xxxxx xxxxxxxxxx xx xxxxxxx xxx xxxxxxxxx **xx** (2015), no. N, xxx–xxx

MAHALANOBIS BASED k-NEAREST NEIGHBOR FORECASTING VERSUS TIME SERIES FORECASTING METHODS

VINDYA I. KUMARI PATHIRANA[1] AND KANDETHODY M. RAMACHANDRAN[2]

[1]Department of Mathematics and Statistics,
[2]Interdisciplinary Data Sciences Consortium (IDSC)
University of South Florida
Tampa, FL 33620-5700
[1] *E-mail:* vkumari@mail.usf.edu
[2]*E-mail:* ram@usf.edu

ABSTRACT

Foreign exchange (FX) rate forecasting is a challenging area of study. 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 k-nearest neighbor (k-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. As the k neighbors are selected according to a a distance function, the choice of distance plays a key role in the k-NN procedure.

The time series forecasting method, Auto Regressive Integrated Moving Average process (ARIMA) is considered as one of the superior forecasting method for time series data. In this work, we compare the performances of Mahalanobis distance based k-NN forecasting procedure with the traditional ARIMA based forecasting algorithm. In addition, the forecasts were transformed into a technical trading strategy to create buy and sell signals. The two methods were evaluated for their forecasting accuracy and trading performances.

1. INTRODUCTION

The foreign exchange (FX) 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

Received xxxx xx, 2015

currency movements. A preliminary global study by the Bank for International Settlements from the 2013 Triennial Central Bank Survey of Foreign Exchange and OTC Derivatives Markets Activity show that trading in foreign exchange markets averaged 5.3 trillion dollars per day in April 2013. Thus, foreign exchange rates forecasting is one of the challenging and important applications of financial time series prediction.

Foreign exchange rate forecasting is a challenging task due to the nonlinearity and the highly correlated nature of the data [10, 25]. Nonlinear dynamical systems are becoming more popular and relevant forecasting techniques due to these data structure. Neighbor Algorithms is one of the most popular such non-linear pattern recognition algorithm, which dates back to an unpublished report by Fix and Hodges in 1951, [6]. The basic principle of k-nearest neighbor (k-NN) rule is to investigate the past behavior of the currency data so that it can fully capture the dependency of the future exchange rates and that of the past. As a pattern recognition algorithm, k-NN looks for the repetitions of specific price patterns such as major trends, critical or turning points.

k-nearest neighbor forecasting procedure is mainly based on the similarity structure of the past and the present. The recognized "nearest neighbors" are the only data values used in the forecasting algorithm. The term 'nearest' is determined by a distance metric. Thus, it is highly important to have a distance function which captures the true nature of the data. Among nearest neighbor methods, Mahalanobis distance proved to be more efficient [19, 20]. In this paper, we will compare Mahalanobis based nearest neighbor method of forecasting to some of the popular time series based methods.

In section 2, we will give some background material on distance measures, and error measures. In section 3 we will discuss some of our previously obtained results of choosing embedding Dimension (m), number of nearest neighbors (k) and also present the comparison results of Mahalanobis distance and other popular distance choices. In section 4, we will give some background material on times series forecasting methods and present the comparison results of proposed Mahalanobis distance based k-NN and ARIMA forecasting models. Conclusion will be given in section 5.

2. BACKGROUND

2.1. k-Nearest Neighbor Algorithm and the Choice of Distance.

k-nearest eighbor (k-NN) algorithm rank the data and chose the k closest of them based on the distance between the query vector and the historical values. First, we divide the time series data, $\{x_t\}_{t=1}^{n} = \{x_1, x_2, ..., x_n\}$ in to two separate parts; for $T < n$, a training (in-sample) set $\{x_1, x_2, ..., x_T\}$ and a testing (or out-of-sample) set $\{x_{T+1}, x_{T+2}, ..., x_n\}$. In order to identify behavioral patterns in the data, we transform the scalar time series in to time series vectors. We need to choose an embedding dimension (m) and delay time (τ) to create vectors out of the training set. After selecting m and τ, a time series vector at time t

can be written as;

$$x_t^{m,\tau} = (x_t, x_{t-\tau}, ..., x_{t-(m-1)\tau}) \quad \text{for} \quad 1+(m-1)\tau \leq t \leq T \tag{2.1}$$

These m-dimensional vectors are often called as $m-histories$ and the m-dimensional space $\mathbb{R}^m$ is referred to be the phase space of the time series [10].

The primary goal of k-NN method is to use the most relevant vectors out of the training set in the forecasting. The most relevant vectors are the ones having similar dynamic behavior as the delay vector x_T^m. We compare the distance between the delay vector and all the other m-history vectors to choose the vectors with similar dynamic behavior [10]. Then we look for the closest k vectors in the phase space $\mathbb{R}^m$ such that they minimize the distance function $d(x_T^m, x_i)$.

In k-NN algorithm, m and k are predetermined constants. In the literature, the optimal values of m and k are quite ambiguous. There have been quite a lot argument and discussions about the optimal choice of m and k since the NN rule was first officially introduced by Cover and Hart in 1967 [7, 23]. In section 3 We will discuss the choice of m and k for Mahalanobis distance along with other distance choices.

For the forecasting we can incorporate variety of Statistical and time series predicting methods with NN algorithm. In the literature of k-NN forecasting, the most commonly used forecasting method is locally weighted simple linear regression [3, 10]. Thus the future forecasts were obtained using the following locally adjusted linear regression model [7]:

$$\hat{x}_{T+1} = \sum_{n=0}^{m-1} \hat{a}_n x_{T-m\tau} + \hat{a}_m \tag{2.2}$$

The coefficients were fitted by the linear regression of $x_{t_j+1}^m$on $x_{t_j}^m = (x_{t_j}, x_{t_j-\tau}, ..., x_{t_j-(m-1)\tau}$ for $j = 1, 2, ..., k$. Thus the estimated coefficients $\hat{a}_i$ are the values of a_i that minimize

$$\sum_{j=1}^{k} (x_{t_j+1} - a_0 x_{t_j} - a_1 x_{t_j-1} - ... - a_{m-1} x_{t_j-(m-1)\tau} - a_m)^2 \tag{2.3}$$

The data used in equation (2.3) are the only $k(m+1)$ data values obtained from the k-neighbor vectors of size m and the corresponding next values, $x_{t_j+1}^m$ for $j = 1, 2, ..., k$ chosen neighboring vectors, not the entire data.

As the forecasting is completely based on the selected k nearest neighbors, it is highly important to use a distance function which captures the relevant behavior of the data accurately. Many researchers have pointed out the difficulty of choosing a distance measure for the NN algorithm that works well for different types of data. Over the past decades, the most common choice of distance was Euclidean distance [7, 10]. The way it is defined, the Euclidean distance is unable to capture the trend of the highly volatile (hence random) and highly correlated foreign exchange data when choosing the neighbors for the NN algorithm. 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 [13, 23].

Even though the asymptotic probability of error of the NN is independent of the choice of metric, classification performance of finite sample nearest neighbor algorithm is not independent of the distance function [13, 17]. As Nearest neighbor rule is highly sensitive to outliers, selecting irrelevant neighbors can cause increase in forecasting error. In their work, Fukunaga $\&$ Hostetler showed that using a proper distance measure, the variance of the finite sample estimate can be minimized [13]. Short $\&$ Fukunaga investigate the relation between the distance function in k-NN and the error measure [13]. They concluded that the error can be minimized by using an appropriate distance metric without increasing the number of sample vectors.

2.2. k-Nearest Neighbor and Distance Measure.

In time series pattern recognition, an appropriate distance function can categorize data in to clusters by capturing the similarity or dissimilarity between the data. The following distance measures are commonly used in nearest neighbor classification and forecasting algorithm.

Consider n-dimensional vectors $x = (x_1, x_2, ..., x_n)$ and $y = (y_1, y_2, ..., y_n)$ in $\mathbb{R}^m$.

The $Euclidean\ distance$ between x and y is defined as

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{2.4}$$

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.

$Manhattan\ \ distance$ gets its name from the rectangular grid patterns of the streets in Manhattan [18]. The Manhattan distance between x and y in $\mathbb{R}^m$ is defined as:

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i| \tag{2.5}$$

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 sometime, because the distance along each axis is not squared, a large difference in one of the dimensions will not affect the total outcome.

$Mahalanobis\ distance$ was introduced by P. C. Mahalanobis in 1936 by considering the possible correlation among the data [9]. It is defined between two vectors x and y as:

$$d(x,y) = \sqrt{(x-y)'\sum^{-1}(x-y)} \tag{2.6}$$

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

Mahalanobis distance belongs to the class of generalized ellipsoid distance defined by

$$d(x,y) = \sqrt{(x-y)'M(x-y)} \tag{2.7}$$

Here M is a positive definite, symmetric matrix. In the case the 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 [24]. As the axes of ellipsoidal sphere can assume any direction depending upon the data, 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 [7, 17].

In our earlier work, we proposed to use Mahalanobis distance in k-NN algorithm for FX data. We compared the performance of the Mahalanobis distance based k-NN algorithm with popular Euclidean and Manhattan distance based algorithm. Some of the earlier results are given in the next section.

The performance of the Mahalanobis distance based K-nearest neighbor algorithm was compared with the time series forecasting technique, ARIMA in two ways:

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

In the former case, our goal is to capture the deviation of the fitted values against the actual observations. In the latter case, we are interested in looking at the forecasts in financial point of view. For that we create trading signals, buy and sell using a technical trading rule [10] and the performances were evaluated by the commonly used performance measures in practice.

2.3. **Measures of Forecasting Accuracy.**

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

Mean square error (MSE): Mean square error defined by

$$MSE = \frac{1}{n}\sum_{t=1}^{n}(\hat{x}_t - x_t)^2 \tag{2.8}$$

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.

Means 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.

$$MAPE = \frac{100}{n}\sum_{t=1}^{n}|\frac{\hat{x}_t - x_t}{x_t}| \tag{2.9}$$

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.

Theil's U- statistic (U): We considered the following version of Theil's U- statistic to compare the forecasting accuracy of our model.

$$U = \frac{\sqrt{\sum_{t=1}^{n}(\hat{x}_t - x_t)^2}}{\sqrt{\sum_{t}^{n}(\hat{x}_t)^2} + \sqrt{\sum_{t}^{n}(x_t)^2}} \tag{2.10}$$

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.

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:

$$NRMSE = \frac{\sum_{t=1}^{n} e^2(t)}{\sigma} = \frac{\sqrt{\sum_{t=1}^{n}(\hat{x}_t - x_t)^2}}{\sigma} \tag{2.11}$$

Here, σ is the standard deviation of the time series [3]. 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.

2.4. **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. The primary goal of foreign exchange rate forecasting has to be making proper trading signals: buy and sell at each time step so that the trader makes more money. To satisfy this main aspect, first we need to transform forecasted values in to trading signals. The forecasts were transformed into a simple technical trading strategy using the trading rule used by Fernandez-Rodriguez, Sosvilla-Rivero, and Andrada-Felix in their work [6, 7]. Let $\hat{r}_t$ given by

$$\hat{r}_t = \ln(\hat{x}_{t+1}) - \ln(1 + i_t^{'}) - \ln(1 + i_t) \tag{2.12}$$

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 x_{t+1} is the domestic (US) daily interest rate and $i^{'}$ is the foreign country daily interest rate. The trading signals at time t are made based on the estimated return $\hat{t}_t$. The positive returns are executed as long positions (buy) and the negative returns are executed as short position (sell) [6, 7]. So the trading decision can be given as

$$\hat{z}_t = \begin{cases} 1 & ;\text{if} \quad \hat{r}_t > 0 \\ -1 & ;\text{if} \quad \hat{r}_t < 0 \end{cases} \tag{2.13}$$

Based on estimated return, we calculate $estimatedtotal(logaccess)return$ of the trading strategy over the time period $(1, n)$ as

$$\hat{R}_n = \sum_{t=1}^{n} \hat{z}_t r_t \tag{2.14}$$

Here r_t is the actual return at time given by

$$r_t = \ln(x_{t+1}) - \ln(x_t) - \ln(1 + i_t^{'}) - \ln(1 + i_t)$$

We also consider 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,

$$S_R = \frac{\mu_{\hat{R_n}}}{\sigma_{\hat{R}_n}} \tag{2.15}$$

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

3. k-NEAREST NEIGHBOR FORECASTING WITH MAHALANOBIS DISTANCE

3.1. Data.

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-$\acute{a}$-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.

3.2. Embedding Dimension (m) and Number of Nearest Neighbors (k).

The choice of embedding dimension, m, and the number of nearest neighbors, k in the k-NN forecasting procedure is a key issue need to be addressed prior to making trading signals. Therefore, first we conducted an empirical investigation to find the optimal values of m and k. We wanted to figure out whether the choices for m and k are data dependent, and also distance dependent. The forecasting accuracy was compared using all the error measures mentioned in section 2.3, by varying the value of m and k along with different distance functions. 80% of the data was considered as the training set, and the remaining 20% was taken as the testing set. After analyzing the results, the key parameters m and k of the algorithm were chosen as 3 and 20, respectively. The complete results of choosing the embedding dimension (m) and neighborhood size (k) can be found in [20].

3.3. Forecasting Accuracy and Trading Performances of Mahalanobis Distance vs. Other Distance Measures.

As discussed in section 2.2, the choice of distance plays a key role in the nearest neighbor algorithm. in our earlier published work, we have provided sufficient evidence supporting Mahalanobis distance as the choice of distance in k-NN procedure [19, 20]. We came to this conclusion by comparing the forecasting accuracy as well as the trading performances of the 5 currency data sets with Mahalanobis distance and other traditional distances such as Euclidean and absolute distances. The Mahalanobis distance outperforms the traditional distance functions for all the data sets with respect to the forecasting accuracy and trading performances. The details results can be found in [19] and [20].

4. k-NEAREST NEIGHBOR FORECASTING VERSUS TRADITIONAL TIME SERIES FORECASTING

In this section our main goal is to compare the trading performances of Mahalanobis based k-nearest neighbor procedure with popular time series forecasting method, autoregressive moving average (ARIMA) in the foreign exchange market. First, we will briefly discuss some of the most important time series forecasting methods are used in time series data analysis. Then, we will go over the step-by-step model building procedure for the same 5 currency data sets used in the previous section. Finally, we will compare the forecasting accuracy, and the trading performances of time series models with our proposed k-NN forecasting algorithm.

The autoregressive process (AR) and the moving average process (MA) were very useful representations 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.

4.1. **The General mixed Autoregressive Moving Average (ARMA) Models.**

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;

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + ... + \phi_p x_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_p \epsilon_{t-q} \tag{4.1}$$

or

$$\Phi_p(L) x_t = \alpha + \Theta_q(L) \epsilon_t \tag{4.2}$$

where

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

and

$$\Theta_q(L) = 1 + \theta_1 L + \theta_2 L^2 + \theta_3 L^3 + ... + \theta_q L^q \tag{4.4}$$

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 [**?**, 16]. Note that we need to make the assumption of $\Theta_q(L) = 0$ and $\Phi_p(L) = 0$ sharing no common roots [**?**, 16].

4.2. **The General ARIMA Model.**

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., which transforms the general stationary ARMA process in to non stationary ARIMA(p,d,q) process. Here d is the degree of differencing. The difference filter is normally given by

$$(1-L)^d \quad where \quad L^j x_t = x_{t-j} \tag{4.5}$$

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

$$\Phi_p(L)(1-L)^d x_t = \alpha + \Theta_q(L)\epsilon_t \tag{4.6}$$

where

$$\Phi_p(L) = 1 - \phi_1 L - \phi_2 L^2 - \phi_3 L^3 - ... - \phi_p L^p$$
$$\text{and}$$
$$\Theta_q(L) = 1 + \theta_1 L + \theta_2 L^2 + \theta_3 L^3 + ... + \theta_q L^q$$

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 Kwiatkowski-Phillips-Schmidt-Shin (KPSS) test and Augmented Dickey-Fuller unit root test are normally used to determine the stationarity of a time series [21].

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

$$ln(L) = 2M - 2ln(L) \tag{4.7}$$

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

$$ln(L) = -\frac{n}{2}ln(2\pi\sigma^2) - \frac{1}{2}\sigma^2\sum_{i=1}^{n}(x_i - \mu)^2 \tag{4.8}$$

Here, μ and σ 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.

4.3. **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 section 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 paper 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 2.3 to compare the forecasting accuracy. As the secondary step, the ARIMA forecasts will be transformed in to trading signals using the same technical trading strategy discussed in section 2.4 and compare withe the trading performances of k-NN procedure.

As discussed in Section 4.2, the order of differencing will be determined using the Kwiatkowski-Phillips-Schmidt-Shin (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. The Akaike information criterion (AIC) was computed for each model to chose the model with the minimum AIC.

4.3.1. ***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:

$$(1 - 0.1329L)(1 - L)x_t = 0.00019 + (1 - 0.1323L)\epsilon_t \tag{4.9}$$

$$x_t = 0.000192 + 1.132915x_{t-1} - 0.132915x_{t-2} + \epsilon_t - 0.132342\epsilon_{t-1} \tag{4.10}$$

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

$$x_t = 0.000192 + 1.132915x_{t-1} - 0.132915x_{t-2} - 0.132342\epsilon_{t-1} \tag{4.11}$$

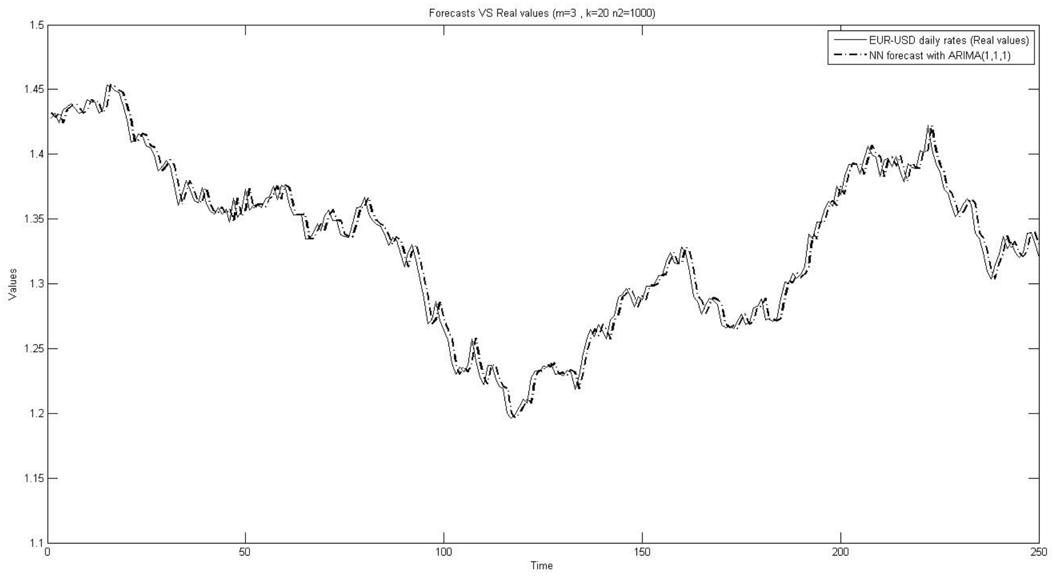

FIGURE 1. ARIIMA(1,1,1) Forecasts and real values for EUR/USD daily exchange rates.

The figure 1 shows the predictions on top of the original time series for the forecasting period.

4.3.2. ***ARIMA Forecasting Model for GBP/USD Daily Rates.***
The forecasting model with minimum AIC for the GBP/USD exchange rates data set was ARIIMA(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:

$$(1 + 0.636309L)(1 - L)x_t = -0.000218556 + (1 + 0.652109L + 0.063216L^2)\epsilon_t \quad (4.12)$$

or

$$x_t = -0.000218556 + 0.363691x_{t-1} + 0.636309x_{t-2} + \epsilon_t + 0.652109\epsilon_{t-1} + 0.063216\epsilon_{t-2} \quad (4.13)$$

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

$$x_t = -0.000218556 + 0.363691x_{t-1} + 0.636309x_{t-2} + 0.652109\epsilon_{t-1} + 0.063216\epsilon_{t-2} \quad (4.14)$$

The 250 out-of-sample forecast with the original time series are given in figure 2.

4.3.3. ***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)

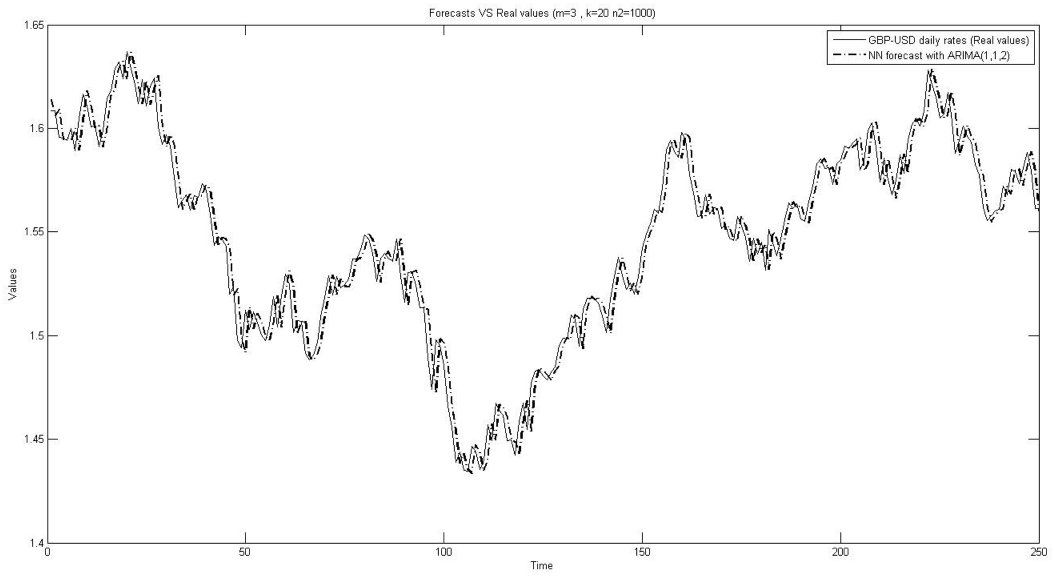

FIGURE 2. ARIMA(1,1,2) Forecasts and real values for GBP/USD daily exchange rates

model, that is a combination of second order autoregressive (AR), and a first order moving average (MA), with the first difference filter.

$$(1 - 0.664950L)(1 - L)x_t = 0.00000098 + (1 - 0.400612L - 0.234753L^2)\epsilon_t \quad (4.15)$$

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

$$x_t = 0.00000098 + 1.0.664950x_{t-1} - 0.664950x_{t-2} - 0.400612\epsilon_{t-1} - 0.234753\epsilon_{t-2} \quad (4.16)$$

The figure 3 shows the forecasts with the actual values.

4.3.4. ***ARIMA Forecasting Model for CHF/USD Daily Rates.***

For the CHF/USD daily rates, we found that ARIIMA(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:

$$(1 + 0.0881215L)(1 - L)x_t = 0.000231 + (1 + 0.371622L)\epsilon_t \quad (4.17)$$

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

$$x_t = 0.000231 + 0.918785x_{t-1} + 0.0881215x_{t-2} - 0.0881215\epsilon_{t-1} \quad (4.18)$$

The figure 4 shows the forecasts with the actual values.

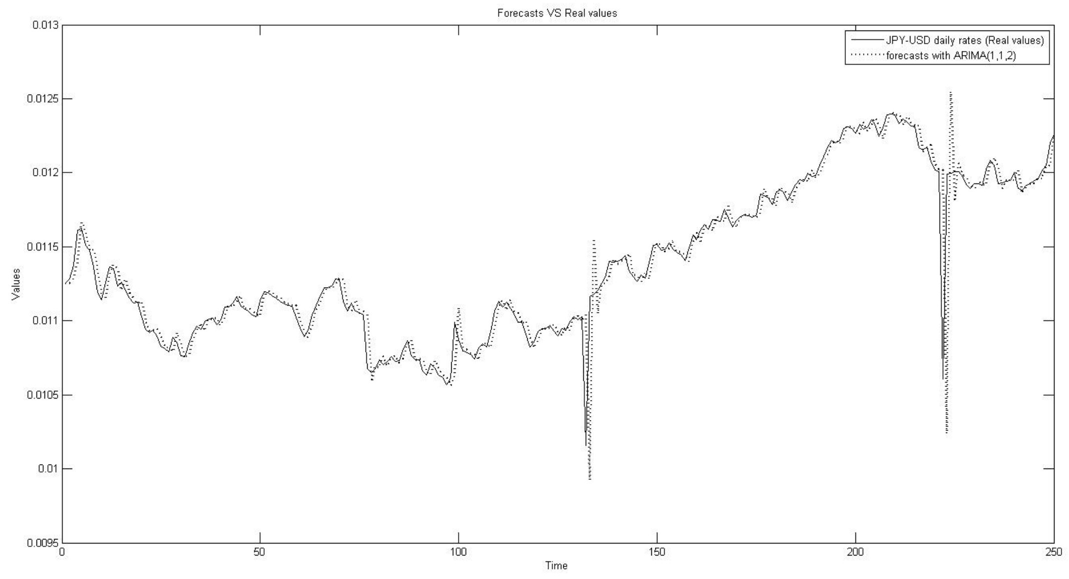

FIGURE 3. ARIMA(1,1,2) Forecasts and real values for JPY/USD daily exchange rates.

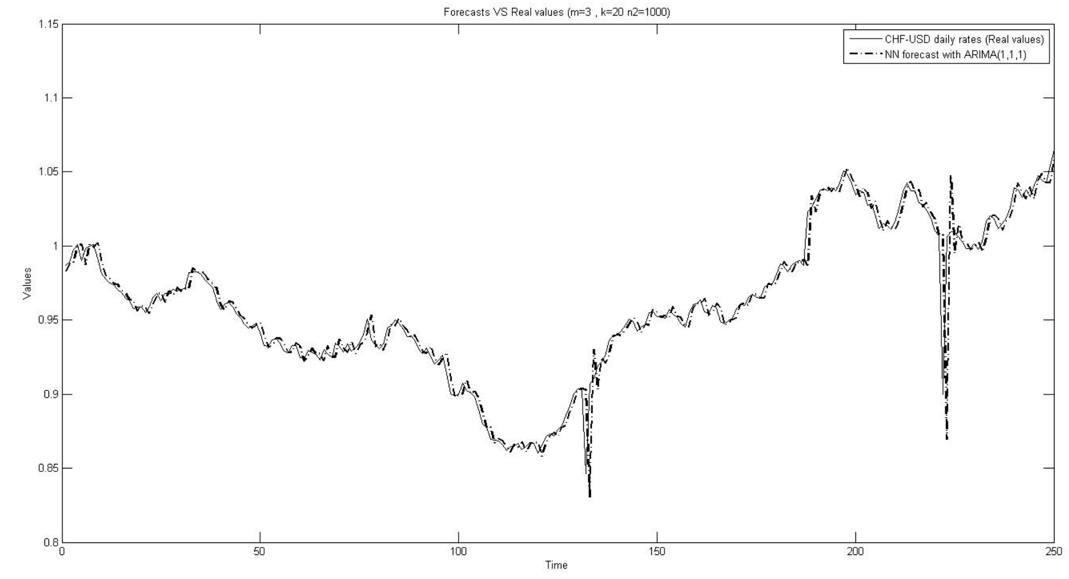

FIGURE 4. ARIMA(1,1,1) Forecasts and real values for CHE?USD daily exchange rates.

4.3.5. ***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)

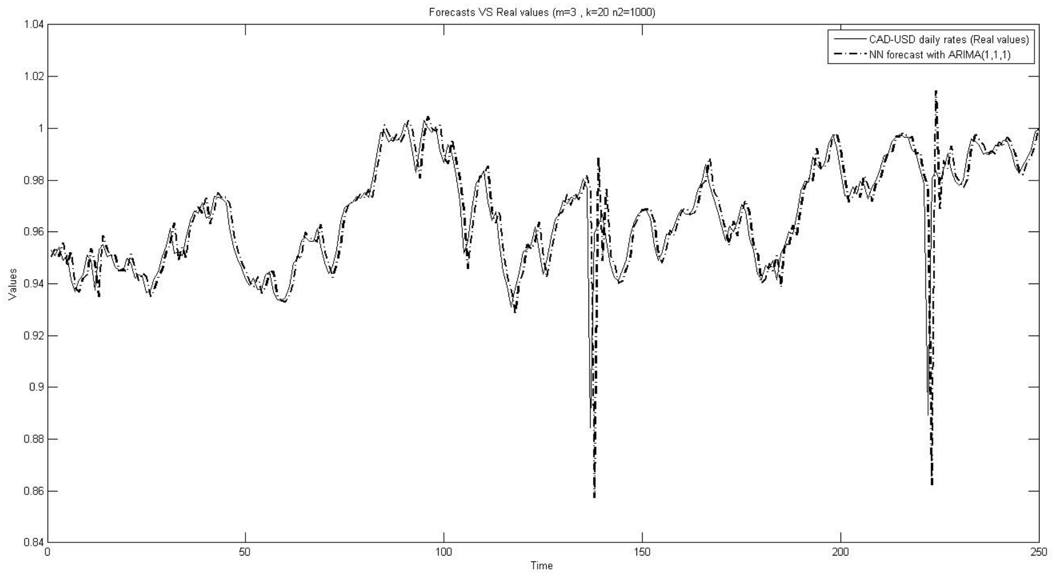

FIGURE 5. ARIMA(1,1,3) Forecasts and real values for CAD/USD daily exchange rates.

with the first difference filter, namely, ARIMA(1,1,3).

$$(1-0.481798L)(1-L)x_t = 0.000055 + (1-0.182836L-0.130882L^2+0.067018L^3)\epsilon_t \tag{4.19}$$

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

$$x_t = 0.000055+1.481798x_{t-1}-0.481798x_{t-2}-0.182836\epsilon_{t-1}-0.130882\epsilon_{t-2}+0.067018\epsilon_{t-2} \tag{4.20}$$

The figure 5 shows the forecasts with the actual values.

4.4. k-Nearest Neighbor Forecasting vs. ARIMA Forecasting - Forecasting Accuracy.

In section 4.3, we discussed in details the general autoregressive integrated moving average forecasting models for the five daily exchange rates data sets. The given figures (fig.1-5) indicate that the ARIMA forecasts follow the actual values pretty well as similar to the case of k-NN forecasting. In this section, our goal is to compare the forecasting accuracy of ARIMA models with our proposed Mahalanobis distance based k-nearest neighbor forecasting procedure.

We considered all the accuracy measures mentioned in section 2.3 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 Mahalanobis distance based k-NN algorithm and ARIMA forecasting models.

TABLE 1. U-statistics with k-NN and ARIMA models

Currency	k-NN forecasting	ARIMA forecasting
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

TABLE 2. Mean square error with k-NN and ARIMA models.

Currency	k-NN forecasting	ARIMA forecasting
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

TABLE 3. Normalized root mean square error with k-NN and ARIMA models

Currency	k-NN forecasting	ARIMA forecasting
EUR	0.16748184	0.14691536
GBP	0.22251281	0.20423380
JPY	0.30304086	0.35985560
CHF	0.21693213	0.28301529
CAD	0.58311581	0.65600438

As can be seen from the obtained results, given by tables 1, 2, & 3, we can see that majority of the time (3 out of 5) Mahalanobis distance 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 paper is to compare the trading performances of both methods, it is necessary to 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.
  $$Total\ variation = \sum_{t=1}^{n}(x_t - \bar{x})^2 \tag{4.21}$$
- 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.
  $$Standard\ deviation,\ \sigma = \sqrt{\frac{\sum_{t=1}^{n}(x_t - \bar{x})^2}{n}} \tag{4.22}$$

TABLE 4. Total Variation and Standard Deviation

Currency	Total variation	Standard deviation
EUR	10.99275618	0.104846346
GBP	36.55015706	0.191180954
JPY	0.000877177	0.000936577
CHF	4.313622939	0.065678177
CAD	4.193662329	0.064758492

Considering the calculated values for total variation and standard deviation (given in table 4), 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 withe 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 section 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 2.4, and compared the results with Mahalanobis distance based nearest neighbor trading decisions.

4.5. 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 (k-NN and ARIMA) considering their trading performances. We transformed the ARIMA forecasts in to trading signals, buy and $sell$ using technical trading strategy discussed in section 2.4. Then, the performance measures, $total$ ($log\ access$) $return$ and $Sharpe\ ratio$ were calculated and compared with those of 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 k-NN algorithm and ARIMA process are given in tables 5 and 6. The final conclusion of forecasting model is pretty much same as that of error measures. Proposed Mahalanobis distance based 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 k-nearest neighbor forecasting model producing more accurate and profitable trading signals compared to the general ARIMA process.

The results from section 4.4 and section 4.5 motivates 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

TABLE 5. Total Return k-NN and ARIMA models

Currency	k-NN forecasting	ARIMA forecasting
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

TABLE 6. Sharpe Ratio with k-NN and ARIMA models

Currency	k-NN forecasting	ARIMA forecasting
EUR	0.27890809	0.50803011
GBP	2.41593434	2.41593434
JPY	0.18429771	0.12818451
CHF	1.67419376	1.42328537
CAD	1.26400087	1.04713284

Mahalanobis distance based k-NN method and the general ARIMA forecasting models.

5. **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. 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. 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 [2].

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 [2] exchange rates. Also, Oyediran & Afieroho 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 [18]. 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. 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 [15]. 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 4.4 and section 4.5 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 the table 4, 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:

The model comparison was performed using the accuracy measures discussed in section 2.3. 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 section 3 and 4.5 suggest that having more accurate forecasts always lead to a higher trading performances.

We followed the same data preparation procedure discussed in section 4.2 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

TABLE 7. Standard deviations of the simulated data

Data Set	Standard deviation
Simulated data set 1	0.896010158
Simulated data set 2	0.400925413
Simulated data set 3	0.283497079
Simulated data set 4	0.126783748
Simulated data set 5	0.040092541
Simulated data set 6	0.012678375
Simulated data set 7	0.004009254
Simulated data set 8	0.001267838
Simulated data set 9	0.001267838

as in section 4.4. One step ahead out of sample forecasts were created for 250 test set and the size of the training window was 1000.

TABLE 8. U-Statistics for k-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.00127 and 0.896

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

The comparison results of U-statistic for the simulated data are presented in Table 8 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 9, 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 9. MSE for k-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.00127 and 0.896

Data Set D_i	Standard deviation	k-NN MSE	ARIMA MSE
D_1	0.896010158	0.75482502	0.620606842
D_2	0.400925413	0.15148122	0.127060368
D_3	0.283497079	0.07574061	0.066274057
D_4	0.126783748	0.01514812	0.018027316
D_5	0.040092541	0.00151481	0.007485055
D_6	0.012678375	0.00015148	0.006544971
D_7	0.004009254	0.00001515	0.006487096
D_8	0.001267838	0.00000151	0.006492738
D_9	0.001267838	0.00000151	0.000405499

We went further and tried to figure out exactly around what value of standard deviation k-NN procedure starting to work better. Table 8 and 9 clearly indicate that somewhere between the values of 0.127 & 0.283, k-NN forecasting procedure has started performing better. To investigate this furthers, 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 10, we can observe that for the standard deviation values below 0.13, the k-NN has a better forecasting accuracy.

6. **Concluding Remarks**

In this paper, our main goal was to compare the proposed Mahalanobis distance based k-NN forecasting with general autoregressive integrated moving average (ARIMA) process, which is assumed to be one of the best time series forecasting technique. As all these

TABLE 10. U-statistic for k-NN forecasts and ARIMA forecasts: Simulated data with standard deviation between 0.009 and 0.15

Simulated data sd	k-NN U-statistic	ARIMA U-statistic
0.15010000	0.031257268	0.030175564
0.13000000	0.02708489	0.02728483
0.12750000	0.026565797	0.026934802
0.12030000	0.0250823020	025948689
0.10540000	0.021970607	0.023958385
0.10030000	0.020912901	0.02330971
0.09100000	0.018988001	0.022171638

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 section (section 3) that choosing an appropriate distance in NN algorithm can improve the forecasting significantly. The results obtained in this paper 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.

References

[1] Tao Ban, Ruibin Zhang, Shaoning Pang, Abdolhossein Sarrafzadeh, Daisuke Inoue, Referential k-NN Regression for Financial Time Series Forecasting in *Neural Information Processing: Lecture Notes in Computer Science* (2013), 8226, 601 - 608.

[2] Sergio Bianchi, Alexandre Pantanella, and Augusto Pianese, Modeling and Simulation of Currency Exchange Rates Using Multifractional Process with Random Exponent., *International Journal of Modeling and Optimization), ESTSP08* (June 2012) 2, No 3

[3] M. Casdagli, Nonlinear forecasting, Chaos and statistics, Modeling Complex Phenomena, Part of the series Woodward Conference pp 131-152, Springer, 1992.

[4] M. Casdagli, Nonlinear prediction of chaotic time series. *Physica D* (1989), 35: 335-356

[5] M. Casdagli, Chaos and Deterministic versus Stochastic Nonlinear Modeling *Journal of Royal Statistical Society, Series B Vol. 54, No 2* (1992), 303–328 .

[6] T. M. Cover, P. E. Hart, Nearest Neighbor Pattern Classification, *IEEE Trans. on Information Theory, IT-13 (1)* (1967), 21–27.

[7] L. Devroye, Gyorfi and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*, Springer-Verlag, New York, 1996.

[8] D. Farmer and J. Sidorowich, Predicting chaotic time series., *Physical review Letters* (1987), 59, 845-848.

[9] Fernando Fern$\acute{a}$ndez-Rodr$\acute{i}$guez, S. Sosvilla-Rivero, J. Andrada-F$\acute{e}$lix, Nearest-Neighbour Predictions in Foreign Exchange Markets, *FEDEA Int. Economics and Finance DEFI* (2002), Working Paper

[10] Fernando Fern$\acute{a}$ndez-Rodr$\acute{i}$guez, S. Sosvilla-Rivero, J. Andrada-F$\acute{e}$lix, Technical Analysis in Foreign Exchange Markets: Linear versus Nonlinear Trading Rules, *FEDEA Int. Economics and Finance DEFI* (2000), Working Paper No. 00-02.

[11] Fernando Fern$\acute{a}$ndez-Rodr$\acute{i}$guez, S. Sosvilla-Rivero, Testing nolinear forecastability in time series: Theory and evidence from the EMS, *Economics Letters* (1998), 59, 49 - 63.

[12] Fernando Fern$\acute{a}$ndez-Rodr$\acute{i}$guez, J. Andrada-F$\acute{e}$lix, S. Sosvilla-Rivero, Combining information in exchange rate forecasting: evidence from the EMS, *Applied Economics Letters* (1997), 4: 7, 441 444

[13] K. Fukunaga and L. D. Hostetler, Optimization of k-Nearest-Neighbor Density Estimates , *IEEE Transactions on information theory, Vol. IT-19* (May 1973), 320–326.

[14] N. A. Gershenfel and A. S. Weigend, The Future of Time Series: Learning and Understanding , in: *Time Series Predictions: Forecasting Future and Understanding the Past* (1993), 1–70.

[15] J. D. Hamilton, *Time Series Analysis*. Princeton, NJ: Princeton University Press, 1994

[16] Gebhard Kirchgassner and Jurgen Wolters, *Introduction to Modern Time Series Analysis* Springer, 2007.

[17] P. C. Mahalanobis, On the Generalized Distance in Statistics, in: *Proceedings of the national Institute of Science of India 12* (1936), 49–55.

[18] Oyelami Benjamin Oyediran and Edooghogho Afieroho, Simulation of Exchange Rates of Nigerian Naira Against US Dollar, British Pound and the Euro Currency, in: *Studies in Mathematical Sciences* 6, No. 2 (2013), 58-70.

[19] Vindya I. K. Pathirana and K. M. Ramachandran, The Distance Choice and Optimal parameter selection in k-NN algorithm for FX data , in: *The special issue in Communications in Applied Analysis*, 18 (2014), 591–612.

[20] Vindya I. K. Pathirana and K. M. Ramachandran, k-Nearest Neighbor Algorithm with Mahalanobis Distance for FX Trading, in: *proceedings of dynamic Systems and Applications*, 6 (2012), 324–328.

[21] Mohsen Pourahmadi, *Foundations of Time Series Analysis and Prediction Theory*, Wiley Series in Probability and Statistics , New York, 2001.

[22] Tim Sauer, James A. Yorke, and Martin Casdagli, Embedology. *Journal of Statistical Physics* (1991), 65

[23] D. Robert Short and K. Fukunaga, The optimal Distance Measure for Nearest neighbor Classification, *IEEE transactions on information theory* (1981), Vol. IT-27, No. 5, 622–627.

[24] T. Uemiya, T. Matsumoto, D. Koizumi, M. Shishibori, K. Kita, Fast Multidimensional Nearest Neighbor Search Algorithm Based on Ellipsoid Distanc, *International Journal of Advanced Intelligence* (2009).

[25] Walters-Williams, and Y. Li, Comparative Study of Distance Functions for Nearest Neighbor, in *Advanced Techniques in Computing and Software Engineeringl*, (2010), (Ed: Elleithy) 123–456.