertation during the past couple of months.

My special thanks goes to my dear friend Hari for his unfolding moral support, personal and professional advice throughout my graduate career.

Finally, I would like to thank Department of Mathematics and Statistics, University of South Florida for giving me the opportunity to be a part of their family.
# Table of Contents

List of Tables ................................................................. iv
List of Figures ............................................................... vi
Abstract ................................................................. vii

Chapter 1 Literature Review and Fundamental Concepts ............... 1
  1.1 Introduction .................................................. 1
  1.2 Foreign Exchange Market .................................. 1
    1.2.1 Foreign Exchange and Quantos ..................... 2
    1.2.2 FX Trading and Economic Theories ................. 3
  1.3 Fundamental Analysis and Fundamental Trading Strategies ....... 5
  1.4 Technical Analysis and Technical Trading Strategies .......... 5
    1.4.1 Technical Analysis: Chart Patterns ................. 6
    1.4.2 Technical Analysis: Moving Average ............... 9

Chapter 2 Nearest Neighbor Forecasting in Finance ............... 11
  2.1 Nearest Neighbor Algorithm ................................ 11
    2.1.1 Introduction ........................................ 11
  2.2 Definition of the Nearest Neighbor Problem ................ 13
  2.3 Nearest Neighbor Rule for Data Classification ............. 14
    2.3.1 Bayes Decision Rule ................................ 15
    2.3.2 Convergence of the Nearest Neighbor Decision Rule 16
    2.3.3 Probability of Error of the Nearest Neighbor ....... 18
  2.4 Dynamical System Approach ................................ 19
  2.5 $k$-Nearest Neighbor Forecasting ......................... 21
  2.6 Conclusion ................................................ 27

Chapter 3 $k$-Nearest Neighbor Forecasting with Mahalanobis Distance 28
  3.1 $k$-Nearest Neighbor Algorithm and the Choice of Distance .... 28
  3.2 Distance .................................................. 29
    3.2.1 The Euclidean Distance ............................ 30
    3.2.2 Manhattan Distance ................................ 30
    3.2.3 The Mahalanobis Distance ......................... 31
  3.3 Measures of Forecasting Accuracy .......................... 32
    3.3.1 Mean square error (MSE) ........................... 33
    3.3.2 Mean absolute percentage error (MAPE) .......... 33
    3.3.3 Theil’s $U$-statistic ($U$) ........................ 33
    3.3.4 Normalized Root Mean Square Error (NRMSE) ....... 34
List of Tables

Table 1  Financial Centers across the world with their opening and closing times according to US-eastern time (Source: Investopedia) ........................................... 2
Table 2  Currencies and their three letter codes .................................................. 3
Table 3  $U$-statistic with Mahalanobis distance ................................................. 37
Table 4  Optimal Choice of Embedding Dimension with $U$-Statistic ...................... 38
Table 5  Optimal Choice of Embedding Dimension with Normalized Root Mean Square Error 38
Table 6  $U$-Statistics .......................................................... 41
Table 7  Normalized Root Mean Square Error ...................................................... 42
Table 8  Sign correction proportion (SCP) with Mahalanobis and Euclidean distances ... 42
Table 9  Mean Absolute Deviation (MAD) with Mahalanobis and Euclidean distance measures 43
Table 10 Mean Square error (MSE) with Mahalanobis and Euclidean distance measures 43
Table 11 Estimated Return ................................................................. 46
Table 12 Sharpe Ratio ................................................................. 46
Table 13 $U$ Statistic with and without weighted regression Mahalanobis distance ...... 50
Table 14 $U$-statistics with $k$-NN and ARIMA models .................................... 62
Table 15 Mean square error with $k$-NN and ARIMA models ............................ 62
Table 16 Normalized root mean square error with $k$-NN and ARIMA models ......... 63
Table 17 Total Variation and Standard Deviation .............................................. 64
Table 18 Total Return $k$-NN and ARIMA models .......................................... 65
Table 19 Sharpe Ratio with $k$-NN and ARIMA models ................................. 66
Table 20 Standard deviations of the simulated data ......................................... 67
Table 21 $U$-Statistics for $k$-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.00127 and 0.896 .............................. 68
Table 22 MSE for $k$-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.00127 and 0.896 .............................. 69
Table 23 $U$-statistic for $k$-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.009 and 0.15 .............................. 70
Table 24 $U$-Statistics with Independent Value Prediction .................................. 79
<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 25</td>
<td>Normalized Root Mean Square Error with Independent Value Prediction</td>
<td>79</td>
</tr>
<tr>
<td>Table 26</td>
<td>$U$-Statistics for Independent Value Prediction and Recursive strategy with Mahalanobis Distance</td>
<td>80</td>
</tr>
<tr>
<td>Table 27</td>
<td>Total Return</td>
<td>80</td>
</tr>
<tr>
<td>Table 28</td>
<td>Total Return with single-step-ahead forecasting and multi-step-ahead forecasting</td>
<td>81</td>
</tr>
</tbody>
</table>
List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>Bar chart (Source: OANDA)</td>
<td>7</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Trend Lines (Source: OANDA)</td>
<td>7</td>
</tr>
<tr>
<td>Figure 3</td>
<td>Candlesticks Chart (Source: OANDA)</td>
<td>8</td>
</tr>
<tr>
<td>Figure 4</td>
<td>An illustration of an ( r )-near neighbor query</td>
<td>14</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Isosurfaces of Euclidean, Weighted Euclidean and Mahalanobis distances</td>
<td>32</td>
</tr>
<tr>
<td>Figure 6</td>
<td>EUR/USD daily rates from Jan 2006 to Dec 2010</td>
<td>36</td>
</tr>
<tr>
<td>Figure 7</td>
<td>U statistic vs. Embedding Dimension with Mahalanobis Distance</td>
<td>37</td>
</tr>
<tr>
<td>Figure 8</td>
<td>U statistic vs. Embedding Dimension with euclidean Distance</td>
<td>38</td>
</tr>
<tr>
<td>Figure 9</td>
<td>U-Statistic vs. ( k ) with Mahalanobis Distance</td>
<td>40</td>
</tr>
<tr>
<td>Figure 10</td>
<td>U-Statistic vs. ( k ) with Euclidean Distance</td>
<td>40</td>
</tr>
<tr>
<td>Figure 11</td>
<td>Forecasted and real values- JPY/USD rates</td>
<td>44</td>
</tr>
<tr>
<td>Figure 12</td>
<td>A delay vector and its 3-NNs with Mahalanobis distance</td>
<td>47</td>
</tr>
<tr>
<td>Figure 13</td>
<td>A delay vector and its 3-NNs with Euclidean distance</td>
<td>48</td>
</tr>
<tr>
<td>Figure 14</td>
<td>A delay vector and it’s ( N.N ) with Mahalanobis distance when there’s a directional change</td>
<td>49</td>
</tr>
<tr>
<td>Figure 15</td>
<td>ARIIMA(1,1,1) Forecasts and real values for EUR/USD daily exchange rates.</td>
<td>57</td>
</tr>
<tr>
<td>Figure 16</td>
<td>ARIIMA(1,1,2) Forecasts and real values for GBP/USD daily exchange rates.</td>
<td>59</td>
</tr>
<tr>
<td>Figure 17</td>
<td>ARIMA(1,1,2) Forecasts and real values for JPY/USD daily exchange rates.</td>
<td>59</td>
</tr>
<tr>
<td>Figure 18</td>
<td>ARIMA(1,1,1) Forecasts and real values for CHE?USD daily exchange rates.</td>
<td>60</td>
</tr>
<tr>
<td>Figure 19</td>
<td>ARIMA(1,1,3) Forecasts and real values for CAD/USD daily exchange rates.</td>
<td>61</td>
</tr>
</tbody>
</table>
Abstract

Foreign exchange (FX) rate forecasting has been a challenging area of study in the past. Various linear and nonlinear methods have been used to forecast FX rates. As the currency data are nonlinear and highly correlated, forecasting through nonlinear dynamical systems is becoming more relevant. The nearest neighbor (NN) algorithm is one of the most commonly used nonlinear pattern recognition and forecasting methods that outperforms the available linear forecasting methods for the high frequency foreign exchange data. The basic idea behind the NN is to capture the local behavior of the data by selecting the instances having similar dynamic behavior. The most relevant $k$ number of histories to the present dynamical structure are the only past values used to predict the future. Due to this reason, NN algorithm is also known as the $k$-nearest neighbor algorithm ($k$-NN). Here $k$ represents the number of chosen neighbors.

In the $k$-nearest neighbor forecasting procedure, similar instances are captured through a distance function. Since the forecasts completely depend on the chosen nearest neighbors, the distance plays a key role in the $k$-NN algorithm. By choosing an appropriate distance, we can improve the performance of the algorithm significantly. The most commonly used distance for $k$-NN forecasting in the past was the Euclidean distance. Due to possible correlation among vectors at different time frames, distances based on deterministic vectors, such as Euclidean, are not very appropriate when applying for foreign exchange data. Since Mahalanobis distance captures the correlations, we suggest using this distance in the selection of neighbors.

In the present study, we used five different foreign currencies, which are among the most traded currencies, to compare the performances of the $k$-NN algorithm with traditional Euclidean and Absolute distances to performances with the proposed Mahalanobis distance. The performances were compared in two ways: (i) forecast accuracy and (ii) transforming their forecasts into a more effective technical trading rule. The results were obtained with real FX trading data, and the results
showed that the method introduced in this work outperforms the other popular methods.

Furthermore, we conducted a thorough investigation of optimal parameter choice with different distance measures. We adopted the concept of distance based weighting to the NN and compared the performances with traditional unweighted NN algorithm based forecasting.

Time series forecasting methods, such as Auto regressive integrated moving average process (ARIMA), are widely used in many areas of time series as a forecasting technique. We compared the performances of proposed Mahalanobis distance based $k$-NN forecasting procedure with the traditional general ARIMA- based forecasting algorithm. In this case the forecasts were also transformed into a technical trading strategy to create buy and sell signals. The two methods were evaluated for their forecasting accuracy and trading performances.

Multi-step ahead forecasting is an important aspect of time series forecasting. Even though many researchers claim that the $k$-Nearest Neighbor forecasting procedure outperforms the linear forecasting methods for financial time series data, and the available work in the literature supports this claim with one step ahead forecasting. One of our goals in this work was to improve FX trading with multi-step ahead forecasting. A popular multi-step ahead forecasting strategy was adopted in our work to obtain more than one day ahead forecasts. We performed a comparative study on the performance of single step ahead trading strategy and multi-step ahead trading strategy by using five foreign currency data with Mahalanobis distance based $k$-nearest neighbor algorithm.
Chapter 1

Literature Review and Fundamental Concepts

1.1 Introduction

The foreign exchange market (forex, FX, or currency market) is the largest financial market in the world [81]. In the present chapter, we begin with summarizing the existing preliminary concepts of the FX market and introduce some of the fundamental concepts that are essential for dealing with FX trading. We also discuss the available forecasting methods and trading strategies in FX trading.

1.2 Foreign Exchange Market

The foreign exchange market is a non-stop cash market where currencies of nations are traded. Foreign currencies are constantly and simultaneously bought and sold across local and global markets, and traders’ investments increase or decrease in value based upon currency movements. The investors goal in FX trading is to profit from foreign currency movements. FX trading was pretty much limited to large financial institutions, corporations, central banks and hedge funds until recent years [28, 41]. Since electronic trading has been introduced, practically anyone with a computer and Internet has the opportunity to trade currencies [28, 41, 64]. As there is always a necessity of exchanging the currencies, the FX market remains the largest, most liquid financial market [22, 41, 61].

Foreign currencies are traded worldwide in the nine major financial centers; London, New York, Tokyo, Zurich, Frankfurt, Hong Kong, Singapore, Paris, and Sydney [41]. As the major trading centers are spread out in various time zones, when one center closes, somewhere on the other side of the world another center will open its center for business [41]. That’s what it makes possible to trade during any time of the day. The following table (table 1.2) illustrates the opening and closing times of major financial centers, according to the eastern standard time.
Table 1: Financial Centers across the world with their opening and closing times according to US-eastern time (Source: Investopedia)

<table>
<thead>
<tr>
<th>Time Zone</th>
<th>Time (ET)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tokyo Opens</td>
<td>7:00 pm</td>
</tr>
<tr>
<td>Tokyo Closes</td>
<td>4:00 am</td>
</tr>
<tr>
<td>London Opens</td>
<td>3:00 am</td>
</tr>
<tr>
<td>London Closes</td>
<td>12:00 pm</td>
</tr>
<tr>
<td>New York Opens</td>
<td>8:00 am</td>
</tr>
<tr>
<td>New York Closes</td>
<td>5:00 pm</td>
</tr>
</tbody>
</table>

The FX market does not need a central marketplace as currency trading is conducted electronically, over-the-counter (OTC) [28, 41, 61]. Even though the daily currency fluctuations are very small (less than 1% [41]), due to the round-the-clock trading and deep liquidity, FX brokers manage to make even a small movement meaningful for traders [28, 41, 61].

There are three main different types of currency markets:

- The spot market
- The forwards market
- The futures market

In spot trading, the currencies are bought and sold for the current price [28, 61]. The spot market has been the largest currency market because the other two types (forwards and futures) are based on spot trading [28, 41]. As in any other forward market, FX forwards contracts are bought and sold over-the-counter between two parties after determining the terms of the agreement [28, 41]. The futures contracts are based upon a standard size and settlement date. These trades happen at public commodities markets [41, 64].

1.2.1 Foreign Exchange and Quantos

Currencies are normally identified using the industry-defined three-letter codes. The following table gives some examples of three letter codes of the most-traded currencies in the world [22, 41].

A currency is quoted in relation to another currency, as a pair. Therefore, the value of one currency is reflected in terms of another [41, 81]. For example, an exchange rate of EUR/USD = 1.12 means
that 1 EUR = 1.12 USD. The currency to the left is called the base currency, and the currency on the right is the quoted or counter currency [22, 41, 64]. The base currency is always equal to one unit, and the quoted currency is how much 1 unit of base currency is equal to in terms of the other currency. If the FX rate is quoted in U.S. dollars as the base currency, it is refereed to a quote as per European terms [41, 81]. On the other hand, an exchange rate with the U.S. dollars as the quoted (variable) currency will be considered to be a quote as per American terms. Thus, EUR/USD = 1.12 is a quote as per American terms [41, 81].

The currencies quotes also can be categorized as direct and indirect quotes [28, 61, 81]. When the domestic currency becomes the quoted currency and foreign currency is the base currency, it’s considered as directly quoted. If foreign currency is the quoted currency and the domestic currency is the base currency, then it is called indirectly quoted. For example, GBP/USD = 1.56 is a direct quote in New York and an indirect quote in London.

Buying/selling a currency pair means a trader intend to buy/sell the base currency. In both cases, the transaction happens in terms of the quoted currency, and whichever currency is quoted first (the base currency) is the one in which the transaction is being conducted [22, 41, 61, 64, 81].

### 1.2.2 FX Trading and Economic Theories

There are couple of important economic theories behind the currency market as well as trading, even though they are not widely applicable to real day-to-day FX trading [28, 41, 64]. The majority

---

#### Table 2: Currencies and their three letter codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Currency Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>Euro</td>
</tr>
<tr>
<td>USD</td>
<td>U.S. dollar</td>
</tr>
<tr>
<td>GBP</td>
<td>Pound sterling</td>
</tr>
<tr>
<td>JPY</td>
<td>Japanese yen</td>
</tr>
<tr>
<td>CHF</td>
<td>Swiss franc</td>
</tr>
<tr>
<td>CAD</td>
<td>Canadian dollar</td>
</tr>
<tr>
<td>AUD</td>
<td>Australian dollar</td>
</tr>
<tr>
<td>NZD</td>
<td>New Zealand dollar</td>
</tr>
</tbody>
</table>

---


of these economic theories are based on the parity condition, which is an economic interpretation of a currency rate based on factors such as inflation and interest rates.

- **Purchasing Power Parity (PPP)** -
  Purchasing power parity calculates the adjustment needed to be made to the FX rate in order to have an equivalent purchasing power between the two counties. According to PPP, the exchange rate of currency 1 to currency 2, \( S \), is given by

  \[
  S = \frac{P_1}{P_2}
  \]

  Here \( P_1 \) and \( P_2 \) are the costs of a particular good in currency 1 and currency 2 respectively [28, 61].

- **Interest Rate Parity (IRP)** -
  Interest rate parity explains the connection between the FX market and international money market [28]. The formula of IRP can be written as:

  \[
  (i_1 - i_2) = (\frac{F - S}{S})(1 - i_2)
  \]

  Here, \( F \) is the forward exchange rate, \( S \) represents the spot exchange rate, and \( i_1 \) & \( i_2 \) are the interest rates in country 1 and country 2, respectively. The basic idea of IRP is not to have arbitrage opportunities in currency trading. The law of one price is the basis behind the interest rate parity [28, 41, 61].

- **International Fisher Effect (IFE)** -
  International Fisher Effect (IFE) relates the percentage change in spot exchange rate over time difference between the two countries’ nominal interest rates. The formula of IFE can be given as follows:

  \[
  \frac{S_1 - S_2}{S_2} = \frac{i_1 - i_2}{1 + i_2}
  \]

  Here, \( S_1 \) represents the spot exchange rate of country 1, \( S_2 \) represents the spot exchange rate of country 2 and \( i_1 \) & \( i_2 \) are the interest rates in country 1 and country 2, respectively. [28, 41, 61].
Apart from the economic theories behind the exchange rate determination discussed here, there are several other models, such as the asset market model, the monetary model, and the balance of payments theory, which discuss FX rates determination. Further details of these methods can be found in the provide references [22, 28, 41, 61].

1.3 Fundamental Analysis and Fundamental Trading Strategies

Fundamental analysis in any financial market mainly focuses on the related economical and financial factors. In the FX market, the fundamental analysis mainly involves studying the economic situation of countries to trade currencies more effectively [41, 81]. These situations may include political and economical events, economic growth and financial strength, interest rates, or financial and political crises of countries [28, 41, 64]. Fundamental analysis of foreign exchange trading is very challenging compared to other financial markets. For example, in stock trading an investor can analyze each company’s revenues, earnings, return on equity, and other data to determine a company’s underlying value and potential for future growth [21, 41].

In the foreign exchange market, evaluating the above mentioned statistics is sometimes not practical, as many of those do not exist [21, 41]. When it comes to FX trading, the trader is looking at an entire economy of a country compare to another, which includes thousands of companies. At the same time, some of the commodity prices, such as fuel price, and gold price, are also considerable factors determining the exchange rate [41, 55]. Many traders are more interested in Technical Analysis compared to fundamental analysis due to this nature of the FX market [28, 41, 55].

1.4 Technical Analysis and Technical Trading Strategies

The technical analysis mainly focuses on the historical price data. Its argument is that the “historical price action predicts future price action” [41]. Since it is a 24-hour market, technical analysis and technical trading strategies have a huge impact on foreign exchange trading. In any financial market, technical analysis is based on the following three main assumptions [28, 41]:

5
1. The market discounts everything.

2. Price moves in trends.

3. History tends to repeat itself.

As all the factors that influence currency prices have already been factored into the FX rates, many traders believe that technical trading strategies are more appropriate for FX trading compared to fundamental trading strategies [28, 41, 61]. There are wide varieties of technical tools to analogize the historical data and technical trading strategies to make trading decisions. In this section, we only focus on the technical analysis commonly used in FX trading.

### 1.4.1 Technical Analysis: Chart Patterns

Most of the technical tools in FX trading are based on different types of charts and trend lines [41]. The following are the main types of price charts used in foreign exchange trading as a technical tool to determine the currency movements.

- **Bar Charts**
  
  This is a type of line chart that includes more information about the currency rates, as compared to a regular price chart. As can be seen in Figure 1 on page 7, each data point is represented by a vertical line where length of a vertical line represents the low and high price for the trading period. The open and closed prices for the same time period are displayed by small horizontal line segments on the left and right sides, respectively. As a bar chart combines the basic information about a currency pair into one chart, traders can determine the direction for the currency [39, 41].

- **Trend Lines**
  
  A trend line chart is a line that can be placed over a price chart to determine the trend of the data. This gives a basic idea of the overall market direction and the price range for the time period [41]. There are three main types of trend lines. Support trend line, which connects the lower price points, resistance trend line, which connects the highest prices, and directional trend line, which highlights the overall market direction for the currency pair for a given time period. [28, 39, 41]. An example of a trend line chart is given in Figure 2 on page 7.
Figure 1.: Bar chart (Source: OANDA)

Figure 2.: Trend Lines (Source: OANDA)
Figure 3. Candlesticks Chart (Source: OANDA)

Candlestick Charts

Figure 3 on page 8 is an example of a candlestick chart. This is one of the personal favorite chart types among the traders as it combines more information compared with the rest [39, 41]. Similar to a bar chart, this chart also provides opening and closing prices, high and low prices, and direction and trend. While the body length of the candlestick displays the change in open and close prices for the period, the color of the candlestick gives an idea of whether the open price is higher or lower the the closing price [41]. An empty candlestick indicates that the closing rate is higher than the opening rate for the time period. In that case, the bottom of the body represents the opening rate, and the top of the body represents the closing rate. A filled or colored candlestick is an indication of a lower closing price than an opening price. In this case, the opening rate is at the top of the body, and the closing rate at the bottom [39, 41].

Apart from the above discussed main chart types, Fibonacci retracement, Bollinger bands, and Market and momentum oscillators are some of the other popular chart types used as technical analysis tools in FX trading. More details of these chart types can be found in the provided references [41, 61, 39].
1.4.2 Technical Analysis: Moving Average

Even though the chart types discussed in section 1.4.1 provide a variety of information related to price movements, sometimes it can be difficult for the FX trader to get a correct idea of the overall trend [41]. The moving average procedure overcomes this difficulty by providing clear information about the price trend [28, 39, 41]. The moving average eliminates the day-to-day fluctuations in the data (for a daily currency data) and provides a clear picture of true trend.

- **Simple Moving Average**

  Simple moving average (MA) is considered as the simplest and most common trading technical rule in the technical analysis literature [35, 41]. In MA technical tool, we simply calculate the average of consecutive data points over a given period of time. If $E_t$ represents the spot exchange rate at time $t$, the moving average of $n$ consecutive days can be calculated as

  $$M A_t(n) = \frac{1}{n} \sum_{i=0}^{n-1} E_{t-i}$$

  A trader can make the moving average less responsive to the price by increasing $n$. This technical trading tool smooths out the data set. By increasing the time period, $n$, a trader will be able to capture the long term trend in the currency prices [41].

- **Linear Weighted Moving Average**

  Linear weighted moving average (WMA) provides an alternative to the simple moving average rule providing more weights to the nearby values.

  $$W M A_t(n) = \frac{1}{\frac{1}{2} + \frac{3}{2} + \ldots + \frac{n}{2}} \sum_{i=0}^{n-1} (n - j) E_{t-i} = \frac{1}{\frac{n}{2} + \frac{n}{2} + \ldots + \frac{n}{2}} \sum_{i=0}^{n-1} (n - j) E_{t-i}$$

- **Exponential Weighted Moving Average**

  It is a common belief among the traders that the nearby values have more impact on the near future values [28, 39, 41]. Exponential weighted moving average is another alternative to the simple moving average rule in which weights are assigned exponentially giving much more weights to the nearby values [41].
Moving average techniques are considered to be the mostly used technical trading strategies by the FX trades [35, 41]. However, the recent literature discusses the major downfalls of these popular methods [35]. According to the definition of simple and weighted moving average, it can be spread out for any given time period. It’s clearly sensitive to the time period, and trend is mainly dependent on the time period that we chose [28, 41, 39].

Another major issue discussed in the financial literature and also a debated factor among the traders is the idea of giving more weights to the recent prices [35, 28, 41]. Even if it seems accurate, giving more weight to the nearby values may end up giving incorrect trading signals as the trend is incorrectly biased towards the end [41].

As the foreign exchange data is nonlinear and highly correlated, the ability of making proper trading signals using a linear technical strategy was questionable among the researchers [35, 41]. Forecasting FX rate was always a challenging task due to the high correlation among the data. According to the past claims of many researchers, linear models are unable to fully capture exchange rate dynamics [35, 33]. Forecasting through nonlinear dynamical systems is becoming more and more relevant due to this nature of the currency data [35, 65, 66]. Nearest neighbor algorithms are one of the most commonly used non-linear pattern recognition methods that outperforms the available linear forecasting methods for the high frequency foreign exchange data.

Fernández-Rodríguez et al. showed that k-nearest neighbor forecasting method outperforms the moving average (MA) based technical trading rule [33, 35]. They have used linear correlation as their similarity measure to select the k-neighbors for forecasting. Their method was applied for currencies from the European Monetary System (EMS), and trading performances were compared with the MA trading rule. From the obtained results they concluded that the k-NN method (with correlation) outperforms the moving average based trading strategy [35].

In this dissertation, we will introduce Mahalanobis distance based nearest neighbor algorithms for foreign exchange rate forecasting and corresponding decision methods. We will also compare proposed forecasting efficiency with other popular forecasting methods.
Chapter 2

Nearest Neighbor Forecasting in Finance

2.1 Nearest Neighbor Algorithm

2.1.1 Introduction

The nearest neighbor rule, since its conception dating back to an unpublished report by Fix and Hodges in 1951, has attracted many followers and continues to be studied by many researchers. In their work, Fix and Hodge numerically evaluate the performance of $k$-nearest neighbor algorithm for small samples under the assumptions of normal distribution [19]. $k$-NN is one of the top 10 data mining algorithms [82]. It is one of those algorithms that are quite simple to understand but work incredibly well in practice. Also, it is surprisingly versatile. $k$-NN algorithm is applicable to various different areas such as engineering, physics, medicine, finance, and computational geometry. NN is a non-parametric lazy learning algorithm. It performs little work when learning from a data set, but expends more effort classifying new examples. Being non-parametric is pretty useful because, when a technique is non-parametric, it means that it does not make any assumptions on the underlying data distribution. Since most of the practical data does not obey the typical theoretical assumptions made, nearest neighbor algorithms based methods are becoming more and more relevant for real world problems.

The nearest neighbor algorithm and its derivatives are often referred to as instance-based learning algorithms. Instance-based learning is a supervised learning technique that is considered as the most fundamental task in machine learning. In supervised learning, a new example, which is considered as an instance with unknown category / label, is classified into a proper category by comparing it to those already seen and in memory. The goal here is to predict the appropriate label of the newly available instant using already labeled categories of the training examples.
This concept description includes a set of stored instances and, possibly, some information concerning their past performances during classification. Storing and using specific instances improves the performance of several supervised learning algorithms. This set of instances can change after each training instance is processed. However, no investigation has analyzed algorithms that use only specific instances to solve incremental learning tasks [1].

The nearest neighbor rule was formally introduced by Cover and Hart in 1967. They showed that its error rate is bounded by twice the Bayes optimum error in the limit [19]. Their original paper also discussed the $k$-nearest neighbor rule in which the majority of the $k$ closest neighbors are used for classification. Nearest neighbor is an improved version of instance-based learning algorithms, which outperforms most of the available other prediction models. It generates classification predictions using only specific instances. It does not maintain a set of abstractions derived from specific instances. Nearest neighbor algorithm simply retains the entire training set during learning. During execution, the new input data/vector is compared to each instance in the training set. The class of the instance that is most similar to the new vector is used as the predicted output class. The similarity is determined by a metric that measures the distance between a new data/vector and the set of data in memory [1, 19, 23].

One of the major advantages of nearest neighbor algorithm is that it is able to learn quickly from a very small dataset, compared to rule induction methods that require a reasonable representation of each rule before they can be induced. NN classifiers can begin to make useful predictions from as little as one example per class. Classification performance often exceeds 75% of the maximum possible after accepting only 25% of a complete data set [1]. This approach, therefore, works well even when limited data is available. The nearest neighbor algorithm has several strengths when compared to most other learning models [90]:

- It learns fast ($O(n)$ for a training set of $n$ instances)
- NN rule is analytically tractable
- It uses local information, which can yield highly adaptive behavior
- It’s guaranteed to learn a consistent training set (i.e., one in which there are no instances with
the same input vector and different outputs) and will not get stuck in local minima.

- Nearest neighbor is nearly optimal in the large sample limit
- It is intuitive and easy to understand, which facilitates implementation and modification.
- It provides good generalization accuracy on many applications [40].

However, in its basic form, the nearest neighbor algorithm has several drawbacks:

- Its distance functions are typically inappropriate for applications with both linear and nominal attributes.
- It has large storage requirements because it stores all of the available training data in the model [90].
- It is slow during execution because all of the training instances must be searched in order to classify each new input vector [90].
- Its accuracy degrades with the introduction of irrelevant attributes.

Over the past decades, many researchers have tried to address these drawbacks in many different ways. Cluster-k-nearest neighbor, probabilistic nearest neighbor, and Kernel nearest neighbor, are some of the examples of such modified versions of the nearest neighbor algorithm [51].

### 2.2 Definition of the Nearest Neighbor Problem

The nearest neighbor problem is an example of an optimization problem; the goal is to find a point (or points) that minimize a certain objective function (in this case, the distance to the query point). We say that a point $p$ as an $r$-near neighbor of a point $q$ if the distance between $p$ and $q$ is at most $r$. In this language, an algorithm either returns one of the $r$-near neighbors, or concludes that no such point exists for some fixed parameter $r$. As an example, in Figure 4 the nearest neighbor of the query point $q$ is the point $p_1$. However, $p_1$, $p_2$, and $p_3$ are $r$-near neighbors of $q$. Nearest neighbor classification has various applications in many different fields of science, business and industry [35, 36]. It can classify new data into different classes according to the available training data sets. In pattern recognition, when new data arrives, the NN algorithm finds the neighbors nearest to the
new value from the specified distance metric [23]. When applied to time series (such as financial data) forecasting, we use the $k$ number of nearest neighbors chosen by the algorithm based on a specified distance function to forecast the value for the next time step.

### 2.3 Nearest Neighbor Rule for Data Classification

In the problem of data classification, data analysts or statisticians have to deal with one of the two situations below:

(i) The complete statistical knowledge of the underlying joint distribution of the observation and the true category are available.

(ii) No knowledge of the underlining joint distribution is available.

When the necessary information is available, the standard Bayes decision rule will yield an optimal decision procedure [19]. The corresponding minimum probability of error (Bayes error) is denoted as $R^*$. In reality, that is often not the case. Especially, when dealing with data such as time series, it is not possible to make distributional assumptions. In the latter case, we start with zero knowledge of the underlying joint distribution. Therefore, we are not in a position to make any theoretical assumptions needed for the the Bayes decision procedure.

However, the relevant information can be inferred from the available data sample. This is where nearest neighbor rule plays a key role as a pattern recognition algorithm in data classification. As it is nonparametric and distribution free, it can be applied for many different types of real world data classifications. In this section, we will give a brief description of nearest neighbor rule introduced...
by Cover and Hart in 1967 [19].

Let \( (x_1, \theta_1), (x_2, \theta_2), \ldots, (x_n, \theta_n) \) be a set of \( n \) pairs such that each \( x_i \) associates a value in a metric space \((X, d)\) where \( d \) is a metric and \( \theta_i \)'s belong to the index set \{1, 2, ..., \( N \)\}. Here we are assuming that \( x_i \) belongs to the class (or category) \( \theta_i \). When a new observation, \( x \), is available, the goal is to assign it to its true class \( \theta \). In order to assign the newly arrived individual to the class in which it belongs, the distance \( d(x_i, x) \) between the new observation \( x \) and \( x_i \); for \( i = 1, 2, \ldots, n \) is calculated. We say \( x^* \in \{x_1, x_2, \ldots, x_n\} \) is a nearest neighbor to \( x \) if

\[
d(x^*, x) = \min \{d(x_i, x)\}; \quad i = 1, 2, \ldots, n
\]  

(2.1)

Thus, the nearest neighbor rule completely depends on the similarity among the data structure and does not require any theoretical assumptions of the underlying distribution of the data. The nearest-neighbor method also has an important guarantee of its classification error bounds. Before we discuss the classification guarantee and convergence properties of the NN classifier, we will review the optimality baseline given by the Bayes probability of error.

### 2.3.1 Bayes Decision Rule

In order to understand the error bounds for the nearest neighbor decision rule, first we briefly discuss the Bayes decision procedure. The main objective in Bayes decision rule is minimizing the probability of error in classifying a given observation \( x \) into one of the \( N \) categories/classes. The Bayes risk serve as a reference-limit of excellence beyond which it is not possible to go. However, we need to make the most common assumption that all the statistics are known.

Let us take \( x \) as the measurement on an individual and \( \Omega \) as the sample space of possible outcomes of \( x \). The decision must be made regarding the membership of the individual in one of the \( N \) specified categories (classes) based on the value of \( x \). Assume \( f_1, f_2, \ldots, f_N \) are the probability densities at \( x \) with respect to a \( \sigma \)-finite measure such that an individual in category \( i \) gives rise to an observation \( x \) according to density \( f_i \) [19]. An error occurs if we make a decision to assign an individual to category \( i \) when it actually belongs to category \( j \). Let \( L(i, j) \) be the loss incurred by this misclassification.
Assuming that $p_1, p_2, \ldots, p_N$; with $p_i \geq 0$ & $\sum p_i = 1$ as prior probabilities of $N$ categories, by Bayes theorem, we can obtain the conditional probabilities of an individual with measurement $x$ belonging to category $i$ as [19],

$$q_i(x) = \frac{p_i f_i}{\sum_{i=1}^{N} p_i f_i}; \quad i = 1, 2, \ldots, N$$

(2.2)

Thus the random variable $x$ transforms the prior probability vector $p(x)$ into the posterior probability $q(x)$ vector. Therefore, we can estimate the conditional loss due to the misclassification as [19]:

$$r_i(x) = \sum_{i=1}^{N} q_i(x) L(i,j)$$

(2.3)

The goal here is to minimize the conditional loss. Obviously, it is a minimum when the individual is assigned to a category $j$ for which $r_i(x)$ is lowest. Note that by minimizing conditional expected loss, we can minimize the unconditional expected loss. The decision that minimizes the conditional expected loss is called the Bayes decision rule.

Let $\delta^\ast$ be the Bayes decision rule with respect to $p$. Then the conditional Bayes risk with respect to decision $\delta^\ast$ is given by

$$r^\ast(x) = \min_i \left\{ \sum_{i=1}^{N} q_i(x) L(i,j) \right\}$$

(2.4)

Subsequently, the overall minimum expected risk, which is called the Bayes risk, can be written as

$$R^\ast = E[r^\ast(x)]$$

(2.5)

Note that the expectation is taken with respect to the compound density $f(x) = \sum_{i=1}^{N} p_i f_i(x)$

### 2.3.2 Convergence of the Nearest Neighbor Decision Rule

The main properties of nearest neighbor rule are based on the assumption that the conditional distribution of $\theta'$ approaches conditional distribution of $\theta$ as $\lim_{x' \to x}$. In their original paper [19], Cover and Hart have discussed the convergence of the NN decision rule under the weakest possible conditions. Since the convergence of the nearest neighbor to $x$ is independent of the metric, the bounds on the risks of the NN rule will also be independent of the metric on $X$ [19].
LEMMA 2.1 Let $X$ be a separable metric space. Let $x$ and $x_1, x_2, \ldots$ be identically distributed random variables taking values in $X$. Let $x'_n$ denote the nearest neighbor to $x$ from the finite sample $\{x_1, x_2, \ldots, x_n\}$. Then $x'_n \to x$ with probability one[19].

Proof. Let $S_x(r) = \{x^* \in X : d(x, x^*) \leq r\}$ be the sphere of radius $r$ centered at $x$, where $d$ is some metric $d$ defined on $X$. Consider a point $x \in X$ having the property that every sphere $S_x(r) > 0$, has none-zero probability measure. That is for any $r > 0$

$$P( (x \in S_x(x))) = \int_{S_x(x)} f(x)dx > 0 \quad (2.6)$$

Then for any $\delta > 0$, we have

$$P\{ \min_{k=1,2,\ldots,n} d(x_k, x) \geq \delta \} = [1 - P(S_x(\delta))]^n \to 0 \quad (2.7)$$

Note that $d(x_k, x)$ is monotonically decreasing in $k$. Therefore, the nearest neighbor to $x$, converges to $x$.

To prove that the the nearest neighbor to $x$ converges to $x$ with probability one, we will show that the set of points (values) failing to have this property have probability measure zero.

Let $X'$ be the set of all such points. Consider a point $x' \in X'$. Then, for some sufficiently small $r'(> 0)$, we have $P(S_{x'}(x')) = 0$. By the definition of the separability of $X$, there exists a countable dense subset $A$ of $X$. Then there exists (by the denseness of $A$) $a_{x'} \in A$ such that $a_{x'} \in S_{r'/2}(x')$.

Thus, there exists a small sphere $S_{r'/2}(a_{x'})$ such that

$$S_{r'/2}(a_{x'}) \subset S_{r'}(x')$$

$$\Rightarrow P[S_{r'/2}(a_{x'})] = 0$$

Note that $x' \in S_{r'/2}(a_{x'})$. Then by the countability of $A$, the possibly uncountable set $X'$ is contained in the countable union of spheres. That is

$$X' \subseteq \bigcup_{x' \in X'} S_{r'}(x')$$

$$X' \subseteq \bigcup_{x \in X'} S_{r'/2}(a_{x'})$$

Since $X'$ is contained in the countable union of measure zero, $P(X') = 0$. \hfill QED

17
2.3.3 Probability of Error of the Nearest Neighbor

Before discussing the error bounds of the NN rule, let us first define the nearest neighbor probability of error. Let $X = \{x_1, x_2, \ldots, x_n\}$ be the set of available the data sample and $x'_n$ be the nearest neighbor to $x$. Let $\theta'_n$ be the category to which the individual $x'_n$ belongs and $\theta$ be indeed the category of $x$. If we denote $L(\theta, \theta'_n)$ as the loss incurred by this misclassification, then we can define $n$-sample NN risk $R(n)$ by the expectation

$$R(n) = E[L(\theta, \theta'_n)] \quad (2.8)$$

and large sample nearest neighbor risk $R$ by

$$R = \lim_{n \to \infty} R(n) \quad (2.9)$$

Note that, here we assume that the pairs $(x, \theta), (x_1, \theta_1), (x_2, \theta_2), \ldots, (x_n, \theta_n)$ are independent, identically distributed random variables in $X \times \Theta$.

**Theorem 2.1**: Cover & Hart error bounds for two-class NN problem

Let $X$ be a separable metric space and let $x$ and $f_1, f_2$ be such that, with probability one, $x$ is either (i) a continuity point of $f_1$ & $f_2$, or (ii) a point of non zero probability measure. Then, the asymptotic NN risk $R$ (probability of error) has the bounds

$$R^* \leq R \leq 2R^*(1 - R^*) \quad (2.10)$$

**Theorem 2.2**: Cover & Hart error bounds for NN, M-class problem with $N \neq 2$

Let $X$ be a separable metric space and let $x$ and $f_1, f_2, \ldots, f_N$ be such that, with probability one, $x$ is either (i) a continuity point of $f_1, f_2, \ldots, f_N$, or (ii) a point of non zero probability measure. Then, the asymptotic NN risk $R$ (probability of error) has the bounds

$$R^* \leq R \leq R^*(2 - \frac{N}{N - 1} R^*) \quad (2.11)$$

18
2.4 Dynamical System Approach

Another approach to NN prediction is related to the forecast in chaotic time series. This is mainly based on the Floris Takens delay embedding theorem in 1981 [15, 16, 33, 80].

Reconstructing the underlying dynamics of a physical system from a experimental time series data set is an important task in many applied areas of time series. Researchers were interested in this phenomena for quite some time and there were some positive approaches suggested by even before Takens 1981 theorem [69, 89]. In 1981, Floris Takens gave more information and guarantees of the reconstruction of a time series accurately with a sufficient number of observations at equally spaced times [69, 70, 80]. He provided the conditions under which a chaotic dynamical system can be reconstructed from a sequence of observations of the state of a dynamical system. This reconstruction preserves the properties of the dynamical system that do not change under smooth coordinate changes [15, 16, 80]. This theorem uses an observation function to construct the embedding function. Takens delay embedding theorem was proved to hold for finite-dimensional subsets of infinite-dimensional spaces [80].

THEOREM 2.3: Takens Theorem 1(1981)

Let $M$ be a compact manifold of dimension $m$. For pairs $(\varphi, y)$, $\varphi : M \to M$ a smooth (at least $C^2$) diffeomorphism and $y : M \to \mathbb{R}$ a smooth function, it is a geometric property that the map $\Phi_{(\varphi, y)}(x) : M \to \mathbb{R}^{2m+1}$, defined by

$$
\Phi_{(\varphi,y)}(x) = (y(x), y(\varphi(x)), \ldots, y(\varphi^{2m}(x)))
$$

(2.12)

is an embedding.

Takens also verified the validity of this theorem for a compact set $M$ by restricting the observables to a proper function [80].

THEOREM 2.4: Takens Theorem 2(1981)

Let $M$ be a compact manifold of dimension $m$. For pairs $(X, y)$, $X$ a smooth (i.e. $C^2$) vector field
and \( y \) a smooth function on \( M \), it is a geometric property that the map \( \Phi_{(X,y)}(x) : M \rightarrow \mathbb{R}^{2m+1} \), defined by
\[
\Phi_{(X,y)}(x) = (y(x), y(\varphi_1(x)), \ldots, y(\varphi_{2m}(x))
\]
(2.13)
is an embedding, where \( \varphi(t) \) is the flow of \( X \).

The state space reconstruction theory [63] and the embedding theorem [70, 80] are considered as the most important aspects of the nonlinear time series analysis. According to the theorem 2.4, a time series can be converted into a phase trajectory in state space by choosing an appropriate embedding dimension and delay time [33, 69]. As the dynamic on the reconstructed trajectory is equivalent to that of the original system, we can establish a model to predict the future [16, 32, 63, 69].

Following the idea of Takens theory, the time series data \( x_t \) (for \( t = 0, 1, 2, \ldots \)) has a smoothly deterministic explanation if there exists a system \((f, F, a_0)\) such that
\[
f : \mathbb{R}^n \rightarrow \mathbb{R} \quad \text{and} \quad F : \mathbb{R}^n \rightarrow \mathbb{R}^n
\]
(2.14)
are smooth (in \( C^2 \)) and
\[
x_t = f(a_t), \quad a_t = F(a_{t-1}), \quad t = 1, 2, \ldots (a_0 \text{ is given})
\]
(2.15)
Here the relationship \( a_t = F(a_{t-1}) \) is unknown. According to the Takens embedding theorem, the underlying state variable \( a_t \) can be learned from the “embeddings” of the observed values \( x_t \) in \( \mathbb{R}^m \) for a sufficiently large \( m (\geq 2n + 1) \) [33, 89].

This idea can be adapted to compute \( m \) - history as a vector of \( m \) observations sampled from a given time series, \((x_1, x_2, \ldots, x_T)\) within equal time intervals \( \tau \in \mathbb{N} \) [15, 36, 33]. Here, \( \tau \) is called the delay time. Let’s consider the simplest case \( \tau = 1 \). Then a \( m \)-dimensional history vector can be given as \( x_t^m = (x_1, x_2, \ldots, x_{t-(m-1)}) \). As in the case of Takens theory, \( m \) is referred to as the embedding dimension, and the \( m \)-dimensional space \( \mathbb{R}^m \) is referred to as the reconstructed phase space of the original time series. Thus, the dynamic of the unknown law \( a_t = F(a_{t-1}) \) is equivalent to the dynamic of \( x_t^m \) and also the dynamics of the \( m \)-histories \( x_t^m \) provide accurate topological properties of the unknown chaotic dynamics [33, 36, 80].
Even though the function $F$ is unknown, as with Taken, if the time series has a smoothly deterministic explanation for an embedding dimension $m$ sufficiently large, there exists a function

$$F^* : \mathbb{R}^m \rightarrow \mathbb{R}^m$$

such that

$$x_{t+1}^m = F^*(x_t^m)$$

Note that this map, $F^*$ has the same dynamic properties as that of the unknown system [33, 80]. This idea of the Takens embedding theorem provides a new geometrical interpretation to the nearest neighbor forecasting. NN method is used to select some geometric segments in the past of the time series, similar to the most recent segment available before the observation we need to forecast [32, 33]. If we can select the $m$-histories with similar dynamic behavior, these history values are employed afterwards in the prediction of the next value in the time series.

The above mentioned result is used in this dissertation for daily foreign exchange rate forecasting with the $k$-nearest neighbor forecasting procedure. According to the idea of Takens theorem, $k$-NN uses the $k$ number of $m$ (with a sufficiently large $m$, at least $2n + 1$) to predict the future value of a time series. In chapter 3, after testing $m = 3, 4, 5, 6$ as the embedding dimension, we found that $m = 3$ is sufficiently accurate. Therefore, in this dissertation, we have used $m = 3$ as the embedding dimension in most of the results.

### 2.5 $k$-Nearest Neighbor Forecasting

The $k$-Nearest neighbor method is used to select the $k$ number of segments from the available time series that are similar to the most recent segment available before the observation we want to forecast. So histories with a similar dynamic behavior are detected in the past series and employed afterward in the prediction of the next term at the end of the series.

In the $k$-NN forecasting algorithm, $m$ (embedding dimension), $k$ (number of nearest neighbors), and $\tau$ (delay time) are pre-determined parameters. In the literature, the optimal values of $m$ and $k$ are quite ambiguous. There has been quite a lot of argument and discussion about the optimal choice.
of \( m \) and \( k \) since the NN rule was first officially introduced by Cover and Hart in 1967 [15, 35, 42].

Once the parameters, \( \tau, m, \) & \( k \) are determined, the future forecasts can be obtained using the information from the past time series through the following steps.

In order to identify behavioral patterns in the data, first the scalar time series \( (x_1, x_2, \ldots, x_t, \ldots, x_T) \) is transformed into vectors. We consider the segments of equal lengths as vectors of \( m \) observations sampled from the original time series at intervals of \( \tau \in \mathbb{N} \) units as follows:

\[
x_{t}^{m, \tau} = (x_{t}, x_{t-\tau}, \ldots, x_{t-(m-1)\tau}); \quad t = m, m+1, \ldots, T
\] (2.18)

Here \( m \) is referred to as the embedding dimension, and \( \tau \) is the delay parameter. The purpose of considering segments of equal length is to detect behavioral patterns in the time series. These \( m \)-dimensional vectors are often called \( m \)-histories [35]. In this manner, we can write the most recent available vector, which is called the delay vector as

\[
x_{T}^{m, \tau} = (x_{T}, x_{T-\tau}, x_{T-2\tau}, \ldots, x_{T-(m-1)\tau})
\] (2.19)

The primary goal of \( k \)-nearest neighbor forecasting is to use the most relevant \( k \) vectors out of the available \( m \)-history vectors to predict the time series value at the next time step, \( t = T + 1 \). The most relevant vectors are the ones having similar dynamic behavior as the delay vector \( x_{T}^{m, \tau} \). The \( k \) number of \( m \)-histories similar to the delay vector are selected using a similarity measure. The similarity measure in the \( k \)-NN forecasting algorithm is a distance function. As the classification mainly depends on chosen distance function, it is highly important to choose an appropriate distance measure that captures the behavior of the time series.

The distance, \( d(x_{T}^{m, \tau}, x_{t}^{m, \tau}) \), between the delay vector and all the other \( m \)-history vectors is calculated in order to select the vectors with similar dynamic behavior [15, 36]. Then we look for the closest \( k \) vectors in the phase space \( \mathbb{R}^m \) such that they minimize the distance function \( d(x_{T}^{m, \tau}, x_{t}^{m, \tau}) \). The selected \( k \)-history vector matrix can be given as follows:
The next task is to predict the value for \( t = T + 1 \). Several methods were suggested and applied by many authors. Adequate average, weighted average and local linear regression are some of the available forecasting methods in the literature [15, 36]. The local linear regression is the most commonly used forecasting technique among NN community.

The following locally adjusted linear regression model [15, 33] can be used to forecast the next value, \( x_{T+1} \) of the time series.

\[
\hat{x}_{T+1} = \sum_{i=0}^{m-1} \hat{a}_i x_{T-(i-1)\tau} + \hat{a}_m
\]  

(2.21)

The estimated coefficients \( \hat{a}_i \) are the values of \( a_i \) that minimize

\[
\sum_{j=1}^{k} \left( x_{t_j+1} - a_0 x_{t_j} - a_1 x_{t_j-1} - \ldots - a_{m-1} x_{t_j-(m-1)\tau} - a_m \right)^2
\]  

(2.22)

The coefficients were fitted by the linear regression of \( x_{t_j+1} \) on \( x_{t_j} = (x_{t_j}, x_{t_j-\tau}, \ldots, x_{t_j-(m-1)\tau}) \) for \( j = 1, 2, \ldots, k \). Therefore, the only data used in the above regression model are the values obtained from the \( k \)-neighbor vectors of size \( m \) and the corresponding next values, \( x_{t_j+1} \) for \( j = 1, 2, \ldots, k \), not the entire data.

The \( k \)-nearest neighbor forecasting procedure can be considered as a two class classification problem in which the data belonging to one class are the only data used to predict the future values.
Also, at each instant the coefficients of the local regression model are updated using the chosen \( m \)-histories. Thus, \( k \)-NN is an on-line learning procedure that can be well adopted to many areas of time series data analysis and forecasting.

Since its first official introduction in 1967 by Cover & Hart [19], the nearest neighbor rule has been used in many areas as a classification method as well as a forecasting technique. Even though the earlier work on NN were more focused on data clustering, after 1981 with the Takens delay embedding and state phase reconstruction theory, the researchers were more interested using \( k \)-NN forecasting method for time series [15, 16, 36, 71].

In 1987, J. D. Farmer and John J. Series used \( k \)-NN as their forecasting technique for the chaotic time series data obtained from different systems, such as the logistic map, the Mackey-Glass delay differential equations, the Rayleigh-Benard convention, and the Taylor-Couette flow [32]. They verified the effectiveness of local approximation using only nearby states to make predictions, and the forecasts were compared using normalized root mean square error [15, 32].

The published work of Martin Casdagli (in 1989, 1991, & 1992) considered as a new era of \( k \)-NN forecasting. He applied \( k \)-NN method with maximum norm to experimentally generated time series from coupled diodes [15]. He also used \( k \)-NN forecasting for numerically generated noise-free time series data, fluid turbulence data, light intensity data and many more [16, 17]. The subsequent extension of Takens time-delay embeddings provided by Also, Casdagli, Sauer, and Yorke in 1991 was considered as a useful theoretical contribution in the area of time series forecasting [17, 69, 70].

As the machine learning techniques were more popular in engineering and science related areas, applications of nearest neighbor in the areas of Economic and Finance were not that popular in the early stage of nearest neighbor forecasting. In 1998, Fernando Fernández-Rodríguez, S. Sosvilla-Rivero and J. Andrada-Félix proposed a nearest-neighbor based procedure to test the existence of nonlinear forecastable dependencies in exchange rates from European Monetary System (EMS) [37]. They have used the Euclidean distance as their similarity measure.
In their work on forecasting the exchange rates of five EMS currencies, Fernando Fernández-Rodríguez, S. Sosvilla-Rivero and, J. Andrada-Félix provided evidence of their proposed simultaneous nearest neighbor method (SNN) performing better than the traditional ARIMA (1,1,0) model the majority of the time [36]. Their obtained results were able to contradict the earlier claims, by Diebold & Nason in 1990 and Satchell & Timmermann in 1995, that the nonlinear predictors are outperformed by the random walk forecasts [36]. The same authors again compared the trading performances (using a technical trading strategy) of NN and SNN with the loving average (MA) technical trading strategy using 9 different currencies from the EMS [35]. Their work verifies that the majority of the time (8 out of 9), the nearest neighbor based technical trading strategy performs better compare to that of MA. They pointed out the potential usefulness of NN in the daily exchange rates market [33, 34, 35].

Following the work of Fernández-Rodríguez et al., many researchers have applied the $k$-nearest neighbor forecasting method in different financial markets during recent years. In 2004, Alicia Troncoso Lora et al.[56] used a time series prediction method based on the $k$-NN technique to the Spanish short-term electric load forecasting problem and compared the predictions with the conventional regression method, applying both models to Spanish transmission system. Considering both in terms of minimum forecasting errors and maximum average, they concluded that the $k$-NN classifier is more accurate for the load forecasting problem than the conventional regression method [56].

In 2003, John Barkoulas et. al. published their work on modeling the stochastic behavior of changes in several short and long term U.S. interest rates using nearest neighbor method. They compared the performances of their method with autoregressive model and a random-walk with-drift mode using root mean square error (RMSE) and concluded that NN exhibits greater out-of-sample forecasting accuracy than that of the linear predictors for most U.S. interest rate series [7].

M. V. Subha and S. Thirupparkadal Nambi [77] used $k$-nearest neighbor as a data mining tool to predict the direction of stock index movement of the popular Indian stock market indices BSE-SENSEX and NSE-NIFTY. Their distance choice for the $k$-NN was Euclidean distance. Comparing
their results to the logistic regression model, they concluded that $k$-NN outperforms the traditional logistic regression method when predicting the future movement of the stock indexes [77].

The nearest neighbor method was applied to five US stock market data sets by Teresa Aparicio et al. in 2002. The aim of their paper, “The Nearest Neighbor Method as a Test for Detecting Complex Dynamics in Financial Series: An Empirical Application,” was to analyze whether those stock time series presented a chaotic dynamic [3]. They also studied the economic value of their forecasts using a simple trading strategy and confirmed the adequacy of the nearest neighbor method.

The multivariate $k$-nearest neighbor regression model was used to predict economic indicators, including GDP modeling [45]. The method was applied for economic indicators of the euro area and compared with other classical ARMA-GARCH modelling [45]. The multivariate $k$-NN regression model was applied to 9-year S&P 500 stock data by Tao Ban, Ruibin Zhang, Shaoning Pang, Abdolhossein Sarrafiad, and Daisuke Inoue in recent years [6]. They provided empirical evidence to support his claim that his method shows better forecasting accuracy compared to the univariate $k$-NN regression.

In 2013, Khalid Alkhatib et. al. applied a $k$-nearest neighbor based forecasting technique to predict stock prices for a sample of six major companies listed on the Jordanian stock exchange. The results were compared with non-linear regression model. According to their results, nearest neighbor approach displayed reasonably better results than that of non-linear regression [2].

S. B. Imandoust et al. provided some theoretical background of the $k$-Nearest Neighbor classification method for economic forecasting in their recently published work [50]. Their focus was to predict companies’ financial distress with the $k$-NN method. They concluded that even though most efforts have exploited traditional statistical methods, nonparametric techniques, such as nearest neighbor, demonstrate more promising results than the traditional approaches [50].

The recent work on nearest neighbor has suggested that improving the basic version by introducing procedures, such as simultaneous nearest neighbor (SNN) [34], cluster $k$-nearest neighbor, and
probabilistic nearest neighbor [49], can lead to better performances.

On the other hand, many studies have recommended (both empirically and theoretically) to use a distance metric that can be well adopted to time series data, in order to improve the accuracy of the NN classification and forecasting [85, 88]. A proper distance metric is highly important in the \( k \)-NN forecasting procedure, as the later half of the algorithm is completely depending on the chosen \( m \)-histories. Previous work by several authors has shown that, having a better similarity metric than standard Euclidean distance can significantly improve the \( k \)-NN forecasting [25, 48, 67, 85, 88].

### 2.6 Conclusion

In the present chapter, we began with surveying several major literatures on \( k \)-NN forecasting in the area of time series forecasting, and data classification. We then defined some fundamental concepts of \( k \)-nearest neighbor algorithm and provided mathematical and statistical background behind the nearest neighbor forecasting. Furthermore, we discussed the error bounds, and convergence properties of the \( k \)-NN classifier. In the later half of the chapter, we provided references for the development of nearest forecasting method in various areas of finance, and specified the importance of choosing an appropriate similarity measure than standard Euclidean distance.
Chapter 3

$k$-Nearest Neighbor Forecasting with Mahalanobis Distance

3.1 $k$-Nearest Neighbor Algorithm and the Choice of Distance.

The principle behind the nearest neighbor methods is to find a predefined number of training samples closest in distance to the desired point. Many researchers have pointed out the difficulty of choosing a distance measure for the NN algorithm that works well for different types of data [88]. Over the past decades, the most common choice of distance was Euclidean distance [23, 85]. Apart from Euclidean distance, several other distance measures such as Manhattan, Minkowski, and Hamming distances have been used in the algorithm for various types of classification problems [85].

Cover and Hart were the first to study the asymptotic convergence of NN rule in 1967 [19]. Their result indicates that the asymptotic probability of error $R_{NN}$ of NN rule is bounded by the Bayes error $R^*$, $R^* \leq R_{NN} \leq 2R^*(1 - R^*)$ regardless of the distance metric [71]. This property was a great support for using the nearest neighbor rule as the large sample case can be expected to perform within twice the minimal possible error [19, 42, 71].

Even though the asymptotic probability of error of the NN is independent of the choice of metric, classification performance of the finite sample nearest neighbor algorithm is not independent of the distance function [42, 71]. As the nearest neighbor rule is highly sensitive to outliers, selecting irrelevant neighbors can increase the forecasting error. Thus, clearly the accuracy of k-nearest neighbor classification significantly depends on the choice of the metric [88].

In their work, Fukunaga & Hostetler showed that using a proper distance measure, the variance of the finite sample estimate can be minimized [42]. Short & Fukunaga investigated the relation between the distance function and the error measure in the $k$-NN classifier [71]. They conclude that the error can be minimized by using an appropriate distance metric without increasing the number of sample vectors.

Similar to any other technical analysis method, nearest neighbor prediction model is also com-
pletely relies on the historical data. When applying for foreign exchange rate forecasting, its main
goal is to investigate the past behavior of the currency rates so that it can fully capture the depen-
dency structure of the future exchange rates and that of the past. We look for the repetitions of
specific price patterns such as major trends, and critical or turning points. Based on the distance
between the query vector and the historical values, we rank the data and chose $k$ number of closest
vectors. As the forecasting is totally dependent on the $k$ nearest neighbors, it is highly important to
use a distance function that captures the behavior of the data accurately.

3.2 Distance.

The distance matrices are often computed as the first step of data analysis. In general, the distance
between two objects describes how far apart the objects are. Distance is a rule of assigning positive
numbers between pair of objects (or points). Distance is a concrete way of describing an element
of some space is closer to or far away from another. Due to this reason, distance concept has been
widely used in the field of time series data clustering [16, 23]. In time series pattern recognition, an
appropriate distance function can categorize data into clusters by capturing the similarity or dissim-
ilarity between the data.

In Probability theory and Statistics, distance can be measured between two statistical objects such
as random variables, probability distributions, and samples. Some of these statistical distances do
not satisfy the metric properties of a standard distance. For example, correlation has been used to
measure the distance between two samples [36].

In Mathematics, a distance function is usually called a metric. It is a generalization of the concept
of physical distance.

**Definition 3.2.1** Let $\Omega$ be a nonempty set. A function $d : \Omega \times \Omega \rightarrow \mathbb{R}$ is called a metric (or a
distance function) on $\Omega$ if the following conditions are satisfied for all $x, y, z \in \Omega$.

(i) Non-negativity: $d(x, y) \geq 0$

(ii) Coincide axiom: $d(x, y) = 0$ if and only if $x = y$

(iii) Symmetry: $d(x, y) = d(y, x)$
(iv) Triangular inequality: \( d(x, z) \leq d(x, y) + d(y, z) \)

The pair \(<\Omega, d>\) is called a metric space. In general, elements of the set \(\Omega\) are called points of the metric space and \(d(x, y)\) referred as the distance between points \(x, y\).

**Definition 3.2.2** A pseudometric space is a generalized metric space. Unlike a metric space, points in a pseudometric space need not be distinguishable. That is, one may have

\[
d(x, y) = 0 \text{ for distinct values } x \neq y
\]

In the next subsections, we will briefly discuss some of the distance measures commonly used in NN algorithm and our proposed Mahalanobis distance measure.

### 3.2.1 The Euclidean Distance

The Euclidean distance between \(x\) and \(y\) in \(\mathbb{R}^n\) is defined as

\[
d(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}
\]  

(3.1)

Euclidean distance is a function which calculates the real straight line distance between two points. It is the most common distance of choice in NN algorithms. Even though it works well for low dimensional data, it performs poorly when the data are high dimensional. Also, Euclidean is not the best distance choice when the data are highly correlated as it does not account the for correlation among the vectors.

### 3.2.2 Manhattan Distance

Manhattan distance gets its name from the rectangular grid patterns of the streets in Manhattan [84]. The Manhattan distance between \(x\) and \(y\) in \(\mathbb{R}^n\) is given by

\[
d(x, y) = \sum_{i=1}^{n} |x_i - y_i|
\]  

(3.2)
As it looks at the absolute difference between the coordinates, the most common and appropriate name for this distance measure is absolute value. It is also recognized as a computationally simplified version of Euclidean distance. Manhattan distance is preferred to Euclidean distance in practice sometimes, because the distance along each axis is not squared in the Manhattan approach. Thus if there is a large difference in one of the dimensions it will not affect the total outcome.

### 3.2.3 The Mahalanobis Distance

The Mahalanobis Distance was introduced in 1936 by P. C. Mahalanobis considering the possible correlation among the data [57]. It is defined between two vectors $x$ and $y$ as:

$$d(x, y) = \sqrt{(x - y)^\prime \sum^{-1}(x - y)}$$

(3.3)

Here, $\sum^{-1}$ is the inverse of variance-covariance matrix $\sum$ between $x$ and $y$ and $'$ denotes the transpose of the matrix. The major difference in Mahalanobis to any other distance measure is that it takes the covariance into account. Due to this reason it is also called the statistical distance as well. The Mahalanobis distance belongs to the class of generalized ellipsoid distance defined by

$$d(x, y) = \sqrt{(x - y)^\prime M(x - y)}$$

(3.4)

Here $M$ is a positive definite, symmetric matrix. In the case of Mahalanobis distance, the matrix $M$ becomes the inverse of variance-covariance matrix. Obviously, this includes Euclidean distances as a special case when $M$ is the identity matrix.

When using Euclidean distance, the set of points equidistant from a given location is a sphere. The Mahalanobis distance stretches this sphere to correct for the respective scales of the different variables, and to account for correlation among variables [84]. As the axes of ellipsoidal sphere can assume any direction depending upon the data (as can be seen figure 5), this is more applicable in the area of time series pattern recognition. Thus, unlike dimensional Euclidean distance, it is possible to express the correlation and weight between dimensions using Mahalanobis distance. Due to these advantages, Mahalanobis distance captures the correlation and the trend of the time series, better compared to other distances [23, 57].
In our work, we have applied $k$-NN algorithm with the above mentioned distance choices for 5 different currency data. Our goal is to compare the performances of the Mahalanobis based algorithm with other distance choices. When comparing the performances, our main focus was based on the following two aspects:

(i) Forecast accuracy

In this case, our primary goal was to capture the deviation of the fitted values against the actual observations. The Various accuracy measures (given in sec 3.3) were use to compare the performances of each distance choice.

(ii) Transforming their forecasts in to a technical trading rule

Here, we were interested to evaluate the forecasts in terms of financial point of view. Thus, we created trading signals, buy and sell, using a technical trading rule [35, 36], and the performances were evaluated by the commonly used performance measures in practice (given in section 3.5.2).

### 3.3 Measures of Forecasting Accuracy.

Let $x_t$ and $\hat{x}_t$, for $t = 1, 2, ..., n$ be the actual and fitted values of the time series data, respectively. To determine the forecasting accuracy of the prediction model for $n$ number of out-of-sample
predictions, the following error measures were used.

### 3.3.1 Mean square error (MSE).

Mean square error defined by

\[ MSE = \frac{1}{n} \sum_{t=1}^{n} (\hat{x}_t - x_t)^2 \]  

(3.5)

is the most commonly used accuracy measure in statistics to determine the difference between the actual and estimated values. It is a scale dependent measure, but gives a basis to compare the forecasts. Due to squaring, MSE gives disproportionate weight to larger errors.

### 3.3.2 Mean absolute percentage error (MAPE).

Mean absolute percentage error is another widely used accuracy measure for non-negative observations. This measure gives a forecasting accuracy as a percentage, so we can compare the errors of the fitted time series that differ in levels.

\[ MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{\hat{x}_t - x_t}{x_t} \right| \]  

(3.6)

Also, the mean absolute percentage error is not affected by larger deviations as MSE. It is zero for a forecasting model with a perfect fit. However, there is no restriction of its upper bound.

### 3.3.3 Theil’s $U$ - statistic ($U$).

We considered the following version of Theil’s $U$ - statistic to compare the forecasting accuracy of our model.

\[ U = \frac{\sqrt{\sum_{t=1}^{n} (\hat{x}_t - x_t)^2}}{\sqrt{\sum_{t=1}^{n} (\hat{x}_t)^2} + \sqrt{\sum_{t=1}^{n} (x_t)^2}} \]  

(3.7)

This is a measure of the degree to which the forecasted values differ from the actual values. $U$
statistic is independent of the scale of the variable, and constructed in such a way that it necessarily lies between zero and one, with zero indicating a perfect fit. However, U statistic does not provide information on forecasting bias, which is better captured by mean square error.

3.3.4 Normalized Root Mean Square Error (NRMSE).

Scale invariant forms of mean square error (MSE) are useful because, often we want to compare the forecasting errors in different scales. The non-dimensional version of MSE we adopted here is the normalized root mean squared error given by:

\[
NRMSE = \sqrt{\frac{\sum_{t=1}^{n} e^2(t)}{\sum_{t=1}^{n} x_t - \hat{x}_t}^2}
\]  

(3.8)

Here, \( \sigma \) is the standard deviation of the time series [1]. The normalized root mean squared error is a frequently used error measure to evaluate the difference between the values predicted by a model and actual observations.

Foreign exchange data trading data contains many directional changes. Thus it is highly important to figure out which method predict better at the places where directional changes occur. The following measure provides a proportion of a accurate directional predictions.

3.3.5 Sign Correction Proportion (SCP).

The sign correction proportion is a measure of forecasting accuracy, which calculates the proportion of accurately captured directional changes by a forecasting model.

Directional change in the times series data is defined as:

\[
d(i) = \begin{cases} 
1 & \text{if } x(t + 1) - x(t) > 0 \\
-1 & \text{if } x(t + 1) - x(t) < 0 
\end{cases}
\]  

(3.9)

In order to identify whether a model forecasts in the same direction as the real data, we compared the directional changes using the forecasted values and actual data. Let \( \delta \) be the function that takes
value 1 when the forecast is in same direction as of the actual value, and 0 otherwise. Then, $\delta$ can be defined as:

$$\delta(i) = \begin{cases} 
1 & \text{if } \hat{d}(i) = d(i) \\
0 & \text{if otherwise}
\end{cases}$$  \hspace{1cm} (3.10)$$

Here, $\hat{d}(i)$ values are the directional changes obtained by the forecasting model. The sign correction proportion (SCP) for $n$ forecasted values can be calculated as

$$SCP = \frac{1}{n} \sum_{i=1}^{n} \delta(i)$$  \hspace{1cm} (3.11)$$

This measure is a widely used accuracy measure to observe how well a model can capture the direction of the time series [72]. Higher the SCP value, better the directional forecasting accuracy.

### 3.4 Selecting Embedding Dimension, ($m$) and Number of Nearest Neighbors, ($k$)

#### 3.4.1 Data

In this section, we first began with collecting foreign exchange rates from 5 different currencies. The data used here are exchange rates of Euro (EUR), British pound sterling (GBP), Swiss franc (CHF), Japanese Yen (JPY), and Canadian dollar (CAD) vis-à-vis American dollar (USD) obtained from the ProQuest Statistical Datasets. These are the daily spot rates of the currencies from January 2006 to December 2010. In order to make the comparison more effective, we have considered 1250 data from each currency, and taken 1000 data values as our training sample. The remaining 250 values were considered as the test sample. The coefficients of the model were updated every time when new information arrived.

#### 3.4.2 Embedding Dimension ($m$).

The choice of embedding dimension, $m$, for the time series is a key issue need to be addressed prior to making trading signals. Therefore, first we conducted an empirical investigation to find the optimal value of $m$. Our choice of $m$ was depending on the five FX rates data sets that was mentioned above. In this empirical investigation, we wanted to figure out whether the choice of $m$ is data dependent, and also distance dependent. Thus, the forecasting accuracy was compared using all the error measures mentioned in section 3.3, by varying the value of $m$ along with different distance
80% of the data was considered as the training set, and the remaining 20% was taken as the testing set. The value of $k$ was set to be 2% of the training sample for this part of the work [35]. The training window size was kept fixed by removing the farthest data value at each time step.

Table 3.4.2 shows the U-statistic values for different embedding dimension, $m$, with Mahalanobis distance. We started with $m = 3$ and increased the value of $m$ up to 6. According to our observations, larger $m$ values did not improve the accuracy of the model. It clearly indicates that for almost all the data sets, the embedding dimension, $m = 3$, gives the minimum error according to U-statistics. These observations are also graphically represented in figure 7. The $U$-statistic value for $m = 6$ is slightly smaller than that of $m = 5$ for JPY/USD rates, however not the smallest. Also, in the case of CAD/USD data, $U$-statistic slightly drops when $m$ changed from 4 to 5, but still $m = 3$ gave the minimum.

We did not make our conclusion only by considering the $U$-statistic. We tested all 5 currencies with mean square error, mean absolute percentage error, and normalized root mean square error as well. All of the error measures indicated $m = 3$ as the optimal value.
Table 3: $U$-statistic with Mahalanobis distance

<table>
<thead>
<tr>
<th>Currency</th>
<th>$m = 3$</th>
<th>$m = 4$</th>
<th>$m = 5$</th>
<th>$m = 6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.00389801</td>
<td>0.00392875</td>
<td>0.00430221</td>
<td>0.00456411</td>
</tr>
<tr>
<td>GBP</td>
<td>0.00345430</td>
<td>0.00348754</td>
<td>0.00359710</td>
<td>0.00362095</td>
</tr>
<tr>
<td>JPY</td>
<td>0.00690517</td>
<td>0.00717526</td>
<td>0.00810838</td>
<td>0.00771370</td>
</tr>
<tr>
<td>CHF</td>
<td>0.00555915</td>
<td>0.00572837</td>
<td>0.00586630</td>
<td>0.00590578</td>
</tr>
<tr>
<td>CAD</td>
<td>0.00597986</td>
<td>0.00642774</td>
<td>0.00612342</td>
<td>0.00643873</td>
</tr>
</tbody>
</table>

These observations did not change significantly even when we replaced the Mahalanobis distance with Euclidean or Absolute distances. The Figure 8 demonstrates the results obtained by comparing $U$-statistics for different $m$ values with Euclidean distance. It verifies that even with a different distance function, the error measure suggests to use relatively smaller value such as 3 as the embedding dimension, $m$.

The tables 3.4.2 and 3.4.2 summarize the obtained results for the choice of embedding dimension for all 5 currencies with U-Statistic and normalized root mean square error. (This part of the work was published in the proceedings of Nonlinear Dynamical Systems and Application, [66])
Figure 8.: U statistic vs. Embedding Dimension with euclidean Distance

Table 4: Optimal Choice of Embedding Dimension with $U$-Statistic

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>GBP</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>JPY</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CHF</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CAD</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 5: Optimal Choice of Embedding Dimension with Normalized Root Mean Square Error

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>GBP</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>JPY</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CHF</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CAD</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Almost all the data sets agreed with the conclusion of $m = 3$ being the optimal embedding dimension. We were able to obtain similar results by comparing MSE and MAPE.
3.4.3 Number of Nearest Neighbors ($k$)

As the next step, we wanted to investigate the effect of number of nearest neighbors ($k$) in the forecasting procedure before comparing the performance of each distance choice. In time series data forecasting, there isn’t a uniform guideline to select the neighborhood size [35, 36]. Especially for the financial data, the approaches used were diverse. Thus, it is often difficult to come up with a unique method [35, 36]. The approach used in this work is the commonly used Casdagli’s (1991) algorithm [15, 16, 36].

In Casdagli’s work [15, 16], the optimal value of $k$ was determined empirically, by testing the algorithm for several value of $k$ between $2(m + 1)$ and $T - (m - 1)$. His approach was to minimize normalized root mean square error, by changing the value of $k$. He also varied the embedding dimension, $m$, and studied the behavior of NRMSE as a function of $k$. His choice of distance was Euclidean distance.

In this work, we have chosen the neighborhood size, $k$, after comparing not only NRMSE, also all the accuracy measures discussed in section 3.3. We did not restrict our distance choice only to the Euclidean distance. We tested the data for $k$($k > m + 1$) using Mahalanobis, Euclidean and Absolute distances. Furthermore, this method was applied to all five currencies before selecting an appropriate value of $k$ for further analysis.

Even though the numerical values are slighting different, all the data sets behaves exactly the same way as $k$ changes. We started with $k = 5$ as $k$ needs to be greater than $m + 1$, which is 4 in our case (as $m = 3$). It can be clearly seen from Figures 9 and 10, for values of $k$ from 5 to 15, the change in U-statistic was significant for all the currencies. Thereafter, even though the forecasting error decreases, the difference is much smaller. Especially, after $k = 20$, the changes are negligible. Therefore, we did not want to increase $k$ further. Also, as $k$-NN method is a data reduction technique, the larger values of $k$ does not justify its primary goal. On the other hand, having a significantly larger value for $k$ may even result in choosing the neighbors that are not very effective on forecasting. The other accuracy measures also supported our conclusion of having an intermediate value as the neighborhood size is appropriate.

We wanted to extend our search for an optimum choice of $k$, with different choices of $m$ as well. For that reason, we considered $m = 4, 5$ and 6 and compared the forecasting accuracy as a function...
Figure 9: U-Statistic vs. $k$ with Mahalanobis Distance

Figure 10: U-Statistic vs. $k$ with Euclidean Distance
of $k$. Similar analysis were performed as of $m = 3$, and behavior of error measures along with the
different distances were investigated. The obtained results were not much of different to those of
$m = 3$.

Even though with a higher $m$ value we need slightly larger $k$, earlier analysis of $m$ indicated that
increasing $m$ does not improve the forecasting accuracy. Considering all these facts, to compare
the performance of Mahalanobis distance based $k$-nearest neighbor algorithm with other distance
choices, the key parameters $m$ and $k$ of the algorithm were chosen as 3 and 20, respectively.

3.5 Comparison of Distance Measures with Respect to the Forecasting Accuracy and Buy
and Sell Decisions.

3.5.1 Forecasting Accuracy

After selecting the key parameters for the $k$-NN forecasting procedure, next step was to compare the
performance of proposed Mahalanobis distance based $k$-NN algorithm with traditional Euclidean
and Absolute distance based algorithms. For the first half, we considered all the accuracy measures
discussed in section 3.3, and compared the performances of each algorithm based on how accurate
their forecasts were. The following tables, given on page 41, the summarized the results for the
currencies EUR, GBP, JPY, CHF, and CAD with the choice of $m = 3$ and $k = 20$.

In table 3.5.1 and 3.5.1 below (page 41), we report the forecasting accuracy for Mahalanobis, Eu-
clidean and absolute distance using $U$-statistic and normalized root mean square error respectively.
The observations were similar for the other error measures as well.

<table>
<thead>
<tr>
<th>Table 6: $U$-Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Currency</td>
</tr>
<tr>
<td>EUR</td>
</tr>
<tr>
<td>GBP</td>
</tr>
<tr>
<td>JPY</td>
</tr>
<tr>
<td>CHF</td>
</tr>
<tr>
<td>CAD</td>
</tr>
</tbody>
</table>
Table 7: Normalized Root Mean Square Error

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.16793291</td>
<td>0.17332175</td>
<td>0.17155159</td>
</tr>
<tr>
<td>GBP</td>
<td>0.22321725</td>
<td>0.24333251</td>
<td>0.24104249</td>
</tr>
<tr>
<td>JPY</td>
<td>0.29681857</td>
<td>0.42612478</td>
<td>0.58432940</td>
</tr>
<tr>
<td>CHF</td>
<td>0.21983992</td>
<td>0.29457650</td>
<td>0.30101833</td>
</tr>
<tr>
<td>CAD</td>
<td>0.58313581</td>
<td>0.58857894</td>
<td>0.59790435</td>
</tr>
</tbody>
</table>

From the obtained results, it is clear that Mahalanobis distance outperforms the other distance measures for all the currencies. Except for the normalized root mean square error for CAD/USD rates, all the other values with Mahalanobis distance were significantly smaller compared to other distances. Especially, in the cases of JPY/USD and CHF/USD rates, the error measures of the proposed algorithm were much smaller than the traditional \( k \)-NN forecasting with Euclidean and Absolute distances.

When comparing sign correction proportion (SCP), the obtained results did not support our previous observations. As can be seen from the table 3.5.1 below, 3 out of 5 currencies indicate that Euclidean distance based method has a higher correction proportion than Mahalanobis based method.

Table 8: Sign correction proportion (SCP) with Mahalanobis and Euclidean distances

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.448</td>
<td>0.52</td>
</tr>
<tr>
<td>GBP</td>
<td>0.496</td>
<td>0.516</td>
</tr>
<tr>
<td>JPY</td>
<td>0.524</td>
<td>0.5</td>
</tr>
<tr>
<td>CHF</td>
<td>0.576</td>
<td>0.548</td>
</tr>
<tr>
<td>CAD</td>
<td>0.584</td>
<td>0.592</td>
</tr>
</tbody>
</table>

The results of SCP show that Mahalanobis distance performed better than Euclidean distance only for JPY and CHF data sets. This was a bit surprising as all the other accuracy measures supported Mahalanobis based algorithm.
Due to this reason, we wanted to conduct a thorough investigation of directional forecasting accuracy. Thus, we calculated mean absolute deviation (MAD) and mean square error (MSE) only for the places where the model did not forecast in the same direction as the actual values. In this way, we can measure how forecasted values were deviated from the actual values even though they are not in the same direction as actual values.

This analysis showed that MAD and MSE values were smaller for Mahalanobis distance compared to Euclidean distance. The question arise here is that, can a model have a lower sign correction proportion and still have a smaller MAD? What’s really happening here is even though the Euclidean based method captures the direction of the real data somewhat better than Mahalanobis; forecasted values can be way off the actual values giving a higher deviation.

Table 9: Mean Absolute Deviation (MAD) with Mahalanobis and Euclidean distance measures

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.010498383</td>
<td>0.011265078</td>
</tr>
<tr>
<td>GBP</td>
<td>0.01103014</td>
<td>0.012782458</td>
</tr>
<tr>
<td>JPY</td>
<td>0.00896291</td>
<td>0.023209785</td>
</tr>
<tr>
<td>CHF</td>
<td>0.007168656</td>
<td>0.012757111</td>
</tr>
<tr>
<td>CAD</td>
<td>0.007185595</td>
<td>0.009144574</td>
</tr>
</tbody>
</table>

Table 10: Mean Square error (MSE) with Mahalanobis and Euclidean distance measures

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.000146502</td>
<td>0.000164365</td>
</tr>
<tr>
<td>GBP</td>
<td>0.000169709</td>
<td>0.000204144</td>
</tr>
<tr>
<td>JPY</td>
<td>0.000302811</td>
<td>0.001211702</td>
</tr>
<tr>
<td>CHF</td>
<td>0.000115296</td>
<td>0.000458595</td>
</tr>
<tr>
<td>CAD</td>
<td>0.000161336</td>
<td>0.000335663</td>
</tr>
</tbody>
</table>

These results indicate that a model may be able to predict more accurately in the same direction as the actual time series, but may still have a larger deviation with the actual values. Therefore, we can conclude that even though the Mahalanobis distance based method seems to perform slightly weaker comparing SCP measure, it forecasts were much closer to the actual data compared to the
Figure 11: Forecasted and real values- JPY/USD rates

Euclidean distance based method.

Figure 11 shows the forecasted rates using Mahalanobis distance (blue color), and Euclidean distance (red color) with the real data for JPY/USD currency market from $t = 150$ to $t = 210$.

As can be seen from the Figure 11, the deviation of Mahalanobis distance based forecasts were much smaller compared to the Euclidean distance based forecasts.

Even though the accuracy measures verify how well a model forecasts, it is highly recommended that we give an economic value for the predicted financial time series. In the next subsection, we will evaluate the economic significance of Mahalanobis distance based forecasting algorithm, and will compare it with standard Euclidean distance based method.

3.5.2 Trading Decisions.

As in any other financial market, in FX market also a trader’s main goal is to make more money out of foreign currency fluctuations. Thus, the primary goal of foreign exchange rate forecasting needs to be making proper trading signals, buy and sell, at each time step so that the trader can make more money. In order to satisfy this main aspect of FX trading, we need to transform the forecasts into trading signals.
Therefore, the obtained forecasts were transformed into a simple technical trading strategy with the trading rule used by Fernandez-Rodriguez, Sosvilla-Rivero, and Andrada-Felix in their work [35, 36]. Let \( \hat{r}_t \) given by

\[
\hat{r}_t = \ln(\hat{x}_{t+1}) - \ln(x_t) + \ln(1 + \hat{i}_t) - \ln(1 + i_t)
\]

be the estimated return from a foreign currency position over the period \((t, t + 1)\), based on the forecasted FX rate at time \(t\). Here \(x_t\) represents the spot exchange rate at time \(t\), \(\hat{x}_{t+1}\) is the forecasted value for time \(t + 1\), and \(i, \hat{i}\) are the domestic (US) and the foreign country daily interest rates, respectively. The trading signal at time \(t\) was made based on the estimated return \(\hat{r}_t\). The positive returns were executed as long positions (buy) and the negative returns were executed as short position (sell) [35, 36]. Thus, the trading decision can be given as

\[
\hat{z}_t = \begin{cases} 
1 & \text{if } \hat{r}_t > 0 \\
-1 & \text{if } \hat{r}_t < 0
\end{cases}
\]

(3.13)

Based on estimated return, we calculated estimated total (log access) return of the trading strategy over the time period \((1, n)\), which can be written as

\[
\hat{R}_n = \sum_{t=1}^{n} \hat{z}_t r_t
\]

(3.14)

Here, \(r_t\) is the actual return at time \(t\), given by

\[
r_t = \ln(x_{t+1}) - \ln(x_t) + \ln(1 + \hat{i}_t) - \ln(1 + i_t)
\]

We also considered the popular performance measure, Sharpe ratio to compare the results along with the estimated total return. The Sharpe ratio, \(S_R\) used here is the mean daily total return of the trading strategy over its standard deviation,

\[
S_R = \frac{\mu_{\hat{R}_n}}{\sigma_{\hat{R}_n}}
\]

(3.15)

Higher values of Sharpe ratio indicate that the model is performing better.

The estimated total return for the technical trading strategy under Mahalanobis distance based
Table 11: Estimated Return

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.52991777</td>
<td>0.47275687</td>
<td>0.46008299</td>
</tr>
<tr>
<td>GBP</td>
<td>4.16807227</td>
<td>4.13638609</td>
<td>4.05861762</td>
</tr>
<tr>
<td>JPY</td>
<td>0.67755404</td>
<td>0.22975657</td>
<td>0.48608330</td>
</tr>
<tr>
<td>CHF</td>
<td>5.42108879</td>
<td>5.16084868</td>
<td>5.16742874</td>
</tr>
<tr>
<td>CAD</td>
<td>4.38589604</td>
<td>3.76747181</td>
<td>4.01797807</td>
</tr>
</tbody>
</table>

nearest neighbor algorithm, and traditional Euclidean and Absolute distances based algorithms for the five currencies are given in table 3.5.2. The final conclusion of distance choice is pretty much same as that of the error measures. The proposed distance choice outperformed the traditional distance choices. The Sharpe ratio also supported our conclusion of choosing Mahalanobis as the distance in $k$-NN algorithm, as can be seen from the table 3.5.2.

Table 12: Sharpe Ratio

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.27890809</td>
<td>0.27932631</td>
<td>0.29488645</td>
</tr>
<tr>
<td>GBP</td>
<td>2.41593434</td>
<td>2.29776019</td>
<td>2.06031102</td>
</tr>
<tr>
<td>JPY</td>
<td>0.18429771</td>
<td>0.01026936</td>
<td>0.13113608</td>
</tr>
<tr>
<td>CHF</td>
<td>1.67419376</td>
<td>1.44768129</td>
<td>1.42328537</td>
</tr>
<tr>
<td>CAD</td>
<td>1.26400087</td>
<td>0.89502219</td>
<td>1.03678437</td>
</tr>
</tbody>
</table>

In their work, Fernández-Rodríguez et al. have compared trading decisions of moving average (MA) technical trading rule with their proposed correlation based $k$-NN decision procedure [35]. Their results showed that $k$-NN procedure (with correlation) outperforms the MA trading performances. In this work, we also compared the proposed Mahalanobis distance based $k$-NN forecasting with correlation based $k$-NN forecasting. Almost all the cased supported the conclusion of Mahalanobis distance based $k$-NN forecasting procedure leads to better trading decisions than the correlation based method. Correlation does not have the translation invariant structure like Mahalanobis distance. This is one of the major reason that the Mahalanobis distance captures the dependency structure in data more accurately, compared to the correlation.
For all these data sets, we have used same number of neighbors and same forecasting technique with each distance measure. From our results, we can clearly see that choosing an appropriate distance in the $k$-NN algorithm can improve the forecasting significantly. As the forecasted values directly effect on trading decision, more accurate forecasts will result in more profitable trading.

### 3.6 Further Analysis on $k$-Nearest Neighbor.

As the $k$ neighboring vectors play a key role in nearest neighbor forecasting, we further investigated on chosen nearest neighbors with different distance measures to observe how well a distance function captures the dynamic behavior of a delay vector. The Figures, 12 and 13 illustrate a delay vector and its 3 closets nearest neighbors with Mahalanobis, and Euclidean distances.

In each of those figures, the graph on the top left corner shows the same delay vector for JPY/USD data with $m = 3$. The other graphs are the 3 closest neighboring vectors selected by Mahalanobis distance (Figure 12) and Euclidean distance (Figure 13). As can be seen from the graphs, Mahalanobis distance captures the time series vectors which are much similar to the delay vector compared to Euclidean distance. These observations support our claim Mahalanobis distance as a better choice.
Even though Mahalanobis distance did a better job capturing relevant time series vectors as neighbors, when a delay vector has data values in different directions, the selected neighbors do not exactly follow the same pattern. The results are similar with other distance choices are well. The Figure 14 on page 49 is an example of such case.

According to Figure 14, for a delay vector with data in different directions, the neighbors captured even by Mahalanobis distance did not behave similar to the delay vector. This is an issue in many time series data forecasting methods. When there are directional changes in data, forecasting methods do not perform well in practice. Due to this reason, the importance of detecting change points in data prior to forecasting have been a major part of time series data analysis [52, 73].

This is a minor setback of NN forecasting method as well. Even though a proper distance measure captures the dynamic behavior of the data, when there is a change point, the algorithm does not perform as good as the other places. To address this, we will work on combining change point detection methods with Mahalanobis distance based k-nearest neighbor algorithm as a part of our future work.
3.7 The $k$-Nearest Neighbor Forecasting with Distance Based Weighting.

In the standard nearest neighbor algorithm, the selected neighbors are usually given equal weights towards forecasting. As our next task, we wanted to compare the distance weighted $k$-NN forecasting with the traditional unweighted $k$-NN forecasting method. Weighting the data can be viewed as giving more importance to the most relevant instances and less importance to the least irrelevant instances. The $k$NN algorithm is a data mining procedure, which captures the most relevant instances. Thus, further weighting the selected neighbors was always questionable.

We used the popular tri-cube weight function defined by:

$$w(u) = \begin{cases} (1 - |u|^3)^3 & \text{for } |u| < 1 \\ 0 & \text{otherwise} \end{cases}$$

(3.16)

as our choice of weighting in this work. The weighting function $u(i)$ is defined as follows:

$$u(i) = \frac{d(i)}{d(k)}$$

Here, $d(i)$ is the distance to the $i^{th}$ nearest neighbor and $d(k)$ is distance to the $k^{th}$ nearest neighbor.
where $k$ is the fartherest neighbor considered. Obviously, farther the neighbor, smaller the weight. We compare the forecasting accuracy of locally weighted $k$-NN and standard $k$-NN algorithm with different values of $k$. We used both Mahalanobis and Euclidean distance for this analysis. The table 3.7 shows the $U$ statistics values for $k$-NN algorithm with Mahalanobis distance.

**Table 13:** $U$ Statistic with and without weighted regression Mahalanobis distance

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis weight $(k = 20)$</th>
<th>Mahalanobis weight $(k = 25)$</th>
<th>Mahalanobis weight $(k = 30)$</th>
<th>Mahalanobis weight $(k = 35)$</th>
<th>Mahalanobis weight $(k = 40)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.00389801</td>
<td>0.00447082</td>
<td>0.00429696</td>
<td>0.00415463</td>
<td>0.00406373</td>
</tr>
<tr>
<td>GBP</td>
<td>0.00345430</td>
<td>0.00369875</td>
<td>0.00360442</td>
<td>0.00353554</td>
<td>0.00347328</td>
</tr>
<tr>
<td>JPY</td>
<td>0.00690517</td>
<td>0.010751872</td>
<td>0.009973787</td>
<td>0.009461335</td>
<td>0.00888432</td>
</tr>
<tr>
<td>CHF</td>
<td>0.00555915</td>
<td>0.005971333</td>
<td>0.005872418</td>
<td>0.005811847</td>
<td>0.005759591</td>
</tr>
<tr>
<td>CAD</td>
<td>0.00597986</td>
<td>0.008575087</td>
<td>0.007683319</td>
<td>0.007121021</td>
<td>0.00676352</td>
</tr>
</tbody>
</table>

The first column in Table 3.7 gives the values for $U$-statistics without weighted regression for $k = 20$. With the same choice of $k$, when we introduced weights, the algorithm did not perform as good unweighted $k$-NN procedure. Then we increased the value of $k$ and performed the same analysis. Even with $k = 40$ we didn’t see results as good as without weighting.

When we further increased the number of nearest neighbors, the distance based weighted method started giving better results, which were almost as good as regular unweighted $k$-NN forecasting method. This idea does not support our primary goal of data reduction. Therefore, using Mahalanobis distance with less number of neighbors (20) with unweighted regression, we can have better forecasts compared to weighted regression. This comparison also supports our main idea of using an appropriate distance to select neighbors in the $k$-NN algorithm.

### 3.8 Conclusion.

As the finite sample $k$-nearest neighbor algorithm is depending on the distance function, choosing an appropriate distance measure can significantly improve the performances of the $k$-NN forecasting. We compared the accuracy of $k$-NN forecasting for foreign exchange data with traditional Euclidean and Absolute distance based algorithms with our proposed Mahalanobis distance based...
algorithm. In this work, we observed that the proposed Mahalanobis distance based method outperform Euclidean distance as well as the absolute distance based methods both in terms of better fit and in terms of the trading decisions. Also, we conducted a thorough analysis of choice of embedding dimension and neighborhood size for nearest neighbor algorithm using five different currencies with different distance choices. The observations suggested that smaller value of $m$ such as 3 and sufficiently large enough value of $k$ give significantly better results.
Chapter 4
Comparison of $k$-Nearest Neighbor Forecasting with Traditional Time Series Forecasting Methods

4.1 Introduction.

In this chapter our main goal is to compare the trading performances of Mahalanobis based $k$-nearest neighbor procedure with some of the popular time series forecasting methods. Even though our main concern is forecasting and decision making in the foreign exchange market, during the literature review of this chapter, we extended our literature search to time series forecasting in general.

In the first half of this chapter, we will discuss some of the most important time series forecasting methods, namely, Autoregressive (AR) process, Moving Average (MA) process, Autoregressive Moving Average (ARMA) process, and Autoregressive Integrated Moving Average (ARIMA) process. Then, we will go over the step-by-step model building procedure for the same 5 currency data sets used in chapter 3. Finally, we will compare the forecasting accuracy, and the trading performances of time series models with our proposed $k$-NN forecasting algorithm.

4.2 Stationary AR, MA and ARMA Models

The autoregressive process (AR) and the moving average process (MA) were very useful representation among the time series community over the past. Both of these models are only applicable to stationary time series data. Each method has its own pros and cons. The ARMA model combines the AR and MA processes to have a better forecasting in time series by taking advantages of both AR and MA methods.
4.2.1 The Autoregressive (AR) Models

In 1926, Yule first introduced the autoregressive model for time series data. It describes the present value of a time series by using its past values and a random shock.

Let $\epsilon_1, \epsilon_2, ..., \epsilon_t$ be a purely random process with mean zero and variance $\sigma^2$. Then the autoregressive process of order $p$, AR($p$) for the time series, $\{x_1, x_2, ..., x_t\}$ is given by;

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + \epsilon_t$$  \hspace{1cm} (4.1)

or

$$\Phi_p(L)x_t = \alpha + \epsilon_t$$  \hspace{1cm} (4.2)

where

$$\Phi_p(L) = 1 - \phi_1 L - \phi_2 L^2 - \phi_3 L^3 - ... - \phi_p L^p;$$  \hspace{1cm} (4.3)

with

$$L^j x_t = x_{t-j}$$  \hspace{1cm} (4.4)

The AR process is always invertible. To be stationary, the root of $\Phi_p(L) = 0$ must lie outside the unit circle [13, 53].

4.2.2 Moving Average (MA) Models

The moving average (MA) process was first introduced by Sluztky in 1927. This can be seen as a linear regression of the current value of the series against current and past white noise errors. A moving average process of order $q$, MA($q$) of the time series data can be written as;

$$x_t = \alpha + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q}$$  \hspace{1cm} (4.5)

or

$$x_t = \alpha + \Theta_q(L) \epsilon_t$$  \hspace{1cm} (4.6)

where

$$\Theta_q(L) = 1 + \theta_1 L + \theta_2 L^2 + \theta_3 L^3 + ... + \theta_q L^q; \quad \text{with} \quad L^j x_t = x_{t-j}$$  \hspace{1cm} (4.7)
A finite MA process is always stationary. The process is invertible if the root of $\Theta_q(L) = 0$ lie outside the unit circle [13, 53].

### 4.2.3 The General mixed Autoregressive Moving Average (ARMA) Model.s

The General ARMA($p,q$) process is a combination of an autoregressive process of order, $p$, and a moving average process of order, $q$. Herman Wold was the person who first put together AR and MA models to create ARMA process in 1938. Since then, this method has been used in many areas of time series. ARMA($p,q$) process is defined as:

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \ldots + \phi_p x_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q}$$  \tag{4.8}

or

$$\Phi_p(L)x_t = \alpha + \Theta_q(L)\epsilon_t$$  \tag{4.9}

where

$$\Phi_p(L) = 1 - \phi_1 L - \phi_2 L^2 - \phi_3 L^3 - \ldots - \phi_p L^p$$  \tag{4.10}

and

$$\Theta_q(L) = 1 + \theta_1 L + \theta_2 L^2 + \theta_3 L^3 + \ldots + \theta_q L^q$$  \tag{4.11}

The ARMA process is invertible if the roots of $\Theta_q(L) = 0$ lie outside the unit circle and stationary if the roots of $\Phi_p(L) = 0$ lie outside the unit circle [13, 53]. Note that we need to make the assumption of $\Theta_q(L) = 0$ and $\Phi_p(L) = 0$ sharing no common roots [13, 53].

### 4.3 The General ARIMA Models

All the above discussed time series forecasting methods are stationary processes. However, in reality, most of the time series are non-stationary. For non-stationary time series, roots of the AR polynomial do not lie outside the unit circle. Therefore, we are not able to use the general mixed ARMA($p,q$) model for forecasting. In such cases, the time series can be converted to a stationary process by differencing. This is also known as the integrated part of the algorithm and that is what
transforms the general ARMA($p,q$) process into an ARIMA($p,d,q$) process. Here $d$ is the degree of differencing. The difference filter is normally given by

$$(1 - L)^d \quad \text{where} \quad L^d x_t = x_{t-d}$$ (4.12)

Generally, $d$ will be a positive integer and represents the number of times $x_t$ must be differenced to achieve a stationary transformation. Typically, $d \in \{0,1,2,\ldots,d\}$. When $d = 0$, the ARIMA process becomes stationary ARMA process. Thus the autoregressive integrated moving average, ARIMA($p,d,q$) can be written as

$$\Phi_p(L)(1 - L)^d x_t = \alpha + \Theta_q(L)\epsilon_t$$ (4.13)

where

$$\Phi_p(L) = 1 - \phi_1 L - \phi_2 L^2 - \phi_3 L^3 - \ldots - \phi_p L^p$$

and

$$\Theta_q(L) = 1 + \theta_1 L + \theta_2 L^2 + \theta_3 L^3 + \ldots + \theta_q L^q$$

Consider a simple case when $p$, $q$, $d$ all equals 1. Then the ARIMA(1,1,1) model can be written as

$$\Phi_1(L)(1 - L)x_t = \alpha + \Theta_1(L)\epsilon_t$$

or we can expand the model as

$$(1 - \phi_1 L)(1 - L)x_t = \alpha + (1 + \theta_1 L)\epsilon_t$$

Which reduces to

$$x_t = \alpha + (1 + \phi_1)x_{t-1} - \phi_1 x_{t-2} + \epsilon_t + \theta_1 \epsilon_{t-1}$$

Sometimes, selecting the best order of the ARIMA($p,d,q$) is a challenging task. As the forecasting is strongly depending on the order of the model, it is highly important to pick the correct order. The procedure needs to be completed in two steps. First we need to figure out the differencing order, of the process. To determine the correct order of differencing, we continue the differencing procedure until the time series becomes stationary. The KPSS [54, 68] and Augmented Dickey-Fuller unit
Root tests \cite{68, 91} are normally used to determine the stationarity of a time series.

Once the correct differencing order, \(d\) is determined, the order of AR polynomial, \(p\), and MA polynomial, \(q\) are determined using either Akaike's information criterion (AIC). The AIC normally measures the quality of each model, relative to each of the other models. It is defined as

\[
\ln(L) = 2M - 2\ln(L)
\]

where, \(M\) is the number of parameters in the model, and \(\ln(L)\) is the unconditional log-likelihood function given by

\[
\ln(L) = -\frac{n}{2} \ln(2\pi \sigma^2) - \frac{1}{2} \sigma^2 \sum_{i=1}^{n} (x_i - \mu)^2
\]

Here, \(\mu\) and \(\sigma\) are the mean and the standard deviation of time series respectively. The AIC is calculated by changing the values of \(p\) and \(q\) in the ARIMA model, and the model with the smallest AIC is usually selected for forecasting.

### 4.4 Foreign Exchange Rates Forecasting with general ARIMA Process

In this section, we will first introduce the data preparation procedure for the same five currency data sets we used in chapter 3. Then, we will determine the appropriate ARIMA model for each data set, and finally compare the ARIMA approach with the Mahalanobis distance based \(k\)-NN forecasting method. In this chapter also, the comparison will be performed according to two main aspects of forecasting. As the primary step, we will consider the different error measures discussed in section 3.3 to compare the forecasting accuracy. As the secondary step, the ARIMA forecasts will be transformed into trading signals using the same technical trading strategy discussed in section 3.5.2 and compare with the trading performances of \(k\)-NN procedure.

As discussed in Section 4.3, the order of differencing will be determined using the KPSS test. For that, we will keep on differencing the series until the data becomes stationary. To figure out the order \(p\) of AR polynomial and \(q\) of MA polynomial, we are considering a positive constant \(m = 5\) with \(p + q = m\). Then, we will vary the values of \(p\) and \(q\) such that \(p + q \leq m\) and estimate the parameters; \(\phi_1, \phi_2, ..., \phi_p, \theta_1, \theta_2, ..., \theta_q\) of each ARIMA\((p,d,q)\) model. Akaike information criterion (AIC) is computed for each model to choose the model with the minimum AIC.
4.4.1 ARIMA Forecasting Model for EUR/USD Daily Rates

Following the step-by-step procedure we introduced above, the forecasting model with minimum AIC for the EUR/USD exchange rates data set was ARIIMA(1,1,1), that is a combination of first order autoregressive (AR), and a first order moving average (MA) with the first difference filter \(d = 1\). The model can be explicitly written with the estimated parameters as below:

\[
(1 - 0.1329L)(1 - L)x_t = 0.00019 + (1 - 0.1323L)\epsilon_t 
\]

\[
x_t = 0.000192 + 1.132915x_{t-1} - 0.132915x_{t-2} + \epsilon_t - 0.132342\epsilon_{t-1} 
\]

By letting \(\epsilon_t = 0\), we have the one day ahead forecasting time series for EUR/USD currency data as

\[
x_t = 0.000192 + 1.132915x_{t-1} - 0.132915x_{t-2} - 0.132342\epsilon_{t-1} 
\]

The figure 15 shows the predictions on top of the original time series for the forecasting period.
4.4.2 ARIMA Forecasting Model for GBP/USD Daily Rates

The forecasting model with minimum AIC for the GBP/USD exchange rates data set was ARIMA(1,1,2) model. This process is a combination of first order autoregressive (AR), and a second order moving average (MA), with the first difference filter. The model can be explicitly written with the estimated parameters as below:

\[(1 + 0.636309L)(1 - L)x_t = -0.000218556 + (1 + 0.652109L + 0.063216L^2)e_t \quad (4.19)\]

or

\[x_t = -0.000218556 + 0.363691x_{t-1} + 0.636309x_{t-2} + \epsilon_t + 0.652109\epsilon_{t-1} + 0.063216\epsilon_{t-2} \quad (4.20)\]

By letting \(\epsilon_t = 0\), we have the one day ahead forecasting time series for GBP/USD currency data as

\[x_t = -0.000218556 + 0.363691x_{t-1} + 0.636309x_{t-2} + 0.652109\epsilon_{t-1} + 0.063216\epsilon_{t-2} \quad (4.21)\]

The 250 out-of-sample forecast with the original time series are given in figure 16.

4.4.3 ARIMA Forecasting Model for JPY/USD Daily Rates

After comparing AIC for JPY/USD rates data set, we came up with the following ARIMA(1,1,2) model, that is a combination of second order autoregressive (AR), and a first order moving average (MA), with the first difference filter.

\[(1 - 0.664950L)(1 - L)x_t = 0.00000098 + (1 - 0.400612L - 0.234753L^2)e_t \quad (4.22)\]

Expanding the autoregressive operator and the difference filter and then letting \(\epsilon_t = 0\), we obtained one day ahead forecasting time series for JPY/USD currency data as

\[x_t = 0.00000098 + 1.0.664950x_{t-1} - 0.664950x_{t-2} - 0.400612\epsilon_{t-1} - 0.234753\epsilon_{t-2} \quad (4.23)\]

The figure 17 shows the forecasts with the actual values.
Figure 16. ARIIMA(1,1,2) Forecasts and real values for GBP/USD daily exchange rates.

Figure 17. ARIMA(1,1,2) Forecasts and real values for JPY/USD daily exchange rates.
4.4.4 ARIMA Forecasting Model for CHF/USD Daily Rates

For the CHF/USD daily rates, we found that ARIMA(1,1,1) as the model with smallest AIC by varying the values $p$ and $q$, after deciding the deference degree as one. The ARIMA process can be explicitly written with the estimated parameters as:

\[
(1 + 0.0881215L)(1 - L)x_t = 0.000231 + (1 + 0.371622L)\epsilon_t
\]  

(4.24)

Expanding the autoregressive operator and the difference filter and then letting $\epsilon_t = 0$, we obtained following one day ahead forecasting model:

\[
x_t = 0.000231 + 0.918785x_{t-1} + 0.0881215x_{t-2} - 0.0881215\epsilon_{t-1}
\]  

(4.25)

The figure 18 shows the forecasts with the actual values.

Figure 18: ARIMA(1,1,1) Forecasts and real values for CHE?USD daily exchange rates.
4.4.5 ARIMA Forecasting Model for CAD/USD Daily Rates

The forecasting model with the minimum AIC value for the CAD/USD daily rates was a combination of first order autoregressive (AR), and a third order moving average (MA) with the first difference filter, namely, ARIMA(1,1,3).

\[(1 - 0.481798L)(1 - L)x_t = 0.000055 + (1 - 0.182836L - 0.130882L^2 + 0.067018L^3)\epsilon_t\] (4.26)

Expanding the autoregressive operator and the difference filter and then letting \(\epsilon_t = 0\), the following one day ahead forecasting model was obtained:

\[x_t = 0.000055 + 1.481798x_{t-1} - 0.481798x_{t-2} - 0.182836\epsilon_{t-1} - 0.130882\epsilon_{t-2} + 0.067018\epsilon_{t-2}\] (4.27)

The figure 19 shows the forecasts with the actual values.

Figure 19: ARIMA(1,1,3) Forecasts and real values for CAD/USD daily exchange rates.
4.5 \textit{k}-Nearest Neighbor Forecasting vs. ARIMA Forecasting - Forecasting Accuracy

In section 4.4, we discussed in details the general autoregressive integrated moving average forecasting models for the five daily exchange rates data sets. The given figures (fig.15-19) indicate that the ARIMA forecasts follow the actual values pretty well as similar to the case of \textit{k}-NN forecasting. In this section, our goal is to compare the forecasting accuracy of ARIMA models with our proposed Mahalanobis distance based \textit{k}-nearest neighbor forecasting procedure.

We considered all the accuracy measures mentioned in section 3.3 and compared the performances of each algorithm based on how accurate their forecasts were. The following tables give the $U$-statistic, mean square error, and normalized root mean square error for the currencies EUR, GBP, JPY, CHF, and CAD with \textit{k}-NN algorithm (with Mahalanobis distance) and ARIMA forecasting models.

\begin{table}[h]
\centering
\caption{$U$-statistics with \textit{k}-NN and ARIMA models}
\begin{tabular}{|l|c|c|}
\hline
Currency & \textit{k}-NN forecasting & ARIMA forecasting \\
\hline
EUR & 0.003898012 & 0.003456137 \\
GBP & 0.003454303 & 0.00318494 \\
JPY & 0.00690517 & 0.008302838 \\
CHF & 0.005559151 & 0.007286362 \\
CAD & 0.005979865 & 0.00731294 \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\caption{Mean square error with \textit{k}-NN and ARIMA models}
\begin{tabular}{|l|c|c|}
\hline
Currency & \textit{k}-NN forecasting & ARIMA forecasting \\
\hline
EUR & 0.00010905 & 0.00008479 \\
GBP & 0.00011439 & 0.00009725 \\
JPY & 0.00024810 & 0.00035878 \\
CHF & 0.00011441 & 0.000196566 \\
CAD & 0.00013689 & 0.000199935 \\
\hline
\end{tabular}
\end{table}
### Table 16: Normalized root mean square error with $k$-NN and ARIMA models

<table>
<thead>
<tr>
<th>Currency</th>
<th>$k$-NN forecasting</th>
<th>ARIMA forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.16748184</td>
<td>0.14691536</td>
</tr>
<tr>
<td>GBP</td>
<td>0.22251281</td>
<td>0.20423380</td>
</tr>
<tr>
<td>JPY</td>
<td>0.30304086</td>
<td>0.35985560</td>
</tr>
<tr>
<td>CHF</td>
<td>0.21693213</td>
<td>0.28301529</td>
</tr>
<tr>
<td>CAD</td>
<td>0.58311581</td>
<td>0.65600438</td>
</tr>
</tbody>
</table>

As can be seen from the obtained results, given by tables 4.5, 4.5, & 4.5, we can see that majority of the time (3 out of 5) proposed Mahalanobis based $k$-NN forecasting model out performs the ARIMA method. In the cases of EUR and GBP, the general ARIMA process seems to forecast relatively better compare to the nearest neighbor forecasting. Even though our primary goal in this chapter is to compare the trading performances of both methods, we wanted to further analyze the data, and come up with an explanation behind this situation. For this purpose, we calculated the following statistical measures for all the data sets:

- **Total Variation** -
  The total variation or the total sum of squares (SST) is a measure of the observed values around the mean. It is comprised the sum of the squares of the differences of each data value with the mean.

  \[
  Total\ variation = \sum_{t=1}^{n} (x_t - \bar{x})^2
  \]  
  (4.28)

- **Standard Deviation** -
  In statistics, the standard deviation is a measure of the spread of scores within a set of data. It is a measure that is used to quantify the amount of variation or dispersion of a set of data values. Smaller the standard deviation, closer the data points to its mean.

  \[
  Standard\ deviation, \sigma = \sqrt{\frac{\sum_{t=1}^{n} (x_t - \bar{x})^2}{n}}
  \]  
  (4.29)
Table 17: Total Variation and Standard Deviation

<table>
<thead>
<tr>
<th>Currency</th>
<th>Total variation</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>10.99275618</td>
<td>0.104846346</td>
</tr>
<tr>
<td>GBP</td>
<td>36.55015706</td>
<td>0.191180954</td>
</tr>
<tr>
<td>JPY</td>
<td>0.000877177</td>
<td>0.000936577</td>
</tr>
<tr>
<td>CHF</td>
<td>4.313622939</td>
<td>0.065678177</td>
</tr>
<tr>
<td>CAD</td>
<td>4.193662329</td>
<td>0.064758492</td>
</tr>
</tbody>
</table>

Considering the calculated values for total variation and standard deviation (given in table 4.5), we observed that the EUR and GBP daily rates have relatively higher total variation and standard deviation compare to the remaining data sets. This means that those data are more spread out comparing to the other data, and hence more volatile. Thus, one of the conclusions we can come up with considering forecasting accuracy is that for FX rates with a relatively higher total variation (or standard deviation), ARIMA($p$,$d$,$q$) model performs relatively better compare to $k$-NN procedure.

One of the reasons behind ARIMA process forecasts better for a time series with a higher volatility is having the moving average (MA) component to adjust according to the previous forecasting errors, unlike the nearest neighbor forecasting, which is something important specially, when the data are more volatile. The novelty of the $k$-nearest neighbor forecasting is to capture the similar history and forecasts better only using the most relevant instance. As we discussed in chapter 3, choosing the correct distance function can make a huge impact on it’s performances. Even though Mahalanobis distance based $k$-nearest neighbor method outperforms the popular time ARIMA process majority of the time (3 out of 5), these results indicate that we can further improve the $k$-NN forecasting method by adjusting the algorithm according to the previous forecasting errors.

As the next step, of the comparison procedure, we converted ARIMA forecasts into trading signals, buy and sell, using the technical trading strategy discussed in section 3.5.2, and compared the results with Mahalanobis distance based nearest neighbor trading decisions.
4.6  \textit{k}-Nearest Neighbor Forecasting vs. ARIMA Forecasting - Comparing Trading Decisions

As it is obvious that currency trader’s main goal is to make more money, in this section we evaluated these two prediction models (\textit{k}-NN and ARIMA) considering their trading performances. We transformed the ARIMA forecasts into trading signals, \textit{buy} and \textit{sell} using technical trading strategy discussed in chapter 3. Then, the performance measures, \textit{total (log access) return} and \textit{Sharpe ratio} (section 3.5.2) were calculated and compared with those of \textit{k}-nearest neighbor forecasting technique. Higher values of these measures indicate that the model is performing better.

The estimated total return and Sharpe ratio for the technical trading strategy under \textit{k}-NN algorithm and ARIMA process are given in tables 4.6 and 4.6. The final conclusion of forecasting model is pretty much same as that of error measures. Proposed Mahalanobis distance based \textit{k}-NN method outperforms the ARIMA process majority of the time. According to the forecasting accuracy, both EUR and GBP daily exchange rates data sets support ARIMA model. However, when comparing total return and Sharpe ratio, GBP/USD daily rates pretty much gave the same numerical values for both the models. Therefore, the results for trading decisions also indicated that the \textit{k}-nearest neighbor forecasting model producing more accurate and profitable trading signals compared to the general ARIMA process.

\begin{table}[h]
\centering
\caption{Total Return \textit{k}-NN and ARIMA models}
\begin{tabular}{|l|l|l|}
\hline
Currency & \textit{k}-NN forecasting & ARIMA forecasting \\
\hline
EUR & 0.52991777 & 0.89316418 \\
GBP & 4.16807227 & 4.16807227 \\
JPY & 0.67755404 & 0.47532224 \\
CHF & 5.42108879 & 5.16742874 \\
CAD & 4.38589604 & 4.03711714 \\
\hline
\end{tabular}
\end{table}

The results from section 4.5 and section 4.6 motivated us to investigate more on the behavior of time series data and the most appropriate forecasting technique. The primary goal of the next section is to study the forecasting accuracy of simulated time series data with both Mahalanobis distance based \textit{k}-NN method and the general ARIMA forecasting models.
Table 19: Sharpe Ratio with $k$-NN and ARIMA models

<table>
<thead>
<tr>
<th>Currency</th>
<th>$k$-NN forecasting</th>
<th>ARIMA forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.27890809</td>
<td>0.50803011</td>
</tr>
<tr>
<td>GBP</td>
<td>2.41593434</td>
<td>2.41593434</td>
</tr>
<tr>
<td>JPY</td>
<td>0.18429771</td>
<td>0.12818451</td>
</tr>
<tr>
<td>CHF</td>
<td>1.67419376</td>
<td>1.42328537</td>
</tr>
<tr>
<td>CAD</td>
<td>1.26400087</td>
<td>1.04713284</td>
</tr>
</tbody>
</table>

4.7 Simulation Data Analysis

Time series data simulation plays an important role in many areas of time series data analysis such as economics & finance, environmental studies, and engineering [31, 4]. It is a whole different area of research, where the researchers have paid much more attention in the recent history. Generating financial time series such as exchange rates data is a challenging task compared to most of the other time series data simulation[62]. A huge amount of empirical contributions been made towards this topic, and variety of economical, financial and time series models been proposed and experimented by many academic and industrial researchers during the last two decades. As most of the traditional financial theory based methods failed to match the features displayed by the actual data, many alternative models were proposed to overcome the issues of these traditional theory based models [8, 62].

The purpose of any foreign currency generating algorithm is to replicate a certain exchange rate by considering all the financial and economical factors related to those two countries, which is a complicated task. In their work Bianchi, Pantanella, and Pianese claimed that using their proposed multifractional process with random exponent, they were successfully able to replicate EUR/JPY and EUR/USD [8] exchange rates. Also, Oyediran & Afierho have worked on developing an algorithm to simulate many different FX rates such as European euro, British pound sterling and the US dollar against the Nigeria naira [62]. Their simulation models were also developed after analyzing the historical data of the corresponding currency rates.

All these simulation algorithms have one main goal in common. Their goal was to develop a procedure well capture the behavior of a given currency rate, which was not our intension of simulation study in this work. The goal here is to capture the behavior(s) of a time series to decide which
forecasting algorithm ($k$-NN or ARIMA) would be more beneficial. Even though our primary interest is forecasting and decision making in foreign exchange market, for the simulation study we considered time series data in general.

Auto regressive (AR), moving average (MA), and general and mix ARIMA models are the most popular time series data simulation techniques among the time series research community [12, 27]. These time series processes have been used by many researchers over the recent history to replicate time series data using different computer software such as MATLAB and R [30, 46]. For the simulation data analysis, several time series data sets were simulated in MATLAB environment with the use of the built-in MATLAB functions “arima” and “simulate”. Since the data were simulated using ARIMA process, there is always a possibility of having an advantage of using an ARIMA forecasting model.

The observations from section 4.5 and section 4.6 lead to the conclusion that for a time series data with a higher volatility, ARIMA forecasting procedure works better compared to the $k$-nearest neighbor method. As can be seen from table the 4.5, both EUR and GBP data sets have higher volatility measures compared to the rest. Due to this reason, the time series were simulated by varying the standard deviation. We have chosen a range from 0.00126 to 0.896 to capture the range of our data sets’ standard deviations. The simulated 9 data sets and their standard deviations listed below from highest standard deviation to lowest standard deviation:

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulated data set 1</td>
<td>0.896010158</td>
</tr>
<tr>
<td>Simulated data set 2</td>
<td>0.400925413</td>
</tr>
<tr>
<td>Simulated data set 3</td>
<td>0.283497079</td>
</tr>
<tr>
<td>Simulated data set 4</td>
<td>0.126783748</td>
</tr>
<tr>
<td>Simulated data set 5</td>
<td>0.040092541</td>
</tr>
<tr>
<td>Simulated data set 6</td>
<td>0.012678375</td>
</tr>
<tr>
<td>Simulated data set 7</td>
<td>0.004009254</td>
</tr>
<tr>
<td>Simulated data set 8</td>
<td>0.001267838</td>
</tr>
<tr>
<td>Simulated data set 9</td>
<td>0.001267838</td>
</tr>
</tbody>
</table>

The model comparison was performed using the accuracy measures discussed in section 3.3.
We only focused on deviation in fit for this comparison. To compare the trading decisions, it is necessary to simulate the interest rates, and also the time series data replicating real FX data of a certain country, which is not our interest here. Also the obtained results in chapter 3 and 4 suggest that having more accurate forecasts always lead to a higher trading performances.

We followed the same data preparation procedure discussed in section 4.3 to build the best model for each data set when using ARIMA process for forecasting. Even though the data was simulated with the specified orders and parameters, we again tested them for the appropriate differencing order and AR order, $p$, and MA order, $q$. For Mahalanobis distance based $k$ nearest neighbor algorithm the parameter $m$ was set to be 3 and $k$ was set to be 20 as in section 4.5. One step ahead out of sample forecasts were created for 250 test set and the size of the training window was 1000.

<table>
<thead>
<tr>
<th>Data Set $D_i$</th>
<th>Standard deviation</th>
<th>$k$-NN $U$-statistic</th>
<th>ARIMA $U$-statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>0.896010158</td>
<td>0.18665837</td>
<td>0.14858764</td>
</tr>
<tr>
<td>$D_2$</td>
<td>0.400925413</td>
<td>0.08280900</td>
<td>0.07039754</td>
</tr>
<tr>
<td>$D_3$</td>
<td>0.283497079</td>
<td>0.05881467</td>
<td>0.05116939</td>
</tr>
<tr>
<td>$D_4$</td>
<td>0.126783748</td>
<td>0.02643441</td>
<td>0.02684713</td>
</tr>
<tr>
<td>$D_5$</td>
<td>0.040092541</td>
<td>0.00837700</td>
<td>0.01733388</td>
</tr>
<tr>
<td>$D_6$</td>
<td>0.012678375</td>
<td>0.00265053</td>
<td>0.01621586</td>
</tr>
<tr>
<td>$D_7$</td>
<td>0.004009254</td>
<td>0.00083831</td>
<td>0.01614590</td>
</tr>
<tr>
<td>$D_8$</td>
<td>0.001267838</td>
<td>0.00026511</td>
<td>0.01615349</td>
</tr>
<tr>
<td>$D_9$</td>
<td>0.001267838</td>
<td>0.00185742</td>
<td>0.01614677</td>
</tr>
</tbody>
</table>

The comparison results of $U$-statistic for the simulated data are presented in Table 4.7 above. It can be clearly seen that for the data sets 1, 2, and 3, ARIMA based forecasting models had lower $U$-statistic values compared to those of Mahalanobis distance based $k$-NN forecasting. Those are the data sets with higher standard deviations. When the standard deviation is getting smaller and smaller, $k$-NN forecasting algorithm started to perform comparatively better than general ARIMA process. For the data sets 6 trough 9, the difference between the $U$-statistic values are significant. This supports the claim that for a time series data with a lower standard deviation, the $k$-NN method has a higher forecasting accuracy compared to ARIMA.
All the other error measures support the same argument. As can be seen from the table 4.7, when comparing mean square error (MSE) also we were able to observe that for the data sets 1, 2, and 3, k-NN forecasting method demonstrating relatively poor performance compared with ARIMA. Even these data were simulated using general ARIMA process, for the time series data with a lower volatility, k-nearest neighbor forecasting method (with Mahalanobis distance) outperforms the ARIMA method.

Table 22: MSE for k-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.00127 and 0.896

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Standard deviation</th>
<th>k-NN MSE</th>
<th>ARIMA MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D_1$</td>
<td>0.896010158</td>
<td>0.75482502</td>
<td>0.620606842</td>
</tr>
<tr>
<td>$D_2$</td>
<td>0.400925413</td>
<td>0.15148122</td>
<td>0.127060368</td>
</tr>
<tr>
<td>$D_3$</td>
<td>0.283497079</td>
<td>0.07574061</td>
<td>0.066274057</td>
</tr>
<tr>
<td>$D_4$</td>
<td>0.126783748</td>
<td>0.01514812</td>
<td>0.018027316</td>
</tr>
<tr>
<td>$D_5$</td>
<td>0.040092541</td>
<td>0.00151481</td>
<td>0.007485055</td>
</tr>
<tr>
<td>$D_6$</td>
<td>0.012678375</td>
<td>0.00015148</td>
<td>0.006544971</td>
</tr>
<tr>
<td>$D_7$</td>
<td>0.004009254</td>
<td>0.00001515</td>
<td>0.006487096</td>
</tr>
<tr>
<td>$D_8$</td>
<td>0.001267838</td>
<td>0.00000151</td>
<td>0.006492738</td>
</tr>
<tr>
<td>$D_9$</td>
<td>0.001267838</td>
<td>0.00000151</td>
<td>0.000405499</td>
</tr>
</tbody>
</table>

We went further and tried to figure out exactly around what value of standard deviation k-NN procedure starting to work better. Table 4.7 and 4.7 clearly indicate that somewhere between the values of 0.127 & 0.283, k-NN forecasting procedure has started performing better. To investigate this further, couple of more data sets were simulated with standard deviation between 0.009 and 0.15. Then, we followed the exact same procedure and predicted 250 future values. From the given results of $U$-statistic values in table 4.7, we can observe that for the standard deviation values below 0.13, the k-NN has a better forecasting accuracy.

4.8 Concluding Remarks

In this chapter, our main goal was to compare the proposed k-NN forecasting with general autoregressive integrated moving average (ARIMA) process, which is assumed to be one of the best time
Table 23: \( U \)-statistic for \( k \)-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.009 and 0.15

<table>
<thead>
<tr>
<th>Simulated data ( sd )</th>
<th>( k )-NN ( U )-statistic</th>
<th>ARIMA ( U )-statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.15010000</td>
<td>0.031257268</td>
<td>0.030175564</td>
</tr>
<tr>
<td>0.13000000</td>
<td>0.02708489</td>
<td>0.02728483</td>
</tr>
<tr>
<td>0.12750000</td>
<td>0.026565797</td>
<td>0.026934802</td>
</tr>
<tr>
<td>0.12030000</td>
<td>0.0250823020</td>
<td>0.025948689</td>
</tr>
<tr>
<td>0.10540000</td>
<td>0.021970607</td>
<td>0.023958385</td>
</tr>
<tr>
<td>0.10030000</td>
<td>0.020912901</td>
<td>0.02330971</td>
</tr>
<tr>
<td>0.09100000</td>
<td>0.018988001</td>
<td>0.022171638</td>
</tr>
</tbody>
</table>

series forecasting technique. As all these forecasting methods are data driven models, giving an optimal forecasting model works with all types of data is practically a difficult task.

From our results, we can conclude that \( k \)-nearest neighbor forecasting algorithm with Mahalanobis distance function outperforms the popular time series forecasting technique, general ARIMA process, majority of the time. For the data sets with a relatively higher total variation (or highly volatile), ARIMA methods seems to work better compared to the \( k \)-NN forecasting. Our simulation data study supported this claim as well. Considering the accuracy measures (\( U \)-statistic and MSE), we can conclude that for time series data with a smaller standard deviation, \( k \)-NN forecasting procedure more appropriate than the ARIMA process.

The nearest neighbor algorithm is a nonparametric, on-line learning algorithm. Thus, it does not require any distributional assumptions, and data preparation ahead of time. Unlike nearest neighbor, ARIMA process requires model building procedure to select proper differencing order (\( d \)), autoregressive order (\( p \)), and moving average order (\( q \)). The obtained results proved that even with all these model building procedure, still the ARIMA process worked better only for one currency data set according to the trading decisions. We discussed in the previous chapter (chapter 3) that choosing an appropriate distance in NN algorithm can improve the forecasting significantly. The results obtained in this chapter further support our earlier conclusion. Also, we noticed that the \( k \)-NN forecasting method can be further improve by adjusting the algorithm according to the previous forecasting errors, which will be part of our future work.

70
Chapter 5
Multi-step Ahead Forecasting of FX Rates with k-Nearest Neighbor Algorithm

5.1 Multi-step Ahead Time Series Forecasting.

Multi-stepahead forecasting is a process of predicting a sequence of values in a time series [43]. Even though the majority of the available forecasting models mainly focus on predicting a value for a one-step-ahead time interval, it is highly important to have more than one forecast in many areas of time series forecasting. As an obvious example, we prefer to know the weather forecast for at least a week ahead rather than for a single day. This may help us to figure out the highest and lowest temperature, chances of rain, snow, etc. and plan our regular day-to-day activities accordingly.

Accurate prediction over a long period of time is still a challenging task in financial time series forecasting as in any other area of time series. However, during the past decade, there has been a growing trend of multi-step-ahead forecasting in financial and economical data. In their paper, Xiong, Bao and Hu pointed out the importance of the interval-valued stock price index forecasting for a more profitable stock trading. They claimed that having a larger period of prediction values is beneficial for creating verity of technical trading rules [92].

In their work, McElory and McCracken [60] applied the multi-step ahead forecasting of vector times series for euro area real gross domestic product data and inflation rates data. Even though their main focus was pretesting the data before choosing a forecasting model, they had the multi-stage prediction for the real FX data with several models. They also mentioned the computational difficulty of using non-linear forecasting models to make multi-step-ahead forecasts. They used the direct and indirect multi-step-ahead forecasting techniques with verity of models, such as LSTAR (logistic smooth transition auto regression) and ESTAR (Exponential smooth transition auto regression) [60]. They also stated that different well-trained researchers will often come up with very different multi-step-ahead forecasts using the same data [60, 87].
Let \((x_1, x_2, \ldots, x_t)\) be the available time series data. In multi-step-ahead prediction, our task is to forecast the next \(h\) values given by \((x_{t+1}, x_{t+2}, \ldots, x_{t+h})\). In the literature of time series analysis, the following strategies are the commonly used methods for multi-step ahead forecasting.

### 5.1.1 Recursive Strategy

The recursive Strategy is the most common \(h\)-step ahead prediction method that has been used in the literature. This is also called the iterative or multi-stage strategy [18, 79]. The process is very intuitive. Let \(\hat{x}_{t+1}\) be the forecasted value for \(x_{t+1}\). Then

\[
\hat{x}_{t+1} = \hat{f}(x_t, x_{t-1}, \ldots, x_{t-p+1})
\]  

(5.1)

where \(x_t, x_{t-1}, \ldots, x_{t-p+1}\) are the previous \(p\) values used to predict \(x_{t+1}\) and \(\hat{f}_1\) represents the one-step-ahead forecasting model learned from the available data to predict \(x_{t+1}\). Once we have the forecast for \(x_{t+1}\), we use it as a part of input variables forecasting the next time step. Therefore, the forecasted value for \(x_{t+2}\) can be written as follows:

\[
\hat{x}_{t+2} = \hat{f}(\hat{x}_{t+1}, x_t, x_{t-1}, \ldots, x_{t-p+2})
\]  

(5.2)

We continue the same process until we have the prediction values for the desired \(h\)-steps. It is clear that the recursive strategy is highly sensitive to forecasting errors. When the forecasting horizon \(h\) exceeds \(p\), all the inputs become estimated values. Thus, it can be seen clearly that the recursive strategy suffers from low performance due to the error accumulation [18, 79].

### 5.1.2 Direct Strategy

The direct strategy [18, 75, 79] predicts the future \(h\) values of a time series using separate (independent) \(h\) models. This method is also called the independent value prediction [18]. In this case, we learn \(h\) independent models from the data without using any forecasted values [18, 79]. Thus, the \(i^{th}\) forecasted value can be written as

\[
\hat{x}_{t+i} = \hat{f}_i(x_t, x_{t-1}, \ldots, x_{t-d_i+1}); \quad i = 1, 2, \ldots, h
\]  

(5.3)
where \( \hat{f}_i \) represents the \( i^{th} \) forecasting model learned from the data. This strategy does not suffer from error accumulation like recursive strategy as we do not use any forecasted value as an input for the model. However, the issue here is that, as the \( h \) models are learned independently, the complex dependencies between the variables \( \hat{x}_{t+i} \) are ignored [78, 79]. This might affect the forecast accuracy and hence might lead to a higher forecasting error. The other problem is the larger computational time, even though it is not a major problem with the modern high power computers.

5.1.3 DirRec Strategy

The DirRec strategy can be considered as a combination of both direct and recursive strategy [74, 75]. Similar to the direct strategy, we use \( h \) different models to forecast the next \( h \) values. The difference here is that after each forecast, the predicted value is fed back to the model as an input before the next forecast.

\[
\hat{x}_{t+1} = \hat{f}_1(x_t, x_{t-1}, \ldots, x_{t-p_1+1})
\]

\[
\hat{x}_{t+2} = \hat{f}_2(\hat{x}_{t+1}, x_t, x_{t-1}, \ldots, x_{t-p_2+1})
\]

\[
\vdots
\]

\[
\hat{x}_{t+h} = \hat{f}_2(\hat{x}_{t+1}, \hat{x}_{t+2}, \ldots, \hat{x}_{t+h-1}, x_t, x_{t-1}, \ldots, x_{t-p_h+1})
\]

In their work, Sorjamaa and Lendasse applied DirRec strategy for long-term time series forecasting for two real data sets and proved it works better compared to recursive and direct strategy [74]. The main issue with this method is the computational time. However, as their final conclusion, they suggested to use the direct strategy if the computational time is an issue. But for high accuracy, DirRec strategy is a better choice than both the recursive and the direct strategies [74].

5.1.4 Multi-Input-Multi-Output (MIMO) Strategy

All the above multi step-ahead forecasting techniques are multiple input-single output methods. Thus, in order to estimate \( h \) future time series values, we need to continue the process \( h \) number of times. Using MIMO strategy, on the other hand, the \( h \) forecasts are obtained using a single model.
as a vector output [79, 10].

\[
(\hat{x}_{t+1}, \hat{x}_{t+2}, \ldots, \hat{x}_{t+h-1}) = \hat{f}(x_t, x_{t-1}, \ldots, x_{t-p+1}) \tag{5.7}
\]

Here \( \hat{f} : \mathbb{R}^p \rightarrow \mathbb{R}^h \) is a vector valued function. Recently, this method has been applied in many areas of time series data analysis [78, 10]. The main drawback of this technique is that it uses a single model to predict the entire horizon. This can lead to a lack of flexibility in the forecasting [78, 79].

### 5.1.5 DIRMO Strategy

The DIRMO strategy is built by preserving the important aspects of both direct and MIMO strategies. The term \( DIRMO \) is made from the first 3 letters of \( DIRECT \) and the last 2 letters of \( MIMO \). Here, the \( h \) predictions are done as \( n \) blocks, where \( n = h/s \) and \( s \) is the number of output from each block. Therefore, this method is also a multi-input-multi-output method that does not use the same model to forecast desired future values. Instead, it is made of \( n \) forecasting models, which can be given as [79]:

\[
(\hat{x}_{t+1}, \hat{x}_{t+2}, \ldots, \hat{x}_{t+s}) = \hat{f}_1(x_t, x_{t-1}, \ldots, x_{t-p_1+1}) \tag{5.8}
\]

\[
(\hat{x}_{t+s+1}, \hat{x}_{t+s+2}, \ldots, \hat{x}_{t+s}) = \hat{f}_2(x_t, x_{t-1}, \ldots, x_{t-p_2+1}) \tag{5.9}
\]

\[
\ldots
\]

\[
(\hat{x}_{t+ns-s}, \hat{x}_{t+ns-s+1}, \ldots, \hat{x}_{t+ns}) = \hat{f}_n(x_t, x_{t-1}, \ldots, x_{t-p_n+1}) \tag{5.10}
\]

One of the important facts about the DIRMO strategy is that it improves the flexibility of the forecasts. Researchers in the time series analysis and machine learning communities have successfully applied this strategy for multi-step ahead forecasts in their recent work [78, 79].

Even though this strategy seems to give the most promising results, it also has its own pros and cons. The DIRMO technique of multi-step ahead forecasting combines all the positive facts of direct, MIMO and DIREC strategies. However, this increases the number of parameters needed to be estimated. To use this method, we need to determine both \( n \) and \( s \). The other negative fact about
this technique is that it completely ignores the previous forecasts when predicting the next block of predictions.

5.2 $k$-Nearest Neighbor Algorithm and Multi-step Ahead Forecasting.

Many researchers claim that the $k$-NN algorithms outperform the available linear forecasting methods for financial time series data. Most of the available work in the literature supports this claim with one time step ahead forecasting in many areas of finance, including foreign exchange rates. Researchers have adopted $k$-nearest neighbors and lazy learning algorithms as forecasting models to compare the performances of multi-step-ahead strategies discussed in section ??.

Our goal in this work is not to compare the performances of the different multi-step-ahead forecasting strategies. We want to improve the performance of the $k$-NN forecasting method for currency data by adopting the idea of multi-step-ahead forecasting. In chapter 3, we already introduced Mahalanobis distance as the choice of distance instead of the traditional Euclidean distance and provided various forecasting accuracy measures and trading performance measures to support our claim. Also, we conducted a thorough empirical investigation of parameter selection of the algorithm for the FX data. As the next stage of our work, we are interested in looking into multi-step ahead forecasting in foreign exchange data using the $k$-NN algorithm and comparing the performances for different distance functions.

As our primary goal of this work is to improve the $k$-nearest neighbor forecasting algorithm for the FX market to make better trading signals and maximize the final outcome, we compare the trading performances of the single step ahead forecasting model with the multi-step ahead forecasting algorithm with proposed Mahalanobis distance and with traditional distances, such as Euclidean.

In the next section we will discuss the adopted multi-step ahead forecasting strategy for the $k$-NN forecasting algorithm. The method is applied to the daily foreign currency rates data discussed in 3.4.1. Here we have taken the value of $h$ to be 5, which means we forecast the next 5 days exchange rates in order to make the trading decisions today.

As recursive strategy is the most intuitive and commonly used multi-step ahead prediction technique, we will be comparing the result of the proposed technique and the traditional recursive strategy as well. Also, we will introduce a new version of the technical trading rule, which was discussed
in section 3.5.2. As we compare the performances with our earlier results obtained in chapter 3, we use the same embedding dimension, \( m = 3 \) and the number of nearest neighbors, \( k = 20 \).

### 5.2.1 \( k \)-Nearest Neighbor Forecasting with Recursive (multi-stage) Strategy.

The forecasted value for next time step (\( \hat{x}_{T+1} \)) is obtained using the local linear regression model discussed in section 5.1.1. To forecast \( x_{T+2} \), we used the same regression model and fed back \( \hat{x}_{T+1} \) as an input to the model. We continued this process until all the \( h \) forecasts were obtained. The local regression model for the \( i^{th} \) forecasting can be given as:

\[
\hat{x}_{T+i} = \begin{cases} 
\hat{a}_0 x_T + \hat{a}_1 x_{T-1} + \ldots + \hat{a}_{m-1} x_{T-(m-1)} + \hat{a}_m & \text{if } i = 1 \\
\hat{a}_0 \hat{x}_{T+1} + \hat{a}_1 x_T + \ldots + \hat{a}_{m-1} x_{T+i-m} + \hat{a}_m & \text{if } i \leq m \\
\hat{a}_0 \hat{x}_{T+i-1} + \hat{a}_1 \hat{x}_{T+i-2} + \ldots + \hat{a}_{m-1} \hat{x}_{T+i-m} + \hat{a}_m & \text{if } i > m
\end{cases}
\] (5.11)

Note that when \( i > m \), all the \( x_t \)'s become estimated values.

### 5.2.2 \( k \)-Nearest Neighbor Forecasting with Independent Value Prediction Method

The multi-step-ahead forecasting strategy we adopted in this work is a combination of the idea of independent value prediction [18] and the DirRec strategy. Even though in the literature the DIRMO technique seems to work better compared to DirRec strategy [79], as we only consider 5 days ahead forecasts (\( h=5 \)), we do not want to go for additional parameter estimations. Also, since the forecasting horizon, \( h \), is relatively very small it is not that important to divide into \( n \) blocks as well.

The \( DIRMO \) multi-step ahead forecasting method is more relevant when apply for a relatively longer period of time. As an example, Souhaib Ben Taieb et al. compared the performances of multi-step ahead forecasting strategies on the time series data from the NN5 data set, and they performed the forecasting of 56 days ahead values [79]. When the forecasting horizon is large, using DirRec strategy might have disadvantages, such as longer computational time and the negative effect of not considering the complex dependency between the variable, as mentioned in section 5.1.3. However, in our case, since \( h = 5 \), the above two issues do not play a big role in the algorithm.
Let \( \{ x_t \}_{t=1}^N = \{ x_1, x_2, \ldots, x_N \} \) be the time series data. Consider the training set \( \{ x_1, x_2, \ldots, x_T \} \), and the test the set, \( \{ x_{T+1}, x_{T+2}, \ldots, x_N \} \) as discussed in chapter 3. Then the delay vector at time \( t = T \) is given by

\[
x^m_T = (x_T, x_{T-1}, \ldots, x_{T-(m-1)})
\]

(5.12)

To avoid any confusion with the time index, we will take its \( k \) nearest neighboring vectors as

\[
x^m_{Tj} = (x_{Tj}, x_{Tj-1}, \ldots, x_{Tj-(m-1)}) \quad \text{for} \quad j = 1, 2, \ldots, k
\]

(5.13)

Then the \( k \times m \) matrix containing the \( k \) neighboring vectors is given by

\[
A_T = \begin{pmatrix}
x_{T1} & x_{T1-1} & \cdots & x_{T1-(m-1)} \\
x_{T2} & x_{T2-1} & \cdots & x_{T2-(m-1)} \\
\vdots & \vdots & \ddots & \vdots \\
x_{Tk} & x_{Tk-1} & \cdots & x_{Tk-(m-1)}
\end{pmatrix}
\]

(5.14)

The one step ahead forecasted value, \( \hat{x}_{T+1} \) is obtained with the local linear regression model using the above data as input to estimate the regression coefficients as discussed in chapter 3. Then we used the idea of the sliding window algorithm [18] to obtain the \( k \times m \) nearest neighbor matrix to forecast the values for the next \( h - 1 \) values as below;

\[
A_{T(i+1)} = \begin{pmatrix}
x_{T1+i} & x_{T1-1+i} & \cdots & x_{T1-(m-1)+i} \\
x_{T2+i} & x_{T2-1+i} & \cdots & x_{T2-(m-1)+i} \\
\vdots & \vdots & \ddots & \vdots \\
x_{Tk+i} & x_{Tk-1+i} & \cdots & x_{Tk-(m-1)+i}
\end{pmatrix}
\]

(5.15)

where \( i = 1, 2, \ldots, h - 1 \). As our \( k \)-NN forecasting method is an on-line forecasting technique, at each time step we will have \( h \) regression models for the next \( h \)-step ahead forecasting. We fed back \( \hat{x}_{T+i} \) as an input value to the forecasting model similar to the DirRec strategy.
5.2.3 Forecasting Accuracy of Multi-step Ahead Forecasting

To calculate accuracy measures, we used the same error measures discussed in section 3.3. The prediction error \( e_t \), at time, \( t \), was taken as the average of the \( h \) forecasting errors.

\[
e_t = \frac{1}{h} \left[ (\hat{x}_{t+1} - x_{t+1}) + (\hat{x}_{t+2} - x_{t+2}) + \ldots + (\hat{x}_{t+h} - x_{t+h}) \right] \quad (5.16)
\]

The following formulas define the modified versions of the error measures we consider in the error calculations in the multi-step ahead forecasting model.

Mean square error (MSE):

\[
MSE = \frac{1}{n} \sum_{t=1}^{n} (e_t)^2 \quad (5.17)
\]

Mean absolute percentage error (MAPE):

\[
MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{e_t}{y_t} \right| \quad (5.18)
\]

Theil’s \( U \)-statistic (\( U \)):

\[
U = \frac{\sqrt{\sum_{t=1}^{n} (e_t)^2}}{\sqrt{\sum_{t=1}^{n} (\hat{y}_t)^2} + \sqrt{\sum_{t=1}^{n} (y_t)^2}} \quad (5.19)
\]

Normalized Root Mean Square Error (NRMSE):

\[
NRMSE = \frac{\sqrt{\sum_{t=1}^{n} e^2(t)}}{\sigma} \quad (5.20)
\]

In all the above formulas, \( \hat{y}_t = \frac{1}{h} [\hat{x}_{t+1} + \hat{x}_{t+2} + \ldots + \hat{x}_{t+h}] \) and \( y_t = \frac{1}{h} [x_{t+1} + x_{t+2} + \ldots + x_{t+h}] \).

First, we compare the accuracy measures mentioned above for \( k \)-nearest neighbor forecasting with Mahalanobis, Euclidean, and Absolute distances. We consider both recursive and independent value prediction strategies separately. The following tables give the summarized results for the currencies EUR, GBP, JPY, CHF, and CAD with the choice of \( m = 3, k = 20, \) and \( h = 5 \).

In table 5.2.3, we report the forecasting accuracy for Mahalanobis, Euclidean and absolute dis-
Table 24: $U$-Statistics with Independent Value Prediction

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.00548527</td>
<td>0.00554013</td>
<td>0.00551991</td>
</tr>
<tr>
<td>GBP</td>
<td>0.00466984</td>
<td>0.00471515</td>
<td>0.00474155</td>
</tr>
<tr>
<td>JPY</td>
<td>0.00726693</td>
<td>0.00869652</td>
<td>0.01080483</td>
</tr>
<tr>
<td>CHF</td>
<td>0.00710084</td>
<td>0.00748465</td>
<td>0.00759857</td>
</tr>
<tr>
<td>CAD</td>
<td>0.00657776</td>
<td>0.00774412</td>
<td>0.00785634</td>
</tr>
</tbody>
</table>

Distance using $U$-statistic. It is clear that, Mahalanobis distance performs better compared to other distance measures. The normalized root mean square error given in the following table also supports our claim. The observations were similar when comparing the other error measures as well.

Table 25: Normalized Root Mean Square Error with Independent Value Prediction

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.23438668</td>
<td>0.23668555</td>
<td>0.23580829</td>
</tr>
<tr>
<td>GBP</td>
<td>0.30086595</td>
<td>0.30375289</td>
<td>0.30545765</td>
</tr>
<tr>
<td>JPY</td>
<td>0.31898631</td>
<td>0.38175430</td>
<td>0.47410321</td>
</tr>
<tr>
<td>CHF</td>
<td>0.27712096</td>
<td>0.29207254</td>
<td>0.29652012</td>
</tr>
<tr>
<td>CAD</td>
<td>0.63396536</td>
<td>0.74625605</td>
<td>0.75696623</td>
</tr>
</tbody>
</table>

From the obtained results, it is clear that the Mahalanobis distance performs better compare to other distance measures for all the currencies, even though the values were not significantly different. We also compare the measures of forecasting accuracy of the recursive strategy and the independent value prediction method.

Our primary goal here was not the error comparison. We wanted to focus on the decision making aspect of the multi-step ahead prediction. Thus, as the next step, we transformed the multi-step ahead forecasts into a technical trading rule similar to what we mentioned in section 3.5.2.
Table 26: $U$-Statistics for Independent Value Prediction and Recursive strategy with Mahalanobis Distance

<table>
<thead>
<tr>
<th>Currency</th>
<th>Independent Value Prediction</th>
<th>Recursive Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.00548527</td>
<td>0.03485111</td>
</tr>
<tr>
<td>GBP</td>
<td>0.00466984</td>
<td>0.09609811</td>
</tr>
<tr>
<td>JPY</td>
<td>0.00726693</td>
<td>0.03549428</td>
</tr>
<tr>
<td>CHF</td>
<td>0.00710084</td>
<td>0.01224103</td>
</tr>
<tr>
<td>CAD</td>
<td>0.00657776</td>
<td>0.05778198</td>
</tr>
</tbody>
</table>

5.2.4 Trading Decisions with Multi-step Ahead Forecasting.

To make the trading decisions, buy and sell, we modified the estimated return, $\hat{r}_t$, by considering the average of log access returns for the next $h$-time step ahead forecasts.

$$\hat{r}_t = \frac{1}{h} [\ln(\hat{x}_{t+1}) + \ln(\hat{x}_{t+2}) + ... + \ln(\hat{x}_{t+h})] - \ln(x_t) + \ln(1 + i_t') - \ln(1 + i_t)$$  \hspace{1cm} (5.21)

The positive returns are considered as buy signals, and negative returns are considered as sell signals. The rest of the decision making process was similar to what we described in section 3.5.2.

First, we compared the trading performances having $h$ prediction values with Mahalanobis distance and the other distances.

Table 27: Total Return

<table>
<thead>
<tr>
<th>Currency</th>
<th>Mahalanobis distance</th>
<th>Euclidean distance</th>
<th>Absolute distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.691171763</td>
<td>0.60727455</td>
<td>0.665890793</td>
</tr>
<tr>
<td>GBP</td>
<td>4.168072273</td>
<td>4.168072273</td>
<td>4.168072273</td>
</tr>
<tr>
<td>JPY</td>
<td>0.765389811</td>
<td>-0.054455887</td>
<td>-0.047796469</td>
</tr>
<tr>
<td>CHF</td>
<td>5.167428736</td>
<td>5.146126768</td>
<td>5.202706825</td>
</tr>
<tr>
<td>CAD</td>
<td>4.431424102</td>
<td>4.024764579</td>
<td>4.031466263</td>
</tr>
</tbody>
</table>

The estimated total return for the technical trading strategy under different distance measures are given in table 5.2.4. The final conclusion of distance choice was similar to that of the error measures. Mahalanobis distance outperforms the traditional distance functions in most of the cases except for the CHF data set.
Finally, we compare the trading performances with single-step-ahead forecasting and multi-step ahead forecasting with Mahalanobis distance. The obtained results support the claim that having more than one prediction helps the trader to make more accurate trading decisions and hence earn more money.

<table>
<thead>
<tr>
<th>Currency</th>
<th>Single-step-ahead Forecasting</th>
<th>Multi-step-ahead Forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUR</td>
<td>0.5299177667</td>
<td>0.691171763</td>
</tr>
<tr>
<td>GBP</td>
<td>4.168072273</td>
<td>4.168072273</td>
</tr>
<tr>
<td>JPY</td>
<td>0.677554044</td>
<td>0.765389811</td>
</tr>
<tr>
<td>CHF</td>
<td>5.421088794</td>
<td>5.167428736</td>
</tr>
<tr>
<td>CAD</td>
<td>4.377121756</td>
<td>4.431424102</td>
</tr>
</tbody>
</table>

5.3 Concluding Remarks

In this work, we conducted an empirical study on multi-step-ahead forecasting in the FX market with \(k\)-nearest neighbor algorithm. As in the single step prediction, our proposed Mahalanobis distance measure outperforms the traditional Euclidean and absolute distance measures when considering both forecasting accuracy and trading decisions. Our primary goal here was to adopt a commonly used and the most appropriate multi-step ahead prediction technique for the FX trading trading. The results supported our claim that having more forecasts can improve the trading decisions in FX trading.
Chapter 6
Concluding Remarks and Future Work

In this dissertation, we used Mahalanobis distance in place of traditional Euclidean distance in $k$-nearest neighbor forecasting procedure. We showed that comparing forecasting accuracy and trading decisions, Mahalanobis distance based forecasting procedure outperforms the traditional Euclidean and absolute distance based $k$-NN majority of the time for foreign exchange data.

Selecting embedding dimension, $m$ and the number of history vectors, $k$ is highly important in $k$-NN forecasting. We tested different values of embedding dimension and the number of history vectors comparing various forecasting accuracy measures, such as mean square error (MSE), $U$-statistic, and normalized root mean square error (NRMSE) for real FX data. We concluded that, for the $k$-NN forecasting procedure, an embedding dimension of size 3 and 20 history vectors are sufficient enough for daily FX data.

Time series forecasting techniques such as autoregressive moving average process (ARIMA) are widely used in time series data. As the next task, we have compared the Mahalanobis distance based nearest neighbor forecasts with ARIMA forecasting. Both forecasts were converted into trading decisions using the same technical trading strategy. The results showed that Mahalanobis distance based nearest neighbor procedure outperforms the time series method ARIMA majority of the time in terms of forecasting accuracy and trading performances.

Multi-step ahead prediction is an important task in many areas of time series. We compared the trading performances of multi-step ahead forecasting $k$-NN forecasting with the single step ahead forecasting for the same currency data for day trading. The results supported our claim of having more forecasts can improve the trading performances in FX trading.
In this dissertation, we provided evidence that Mahalanobis distance based $k$-nearest neighbor procedure outperforms the traditional Euclidean distance based algorithm by comparing both the forecasting accuracy and trading performances. During this study, we identified a couple of important facts, which can improve the $k$-NN algorithm even further. For highly volatile time instances, uniformly selected neighbors might not be that accurate. In our future work, instead of choosing a fixed number of neighbors at each time instance, we will employ dynamically adoptive number of neighbors for the nearest neighbor forecasting algorithm.

As the covariance measures the linear dependency between data, our next task is to adopt a different distance concept which captures the nonlinear dependency. As the currency data are nonlinear, having a similarity measure which captures the nonlinear dependency might be able to further strengthens the $k$-NN forecasting procedure.

Our results showed that with a proper distance choice, $k$-NN procedure can well capture the similar instances. Even with Mahalanobis distance, when there is a directional change in data, the forecasts did not follow the true direction of the data accurately. Therefore, our future research will combine the change point detection method with Mahalanobis distance based $k$-NN for foreign exchange rate to forecast improved results.


86


87


89


[69] James C Robinson, A topological delay embedding theorem for infinite-dimensional dynamical systems. Nonlinearity 18 (2005), 21352143


