2019

Can Machine Learning on Economic Data Better Forecast the Unemployment Rate?

Aaron S. Kreiner
Oberlin College

Can Machine Learning on Economic Data Better Forecast the Unemployment Rate?

Aaron Kreiner
Oberlin College and Nomura Securities
aaron.kreiner@oberlin.edu

May 17, 2019

Abstract

This paper examines different machine learning methods to project the U.S. unemployment rate one year ahead. The forecasts include a "naive forecast" equal to the current unemployment plus the change of unemployment over the last year, along with forecasts from a Lasso regression and a neural network model. The last two models, which can be quickly run using an SQL database, select data from the Federal Reserve Economic Database (FRED) and are fitted ("trained") in-sample from 1970 to 2000 to forecast quarterly unemployment rates over 2001 to 2018. The training window is updated in each forecast quarter to include new data. A rolling-window and non-rolling window period are tested for the training window. This paper finds that a non-rolling neural network model forecasts bests and outperforms the Survey of Professional Forecasters (SPF) across all time periods as does our Lasso regression model, though to a lesser extent. From experiments dropping broad categories of FRED, international data were the most important in forecasting the unemployment rate, followed in order by data from the FRED categories: Population, Employment, Labor Markets; and Money, Banking, and Finance.

JEL Codes: C45, C10, E24

Key Words: machine learning, forecasting, neural networks, artificial intelligence, unemployment rate

* I thank Barbara Craig, Ed McKelvey, and John Duca along with other Oberlin faculty and Oberlin student honors seminar participants for suggestions.

1. Introduction

Forecasting the unemployment rate has remained a challenge for decades because conditional errors of forecasts rely on the overall strength of economic growth. Forecasting increases or declines in unemployment would be important in capturing shifts in growth relative to a time-varying, underlying trend in potential real GDP growth. To capture this change in the relative growth rate, this paper implements a four-quarter ahead forecast for the seasonally adjusted, civilian unemployment rate. Starting from an in-sample training period of 1970-2000, we start forecasting the unemployment rate four-quarters ahead for 2001:q4 and then progressively add one quarter more data to project the next quarter over 2001:q4-2018:q4. We examine three types of forecasting models, including a "naive forecast" equal to the current unemployment plus the change of unemployment over the last year, along with forecasts from a Lasso regression and a neural network model, the latter two of which can be quickly run using an SQL database.

The latter two models use machine learning techniques to select data from a large number (over 600,000) of macroeconomic variables in the Federal Reserve Economic Database (FRED), ranging from measures of aggregate economic activity such as the money supply, to city-level data. We find that a non-rolling neural network model forecasts bests and outperforms the Survey of Professional Forecasters (SPF) across all time periods as does a Lasso regression model, though to a lesser extent. From experiments dropping broad categories of FRED, international data were the most important in forecasting the unemployment rate, followed in order by data from the FRED categories: Population, Employment, Labor Markets; and Money, Banking, and Finance.

To establish these findings, the paper is organized as follows. Section 2 reviews relevant literature on machine learning for forecasting unemployment and using marginal information that forecasters may ignore. Section 3 details the data selection methodology that is applied to the

FRED Database. Section 4 begins by providing a theoretical overview of the machine learning techniques used—including principal components analysis, Lasso regression, and finally neural networks—before outlining the forecasting algorithms for each statistical model. The results are presented in Section 5, with perspective and broader lessons discussed in the conclusion.

2. Background

2.1 Literature Review

A number of approaches have been used in the literature to project the unemployment rate. Of the many papers dealing with this topic, we focus on the few most relevant. We first review econometric approaches used to project the unemployment rate, and then discuss the use of machine learning projections in more detail.

2.1.1 Econometric Approaches

Several papers use conventional econometric techniques to see if they can out-forecast some average, benchmark set of forecasts from economists. Typically, averages from the Survey of Professional Forecasters (SPF) are used for a benchmark as the SPF has consistently recorded quarterly forecasts from an unbalanced survey of 50 economists covering consistently spaced forecast horizons since 1968.

One of the earliest to beat the SPF was Montgomery et. al. (1998) who use a leading indicator (initial jobless claims) along with the current and lagged version of the unemployment rate in univariate models, multivariate linear models, threshold autoregressive models, and Markov switching autoregressive models. Of these, only the threshold autoregressive and Markov switching autoregressive models outperform the SPF on all horizons and time periods tested.

Barnichon and Nekarda (2012) incorporate labor force flows and exploit the convergence of the unemployment rate to its steady state. Their forecast of labor force flows determines steady-

state unemployment and the speed at which the steady state converges to the actual unemployment. By calculating rolling labor force flows, they project unemployment one through four quarters ahead, and with their forecasts providing an average 30% improvement over the SPF and outperforming the forecasting models of Montgomery et. al. (1998) as well.

While Barnichon and Nekarda use theory-based calculations for forecasting, Meyer and Tasci (2015) use autoregressive models, finding flows into and out of unemployment inform short-term unemployment forecasts. For longer horizons, predictions based the natural rate perform better. Their forecasts for both horizons also beat the SPF. With respect to RSME, Barnichon and Nekarda (2012) provide the best forecasts up until 2012, but it is unclear whether their model would continue to do so if forecasts were extended to 2018.

2.1.2 Machine Learning Approaches

Of the few published studies using machine learning techniques to forecast unemployment, three are most relevant to our study. Two Sigma (2016) built a machine-learning model using New York City (NYC) taxi data to project the NYC unemployment rate. That study considered variables such as taxi wait times, number of passengers per day, and taxi revenue from thousands of variables and observations in their data set. Principal component analysis was applied to the data and neural networks were used to project the quarterly NYC unemployment rate. Xu et. al. (2013) used search engine query data from Google to predict the U.S. unemployment rate. Factors included the perceived strength of labor markets and job reports based on google searches. These factors were placed into neural networks and decision trees, the latter of which employs machine learning to weight factors based on a recursive tree structure dependent on input data. Xu et. al. (2013) found their model slightly outperformed the mean SPF forecast.

Most similar to our paper is Cook and Hall (2017), who use neural networks based on a single macroeconomic indicator—monthly lags of the civilian unemployment rate to forecast the unemployment rate 1-, 3-, 6- and 12 months ahead using 20 years of unemployment data. Their model outperforms the SPF over short horizons (1- and 3- months ahead), but not over longer 6 and 12 month horizons. Our study is similar to Cook and Hall's in using a neural network approach, but goes further by using machine learning techniques to draw on many variables.

2.2 Economic Significance of Machine Learning

New machine learning models could have widespread use and have a large impact on economics. According to Mullainathan and Spiess (2017), most econometric models focus on the best estimates for coefficients on regressors (the β's in the in-sample data, whereas machine learning focuses on explaining the independent variable $\hat{y}$ on out-of-sample data. In other words, econometricians focus on the weight of factors in-sample, whereas machine learning focuses on the out-of- sample performance in predicting the dependent variable. For these reasons, machine learning is best suited for economic analysis that requires high out of sample performance, such as forecasting. Some examples of machine learning in economics include measuring economic activity using satellite images, classifying industries based on their 10-K filings, and building forecasts for core economic indicators. (Mullainathan and Spiess, 2017)

Our paper focuses on out-of-sample forecast accuracy, showing that machine learning provides superior out-of-sample forecasts of the unemployment rate relative to the mean forecast from the SPF. Machine learning forecasts can therefore be used by economists and traders to price labor market activity. While this method cannot produce a set of β's, this is compensated by adding and dropping key data categories and testing out of sample performance before and after omitting sets of data. The use of this methodology indicates which types of variables have the largest benefit

to improving forecasts. If omitting a set of variables causes the out of sample error to increase the most, then the machine learning algorithm puts the highest weight on these variables.

3. Data

We draw data from the Federal Reserve Economic Data (FRED) database, which contains about 600,000 variables that are binned into categories labelled with a category number. Categories and data can lie within other categories. In this way, FRED is structured as a recursive tree database with categories and data stored in trees. For example, category 0 contains 8 of the prime categories in FRED: Money, Banking, and Finance; Population, Employment, and Labor Markets; National Accounts; Production and Business Activity; Prices; International Data; US Regional Data; and Academic Data. Table 1 summarizes which types of data are in each of these prime categories. These categories will be used to assess which types of variables significantly affect unemployment. The algorithm for scraping the data identifies all the categories by making recursive calls to the FRED tree structure. Next, all of the variables are collected from each category. Data are the actual time series for a particular variable. Only data series that meet the following criteria are retained in the final model:

1. *Frequency:* The variable must be monthly or quarterly. If a variable has a frequency higher than monthly, the series is converted to a monthly series using FRED's conversion algorithm. This criterion accommodates the vector construction of inputs and outputs. The forecasting algorithm section will discuss this further.
2. *Consistency:* The variable must have consistent data from January 1970 to September 2018. The data series cannot have any missing values. This criterion ensures the forecast uses variables that are continuously available.

Category	**Types of Variables Included**

Money, Banking, and Finance	Interest Rates, Exchange Rates, Monetary Data, US Financial Indicators, Banking, Business Lending, Foreign Exchange Intervention
Population, Employment, and Labor Markets	Current Population Survey (Household Survey) , Current Employment Statistics (Establishment Survey) , ADP Employment, Education, Income Distribution, Job Openings and Labor Turnover (JOLTS), Labor Market Conditions, Population, Productivity and Costs, Minimum Wage, Weekly Initial Claims, Tax Data
National Accounts	National Income and Product Accounts, Federal Government Debt, Flow of Funds, US Trade and International Transactions
Production and Business Activity	Business Cycle Expansions & Contractions, Construction, Finance Companies, Health Insurance, Housing, Industrial Production and Capacity Utilization, Manufacturing, Retail Trade, Services, Technology, Transportation, Wholesale Trade
Prices	Commodities, Consumer Prices Indexes (CPI and PCE), Cryptocurrencies, Employment Cost Index, Health Care Indexes, House Price Indexes, Producer Price Indexes, Trade Indexes
International Data	Countries, Geography, Indicators, Institutions
US Regional Data	States, Census Regions, BEA Regions, BLS Regions, Federal Reserve Districts, Freddie Mac Regions
Academic Data	Banking and Monetary Statistics 1914-1941, Data on the nominal term structure model from Kim and Wright, Historic Federal Reserve Data, NBER Macrohistory Database, Penn World Table 7.1, Penn World Table 9.0, Recession Probabilities, Weekly US and State Bond Prices, 1855-1865, Economic Policy Uncertainty, Sticky Wages and Comovement, A Millennium of Macroeconomic Data for the UK.

Table 1: Summary of the Types of Variables in FRED.

3. *Revision History*: The variable's data are from the first revision if available. Otherwise, use the revision that was first released. This minimizes the use of information available in the future. This represents the first figure published for the first revision available.
4. *Forecasts*: All variables that are from a forecast are omitted to preclude the use of future information and ensures that the model does not gain performance from an outside forecast.
5. *Discontinued Series*: If a series is discontinued, then it is omitted. This criterion ensures the forecast uses variables that are currently available.
6. *Lagged*/*Leading Variables:* If a variable is lagged or from the future, it is omitted to avoid using future information and adding noise from time series reported at the wrong date.

The dependent variable is the seasonally adjusted civilian unemployment rate. Other filtered variables are the independent variables, including lags of the unemployment rate. The current and four lags of unemployment are used to forecast unemployment four quarters later. After applying the above criteria, most variables are dropped. Table 2 summarizes how many variables in each prime category were considered and selected. On average, fewer than 2% of the variables in each category met the filtering criteria. In particular, none of the academic variables were selected while 17% of the national accounts variables were selected (the highest in percentage terms). In absolute terms, international variables had the highest number of variables in the final model.

Finally, the mean forecast from the Survey of Professional Forecasters (SPF) compiled by the Federal Reserve Bank of Philadelphia is used as a benchmark to gauge the performance of each machine learning model. The SPF is an unbalanced survey of 1-year ahead forecasts from 50 independent forecasters collected over 1970-2018 on a quarterly basis.

Category	Number of Variables Before Filter	Number of Variables After Filter	Survival Rate
Money, Banking, & Finance	9,424	235	2.49%
Population, Employment, & Labor Markets	25,313	906	3.58%
National Accounts	17,435	2,919	16.74%
Production & Business Activity	11,119	844	7.59%
Prices	14,158	769	5.43%
International Data	158,700	4,356	2.74%
US Regional Data	337,176	617	0.18%
Academic Data	15,642	0	0.00%
Total	588,967	10,646	1.81%

Table 2: Number of Variables in Each Category and Selected for the final model.
(Sources: FRED and authors' calculations. "Survival Rate" measures the percent of variables per category that were selected for the final model.

4. Theoretical Models

This section summarizes and details the mathematics and algorithms behind the statistical models used in the forecasting algorithm section. In particular, principal components analysis is discussed first as a black box that reduces the dimensionality of the variables used for the forecast. Next, neural networks are presented as a highly non-linear black box for forecasting, which relies on computational techniques that mimic those used in the human brain for processing information. Finally, a Lasso regression is presented for forecasting that relies on variable selection (or omitting irrelevant variables) and a dynamically sized error term.

4.1 Principal Components Analysis

Principal components analysis (PCA) is a dimensionality reduction algorithm that inputs a dataset and outputs another with fewer variables and linearly independent data columns. Principal components analysis is used to reduce multicollinearity and redundancy in the variables of the forecasting model. Failure to do so causes overfitting and contributes noise to forecasts from the final model. Subsections 4.2 and 4.3 provide more details on the black boxes used in forecasting.

In implementing PCA, the convexity of the problem solved by the PCA algorithm ensures that the computer can solve the problem efficiently (see Crawford, 2015).

4.2 Lasso Regression

A Lasso regression can use the data to create unemployment forecasts. A Lasso or "least absolute shrinkage and selection operator" is a type of regression that uses both variable selection and regularization to maximize prediction accuracy and coefficient interpretation. Regularization refers to techniques that reduce noise in the training data. A Lasso regression can assign variables a weight of 0 (even if they carry variance) and thus has a dimensionality reduction element embedded within it. This aids in the interpretation of estimated coefficients by reducing the number that need to be interpreted. A Lasso regression also has regularization as a feature. This allows the weights of the regression to not be too high or too low arising from noise or over-fitting.

The mathematics behind the Lasso Regression are similar to those of a linear regression. Suppose we have independent variable data X with k factors and dependent variable Y. X and Y are expressed in matrix notation. The Lasso minimizes the sum of squared errors with an upper bound on the sum of the absolute values of the model parameters (or regularization of the parameters). Let β be the matrix of coefficients for this regression. The Lasso, therefore, solves the following optimization problem:

$$min\ \frac{||Y - X\beta||_2^2}{n} \tag{1}$$

$$\text{subject to: } \sum_{j=1}^{k} ||\beta||_1 < \text{t} \tag{2}$$

We note eq. (1) is equivalent to ordinary least squares. $||Y - X\beta||_2^2 = \sum_{i=0}^{n}(Y_i - \beta X_i)^2$ or the sum of squared residuals and $||\beta||_1 = \sum_{j=1}^{k} |\beta_j|$. t is the upper bound for the absolute sum of the coefficients. The smaller the value of t, the smaller the absolute size of the coefficients will be

in the final regression. t therefore controls for overfitting and noise by ensuring that any one coefficient is not too large. Equations (1) and (2) can be rewritten in terms of a new parameter: λ. Shrinkage refers to the degree the magnitude of the coefficients can be reduced as compared to OLS. When λ=0 it corresponds to no shrinkage and t equals infinity; this is OLS. λ must be non-negative because shrinkage refers to the absolute magnitude of the coefficients, which must be positive. Using this theory, the optimization problem can be re-written as follows:

$$\hat{\beta}(\lambda) = \underset{\beta}{\operatorname{argmin}} \left(\frac{\left|\left|Y - X\beta\right|\right|_2^2}{n} + \lambda ||\beta||_1 \right) \quad (3)$$

Equation (3) represents the *optimization* problem for $\hat{\beta}$ as a function of the shrinkage parameter λ and the matrix of unknown β's. Equations (1) and (2) describe the same optimization problem as equation (3), but λ is more widely used in the literature and econometric packages as compared to t. As is the case with principal components and OLS, the Lasso regression has a convex optimization function and thus estimates for $\hat{\beta}$ are relatively easy to find. In the optimal solution, a $\hat{\beta}_j$ can be zero meaning the weight is also zero. The shrinkage parameter allows series with non-zero variance to have the potential to have the weight be zero.

Even though the Lasso regression has some embedded functionality to deal with over-fitting and noise, reducing these two characteristics is ideal before the Lasso regression is run. In the context of the Lasso regression, the idea of zero weight on some factors helps reduce overfitting. However, to guarantee overfitting does not occur, it is best to have more rows of data than factors. For this reason, it is recommended that a dimensionality reduction algorithm like PCA be used before using a Lasso regression (see Fonti 2017). This is discussed further in Section 5. A Lasso regression's objective function is convex and runs quickly, making it appealing to analyze big data.

4.3 Artificial Neural Networks (ANN's)

An artificial neural network (or ANN) is a statistical computational system modeled after biological neural networks in the brain. The ANN "learns" by being given examples that pair inputs to outputs. These pairs are called the "training data" or the "fit data". In contrast to OLS or a Lasso regression, the form of the model is not specified beforehand. While an optimization procedure is used, the actual model is dynamic and changes depending on the input. In this way, ANN's are highly non-linear and can differ drastically depending on the nature of the data. While it is more difficult to interpret the weights in these models, they potentially offer a higher degree of accuracy in forecasting, which is ideal since the goal is to find the model with the lowest root mean squared error for forecasting out-of-sample or "testing" data. In-sample fit and the interpretation of weights on the fit data are secondary to the model's ability to forecast the test data.

Because an ANN is constructed based on brain neural activity, it is helpful to start by considering a rudimentary model of the brain. For a simple example and applying some key terms associated with ANN's, consider the ubiquitous depiction of thinking in Figure 1. Suppose Figure 1 models the brain's decision of whether to walk left or right. The brain has five inputs, denote them a, b, c, d, and e. Each could represent a binary variable, such as whether a wall is immediately to our left or right. There is only one output, which is the trinary variable: 0 if going right, 1 if going left, and 2 if doing nothing. Each circle depicts a neuron, a computational unit that takes in a series of inputs and produces an output. Consider the bottom-most circle closest to "inputs." This neuron has five inputs and it outputs its computed value to four neurons in the middle column. The neurons in the middle column takes those outputs as inputs, and then computes their output, which it provides to the final neuron. This final neuron then makes the final computation and produces the final output.

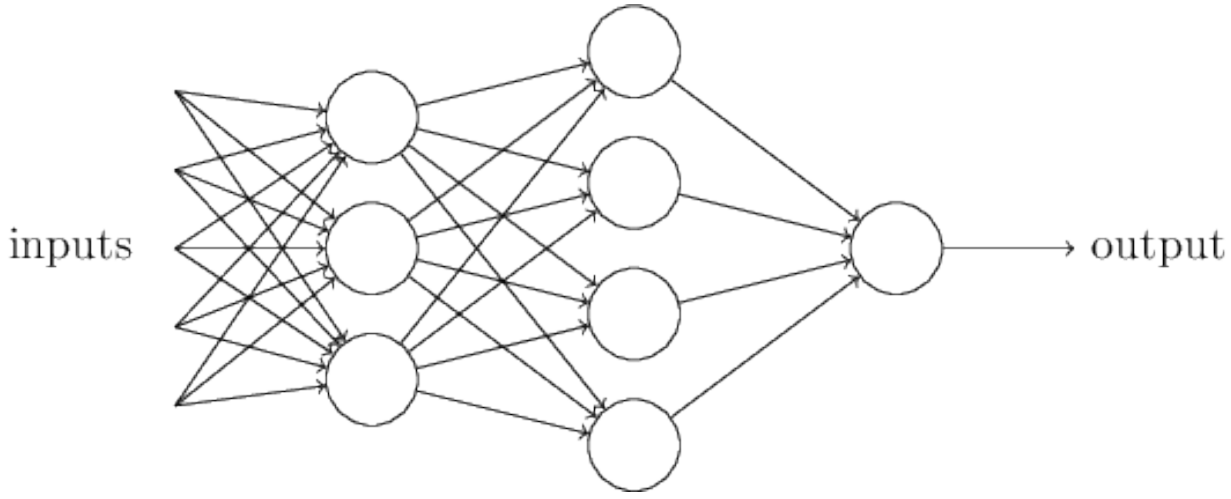

Figure 1: A Schematic of Human Brain Analytical Processes

There are several key terms from this example. The three circles closest to "inputs", the next four circles, and then the circle closest to output form "hidden layers." These are the computational units that transform inputs to outputs. The rule that converts a neuron's input to an output is considered to be an activation function. Because a neuron is a simple processing unit, the activation function cannot be overly complex. The non-linearity of the network derives from the repeated iteration, clustering, and connection of neurons. Each ray in the diagram connects a neuron to another neuron. Consider an arbitrary connection of neuron *a* to neuron *b*, where neuron *a* is an input to neuron *b*. The output of neuron *a* has an attached weight in the propagation function of neuron *b*. This is true for every connection of neurons in the diagram. One can therefore associate a set of weights with the above network.

Each neuron also has an associated bias that measures the intercept of the propagation function. The weights determine the slope of the parameters of the activation function while the bias determines the intercept. This is needed to provide optimal output at each layer of the network.

To provide more details on the calculation embedded in an arbitrary neuron in a neural network, consider an arbitrary neuron: j. Neuron j receives the outputs $o_{i_1}, o_{i_2} \ldots, o_{i_n}$ from neurons

$i_1, i_2, \dots, i_n$ (which are connected to neuron j). The propagation function of neuron j is defined as the transformation of each of the outputs with respect to the weights into one network input. This is the weighted sum of the outputs from neurons $i_1, i_2, \dots, i_n$ and the weights $w_{i_1,j}, w_{i_2,j}, \dots, w_{i_n,j}$. $w_{i_1,j}$ represents the weight from neuron $i_1$to neuron j. Let $I = \{i_1, i_2, \dots, i_n\}$. Let the output of the propagation function be net_j. net_j is therefore:

$$net_j = bias_j + \sum_{i \in I} o_i * w_{i,j} \quad (4)$$

In equation (4), net_j is a number with singular dimension and $bias_j$ is the calculated intercept. The propagation function thus reduces the dimensionality of the inputs of neuron j from n to 1. Next, the activation function transforms net_j into a probability between 0 and 1. This paper uses Python for the implementation of ANN's.

There are four types of activation functions used in Python's artificial neural networks library: *Identity*, *Logistic*, *Tanh*, and *Relu*. These functions take the output from the propagation function and transform it into the output of the overall neuron. These functions determine if a neuron fires. In other words, a neuron firing determines if the neuron is important in fitting the in-sample data. The functions are described below:

1. *Identity*: f(net_j)= net_j. net_j is rounded to 1 if $net_j > 1$ or rounded to 0 if $net_j < 0$. This is the simplest of activations.
2. *Logistic:* f(net_j)= $\frac{1}{1+e^{-net_j}}$. This is the most commonly used activation as it has the best success on most datasets as compared to the other three activation functions. The function is a sigmoid with asymptotes at 0 and 1.
3. *Tanh*: f(net_j)=tanh (net_j). Once again, this activation has asymptotes at 0 and 1. The curvature of tanh is less steep than the sigmoid, meaning each activation probability is

lower than the sigmoid.

4. *Relu*: A rectified unit linear function is f(net_j)=max $(0, net_j)$. If this quantity exceeds 1, then the output is rounded down to 1.

From these, tests select which activation function provides the greatest performance on the out of sample data as gauged by the RMSE. Each of these activations returns a probabilistic value from 0 to 1. We say this neuron *fires* if the probability is greater than the predetermined threshold value. It can be shown that the threshold value in a given neuron is the maximum gradient of the activation function. A firing neuron represents that the inputs to neuron j have weight on the inputs of the neurons in the next hidden layer. The output function passes the neuron's firing state to the next neuron as an input. It is important to note that the neuron outputs a singular firing state, but sends this output to multiple neurons in the next hidden layer. This process is depicted in Figure 2, which modifies a chart from Kriesel's (2009).

However, it is important to note the final output (which is the unemployment rate in this study) is a continuous positive number. The neurons before the output have activation functions that output real numbers (such as *ReLu*), instead of being restricted between 0 and 1. This is known as MLP Regression, or multilayer perceptron regression.

The above paragraphs describe the calculation behind neural networks. We now describe the process of obtaining weights and biases within the network by explaining the algorithm behind the MLP regression and the calculation of weights and biases within a neural network. An optimization procedure is used that minimizes the error between the predicted output values of the training data and the actual output values. This can be expressed mathematically as follows. Let there be *p* pairings of inputs and outputs. Let *x*(i) be a n-dimensional vector for the i-th input and d(i) be a singleton for the actual output of *x*(i). In addition, let *w* be the unknown weight matrix of

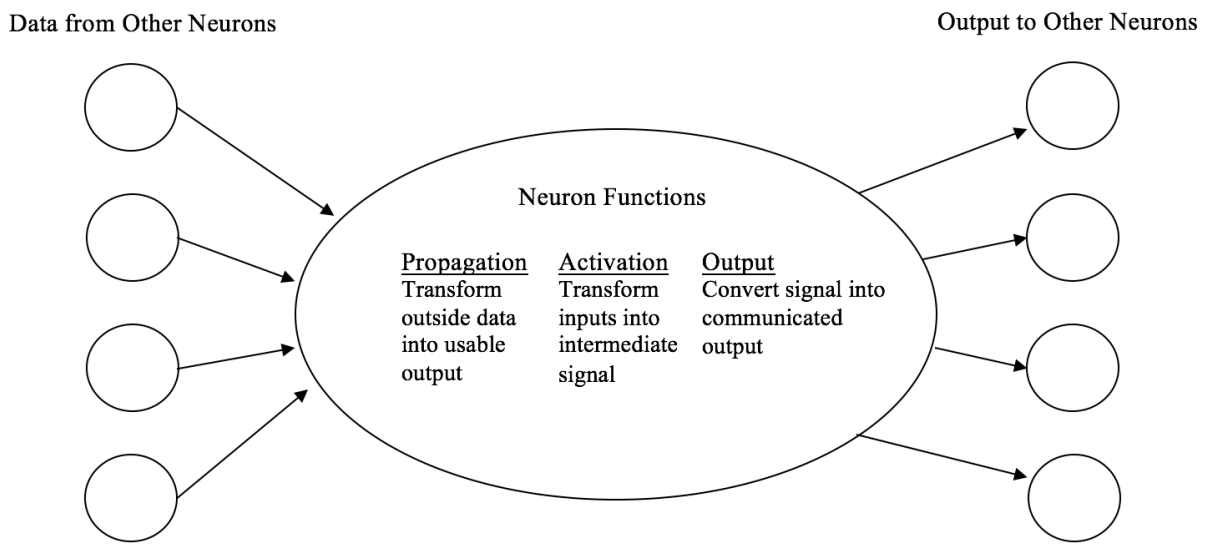

Figure 2: Numerical Processing in a Given Neuron
(Note: each of the data outputs are the same, and then enter as inputs to the neurons in the next hidden layer.)

the ANN and let b be the unknown bias network of the ANN. Thus, we would like to minimize the squared error of the predicted output as per the ANN and the actual output. The predicted output can be calculated per the methodology in the above paragraph. If $y(x: w; b)$ denotes the predicted output of x conditional on w and b, the objective function is:

$$min \sum_{i=1}^{p} \|y(x(i): w; b) - d(i)\|^2 \qquad (5)$$

The y function is dependent on the activation of the neurons in the ANN, the number of hidden layers, and the number of inputs. These determine how large the optimization problem is. However, the optimization function is convex, making it easily solvable on a computer. This process generates the weights and biases for the ANN based on the training data. We can then take the fit data, using the methodology described above, to calculate the predicted outputs. In this way, neural networks learn based on the training data and forecast based on the fit data (see Kriesel, 2009).

As a collective, neural networks are a highly non-linear black box that operates in a similar fashion to neural networks in the brain. Networks are comprised of neurons, which contain propagation, activation, and output functions for computation. The choice of activation function depends on the data and maximizes the performance on the out of sample data. Neurons are connected through a matrix of weights and biases, which weight neurons' outputs into inputs into neurons in the next hidden layer. To obtain the weights and biases, an optimization procedure minimizes the squared error of the predicted output versus the actual output on each of the vectors in the training data.

5. An Overview of Forecasting Algorithms

This section summarizes how forecasts are calculated for each model type. The models include: the "naive" forecast, the neural network, and Lasso Regression. Each model's forecast performance is assessed relative to the benchmark, mean from SPF forecasts.

5.1 The "Naive" Forecast

The naive forecast of the unemployment four quarters from now is the current unemployment plus the change of unemployment over the prior four quarters.

5.2 Neural Networks

The forecasting methodology for both machine learning models are depicted in Figure 3. The table at the top of the figure provides the data for the forecast. Each row represents a quarter of data, where the independent variables span from 1970:q1 to 2017:q4. In this figure, there are two types of variables: quarterly (Y) and monthly (X). While the table has only two variables, there are thousands more independent variables in the actual model. Only two are presented for ease of reading the figure. Quarterly data are placed in a row where the "Data Date" matches the reported date of the variable. For example, if Y=1 in 1970:q1, it will be placed in the first empty

square. Monthly variables are slightly more complicated. Within a quarter there are three months. Therefore, each monthly independent variable will occupy 3 columns, each of which can be considered a separate variable. The first column represents the first month of the quarter, the second column the second month of the quarter, and so on. For instance, suppose the monthly variable X has 1970:q1 data 5,6,7. Each of these numbers is placed consecutively in the appropriate month. This then produces Table 3 for the independent variables for 1970:q1. Using this methodology, the entire table can be filled for all the quarters and independent variables. The final column is the dependent variable. This column is a one-year ahead forecast of the independent variables because this paper forecasts the unemployment rate one year ahead.

The first forecast uses data from 1970:q1 to 2000:q3 to forecast 2001:q3. First, the independent variable data from this time period are put into a PCA algorithm with a variance threshold of 0.99. The algorithm is a preprocessing step for the actual forecasts. Then, one year's worth of lags is added for each column of independent variable data. These are appended to the dataset as separate columns for each variable. The forecast takes the training data from 1970:q1 to 2000:q3 and trains the model. The weights, biases, and other metrics (depending on the type of model) from the 1970:q1-2000:q3 data are used to forecast the unemployment rate. The above process (including the PCA algorithm) is repeated for each forecast. For the non-rolling window variant, the beginning of training is fixed. In other words, the first pair of inputs and output will always be 1970:q1. In the rolling window variant, the beginning of training increases by one quarter for each forecast. This means that the second forecast has training start in 1970:q2 instead of 1970:q1. The rolling window, therefore, has a fixed number of rows of data for training. Both types of forecasts will roll in one quarter of new data as available. In particular, for the second forecast, 2001:q4 can now be added to the independent variables of the model.

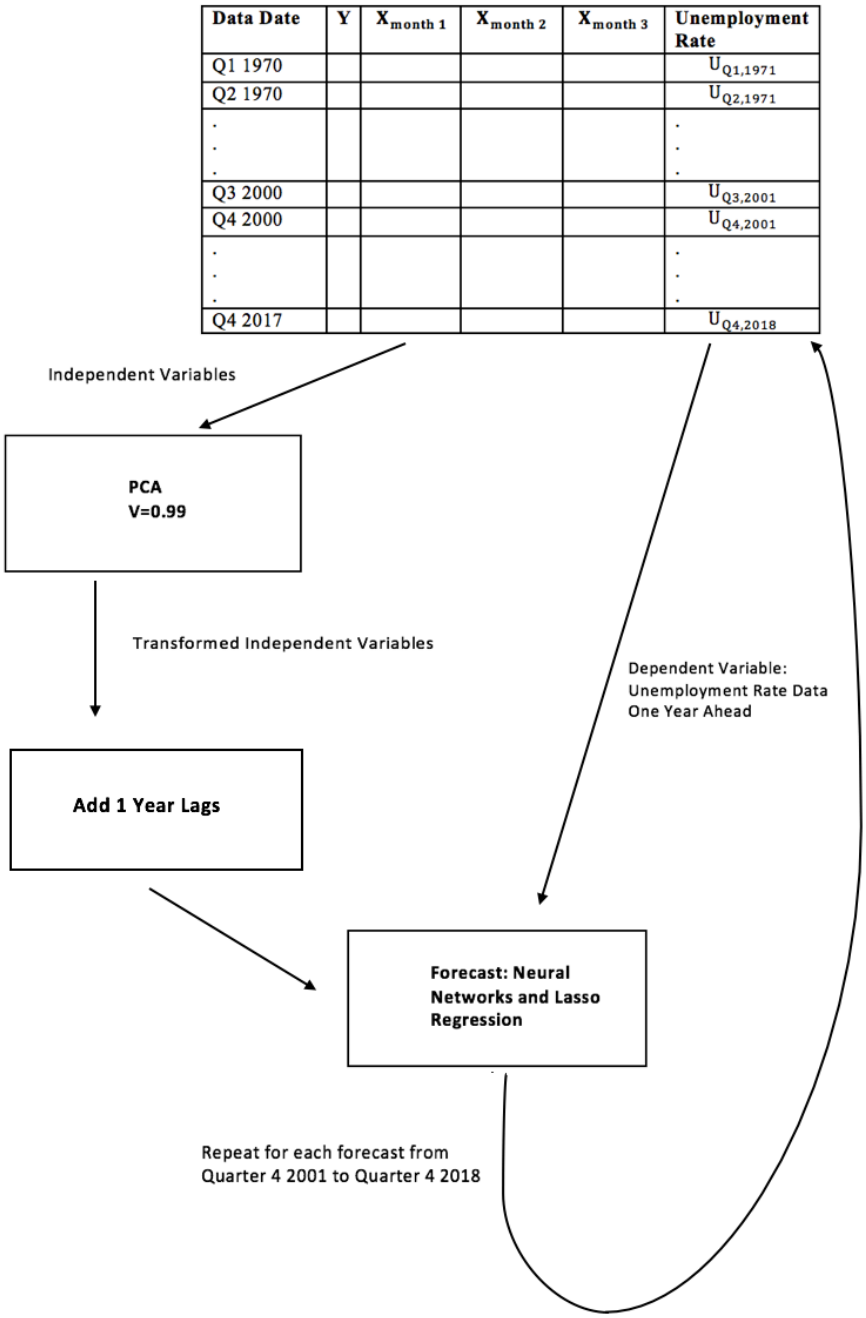

Figure 3: The Forecasting Algorithms for the Lasso and Neural Network Models

Data Date	Y	$X_{month\ 1}$	$X_{month\ 2}$	$X_{month\ 3}$
Q1 1970	1	5	6	7

Table 3: An Example of the Independent Variable Reporting for 1970:q1

For the neural network variant, we note that the weights, biases, thresholds, etc. change from forecast to forecast during the 2001-2018 forecasting period. However, there are parameters that are fixed. The number of hidden layers, number of neurons per hidden layer, activation function, and rolling versus non-rolling window remain constant across the forecasting period. However, each of these fixed hyper-parameters is varied to determine the optimal forecast configuration. In other words, the entire forecast is re-run making small changes in each of these parameters. This paper tests 1-150 neuron hidden layer size; one, two, and three hidden layers; four different activation functions; and rolling and non-rolling windows.

5.3 Lasso Regression

The Lasso Regression uses the same overall structure as the neural network model. However, the only hyper-parameter tested is the choice of the shrinkage parameter: λ. λ is varied to generate a series of forecasts from 2001-2018 to determine the optimal forecast configuration.

6. Results

The root mean squared error is used to assess forecast accuracy relative to the SPF and other models. The error is rounded to the nearest 0.01 as forecasts are often similarly rounded.

6.1 Finding the Best Neural Network Model

This subsection details finding the best model for the entire dataset using neural networks. For the neural network model, the optimal neural network configuration is first found. One, two, and three hidden layers were tested—layers greater than three were not tested because they were too computationally excessive for the computer used. Within each hidden layer, the hidden layer size was varied between 1 and 150. Hidden layer sizes over 150 were not tested because the RMSE was at a minimum at about 120 neurons for each number of these hidden layer sizes. Figure 4 plots the RMSE for each off these network configurations. There are three series, one for each number

of hidden layers. Within each series, the number of neurons per hidden layer is varied. Based on the chart, two and three hidden layers on average outperform one hidden layer. The best configuration, based on RMSE, occurs at a configuration with three hidden layers and 97 neurons per layer. We will call this configuration (97,97,97). The RMSE for this configuration is 0.20.

We next identify the best activation function for the neural network. We use the optimal configuration found above as a given for this test. While in theory it is best to test all neural network configurations and activations, it would be too time consuming to test all possibilities. Instead, the best choice for each test is passed onto the next test. The *activations logistic, identity, logistic, tanh,* and *relu* are tested. The results of this test are shown in Figure 5. Logistic is the best activation function with respect to RMSE by a landslide.

This finding is also consistent with the neural networks papers discussed earlier in the literature review. Given the above two characteristics, the final attribute tested is the rolling versus non-rolling window of the training or fit period. For the latter, the training starts in 1970:q1. In the rolling variant, the beginning of training rolls ahead by one quarter for each projection. The rolling and non-rolling results are reported in Figure 6.

The fixed time window (or non-rolling) greatly outperforms the rolling time window variant. This result has economic significance. When predicting future unemployment, omitting macroeconomic data from relatively old recessions and expansions has a large positive impact on the error. Rolling out the old data increases error because all past cycles have some effect on new cycles. In other words, including old unemployment and other data patterns helps capture the nature of business cycles. The more cycles embedded within the model, the higher the probability the model can predict unemployment rate in new cycles. This finding is also confirmed by Montgomery et. al. (1998) who find that including lags increases their model's forecast accuracy.

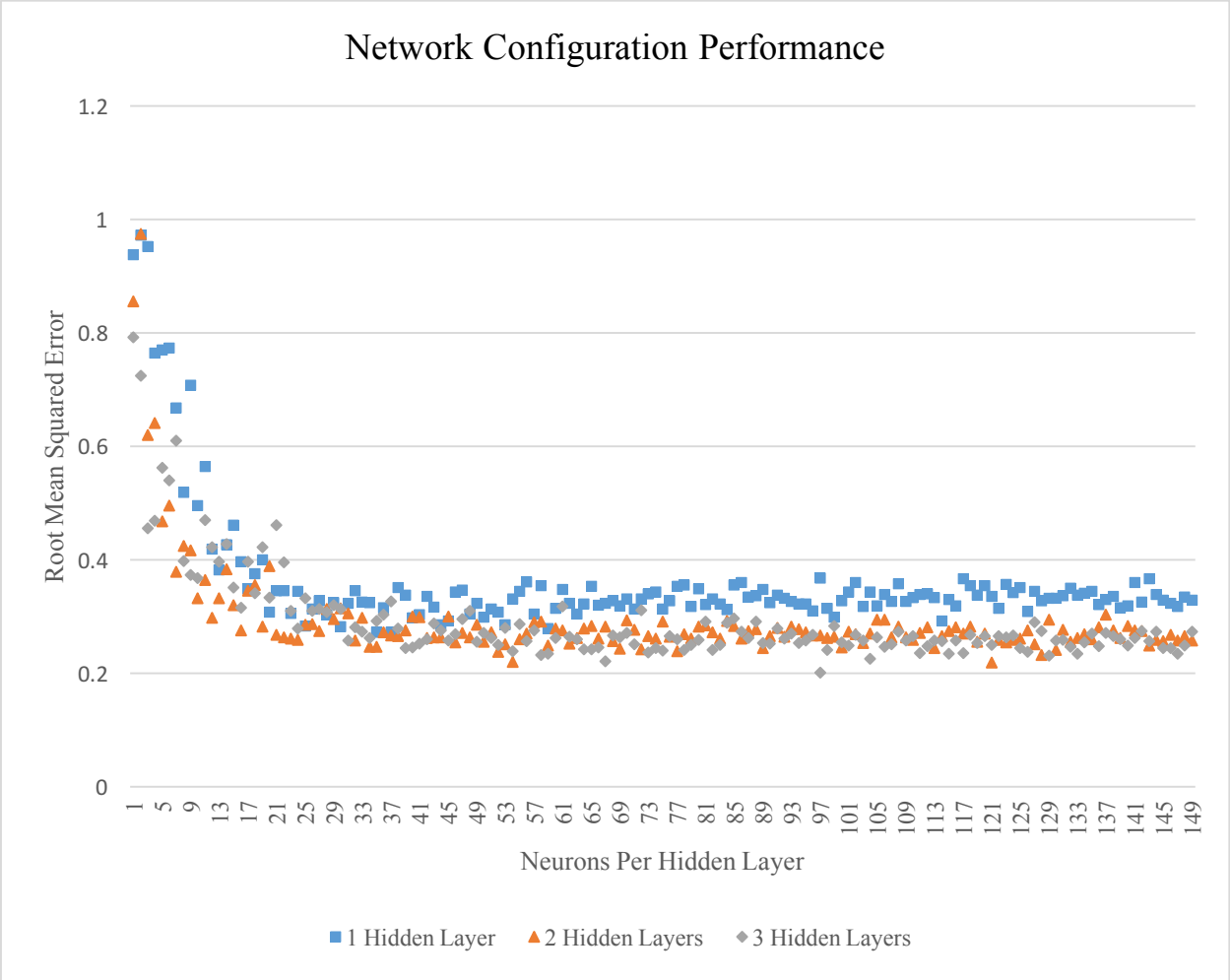

Figure 4: Summary of Hidden Layer Performance.
(Sources: FRED and the authors' calculations.)

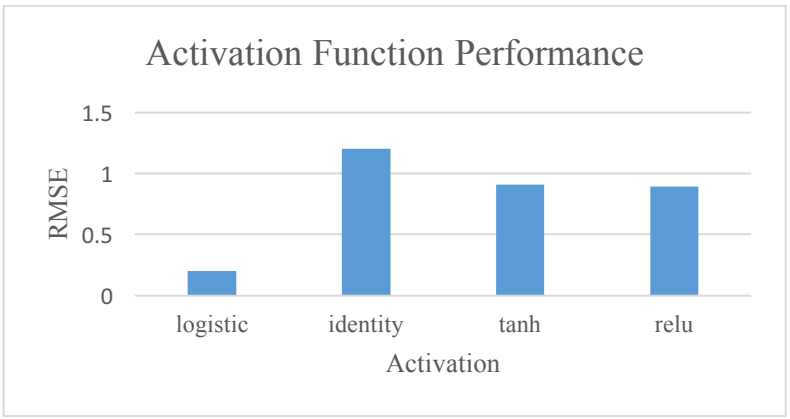

Figure 5: Summary of activation function performance.
(Sources: FRED and authors' calculations.)

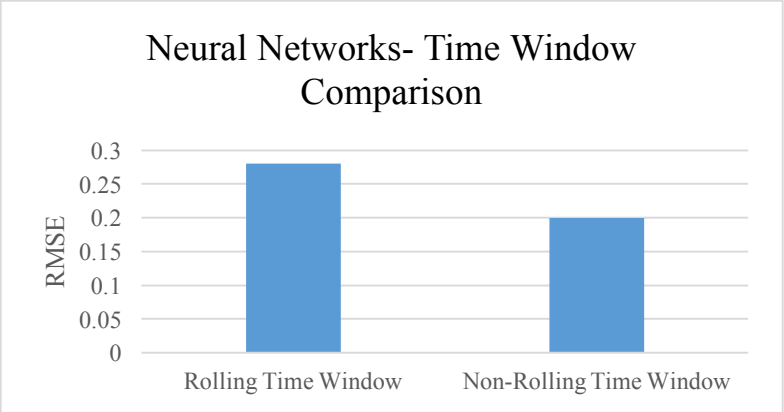

Figure 6: Rolling vs. Non-Rolling model performance
(Sources: FRED and authors' calculations.)

6.2 Finding the Best Lasso Regression Model

Using the findings of Subsection 6.1, we use the non-rolling training window as it had the best RMSE. The only test in this subsection is varying λ to find the appropriate shrinkage parameter that minimizes error. Results are shown in Figure 7. λ is varied from 0 until λ=1. λ is not varied outside this range as λ is shown to monotonically increase after λ=0.15. This λ has the lowest RMSE: 0.31. This will represent the best of the Lasso Regression type models.

6.3 Full (2001-2018) Model Results

This section details the results of using the best neural network configuration, Lasso regression configuration, and the "naive model" to project the unemployment rate. The configurations were found in sections 6.1 and 6.2. For the Lasso and neural network model, this paper considers it a "full" model because all the variables identified from Section 3 are used. Figure 8 plots each of the models' unemployment projections over time against the benchmark. Table 4 provides the RMSE for each model over the whole time period and stratified in three time periods: 2001-2006, 2007-2012, and 2013-2018. These time stratifications were chosen to document model performance over three consecutive 5-year time intervals. We note the data from FRED for the

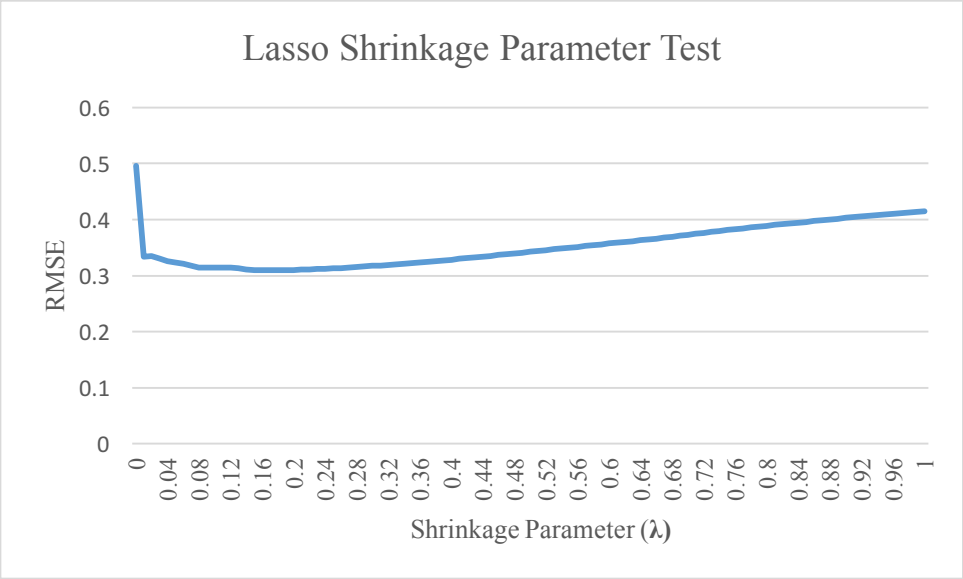

Figure 7: Lasso Shrinkage Parameter Test Results
(Sources: FRED and authors' calculations.)

Lasso and neural network models has 185 principal components, 10,646 variables, 1,905,870 observations. These will come into play in the next section.

As reflected in Figure 8 and Table 4, the naive forecast severely underperforms relative to the other models. The Naive forecast has five times the RMSE of the Lasso and eight times the RMSE in the long sample variant. The 2001-2007 and 2007-2012 show similar results. However, for the 2013-2018 period, the Naive Forecast has the same RMSE as the Lasso. This low error is likely due to the downward negative linear trend of the unemployment rate during this time period. One would not expect the Naive Forecast to perform well given that it only takes into account the current and prior four quarter's data on the unemployment rate. By including other key variables, the SPF and the Lasso, and neural network models can obtain higher precision in projection. The neural network model beats the SPF and other models across all time periods. In particular, 2007-2012 was dominated by the Great Recession and the Global Financial Crisis. For the RMSE over this period, most of the error occurs from late-2008 to early 2009. The error of the neural network

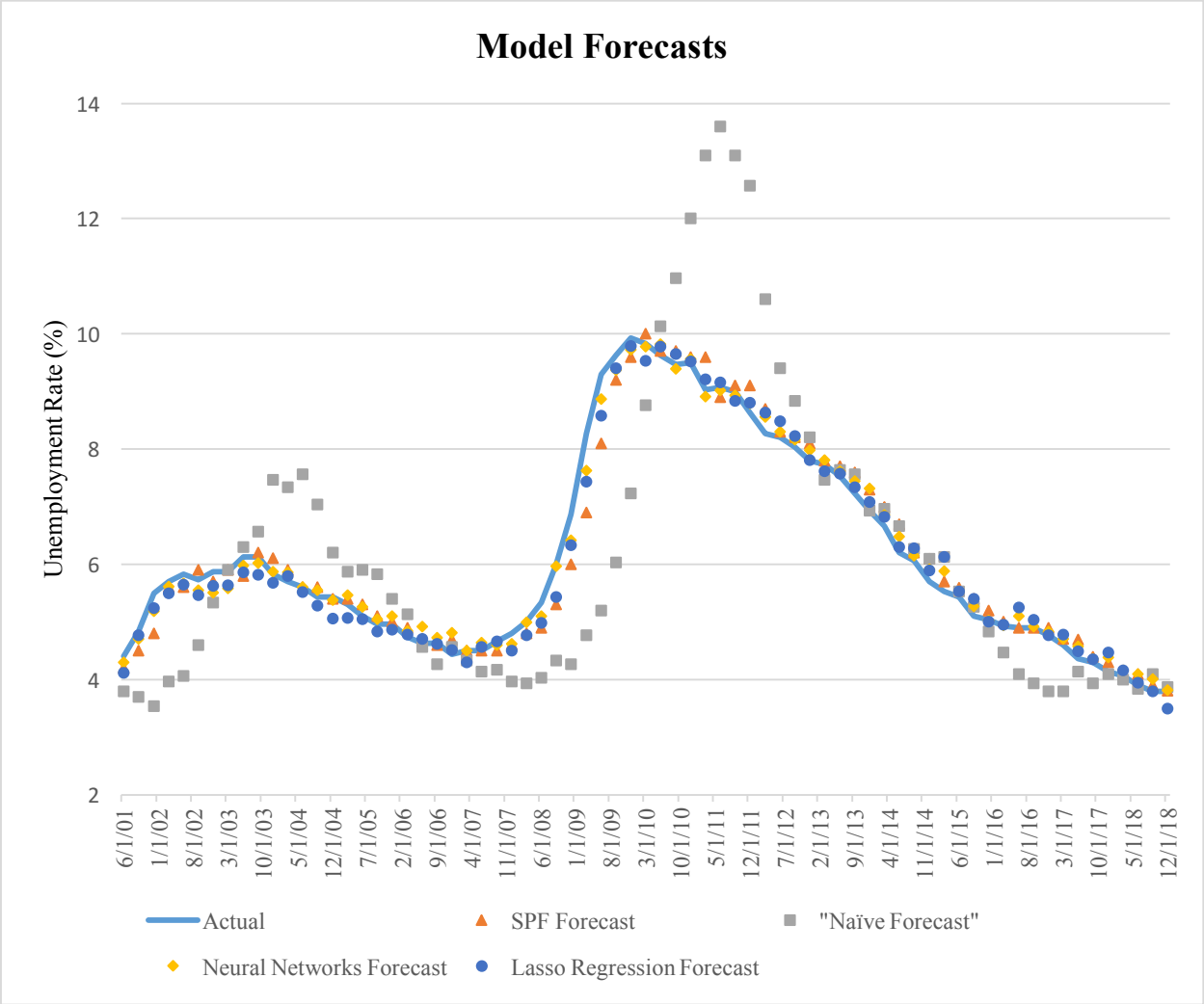

Figure 8: Forecasts of the Unemployment Rate over 2001-2018.
(Sources: FRED and authors' calculations. For example, a forecast made at 6/1/01was projected with the data as late as 2000. See pp. 1-2 for the definition of 4-step ahead forecast.)

Type of Model	RMSE Overall	RMSE 2001-2006	RMSE 2007-2012	RMSE 2013-2018
SPF Forecast	0.35	0.23	0.51	0.23
"Naive Forecast"	1.60	1.10	2.50	0.45
Neural Networks	0.2	0.19	0.23	0.18
Lasso Regression	0.31	0.2	0.34	0.37

Table 4: Forecast Errors By Time Period
(Sources: FRED and authors' calculations.)

model is less than one-half that of the SPF and about two-thirds that of the Lasso regression. This subsample period marks a key pivot point from expansion into a recession. Therefore, the neural net model forecasted unemployment the best among the models considered during the financial crises. The end of 2010 marked another pivot point, just after the end of the recession and entering into expansion. As reflected in Table 4 and Figure 8, the neural network model best predicted the unemployment during the recession and the following recovery.

6.4 Categories of FRED that Most Contribute to Unemployment Rate Forecasts

This subsection takes a deeper look and examines which types of variables have the biggest impact on unemployment rate forecasts. We use the neural network model for this as it has the best performance as shown earlier. First, it is helpful to consider the number of principal components that remain after each category is dropped. This determines the relative amount of variation within the dropped category that leaves the data set. More variation means a higher information content for a category of variables. Relevant statistics are provided in Table 5. We note that there were no variables from the “academic category” in the initial neural net model, perhaps because these variables are transformations of other underlying variables, whose content is embodied in the principal components identified from our PCA pre-processing algorithm. Data from the FRED categories “International” and “National Accounts” have the highest information content. This is consistent with both categories yielding the largest numbers of variables selected for inclusion by the scraping criteria. The "# Principal Components" column represents the number of principle components the model contained for the last forecast. The numbers of principal components from other categories fit this pattern as well: more variables in the dropped category reflects fewer principal components when the category is dropped.

Tables 6 and 7 summarize the RMSE of forecasts when each category is dropped, and are stratified with respect to the same time periods in subsection 6.3. Table 8 then lists the three most influential categories per time period when projecting the unemployment rate. The biggest loss in fit over the full sample occurs when international data are dropped, as indicated in Tables 5-7. This plausibly reflects that international variables provide much marginal information—particularly about shifts in the composition of national expenditure that affected U.S. labor demand. This pattern is consistent with Autor et. al's (2016) view that a China shock hit U.S. labor markets after China entered the WTO in the early 2000s. This timing is consistent with the omission of international data having the largest impact on RMSE for the 2001-06 subsample, when comparing the model to the baseline. From a broader perspective these results reflect that the broader rise of globalization and international integration have impacted U.S. labor markets. Nevertheless, the marginal information content of international data appears to decrease since the early 2000s. This may reflect that the net pace of globalization and integration has ebbed, consistent with evidence that world trade has stagnated, if not receded.

Category Taken Out of Model	**# Vars Taken Out (Monthly)**	**# Vars Taken Out**	**# Total Vars Taken Out**	**# Principal Components**
Academic Data	N/A	N/A	N/A	N/A
International Data	995	3361	4356	182
Money,Banking,Finance	139	96	235	185
National Accounts	84	2835	2919	182
Population, Employment, Labor Markets	741	165	906	185
Prices	765	4	769	185
Production, and Business Activity	606	238	844	184
US Regional Data	601	16	617	184

Table 5: A summary of number of variables taken out per category. This is stratified for quarterly and monthly variables. (Sources: FRED and authors' calculations. Note that the base neural net model (from the previous subsection) had 185 principal components.)

Category Taken Out of Model	RMSE 2001-18	RMSE 2001-06	RMSE 2007-12	RMSE 2013-18
Academic Data	N/A	N/A	N/A	N/A
International Data	0.29	0.31	0.33	0.22
Money,Banking,Finance	0.25	0.24	0.31	0.19
National Accounts	0.24	0.23	0.26	0.23
Population, Employment, Labor Markets	0.26	0.26	0.31	0.20
Prices	0.24	0.18	0.29	0.22
Production and Business Activity	0.23	0.19	0.31	0.16
Regional Data	0.23	0.21	0.27	0.19

Table 6: Dropped Category Results By Sample Period (Sources: FRED and authors' calculations. Once again, we note academic data has no variables in any of the models.)

Category Omitted	RMSE 2001-18 +/- Above Baseline	RMSE 2001-06 +/- Above Baseline	RMSE 2007-12 +/- Above Baseline	RMSE 2013-18 +/- Above Baseline
Academic	N/A	N/A	N/A	N/A
International	0.09	0.12	0.10	0.04
Money, Banking, Finance	0.05	0.05	0.08	0.01
National Accounts	0.04	0.04	0.04	0.05
Population, Employment, Labor Markets	0.06	0.07	0.08	0.02
Prices	0.03	0.00	0.06	0.03
Production & Business Activity	0.03	0.00	0.08	-0.03
Regional Data	0.03	0.03	0.04	0.01

Table 7: Dropped Category Results with Respect to the Baseline SPF Forecast
(Sources: FRED and authors' calculations. "+/- above baseline" column compares the best base neural network RMSE with that of the model with the omitted category. "+/- above baseline"=RMSE omitted category- RMSE base. The RMSE base has a value of 0.20 from before. This measures the increase in error of omitting a category with respect to the benchmark. In general, the higher the error, the larger the impact the omitted variable category has on unemployment projections.)

Time Period	Most Influential Category	2nd Most Influential Category	3rd Most Influential Category
Overall	International	Population, Employment, Labor Markets	Money,Banking,Finance
2001-06	International	Population, Employment, Labor Markets	Money,Banking,Finance
2007-12	International	Production and Business Activity	Money,Banking,Finance
2013-18	National Accounts	International Data	Prices

Table 8: Summary of Categories Having the Largest Impact
(Sources: FRED and authors' calculations. These are with respect to the baseline calculations to ensure the correct neural net forecast is accounted for.)

The second greatest loss in fit is from omitting labor and demographic data. This is with respect to the entire forecast and the 2001-2006 forecasts, and reflects how past labor variables have high predictive power for future unemployment. The effect is highest over 2007-2012 sample and second in the 2001-2006 sample, consistent with Autor et al.'s (2016) China shock hypothesis.

The third largest loss in fit from forecasting the full, 2001-2006, and 2007-2012 samples is from omitting financial data. This likely reflects the outsized effects of the global financial crisis and that the last two recessions have been triggered and amplified by asset price declines. These include the out-sized roles for stock prices and the Internet bust in the 2001 recession and how the housing bust and its correlated spillover effects on other sectors triggered and deepened the Great Recession.

There are other interesting findings. First, "National Accounts" data had the highest information content across major categories for forecasts over 2013-2018. A plausible explanation is that the pace of aggregate domestic demand and of major demand components dominated the period. This may reflect that the pace of decline in the unemployment rate during this time coincided with the domestic impact of variation in fiscal and monetary policy effects, as well as the recovery of domestic consumption and investment demand. Another interesting result is that price data has the third largest effect on forecast fit over 2013-18. Efforts by the Federal Reserve

to return inflation to its 2 percent target and the implied reaction of the stance of monetary policy to inflation data ostensibly affected the degree to which the Federal Reserve's policy stance contributed to economic growth and declines in the unemployment rate.

7. Conclusion

This paper forecasts the unemployment rate four-quarters ahead using machine learning techniques applied to wide-ranging variables available from FRED. Models estimate relationships in-sample over the period 1970-2000 and then forecasts the unemployment rate quarterly since 2001. The training window is updated each quarter to include new data. A rolling and non-rolling period is tested for the training window. We find that a non-rolling neural network model performs best and outperforms the SPF across all time periods, which a Lasso regression approach outperformed to a lesser extent. From experiments dropping broad categories of FRED, international data were the most important in forecasting the unemployment rate, followed in order by data from the FRED categories: Population, Employment, Labor Markets; and Money, Banking, and Finance. This suggests that if one were to create a forecast with significantly fewer variables, retaining variables from these categories would maximize prediction accuracy. Compared to other forecasts, this forecasting method can be efficiently implemented. To generate new forecasts, an SQL database can be used to update variables and produce new forecasts in minutes. Among potential improvements we plan to test additional network configurations and consider additional macroeconomic variables not included in FRED, such as measures of geopolitical risk and consumer confidence.

References

David H. Autor, David Dorn, and Gordon H. Hanson, 2016. "The China Shock: Learning from Labor Market Adjustment to Large Changes in Trade," *Annual Review of Economics* 8(1): 205-40.

Barnichon, Regis, and Christopher Nekarda. 2012. "The Ins and Outs of Forecasting Unemployment: Using Labor Force Flows to Forecast the Labor Market." *Brookings Papers on Economic Activity* 2: 83-117.

Cook, Thomas, and Aaron Hall. 2017. "Macroeconomic Indicator Forecasting with Deep Neural Networks." *Research Working Paper RWP 17-11*, Federal Reserve Bank of Kansas City, www.kansascityfed.org/publications/research/rwp/articles/2017/macroeconomic-indicator-forecasting-deep-neural-networks

Crawford, Jesse. 2009. "Principal Components." *Tarleton State University*, Tarleton State University, 2009, faculty.tarleton.edu/crawford/documents/math505/PrincipalComponents.pdf.

Federal Reserve Economic Data (FRED). 2019. Federal Reserve Bank of St. Louis, www.fred.stlouisfed.org

Fonti, Valeria. 2017. *Feature Selection Using LASSO - VU*. University of Amsterdam, March. beta.vu.nl/nl/Images/werkstuk-fonti_tcm235-836234.pdf.

Kriesel, David. 2009. *A Brief Introduction to Neural Networks*. Self-Published (PhD in Data Science).

Meyer, Brent, and Murat Tasci. 2015. "Lessons for Forecasting Unemployment in the U.S.: Use Flow Rates, Mind the Trend." Working Paper 1502, Federal Reserve Bank of Cleveland. www.clevelandfed.org/newsroom-and-events/publications/working-papers/2015-working-papers/wp-1502-lessons-for-forecasting-unemployment-in-the-us-use-flow-rates-mind-the-trend.aspx

Montgomery, Alan L., Victor Zarnowitz, Ruey S. Tsay, and George C. Tiao. 1998. "Forecasting the U.S. Unemployment Rate." *Journal of the American Statistical Association* 93, 478–493.

Mullainathan, Sendhil, and Jann Spiess. 2017. "Machine Learning: An Applied Econometric Approach." *Journal of Economic Perspectives* 31 (2): 87-106.

Survey of Professional Forecasters. 2019. "Unemployment Rate Forecast." *Civilian Unemployment Rate (UNEMP) - Historical Survey Data - Philadelphia Fed.* www.philadelphiafed.org/research-and-data/real-time-center/survey-of-professional-forecasters/data-files/unemp

Two Sigma. 2016. "Predicting New York City Unemployment Rate Using Taxi Cab Data." 2016.

Xu, Wei, Ziang Li, and Qing Chen. 2013. "Forecasting the Unemployment Rate by Neural Networks Using Search Engine Query Data." *Service Oriented Computing and Applications* 7(1), 33–42.