[1]* R Sivasamy

Machine Learning-Based GRU, LSTM, HMM, and SARIMA Models for Gold Pricing

Journal of Electrical Systems

Abstract: - The principal objective is to assess the properties of a three-state hidden Markov model (HMM) over an 11,151-day historical gold price series. To find such a forecast, the gold price data are split into training and test samples, with the test subset containing current values for a 453-day period. Long short-term memory (LSTM), seasonal autoregressive integrated moving average (SARIMA), and gated recurrent units (GRUs) were the methods used to analyze the data. SARIMA, our reference model, handles the linear portion of time-series forecasts. The use of HMMs involves a wide range of topics, such as probability estimation, simulation using random number generators, estimation, and parametric characterization. LSTMs and GRUs are designed with a few internal gates to determine which of our time-series data is critical for keeping or discarding when solving short-term memory problems. The most important objective is to identify a high performer by utilizing performance metrics such as the root mean square error (RMSE) and mean absolute error (MAE). The outcomes were compared across methods using both quantitative and graphical presentations.

Keywords: Machine learning, forecasts, short-term memory, transition probability, posterior probability.

I. INTRODUCTION

Due to the clear interest of many large organizations, stock market prediction has been one of the most active study fields in the past. Different machine learning algorithms have been used in the past, with differing levels of success. However, because stock data is uncertain, seasonal, and non-stationary, stock forecasting is still highly constrained. Since projections based only on historical stock data are subject to various outlier influences, this task becomes considerably more difficult.

A. Review of Literature

Numerous scholarly publications address the statistical analysis of sequential data or time series modelling. The readers can find details pertaining to autoregressive integrated moving average (ARIMA), and SARIMA model fitting aspects in [1], [2] and [3].

An approach to the economic analysis of nonstationary time series was first introduced by [4]. He proposed a very tractable approach to modelling changes in regime as the outcome of a hidden discrete-state Markov process. In addition, the econometrician presented an algorithm for drawing such estimation of population parameters by the method of maximum likelihood and provides the foundation for forecasting future values of the series, [5]

Value-at-Risk was assessed by [6] using a "Recurrent Neural Network (RNN)". How to comprehend and enhance early stopping for learning with noisy labels in time series forecasting techniques was covered by [7]. For, how RNN learns the smoothing coefficients in addition to predicting the time series sequence and its predictive intervals (PI), see [8].

[9] has published a tutorial on hidden Markov models with a few applications in speech recognition. [10] created Hidden Markov Models for speech Recognition. [11] discussed several types of hidden Markov models for discrete valued spaces for both observable and latent processes. [12] focused on biological sequential analysis using probabilistic models. [13] wrote an introduction to hidden Markov models and Bayesian networks and their applications to real world problems. [14] investigated the switching pattern of subscribers of telecom sector of India. [15] studied the primary factors influencing the brand switching in cellular industry.

(Tanaka, 2017) considered a semi-observable discrete-time Markov decision process with partial discounted payoff, where he integrated hidden Markov models based on unobservable states and the impact of decisions over time. (Abbasimehr and Shabani, 2021) used clustering algorithms, then time series forecasting techniques

[1] *Corresponding author: Statistics Department, University of Botswana, Gaborone, Botswana

orcid.org/0000-0002-3158-928X

Email: sivasamyr@ub.ac.bw

Copyright © JES 2024 on-line : journal.esrgroups.org

are used to predict the behaviour of each segment. [16] have constructed a statistical model is to capture and quantify the relationships between the attitudes of the consumers. The statistical model used for the estimating the optimum possible sequence probabilities in repeat purchase and switching behaviour.

[17] and [18] pointed out that insufficient or excessive rain can cause great damage to the crops, and the environment. The artificial intelligence is an alternative predictive method which is not limited by the assumption of linearity, for such details refer to [19] [20], [21], [22], [3] and [2].

Another technique that has been widely used to model time series data is higher-order Markov model (HOMM) or HOMM-Fuzzy model based on Fuzzy set theory introduced by [23]. Markov models related to inventory systems, control charts, and forecasting methods, can be found in [24], [25] and [26]

[27] introduced time-varying and time-invariant fuzzy time series (FTS) models for forecasting. [28] [29], Chen and Chung [30],Ching et al. [31], [32] , [33], and [34] extended the FTS method to higher-order fuzzy time series for improving the forecasting accuracy of the technique.

[35] investigated climate change and obtained forecasts for rainfall patterns in the Manga-Bawku area of Ghana. [36] used 'Effective interval models' of fuzzy time series for forecasting. A special feature of FTS is that the sharing of the conversation universe, say U, has become one of the most important factors in any FTS application, see [37].

[38], [39] proposed a Markov chain (MC) model and according to the order they developed a unique parameter estimation method based on linear programming to achieve their goals.

[40] proposed neural network forecasting accuracy for various time series datasets.

B. Sectional organization

Section 2 covers the basic theoretical components of ARIMA models. Then capacity of HMMs to analyse and forecast time-based events, state emission probability distribution, and starting probability vector is further described. Furthermore, a few distinct subsets of state space models (SSMs) are shown, including recurrent neural networks (RNN), LSTM, and GRU models. The findings attributed to SARIMA on daily gold prices are summarized in Section 3. Then, it also focuses on the HMM model for estimating out-of-time forecasts and discusses how it may be applied to capture non-linear dynamics and emulate an underlying system. A few findings from the examination of gold price data produced by the LSTM and GRU models are presented in this section. Section 4 provides a summary of the findings as well as some areas for further research.

II. THEORETICAL BACKGROUNDS OF ARIMA, HMM, LSTM AND GRU MODELS

A stationary time series ARIMA forecasting model is a linear (i.e., regression-type) equation where the predictors consist of lags of the dependent variable and/or lags of the forecast errors.

The general ARIMA (p, d, q) model for time series $\{y_t, t \in Z\}$ with the backward shift operator B, i.e., $B^j y_t = y_{t-j}$, and white noise ε_t :

$$\left(1 - \sum_{i=1}^{p} \varphi_i \ B^i\right)(1-B)^d \ y_t \ = (1 + \sum_{i=1}^{q} \theta_i \ B^i)\, \epsilon_t \qquad (1)$$

φ_i are the auto regressive (AR) parameters, d is the number of nonseasonal differences needed for stationarity, and θ_i denote the moving average (MA) parameters respectively.

The auto regressive moving average i.e., ARMA (p, q) model results from setting the time lag d=0 in ARIMA (p d q) to remove the trend and make the original time series stationary.A seasonal ARIMA (p, d, q) (P, D, Q) i.e., SARIMA, model for time series $\{y_t, t \in Z\}$ can be written as in **Error! Reference source not found.**:

$$\left(1 - \sum_{i=1}^{p} \varphi_i \ B^i\right)\left(1 - \sum_{i=1}^{P} \emptyset_i \ B^{im}\right)(1-B)^d \ (1-B^m)^d \ y_t \qquad (2)$$

$$= (1 + \sum_{i=1}^{p} \theta_i \ B^i)\ (1 + \sum_{i=1}^{P} \vartheta_i \ B^{im})\epsilon_t$$

where m is the length of the seasonal period, φ_i and Φ_i are the AR parameters of non-seasonal and seasonal parts, respectively, while θ_i and ϑ_i denote the MA parameters of non-seasonal and seasonal parts 'm' in that order. The Box-Jenkins modelling begins with the assumption that the time series can be approximated by an ARIMA model if the series is non-stationary.

A. HMMs can predict and analyse time-based phenomena, how?

The States Spaces Models (SSMs) are traditionally used to model a dynamic system via state variables that constitute a subset of linear invariant (or stationary) portion of the dynamic system. Deterministic versions of SSMs (e.g. LSTMs) proved extremely successful in modelling complex time series data. SSMs can describe a system with input u_t and output y_t in terms of a latent Markovian state x_t. Based on a transition model 'f' and an observation model 'g', as well as process and measurement noise ϵ_t and γ_t, a time-discrete SSM is given by

$x_{t+1} = f(x_t, u_t) + \epsilon_t\ ,\ \ y_t = g(x_t, u_t) + \gamma_t$	*(3)*

A hidden Markov model or HMM, is a type of (SSM) in which the hidden states are discrete, so $x_t \in \{1, \ldots, n_x\}$. The observations may be discrete, $y_t \in \{1, \ldots, n_y\}$, or continuous, y_t or some combination. The hidden state process$\{x_t\}$evolves over time according to a Markov process, possibly conditional on external inputs or controls ut , and each hidden state generates some observations y_t at each time step. We get to see the observations, but not the hidden state. The goal is to infer the hidden state given the observations. However, we can also use the model to predict future observations, by first predicting future hidden states, and then predicting what observations they might generate. By using a hidden state, say z_t (of x_t), to represent the past observations, $y_{1:(t-1)}$, the model can have `infinite'memory, unlike a standard Markov model.

An HMM is a type of recurrent neural network (RNN) where the output from the previous step is fed as input to the current step with the help of a Hidden Layer having its Hidden state, which remembers some information about a sequence. The state is also referred to as Memory State since it remembers the previous input to the network.

We now discuss the basic elements of an HMM. The basic things are now explored to know basic components of HMM and basics how HMM model works and how observation sequence is generated from hidden states.

Hidden Markov Model (HMM) is a method for representing most likely corresponding sequences of observation data. HMM has two parts: hidden and observed. The hidden part consists of hidden states which are not directly observed, their presence is observed by observation symbols that hidden states emits. The transition and emission matrix are the main parameters to build HMM: The transition matrix is a probability of switching from one state to another. Emission matrix is a selection probability of the element in a list.

Denote the stochastic process of hidden states by X(t) that depends on another process Y(t), the observables. Suppose that the hidden state '$x_t = k$' of X(t) occurring at time $t \in [0, T)$ is unobservable for $k \in S = \{1, 2, \ldots, K\}$ and the dynamics of the sequence {X(t))} is governed by a Markov process whose transition probability matrix is $P = (p_{ij})$ such that the elements of each row of the matrix **P** are added to one.

$p_{ij} = Pr\ (x_t = j \mid x_{t-1} = i, x_{t-2}, \ldots, x_1) = Pr\ (x_t = j \mid x = i)$	*(4)*

Let the initial distribution of the state x be denoted by a vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \ \ldots, \pi_K)$.

$\pi_j = Pr\ (x_0 = j)$ *and* $\sum_{j=1}^{K} \pi_j = 1$	*(5)*

B. State emission probability distribution, how?

The hidden and observable parts of this HMM are linked by a state emission probability distribution, which means that each transition between hidden states emits an observable element, m=1, 2, …, M. Furthermore, each hidden state can emit all observables, such that all the emission probabilities of each hidden state add up to 1.

Emission matrix $\mathbf{B} = \{b_s(m)\}$ of size K × M, where $b_x(m)$ is the probability of the hidden state 'emitting the observed element 'm $\in \mathbf{E} = \{1, 2, \ldots, M\}$' of the observable Y(t) process:

$b_x(m) = P(y_t = m \mid x_t = x); x \in \{1, \ldots, K\}, m \in \{1, \ldots, M\}$	*(6)*

Here, the observation at time '*t'* is only dependent on the current hidden state, not on previous hidden states or observations:

$Pr(y_t \mid y_{t-1}, y_{t-2}, \ldots, y_1, x_t, x_{t-1}, x_{t-2}, \ldots, x_1) = Pr(y_t \mid x_t)$	*(7)*

We expect to see observable components at a given time t to be independent of one another. We have now modelled an HMM with sequences of observation elements for the inputs and hidden states for the outputs. The observable series is the set of data that may be immediately apparent, whereas the hidden state series is the underlying method that produces the observations. The probability statements *(1)* through (7) are primarily focusing on the observation elements which are discrete or categorical, the resulting HMM is called the multinomial model, say $\beta = (\boldsymbol{\pi}, \mathbf{P}, \mathbf{B})$. For an ergodic Markov process, the initial distribution can be simply set to the stationary distribution $\boldsymbol{\pi}\,\mathbf{P} = \boldsymbol{\pi}$ and $\boldsymbol{\pi}\,\mathbf{e} = 1$.

There are four special features of an HMM: 1. Likelihood estimation i.e., $Pr(Y_{1:t}=y_{1:t})$, 2. Parameter estimation i.e., $\hat{\beta} = (\hat{\boldsymbol{\pi}}, \hat{\mathbf{P}}, \hat{\mathbf{B}})$, 3. Inference on Hidden states for a given $\{y_{1:t}$ for t=1, 2, …, T} and 4. Prediction of future outcomes. An HMM answers several questions.

Evaluation — how much likely is that something observable will happen? or what is probability of observation sequence? The solution is obtained through either 'Forward algorithm' or Backward algorithm.

Decoding — what is the reason for observation that happened? or what is most probable hidden states sequence when you have observation sequence? It can be answered by 'Viterbi algorithm'.

Learning — what I can learn from observation data? or how to create HMM model or models from observed data? This is answered by 'Baum-Welch' approach.

The hidden states, say x_t, and observation symbols, say y_t, are bind by state emission probability distribution., say $B_{x_t}(y_t)$. This is how every ***transition to hidden state emits observation symbol***. Moreover, every hidden state can emit all observation symbols, only probability of emission one or the other symbol differs. Note that all emission probabilities of each hidden states sum to 1, i.e.,

$\sum_{y_t)} B_{x_t}(y_t) = 1$	*(8)*

When you have hidden states there are two more states that are not directly related to model but used for forward and backward calculations. They are: (i) initial state, and (ii) terminal state.

Likelihood estimation: We now obtain the probability of joint occurrence of an observed sequence $Y_{1:t} = (Y_1=y_1, Y_2=y_2, \ldots, Y_t=y_t)$ for t=1, 2, …, T. The forward-backward algorithm is used to calculate the likelihood of a sequence of observations given the model parameters $\beta = (\boldsymbol{\pi}, \mathbf{P}, \mathbf{B})$ and to estimate the hidden states that generated these observations. We now define the forward probability $\theta(i)$ and backward probability $\emptyset(i)$ for $i \in$ S= {1, 2, …, K}:

$\theta_{t_i}(i) = Pr\ (Y_{1:t}\ , X_{t_i} = s_x/\beta)\ ;\ \emptyset_{t_i}(i) = Pr\ (Y_{(t_i+1):T}/\ X_{t_i} = x_{t_i}, \beta)$	*(9)*

Thus, we see that,

$\theta_1(i) = \pi_i\ b_i(y_1)$*, for i= 1, 2, …, K;* $\theta_{t+1}(i) = \sum_{j=1}^{K} \theta_t(j)\ p_{ij}\ \ b_j(y_{t+1})$*, for t=1, 2…, (T-1)*	*(10)*
$\emptyset_T(i) = 1$*, for i= 1, 2, …, K;* $\emptyset_t(i) = \sum_{j=1}^{K}\ p_{ij}\ \ b_j(y_{t+1})\ \emptyset_{t+1}(j)$*, for t=(T-1),(T-2),…,2,1*	*(11)*

Thus, the likelihood becomes:

$P(Y/\beta)=\sum_{i=1}^{K} \theta_T(i)=\sum_{j=1}^{K} \pi_j\ b_j(y_1)\ \emptyset_1(j)$	*(12)*

Optimizing the most likely state sequence: The Viterbi Algorithm is being used to solve the optimization problem for a given observation sequence $O_{1:t} = (O_1=o_1, O_2=o_2, \ldots, O_t=o_t)$ and a model $\beta = (\boldsymbol{\pi}, \mathbf{P}, \mathbf{B})$.

Maximum likelihood (ML)Specific cases of RNN estimator of $\beta = (\boldsymbol{\pi}, \mathbf{P}, \mathbf{B})$, i.e., $\hat{\beta}= (\hat{\boldsymbol{\pi}}, \hat{\mathbf{P}}, \hat{\mathbf{B}})$:To adjust the model parameters $\boldsymbol{\pi}, \mathbf{P}, \mathbf{B}$ and to maximize $P\ (O/\beta)$ using the given states, we can apply 'Expectation Maximization (EM)' algorithm, (a.k.a. Baum-Welch algorithm)

Prediction of future outcomes: We predict the next output and next hidden state by applying ML method for a given observation sequence and the model.

C. Specific cases of RNN

Use Because RNN or LSTM) models can simulate and predict nonlinear time-variant system dynamics, researchers have been looking into forecasting techniques that use these models. For any kind of time series forecasting issue, there are several LSTM model varieties that may be applied. Additionally, the success of such data-driven strategies based on LSTM and RNN is encouraging.

For their internal processes, known as gates, to determine which data in our time series sequence is crucial to retain or discard, LSTMs and GRUs were developed as the answer to short-term memory problems. For instance, a time series of sufficient length may find it difficult to transfer information from earlier to later time steps. To generate accurate predictions, LSTMs and GRUs can transmit pertinent information down the lengthy chain of sequences. It is widely accepted that lengthy sequence processing is a strong match for both LSTMs and GRUs.

We will present univariate time series forecasting using both the LSTM and GRU models. A function that converts a series of historical observations as input to an output observation will be learned by both approaches. For the LSTM to learn from, the series of observations must be converted into a few examples. Using performance metrics like mean absolute error (MAE), RMSE, and MSE, the major goal is to identify the best performer. The outcomes are presented in a comparison of all employed methods using both numerical and graphical representations.

Vanilla LSTM: The read-write-and-forget concept governs how it operates. Given an input set of data, the network reads and writes the most significant information while disregarding the data that is not crucial for forecasting the result. The RNN introduces three new gates in the following manner to accomplish this.

The size or form of the training data, as well as the input component (samples, timesteps, features), are accessed by RNN. It has an output layer that is used to produce predictions, one hidden layer of LSTM units, and one feature count (for univariate time series). A model using LSTM units in the hidden layer (50, 60, etc.) and an output layer that predicts a single numerical value could be created. We must reshape the single input sample before making the prediction since the model requires the input shape to be three-dimensional with [samples, timesteps, features].

In order to always have many samples in the training dataset, we generated our dataset and passed it as an argument to the 'split_sequence()' method. This number of time steps is known as the input number. After that, the mean squared error, or "mse," loss function is used to improve the model by fitting using the effective Adam form of stochastic gradient descent.

III. RESULTS DUE TO SARIMA ON SALES AMOUNT OF MONTHLY GOLD PRICES

After We analys(ed) the historical gold prices using hmmlearn, downloaded from: https://www.gold.org/goldhub/data/gold-prices. We also calculate the daily change in gold price and restrict the data from 2008 onwards (Lehmann shock and Covid19!). In general dealing with the change in price rather than the actual price itself leads to better modelling of the actual market conditions

The SARIMA time series forecasting method is supported in Python via the Statsmodels library. To use SARIMA there are three steps, they are:

1. Define the model for the training data set.

2. Fit a few candidate models using auto_arima() and select the best with the smallest AIC (Akaike information criterion).

3. Make a prediction for test-data with the model selected as best in step 2.

We must choose the best ARIMA model order after preparing our data for ARIMA modeling. Several criteria, including the Akaike information criterion (AIC) and the Bayesian information criterion (BIC), allow us to choose the best ARIMA model order that minimizes overfitting and underfitting while providing a good match for our data. A model is better if its AIC or BIC is lower. We may use Python or R, or any other computer language, to do this. To obtain a series of plots that display the residuals, autocorrelation, normality, and density of the fitted model, we may also employ the plot diagnostics approach.

Let's try and force an external predictor, called 'exogenous variable' into the model. This model is called the SARIMA model. Let the value of the variable the seasonality period as 12.

We use the command **'auto_arima function to gold price series'** which selects a best SARIMA model as **ARIMA (1,0,0) (2,1,0) [12].** We predicted the forecasts for the recent 453 days of the gold prices considered in the testing portion using the SARIMA ().

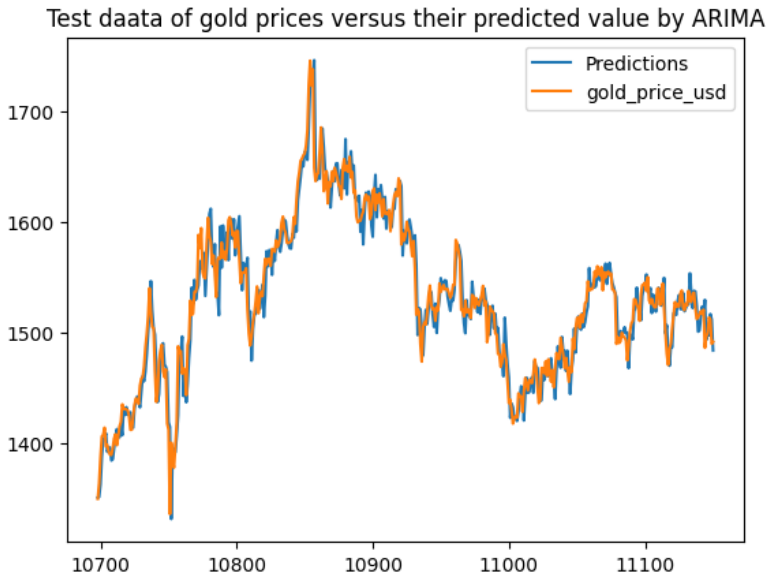

Figure 1: Displays the test data graphs of size 453 together with the projected series from the ***ARIMA (1,0,0) (2,1,0)[12] model that yields*** *root mean squared error as 17.4*

Together with the original time series data, Figure 1 displays the projected values, or the expected values of the time series for the given number of 451 days ahead. By examining the graph, one may judge how effectively the model predicts the time series' future behaviour and determine how large or small the predictions' uncertainty is. The predicted values exhibit the same fluctuations as the actual data. The graphic also highlights some of the ARIMA model's drawbacks and difficulties, such as the assumption that the time series is linear and stationary, which may not hold true for some real-world data.

A. HMM model for finding out-of-time projections Headings

Headings An underlying Markov process or chain with hidden states underlies an HMM model. There are measures y(t) that are observable and directly impacted by these latent states, even if these latent states, let's say x(t), are not immediately apparent to us at time 't'.

The overall picture of a typical HMM, where the states follow one after another, and each one gives rise to an observation:

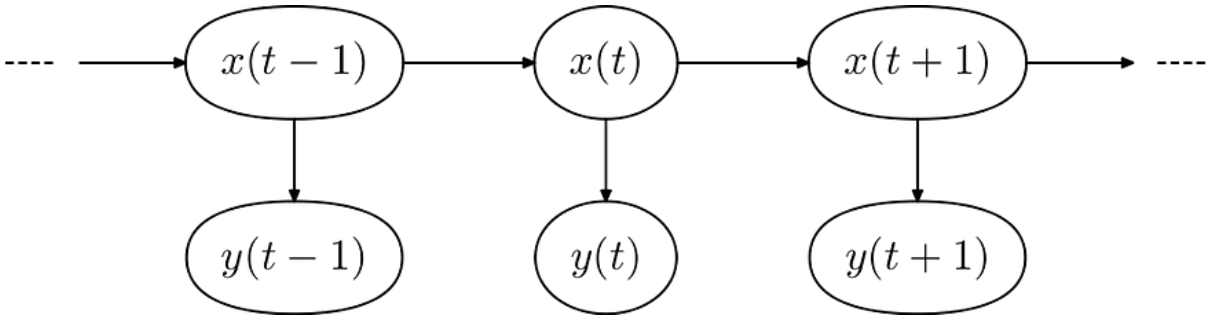

Figure 2: The overall picture of a typical HMM, where the states follow one after another, and each one gives rise to an observation:

Thus, Figure 2 shows the basic structure of a HMM that enables us to capture non-linear dynamics and simulate an underlying system. The basic presumption is that the subsequent observation in the sequence depends solely on the previous one since it is an extension of the Markov model. We can also determine the likelihood of a hidden state transitioning to another (or switching regimes) using an HMM.

An HMM model is often used to sample fresh data, but we can also create forecasts outside of samples by implementing something like to a search space method. Because an HMM model does not have a formula that we can enter in t+1 to get the prediction, using one to forecast is a difficult undertaking. Rather, using an HMM for forecasting may be thought of as a search across a space of observable variables to identify the most likely one.

Said another way, given the prior sequence history and transition probabilities, we determine the prediction Y(t+1) as the result that maximizes the posterior probability of witnessing it. Below are the mathematical formulas for this method: Let θ denote parameters of the HMM. We want to find Y(t+1) such that posterior probability is maximised using the Bayes rule:

$$Y_{t+1} = argmax_{Y_{t+1}} P(Y_{t+1} / Y_t , Y_{t-1} ,,, Y_1, \theta), \quad (13)$$

$$Y_{t+1} = \text{argmax}_{Y_{t+1}} \frac{P(Y_{t+1} = y, Y_t , Y_{t-1} ,,, Y_1, \theta)}{P(Y_t , Y_{t-1} ,,, Y_1, \theta)} \quad (14)$$

Posterior probability simplification: Next, generate a set of potential Y(t+1) values, score them using the fitted HMM model, and select the one that maximizes this score. The procedure is then repeated to forecast Y(t+2) once the Y(t+1) values have been added to the sequence. Thus, the following stages might be used to generalize this process:

1. Determine the group of likely results or contenders.

2. Choose the result that maximizes the posterior probability among the set above.
3. Increase the sequence's highest probability result.

4. Continue

B. Fitting HMM for Out of Time Forecasting Process

To fit the HMM model we will require a data=sequence of observations. We predicted the forecasts for the same 451 items considered in the testing portion using the SARIMA (): **Training and Testing data sets are decided as below:**

Split data into train / test sets

train = data.iloc[:len(data)-453] , test = data.iloc[len(data)-453:]

Fit an HMM on the training set with three state: 0=Low price, 1=medium, and 2=High price

Build the HMM model and fit to the gold price switching patterns.

model2 = hmm.GaussianHMM(n_components = 3, covariance_type = "diag", n_iter = 50, random_state = 42); model2.fit(X),

Predict the hidden states corresponding to test-X.

Z = model.predict(test-X)

Afterwards, use the model's output based on the train data of size 3584 prices to plot the model's state predictions against the data in Figure 2: The overall picture of a typical HMM, where the states follow one after another, and each one gives rise to an observation: Figure 2, we discover that states 0, 1, and 2 seem to correlate to low volatility, medium volatility, and high volatility.

Figure 3: Market volatility as modelled using a Gaussian emissions Hidden Markov Model. Blue/state 0 — low volatility, orange/state 1— medium volatility, green/state 2 — high volatility

From the graphs of Figure 3: Market volatility as modelled using a Gaussian emissions Hidden Markov Model. Blue/state 0 — low volatility, orange/state 1— medium volatility, green/state 2 — high volatility, we find that periods of high volatility correspond to difficult economic times such as the Lehmann shock from 2008 to 2009, the recession of 2011–2012 and the covid pandemic induced recession in 2020. Furthermore, we see that the price of gold tends to rise during times of uncertainty as investors increase their purchases of gold which is seen as a stable and safe asset.

Next, we score each of these outcomes using the fitted HMM for the projected series derived by the ARIMA outlined in the previous section and compute the posterior probability of each of these events occurring next in the sequence.

Next, the outcome that maximizes this likelihood is chosen to determine the anticipated outcome. Following that, the expected outcome is added to the series, and the process is repeated for the next outcome (dynamic forecasting). The transition probability matrix for the three states model using GaussianHMM(n_components=3,..), model.fit(train) and

model.predict(test) commands of Python is estimated as in Table 1s:

Table 1: Transition matrix using the test data of gold prices of 453 days in GaussianHMM(n_components=3,..),

Transition Probability Matrix GaussianHMM(n_components=3,..)			
P=1	0	0.99911318	0.000000000
	0.99068500	0.00931500	0.000000000
	0.00092747	0.00076408	0.99830844

Trader Analysis: From Table 1 transition matrix, we assert that the HMM model indeed yields 3 unique hidden states. These numbers have no intrinsic meaning - which mode corresponds to which system must be confirmed by looking at the model parameters. However, if 0 = low price, 1 = medium price, and 2 = high price states, the one-step conditional transition probabilities that are larger fractions are P(0→1) =0.99911318, P(1→0) =0.990685, and P(2 → 2) = 0.99830844. These can be the best tips for all traders to buy new stocks or sell stocks if any. When the correlation between stocks weakens, stocks that trade at a higher price are to be sold or stocks that trade at a lower price are to be bought.

C. Visualization of the gold price switching patterns

After finding out the most probable outcome, each forecast was calculated, and the corresponding graph is drawn for such daily forecasts for 453 days.

The HMM model yielded a noteworthy projection in the given scenario, indicating a notable shift in sales in recent months or a regime transition characterized by low price/volatility.

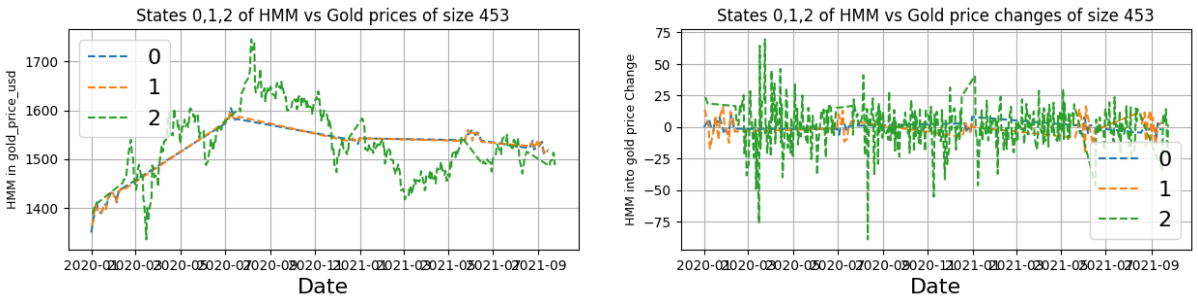

Figure 4: Graphs for predicted values of 451 day's gold prices observed during 2020 -2022 period using ARIMA and HMM models and the stationary gold price change (first difference, top-right) using HMM

Figure 4 makes it evident that HMM was able to predict a low volatility in sales for the previous 451 days with greater sales. Such scenarios were shown to be poorly anticipated by other standard models, such as ARIMA (SARIMA).

Four subplots in Figure 5 illustrate the movements of the three-state Markov process linked to the Gaussian HMM during the last 451 days of the gold price data that are being examined. For the LSTM, GRU, and ARIMA (SARIMA) models, these situations are displayed.

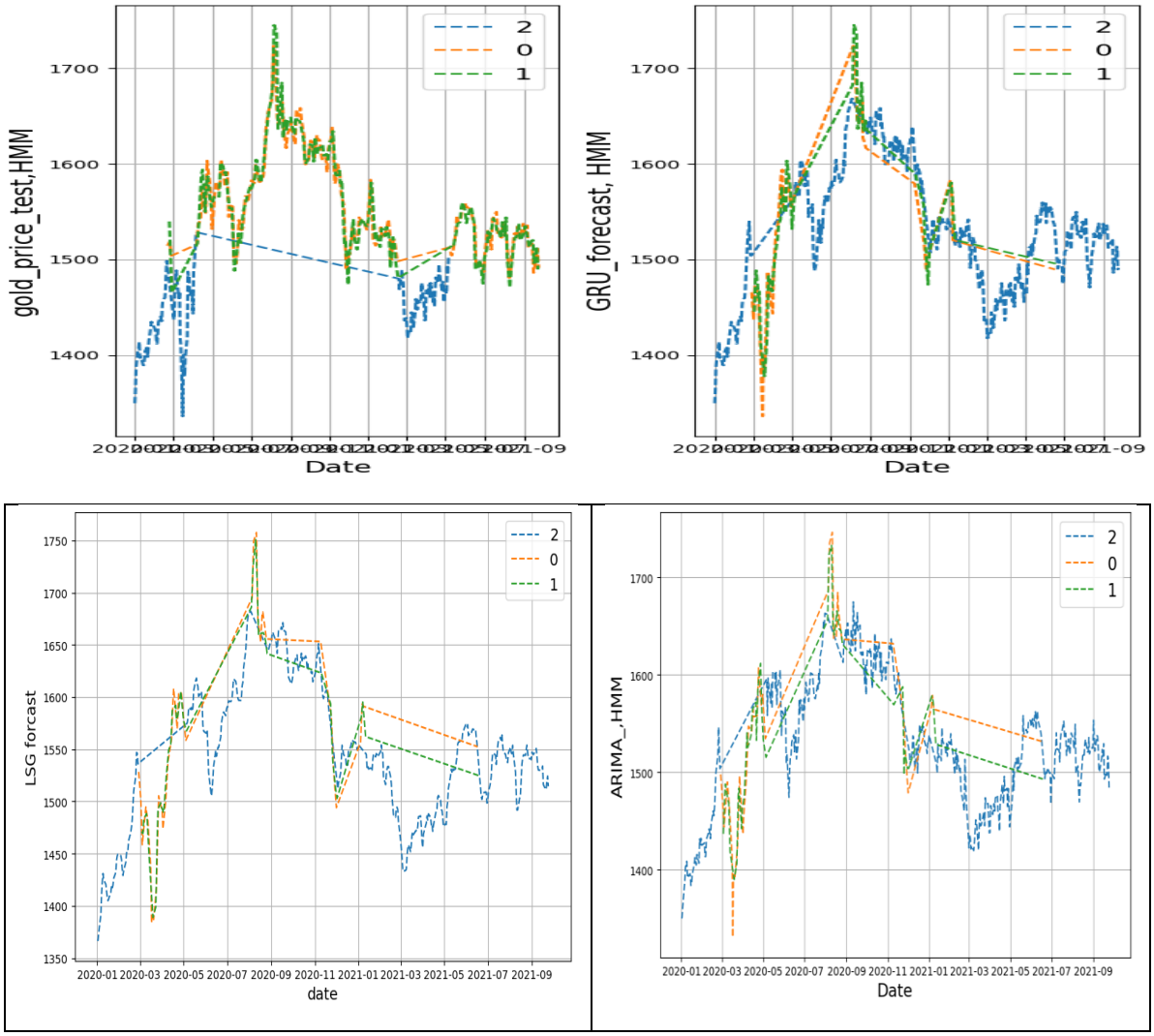

Figure 5: Trasitions of states 0, 1 and 2 of HMM: top left for test data, top right for GRU's prediction, bottom left for LSTM' prediction and bottom right for ARIMA predictions.

Python package: We performed a quick look at a generative probabilistic model called a hidden Markov model, which is also used for analysing sequential data on gold price series. We looked at straightforward research that used the Python package 'hmmlearn' to clearly demonstrate the mathematical operation of hidden Markov models. The three distinct hidden states of the model corresponded to the three potential levels of market volatility.

*D. Analysis of Gold Price series by **LSTM and GRU models***

Now let's review some comparison problems between LSTM and GRU models: LSTMs are a superior option for capturing long-term dependencies in sequences than regular RNNs. GRUs are an LSTM variant with a streamlined gating mechanism. In order to analyse data in parallel, the transformers of both tools shift their emphasis from recurrence to self-attention processes.

In this section, we highlight the results of LSTM and GRU models that were used to analyse the gold price and its forecasts. With a comparable structure and set of hyperparameters, graphs the of Figure 4 and root mean squared error values categorically show that the GRU model outperformed the LSTM.

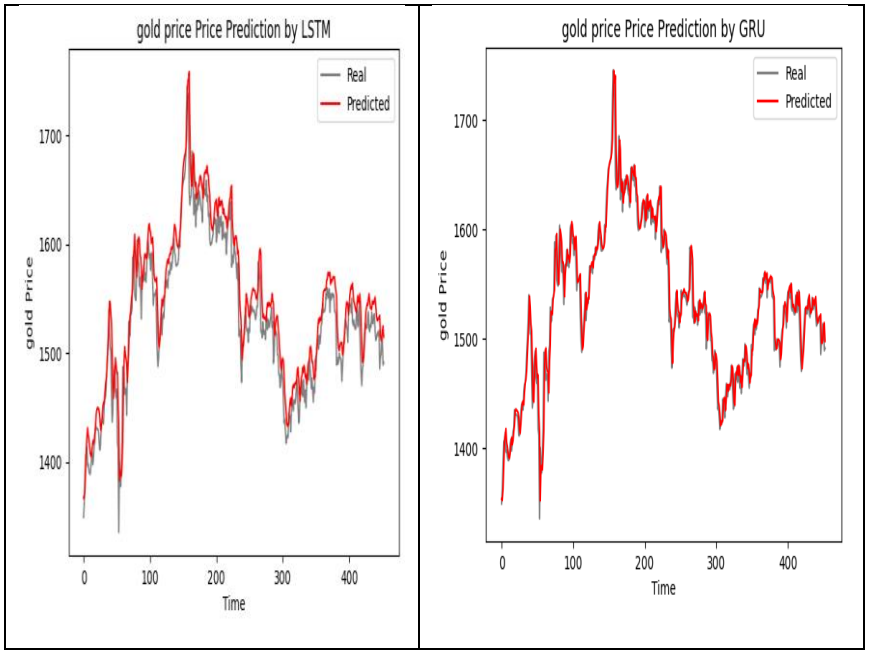

Figure 6 Displays the test data graphs of size 451 together with the projected series from the GRU model (right) and the LSTM model (left). LSTM's root mean squared error is 23.96, whereas GRU's is 16.32.

The test data graphs of size 451 are shown in Figure 6 together with the predicted series from the LSTM model (left) and the GRU model (right). The RMSE for an LSTM is 23.96, whereas that of a GRU is 16.32. The LSTM fared worse than the GRU, as the findings demonstrate.

According to Figure 1, the RMSE of the ARIMA (1,0,0) (2,1,0)[12] model is 17.4, which is less than the RMSE of the LSTM model. As a result, the LSTM model is unstable, ARIMA exceeds LSTM, and GRU surpasses both, making GRU more durable and able to last longer periods of time.

IV. Conclusion

The paper covered the software implementation of two more recurrent neural networks, LSTM and GRU, as well as ARIMA and HMM. A data= historical gold price data covering 11,151 days was divided into two subsets, dubbed train and test, in order to fit the models of each. The ARIMA's predicted series for test section 453, say X, was used for additional follow-ups.

For the same set X of size 453 we made projections using ARIMA. The ARIMA (1,0,0) (2,1,0)[12] model was found to be the best model for the reference model (S)ARIMA, with an RMSE of 17.4.

The "time series forecasting" characteristics of the hidden Markov model, that is named as a type, GaussianHMM(n_components=3...), were then assessed using the Python model.fit(train) and model.predict(test=X) functions, which correspond to a 3-stae Gaussian model. For the time series X obtained for the ARIMA, we gathered many results from the fitted HMMM and calculated the posterior probability of each state that will occur next in the sequence that maximizes this likelihood. Table 1 presents the calculated transition probability matrix for the three states model.

An output layer and a single lSTM/GRU layer with 125 units make up the model structure. We normalize the test set and repeat the preprocessing. The dataset was processed, divided into samples, reshaped, and then we made predictions and inversely turned them into standard form. To fully compare the findings, we simply replaced the LSTM layer with the GRU layer, leaving everything else unchanged.

Figures 1 through 6 visually depict representations of the three-state hidden Markov model (HMM) across the 11,151-day forecasted gold price series. This work therefore establishes the effectiveness of HMM as a model for sequence out-of-time forecasting, especially when a behaviour regime transition may not be readily apparent.

The accuracy of the forecast can be further improved by using a more intricate search space of observation variables or by multiplying the number of variables observed. Concurrently, as the solution is a local optimum

and the model is prone to overfitting, care should be used while fitting HMMs. HMM applications are versatile enough to be employed in modelling dynamic systems and estimating future states based on sequences seen in the future actions.

By reviewing all the results of this article, we now summarise the important outcome as follows: the LSTM model is underperformed, ARIMA performs better than LSTM, but the GRU outperforms over both of ARIMA and LSTM and hence GRU is resilient and can withstand longer periods of time. HMM application can be used for modelling dynamic systems and forecasting future states based on sequences that have been seen because of their flexibility. The GRU model beats both ARIMA and LSTM, indicating that it is more durable and able to endure longer periods of time than the LSTM model, which underperforms.

ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression, “One of us (R.B.G.) thanks . . .” Instead, try “R.B.G. thanks”. Put applicable sponsor acknowle`dgments here; DO NOT place them on the first page of your paper or as a footnote.

REFERENCES

[1] G. Box, G. Jenkins and G. Reinsel, Time Series Analysis, Forecasting and Control, Prentice Hall, 1994.

[2] D. M. M. A. Khan, " Forecasting of Gold Prices (Box Jenkins Approach)," International Journal of Emerging Technology and Advanced Engineering, pp. pp. 662-670., 2013.

[3] M. Khashei and M. Bijari, "An artificial neural network (p, d, q) model for timeseries forecasting," Expert Systems with applications, pp. pp. 479-489., 2010.

[4] J. D. Hamilton, "A new approach to the economic analysis of nonstationary time series and the business cycle.," Econometrica: Journal of the Econometric Society, p. pp. 357–384, 1989.

[5] H. Abbasimehr and M. Shabani, "A new framework for predicting customer behavior in terms of RFM by considering the temporal aspect based on time series techniques," J. Ambient. Intell. Humaniz. Comput, vol. 12, p. 515–531, 2021.

[6] Z. Du, M. Wang and Z. Xu, ".On Estimation of Value-at-Risk with Recurrent Neural Network," in Second International Conference on Artificial Intelligence for Industries (AI4I), 2019.

[7] Y. Bai, E. Yang, B. Han, Y. Yang, J. Li, Y. Mao and T. Liu, "Understanding and improving early stopping for learning with noisy labels.," Advances in Neural Information Processing Systems, 2021.

[8] S. Smyl, G. Dudek and P. Pełka, "ES-dRNN: A Hybrid Exponential Smoothing and Dilated Recurrent Neural Network Model for Short-Term Load Forecasting," IEEE transactions on neural networks and learning systems, 2023.

[9] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proceedings of the IEEE, pp. 77(2), 257–286. doi:10.1109/5.18626., 1989.

[10] B. H. Juang and L. R. Rabiner, "Hidden Markov Models for speech Recognition.," Technometrics, vol. 33, no. 3, pp. 251-272, 1991.

[11] MacDonald and W. Zucchini, Hidden Markov and Other Models for Discrete-Valued, Boca Raton, FL.: CRC Press,, 1997.

[12] R. Durbin, S. Eddy, A. Krogh and G. Mitchison, Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids, Cambridge, UK: Cambridge University Press, 1998.

[13] Z. Ghahramani, "An Introduction to Hidden Markov Models and Bayesian Networks," Int. Journal of Pattern Recognition and Artificial Intelligence, vol. 15, no. 1, pp. 9-42, 2001.

[14] P. Gautam and A. Chandhok, "Switching Behaviour of Subscribers in Indian Telecom Sector," International Journal of Research in Finance and Marketing, vol. 1, no. 3, pp. 183-191, 2011.

[15] Sarwat, A. K. Chandio, S. Shaikh, M. Bhand, B. A. Ghumro and A. K. Khuhro, "Factors Behind Brand Switching in Cellular Networks," International Journal of Asian Social Science, vol. 3, no. 2, pp. 299-307, 2013.

[16] K. Kumaraswamy and N. C. Bhatracharyulu, "Statistical brand switching model: an Hidden Markov approach," OPSEARCH, vol. 60, p. 942–950, 2023.

[17] W. Thupeng, "Statistical Modelling of Annual Maximum Rainfall for Botswana using Extreme Value Theory," International Journal of Applied Mathematics & Statistical Sciences, vol. 8, no. 2, pp. 1-10, 2019.

[18] Lalmuanzuala, N. K. Sathyamoorthy, S. Kokilavani, R. Jagadeeswaran and B. Kannan, "Drought Analysis in Southern Agroclimatic Zone of Tamil Nadu using Standardized Precipitation Index," International Journal of Environment and Climate Change, vol. 12, no. 11, pp. 577-585, 2022.

[19] T. L. Hill, M. Marquez, O'Connor and M. Remus, "Artificial neural network models for forecasting andmdecision making," International Journal of Forecasting, pp. pp. 5-15., 1994.

[20] .. K. C. Luk, J. E. Ball and A. Sharma, "A study of optimal model lag and spatial inputs to artificial neural network for rainfall forecasting (Journal of Hydrology,," Vienna, Austria, pp. pp. 56-65., 2000.

[21] Palmer, J. José Montaño and A. Sesé, "Designing an artificial neural network for forecasting tourism time series," Tourism Management, pp. pp. 781-790, 2006.

[22] H. Aladag, M. A. Basaran, E. E. U. Yolcu and ,. V. R. Uslu, "Forecasting in high order fuzzy time series by using neural networks to define fuzzy relations," Expert Systems with Applications, pp. pp. 4228-4231., 2009.

[23] L. Zadeh, "Fuzzy sets," Information & Control, vol. 8, pp. pp 338-353., 1965.

[24] K. Luk, J. E. Ball and A. Sharma, " A study of optimal model lag and spatial inputs to artificial neural network for rainfall forecasting," Journal of Hydrology Vienna, Austria, , pp. 56-65., 2000.

[25] T. Tanaka, "A partially observable discrete-time Markov decision process with a fractional discounted reward," J. Inform. Optim. Sci. Ser, vol. 38, no. 1, p. 21–37 , 2017.

[26] Sivasamy, R; Molefe, Wilford B, "Markov Models Related to Inventory Systems, Control Charts, and Forecasting Methods," in Markov Model [Working Title], London, SW7 2QJ, UK, IntechOpen, 2023, p. 19.

[27] Q. Song and B. Chissom, "Forecasting enrollments with fuzzy time series – Part II," Fuzzy Sets & Systems, pp. 62 (1), pp 1-8., 1994.

[28] S. M. Chen, "Forecasting enrollments based on fuzzy time series," Fuzzy Sets & Systems, pp. 81 (3), pp 311-319., 1996.

[29] S. Chen, "Forecasting enrollments based on high-order fuzzy time series," Cybernetics & Systems: An International Journal, pp. 33, pp 1-16., 2002..

[30] S. Chen and N. Chung, "Forecasting enrolments using high-order fuzzy time series & genetic algorithms," International Journal of Information & Management Sciences, pp. 21, pp 485-501., 2006.

[31] W. K. Ching, E. S. Fung and M. K. Ng, ". A Multivariate Markov Chain Model for Categorical Data Sequences and its Applications in Demand Predictions," IMA Journal of Management Mathematics, pp. 13, 187-199, 2002.

[32] W. Ching, M. Ng and E. Fung, "Higher-order Multivariate Markov Chains and their Applications," Linear Algebra and its Applications,, pp. 428, 492-507, 2008.

[33] K. Dao Xuan and L. Tuyen, "A Higher order Markov model for time series forecasting," Int. J. Appl. Math. Stat, vol. 57, no. 3, pp. 1-18, 2018.

[34] F. R. &. N. Gharneh, "Forecasting Method Based on High Order Fuzzy Time Series And Simulated Annealing Technique,," South African Journal of Industrial Engineering , pp. Vol 23 (2): pp 176-190, 2012.

[35] Paul, C. Eugene, A. Ebenezer Ebo Yahans, K. Raymond Webrah and A. Edna Pambour, "Analyzing and forecasting rainfall patterns in the Manga-Bawku area, northeastern Ghana: Possible implication of climate change," vol. 5, pp. 1-9, 2021.

[36] K. Huarng, "Heuristic models of fuzzy time series for forecasting," Fuzzy Sets & Systems,, pp. 123 (3), pp 369-386., 2001.

[37] R. Tsaur, J. Yang and H. Wang, "Fuzzy relation analysis in fuzzy time series model,," Computers and Mathematics with Applications,, pp. 49 (4), pp 539-548., 2005.

[38] R. Sivasamy, K. Senthamarai Kannan, Adebayo Fatai and K. M. Karuppusamy, "Higher Order Markov Chain fitting for data on Rain Fall and Rice production," YMER Digital, http:// ymerdigital.com, pp. Issue 10, Vol. 30, pp 156-170, 2021.

[39] J. Sullivan and W. H. Woodall, "A comparison of fuzzy forecasting and Markov modeling," Fuzzy Sets and Systems, p. 64(3):279± 293, 1994.

[40] G. P. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," European Journal of Operational Research, pp. pp. 501-514., 2005.