Neural Networks for Inflation Rate Forecasting

Marco Jiralerspong
ECON 420
McGill University
marco.jiralerspong@mail.mcgill.ca

Abstract

We test the ability of 3 neural network architectures to forecast inflation rates using the FRED-MD database of macroeconomic variables. We find that the MLP and CNN architectures outperform baseline models once they are tuned appropriately even though they still struggle to outperform ridge regression (with some data modifications).

1 Introduction

Many of the recent innovations in the field of machine learning (ML) have yet to be applied to problems commonly considered in economics. While ML is certainly no holy grail and cannot be used effectively for every problem involving the slightest bits of data, there are certain areas where it can be useful beyond what is provided by more traditional economic tools. These areas include unsupervised learning techniques for creating explanatory/outcome variables, systematic methods for specifying complex models for large datasets and potentially new ML-inspired econometric methods for estimation; other areas such as the identification of causal effects are expected to remain untouched [2] .

In this paper, we examine the potential of ML techniques for the use case of economic forecasting (specifically, the inflation rate). To do so, we use the FRED-MD database, a monthly database stretching from 1959 to today with a large amount of US macroeconomic variables [7] that has widely been used for forecasting and macroeconomic modeling. With the dataset, we build up from simple benchmark models (naive, moving average, linear regression) before eventually moving on to multilayer perceptrons (MLP), convolutional neural networks (CNN) and recurrent neural networks (RNN). The performance of each of these models is compared on the task of inflation rate forecasting. We also analyze the impact of various data transformations and hyperparameter values.

2 Related Work

Much of the previous literature on the topic of inflation rate forecasting either uses a combination of econometric models (ARIMA, RDRMA, etc.) [9], Philips curves [3], Bayesian models [13], or explores the predictive capacity of various external variables [12]. While some papers have also considered the viability of neural network models, the work is often dated (i.e. created before many of the recent advances on the topic) [10] or used comparatively small datasets (at least relative to FRED-MD) [1]. In addition, none of them consider the possibility of using a convolutional neural network (CNN) which, despite seeming like an inappropriate choice for the task at first glance, has shown some viability in times series forecasting [4]. The possibility of using an recurrent neural network (RNN) or long short-term memory (LSTM) is also lacking for larger datasets.

Nonetheless, there has been recent work using the FRED-MD dataset for inflation rate forecast but it doesn't touch on the potential for neural networks, opting instead for older ML methods such as random forests, LASSO models, etc. (with random forests offering the best performance) [8]. Other work using the same database showed that the power of ML for macroeconomic forecasting comes from its ability to represent nonlinear relationships [5].

3 Dataset

The FRED-MD database is a monthly dataset of 128 American macroeconomic variables that contains data from January 1st 1959 to the present day. It is maintained by the Federal Reserve Bank of St. Louis and is designed to be a common starting point for analysis using "big data". It does so by being publicly available with revisions and changes being managed by the bank [7].

As such, it constitutes a great option for the evaluation of neural networks for inflation rate forecasting. While not as large as other datasets that are often used in the ML literature (i.e. downloading it doesn't take a full day), it still has a lot more features than other datasets that have been used for the same task. These additional features should improve the performance of the neural network models as they tend to not suffer from the curse of dimensionality that afflicts simpler methods. The bigger issue is the lack of samples. The time series in the dataset have at most 732 values whereas neural networks are often trained on tens of thousands of examples (e.g. ImageNet has over 1.5 million images), but this is inevitable given the nature of monthly time series.

The features in the database include the following main categories, each of which has multiple values:

- Output/income measures
- Labor market statistics
- Housing permit data
- Consumption measures
- Money/credit measures
- Price data
- Stock market data

As for our specific goal of inflation forecasting, we use the *CPI : All Items* indicator and compute the inflation at period t to be $[\log(\pi_t) - \log(\pi_{t-1})] \cdot 100$ (where π_t is the value of the indicator at period t). This definition follows the one used in [8] (although which CPI indicator they use is not indicated, we use the one for all items as it is the most general and transform it as described above).

3.1 Feature values and transformations

Most of the features in the dataset seem to consist of generally increasing time series. Since many of the features still follow a generally stable linear climb indicating that further feature transformations (e.g. considering the difference with the previous value) might be useful.

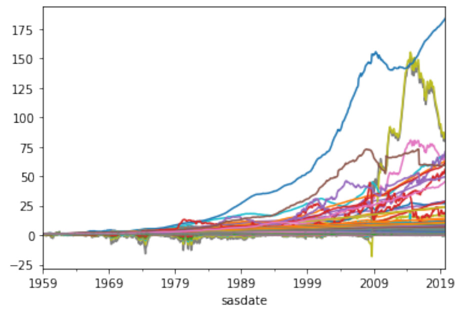


Figure 1: Each feature's time series overlaid, normalized so that they all start at 1

As for the inflation data, it consists mostly of slightly positive values that vary from month to month with a few big dips during recessions.

There are a couple hundred missing values, mostly from a few features that were not available in the first decade covered by the dataset as well as some features that are not yet available in the latest batch of data. While the option of using the mean/median value seems like an easy fix, it doesn't account for many of these features being increasing time series. As such, replacing the missing value

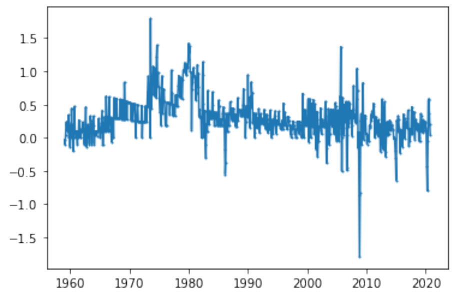


Figure 2: Computed inflation rate over time

at the start of the series with the median value would most likely yield a value that would make more sense in the middle of the series. Instead, we fill the missing values using the next available value which provides a simple and adequate replacement (better options are available but are beyond the scope of this paper). Additionally, we get rid of the year 2020 as the mix of missing data and the pandemic would most likely skew results.

4 Comparison Models

4.1 Baseline Approach

With the data cleaned and transformed, we use 2 baseline models to create useful benchmarks that we can compare further results to. The first we use is the naive model which simply predicts the previous inflation value:

$$\hat{\pi}_t = \pi_{t-1}$$

The second is the simple moving average model which predicts the mean of all values seen so far:

$$\hat{\pi}_t = \frac{\sum_{i=1}^{t-1} \pi_i}{t-1}$$

These models yields mean squared errors (MSE) of **0.071** and **0.069** respectively. With these values in mind, we now proceed to the evaluation of models that make use of the large dataset.

4.2 Simple Regression Models

We begin by using the dataset as is by taking the values of all features at time $t-1$ to predict the inflation value at time t. For each model, we evaluate its performance by performing k-fold cross validation (with $k = 10$). Our first 2 models consist of linear regression and ridge regression using the base dataset and the following data transformations:

1. Using a MinMaxScaler to ensure all features are positive and then taking the log.
2. Differencing the time series.
3. Various lags of all features after applying 1.

The average MSE over all 10 folds in each of these cases is summarized below:

Model \Transformation	None	Scaled log	Differencing	Lag-1	Lag-2	Lag-3
Linear regression	2.077	0.303	**0.221**	0.997	8.795	10.633
Ridge regression	1.548	**0.051**	0.449	0.052	0.055	0.059

5 Neural Network Models

Ultimately, at best with these simple techniques, we get a significant improvement over the baseline models with the best performance coming from the ridge regression model using the log transformed data. This improvement indicates that the other variables included have some predictive power when it comes to forecasting the inflation rate

5.1 Multi Layer Perceptron

We now seek to improve over the results from the linear regression section. We begin with a simple neural network consisting of a 2 hidden layers and 1 output layer with no particular enhancements (i.e. we use the default values suggested by Keras). This shallow MLP has trouble converging to a solution, either performing poorly or diverging to nonsensical values.

To improve the convergence of the model, we apply 5 changes mentioned in [6] that typically improve model performance and stability:

1. Normalizing the features to have 0 mean and unit variance.
2. Using stochastic gradient descent with momentum instead of other optimizers.
3. Implementing early stopping so the model stops training once validation loss increases.
4. Tweaking the learning rate.
5. Adding gradient clipping.

We select 0.0005 as a learning rate as it performs well, is stable and converges relatively quickly. Combining all the improvements above and cross-validating the model yields an average MSE of **0.103**.

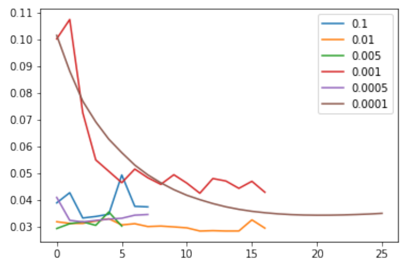


Figure 3: Impact of various learning rates

Now that we have a base model that converges, we see if increasing the depth of the model improves performance. To do so, we add 5 hidden layers of varying size in the middle of the model as well as batch normalization to help prevent overfitting. To better train this larger model, we increase the number of epochs as well as the patience of our early stopping (i.e. how many epochs with no improvement are allowed before we stop training). These combined changes yield a new average MSE of **0.060**, outperforming the baseline models and linear regression but still shy of the best ridge regression model.

5.2 CNN Model

As we switch to considering CNN models, our approach changes slightly. Instead of using the entire dataset, we focus instead solely on the inflation time series. While we could technically performing convolutions on multiple features at once, say by treating the 2D arrays as images, doing so doesn't make much sense as the horizontal relationships between features aren't particularly meaningful. Instead, we split the series into input/output pairs where the goal is to predict the inflation at time t given the previous n inflation rates (n is the sequence size).

Thus, from our initial series, we get $732 - n$ input/output pairs. We train our convolutional network on these pairs as follows. The first layer consists of a one-dimensional convolution which passes a kernel over the sequence to generate its output. The output is passed through a max pooling layer which is then flattened and used as input to a hidden layer. Finally, the output layer takes the output of the hidden layer and uses it to generate the predicted inflation rate.

An important hyperparameter for this simple model is the sequence size n. We compare different values below before eventually settling on 8. The resulting model gets a cross-validated MSE of **0.062** (a similar performance to that of the MLP using much less data).

With these promising results, we proceed to evaluating a deeper version of the model that resembles a simplified version of WaveNet [11] as specified in [6]. To do so, we use multiple convolutional layers in sequence, each with increasing dilation rates (meaning the inputs to the layers are more spread

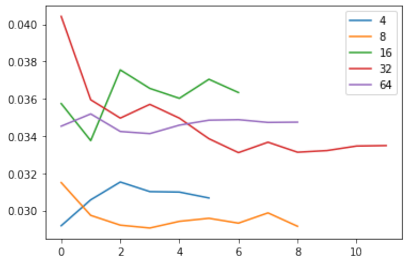
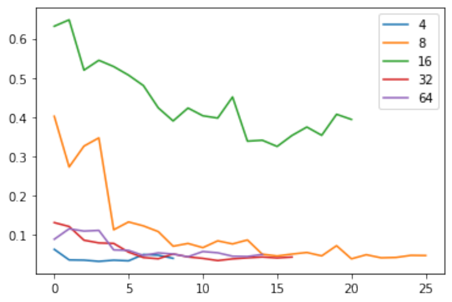


Figure 4: Impact of various sequence sizes

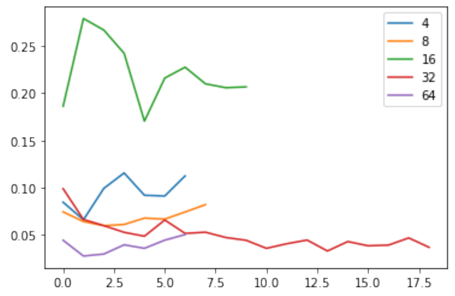
apart as we go deeper in the model). No max pooling is done but the model ends once again with a hidden layer followed by an output layer. Unfortunately, the model seems to be too complicated for the task as it performs worse than our simplistic initial model (cross-validated MSE of **0.067**).

5.3 RNN Model

For our RNN models, we somewhat combine our 2 previous approaches. As in the CNN section, we split the data into subsequences except that this time, we use all features as we did for the MLP models. Our very first model consists simply of a single RNN layer followed by an output layer. Once again, sequence size is an important hyperparameter so we compare the performance of different values. Interestingly, for the RNN, the longer sequences as well as the very short sequence seem to perform better, contrasting what we found for CNN architectures. We also perform the same experiment with the same model except that the RNN layer is replaced with a LSTM layer.



(a) Sequence size impact on RNN



(b) Sequence size impact on LSTM

Once again, we get the same result indicating that longer sequences are better, with the best performance coming from sequence sizes of length 64. Using a sequence size of 64, we compare the cross-validation performance of both models. The LSTM architecture performs significantly better as they get average MSEs of **0.415** and **0.135** respectively. This difference indicates that the addition of memory is an important component for the task. Now that we have a good base model, as we've been doing consistently, we proceed to make it deeper and see how that affects performance. We add 3 more LSTM layers as well as another dense layer before the output layer and get a performance improvement of **0.086** (still shy of the performance of the baseline models.

6 Conclusion

Ultimately, while the various recent advances in neural network architectures can be used for inflation forecasting, they struggle to outperform simpler models such as ridge regression with some data transformations. Nonetheless, they still manage to outperform naive models and linear regression, with both a MLP and CNN architectures achieving respectable MSEs of around **0.06** when cross-validated. However, to reach these results, care must be taken to take measures that improve the likelihood of convergence such as batch normalization, early stopping, etc.

There is still a lot of room for improvement. Hyperparameters have only been optimized very simply and a more systematic approaches such as grid search could provide even further performance improvements. Additionally, more complicated models that combine all three techniques in various manners could be promising (even a simple meta-model that combines the output of all 3 techniques would surely yield lower MSE as the different approaches should yield somewhat uncorrelated results). Analyses that include similar data from other countries could also be used to bolster the size of the training set, allowing for better training of the various models. Also, many of the models performed very well on certain intervals but poorly on others, indicating that removing certain periods of financial crisis could yield results that were less skewed. Finally, more robust or creative forms of feature engineering could also be used to improve performance.

References

[1] A. Almosova and N. Andresen. Nonlinear inflation forecasting with recurrent neural networks, 2019.

[2] S. Athey. The impact of machine learning on economics. In *The Economics of Artificial Intelligence: An Agenda*, pages 507–547. National Bureau of Economic Research, Inc, 2018.

[3] A. Atkeson, L. E. Ohanian, et al. Are phillips curves useful for forecasting inflation? *Federal Reserve bank of Minneapolis quarterly review*, 25(1):2–11, 2001.

[4] A. Borovykh, S. Bohte, and C. W. Oosterlee. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

[5] P. G. Coulombe, M. Leroux, D. Stevanovic, and S. Surprenant. How is machine learning useful for macroeconomic forecasting? *arXiv preprint arXiv:2008.12477*, 2020.

[6] A. Géron. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.

[7] M. W. McCracken and S. Ng. FRED-MD: A Monthly Database for Macroeconomic Research. Working Papers 2015-12, Federal Reserve Bank of St. Louis, June 2015.

[8] M. C. Medeiros, G. F. Vasconcelos, A. Veiga, and E. Zilberman. Forecasting inflation in a data-rich environment: the benefits of machine learning methods. *Journal of Business & Economic Statistics*, pages 1–22, 2019.

[9] A. Meyler, G. Kenny, and T. Quinn. Forecasting irish inflation using arima models. 1998.

[10] S. Moshiri, N. E. Cameron, and D. Scuse. Static, dynamic, and hybrid neural networks in forecasting inflation. *Computational Economics*, 14(3):219–235, 1999.

[11] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[12] J. H. Stock and M. W Watson. Forecasting output and inflation: The role of asset prices. *Journal of Economic Literature*, 41(3):788–829, 2003.

[13] J. H. Wright. Forecasting us inflation by bayesian model averaging. *Journal of Forecasting*, 28(2):131–144, 2009.