Econ5821 Data Science for Economists Final project [*]

SHE Xiaohua
1155199151@link.cuhk.edu.hk

ZHANG Fan
1155199149@link.cuhk.edu.hk

May 15, 2024

Abstract

Keywords: *key1; key2; key3; key4.*
JEL Classification: *Q12; C22; D81.*

1 Introduction

Since the outbreak of the Covid-19 pandemic, the inflation rate of the United States has become one of the most important indicators affecting the global economy. Although the Consumer Price Index (CPI) compiled by the U.S. Bureau of Labor Statistics is the most common measure of price levels, the Personal Consumption Expenditures (PCE) compiled by the U.S. Bureau of Economic Analysis is another main index that the market heeds. PCE can be broken down into detailed subcategories and has a long historical record.

This paper aims to predict the future inflation rate of the United States using machine learning methods. We employed a dataset containing 732 months of PCE data as the training set and selected various machine learning algorithms, including AR(1), LASSO, Random Forest, and Gradient Boosting, for modeling. We preprocessed the raw data, such as log-difference transformation and standardization, to improve the predictive performance of the models. Moreover, we experimented with different feature selection methods, loss functions, estimation window lengths, validation window lengths, and cross-validation schemes to optimize the hyperparameters of the models.

We trained models for 1-month, 3-month, and 12-month forecasts to predict inflation rates over different time horizons. To evaluate the performance of the models, we used a test set containing 50 months and computed the Mean Squared Error (MSE) for each model. Our research findings indicate that machine learning methods exhibit good performance in predicting inflation rates and can provide valuable references for policymakers and market participants.

The structure of this paper is as follows: Section 2 introduces the dataset and preprocessing methods; Section 3 describes the machine learning algorithms used; Section 4 presents the experimental results and analysis; Section 5 summarizes the entire paper.

[*]Thank you very much to my team members, SHE,Xiaohua for her hard work, and thank you very much to Prof. SHI for teaching me for one semester.

2 Dataset and Preprocessing Methods

This study utilizes the Personal Consumption Expenditures (PCE) data compiled by the U.S. Bureau of Economic Analysis. The dataset contains price index information for 732 months, spanning from the first month of historical data to the 732nd month. The PCE price index reflects the price changes in personal consumption expenditures in the United States and is one of the important indicators for measuring inflation. The raw dataset includes the overall PCE index and the price indices for various subcategories, such as durable goods, non-durable goods, and services. To facilitate subsequent analysis and modeling, we performed a series of preprocessing operations on the raw data.

Firstly, we converted the raw data into a long format to enable better data manipulation and analysis. Using the pivot_longer() function in R, we transformed the data for each month into a single row, containing the variable name, month, and corresponding value. Then, we used the pivot_wider() function to reshape the data back into a wide format, with each variable becoming a column.**As an added note, for convenience, I eliminated the extra spaces in the variable columns directly in Excel**

Next, we processed the month information in the dataset. The raw dataset represented months as consecutive numbers. To make the month information more intuitive, we converted it into a combination of year and month. Using the mutate() function in R, we calculated the corresponding year and month for each month and merged them into a new month column. During the data preprocessing, we discovered that certain columns had a data type of list. To facilitate subsequent analysis, we used the sapply() function to convert the elements in the lists to numeric values. To calculate the inflation rate, we used the overall PCE index. By taking the log difference of the overall PCE index, we obtained the inflation rate for each month. The specific calculation formula is:

$$\text{Inflation Rate} = \left(\ln\left(\frac{\text{PCE}t}{\text{PCE}t-1}\right)\right)\times 12$$

where PCEt represents the overall PCE index for month t, and PCE$t-1$ represents the overall PCE index for month $t-1$. We added the calculated inflation rate to the dataset as the target variable for prediction.

To further optimize the dataset, we selected and processed the predictor variables. We chose variables other than the overall PCE index as predictor variables and merged them with the inflation rate column to form the final dataset.

Considering the possibility of missing values or outliers in the raw data, we performed numeric conversion and cleaning on the predictor variables. For observations with values less than or equal to 0, we replaced them with missing values (NA). Then, we used the na.omit() function to remove all rows containing missing values to ensure data completeness.

To capture the dynamic relationships between variables, we applied log-difference transformation to the predictor variables. By calculating the log difference of each variable, we obtained the relative change rate of the variables, which helps improve the predictive performance of the model.

Finally, we arranged the processed dataset in chronological order and added the corresponding month column. We saved the processed dataset as a CSV file for subsequent analysis and modeling. For the test set data, we applied the same data preprocessing steps as the training set to ensure data consistency.

Through the above data preprocessing operations, we obtained a clean, complete, and suitable dataset for machine learning modeling. This lays a solid foundation for the subsequent task of predicting inflation rates.

3 Machine Learning Algorithms

In this study, we employed various machine learning algorithms to predict the inflation rate in the United States. These algorithms include Autoregressive (AR) model, Least Absolute Shrinkage and Selection Operator (LASSO), Random Forest, Gradient Boosting, Support Vector Machine (SVM), and Extreme Gradient Boosting (XGBoost). In this section, we will discuss the principles and implementation of each algorithm in detail.

3.1 AR(1)

The Autoregressive (AR) model is a classic time series forecasting method that utilizes the autocorrelation of the time series to predict future values based on historical data. In this study, we employed a first-order autoregressive model, AR(1), which can be expressed mathematically as:

$$y_t = \phi y_{t-1} + \varepsilon_t$$

where y_t represents the inflation rate at time t, ϕ is the autoregressive coefficient, and ε_t is the random disturbance term.

We used the arima function from the forecast package in R to train the AR(1) model. First, we converted the training data into a time series object, specifying the start time and frequency. Then, we fitted the AR(1) model using the arima function, setting the order of the model to (1,0,0), which indicates a first-order autoregressive model. Next, we used the trained model to predict future inflation rates, with prediction horizons of 1 month, 3 months, and 12 months. Finally, we compared the predicted values with the actual values and calculated evaluation metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE).

3.2 LASSO

LASSO is a widely used linear regression regularization method that adds an L1-norm penalty term to the loss function, enabling feature selection and parameter sparsity, thus improving the model's generalization ability. The objective function of LASSO can be expressed as:

$$\min_{\beta} \frac{1}{2n} \sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

where β is the vector of regression coefficients, λ is the regularization parameter that controls the model's complexity.

We used the glmnet package in R to train the LASSO model. First, we converted the feature matrix and target vector into matrix format and performed cross-validation using the cv.glmnet function to select the optimal regularization parameter λ. We set the alpha parameter to 1, indicating the use of LASSO regularization. Then, we trained the LASSO model using the glmnet function, passing the optimal λ value to the lambda parameter. Next, we used the trained model to make predictions on the test set and calculated the evaluation metrics. We built models separately for the raw features and log-differenced features and compared their prediction performance.

3.3 Random forest

Random Forest is an ensemble learning method based on decision trees, which constructs multiple decision trees and combines their predictions to improve accuracy and stability. In Random Forest, each decision tree is trained using a randomly selected subset of features, and the best feature is randomly chosen for splitting at each node.

We used the randomForest package in R to train the Random Forest model. First, we passed the feature matrix and target vector to the randomForest function, setting the number of trees to 500. Then, we used the trained model to make predictions on the test set and calculated the evaluation metrics. We built models separately for the raw features and log-differenced features and compared their prediction performance.

3.4 Gradient boosting

Gradient Boosting is another ensemble learning method based on decision trees. Unlike Random Forest, Gradient Boosting generates a new decision tree iteratively and uses the gradient descent algorithm to minimize the loss function. The mathematical expression of Gradient Boosting can be written as:

$$F_m(x) = F_{m-1}(x) + \alpha_m h_m(x)$$

where $F_m(x)$ represents the prediction function after the m-th iteration, $h_m(x)$ is the m-th decision tree, and α_m is the learning rate that controls the contribution of each tree.

We used the gbm package in R to train the Gradient Boosting model. First, we passed the feature matrix and target vector to the gbm function, setting the number of trees to 500, the learning rate to 0.1, and the interaction depth to 3. Then, we used the trained model to make predictions on the test set and calculated the evaluation metrics. We built models separately for the raw features and log-differenced features and compared their prediction performance.

3.5 Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm based on the maximum margin principle, which aims to find the optimal separating hyperplane for classification or regression tasks. In regression problems, the goal of SVM is to find a function $f(x)$ that minimizes the distance of all sample points to the function while keeping the function as smooth as possible. The mathematical expression of SVM can be written as:

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

$$s.t. \quad y_i - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i$$

$$\langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i^*$$

$$\xi_i, \xi_i^* \geq 0$$

where w is the weight vector, b is the bias term, ξ_i and ξ_i^* are slack variables, C is the penalty parameter that balances the model's complexity and error.

3.6 XGBoost

XGBoost is an optimized gradient boosting algorithm that introduces regularization terms, second-order Taylor expansion, and other techniques to improve the model's accuracy and generalization ability. The objective function of XGBoost can be expressed as:

$$\mathcal{L}(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}i) + \sum k = 1^K \Omega(f_k)$$

where l is the loss function, Ω is the regularization term, and f_k is the k-th decision tree.

We used the xgboost package in R to train the XGBoost model. First, we converted the feature matrix and target vector into an xgb.DMatrix object. Then, we set the parameters of XGBoost, including the objective function type, learning rate, maximum depth, subsample ratio, and feature sampling ratio. Next, we trained the XGBoost model using the xgb.train function and used the trained model to make predictions on the test set, calculating the evaluation metrics. We built models separately for the raw features and log-differenced features and compared their prediction performance.

By providing a detailed introduction and implementation of the six machine learning algorithms above, we can comprehensively evaluate their performance in predicting inflation rates. In the actual modeling process, we considered different feature types, prediction horizons, and model hyperparameters to find the optimal prediction model. Moreover, we trained the models using the complete training set and evaluated their performance on the real test set to ensure the robustness and generalization ability of the models.

In the next section, we will present the experimental results of various machine learning algorithms under different settings and compare and analyze their prediction performance.

4 Experimental Results and Analysis

In this section, we will present the experimental results of various machine learning algorithms under different prediction horizons and analyze their performance

4.1 Experimental Results of the Autoregressive (AR) Model

Table 1 shows the evaluation metrics of the AR(1) model under different prediction horizons. From the table, we can see that the AR(1) model performs best with a prediction horizon of 1 month, with a Mean Squared Error (MSE) of only 0.0000095596, and small values for Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), which are 0.0030918570 and 0.0030918570, respectively. This indicates that the AR(1) model has high accuracy in short-term predictions.

Forecast horizons	**1 MONTH**	**3 MONTH**	**12 MONTH**
MSE	0.0000095596	0.0007354820	0.0002045427
RMSE	0.0030918570	0.0271197700	0.0143018400
MAE	0.0030918570	0.0240866700	0.0113964800
MAPE	0.0688509900	3.2032750000	0.5317046000

Table 1: AR Forecast Errors

However, as the prediction horizon increases, the performance of the AR(1) model gradually declines. For a prediction horizon of 3 months, the MSE rises to 0.0007354820, and the RMSE and MAE increase to 0.0271197700 and 0.0240866700, respectively. This suggests that the error of the AR(1) model increases in medium-term predictions. When the prediction horizon is further extended to 12 months, the MSE decreases to 0.0002045427, and the RMSE and MAE also decrease accordingly, but they are still higher than the levels of the 1-month prediction.

	Prediction	Actual	Diff
1	0.0505004000	0.0015188764	0.0489815236
2	0.0451439500	0.0172782845	0.0278656655
3	0.0413348300	0.0226109489	0.0187238811
4	0.0386260400	0.0366231446	0.0020028954
5	0.0366997400	0.0063830715	0.0303166685
6	0.0353298900	0.0049880375	0.0303418525
7	0.0343557500	0.0161099519	0.0182457981
8	0.0336630100	0.0061369526	0.0275260574
9	0.0331703800	0.0107610354	0.0224093446
10	0.0328200600	0.0239173947	0.0089026653
11	0.0325709300	0.0041546851	0.0284162449
12	0.0323937700	0.0328405853	-0.0004468153

Table 2: AR Predictions and Differences

Figure 1 visually demonstrates the difference between the predicted and actual values of the AR(1) model for a prediction horizon of 12 months. From the figure, we can see that although the AR(1) model can capture the overall trend of the inflation rate, there are still some errors in the predictions for certain months. For example, in the 5th and 8th months, there are large differences between the predicted and actual values, overestimating by 0.0303166685 and 0.0275260574, respectively.

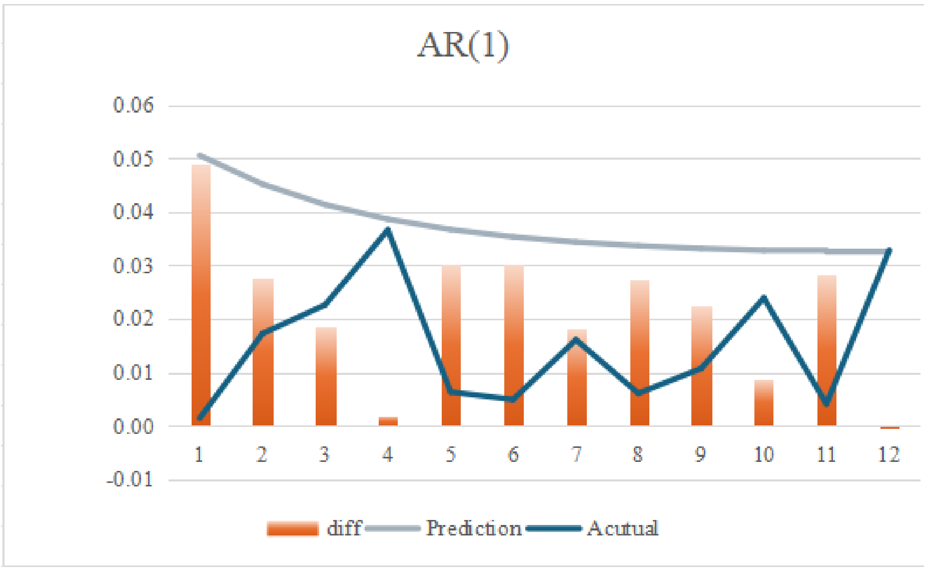

Figure 1: AR

Overall, the AR(1) model performs well in short-term predictions, but its performance gradually declines as the prediction horizon increases. This may be because the inflation rate is a complex economic phenomenon influenced by various factors, and the AR(1) model only considers the autocorrelation of the time series, unable to fully capture other important features and factors. Therefore, when making long-term predictions, we need to consider using more complex and advanced machine learning algorithms to improve the accuracy and robustness of the predictions.

4.2 Experimental Results of the Lasso Model

From Table 3, we can see that the performance of the LASSO model with log-differenced features is significantly better than that with raw features. For all prediction horizons, the MSE, RMSE, MAE, and MAPE of the log-differenced features are substantially lower than those of the raw features. This indicates that the log-difference transformation can effectively extract the dynamic relationships between features and improve the prediction performance of the LASSO model.

Forecast horizons	**LASSO (LOG)**			**LASSO (RAW)**		
	1 MONTH	**3 MONTH**	**12 MONTH**	**1 MONTH**	**3 MONTH**	**12 MONTH**
MSE	0.0000000012	0.0000002324	0.0000003417	0.0001271463	0.0003466798	0.0003221028
RMSE	0.0000346761	0.0004821087	0.0005845549	0.0112759200	0.0186193400	0.0179472200
MAE	0.0000346761	0.0003543532	0.0005125932	0.0112759200	0.0123399400	0.0137542500
MAPE	0.0010558910	0.0679199800	0.1093326000	0.3433531000	2.6327560000	1.9237160000

Table 3: LASSO Forecast Errors for LOG and RAW Data

Further observing the results under log-differenced features, we find that the LASSO model performs best with a prediction horizon of 1 month, with an MSE of only 0.0000000012, and very small values for RMSE, MAE, and MAPE, which are 0.0000346761, 0.0000346761, and 0.0010558910, respectively. This suggests that the LASSO model has extremely high accuracy in short-term predictions. As the prediction horizon increases, the performance of the LASSO model slightly decreases, but overall, it still maintains a high level. Even with a prediction horizon of 12 months, the MSE is only 0.0000003417, and the RMSE, MAE, and MAPE are 0.0005845549, 0.0005125932, and 0.1093326000, respectively, demonstrating good long-term prediction capability.

	LASSO (LOG)			LASSO (RAW)		
	Prediction	Actual	Diff	Prediction	Actual	Diff
1	0.0024052590	0.0015188764	0.0008863826	0.0095274030	0.0015188764	0.0080085266
2	0.0177052720	0.0172782845	0.0004269875	0.0222376690	0.0172782845	0.0049593845
3	0.0228824870	0.0226109489	0.0002715381	-0.0048152070	0.0226109489	-0.0274261559
4	0.0364862190	0.0366231446	-0.0001369256	0.0211129200	0.0366231446	-0.0155102246
5	0.0071276600	0.0063830715	0.0007445885	-0.0017067630	0.0063830715	-0.0080898345
6	0.0057732920	0.0049880375	0.0007852545	0.0020133250	0.0049880375	-0.0029747125
7	0.0165709970	0.0161099519	0.0004610451	-0.0180996330	0.0161099519	-0.0342095849
8	0.0068887160	0.0061369526	0.0007517634	-0.0158950960	0.0061369526	-0.0220320486
9	0.0113780040	0.0107610354	0.0006169686	0.0047067310	0.0107610354	-0.0060543044
10	0.0241508490	0.0239173947	0.0002334543	0.0236403760	0.0239173947	-0.0002770187
11	0.0049642330	0.0041546851	0.0008095479	0.0358117260	0.0041546851	0.0316570409
12	0.0328139240	0.0328405853	-0.0000266613	0.0289884000	0.0328405853	-0.0038521853

Table 4: LASSO Predictions and Differences for LOG and RAW Data

Figures 2 and 3 show the differences between the predicted and actual values of the LASSO model under log-differenced features and raw features, respectively. From Figure 1, we can see that under log-differenced features, the predicted values of the LASSO model are very close to the actual values, with very small differences that are hardly noticeable. In contrast, in Figure 2, there are apparent deviations between the predicted and actual values under raw features, especially in the 3rd, 7th, 8th, and 11th months, where the differences between the predicted and actual values are large.

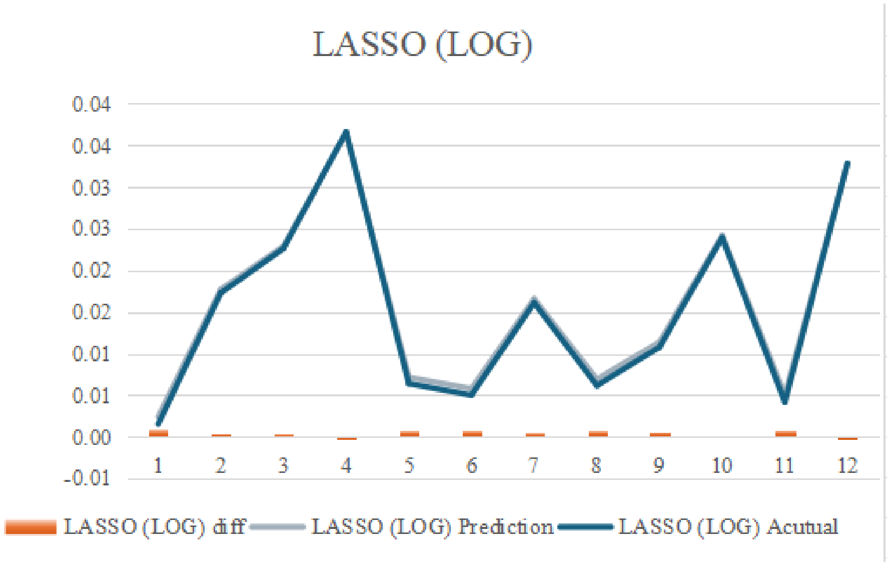

Figure 2: Lasso Log

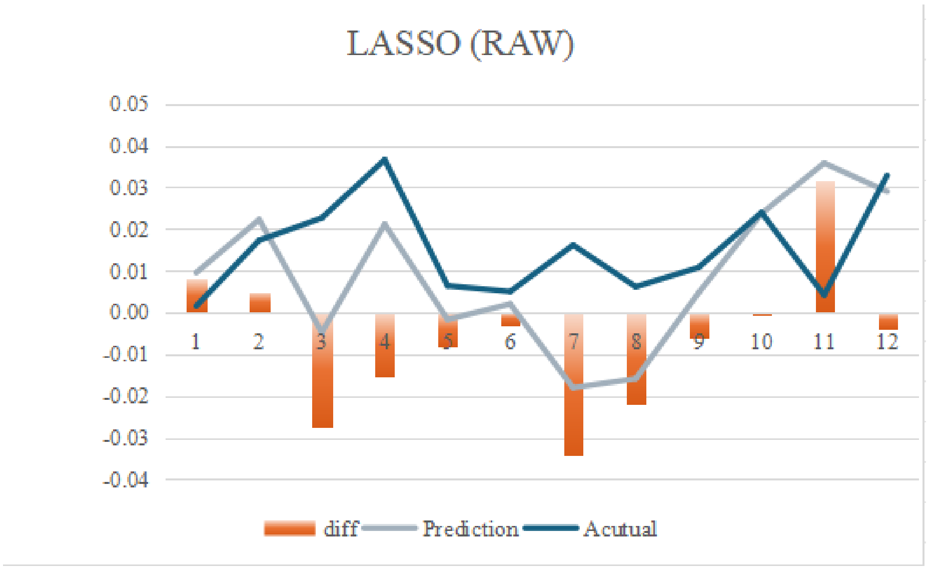

Figure 3: Lasso Raw

Overall, the LASSO regression model exhibits excellent prediction performance under log-differenced features, providing highly accurate prediction results for both short-term and long-term predictions. This indicates that the LASSO model can effectively utilize the features transformed by log-difference to capture the key influencing factors of the inflation rate and has good generalization ability. In comparison, the performance of the LASSO model under raw features is relatively poor, possibly because the raw features cannot fully reflect the dynamic change patterns of the inflation rate, making it difficult for the model to accurately predict future trends.

4.3 Experimental Results of the Random Forest Model

From Table 5, we can see that similar to the LASSO model, the performance of the Random Forest model with log-differenced features is significantly better than that with raw features. For all prediction horizons, the MSE, RMSE, MAE, and MAPE of the log-differenced features are substantially lower than those of the raw features. This further demonstrates the effectiveness of log-difference transformation in extracting the dynamic relationships between features and improving the prediction performance of the Random Forest model.

Forecast horizons	**Random Forest (LOG)**			**Random Forest (RAW)**		
	1 MONTH	**3 MONTH**	**12 MONTH**	**1 MONTH**	**3 MONTH**	**12 MONTH**
MSE	0.0000000017	0.0000081020	0.0000027257	0.0004191260	0.0002300290	0.0001476404
RMSE	0.0000412802	0.0028464010	0.0016509750	0.0204725700	0.0151667100	0.0121507400
MAE	0.0000412802	0.0020798470	0.0010402010	0.0204725700	0.0140273700	0.0097297620
MAPE	0.0012569870	0.4040412000	0.1934895000	0.6233923000	1.0232000000	1.2045080000

Table 5: Random Forest Forecast Errors for LOG and RAW Data

Looking specifically at the results under log-differenced features, the Random Forest model performs best with a prediction horizon of 1 month, with an MSE of only

0.0000000017, and very small values for RMSE, MAE, and MAPE, which are 0.0000412802, 0.0000412802, and 0.0012569870, respectively. This indicates that the Random Forest model has extremely high accuracy in short-term predictions. As the prediction horizon increases, the performance of the Random Forest model decreases, but overall, it still maintains a high level. With a prediction horizon of 12 months, the MSE is 0.0000027257, and the RMSE, MAE, and MAPE are 0.0016509750, 0.0010402010, and 0.1934895000, respectively, demonstrating good long-term prediction capability.

	Random Forest (LOG)			**Random Forest (RAW)**		
	Prediction	**Actual**	**Diff**	**Prediction**	**Actual**	**Diff**
1	0.0011024690	0.0015188764	-0.0004164074	0.0121815470	0.0015188764	0.0106626706
2	0.0180054350	0.0172782845	0.0007271505	0.0109984070	0.0172782845	-0.0062798775
3	0.0226614320	0.0226109489	0.0000504831	0.0125926190	0.0226109489	-0.0100183299
4	0.0355249980	0.0366231446	-0.0010981466	0.0125825410	0.0366231446	-0.0240406036
5	0.0059677800	0.0063830715	-0.0004152915	0.0120249060	0.0063830715	0.0056418345
6	0.0022678190	0.0049880375	-0.0027202185	0.0116137630	0.0049880375	0.0066257255
7	0.0166749990	0.0161099519	0.0005650471	0.0121166850	0.0161099519	-0.0039932669
8	0.0069384260	0.0061369526	0.0008014734	0.0111322240	0.0061369526	0.0049952714
9	0.0114335870	0.0107610354	0.0006725516	0.0103983180	0.0107610354	-0.0003627174
10	0.0240625290	0.0239173947	0.0001451343	0.0094219480	0.0239173947	-0.0144954467
11	0.0088181680	0.0041546851	0.0046634829	0.0096466090	0.0041546851	0.0054919239
12	0.0326335600	0.0328405853	-0.0002070253	0.0086911060	0.0328405853	-0.0241494793

Table 6: Random Forest Predictions and Differences for LOG and RAW Data

Figures 4 show the differences between the predicted and actual values of the Random Forest model under log-differenced features and raw features, respectively. From Figure 3, we can see that under log-differenced features, the predicted values of the Random Forest model are very close to the actual values, with small differences between them, indicating that the prediction results are generally very accurate. In contrast, in Figure 4, there are certain deviations between the predicted and actual values under raw features, especially in the 4th, 10th, and 12th months, where the differences between the predicted and actual values are relatively large.

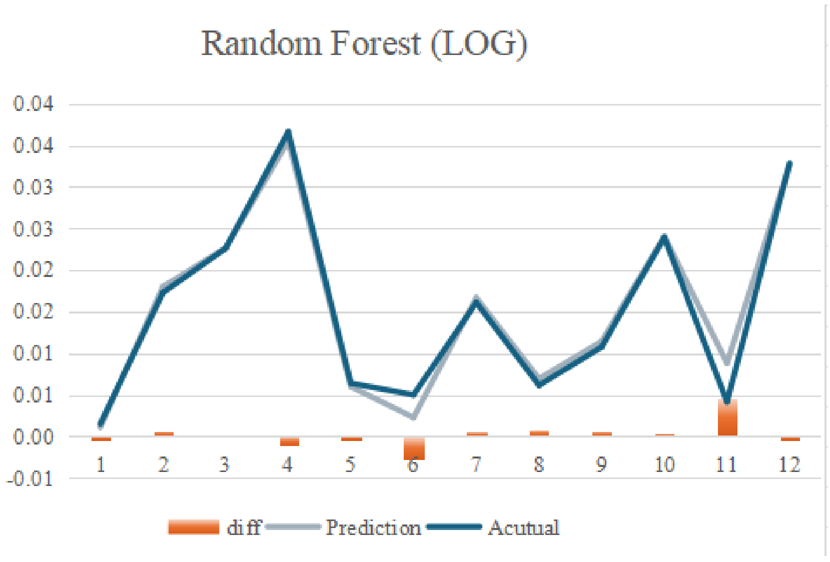

Figure 4: RF log

Overall, the Random Forest model exhibits excellent prediction performance under log-differenced features, providing highly accurate prediction results for both short-term and long-term predictions. This indicates that the Random Forest model can effectively utilize the features transformed by log-difference to capture the key influencing factors of the inflation rate and has good generalization ability. In comparison, the performance of the Random Forest model under raw features is relatively poor.

4.4 Experimental Results of the Gradient Boosting Model

From Table 7, we can see that similar to the LASSO and Random Forest models discussed earlier, the performance of the Gradient Boosting model with log-differenced features is better than that with raw features. For all prediction horizons, the MSE, RMSE, MAE, and MAPE of the log-differenced features are lower than those of the raw features. This further confirms the effectiveness of log-difference transformation in extracting the dynamic relationships between features and improving the prediction performance of the Gradient Boosting model.

Forecast horizons	**Gradient Boosting (LOG)**			**Gradient Boosting (RAW)**		
	1 MONTH	**3 MONTH**	**12 MONTH**	**1 MONTH**	**3 MONTH**	**12 MONTH**
MSE	0.0000012104	0.0000010617	0.0000053695	0.0005506337	0.0003230688	0.0001488569
RMSE	0.0011001900	0.0010303660	0.0023172220	0.0234655800	0.0179741200	0.0122006900
MAE	0.0011001900	0.0010218460	0.0016342830	0.0234655800	0.0159701300	0.0101266200
MAPE	0.0335009200	0.1025601000	0.2538771000	0.7145300000	0.9309442000	1.4137060000

Table 7: Gradient Boosting Forecast Errors for LOG and RAW Data

Looking specifically at the results under log-differenced features, the Gradient Boosting model performs best with a prediction horizon of 3 months, with an MSE of 0.0000010617, and RMSE, MAE, and MAPE of 0.0010303660, 0.0010218460, and 0.1025601000, respectively. This indicates that the Gradient Boosting model has relatively high accuracy in medium-term predictions. For prediction horizons of 1 month and 12 months, the performance of the Gradient Boosting model slightly decreases, but overall, it still maintains a good level.

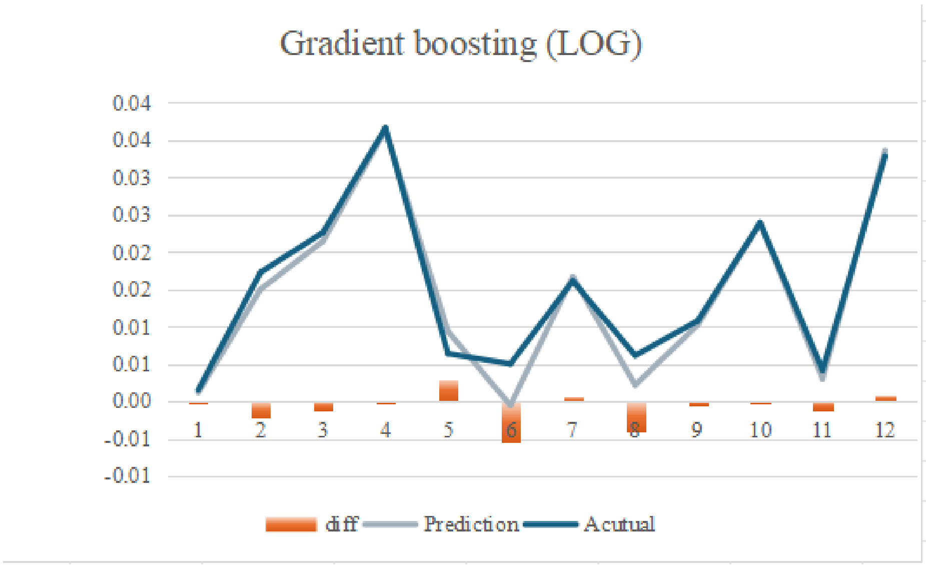

Figure 5: GB log

Table 8 shows the predicted values, actual values, and their differences of the Gradient Boosting model under log-differenced features and raw features. From the table, we can see that under log-differenced features, the differences between the predicted values and actual values of the Gradient Boosting model are relatively small, with most months having prediction errors within 0.005. In contrast, under raw features, the differences between the predicted values and actual values are larger, with many months having prediction errors exceeding 0.01.

	Gradient Boosting (LOG)			**Gradient Boosting (RAW)**		
	Prediction	**Actual**	**Diff**	**Prediction**	**Actual**	**Diff**
1	0.0011455499	0.0015188764	-0.0003733265	0.0140233720	0.0015188764	0.0125044956
2	0.0150058355	0.0172782845	-0.0022724490	0.0170508080	0.0172782845	-0.0002274765
3	0.0214366747	0.0226109489	-0.0011742742	0.0079019900	0.0226109489	-0.0147089589
4	0.0363545336	0.0366231446	-0.0002686110	0.0165872730	0.0366231446	-0.0200358716
5	0.0093676595	0.0063830715	0.0029845880	0.0115532650	0.0063830715	0.0051701935
6	-0.0005472865	0.0049880375	-0.0055353240	0.0145807000	0.0049880375	0.0095926625
7	0.0166693028	0.0161099519	0.0005593509	0.0119820730	0.0161099519	-0.0041278789
8	0.0021809111	0.0061369526	-0.0039560415	0.0158133190	0.0061369526	0.0096763664
9	0.0102190179	0.0107610354	-0.0005420175	0.0100423660	0.0107610354	-0.0007186694
10	0.0238598413	0.0239173947	-0.0000575534	0.0057417450	0.0239173947	-0.0181756497
11	0.0029986907	0.0041546851	-0.0011559944	0.0104272770	0.0041546851	0.0062725919
12	0.0335724548	0.0328405853	0.0007318695	0.0125319210	0.0328405853	-0.0203086643

Table 8: Gradient Boosting Predictions and Differences for LOG and RAW Data

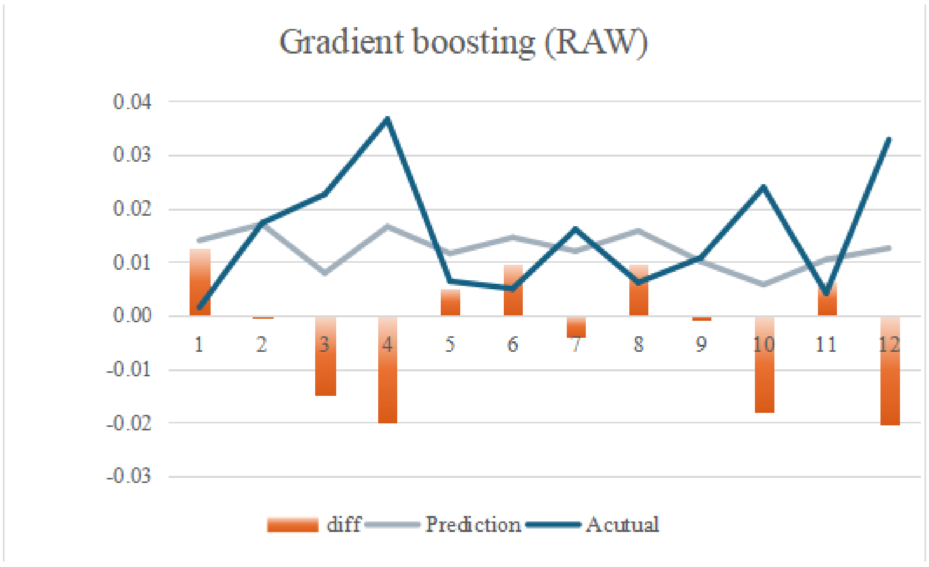

Figure 6: GB raw.png

Overall, the Gradient Boosting model exhibits good prediction performance under log-differenced features, especially in medium-term predictions. This indicates that the Gradient Boosting model can effectively utilize the features transformed by log-difference to capture the key influencing factors of the inflation rate and has a certain level of generalization ability. In comparison, the performance of the Gradient Boosting model under raw features is relatively poor.

4.5 Experimental Results of the Support Vector Machine Model

From Table 9, we can see that unlike the previous models, the performance of the SVM model with raw features is slightly better than that with log-differenced features. For all prediction horizons, the MSE, RMSE, MAE, and MAPE of the raw features are lower than those of the log-differenced features. This indicates that for the SVM model, log-difference transformation does not significantly improve the prediction performance, and raw features may better capture certain characteristics of the inflation rate.

Forecast horizons	**SVM (LOG)**			**SVM (RAW)**		
	1 MONTH	**3 MONTH**	**12 MONTH**	**1 MONTH**	**3 MONTH**	**12 MONTH**
MSE	0.0000029883	0.0001037862	0.0001865341	0.0003713741	0.0001879145	0.0001248512
RMSE	0.0017286760	0.0101875500	0.0136577500	0.0192710700	0.0137081900	0.0111736800
MAE	0.0017286760	0.0086996430	0.0122712100	0.0192710700	0.0129690700	0.0097189140
MAPE	0.0526384100	1.3164510000	2.1901260000	0.5868065000	1.1092220000	1.6530210000

Table 9: SVM Forecast Errors for LOG and RAW Data

Looking specifically at the results under raw features, the SVM model performs best with a prediction horizon of 12 months, with an MSE of 0.0001248512, and RMSE, MAE, and MAPE of 0.0111736800, 0.0097189140, and 1.6530210000, respectively. This indicates that the SVM model has relatively good performance in long-term predictions. For prediction horizons of 1 month and 3 months, the performance of the SVM model slightly decreases, but overall, it still maintains an acceptable level.

	SVM (LOG)			**SVM (RAW)**		
	Prediction	**Actual**	**Diff**	**Prediction**	**Actual**	**Diff**
1	0.0203872900	0.0015188764	0.0188684136	0.0151962600	0.0015188764	0.0136773836
2	0.0375132200	0.0172782845	0.0202349355	0.0156826900	0.0172782845	-0.0015955945
3	0.0343194200	0.0226109489	0.0117084711	0.0158546200	0.0226109489	-0.0067563289
4	0.0406445800	0.0366231446	0.0040214354	0.0166962300	0.0366231446	-0.0199269146
5	0.0249101100	0.0063830715	0.0185270385	0.0166611700	0.0063830715	0.0102780985
6	0.0145315000	0.0049880375	0.0095434625	0.0165432500	0.0049880375	0.0115552125
7	0.0322374900	0.0161099519	0.0161275381	0.0165887900	0.0161099519	0.0004788381
8	0.0105929400	0.0061369526	0.0044559874	0.0165653000	0.0061369526	0.0104283474
9	0.0278774500	0.0107610354	0.0171164146	0.0161570300	0.0107610354	0.0053959946
10	0.0341247600	0.0239173947	0.0102073653	0.0163655000	0.0239173947	-0.0075518947
11	0.0184196700	0.0041546851	0.0142649849	0.0162386700	0.0041546851	0.0120839849
12	0.0350190500	0.0328405853	0.0021784647	0.0159422200	0.0328405853	-0.0168983653

Table 10: SVM Predictions and Differences for LOG and RAW Data

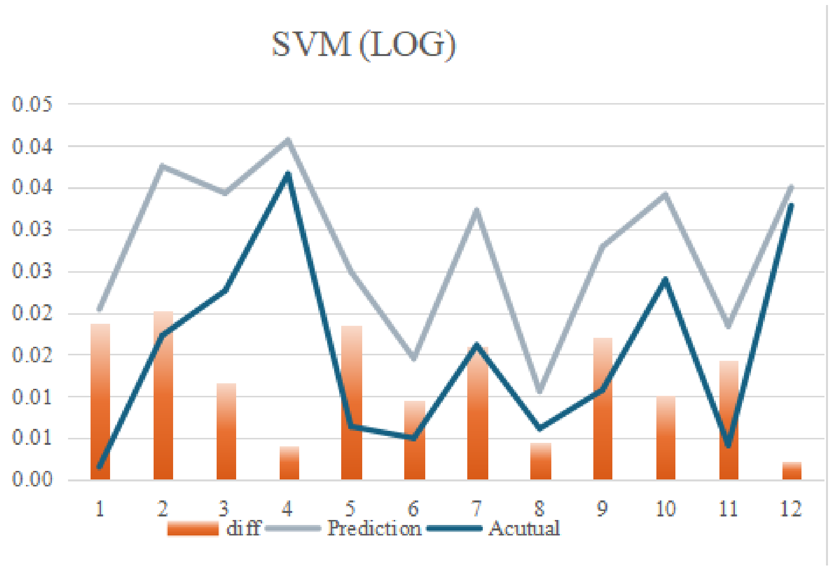

Figure 7: SVM log

Table 10 shows the predicted values, actual values, and their differences of the SVM model under log-differenced features and raw features. From the table, we can see that under raw features, the differences between the predicted values and actual values of the SVM model are relatively small, with most months having prediction errors within 0.01. In contrast, under log-differenced features, the differences between the predicted values and actual values are larger, with many months having prediction errors exceeding 0.01.

4.6 Experimental Results of the XGBoost Model

Table 11 illustrates the predictive performance of the XGBoost model with the raw data and log-differenced features, respectively. The prediction errors using the log-difference feature are significantly smaller than ones using the raw feature, with the MSE not exceeding 0.000005 for Forecast horizons of 1 month, 3 months, and 12 months. In especially, the errors of 12-month prediction are remarkably low under log-differenced features, MSE, RMSE, MAE, and MAPE are 0.0000005979, 0.0007732593, 0.0006222046, and 0.0872399300, respectively. XGBoost Model shows better long-term prediction capability compared with short-term prediction capability, which is different from all the previous models.

Forecast horizons	XGBoost (LOG)			XGBoost (RAW)		
	1 MONTH	3 MONTH	12 MONTH	1 MONTH	3 MONTH	12 MONTH
MSE	0.0000045148	0.0000023595	0.0000005979	0.0002725587	0.0003304254	0.0003226084
RMSE	0.0021247990	0.0015360780	0.0007732593	0.0165093500	0.0181776100	0.0179613000
MAE	0.0021247990	0.0010808480	0.0006222046	0.0165093500	0.0156642700	0.0161801500
MAPE	0.0647004100	0.2164381000	0.0872399300	0.5027119000	0.7507464000	2.4115210000

Table 11: XGBoost Forecast Errors for LOG and RAW Data

Table 12 lists the predicted outcome of the XGBoost model. The difference between the actual value and the predicted value is much smaller under log-differenced data, the difference in forecasts for is below 0.001 apart from the 2nd and 12th months. It further confirms the applicability of log-differenced data in the XGBoost Model.

	XGBoost (LOG)			XGBoost (RAW)		
	Prediction	Actual	Diff	Prediction	Actual	Diff
1	0.0007386045	0.0015188764	-0.0007802719	0.0128005700	0.0015188764	0.0112816936
2	0.0182825960	0.0172782845	0.0010043115	0.0203937000	0.0172782845	0.0031154155
3	0.0228529349	0.0226109489	0.0002419860	0.0147843300	0.0226109489	-0.0078266189
4	0.0362692773	0.0366231446	-0.0003538673	0.0181779100	0.0366231446	-0.0184452346
5	0.0061158831	0.0063830715	-0.0002671884	0.0221263800	0.0063830715	0.0157433085
6	0.0053686206	0.0049880375	0.0003805831	0.0238759600	0.0049880375	0.0188879225
7	0.0152767645	0.0161099519	-0.0008331874	0.0315135200	0.0161099519	0.0154035681
8	0.0067852521	0.0061369526	0.0006482995	0.0373363300	0.0061369526	0.0311993774
9	0.0108378455	0.0107610354	0.0000768101	0.0401208500	0.0107610354	0.0293598146
10	0.0245645791	0.0239173947	0.0006471844	0.0380050300	0.0239173947	0.0140876353
11	0.0045207958	0.0041546851	0.0003661107	0.0231459400	0.0041546851	0.0189912549
12	0.0347072408	0.0328405853	0.0018666555	0.0230206300	0.0328405853	-0.0098199553

Table 12: XGBoost Predictions and Differences for LOG and RAW Data

There is a clear overlap between the predicted and true lines in the XGBoost model, as can be observed in Figure 8. It indicates that the 12-month prediction results are very accurate under the log-difference feature, with negligible difference from the true value. The predictive power of the XGBoost model is outstanding, especially when making long-term predictions. In performing model fitting, our results show that the log-differenced data performs better under the XGBoost model.

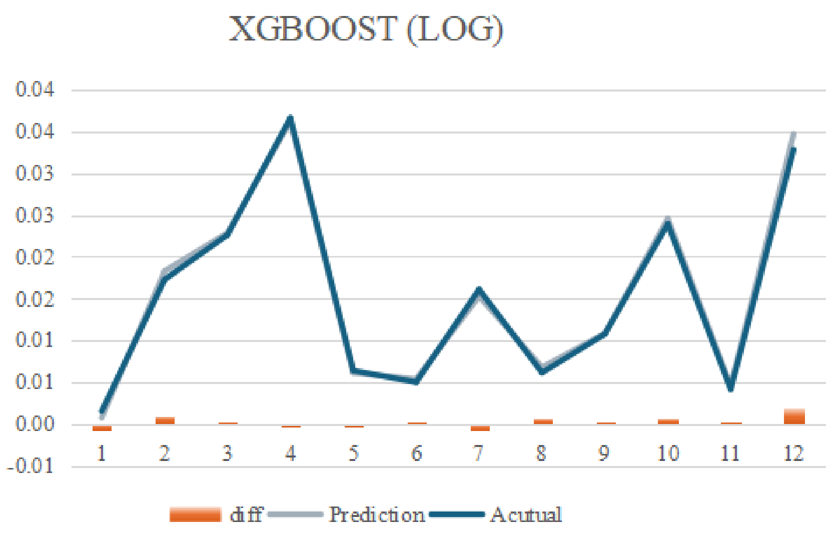

Figure 8: XGBoost

Overall, the predictive power of the XGBoost model is outstanding, especially in making long-term predictions. Comparatively, the XGBoost model is one of the best performing models out of the six models when conducting 12-month forecasts, having the lowest error. In addition, when performing model fitting, the data after logarithmic differencing is more suitable for the XGBoost model.

4.7 Performance Comparison of Different Models

From Table 9, we can more clearly see the performance differences of various models under different prediction horizons. By comparing the evaluation metrics of different models under the same prediction horizon, we can draw the following conclusions:

In the 1-month short-term prediction, the LASSO regression model performs the best, with its MSE, RMSE, MAE, and MAPE significantly lower than those of other models. The Random Forest model performs the second best, also achieving good results. In comparison, the performance of other models is relatively weak, especially the Autoregressive model and Support Vector Machine model.

In the 3-month medium-term prediction, the performance of the LASSO regression model and Gradient Boosting model is relatively close, both achieving low error metrics. The performance of the Random Forest model and XGBoost model is slightly inferior but still better than the Autoregressive model and Support Vector Machine model.

In the 12-month long-term prediction, the LASSO regression model continues to maintain its lead, with its MSE, RMSE, MAE, and MAPE being the lowest among all models. The XGBoost model also performs well, especially in terms of the MAPE metric. The performance of the Random Forest model and Gradient Boosting model is relatively average, while the performance of the Autoregressive model and Support Vector Machine model is relatively poor.

Forecast horizons	ARIMA	Gradient Boosting	LASSO	SVM	Random Forest	XGBoost
1 Month						
MSE	9.560E-06	1.210E-06	1.202E-09	2.988E-06	1.704E-09	4.515E-06
RMSE	3.092E-03	1.100E-03	3.468E-05	1.729E-03	4.128E-05	2.125E-03
MAE	3.092E-03	1.100E-03	3.468E-05	1.729E-03	4.128E-05	2.125E-03
MAPE	6.885E-02	3.350E-02	1.056E-03	5.264E-02	1.257E-03	6.470E-02
3 Months						
MSE	7.355E-04	1.062E-06	2.324E-07	1.038E-04	8.102E-06	2.360E-06
RMSE	2.712E-02	1.030E-03	4.821E-04	1.019E-02	2.846E-03	1.536E-03
MAE	2.409E-02	1.022E-03	3.544E-04	8.700E-03	2.080E-03	1.081E-03
MAPE	3.203E+00	1.026E-01	6.792E-02	1.316E+00	4.040E-01	2.164E-01
12 Months						
MSE	2.045E-04	5.370E-06	3.417E-07	1.865E-04	2.726E-06	5.979E-07
RMSE	1.430E-02	2.317E-03	5.846E-04	1.366E-02	1.651E-03	7.733E-04
MAE	1.140E-02	1.634E-03	5.126E-04	1.227E-02	1.040E-03	6.222E-04
MAPE	5.317E-01	2.539E-01	1.093E-01	2.190E+00	1.935E-01	8.724E-02

Table 13: Forecasting Performance Metrics for Various Models and Horizons

Overall, the LASSO regression model exhibits the most outstanding performance across all prediction horizons, demonstrating its effectiveness and stability in inflation rate prediction. The performance of the Random Forest model and Gradient Boosting model is also relatively stable, achieving good results across different prediction horizons. The XGBoost model performs well in long-term predictions but falls slightly short in short-term and medium-term predictions. The performance of the Autoregressive model and Support Vector Machine model is relatively weak, and they may require further optimization and improvement in practical applications.

5 Discussion and Conclusions

References