JOIM
www.joim.com

Journal Of Investment Management, Vol. 23, No. 3, (2025), pp. 4–17

A TRANSPARENT ALTERNATIVE TO NEURAL NETWORKS WITH AN APPLICATION TO PREDICTING VOLATILITY

Megan Czasonis[a], *Mark Kritzman*[b] *and David Turkington*[c]

Many prediction tasks in economics and finance involve complicated relationships that lie beyond the reach of linear regression analysis. Neural networks can capture these complex dynamics, but they are notoriously opaque and difficult to implement. We show that an alternative model-free prediction method, called relevance-based prediction, captures many of the same complex dynamics as neural networks, but with transparency into how each observation contributes to each prediction and how each predictive variable contributes to the reliability of each prediction. We describe both prediction methods and compare their respective efficacy for predicting stock market volatility.

The successful application of policy and strategy by government officials and investment professionals requires the ability to predict outcomes of key variables given observations of the economy and financial markets. The most common tool for forming predictions from data is linear regression analysis which was invented by Carl Friedrich Gauss circa 1795 to predict astronomical motion. However, the forces that govern the motion of planets and stars are much more stable than those that influence human behavior; therefore, many prediction tasks in economics and finance lie beyond the reach of linear regression analysis.

Many researchers have, therefore, turned to neural networks to address the complex dynamics of economic and financial data. However, a neural network is difficult to implement and notoriously opaque. An alternative prediction method called relevance-based prediction, hereafter referred to as RBP, addresses the same complex dynamics as neural networks but in a way that is easier

[a]Managing Director, State Street Associates, Cambridge, MA, USA. E-mail: mczasonis@statestreet.com
[b]Chief Executive Officer, Windham Capital Management, Cambridge, MA, USA; Senior Lecturer, MIT Sloan School of Management, Cambridge, MA, USA.
E-mail: kritzman@mit.edu
[c]Senior Managing Director and Head of State Street Associates, Cambridge, MA, USA.
E-mail: dturkington@statestreet.com

to implement and transparent. To appreciate the fundamental differences between these alternative prediction methods, we highlight some of their key features before we describe each approach in detail.

Features of neural networks

- Neutral networks are models. They discard the data used to train them and only retain the derived parameters which they use for all prediction tasks.
- Neutral network parameters are derived empirically.
- Because neutral networks discard the original data, it is impossible to observe the effect of an observation on a prediction.
- Because a neural network's internal calculation logic relies on large quantities of complex transformations, the interim outputs of each calculation leading to the prediction are difficult if not impossible to interpret.
- Neutral networks reveal nothing about the reliability of a prediction until the outcome is observed.
- Neutral networks estimate as many parameters as necessary to model the complexity.

Features of RBP

- RBP is a model-free prediction routine that retains the original data and adapts the selection of observations and predictive variables to each prediction task.
- RBP selects the observations and predictive variables based on theoretically justified principles.
- RBP explicitly reveals the effect of each observation on the prediction.
- RBP quantifies how each predictive variable contributes to the reliability of a prediction.
- RBP reveals the reliability of a prediction before it is made.
- RBP considers as many combinations of observations and variables as necessary to address complexity.

We next describe neutral networks in detail.

1 Neural Network

A neutral network is a type of prediction model. The general feature of a model-based approach to prediction is that it forms a prediction $\hat{y}_t = f(x_t \mid \Theta)$ for a task x_t by mathematically transforming the values of the input vector according to a prespecified set of rules f that rely on a prespecified vector of parameters Θ. Rather than choose Θ arbitrarily, it is common to use historical observations for the outcomes Y and predictive variables X to choose the parameters $\hat{\Theta} = \mathcal{A}(Y, X, f \mid \Lambda)$.

The function $\mathcal{A}$ is often called a model training algorithm, and it depends on a prespecified vector of hyperparameters Λ which govern the training process. Rather than choose Λ arbitrarily, it is common to use historical observations to choose the hyperparameters $\hat{\Lambda} = \mathcal{H}(Y, X, \mathcal{A} \mid \Xi)$ based on a prespecified set of choices Ξ. The process $\mathcal{H}$ often separates observations of Y and X into subsets of training and validation samples called folds and evaluates the average efficacy of the choices for Λ, which is called cross-validation.[1]

The training process $\mathcal{A}$ identifies one set of parameters to use for all prediction tasks, and the cross-validation process $\mathcal{H}$ identifies one set of hyperparameters based on average efficacy across folds. The training process typically applies gradient descent or another iterative optimization algorithm to change model parameters with the goal of minimizing a chosen function of the model's prediction error, which is called the loss function. One training iteration across all eligible training observations is called an epoch. The training process iterates through multiple epochs

and the cross-validation process iterates through folds. Once the full training and validation process is complete, the model forms predictions based solely on the final parameters as $\hat{y}_t = f(x_t \mid \hat{\Theta})$.

Neural networks refer to a class of models $\hat{y}_t = f_{NN}(x_t \mid \Theta)$ that apply sequences of linear combinations and nonlinear transformations of input variables x_t to make a prediction. The calculation logic can be viewed as a connected network of a chosen number of nodes arranged in a chosen number of layers. Each node corresponds to a set of weights and a constant term, called a bias, that governs how it operates. The total collection of weights and bias terms comprise the parameters of the neural network which are initialized randomly before training and are refined throughout the training process. Given a set of values for the parameters, the neural network uses the corresponding weights and bias terms associated with each node in the first calculation layer to compute a linear combination for that node. Each node then transforms its linear combination into an input for the next layer of the network using a chosen nonlinear function. The network may have any number of subsequent layers that perform the same operations, eventually culminating in a single number as output in the final layer.[2] The model's parameters include the weights and biases associated with every node.

One appealing feature of neutral networks is that they are universal approximators, which means that in a highly general setting they have the capacity to approximate any functional relationship to within any degree of precision if they have enough nodes in the network.[3] Therefore, if a sufficiently large neural network is trained properly for enough epochs, it will achieve zero or near-zero errors for prediction tasks in the training sample. However, for processes that are especially noisy, it may be unwise to fit the training data too closely because extrapolations from noise are harmful for predictions outside the training sample.[4] We can instead stop the training process at an earlier epoch.

Rather than choose the number of epochs arbitrarily, cross-validation guides the decision. For a given validation fold, we apply parameter training via backpropagation to the fold's training subsample for one epoch and then use the resulting model parameters to predict each task in the fold's validation sample. We record the average validation loss of these predictions, typically measured as the mean squared error, and repeat epochs until the validation loss fails to improve the mean squared error for a chosen number of epochs, at which point we select the parameters from the loss-minimizing epoch. This process is called early stopping. The number of further epochs to consider at each juncture is called the patience hyperparameter. We compute the validation loss for each combination of the hyperparameters for learning rate and patience and record the learning rate and number of epochs used in the minimum loss case. We repeat this process for all folds and record the average optimal learning rate and number of epochs. Finally, we use these values of the hyperparameters to train the model on the full sample of available training observations, and we record the optimal model parameters which are used thereafter for prediction.[5]

To promote clarity, it may be instructive to describe the explicit steps we employed to construct the neural network models used for our analysis.

Construction of a neural network

(1) Specify model

   (a) Predictive variables
   (b) Number of layers
   (c) Number of nodes

   (d) Nonlinear transformation (activation) function

(2) Specify training process

   (a) Folds
   (b) Candidate values for hyperparameters

(3) Select initial values of hyperparameters

   (a) Learning rate
   (b) Patience

(4) For the initial prediction of the first observation in the training sample of the first fold, choose a matrix of random weights to apply to the predictive variables; that is, a weight for each variable for the first node in the first hidden layer, a different weight for each variable for the second node in the first hidden layer, and so on to create a weight matrix. Add a bias term for each node.
(5) Multiply the variable input vector by the weight matrix and add the bias terms to get the intermediate values of the nodes in the first hidden layer.
(6) Apply the nonlinear transformation function to the intermediate values of the nodes in the first hidden layer to get the final values of the nodes in the first hidden layer. These final values serve as substitutes for the original variable values in subsequent calculations.
(7) Repeat steps 4 through 6 to implement additional hidden layers.
(8) Calculate the initial prediction as a linear combination of an additional vector of random weights applied to the final values of the nodes in the final hidden layer to get a prediction.
(9) Calculate the prediction error by subtracting the prediction from the observed value of the outcome.
(10) Compute the gradient (vector of derivatives) of the error with respect to the parameters (weights and bias terms) given the nonlinear transformation function and apply it to update the parameter values based on the chosen learning rate.[6] This process is called backpropagation.
(11) For the second observation, repeat steps 4 through 10, but rather than introduce new random parameters, use the final parameters from the prior observation.
(12) For all remaining observations, repeat step 11.
(13) Repeat steps 4 through 12 for each fold.
(14) Based on the validation sample of each fold, calculate the mean squared errors of predictions for all the folds and record their average for the first epoch.
(15) Repeat steps 4 through 14 until the average of the average mean squared error fails to reach a new minimum for a chosen number of consecutive epochs, which is the patience hyperparameter.
(16) Repeat steps 4 through 15 for every combination of hyperparameters.
(17) Find the combination of hyperparameters with the minimum average mean squared error.
(18) Now using the full sample of observations, repeat steps 4 through 15 with the learning rate from the epoch with the lowest average mean squared error, together with the number of epochs with the lowest average mean squared error, to estimate the full-sample parameters.
(19) Form a prediction from the parameters generated from step 18.

This description of the process for constructing a neural network reveals the many nuances and loops that go into a neural network, and it highlights the opacity of the mechanism that translates the original variables and parameters into a prediction.

We now turn to RBP, which addresses the complex dynamics of economic and financial data in a way that is easier to implement and fully transparent.

2 Relevance-Based Prediction

As comprehensively described by Czasonis *et al.* (2024a), RBP is a model-free prediction routine that forms a prediction as a weighted average of prior observations. The weights for a given prediction task are determined by a principled approach that considers the statistical relevance of each observation to the prediction task and relies on different subsamples of observations and variables based on their relative merit for the task. Unlike a model-based approach, such as a neural network, RBP does not have parameters that apply to every prediction task. Instead, it considers the observations for Y and X separately for individual prediction tasks.

RBP addresses complexities in data by selectively censoring the observations and variables that inform a prediction. These choices are codependent, because the choice of observations affects the efficacy of the variables, and the choice of variables affects the efficacy of the observations. Therefore, RBP constructs a grid of combinations of variables (columns) and observations (rows). Each cell in the grid has a set of prediction weights, a prediction, and a measure of reliability. The composite grid prediction is simply the reliability-weighted average of the predictions from all the cells. RBP avoids overfitting because the measure of reliability for each grid cell inherently penalizes small samples of observations and small subsets of variables which are generally more likely to produce spurious relationships than larger samples and subsets.

In addition to the notion of codependence discussed above, the key features of RBP are relevance, which determines the weight of observations for a prediction, and fit, which determines the reliability of a given prediction.

A typical grid specification includes every subset of predictive variables combined with multiple observation censoring thresholds. For a given cell, which corresponds to a specific set of predictive variables X and a censoring threshold r^*, we form prediction weights for each observation x_i based exclusively on X and the prediction task x_t, as follows:

$$w_{it\theta} = \frac{1}{N} + \frac{\lambda^2}{n-1}(\delta(r_{it})r_{it} - \varphi\bar{r}_{sub}) \quad (1)$$

Equation (1) is based on the statistical relevance r_{it} of each observation x_i to the prediction task x_t. The derived quantities involving relevance are defined as follows. We use the Mahalanobis distance to measure the two key components of relevance: similarity and informativeness.

$$r_{it} = sim(x_i, x_t) + \frac{1}{2}(info(x_i, \bar{x}) + info(x_t, \bar{x})) \quad (2)$$

$$sim(x_i, x_t) = -\frac{1}{2}(x_i - x_t)\Omega^{-1}(x_i - x_t)' \quad (3)$$

$$info(x_i, \bar{x}) = (x_i - \bar{x})\Omega^{-1}(x_i - \bar{x})' \quad (4)$$

$$info(x_t, \bar{x}) = (x_t - \bar{x})\Omega^{-1}(x_t - \bar{x})' \quad (5)$$

An observation censoring function $\delta(r_{it})$ determines the censored and retained observations.

$$\delta(r_{it}) = \begin{cases} 1 & \text{if } r_{it} \geq r^* \\ 0 & \text{if } r_{it} < r^* \end{cases} \quad (6)$$

$$n = \sum_{i=1}^{N} \delta(r_{it}) \quad (7)$$

$$\varphi = \frac{n}{N} \quad (8)$$

$$\bar{r}_{sub} = \frac{1}{n}\sum_{i=1}^{N}\delta(r_{it})r_{it} \tag{9}$$

$$\lambda^2 = \frac{\sigma^2_{r,full}}{\sigma^2_{r,partial}} = \frac{\frac{1}{N-1}\sum_{i=1}^{N} r_{it}^2}{\frac{1}{n-1}\sum_{i=1}^{N}\delta(r_{it})r_{it}^2} \tag{10}$$

We express Equation (1) in terms of the quantities defined above because these components provide intuition for the formula. The uninformed baseline for weights is $1/N$, and weights are tilted higher or lower based on the statistical relevance of the retained observations. The relevance-based tilts subtract the average relevance for the retained observations so that they are centered on zero, and they are scaled by λ^2 to properly extrapolate from the subsample of retained observations.

The prediction for grid cell θ is the weighted average of outcomes using Equation (1).

$$\hat{y}_{t\theta} = \sum_{i=1}^{N} w_{it\theta}\, y_i \tag{11}$$

We define adjusted fit as follows, where K is the number of variables in X. Adjusted fit provides a measure of expected reliability for each grid cell's prediction.

$$adjusted\ fit_{t\theta} = K(fit_t + asymmetry_t) \tag{12}$$

$$fit_{t\theta} = \rho(w_{t\theta}, y)^2 \tag{13}$$

$$asymmetry_{t\theta} = \frac{1}{2}(\rho(w_t^{(+)}, y) - \rho(w_t^{(-)}, y))^2 \tag{14}$$

In Equation (14), $w_t^{(+)}$ represents the weights formed from retained observations while $w_t^{(-)}$ represents the weights formed from the complementary set of censored observations. Fit inherently penalizes small samples because weights close to zero dampen the correlation between weights and outcomes. Multiplication by K inherently penalizes small subsets of variables which are more likely to exhibit spurious relationships. Asymmetry is a positive contributor to adjusted fit because in the presence of asymmetry we should trust the retained relevant observations more than the censored less relevant observations on principle.

The total prediction weights for task t are computed as a grid-weighted average of the observation weights from all the cells.

$$\psi_\theta = \frac{adjusted\ fit_{t\theta}}{\sum_{\tilde{\theta}} adjusted\ fit_{t\tilde{\theta}}} \tag{15}$$

$$w_{it,grid} = \sum_{\theta} \psi_\theta w_{it,\theta} \tag{16}$$

We compute the total predictions for task t and its corresponding fit the same way we do for a single grid cell, using Equation (16).

$$\hat{y}_{t,grid} = \sum_{i=1}^{N} w_{it,grid}\, y_i \tag{17}$$

$$fit_{t,grid} = \rho(w_{t,grid}, y)^2 \tag{18}$$

We now summarize the explicit steps for constructing a relevance-based prediction (RBP). .

(1) For a given prediction task, calculate the relevance of each observation to the prediction based on the full sample of observations.
(2) For a given combination of predictive variables and a relevance-filtered subsample of observations, form a prediction as a weighted average of observed outcomes in which the weights are based on relevance.
(3) Calculate the adjusted fit of the prediction.
(4) Repeat steps 2 and 3 for all chosen combinations of predictive variables and chosen subsamples of relevance-filtered observations.
(5) Form a composite prediction as a reliability-weighted average of all the predictions from

the different combinations of predictive variables and subsamples of observations.

2.1 Theoretical foundations of RBP

Unlike neural networks, which rely exclusively on sample-specific empirical validation, RBP is justified by theory and further supported by surprising mathematical equivalences.

- *Information theory and the Mahalanobis distance*. Information theory shows that the information contained in an observation is the negative logarithm of its likelihood. The Central Limit Theorem shows that the probability of an observation from a multivariate normal distribution is proportional to the exponential of a negative Mahalanobis distance. Thus, the information given by an observation from a multivariate normal distribution is proportional to a Mahalanobis distance (see Czasonis *et al.*, 2022b).
- *Convergence of RBP to linear regression analysis.* When RBP uses all the predictive variables and is applied across all observations, it gives the same prediction as linear regression analysis (see Czasonis *et al.*, 2022b).
- *Convergence of fit to R-squared.* For a full-sample linear regression analysis, the informativeness-weighted average fit across all prediction tasks equals R-squared (see Czasonis *et al.*, 2022b).

3 Predicting Volatility

We test both neural networks and RBP for predicting the volatility of the stock market. For context, we also show results for linear regression analysis.

3.1 Data and methodology

The outcome we aim to predict is the subsequent one quarter (63-day) volatility of daily total returns of the S&P 500 index. We use 14 predictive variables as described in Exhibit 1, which are observed at the time of each prediction.[7]

We set up our out-of-sample historical test as follows. Starting on December 31, 1999, we obtain

Exhibit 1 : Predictive variables.

Predictive Variable	Proxy	Source
Market Conditions		
Trailing 1-month volatility	Trailing 21-day volatility of daily S&P 500 returns	Bloomberg
Trailing 3-month volatility	Trailing 63-day volatility of daily S&P 500 returns	Bloomberg
Implied volatility	CBOE VIX (with proxy based on options prices before 1990)	CBOE
Trailing 1-month market return	Trailing 21-day return of the S&P 500	Bloomberg
Trailing 3-month market return	Trailing 63-day return of the S&P 500	Bloomberg
Financial Conditions		
Short-term interest rate (level)	Fed funds rate: 12-m average	FRED
Short-term interest rate (change)	1-year change in short-term interest rate	FRED
Long-term interest rate (change)	1-year change in 10y constant maturity rate	FRED
Credit spread (change)	1-year change in Baa corp. bond yield - 10y const. maturity rate	FRED
Economic Conditions		
Growth	1-year % change in industrial production	FRED
Payrolls	1-year % change in non-farm payrolls	FRED
Inflation	1-year % change in Consumer Price Index (CPI)	FRED
Money supply	3-year % change in M2 money supply	FRED
Debt-to-GDP	3-year change in public debt/GDP	FRED

a sample of monthly observations for outcomes and predictive variables from January 1986 to September 1999. We end in September to account for the subsequent 3-month period corresponding to the final volatility outcome in our input data. We train a neural network for this sample following the process described earlier and store the parameters and prediction function for that model. We then use the neural network to generate a prediction for the daily volatility of the S&P 500 for the first 3-month period of the year 2000. We also use the RBP process described earlier to generate a prediction for the same outcome using the same input data. Next, we move forward 1 month to January 31, 2000 and generate predictions for the 3-month period forward from that date. The prediction task is updated to reflect conditions as of January 31, but the data from which the prediction is formed for both the neural network and RBP remains as before. We proceed in this fashion, forming predictions each month from December 31, 1999 through December 31, 2004. At this point, we augment the available training sample to span January 1986 to September 2004, recalibrate the neural network model on the new data, and allow RBP to use the same augmented data sample. We proceed in this fashion, updating the available data sample for both prediction routines every five years until the final prediction is made on September 30, 2023.

Additional neural network specifications are as follows:

Activation function: Logistic sigmoid, $\phi(a) = (1 + e^{-a})^{-1}$

Structures (all fully connected from one layer to the next):

- 1 hidden layer, 1,000 nodes
- 1 hidden layer, 100 nodes
- 10 hidden layers, 100 nodes per layer
- 10 hidden layers, 10 nodes per layer

Training and cross-validation

- Five folds of equal size sequential time blocks[8]
- Candidate initial learning rates: 0.0005, 0.0075, 0.001, 0.00125, 0.0015[9]
- Candidate patience parameters (for early stopping):
  - For shallow (1 hidden layer) structures: 2 to 20 in increments of 2
  - For deep (10 hidden layer) structures: 50 to 200 in increments of 25
- Minimum epochs (before early stopping is allowed):[10]
  - For shallow structures: 100
  - For deep structures: 1,000

For RBP, we consider a grid consisting of every possible variable combination (grid columns) and observation censoring percentile thresholds of 0, 0.2, 0.5, and 0.8 (grid rows). We consider censoring based on relevance as well as censoring based on similarity, which essentially multiplies the number of grid cells by two. Similarity censoring may be useful if nearby observations alone—and not those observations that are also the most informative—are the best predictors for a given prediction task. We allow RBP to consider both possibilities. Note that the cells that use censoring thresholds of zero are equivalent to linear regression predictions for a given set of predictive variables. There are more than 16,000 grid cells; however, we use a sparse sampling method whereby for each prediction we consider the full sample linear cell, each of the 14 single variable linear cells, and 100 randomly selected cells from the rest of the grid, for a total of 115 cells for each prediction task.

4 Results

Our purpose in carrying out this analysis was not to pass judgment on the best model or routine

for predicting volatility but rather to determine if RBP could serve as a reliable substitute for neural networks when faced with complex data. The following exhibits give a decisive affirmative response to this question.

Exhibit 2 compares linear regression analysis, RBP, and four neural network models that differ only by their number of layers and nodes. It shows the actual results for outcomes that were predicted to be high and for outcomes that were predicted to be low. We define high predictions as those above the 75th percentile and low predictions as those below the 25th percentile.[11] We further partitioned the RBP results into the 50% most reliable predictions and the 50% least reliable predictions as indicated by fit. The results reveal that both RBP and the neural network models distinguished high volatility outcomes from low volatility outcomes more accurately than linear regression analysis, indicating that the relationship between volatility and the predictive variables is nonlinear. Exhibit 2 also shows that the full sample of predictions generated by RBP performed as well as the most successful neural network model, as indicated by the spread between the outcomes of high and low predictions. However, the RBP predictions known in advance to be the 50% most reliable predictions performed considerably better than the most successful neural network model.

Exhibit 3 shows the correlations of the outcomes with the predictions as well as the correlations of the predictions with each other. In this exhibit we considered the full sample of RBP predictions. Exhibit 3 reveals that the best neural network model (10 layers, 100 nodes) was most highly correlated with the actual outcomes (54%), but that RBP was nearly as highly correlated (52%), even though the less reliable predictions were included along with the more reliable predictions to calculate this correlation. It is also interesting to note that the predictions from linear regression

Exhibit 2: Predictions of 1 quarter daily S&P 500 volatility. Average out-of-sample outcomes.

		RBP			Neural Network			
	LR	All	High Fit	Low Fit	1L of 1,000N	1L of 100N	10L of 100N	10L of 10N
Low Predictions	0.87%	0.70%	0.67%	0.73%	0.76%	0.84%	0.71%	0.77%
High Predictions	1.34%	1.45%	1.66%	1.24%	1.32%	1.33%	1.47%	1.43%
High/Low	1.5	2.1	2.5	1.7	1.7	1.6	2.1	1.9

Exhibit 3: Correlations.

				Neural Networks			
	Actual	LR	RBP	1L of 1,000N	1L of 100N	10L of 100N	10L of 10N
Actual	1.00						
Linear	0.34	1.00					
RBP	0.52	0.78	1.00				
NN - 1L of 1,000N	0.33	0.85	0.78	1.00			
NN - 1L of 100N	0.33	0.92	0.66	0.82	1.00		
NN - 10L of 100N	0.54	0.69	0.87	0.73	0.65	1.00	
NN - 10L of 10N	0.45	0.81	0.77	0.78	0.77	0.90	1.00

analysis were highly correlated with the two single-layer neural network models, suggesting that the deeper models are better at capturing nonlinearities in the data. The RBP predictions, by contrast, were 87% correlated with the predictions of the best neural network model, which is one of the deeper models.

Exhibit 4 offers further evidence that RBP has the potential to serve as an effective substitute for neural networks. First, we should note that the average value of the predictions and their standard deviations merely indicate that the predictions are reasonable on average. They do not convey information about the quality of the individual predictions. The root mean squared errors show that the RBP predictions were more reliable than three of the four neural network models and nearly as reliable as the best neural network model, even when the less reliable predictions were included. However, the subset of the 50% most reliable RBP predictions had a lower root mean squared error than even the best neural network model. The final row of Exhibit 4 shows the same correlations of the predictions and outcomes as the first column of Exhibit 3 but with the addition of the correlation of the 50% most reliable RBP predictions and outcomes. This subset of reliable predictions was significantly more highly correlated (0.66) with outcomes than even the best neural network model (0.54).

4.1 The virtue of transparency

The foregoing exhibits offer tantalizing evidence that RBP has the potential to address complex dynamics as effectively as neural networks. Moreover, as we mentioned previously, RBP has two major advantages over neural networks: its ease of implementation and its transparency.

Exhibit 4: Prediction statistics.

			RBP		Neural Networks			
	Actual	LR	All	High Fit	1L of 1,000N	1L of 100N	10L of 100N	10L of 10N
Average	1.1%	1.0%	1.0%	1.0%	1.1%	1.0%	1.0%	1.0%
Standard deviation	0.6%	0.3%	0.4%	0.5%	0.5%	0.4%	0.3%	0.2%
Root mean squared error		0.60%	0.54%	0.50%	0.63%	0.61%	0.53%	0.56%
Correlation to actual		0.34	0.52	0.66	0.33	0.33	0.54	0.45

Exhibit 5: January 2008 and June 2014 predictions. Most and least relevant observations.

January 2008	
Prediction	1.3%
Fit	11.4%
Actual	1.4%

	Most Relevant			Least Relevant		
	Oct-87	Oct-90	Nov-87	Mar-95	Aug-94	Feb-95
Weight	5.0%	3.0%	3.0%	−0.1%	−0.1%	−0.2%
Actual	1.9%	1.0%	1.7%	0.5%	0.7%	0.5%

June 2014	
Prediction	0.6%
Fit	19.9%
Actual	0.6%

	Most Relevant			Least Relevant		
	Sep-04	Jul-04	Jun-04	Nov-08	Oct-87	Oct-08
Weight	2.3%	2.0%	2.0%	−0.5%	−0.6%	−0.7%
Actual	0.7%	0.7%	0.7%	2.6%	1.9%	3.3%

Exhibit 6: Variable importance.

	January 2008	June 2014
Trailing 1-month volatility	0.08	0.20
Trailing 3-month volatility	0.18	0.10
Implied volatility	0.33	0.40
Trailing 1-month market return	0.16	0.36
Trailing 3-month market return	0.16	0.11
Short-term interest rate (level)	0.02	0.14
Short-term interest rate (change)	0.12	0.22
Long-term interest rate (change)	0.12	0.23
Credit spread (change)	0.11	0.30
Growth	0.08	0.04
Payrolls	0.12	0.29
Inflation	-0.12	0.15
Money supply	0.05	-0.54
Debt-to-GDP	0.19	0.18

RBP's relative ease of implementation should be apparent from our earlier descriptions of the processes we employed to construct both prediction methods.

Exhibits 5 and 6 illustrate the transparency of RBP. Exhibit 5 shows the three most relevant and least relevant observations for two predictions of volatility, the quarter following January 31, 2008, during the midst of the Global Financial Crisis, and the quarter following June 30, 2014, when VIX was near an all-time low following a 5-year bull market. The most relevant observations for the January 2008 prediction of volatility included the 1987 portfolio insurance-induced stock market crash and the 1990 stock market crash following Iraq's invasion of Kuwait, while the three least relevant observations, which received negative weights, were extraordinarily quiescent periods. The most relevant observations for the June 2014 volatility prediction were similarly calm periods, while the three least relevant observations were highly turbulent periods. Interestingly, one of the most relevant observations for the January 2008 prediction was one of the least relevant observations for the June 2014 prediction.

Exhibit 6 shows the importance of the predictive variables to the same two predictions highlighted in Exhibit 5.[12] Blue cells convey more importance while red cells convey less importance. Variable importance is calculated as the average adjusted fit of the grid cells that include a given variable minus the average adjusted fit of the grid cells that do not include that variable.[13] It shows that implied volatility was most important to both predictions, as we should expect if we trust the wisdom of crowds, but that the other variables had varying degrees of importance. Trailing one-month market return, which was only minimally important for predicting the January 2008 prediction, was the second most important variable for predicting the June 2014 outcome, which is not surprising given that this outcome followed a 5-year bull market.

RBP's ability to give visibility into how the observations contribute to the formation of a prediction is critical to assessing a prediction's reliability. Even though fit gives a preview of a prediction's reliability, it does so based only on data. It could be the case, for example, that a highly relevant observation is a data error or that it should be discounted for some other reason. An unusual prediction might draw scrutiny and lead us to examine the observations that are most relevant to its formation, which might help us discover the data error or identify the other reason to discount the observation. It is only because RBP is transparent that this scrutiny is possible. Neural networks, by contrast, thoroughly obscure all information about the effect of observations on a prediction.

5 Conclusion

Neural networks form predictions from data with complex dynamics. By performing many sequences of linear combinations and nonlinear transformations of the original inputs and by cross validating the results along the way, a neural network has the potential to extract nearly all the useful information from a dataset to form a prediction. And the prediction is likely to be highly reliable to the extent the training sample closely represents the circumstances of the prediction. However, if the prediction circumstances are not well represented in the training sample, the prediction may not be reliable. Moreover, if the training data is noisy, there is a significant chance of overfitting the neural network, which could compromise the reliability of the predictions. These challenges are exacerbated because a neural network obscures the way the observations and predictive variables influence a prediction. It is, therefore, impossible to assess the quality of a prediction from a neural network until the outcome is known, which may be too late.

RBP also forms predictions from data with complex dynamics, but unlike neural networks, it is model-free and theoretically grounded. RBP forms a prediction as a weighted average of observed outcomes in which the weights are based on a statistical measure called relevance. Relevance is composed of similarity and informativeness, which are both measured as Mahalanobis distances. RBP also depends on fit which measures the alignment of relevance weights and outcomes. Fit determines how to blend observations and predictive variables to give the most reliable prediction, and it gives guidance about the reliability of a prediction before it is made. RBP is remarkably transparent. It reveals explicitly how each observation informs a prediction. It shows how each predictive variable contributes to the reliability of a prediction. And it discloses the reliability of a prediction before it is made.

We applied both prediction methods to predict stock market volatility, along with linear regression analysis to provide context. We considered four specifications of neural network models. Our results showed that the most successful neural network specification and our RBP routine both generated significantly more reliable predictions than linear regression analysis, thus warranting their use. Our results also showed that the most successful neural network specification and RBP produced similar predictions that were similarly effective. It is important to note, though, that we would not have known in advance which neural network specification would work best.

We then illustrated the transparency of RBP by showing the three most and three least relevant observations to a chosen prediction of a high outcome and a chosen prediction of a low outcome. We also showed the relative importance of the predictive variables for these two predictions.

Given the strong theoretical foundation of RBP, along with the results of our empirical analysis, we believe that RBP could serve as a compelling substitute for neural networks, especially considering its ease of implementation and transparency.

Notes

This material is for informational purposes only. The views expressed in this material are the views of the authors, are provided "as-is" at the time of first publication, are not intended for distribution to any person or entity in any jurisdiction where such distribution or use would be contrary to applicable law and are not an offer or solicitation to buy or sell securities or any product. The views expressed do not necessarily represent the views of Windham Capital Management, State Street

Endnotes

1 Choosing one model among many candidate models, or choosing to combine multiple models in an ensemble, can also be considered in this context by allowing some of the Θ or Λ values to govern the degree to which an individual model is used to form predictions within a broader multi-model definition of f.

2 The network architecture we consider in the paper, in which the inputs for each calculation layer derive exclusively from the immediately preceding layer, is called a feed-forward network.

3 For more details, see for example, Hornik (1991).

4 Czasonis *et al.* (2024a) compare RBP to the high-complexity models (HCM) of Kelly *et al.* (2024), which are closely related to neural networks. HCMs generate large numbers of nonlinear variable transformations as in a neural network with one hidden layer, but HCMs use random weights and bias terms rather than apply backpropagation to estimate those parameters. The result is a (regularized) linear regression applied to the transformed variables, which is also equivalent to RBP applied to all variables and all observations. Czasonis *et al.* show that with a large enough number of randomly sampled transformations, HCMs render predictions that place 100% weight on a single observation for each prediction task in the training sample. Thus, while these predictions avoid amplifying noise with erratic extrapolations, they also fail to remove the noise that is already included in each training observation because the prediction is simply the training observation itself. Likewise, neural networks can be viewed as complex nonlinear transformations of prediction inputs that are then used in a linear regression, owing to Jacot *et al.* (2018) who show the convergence of infinitely wide neural networks to kernel prediction using a Neural Tangent Kernel (NTK). The NTK is equal to a transformation function that transforms a vector of prediction inputs into a vector of parameter gradients, capturing the sensitivity of that input's prediction to each of the model's parameters. To the extent this approximation holds for a given (finite) neural network, we may view the model as a nonlinear transformation into a large set of variables that enter a (regularized) linear regression. This relationship, together with the convergence of RBP to linear regression, may offer additional insights into the nature of overfitting for neural networks.

5 Note that the final training of the model on all available training observations does not have a validation sample, therefore we cannot use the early stopping rule based on the patience hyperparameter. Instead, we perform the final training exercise using the number of epochs that were chosen by the optimal patience hyperparameter during validation.

6 We use the currently popular "Adam" method introduced by Kingma and Ba (2014) to compute adaptive learning rates that differ across parameters and across training iterations based on a moving average of the estimated average and standard deviation of the derivative associated with each parameter. The learning rate hyperparameter we evaluate is the initial learning rate for the Adam algorithm.

7 We account for publication lags as necessary for the economic variables. They represent the latest available values for each date.

8 The time blocks include nearly equal numbers of observations, though some may have slightly more or fewer observations to facilitate allocation of the full observation set into five validation subsets with integer numbers of observations. Due to the rolling nature of quarterly observations measured monthly, we exclude from each fold's training sample any observations where Y values would overlap with those of the fold's validation sample. Selecting consecutive time blocks as folds avoids problematic overlap more generally.

9 The conventional default choice for the initial learning rate of the Adam algorithm is 0.001. We allow for the possibility of selecting both lower and higher values.

10 Minimum epochs allows for the possibility that early training can result in more erratic changes in validation loss than later training.

11 We also compared predictions above and below the median, which produced similar spreads.

12 We measure variable importance using information contained in the prediction grid that underlies RBP. It is calculated according to the following equation: $RBI_{tk} = \sum_{\theta} \alpha_{\theta} \frac{\Delta_k(\theta)(adjusted\ fit_{t\theta}) - (1-\Delta_k(\theta))(adjusted\ fit_{t\theta})}{\sum_{\tilde{\theta}} \Delta_k(\tilde{\theta})}$. See Czasonis *et al.* (2024b) for additional detail.

13 It is important to remember that this measure of variable importance pertains to each individual prediction task. A positive value measures the extent to which groups of variables that include a given variable identify more robust patterns than the same groups without that variable. A value of zero indicates that a variable

only contributes noise, but that noise is benign due to a large sample size that diversifies away its effect on the prediction. A negative value indicates that a variable contributes harmful noise that obscures the helpful patterns revealed by the other variables.

References

Czasonis, M., Kritzman, M., and Turkington, D. (2022a). "Relevance," *The Journal of Investment Management* **20**(1), 37–47.

Czasonis, M., Kritzman, M., and Turkington, D. (2022b). *Prediction Revisited: The Importance of Observation*, (Hoboken, NJ: John S. Wiley & Sons).

Czasonis, M., Kritzman, M., and Turkington, D. (2023). "Relevance-Based Prediction: A Transparent and Adaptive Alternative to Machine Learning," *The Journal of Financial Data Science* **5**(1), 27–46.

Czasonis, M., Kritzman, M., and Turkington, D. (2024a). "The Virtue of Transparency: How to Maximize the Utility of Data Without Overfitting," *MIT Sloan Research Paper* (July).

Czasonis, M., Kritzman, M., and Turkington, D. (2024b). "Relevance-Based Importance: A Comprehensive Measure of Variable Importance in Prediction," *MIT Sloan Research Paper* (December).

Hornik, K. (1991). "Approximation Capabilities of Multilayer Feedforward Networks," *Neural Networks* **4**(2), 251–257.

Jacot, A., Gabriel, F., and Hongler, C. (2018). "Neural Tangent Kernel: Convergence and Generalization in Neural Networks," *NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems*.

Kelly, B. T., Malamud, S., and Zhou, K. (2024). "The Virtue of Complexity in Return Prediction," *Journal of Finance* **79**(1), 259–503.

Kingma, D. and Ba, J. (2014). "Adam: A Method for Stochastic Optimization," arXiv: 1412.6980.

Keywords: Cross-validation; fit; grid prediction; hyperparameter; neural network; relevance-based prediction (RBP).