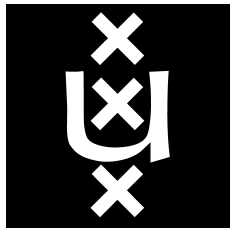
THE SUPERVISED AND UNSUPERVISED LEARNING DILEMMA IN FORECASTING INFLATION

DUC MINH NGUYEN



UNIVERSITY OF AMSTERDAM
SUPERVISOR: TIMO SCHENK
25 JUNE 2024

STATEMENT OF ORIGINALITY

This document is written by Student Duc Minh Nguyen who declares to take full responsibility for the contents of this document. I declare that the text and the work presented in this document are original and that no sources other than those mentioned in the text and its references have been used in creating it. UvA Economics and Business is responsible solely for the supervision of completion of the work and submission, not for the contents.

ABSTRACT

The emergence of machine learning created vast opportunities for predicting macroeconomic variables, including inflation. Supervised and unsupervised learning are the two main competing paradigms in machine learning. In this paper, supervised learning and unsupervised learning forecasting models are compared with each other to find a suitable one for inflation prediction. In total, eleven different models are assembled, tested, and compared. Ex-post analysis shows that the supervised learning algorithms outperform their unsupervised counterparts in terms of forecast error and forecast stability. Specifically, the linear supervised machine learning models - Ridge, OLS linear regression, and LASSO - accomplish this forecasting problem exceptionally well. Nevertheless, it is the specific characteristic of the inflation dataset that ultimately decided the outcome of the battle between supervised and unsupervised learning.

1. INTRODUCTION

Inflation is one of the crucial macroeconomic variables that influence the decisions of economic agents, and predicting inflation has always been an important task for economists. The classical time series models or naïve forecast methods have been widely used for this task throughout history (Stock and Watson, 1999). However, the recent developments in machine learning provided novel approaches for inflation forecasting, and a question then arose whether they outperform the classical econometric models.

When considering machine learning methods for predicting inflation, a question of interest arises whether supervised or unsupervised learning would suit better. These two represent major paradigms in machine learning theory, and both have their advantages and disadvantages.

The goal of this research is to investigate if supervised or unsupervised learning, or their combinations offers superior predictive capabilities of the US inflation rate. Therefore, for comparison the forecast models are divided into four groups: 1) supervised learning models without regularization, 2) supervised learning models with regularization, 3) unsupervised learning models, and 4) classical models. The classical statistical models are included to benchmark against supervised and unsupervised learning algorithms. The group of classical forecast models consists of the univariate ARMA(p, q) and the naive rolling mean forecast, together with the multivariate VAR(p) model. The Feed Forward Neural Network (FNN) and Linear Regression with OLS comprise the group of unregularized supervised learning. The regularized supervised learning group includes the Support Vector Regression (SVR), the Lasso, and the Ridge forecast models. Finally, the unsupervised learning models consist of the k-means clustering model, the Gaussian Hidden Markov Model (GHMM), and the Generative Adversarial Networks (GAN). The forecast horizon for every model is a one-year-ahead forecast, or, since the data is monthly, a twelve-month-ahead forecast. After each model completes their prediction, the combinations of models via equal weights are considered.

This study is designed to answer the following research question: Do supervised learning or unsupervised learning algorithms forecast inflation more accurately?

Literature Review

Araujo and Gaglianone (2023) shed light on machine learning in inflation prediction by investigating the performance of different ML techniques on the forecast of Brazilian inflation and comparing them with classical models such as ARMA and VAR. They discovered that machine learning methods tended to outperform the classical models on noisy and high-dimensional datasets. In the research by Iaousse et al. (2023), it was found that generally, machine learning forecasts, such as support vector regression or LSTM algorithms have better

forecast accuracy than the classical ARIMA models. They argued that the classical statistical models require more assumptions, and tend to underperform in the presence of outliers. However, they also noted that the classical models perform better with univariate time series and on small datasets. Similarly, Reichlin (2009) in his research on US inflation approached the sophisticated machine learning models with caution, pointing out that they do not always outperform the simpler models, such as naïve historical averages.

Regarding the two main paradigms in machine learning, the supervised learning algorithm has an advantage in precision since it is trained on human-labeled data (Nasteski, 2017). However, Jabbar and Khan (2014) noted that it suffers from problems of overfitting when the noise and variation between training and test datasets are significant. In contrast, unsupervised learning methods do not rely on labeled datasets and instead learn based on exploring patterns in the data (Saravanan and Sujatha, 2018), so the risk of overfitting is less. Oral et al. (2012) verified that unsupervised learning outperforms supervised learning on complex and variable datasets. However, unsupervised learning methods can have lower accuracy since they are not trained on labeled data (Alloghani et al., 2020). Amin-Naseri and Soroush (2008) proposed that a combination of supervised and unsupervised learning can lead to superior forecasts.

A classical way to mitigate overfitting in supervised learning is through regularization techniques (Jabbar and Khan, 2014), for instance with LASSO and Ridge (Diebold and Shin, 2019). It provides a powerful tool for supervised learning to compete with its unsupervised counterpart regarding overfitting. This research direction is of particular interest for study.

The existing literature adequately addresses the general trade-off between supervised and unsupervised learning. However, research on this dilemma in the domain of time series forecasting remains limited. Building upon the existing literature, this paper further investigates the trade-off between supervised and unsupervised learning paradigms in the domain of inflation prediction, and especially, this paper explores how regularization techniques can be used for supervised learning to benchmark against unsupervised learning.

2. METHODOLOGY

This section discusses how inflation forecasting is conducted, outlining methodologies, data, and models.

2.1. *Data*

As this research aims to forecast inflation, the data used for this purpose is a time series of CPI-based inflation rate in the United States of America, retrieved from the OECD Library [1].

[1] https://www.oecd-ilibrary.org/economics/data/prices/consumer-prices-complete-database_0f2e8000-en

It is monthly data from January 1978 to December 2023 (552 observations) and is measured in percentage change in consumer prices compared to the previous period.

For multivariate models, auxiliary predictors are added to forecast the inflation series. Firstly, the US unemployment rate [2] is included as a predictor variable, based on the theory of the Phillips curve (Stock and Watson, 1999). Next to that, the variables that represent the supply-side shock are added. The first one is the crude oil price (West Texas Intermediate) [3], as the oil price has always been a crucial determinant of inflation (Renou-Maissant, 2019). Next, the exchange rates are included, namely the Japanese yen to US dollar [4] and US dollar to British pound sterling rates [5]. The last supply-side variable the research uses is the producer price index (PPI)[6], based on all commodities categories in the US. Finally, this paper considers another demand-side predictor variable other than the unemployment rate, which is the consumer sentiment index, based on the survey of the University of Michigan [7]. The higher value of the index indicates that the consumers are more optimistic about the current economic situation, meanwhile, the lower index indicates pessimistic expectations. This predictor is included since consumer sentiment significantly influences inflation, and vice-versa (Juster et al., 1972). From now for the multivariate models, combine all the predictor series and the inflation series into a single 7-dimensional series $\{X_t\}_{t=1}^{552}$:

$$X_t = \Big(\text{Inflation}_t \quad \text{Unemployment}_t \quad \text{Oil}_t \quad \text{YenExch}_t \quad \text{PoundExch}_t \quad \text{PPI}_t \quad \text{ConsumerSent}_t\Big)$$

To conduct forecasting, the inflation series and all the predictors series are split into the training set which includes the observations from January 1978 to February 2006, and the test set which comprises observations from March 2006 to December 2023. The models are fitted in the training set and then are employed to generate the out-of-sample prediction on the test set. The forecast horizon for every forecast technique is 12 months ahead.

2.2. *Classical Models*

Historical Average Forecast

The simplest model in this study is the naive historical average. The forecast is done on a rolling basis and is computed as the mean of the window of size 60 months (5 years). To illustrate, denote the US inflation series as $\{y_t\}_{t=1}^{552}$ and suppose at time t, the historical observations

[2] https://fred.stlouisfed.org/series/UNRATE
[3] https://fred.stlouisfed.org/series/WTISPLC.
[4] https://fred.stlouisfed.org/series/DEXJPUS.
[5] https://fred.stlouisfed.org/series/DEXUSUK.
[6] https://fred.stlouisfed.org/series/PPIACO.
[7] https://fred.stlouisfed.org/series/UMCSENT.

$\{y_k\}_{k=1}^t$ are available. Then the 1-step ahead historical average forecast is:

$$\hat{y}_{t+1} = \frac{1}{60} \sum_{k=t-59}^{t} y_k \tag{1}$$

For the h-step ahead forecast with $h > 1$, the previous forecasts are used to fill in the observations after time t in the window:

$$\hat{y}_{t+h} = \frac{1}{60}\Big(\sum_{k=t-60+h}^{t} y_k + \sum_{i=t+1}^{t+h-1} \hat{y}_i \Big) \tag{2}$$

Simple naive forecast methods such as this can outperform more complicated machine learning algorithms (Reichlin, 2009).

ARMA Model

Assume that $\{y_t\}_{t=1}^{552}$ follows an Autoregressive Moving Average process ARMA(p, q):

$$y_t = \phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i} + \varepsilon_t + \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} \tag{3}$$

Also assume that the white noise process $\{\varepsilon_t\} \overset{\text{i.i.d.}}{\sim} N(\mu, \sigma^2)$. The optimal p and q are chosen in the Analysis section using the Bayesian Information Criterion (BIC) for $p, q = 1, ..., 5$. This ARMA(p, q) model is estimated based on the training sample $\{y_t\}_{t=1}^{337}$ using the Maximum Likelihood Estimation, from which the MLE estimators $\widehat{\phi}_0, \widehat{\phi}_1, \ldots, \widehat{\phi}_p, \widehat{\theta}_1, \ldots, \widehat{\theta}_q, \widehat{\sigma}^2$ are derived.

The estimated parameters are used to generate h-step ahead forecast $\widehat{y}_{t+h}$ recursively:

$$\widehat{y}_{t+h} = \mathbb{E}\big[y_{t+h} \big| y_1, \ldots y_t, \varepsilon_1, \ldots, \varepsilon_t\big] = \widehat{\phi}_0 + \sum_{i=1}^{p} \widehat{\phi}_i y_{t+h-i} + \sum_{j=1}^{q} \widehat{\theta}_j \varepsilon_{t+h-j} \tag{4}$$

for the available data $y_1, \ldots y_t, \varepsilon_1, \ldots, \varepsilon_t$ at time t. For the forecast horizon of 12 months, $\widehat{y}_{t+h}$ is computed for each $h = 1, ..., 12$ in the test sample. After each 12-step prediction, the previous 12 historical data points are added to the training sample, and the MLE estimation is performed again to generate future forecasts.

VAR Model

The Vector Autoregressive model VAR(p) is considered in this paper to simultaneously forecast both the inflation and the predictors. The reason is that some of the selected predictors such as unemployment and consumer sentiment index can potentially be endogenous (Juster

et al., 1972), so it is safer to model them simultaneously with inflation.

Assume the combined series $\{X_t\}_{t=1}^{552}$ follows a VAR(p) process:

$$X_t = \boldsymbol{\alpha} + \sum_{i=1}^{p} \boldsymbol{\Phi}_i X_{t-i} + \boldsymbol{\varepsilon}_t \tag{5}$$

with $\boldsymbol{\varepsilon}_t \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \Omega_\varepsilon)$. The value p is again determined with BIC. Based on the training sample the parameter estimates $\widehat{\boldsymbol{\alpha}}$ and $\{\widehat{\boldsymbol{\Phi}}_i\}_{i=1}^{p}$ are computed with MLE. The h-step ahead prediction is then found recursively:

$$\widehat{X}_{t+h} = \widehat{\boldsymbol{\alpha}} + \sum_{i=1}^{p} \widehat{\boldsymbol{\Phi}}_i \widehat{X}_{t+h-i} \tag{6}$$

Similar to the case of the ARMA model, the most recent 12 historical data points are appended to the training sample after each 12-step ahead forecast.

2.3. *Unregularized Supervised Learning Models*

Feed Forward Neural Network

The first machine learning forecast method considered in this paper is the Feed-Forward Neural Network (FFN). It is a classical 2-layer Artificial Neural Network with the structure described in Figure 1.

The input data for FFN training is obtained by dividing the training inflation dataset $\{y_t\}_{t=1}^{337}$, from January 1978 to February 2006, into windows of size 24 months $W_k = \{y_k, ..., y_{k+23}\}$ for $k = 1, 2, ..., 302$. The target values for the FFN training are the next 12 months after W_k, i.e. the model learns how to predict twelve months in the future given W_k. Therefore, the input-target pairs $(W_1, \{y_t\}_{t=25}^{36}), ..., (W_{302}, \{y_t\}_{t=326}^{337})$ form the training dataset for the FFN. With these, train the neural network by feeding the windows W_k into the input layer. Since the dimension of W_k is 24, this will be the number of neurons in the input layer. Then, an affine transformation is performed in the data in the input layer:

$$z_k^{(1)} = A^{(1)} W_k + b^{(1)} \tag{7}$$

where $A^{(1)}$ is a $m \times 24$ constant matrix, m is the number of neurons in the hidden layer, and $b^{(1)}$ is the bias vector. The number of neurons m is a tuning parameter: a too small m leads to underfitting, while too large m overfits the model. Next, $z_k^{(1)}$ is fed into the hidden layer, which

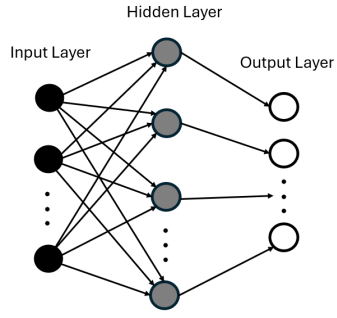


FIGURE 1.— 2-Layer Feed Forward Neural Network

then the data go through the Leaky Rectified Linear Unit (LeakyReLu) activation function:

$$LeakyReLu(x) = \begin{cases} x\,, & \text{if } x \geq 0 \\ 0.01x\,, & \text{if } x < 0 \end{cases} \tag{8}$$

Finally, the affine transformation is applied again to generate the output $\{\widehat{y}_t\}_{t=k+1}^{k+12}$ in the output layer. The estimation $\{\widehat{y}_t\}_{t=k+1}^{k+12}$ is then benchmarked against the real target $\{y_t\}_{t=k+1}^{k+12}$ so that the neural network updates its parameters with backpropagation. Once the FFN model is trained, it can make out-of-sample predictions 12 steps ahead of the inflation test data set from March 2006 to December 2023.

Linear Regression with Ordinary Least Squares

Linear regression is one of the simplest machine learning algorithms and the ordinary least squares (OLS) is its widely used estimation method. The training data for the linear regression, similar to the case of FFN, is obtained by dividing the training inflation dataset $\{y_t\}_{t=1}^{337}$ into windows of size 24 months $W_k = \{y_k, ..., y_{k+23}\}$ for $k = 1, 2, ..., 313$. But this time the target values are different from FFN: instead of the observations of 12 the months after, only a single future month value y_{k+24} is considered. Then, the training data in terms of input-target pairs is $(W_1, y_{25}), ..., (W_{313}, y_{337})$, so that the linear regression model is:

$$y_{k+24} = \beta_0 + W_k\boldsymbol{\beta} \quad \text{for } \; k = 1, 2, ..., 313 \tag{9}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{24})$ is the coefficient vector and W_k is represented as a row vector. The coefficients $\boldsymbol{\beta}$ and β_0 are estimated by minimizing the residual sum of squares (RSS) which yields the estimated $\widehat{\boldsymbol{\beta}}_{ols}$ and $\widehat{\beta}_0$ (Heij et al., 2004). Then, the one-step-ahead prediction at time t is equal to:

$$\widehat{y}_{t+1} = \widehat{\beta}_0 + W_{t-23}\widehat{\boldsymbol{\beta}}_{ols} \tag{10}$$

For the h-steps ahead forecast with $h > 1$, move the window $h-1$ steps ahead by appending the previous predictions to the window and repeat the OLS estimation again.

2.4. *Regularized Supervised Learning Models*

For all three regularized supervised learning models, the same training data set as for OLS is used, i.e. the window-target pairs $(W_1, y_{25}), ..., (W_{313}, y_{337})$.

Support Vector Regression

The Support Vector Regression (SVR) is one of the main kernel learning algorithms (Tung and Wong, 2009). Support Vector Regression generally works as follows:
let $(\mathbf{x_1}, Y_1), ..., (\mathbf{x_n}, Y_n) \in \mathbb{R}^m \times \mathbb{R}$ be a given data set of pairs of explanatory and target variables. The SVR aims to estimate targets $\{Y_i\}_{i=1}^n$ by finding a function $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ such that the error $|Y_i - f(x_i)|$ is at most ε (> 0). The SVR transforms the input data $\mathbf{x_i}$ to a feature space $\mathcal{F}$ (possibly higher-dimensional), via a mapping $\phi : \mathbb{R}^n \longrightarrow \mathcal{F}$. Then in this feature space, the relationship between $\mathbf{x_i}$ and Y_i is modeled linearly. The function f describes this whole process, and has the following form:

$$f(\mathbf{x}) = \langle w, \phi(\mathbf{x}) \rangle + b \tag{11}$$

where $\langle \cdot, \cdot \rangle$ is the inner product operator on $\mathcal{F}$, $w \in \mathcal{F}$ is a weight vector, and $b \in \mathbb{R}$ is the bias value. The weight vector w is chosen to be minimal, and such that the deviation between Y_i and $f(\mathbf{x_i})$ is at most ε. This leads to the following optimization problem (Guajardo et al., 2006):

$$\min_{w,b,\xi_i,\xi_i^*} \ \frac{1}{2}\|w\| + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \tag{12}$$

$$\text{s.t. } Y_i - \langle w,\ \phi(\mathbf{x_i}) \rangle - b \leq \varepsilon + \xi_i \quad \text{for } i = 1, ..., n \tag{13}$$

$$\langle w,\ \phi(\mathbf{x_i}) \rangle + b - Y_i \leq \varepsilon + \xi_i^* \quad \text{for } i = 1, ..., n \tag{14}$$

$$\xi_i, \xi_i^* \geq 0 \quad \text{for } i = 1, ..., n \tag{15}$$

where ξ_i, ξ_i^* are the added slack variables to ensure the feasibility of the solution, and C is the regularization penalty parameter aimed at reducing overfitting. In this study, C is chosen based on K-fold cross-validation. Let $\{\lambda_i\}_{i=1}^n$ and $\{\lambda_i^*\}_{i=1}^n$ be the Lagrange multipliers of the first and second constraint respectively. Then, by Guajardo et al. (2006), the solution to this optimization problem gives the estimate of Y_i in terms of function f:

$$\hat{Y}_i = f(\mathbf{x}_i) = \sum_{j=1}^{n}(\lambda_j - \lambda_j^*)\langle \phi(\mathbf{x}_j), \phi(\mathbf{x}_i) \rangle + b = \sum_{j=1}^{n}(\lambda_j - \lambda_j^*)K(x_j, x_i) + b \tag{16}$$

The mapping ϕ is generally not known, so instead the inner product $\langle \phi(\mathbf{x}_j), \phi(\mathbf{x}_i) \rangle$ is replaced with the so-called kernel function $K(x_j, x_i)$. The kernel function used in this study is the Radial Basis Function (RBF) kernel:

$$K(x_j, x_i) = \exp\Big(- \frac{\|\mathbf{x}_j - \mathbf{x}_i\|}{2} \Big) \tag{17}$$

where $\| \cdot \|$ is the l^2 norm. To forecast inflation with SVR, consider the input-target pairs $(W_1, y_{25}), ..., (W_{313}, y_{337})$ for SVR training. The SVR model is fitted on these data pairs and then is used to make out-of-sample predictions on a rolling basis. The forecast horizon is again 12 months, and the h-step ahead forecast for $h > 1$ is given by:

$$\hat{y}_{t+h} = f\Big(\{ y_{t+h-24}, ..., \ y_t, \ \hat{y}_{t+1}, ..., \ \hat{y}_{t+h-1} \} \Big) \tag{18}$$

LASSO model

Both LASSO and Ridge tries to estimate the coefficients of $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_{24})$ of a linear model (Diebold and Shin, 2019) similar to OLS. The Lasso regression aims to minimize the following:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{313} \Big(y_{i+24} - \beta_0 - \sum_{j=1}^{24} \beta_j W_{ji} \Big)^2 + \lambda \|\boldsymbol{\beta}\|_1 \tag{19}$$

where W_{ji} is the j-th element of window i, $\|\boldsymbol{\beta}\|_1$ is the l^1 norm of the coefficient vector $\boldsymbol{\beta}$, and λ is the penalty coefficient. The LASSO eliminates the insignificant observations in each window by reducing the corresponding β_j coefficient close to 0, avoiding overfitting in this way. Note that if $\lambda = 0$ then this minimization problem is equivalent to that of OLS. The estimated LASSO coefficients are utilized to linearly predict the observations in the test data set on a rolling basis:

$$\widehat{y}_{k+24} = \widehat{\beta}_0^L + \sum_{j=1}^{24} \widehat{\beta}_j^L W_{jk} \tag{20}$$

for $k \geq 314$.

Ridge model

The Ridge regression minimizes the same quantity as Lasso, except for the difference in the penalty term:

$$\min_{\boldsymbol{\beta}} \sum_{i=1}^{313} \Big(y_{i+24} - \beta_0 - \sum_{j=1}^{24} \beta_j W_{ji} \Big)^2 + \lambda \|\boldsymbol{\beta}\|_2 \tag{21}$$

Instead of l^1-norm, the Ridge considers the l^2-norm of $\boldsymbol{\beta}$. The essential difference now is that the Ridge does not eliminate separate variables like Lasso, but jointly shrinks the effect of all variables by reducing the value of all β_j. One computed the coefficient estimates, the out-of-sample prediction of the Ridge model is similar to Lasso:

$$\widehat{y}_{k+24} = \widehat{\beta}_0^R + \sum_{j=1}^{24} \widehat{\beta}_j^R W_{jk} \quad \text{for } k \geq 314 \tag{22}$$

2.5. *Unsupervised Learning Models*

The forecasting approach of unsupervised learning algorithms differs from that of supervised ones. The unsupervised model explores the underlying structure in the time series and then uses this to simulate new datasets that act as a continuation of the time series. Essentially, forecasting with unsupervised learning is a task of generating synthetic data.

K-means Clustering

The k-means is an unsupervised learning clustering algorithm that aims to split the dataset into k clusters and then assign new data points to these clusters. The number of clusters k is a hyperparameter, which is determined in this research using the Elbow method (Ketchen Jr. and Shook, 1996). The inflation training dataset $\{y_t\}_{t=1}^{337}$ is split into k clusters $\mathbf{C} = \{C_1, ..., C_k\}$ such that the sum of squared distances between points and the cluster's mean (centroid) in each cluster is the smallest:

$$\mathbf{C} = \underset{\{C_1,\ldots,C_k\}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{y \in C_i} \|y - \mu_i\|^2 = \underset{\{C_1,\ldots,C_k\}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{y \in C_i} \left\| y - \frac{1}{|C_i|} \sum_{x \in C_i} x \right\|^2 \tag{23}$$

where $\|\cdot\|$ is the l^2-norm. After obtaining the optimal cluster set $\mathbf{C}$, at any time point t take the window of the past 24 inflation data $W_t = \{y_{t-23}, \ldots, y_t\}$. Then for each data point in W_t classify to which cluster in $\mathbf{C}$ it likely belongs to. This is done by assigning the data point to the cluster C_i for which the l^2 distance between the centroid μ_i and the data point is minimal. After obtaining the cluster labeling of all points in W_t, determine the cluster label $\widetilde{i}$ that is the most frequently repeated in W_t. Then for all data points in W_t with label $\widetilde{i}$, compute their mean $\overline{y}_{\widetilde{i}}$ and variance $s_{\widetilde{i}}^2$. Then, the 12-step forecast is a vector of 12 random simulations from the normal distribution with this mean and variance:

$$\widehat{y}_{t+h} \overset{\text{i.i.d.}}{\sim} N(\overline{y}_{\widetilde{i}},\ s_{\widetilde{i}}^2) \quad \text{for } h = 1, ..., 12 \tag{24}$$

Gaussian Hidden Markov Model

Given a sequence of observations $\{o_t\}_{t=1}^T$ in discrete time, the Gaussian Hidden Markov Model (GHMM) assumes that there is a Markov Chain $\{S_t\}_{t=1}^T$, not directly observed, that

generates these $\{o_t\}_{t=1}^T$ via a (multivariate) normal distributions. The state-space of the Markov Chain $\{S_t\}_{t=1}^T$ consists of K states $\{1, ..., K\}$, and let A and π be the transition matrix and initial state distribution of this chain respectively (Rabiner and Juang, 1986). The number of states K is a hyperparameter to be optimized, using BIC. The GHMM assumes that each state $k \in \{1, ..., K\}$ corresponds to a distinct (multivariate) normal distribution $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. The process of generating the sequence $\{o_t\}_{t=1}^T$ works as follows: if the state at time t is $k \in \{1, ..., K\}$, i.e. $S_t = k$, then the observation o_t is generated as a random draw from $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, or formally:

$$O_t | S_t = k \ \sim \ \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{25}$$

where O_t is the random variable of the possible observations at time t. The way that the chain $\{S_t\}_{t=1}^T$ emits observations $\{o_t\}_{t=1}^T$ is modeled with a $K \times T$ emission matrix B with elements $(B)_{k,t} = \big(f_k(o_t)\big)_{k,t}$, where f_k is the density function of $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$. Combining these, the parameters vector of a Gaussian Hidden Markov Chain is $\boldsymbol{\theta} = \Big(A, B, \pi, \boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, ..., \boldsymbol{\Sigma}_k\Big)$, so that $\{o_t\}_{t=1}^T \sim \text{GHMM}(\boldsymbol{\theta})$.

To forecast inflation, the combined series $\{X_t\}_{t=1}^{552}$ is considered as the observed sequence in the GHMM. Then, at any time point t consider the window of the last 24 months $W_t = \{y_{t-23}, ..., y_t\}$, on which the parameters of the hidden Markov chain $\{S_{t-23}, ..., S_t\}$ are estimated. This is done by the so-called Baum-Welch algorithm (Catello et al., 2023) that maximizes the likelihood function of the GHMM to simultaneously find the estimates $\widehat{\boldsymbol{\theta}} = \Big(\widehat{A}, \widehat{B}, \widehat{\pi}, \widehat{\boldsymbol{\mu}}_1, ..., \widehat{\boldsymbol{\mu}}_K, \widehat{\boldsymbol{\Sigma}}_1, ..., \widehat{\boldsymbol{\Sigma}}_k\Big)$.

Then, forecasting with GHMM is essentially a simulation task. To predict the next 12 steps $\{X_{t+h}\}_{h=1}^{12}$, firstly, use the estimated $\widehat{\pi}$ and $\widehat{A}$ to compute the most likely state k at time $t+1$, i.e. $\widehat{S}_{t+1} = k$. Then simulate a value x_{t+1} from the corresponding multivariate normal $\mathcal{N}(\widehat{\boldsymbol{\mu}}_k, \widehat{\boldsymbol{\Sigma}}_k)$. Repeat this process to generate the next 11 time steps and store all the 12 values in a vector $\mathbf{x}^i = \{x_{t+1}, \dots x_{t+12}\}$. Repeat this whole process 100 times to generate 100 potential 12-step ahead forecasts $\{\mathbf{x}^i\}_{i=1}^{100}$. To choose from the simulated values the one that will be the next forecast, compute for each $\mathbf{x}^i$ the probability that it is generated from the estimated Gaussian Hidden Markov Model of the window W_t, i.e the probability $\mathbb{P}\Big(\{X_{t+h}\}_{h=1}^{12} = \mathbf{x}_i \ \big| \ W_t \sim \text{GHMM}(\widehat{\boldsymbol{\theta}})\Big)$ (in Bayesian statistics this is known as the Likelihood probability). This probability is calculated with the so-called forward algorithm (Catello et al., 2023). A detailed explanation of the forward algorithm can be found in the Appendix. Then, the 12-step prediction $\{\widehat{X}_{t+h}\}_{h=1}^{12}$ is equal to the simulated value $\mathbf{x}^j$ that maximizes its likelihood probability:

$$\{\widehat{X}_{t+h}\}_{h=1}^{12} = \underset{\mathbf{x}^j \in \{\mathbf{x}^i\}_{i=1}^{100}}{\text{argmax}} \ \mathbb{P}\Big(\{X_{t+h}\}_{h=1}^{12} = \mathbf{x}^j \ \big| \ W_t \sim \text{GHMM}(\widehat{\boldsymbol{\theta}})\Big) \tag{26}$$

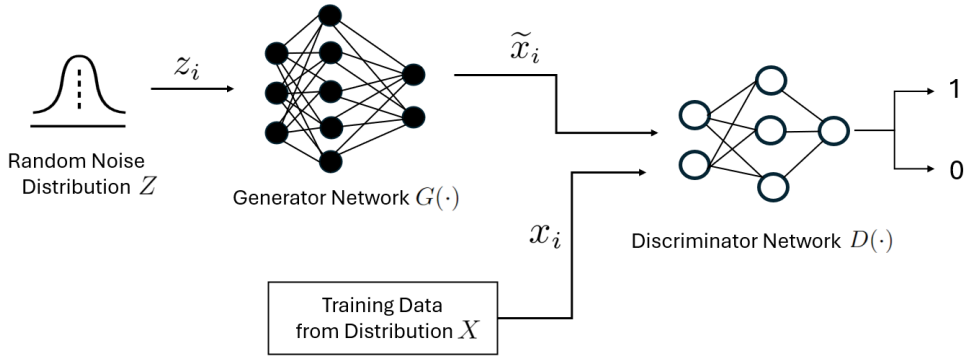


FIGURE 2.—Generative Adversarial Network Diagram

Generative Adversarial Networks

The Generative Adversarial Network (GAN) is a state-of-the-art neural network architecture (Saxena and Cao, 2022) with two competing neural networks: the Generator Network and the Discriminator Network. The application of this model in time series forecasting has already been developed (Labiad et al., 2023), however, it is still in its infancy. The training procedure of GAN is illustrated in Figure 2.

Let $\{x_i\}_{i=1}^n \subset \mathbb{R}^m$ be a given training dataset. The Generator Network is represented by a mapping $G : \mathbb{R}^d \longrightarrow \mathbb{R}^m$, where the inputs z_i are randomly drawn from a noise distribution Z. The output of the Generator Network is then $G(z_i)$, which has the same dimension m as the training data. Denote the outputs to be $\widetilde{X}_i := G(z_i)$. The Discriminator Network is a classifier model that aims to distinguish between the real data from the training sample and the generated data from the Generator Network. Represent the Discriminator Network by a mapping $D : \mathbb{R}^m \longrightarrow \{0, 1\}$, which tries to assign value 1 to the real data, $D(x_i) = 1$, and value 0 to the simulated data, $D(\widetilde{x}_i) = 0$. Real and generated data are fed into the Discriminator Network to train it to be able to classify. In the training of GAN, the Generator Network tries to simulate data so close to the real one that the Discriminator Network classifies it as 1, meanwhile, the Discriminator Network aims to detect as much generated data from the Generator Network as possible. By Wang (2020), this process is modeled via a zero-sum game with the cost function $V(G, D)$ such that the optimal solution is found by:

$$\min_G \max_D V(G, D) = \min_G \max_D \Big\{ \mathbb{E}\Big[\log D(X) \Big] + \mathbb{E}\Big[\log(1 - D(G(Z)) \Big] \Big\} \tag{27}$$

where X is the distribution of the real data. This optimization problem is solved in the training process with the gradient method: by ascending the gradient of V with respect to the parame-

ters of D and descending the gradient with respect to the parameters of G.

In this study, the GAN model serves as a tool to generate new inflation series, which essentially acts as the forecast values. For this purpose, at any time t consider the window of the past 36 months $W_t = \{y_{t-35}, ..., y_t\}$. Then, divide this window into smaller windows of size 12: $\{y_i, ..., y_{11+i}\}$ for $i = t-35,\ ...,\ t-11$. These smaller windows act as the training data for the GAN model. For both the Generator and Discriminator Networks, the 2-layer FFNs are used. It is also decided to include LeakyReLU activation functions in the hidden and output layers of the generator network, meanwhile, the ReLU activation function is used in the hidden layer of the discriminator network and the sigmoid activation function in its output layer. The random noise distribution Z in this research is the n-dimensional multivariate standard normal distribution $\mathcal{N}\big(\mathbf{0}, \mathbf{I}\big)$. The dimension n of the noise distribution is a hyperparameter to be chosen. Then, the GAN model is trained to yield the estimated Generator $\widehat{G}$ and Discriminator $\widehat{D}$.

Next, the forecast procedure of GAN is similar to that of the Gaussian Hidden Markov Model. Using the estimated Generator $\widehat{G}$, simulate a set of 100 synthetic time series $\{\widehat{\mathbf{y}}_i\}_{i=1}^{100}$ each of size 12 months. Then each of the simulated series is fed into the estimated discriminator $\widehat{D}$ so that any series that is labeled 0 by $\widehat{D}$ is eliminated. From the survived simulated series, choose as the forecast the one with the smallest absolute mean difference from the mean of the previous historical 12 months:

$$\{\widehat{y}_{t+h}\}_{h=1}^{12} = \underset{\widehat{\mathbf{y}}_k \in \{\widehat{\mathbf{y}}_i\}_{i=1}^{100}}{\operatorname{argmin}} \left\{ \left| \overline{\widehat{\mathbf{y}}_k} - \overline{\{y_{t-11}, \ldots, y_t\}} \right| \;\middle|\; \widehat{D}(\widehat{\mathbf{y}}_k) = 1 \right\} \tag{28}$$

After this 12-step prediction is finished, move to the next window W_{t+12} and fit a new GAN model again on W_{t+12} to perform the next forecast.

2.6. *Forecast Combination and Evaluation*

Following the forecasting with each model, the study proceeds to investigate the forecast combinations of models, using equal average weights. According to Diebold and Shin (2019) the equal-weight combinations perform well under quadratic error functions, which is the case with this paper. The models are combined within categories: unregularized supervised, regularized supervised, all supervised learning, and unsupervised combinations. These combinations allow for further investigation of the difference in performance of supervised and unsupervised learning as it allows to look into how models simultaneously predict inflation.

At the end of the research, the forecasts of all the models and combinations are evaluated,

using the Mean Squared Error (MSE) for each predicted series $\{\widehat{y}_{it}\}_{t=338}^{552}$ on the test data:

$$\text{MSE}_i = \frac{1}{215}\sum_{t=338}^{552}(y_t - \widehat{y}_{it})^2 \tag{29}$$

The MSE is only calculated on the test sample since this paper aims to evaluate the out-of-sample performance of the models. The MSE is chosen because the combined forecasts in this paper use equal weights, which works well with the square loss functions such as MSE (Diebold and Shin, 2019). The MSEs are compared between the four categories of models to answer the research question.

When the MSEs are calculated, it is of interest to determine whether the differences in errors between the predictions are statistically significant, or it is simply due to the specifics of the research design. For this purpose, the Diebold-Mariano test (Diebold and Mariano, 1995) is performed for each pair of forecast models. The Diebold-Mariano (DM) test works as follows, let $\{\widehat{f}_t\}_{t=338}^{552}$ and $\{\widehat{g}_t\}_{t=338}^{552}$ be the two competing predictions of $\{y_t\}_{t=338}^{552}$. For each t compute the squared errors $e^2_{\widehat{f}t} = (y_t - \widehat{f}_t)^2$ and $e^2_{\widehat{g}t} = (y_t - \widehat{g}_t)^2$. Then the hypotheses of the Diebold-Mariano test are:

$$H_0: \ \mathbb{E}\big[e^2_{\widehat{f}t} - e^2_{\widehat{g}t}\big] = 0 \qquad \text{vs} \qquad H_1: \ \mathbb{E}\big[e^2_{\widehat{f}t} - e^2_{\widehat{g}t}\big] \neq 0 \tag{30}$$

The null hypothesis states that the difference in the observed MSE of $\{\widehat{f}_t\}_{t=338}^{552}$ and $\{\widehat{g}_t\}_{t=338}^{552}$ is not a consequence of the actual difference between the two forecasts. Meanwhile, the alternative hypothesis conjectures that one of the models is actually more precise. For each pair of models, the p-value of their Diebold-Mariano test on the out-of-sample forecast is reported.

3. RESULTS

The research results are discussed and analyzed in this section. Firstly, the chosen hyperparameter values of the models are presented. Then, the out-of-sample forecast results of supervised learning, unsupervised learning, their combinations, and classical models are reported and compared.

3.1. *Hyperparameter Tuning*

The hyperparameters for the ARMA(p, q) model are chosen based on BIC to be $p = 5$ and $q = 5$. While for the VAR model, the BIC optimal lag order is $p = 2$. For the two-layer feed-forward neural network, 200 neurons are constructed for the hidden layer. For Support Vector Regression, the K-fold cross-validation with $K = 10$ yields the hyperparameter value of $C = 200$. Using the same $K = 10$, the K-fold cross-validation yields the penalty rate for Lasso with $\lambda = 0.3$ and for Ridge with $\lambda = 0.3$. For the k-means, using the Elbow method, the optimal

Model	MSE
Classical Models	
Historical Average	4.130877
ARMA(5, 5)	2.510464
VAR(2)	5.447538
Unregularized Supervised Learning Models	
Feed Forward Neural Network	2.753743
Linear Regression with OLS	1.723043
Regularized Supervised Learning Models	
Support Vector Regression	4.539227
LASSO	2.227984
Ridge	1.713232
Unsupervised Learning Models	
k-means	6.009907
Gaussian Hidden Markov Model	5.233943
Generative Adversarial Network	6.975622
Combinations of Models	
Unregularized Supervised Learning Combination	1.990055
Regularized Supervised Learning Combination	2.133633
All Supervised Learning Combination	2.028813
Unsupervised Learning Combination	4.772801

TABLE I

MEAN SQUARED ERROR OF OUT-OF-SAMPLE FORECASTS

number of clusters is graphically determined to be $k = 3$. Finally, the number of states in the hidden chain of the Gaussian Hidden Markov Model is chosen to be $K = 5$ based on BIC, meanwhile, for the GAN model the noise dimension is taken to be 12 and the number of neurons in the hidden layer of the generator and discriminator are 120 and 30 respectively.

3.2. *Forecast Results*

The out-of-sample forecast performance of every model and combinations of models are measured in terms of MSE and are presented in Table I. The statistical significance of MSE differences between forecast pairs is displayed in Table II in terms of the p-values of the Diebold-Mariano test. The best performing model turns out to be the Ridge regression with the lowest MSE of 1.71, followed closely by OLS Linear Regression with an MSE of 1.72. The MSE-optimal combination of models belongs to the combination of unregularized supervised learning models with an MSE of 1.99. Overall, the supervised learning models have lower forecast errors than the unsupervised learning models.

Figure 3 displays the inflation forecast results of supervised and unsupervised learning in two separate plots. The supervised learning plot illustrates that the predictions from linear models — Ridge, OLS Linear Regression, and LASSO — closely resemble the actual inflation.

Forecast Pair	**p-value**
Historical Average - Support Vector Regression	0.54281
Historical Average - Linear Regression with OLS	0.00001 ***
Historical Average - Feed Forward Neural Network	0.01544 ***
Historical Average - LASSO	0.00007 ***
Historical Average - Ridge	0.00000 ***
Historical Average - K-means	0.00001 ***
Historical Average - Gaussian Hidden Markov Model	0.04870 ***
Historical Average - Generative Adversarial Network	0.00000 ***
ARMA - Support Vector Regression	0.00011 ***
ARMA - Linear Regression with OLS	0.00001 ***
ARMA - Feed Forward Neural Network	0.46573
ARMA - LASSO	0.10415
ARMA - Ridge	0.00001 ***
ARMA - K-means	0.00001 ***
ARMA - Gaussian Hidden Markov Model	0.00000 ***
ARMA - Generative Adversarial Network	0.00000 ***
VAR - Support Vector Regression	0.17025
VAR - Linear Regression with OLS	0.00000 ***
VAR - Feed Forward Neural Network	0.00027 ***
VAR - LASSO	0.00000 ***
VAR - Ridge	0.00000 ***
VAR - K-means	0.57023
VAR - Gaussian Hidden Markov Model	0.78008
VAR - Generative Adversarial Network	0.12022
Support Vector Regression - Feed Forward Neural Network	0.00001 ***
Support Vector Regression - LASSO	0.00000 ***
Support Vector Regression - Ridge	0.00000 ***
Support Vector Regression - K-means	0.09522
Support Vector Regression - Gaussian Hidden Markov Model	0.30681
Support Vector Regression - Generative Adversarial Network	0.00904 ***
Linear Regression with OLS - Support Vector Regression	0.00000 ***
Linear Regression with OLS - Feed Forward Neural Network	0.00031 ***
Linear Regression with OLS - LASSO	0.00002 ***
Linear Regression with OLS - Ridge	0.01330 ***
Linear Regression with OLS - K-means	0.00000 ***
Linear Regression with OLS - Gaussian Hidden Markov Model	0.00000 ***
Linear Regression with OLS - Generative Adversarial Network	0.00000 ***
Feed Forward Neural Network - LASSO	0.05418
Feed Forward Neural Network - Ridge	0.00026 ***
Feed Forward Neural Network - K-means	0.00005 ***
Feed Forward Neural Network - Gaussian Hidden Markov Model	0.00001 ***
Feed Forward Neural Network - Generative Adversarial Network	0.00000 ***
LASSO - Ridge	0.00001 ***
LASSO - K-means	0.00000 ***
LASSO - Gaussian Hidden Markov Model	0.00000 ***
LASSO - Generative Adversarial Network	0.00000 ***
Ridge - K-means	0.00000 ***
Ridge - Gaussian Hidden Markov Model	0.00000 ***
Ridge - Generative Adversarial Network	0.00000 ***
K-means - Gaussian Hidden Markov Model	0.24766
K-means - Generative Adversarial Network	0.11578
Gaussian Hidden Markov Model - Generative Adversarial Network	0.01971 ***
Unregularized Supervised Learning Combination - Unsupervised Learning Combination	0.00000 ***
Regularized Supervised Learning Combination - Unsupervised Learning Combination	0.00000 ***
All Supervised Learning Combination - Unsupervised Learning Combination	0.00000 ***
	*** — p-value < 0.05

TABLE II

P-VALUES OF THE DIEBOLD-MARIANO TESTS FOR FORECAST PAIRS

Indeed these models occupy the top three most MSE optimal forecasts among the individual models in Table I. Notably, Ridge and OLS generate smooth forecasts that capture the micro-fluctuations in the real data. In contrast, the LASSO model's predictions appear more

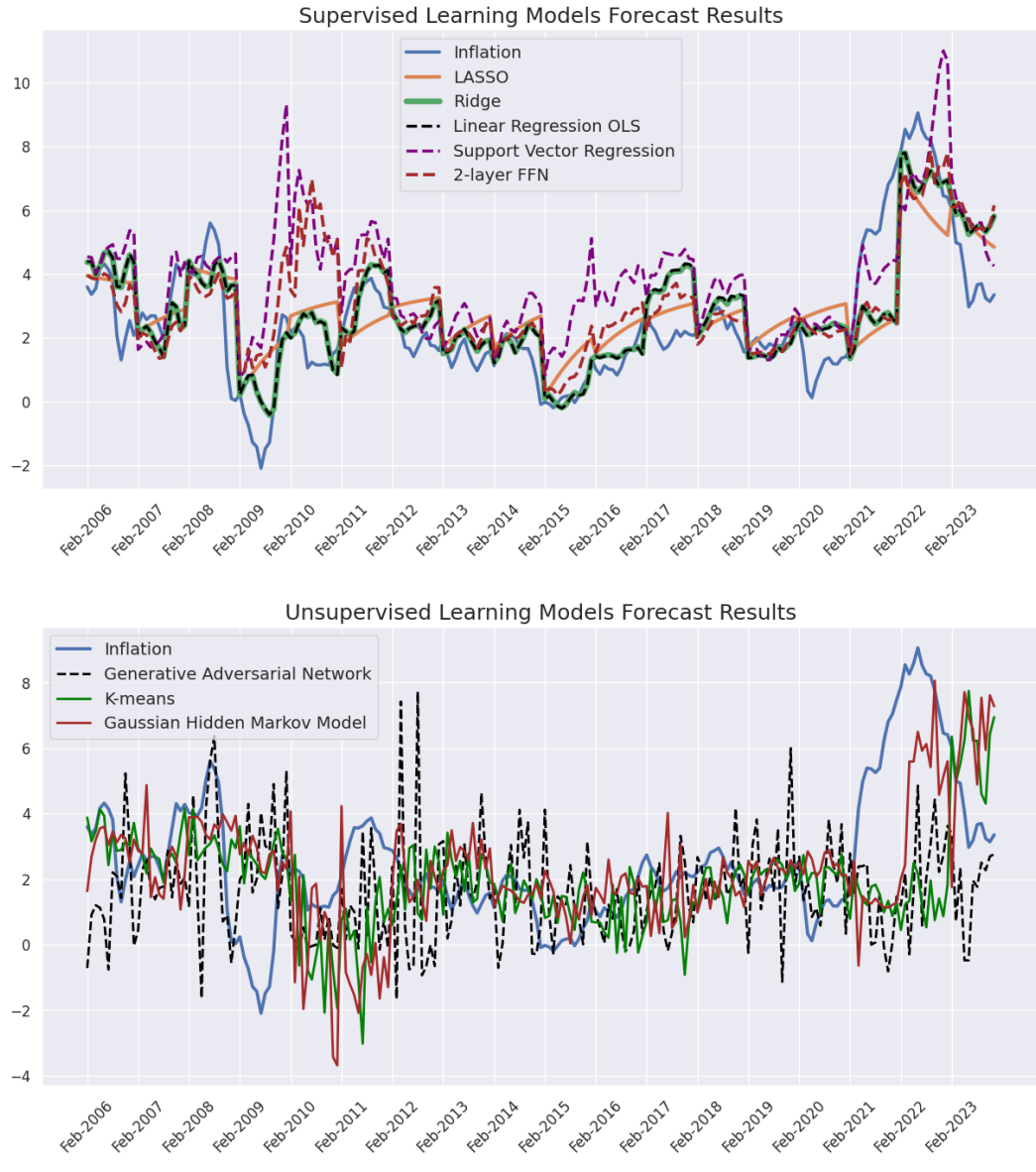


FIGURE 3.—Forecast Results of Supervised and Unsupervised Learning Models

piecewise linear. Moreover, looking at Figure 3, it is possible to observe that Ridge and OLS forecasts are almost identical and visually indistinguishable. However, the Diebold-Mariano test of the Ridge-OLS pair yields a p-value of 0.013, indicating a statistically significant difference in accuracy between these models. Thus, it is possible to say that the regularization of linear regression with Ridge led to an improved MSE over OLS, although only marginally by 0.01. Conversely, the regularization with LASSO yields a worse forecast with an MSE of 2.22

compared to 1.72 of OLS. This outcome can be explained by the fact that the dimension of the training data is 24 (the size of window W_k) which is smaller relative to 337 - the number of observations in the training data. In this type of low-dimensional dataset, LASSO's variable elimination approach tends to perform worse than Ridge's shrinkage (Tibshirani, 2011).

Figure 3 shows that the non-linear supervised learning models, SVR and 2-layer FFN, perform slightly worse than the linear models. According to Table I, their MSEs are indeed larger than those of Ridge, OLS linear regression, and LASSO. Looking at Table II, these error discrepancies are statistically significant as indicated by the p-values of the DM tests between the SVR, FFN models and the Ridge, Lasso, and OLS models, all of which are below the 0.05 threshold. Although the Feed Forward Neural Network generally generates a precise inflation forecast with a low MSE of 2.75, it inaccurately predicts a spike in inflation during 2009-2010. On the other hand, Support Vector Regression consistently overestimates inflation from 2009 to 2017, despite relatively accurate predictions in other periods. As a consequence, SVR has a higher MSE of 4.53 than FFN, and the Diebold-Mariano test between these models is significant at a p-value of 0.00001. This outcome is somewhat paradoxical given that SVR uses regularization but performs worse than the unregularized FFN.

Thus, for both linear and non-linear supervised learning models, regularization does not lead to an improved forecast, except only for the case of Ridge and OLS with minimal improvement. This contrasts with findings by Jabbar and Khan (2014) and suggests that overfitting may not be a dominant issue in the US inflation dataset.

The second plot in Figure 3 exhibits predictions by the unsupervised learning models. The forecast produced by the Gaussian Hidden Markov Model has an MSE of 5.23 and seems to capture the pattern of inflation but with some delay. A similar pattern is observed for the prediction by the k-means algorithm, however, it has a larger MSE of 6.0. The DM test result between GHMM and k-means is insignificant, however. The Generative Adversarial Network simulates an extremely randomized forecast, which fluctuates at various points between positive and negative values. It leads to a mean squared error of 6.97, the highest among all models.

In terms of mean squared errors, the unsupervised learning algorithms perform worse compared to their supervised counterparts. The best-performing unsupervised model, GHMM, achieves an MSE of 5.23, which is higher than that of the least optimal supervised model, SVR, with an MSE of 4.53. However based on Table II, the DM test between GHMM and SVR yields an insignificant p-value of 0.3. The Feed-Forward Neural Network, on the other hand, has a statistically significantly lower MSE than all the unsupervised learning models. Similarly, the linear supervised learning models - Ridge, OLS Linear Regression, and LASSO - exhibit substantially lower forecast errors compared to any unsupervised learning model. This

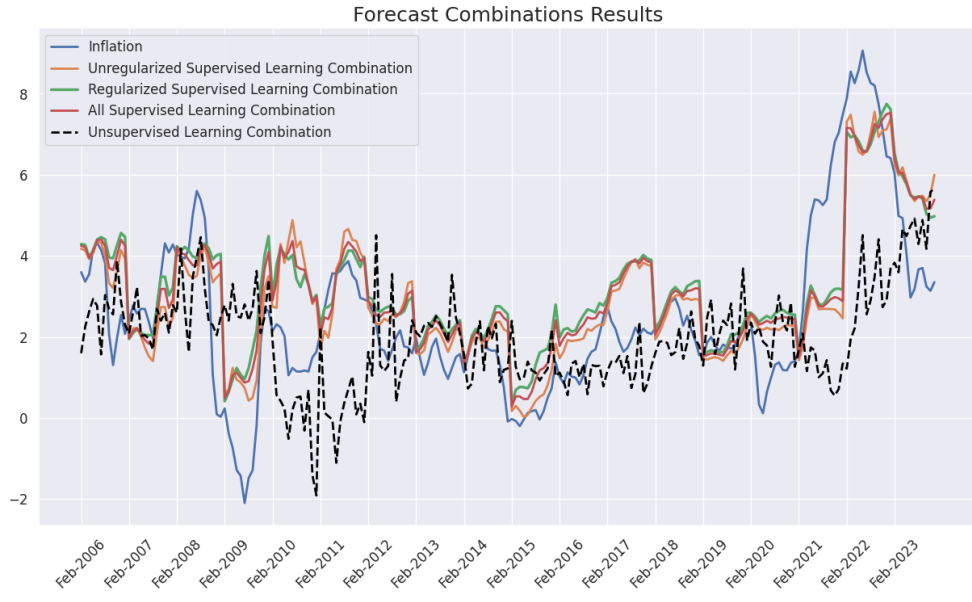


FIGURE 4.—Forecast Results of Combinations of Models

MSE difference is also statistically significant: all the p-values of the Diebold-Mariano test of the linear models against unsupervised models are extremely small. The superior performance of supervised learning aligns with Nasteski (2017) that supervised learning tends to be more precise than unsupervised learning on low-complexity data.

Furthermore, looking at Figure 3, it is evident that the forecasts generated by unsupervised learning models are highly noisy and fluctuating compared to those produced by supervised learning models. This could be explained by the fact that the unsupervised learning models rely on random simulation to obtain forecasts, meanwhile, predictions by the supervised models are the fixed values derived from estimated parameters. The findings by Oral et al. (2012) explain this result to some extent: since unsupervised learning excels on highly variable datasets, on the less noisy inflation data it might generate a large variance.

Next, the results of forecast combinations are exhibited in Figure 4. Firstly, it is possible to observe that combining the unsupervised learning models with equal weights results in a less fluctuating forecast compared to each model individually. Indeed, this leads to a significantly improved inflation prediction with an MSE of 4.77 from Table I. This MSE is lower than that of any individual unsupervised learning model and almost as precise as SVR. However, it remains higher than the MSEs of the three supervised learning combinations, with these error differences being statistically significant according to Table II. Notably, combining unsupervised learning with equal weights reduces the large variability present in the individual models. This effect is not observed in the supervised learning combinations, because likely

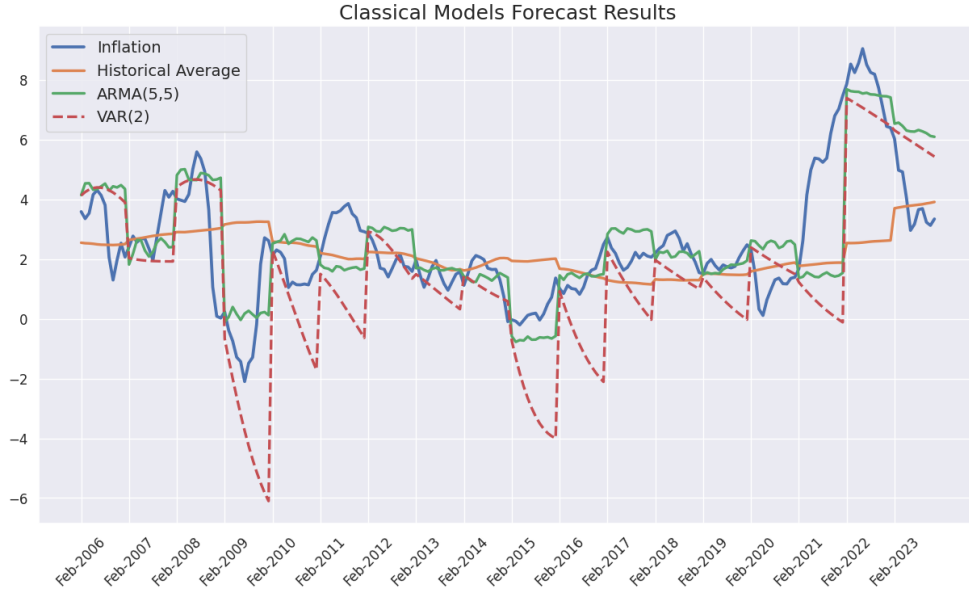


FIGURE 5.—Forecast Results of the Classical Models

the volatility of the supervised learning forecasts is already low in the first place. As a consequence, the unsupervised learning combination is the only case where the combination of models leads to an improved MSE performance over its individual components, which is not the case with the three combinations of supervised learning models. Among the supervised learning combinations, the prediction without regularization achieves an MSE of 1.99, lower than the combination with regularization, which has an MSE of 2.13. This outcome confirms the previous result of this research that the regularized models do not always outperform their non-regularized counterparts.

Finally, the forecast results of the classical models are plotted in Figure 5. The ARMA(5,5) predicted series follows the inflation series closely, accurately replicating its fluctuations. This is not surprising since the ARMA forecast has the third-lowest MSE of 2.51 among all the individual models. In contrast, the historical average forecast fails to capture the variations in the inflation data, as seen in Figure 5. The VAR(2) model exhibits the least accurate prediction among the classical models, with an MSE of 5.44. It tends to severely magnify the negative values of inflation, especially in the 2010 and 2015 time periods.

It is only safe to say that the linear supervised learning algorithms - Ridge, OLS, and LASSO - outperformed the classical models, especially in terms of mean squared error. Meanwhile, the unsupervised learning models generally performed worse than the classical models, with only the exception that the GHMM forecast has a lower MSE than the VAR(2) model. Although, the p-value of the DM test between GHMM and VAR(2) is insignificant at 0.78. Moreover,

this research found that the classical model ARMA(5,5) achieved one of the lowest MSE of 2.51, comparable to LASSO and FFN in terms of precision. Table II shows that the DM test's p-value for ARMA against FFN is 0.46, and against LASSO is 0.10, both of which are much larger than the 0.05. This finding is consistent with Iaousse et al. (2023) that the classical linear models tend to perform well on low-dimensional datasets. It also aligns with Reichlin (2009) that the more complex machine learning algorithms do not always outperform the simpler classical ones.

4. CONCLUSION

In this section, the research question is answered based on the results of the analysis. The additional findings and suggestions for further research are also discussed.

The analysis of results leads to a conclusion that the supervised learning models, both regularized and unregularized, outperform the unsupervised models in forecasting US inflation. The mean squared errors of the unsupervised learning models are significantly larger than those of any supervised learning model. Even when taking an equal-weight combination of unsupervised learning, its MSE remains larger than that of the supervised learning models. This result aligns with Nasteski (2017) that generally supervised learning algorithms have higher precision than unsupervised ones since they are trained on labeled data. Moreover, the unsupervised learning models produce more unstable and fluctuating forecasts compared to the supervised models, likely because unsupervised algorithms rely on simulation to generate forecasts.

While this research concluded that supervised learning outperformed its unsupervised counterpart in predicting US inflation, it is incorrect to generalize that unsupervised learning is inferior in the domain of forecasting. As stated by Oral et al. (2012) unsupervised learning tends to outperform supervised learning on highly complex and noisy datasets, on which supervised learning suffers heavily from overfitting. Therefore, the low-frequency and low-dimensional inflation rate data may not be suitable for unsupervised algorithms. Moreover, it is determined that the linear machine learning models perform the best in this research, suggesting a strong linearity in the US inflation data. Unsupervised learning models can work better for more non-linear and complex datasets, such as multivariate volatility or high-frequency stock data, which is a suggestion for further research. Furthermore, only three unsupervised models are considered in this paper in contrast to five supervised learning ones. A recommendation would be to include more unsupervised learning models, such as Boltzmann machines or Functional Principal Component Analysis (FPCA) for future research.

REFERENCES

ALLOGHANI, MOHAMED, DHIYA AL-JUMEILY, JAMILA MUSTAFINA, ABIR HUSSAIN, AND AHMED J. ALJAAF (2020): "A Systematic Review on Supervised and Unsupervised Machine Learning Algorithms for Data Science," in *Supervised and Unsupervised Learning for Data Science*, ed. by Michael W. Berry, Azlinah Mohamed, and Bee Wah Yap, Cham: Springer International Publishing, 3–21. [2]

AMIN-NASERI, M.R. AND A.R. SOROUSH (2008): "Combined use of unsupervised and supervised learning for daily peak load forecasting," *Energy Conversion and Management*, 49 (6), 1302–1308. [2]

ARAUJO, GUSTAVO SILVA AND WAGNER PIAZZA GAGLIANONE (2023): "Machine learning methods for inflation forecasting in Brazil: New contenders versus classical models," *Latin American Journal of Central Banking*, 4 (2), 100087. [1]

CATELLO, LUIGI, LUDOVICA RUGGIERO, LUCIA SCHIAVONE, AND MARIO VALENTINO (2023): "Hidden Markov Models for Stock Market Prediction," . [10, 23]

DIEBOLD, FRANCIS X. AND ROBERTO S. MARIANO (1995): "Comparing Predictive Accuracy," *Journal of Business & Economic Statistics*, 13 (3), 253–263. [13]

DIEBOLD, FRANCIS X. AND MINCHUL SHIN (2019): "Machine learning for regularized survey forecast combination: Partially-egalitarian LASSO and its derivatives," *International Journal of Forecasting*, 35 (4), 1679–1691. [2, 8, 12, 13]

GUAJARDO, JOSÉ, RICHARD WEBER, AND JAIME MIRANDA (2006): "A Forecasting Methodology Using Support Vector Regression and Dynamic Feature Selection," *Journal of Information & Knowledge Management*, 05 (04), 329–335. [7]

HEIJ, CHRISTIAAN, PAUL DE BOER, PHILIP HANS FRANSES, TEUN KLOEK, AND HERMAN K VAN DIJK (2004): *Econometric Methods with Applications in Business and Economics*, Oxford University PressOxford. [6]

IAOUSSE, MBAREK, YOUNESS JOUILIL, MOHAMED BOUINCHA, AND DRISS MENTAGUI (2023): "Comparative Simulation Study of Classical and Machine Learning Techniques for Forecasting Time Series Data," *International Journal of Online and Biomedical Engineering (iJOE)*, 19 (08), 56–65. [1, 20]

JABBAR, HAIDER KHALAF AND RAFIQUL ZAMAN KHAN (2014): "Methods to Avoid Over-Fitting and Under-Fitting in Supervised Machine Learning (Comparative Study)," in *Computer Science, Communication and Instrumentation Devices*, Research Publishing Services, 163–172. [2, 17]

JUSTER, F. THOMAS, PAUL WACHTEL, SAUL HYMANS, AND JAMES DUESENBERRY (1972): "Inflation and the Consumer," *Brookings Papers on Economic Activity*, 1972 (1), 71–121. [3, 4]

KETCHEN JR., DAVID J. AND CHRISTOPHER L. SHOOK (1996): "THE APPLICATION OF CLUSTER ANALYSIS IN STRATEGIC MANAGEMENT RESEARCH: AN ANALYSIS AND CRITIQUE," *Strategic Management Journal*, 17 (6), 441–458. [9]

LABIAD, BADRE, ABDELAZIZ BERRADO, AND LOUBNA BENABBOU (2023): "Predicting extreme events in the stock market using generative adversarial networks," *International Journal of Advances in Intelligent Informatics*, 9 (2), 218. [11]

NASTESKI, VLADIMIR (2017): "An overview of the supervised machine learning methods," *HORIZONS.B*, 4, 51–62. [2, 18, 20]

ORAL, MUSTAFA, EMEL LAPTALI ORAL, AND AHMET AYDIN (2012): "Supervised vs. unsupervised learning for construction crew productivity prediction," *Automation in Construction*, 22, 271–276. [2, 18, 20]

RABINER, L. AND B. JUANG (1986): "An introduction to hidden Markov models," *IEEE ASSP Magazine*, 3 (1), 4–16. [10]

REICHLIN, LUCREZIA (2009): "Comments on "Phillips Curve Inflation Forecasts" by James H. Stock and Mark W. Watson," in *Understanding Inflation and the Implications for Monetary Policy*, ed. by Jeff Fuhrer, Yolanda K. Kodrzycki, Jane Sneddon Little, and Giovanni P. Olivei, The MIT Press, 195–204. [2, 4, 20]

RENOU-MAISSANT, PATRICIA (2019): "Is Oil Price Still Driving Inflation?" *The Energy Journal*, 40 (6), 199–220. [3]

SARAVANAN, R. AND POTHULA SUJATHA (2018): "A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India: IEEE, 945–949. [2]

SAXENA, DIVYA AND JIANNONG CAO (2022): "Generative Adversarial Networks (GANs): Challenges, Solutions, and Future Directions," *ACM Computing Surveys*, 54 (3), 1–42. [11]

STOCK, JAMES H AND MARK W WATSON (1999): "Forecasting inflation," *Journal of Monetary Economics*, 44 (2), 293–335. [1, 3]

TIBSHIRANI, ROBERT (2011): "Regression shrinkage and selection via the lasso: a retrospective," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 73 (3), 273–282. [17]

TUNG, H. K. K. AND M. C. S. WONG (2009): "Financial Risk Forecasting with Nonlinear Dynamics and Support Vector Regression," *The Journal of the Operational Research Society*, 60 (5), 685–695. [7]

WANG, YANG (2020): "A Mathematical Introduction to Generative Adversarial Nets (GAN)," . [11]

APPENDIX

The Forward Algorithm for the Gaussian Hidden Markov Model

Given a time series $\{x_t\}_{t=1}^{T-1}$ which is assumed to follow a Gaussian Hidden Markov Model with state space $\{1,\dots,K\}$, transition matrix A, initial distribution π, and emission matrix $B = \big(f_k(o_t)\big)_{k,t}$. Let $\{o_t\}_{t=T}^{T^*}$ be an observed time sequence in the future. The forward algorithm (Catello et al., 2023) calculates the likelihood probability of a given series $\{o_t\}_{t=T}^{T^*}$ to come from GHMM with the given parameters and to be the continuation of $\{x_t\}_{t=1}^{T-1}$. This probability is calculated from the forward probability $\alpha_s(k)$ of state k at time s, i.e the probability of observing the sequence $\{o_t\}_{t=T}^{s}$ and ending up at state k at time s. Initialize $\alpha_T(k) = \pi_k b_k(o_T)$. After initialization, the next $\alpha_{T+j}(k)$ for $j = 1, 2, \dots, T^* - T$ are calculated recursively:

$$\alpha_{T+j}(k) = b_k(o_{T+j}) \sum_{i=1}^{K} \alpha_{T+j-1}(i) A_{ik} \tag{31}$$

where A_{ik} is the (i,k) element of the transition matrix A. Then, the likelihood probability is

$$\mathbb{P}\Big(\{X_t\}_{t=T}^{T^*} = \{o_t\}_{t=T}^{T^*} \Big|\ \text{GHMM}(A,\pi,B)\Big) = \sum_{i=1}^{K} \alpha_{T^*}(i) \tag{32}$$