Hidden Markov models for modeling stock prices

Ideas and applications

Sankarasubramanian Ragunathan

MementoAI

August 21, 2024

Agenda

1. Introduction
   - Problem description
   - Parameters of HMM
2. Baum–Welch algorithm
   - Forward procedure
   - Backward procedure
   - γ and ξ probabilities
   - Update procedure
   - Potential issues
3. Optimal number of hidden states
4. Numerical tests - Problem parameters
   - Problem parameters
   - Training and testing dataset
   - Results
5. Conclusion and Future Scope

Problem description

- There exist observations Y that depend on a latent (or commonly referred to as "hidden") Markov process X.
- The outcomes of the observable process Y depend on the "hidden" states X.
- **Objective:** To learn about the hidden states X given the observation Y.

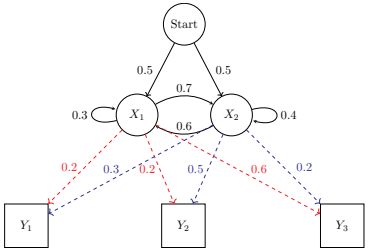


Figure: An example of a hidden Markov model (HMM).

Problem description

Example (HMM for financial modeling)

- **Hidden states:** Bull market X_1, Bear market X_2.
- **Observations:** Open price Y_1, High price Y_2, Low price Y_3, Close price Y_4.
- **Idea:** The behavior of the market (*bullish* or *bearish*) affects the stock price of the company under consideration.
- **Objective:** Find a pattern regarding the how the market nature (given by the hidden states X) affects the stock prices (given by the observations Y)

Question: How to go about figuring out this pattern?

Answer: This is where we utilize HMM to understand the pattern and make future predictions.

Problem description

Definition (hidden Markov models (HMM))

Let $X\colon [0,\infty) \times \Omega \to \mathbb{R}^{d_1}$ and $Y\colon [0,\infty) \times \Omega \to \mathbb{R}^{d_2}$ be discrete-time stochastic processes (where X denotes "hidden" states and Y denotes observations). The tuple of random variables (X_n, Y_n) denotes a *hidden Markov model* (HMM) if X_n is a Markov process that cannot be directly observed, and

$$\mathbb{P}\big(Y_n \in A \,\big|\, X_0 = x_0, \ldots, X_n = x_n\big) = \mathbb{P}\big(Y_n \in A \mid X_n = x_n\big)$$

for all $A \in \mathcal{B}\big(\mathbb{R}^{d_2}\big)$.

Parameters of HMM

The hidden Markov model can be expressed using the following parameters

Parameters	Description
n_S	Total # of hidden Markov states X
n_O	Total # of possible observation states Y
$\pi = \{\pi_i\} = \mathbb{P}(X_0 = i)$	Initial distribution of hidden states
$A = \{a_{ij}\} = \mathbb{P}(X_{n+1} = j \mid X_n = i)$	Transition probability matrix
$B = \{b_j(y_k)\} = \mathbb{P}(Y_{n+1} = y_k \mid X_{n+1} = j)$	Emission probability matrix

- Remark: the transition probability matrix A is a $n_S \times n_S$ matrix, and the emission probability matrix B is a $n_S \times n_O$ matrix.
- The HMM can be described using the parameter $\theta = (\pi, A, B)$.

Problem objective

There are two types of problems that can be studied for use in real-world applications

- **Problem 1:** Given a sequence of observations until some time $T > 0$, i.e. $Y_{\text{obs}} = \{y_n\}_{n=1}^{T}$, we want to compute $\theta^\star$ such that

$$\theta^\star := \arg\max_{\theta} \mathbb{P}\big(Y_{\text{obs}} \mid \theta\big).$$

- **Problem 2:** Given the observation sequence $Y_{\text{obs}} = \{y_n\}_{n=1}^{T}$ and the "optimal" model $\theta^\star$, we want to find the "hidden" states that best explain the observations, i.e. we want to find the "optimal" hidden states $X_{\text{opt}} = \{x_n\}_{n=1}^{T}$ for the observations.

Solution: We make use of the Baum–Welch algorithm to compute $\theta^\star$, and we use the Viterbi algorithm to compute the optimal hidden states X_{opt}.

Baum–Welch algorithm

- **What?** Baum–Welch algorithm is used to obtain the "optimal" parameter $\theta^\star$ by choosing it in such a way that the likelihood of observing Y_{obs} is maximized.
- **Recall:** $\theta^\star = \arg\max_\theta \mathbb{P}\big(Y_{\text{obs}} \mid \theta\big) = \arg\max_\theta \mathbb{P}\big(Y_T = y_T, \ldots, Y_1 = y_1 \mid \theta\big)$.
- **Idea:** We make use of Bayes' theorem to compute the likelihood

$$\mathbb{P}\big(Y_t = y_t, \ldots, Y_1 = y_1 \mid \theta\big) = \sum_{i=1}^{n_S} \underbrace{\mathbb{P}\big(Y_t = y_t, \ldots, Y_1 = y_1, X_t = x_i \mid \theta\big)}_{=:\alpha_i(t)}$$

Baum–Welch algorithm: Forward procedure

$$
\begin{aligned}
\alpha_i(t) &= \mathbb{P}\big(Y_t = y_t, \ldots, Y_1 = y_1, X_t = x_i \mid \theta\big) \\
&= \underbrace{\mathbb{P}\big(Y_t = y_t \mid Y_{t-1} = y_{t-1}, \ldots, Y_1 = y_1, X_t = x_i, \theta\big)}_{= b_i(y_t)} \times \mathbb{P}\big(Y_{t-1} = y_{t-1}, \ldots, Y_1 = y_1, X_t = x_i \mid \theta\big) \\
&= b_i(y_t) \sum_{j=1}^{n_S} \Big\{ \underbrace{\mathbb{P}\big(X_t = x_i \mid X_{t-1} = x_j, Y_{t-1} = y_{t-1}, \ldots, Y_1 = y_1, \theta\big)}_{= a_{ji}} \\
&\qquad \times \underbrace{\mathbb{P}\big(Y_{t-1} = y_{t-1}, \ldots, Y_1 = y_1, X_{t-1} = x_j \mid \theta\big)}_{= \alpha_j(t-1)} \Big\} = b_i(y_t) \sum_{j=1}^{n_S} \alpha_j(t-1) a_{ji}
\end{aligned}
$$

Using this, we can compute $\mathbb{P}\big(Y_{\text{obs}} \mid \theta\big) = \sum_{i=1}^{n_S} \alpha_i(T)$ where

$$\alpha_i(1) = \pi_i b_i(y_1).$$

Baum–Welch algorithm: Backward procedure

The backward procedure to compute $\beta_i(t)$ is given as

$$\beta_i(t) := \mathbb{P}\big(Y_T = y_T, \ldots, Y_{t+1} = y_{t+1} \mid X_t = x_i, \theta\big) = \sum_{j=1}^{n_S} \mathbb{P}\big(Y_T = y_T, \ldots, Y_{t+1} = y_{t+1}, X_{t+1} = x_j \mid X_t = x_i, \theta\big)$$

$$= \sum_{j=1}^{n_S} \mathbb{P}\big(Y_T = y_T, \ldots, Y_{t+1} = y_{t+1} \mid X_{t+1} = x_j, X_t = x_i, \theta\big) \underbrace{\mathbb{P}\big(X_{t+1} = x_j \mid X_t = x_i, \theta\big)}_{=a_{ij}}$$

$$= \sum_{j=1}^{n_S} a_{ij} \underbrace{\mathbb{P}\big(Y_T = y_T, \ldots, Y_{t+2} = y_{t+2} \mid Y_{t+1} = y_{t+1}, X_{t+1} = x_j, \theta\big)}_{=\beta_j(t+1)} \underbrace{\mathbb{P}\big(Y_{t+1} = y_{t+1} \mid X_{t+1} = x_j, \theta\big)}_{=b_j(y_{t+1})}$$

$$= \sum_{j=1}^{n_S} \beta_j(t+1) a_{ij} b_j(y_{t+1}),$$

where $\beta_i(T) = 1$.

Baum–Welch algorithm: γ and ξ probabilities

Using Bayes' theorem, we can compute the terms

$$\begin{aligned}
\gamma_i(t) &:= \mathbb{P}\big(X_t = x_i \mid Y_{\text{obs}}, \theta\big) \\
&= \frac{\mathbb{P}\big(X_t = x_i, Y_{obs} \mid \theta\big)}{\mathbb{P}\big(Y_{obs} \mid \theta\big)} = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{n_S} \alpha_j(t)\beta_j(t)} \\
\xi_{ij}(t) &:= \mathbb{P}\big(X_{t+1} = x_j, X_t = x_i \mid Y_{\text{obs}}, \theta\big) \\
&= \frac{\mathbb{P}\big(X_{t+1} = x_j, X_t = x_i, Y_{\text{obs}} \mid \theta\big)}{\mathbb{P}\big(Y_{\text{obs}} \mid \theta\big)} \\
&= \frac{\alpha_i(t) a_{ij} \beta_j(t+1) b_j(y_{t+1})}{\sum_{k=1}^{n_S} \sum_{\ell=1}^{n_S} \alpha_k(t) a_{k\ell} \beta_\ell(t+1) b_\ell(y_{t+1})}
\end{aligned}$$

The quantitites above help us to obtain $\theta^\star$ from the training data (observations).

Baum–Welch algorithm: Update procedure

- By learning from the training data, we can obtain the "optimal" model parameters $\theta^\star = (\pi^\star, A^\star, B^\star)$ as follows

$$\pi_i^\star = \gamma_i(1), \qquad a_{ij}^\star = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}, \qquad b_i^\star(s_k) = \frac{\sum_{t=1}^{T} \mathbb{1}_{\{y_t = s_k\}} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)},$$

  where s_k denotes a possible observation that can be observed (we refer to this as an observation symbol) for all $k = 1, \ldots, n_O$.
- The function $\mathbb{1}_{\{x \in A\}}$ denotes the *indicator function* and is defined as

$$\mathbb{1}_{\{x \in A\}} \coloneqq \begin{cases} 1, & x \in A \\ 0, & \text{otherwise} \end{cases}, \qquad \forall A \subset \mathcal{B}(\mathbb{R}^d).$$

The above steps are repeated until we reach convergence in the error tolerance ϵ.

Baum–Welch algorithm: Potential issues

1. **Overfitting of model:** It is possible that the model overfits the training data, i.e.

$$\mathbb{P}\big(Y_{\text{obs}} \mid \theta^{\star}\big) > \mathbb{P}\big(Y_{\text{obs}} \mid \theta_{\text{true}}\big)$$

   The algorithm does **NOT** guarantee a global maximum.
2. **Scaling:** Note the terms $\alpha_i(t)$ and $\beta_i(t)$ decays to zero exponentially as t becomes larger. This makes it difficult to implement the algorithm (due to numerical underflow). To remedy this, we make use of the scaling

$$c(t) := \frac{1}{\sum_{i=1}^{n_S} \alpha_i(t)}, \qquad \widehat{\alpha}_i(t) = c(t)\alpha_i(t), \qquad \widehat{\beta}_i(t) = c(t)\beta_i(t)$$

   We then have

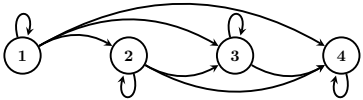
$$\log \mathbb{P}\big(Y_{\text{obs}} \mid \theta\big) = -\sum_{t=1}^{T} \log c(t).$$

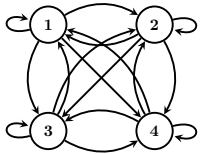
Baum–Welch algorithm: Potential issues

3. **Initial choice for $\theta = (\pi, A, B)$:** Poor initial choice of θ_{initial} can lead to poor prediction of the HMM model. The parameter $\theta^\star$, obtained using θ_{initial}, can correspond to different local maxima than the true value $\implies$ poor model performance.

   Solution: Use random (non-zero and stochastic) or uniform initial values for π and A. For B, it is important to have good initial estimates. To this end, we can use k-means clustering to obtain an idea about the initial emission probabilities.

4. **Choice of HMM:** Depending on how we model the state transitions, we can get different predictions for future observations. Choose a model based on the situation at hand.





Viterbi algorithm

- **What?** Having learned the optimal HMM model parameter $\theta^\star = (\pi^\star, A^\star, B^\star)$, we want to make predictions for the hidden states that best explain the observations.
- **Idea:** We want to make use of the *Viterbi algorithm* to compute the most likely hidden states that maximizes the probability $\mathbb{P}\big(X_{\text{opt}} \mid Y_{\text{obs}}, \theta\big)$.
- This is equivalent to maximizing the probability $\mathbb{P}\big(X_{\text{opt}}, Y_{\text{obs}} \mid \theta\big)$.
- We define the quantity

$$\delta_i(t) \coloneqq \max_{x_1,\ldots,x_{t-1}} \mathbb{P}\big(X_1 = x_1, \ldots, X_t = x_i, Y_1 = y_1, \ldots, Y_t = y_t \mid \theta\big)$$

- From recursion, we have

$$\delta_i(t+1) = b_i(y_{t+1}) \cdot \Big(\max_{1 \le j \le n_S} \delta_t(j) a_{ji} \Big).$$

Viterbi algorithm

- Let ψ denote the sequence (arrays when programming) of the optimal hidden states that maximizes the above probability.
- Initialization procedure:

$$\delta_i(1) = \pi_i b_i(y_1), \qquad \psi_i(1) = 0.$$

- Recursion procedure:

$$\delta_j(t) = b_j(y_t) \cdot \left(\max_{1 \le i \le n_S} \delta_i(t-1) a_{ij} \right),$$
$$\psi_j(t) = \arg \max_{1 \le i \le n_S} \delta_i(t-1) a_{ij}.$$

Viterbi algorithm

- Termination procedure:

$$P^\star = \max_{1 \le i \le n_S} \delta_i(T),$$
$$x_T^\star = \arg \max_{1 \le i \le n_S} \delta_i(T).$$

- Backtracking procedure:

$$x_t^\star = \psi_{x_{t+1}^\star}(t+1).$$

- **Problem:** The Viterbi algorithm for finding the optimal hidden states faces the same numerical underflow problem as the Baum–Welch algorithm.

  Solution: We work with log values instead to prevent the issues of numerical underflow.

Viterbi algorithm - Log scaled

- Initialization procedure:

$$\log \delta_i(1) = \log \pi_i + \log b_i(y_1), \qquad \psi_i(1) = 0.$$

- Recursion procedure:

$$\log \delta_j(t) = \max_{1 \le i \le n_S} \Big(\log b_j(y_t) + \log \delta_i(t-1) + \log a_{ij} \Big),$$
$$\psi_j(t) = \arg \max_{1 \le i \le n_S} \Big(\log \delta_i(t-1) + \log a_{ij} \Big).$$

Viterbi algorithm - Log scaled

- Termination procedure:

$$\log P^\star = \max_{1 \le i \le n_S} \log \delta_i(T),$$
$$x_T^\star = \arg \max_{1 \le i \le n_S} \log \delta_i(T).$$

- Backtracking procedure:

$$x_t^\star = \psi_{x_{t+1}^\star}(t+1).$$

Having computed the optimal hidden states X_{opt} and the optimal model parameters $\theta^\star = (\pi^\star, A^\star, B^\star)$, we can make future predictions for the most likely observations by randomly sampling the associated hidden states and the possible observations.

Optimal number of hidden states

There are two commonly used measures for choosing the "optimal" number of hidden states for the HMM.

$$AIC = -2\log \mathcal{L}(\theta) + 2n_S, \qquad \text{(Akaike Information Criterion)}$$
$$BIC = -2\log \mathcal{L}(\theta) + \log(T)n_S, \qquad \text{(Bayesian Information Criterion)}$$

where $\mathcal{L}(\theta) = \mathbb{P}\left(Y_{\text{obs}} \mid \theta\right)$ is the likelihood of the observations given the HMM parameter θ.

Idea: Compute the information criterion for different number of n_S and T values (design of experiments) and choose that model that has the minimal AIC or BIC value.

Ensure that the HMM model does not overfit to the training data while also ensuring that the patterns in the training data are captured.

Numerical tests - Stock prices

We want to utilize the HMM model to forecast future stock prices of different companies. The stock prices comprise of the following four main data:

1. Opening value (open)
2. High value (high)
3. Low value (low)
4. Closing value (close)

We reframe the 4D observation space to the scaled 3D observation space as follows

$$f_C := \frac{\text{close} - \text{open}}{\text{open}}$$

$$f_H := \frac{\text{high} - \text{open}}{\text{open}}$$

$$f_L := \frac{\text{open} - \text{low}}{\text{open}}$$

Objective: Predict the day's closing value using the day's opening value and the predicted fractional change in the closing value.

We use the following problem parameters:

- # of hidden states,

$$n_S = 4$$

.

- # of possible observations

$$n_O = (n_C + 1) \times (n_H + 1) \times (n_L + 1),$$

where n_C, n_H and n_L denote the number of discretization points for f_C, f_H, and f_L.

$$n_C = 50$$
$$n_H = 20$$
$$n_L = 20$$

- **Reason:** We discretize the observation space so as to allow us compute the probability of observing a certain value for the fractional changes.
- The sequence of possible observations are given as

$$O(i,j,k) = \Big(f_{C,\min} + \frac{i}{n_C}(f_{C,\max} - f_{C,\min}), f_{H,\min} + \frac{j}{n_H}(f_{H,\max} - f_{H,\min}), f_{L,\min} + \frac{k}{n_L}(f_{L,\max} - f_{L,\min})\Big),$$

for all $0 \leq i \leq n_C$, $0 \leq j \leq n_H$, and $0 \leq k \leq n_L$ where

$$\begin{aligned} f_{C,\min} &= -0.1, & f_{C,\max} &= 0.1, \\ f_{H,\min} &= 0.0, & f_{H,\max} &= 0.1, \\ f_{L,\min} &= 0.0, & f_{L,\max} &= 0.1, \end{aligned}$$

- We convert the three-dimensional indices (i,j,k) to one-dimensional indices ℓ, i.e.

$$\ell = k \cdot (n_C + 1) \cdot (n_H + 1) + j \cdot (n_C + 1) + i + 1.$$

- The possible observation sequence is then indexed as $O(\ell)$.

Training and testing dataset

We consider the following stock data to train our HMM. We then compare the predictions generated by our trained HMM against the test data for each stock.

Stock name	Training Data		Testing Data	
Apple Inc.	1st January 2023	1st May 2024	2nd May 2024	1st August 2024
Hyundai Motor Company	1st January 2023	1st May 2024	2nd May 2024	1st August 2024
Bitcoin USD	1st January 2023	1st May 2024	2nd May 2024	1st August 2024
Samsung Electronics Co., Ltd.	1st January 2023	1st May 2024	2nd May 2024	1st August 2024

Table: Stock market data used for training and testing the HMM.

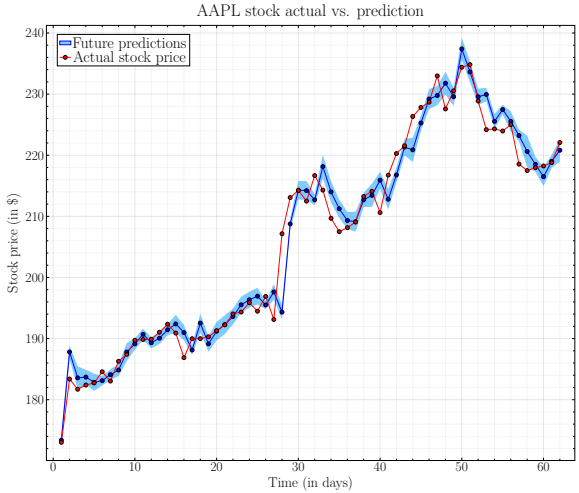
Model initialization

- We initialize the prior probability π and the transition probability matrix A uniformly, i.e.

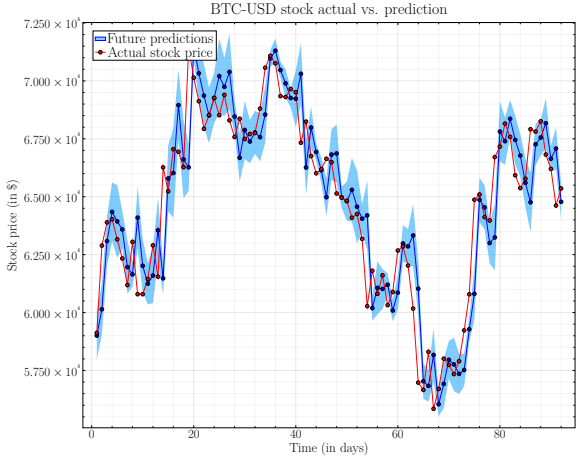
$$\pi = \big(0.25, 0.25, 0.25, 0.25\big), \qquad A = \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix}$$

- The emission probability matrix B is initialized randomly using the Dirichlet distribution.
- We then use the Baum–Welch algorithm to train the HMM and then use the Viterbi algorithm to obtain the optimal hidden states.

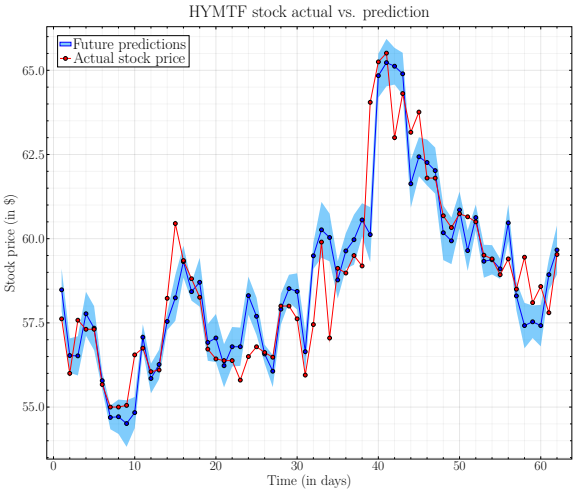
Prediction results



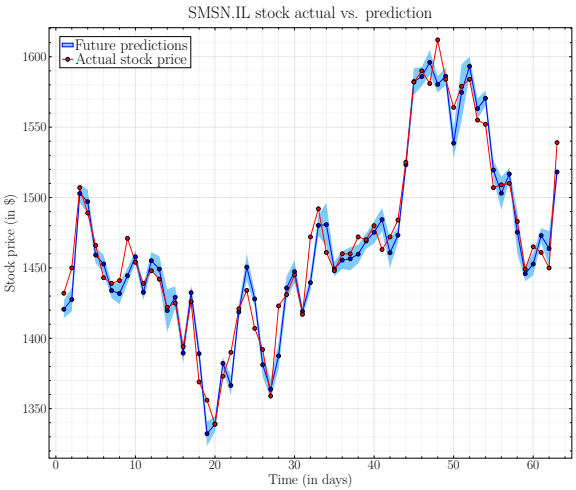
Prediction results



Prediction results



Prediction results



Computational error

We compute the Mean Absolute Percentage Error (MAPE) between the actual data and the prediction made. The MAPE is defined as

$$\text{MAPE} := \frac{1}{N_{\text{test}}} \sum_{k=1}^{N_{\text{test}}} \frac{|\text{prediction}(k) - \text{actual}(k)|}{|\text{actual}(k)|} \times 100,$$

where N_{test} is the number of test data used for the verification of the HMM.

Stock name	**MAPE**
Apple Inc.	1.02467
Hyundai Motor Company	1.25251
Bitcoin USD	1.91255
Samsung Electronics Co. Ltd.	0.727181

Conclusion and Future Scope

- The HMM does not account for correlations between stock prices of different companies in different markets. A possible extension would be to develop a model that captures the correlation between markets.
- Another problem with the HMM is that the possible sequence of observation vectors is discretized. One way to address this issue is to consider continuous time continuous state-space filters (such as the Kalman filter, smoothed particle filters, etc...) for the prediction problem.
- Another possible extension would be to incorporate the possibility of jumps in the stock prices. This would enable the model to capture situations where the stock price can suddenly fall or rise by a large value.

References

Jeff A Bilmes et al.
A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models.
International computer science institute, 4(510):126, 1998.

Luigi Catello, Ludovica Ruggiero, Lucia Schiavone, and Mario Valentino.
Hidden markov models for stock market prediction.
arXiv preprint arXiv:2310.03775, 2023.

Emil Eirola and Amaury Lendasse.
Gaussian mixture models for time series modelling, forecasting, and interpolation.
In *Advances in Intelligent Data Analysis XII: 12th International Symposium, IDA 2013, London, UK, October 17-19, 2013. Proceedings 12*, pages 162–173. Springer, 2013.

References

Aditya Gupta and Bhuwan Dhingra.
Stock market prediction using hidden markov models.
In *2012 Students Conference on Engineering and Systems*, pages 1–4. IEEE, 2012.

Md Rafiul Hassan and Baikunth Nath.
Stock market forecasting using hidden markov model: a new approach.
In *5th international conference on intelligent systems design and applications (ISDA'05)*, pages 192–196. IEEE, 2005.

Md Rafiul Hassan, Baikunth Nath, and Michael Kirley.
A fusion model of hmm, ann and ga for stock market forecasting.
Expert systems with Applications, 33(1):171–180, 2007.

References

Tobias P Mann.
Numerically stable hidden markov model implementation.
An HMM scaling tutorial, pages 1–8, 2006.

Jennifer Pohle, Roland Langrock, Floris M Van Beest, and Niels Martin Schmidt.
Selecting the number of states in hidden markov models: pragmatic solutions illustrated using animal movement.
Journal of Agricultural, Biological and Environmental Statistics, 22:270–293, 2017.

Lawrence R Rabiner.
A tutorial on hidden markov models and selected applications in speech recognition.
Proceedings of the IEEE, 77(2):257–286, 1989.

References

Andreas S Weigend and Shanming Shi.
Predicting daily probability distributions of s&p500 returns.
Journal of Forecasting, 19(4):375–392, 2000.

Yingjian Zhang.
Prediction of financial time series with hidden markov models.
2004.