15th September 2024

MIGe Department

University of study of Trieste

HMM Models for Time-Series Forecasting Stock Market Price

Student:

Gabriel Masella [SM3800048]

Exam project for:

Probabilistic Machine Learning

Table of Contents

1.

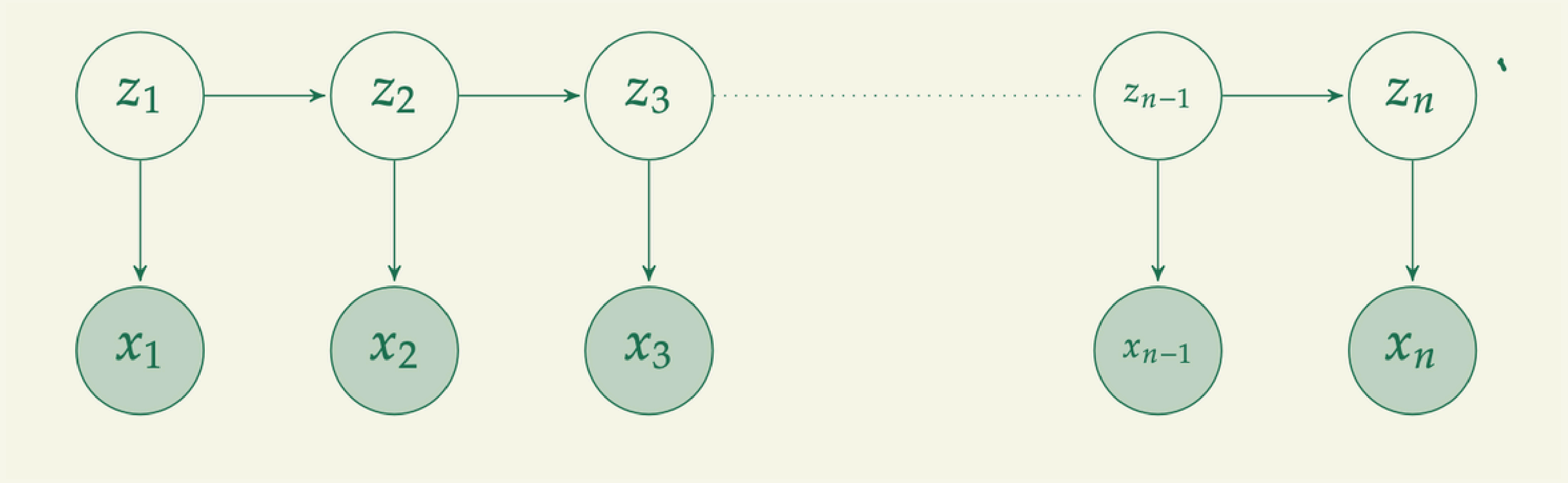
HMM Training Methods

Brief teoretical introduction to all the training techniques used to initialize the initial parameters of the Hidden Markov Model

1.1 EM Algorithm

HMMs are **statistical models** that represent systems with **hidden states** through **observable** sequences, where

$$\begin{cases} x_i = \text{observations} \\ z_i = \text{hidden states} \end{cases}$$



Hidden Markov Models can be described as $\lambda = (A, B, \pi)$ where $\begin{cases} A = \text{transition probability matrix} \\ B = \text{emission probability matrix} \\ \pi = \text{ initial probabilities of the states at } t = 0 \end{cases}$

Training of the above HMM from given sequences of observations is done using the **Baum-Welch algorithm** which uses **Expectation-Maximization (EM)** to arrive at the **optimal parameters** for the HMM:

- **Expectation (E) Step**: Calculate the **expected value of the log-likelihood** with current parameter estimates.
- **Maximization (M) Step: Maximize the expected log-likelihood** to update the parameter estimates.

1.2 K-Means Clustering Algorithm

Another method used in paper [1] to **initialize** the HMMs is based on **K-Means Clustering Algorithm.**

K-Means is an **unsupervised learning** algorithm used to **partition data into K clusters**, where each data point belongs to the cluster with the nearest mean.

It's based on the following steps:

- **Initialization**: Randomly select K initial cluster centroids.
- **Assignment**: Assign each data point to the nearest centroid.
- **Update**: Recalculate the centroids as the mean of all points in each cluster.
- **Repeat**: Iterate the assignment and update steps until convergence.

In this analysis, we will assume the following:

- **Prior and Transition Probabilities**: These are uniformly distributed across all states.
- **Means and Variances**: Each cluster found from K-Means is assumed to be a separate state component.
- **Weights**: Computed as the weights of the clusters, divided equally between the states to obtain the initial emission probabilities.

1.3 Variational Inference

Usually **Second-Order Techniques** are used to estimate parameters of probabilistic models from sample data, treating parameters as random variables with defined distributions.

Variational Bayesian Inference (VI) is a second-order approach that can be viewed as a **variant of the EM algorithm**, used for parameter estimation in probabilistic models.[2]

Advantages:

- **Uncertainty Quantification**: Allows numerical expression of the uncertainty associated with parameter estimation.
- **Incorporation of Prior Knowledg**e: Enables the inclusion of prior knowledge about parameters in the estimation process.
- **Automated Model Selection**: the training process can easily be extended to automate the search for an appropriate number of model components in a mixture density model (e.g., Gaussian mixture model).
- **Training in a Variational Framework:** VI is used to perform HMM training within a variational framework, allowing for more flexible and robust parameter estimation.
- **Univariate Output Distributions**: Typically applied to models with univariate output distributions (i.e., scalar values).

2.

Problem Statement

This section outlines the objective of the project, the data set used to achieve it, and the metrics that will be employed to assess the results.

2. Problem Statement

The **objective** of this project is to **evaluate and compare the performance** of different HMMs initialized in three different ways: **EM Algorithm, K-Means Clustering Algorithm and Variational Inference.**

To this end, a stock market price prediction is executed on **Apple**[AAPL], **NVIDIA**[NVDA] and **Microsoft**[MSFT] stocks from 2020 to 2023 and 2024 as test, knowing **Open and Close prices, High and Low.**

The metric used to evaluate the performance of the algorithm is **Mean Absolute Percentage Error (MAPE)** in accuracy. MAPE is the average absolute error between the actual stock values and the predicted stock values in percentage.

$$\mathrm{MAPE} = \frac{1}{N}\sum_{i=1}^{N}\frac{|pred_i - true_i|}{|true_i|} \times 100\%$$

3.

Implementation

In-depth analysis of the actual process of training and prediction, as well as the methodology behind identifying the optimal model through hyperparametrization tuning.

3.1 Training and Hyperparameter Tuning

Observation set:

The model was trained and tested using the historical daily prices from 2020 of Apple, Nvidia, and Microsoft stock, sourced from publicly available data on Yahoo Finance. These data were then relaborated in order to obtain observations.

Each **observation** in our dataset comprises three distinct values, representing the daily **fractional change, fractional high, and fractional low prices**. [1][3]

$$O_k = \left(\frac{close - open}{open}, \frac{high - open}{open}, \frac{open - low}{open} \right) = (fracChange, fracHigh, fracLow)$$

3.1 Training and Hyperparameter Tuning

Hyperparameter Tuning:

- **Latency**: Tested values of 2, 4, 7, and 14 days.
- **Number of Hidden States**: Ranged from 1 to 7.

Rolling Window Approach:

The training data for the HMM is constructed by using a rolling window approach: each observation sequence spans a fixed duration of x days, we refer to this duration as **latency**.
The window is shifted incrementally along the training period: the first sequence captures observations from the initial time point, while each following sequence incorporates new observations by sliding the window by one day. [3]

BIC and Log-Likelihood

In order to select the optimal model for prediction, **BIC** or **log likelihood** is calculated for each number of hidden states. It should be noted that the three approaches may yield different numbers of hidden states for each latency.

3.2 Prediction with MAP estimation

The array of the **observations** is a **three-dimensional array** consisting of **real values**. Since the probability of guessing any real value is mathematically zero, it was necessary to **discretize** the **observations**.[3]
The number of points used for the discretization are the followings:

Variable	Number of points
fracChange	75
fracHigh	15
fracLow	15

The discretization is made from the minimum and the maximum of **fracChange** in the trainset, and between 0 and maximum for **fracHigh** and **fracLow.**

3.2 Prediction with MAP estimation

Prediction Process:

1. **Historical Observations:**
   - Consider the **latency = x set observations** (so preceding x days).
2. **Sequence Creation:**
   - Append each possible output for the **current day (d)**, creating a **x+1-day sequence** (x historical + 1 potential observation).
   - There are $(n_{fc} \cdot n_{fH} \cdot n_{fL})$ **possibilities** for the **current day**.
3. **Probability Computation:**
   - **Compute** the **probability** for each sequence to be generated from the trained model.
   - Select the observation with the **highest emission probability** as the observation for the next day.

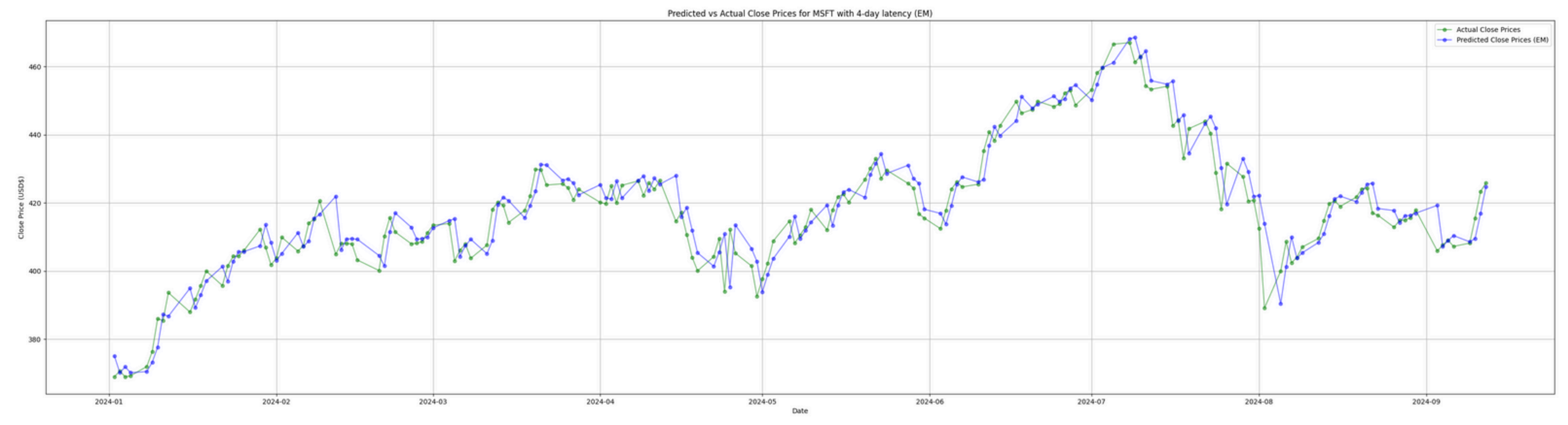
4.

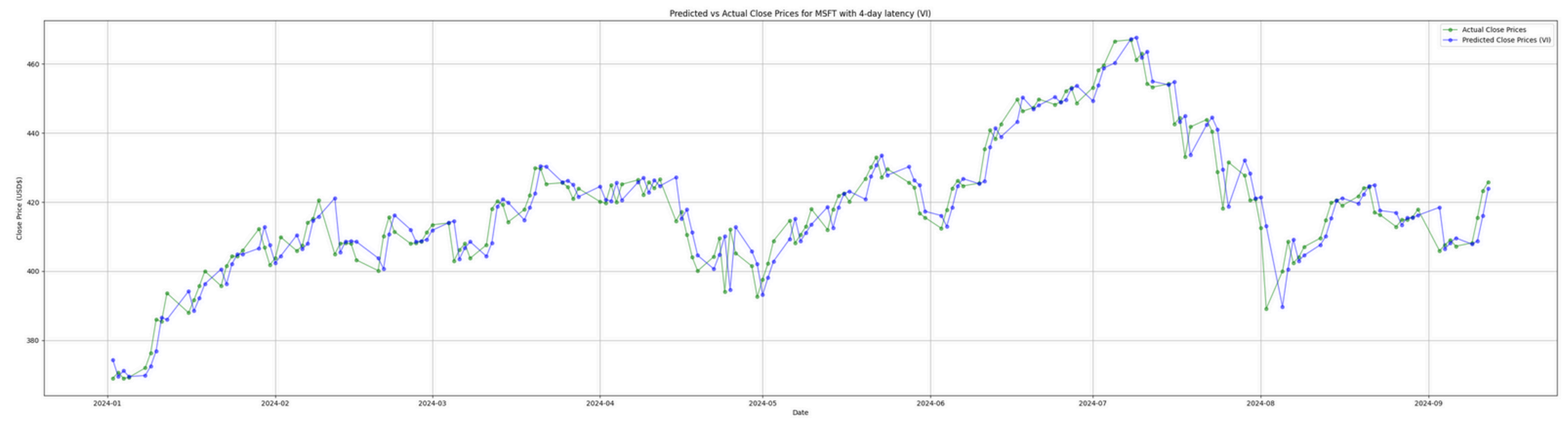
Results

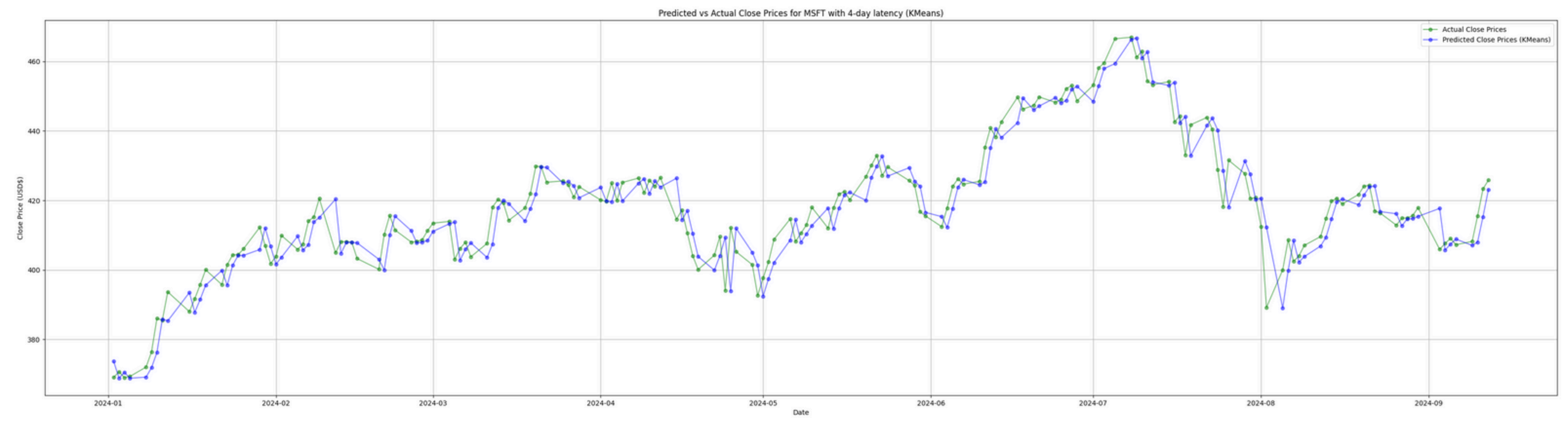
Comparison of **performances** between the training technique through **MAPE Errors**

4.1 Performance Comparison

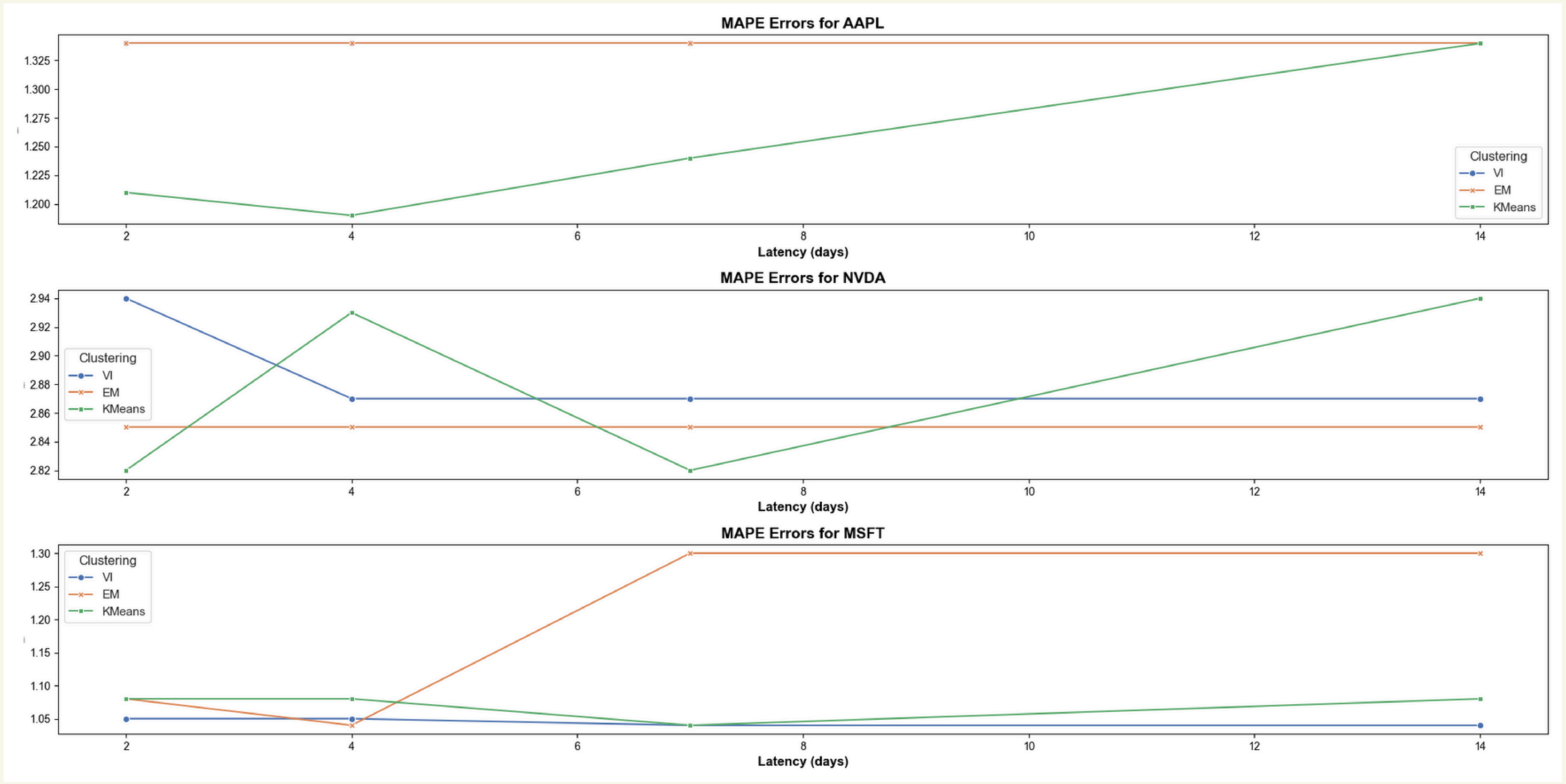
The best







4.1 Performance Comparison



4.1 Performance Comparison

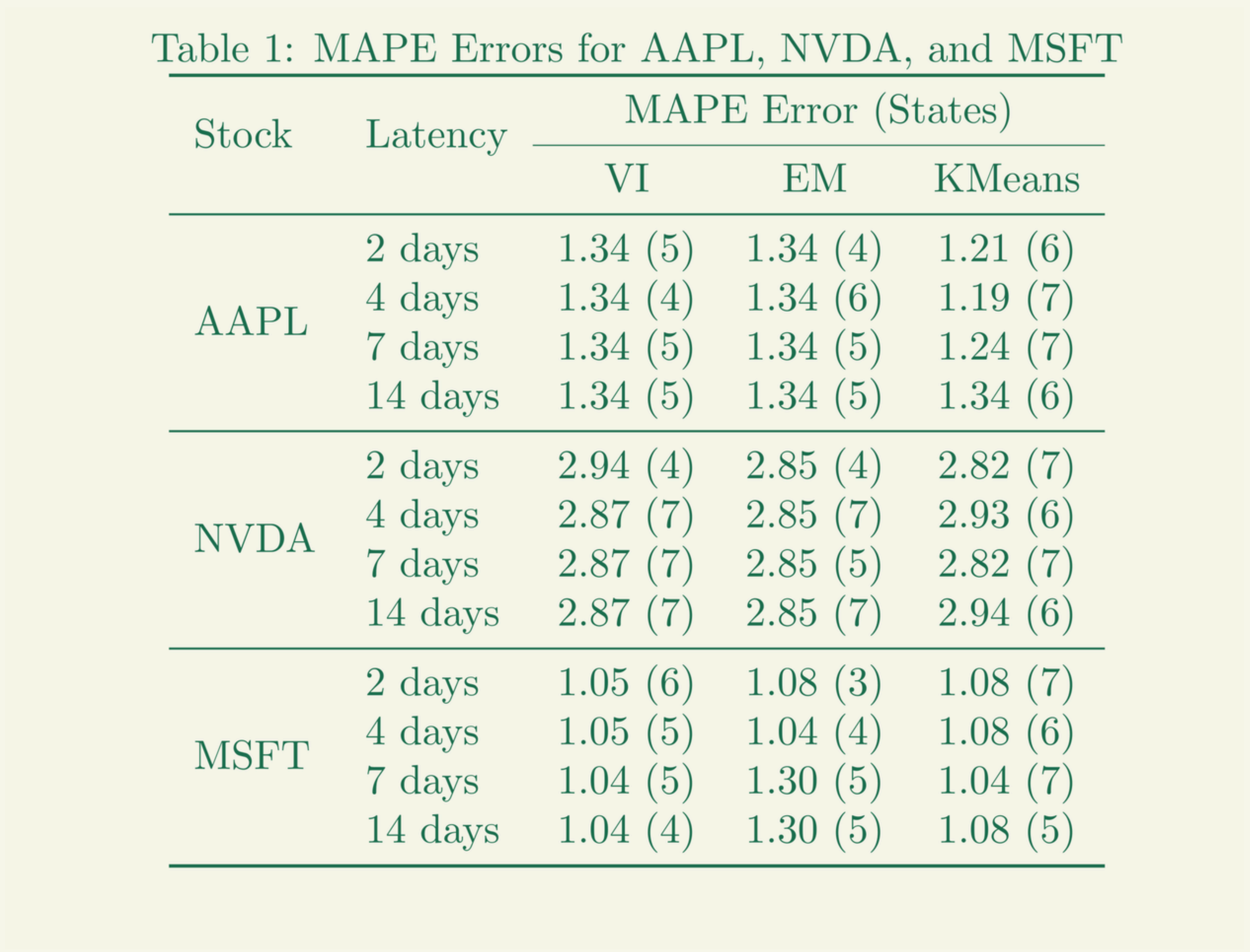


Table 1: MAPE Errors for AAPL, NVDA, and MSFT

Stock	Latency	MAPE Error (States)		
		VI	EM	KMeans
AAPL	2 days	1.34 (5)	1.34 (4)	1.21 (6)
	4 days	1.34 (4)	1.34 (6)	1.19 (7)
	7 days	1.34 (5)	1.34 (5)	1.24 (7)
	14 days	1.34 (5)	1.34 (5)	1.34 (6)
NVDA	2 days	2.94 (4)	2.85 (4)	2.82 (7)
	4 days	2.87 (7)	2.85 (7)	2.93 (6)
	7 days	2.87 (7)	2.85 (5)	2.82 (7)
	14 days	2.87 (7)	2.85 (7)	2.94 (6)
MSFT	2 days	1.05 (6)	1.08 (3)	1.08 (7)
	4 days	1.05 (5)	1.04 (4)	1.08 (6)
	7 days	1.04 (5)	1.30 (5)	1.04 (7)
	14 days	1.04 (4)	1.30 (5)	1.08 (5)

5. Conclusion and Further Improvements

- The **lowest MAPE** is reached for **Microsoft's stock** where **VI (1.04% for 7 or 14 days)** is lower than K-Means and EM, while **NVDA even reaches over 2.8%** with **VI performing slightly better than VI and K-Means** except for the 7 days when K-Means reaches the minimum. **Apple's stock** VI and EM gives equal performance, while K-Means gives the best performance.

- Taking in consideration [1] and [3] results for **APPL** stocks, here **we obtained better results**!

- Overall, it looks like **K-Means Initialization offer better performances**, then Variational Inference and EM Algorithm

- In general, irrespective of the stock **the best latency with the lowest MAPE error on average among the various methods is 4 days**.

- A better approach would be to use a model based on a **mixture of Gaussian distributions** to model the observations. Typically used for continuous data, this method is considered more complex than a single Gaussian distribution.

References

1. A. Gupta and B. Dhingra, **"Stock market prediction using Hidden Markov Models"** 2012 Students Conference on Engineering and Systems, Allahabad, India, 2012, pp. 1-4
2. Christian Gruhl, & Bernhard Sick. (2016). **Variational Bayesian Inference for Hidden Markov Models With Multivariate Gaussian Output Distributions**.
3. Luigi Catello, Ludovica Ruggiero, Lucia Schiavone, & Mario Valentino. (2023). **Hidden Markov Models for Stock Market Prediction.**