Daniel Wen, Andrew Zabelo, Kyle McCandless
06/07/2024

Bayesian Neural Networks for Predicting Factor Returns

BEM114 Final Project Report

Contents

Overview

In Homework 5, we leveraged machine learning to predict returns in the stock market. However, the model that we implemented chooses the factor that has the greatest predicted return without accounting for the uncertainty of its predictions. This is because the neural network we used can only predict the return, but provides no information about the probability distribution of future returns.

To improve upon our existing model, we switched our neural network architecture to a Bayesian Neural Network. Rather than outputting a single value as a prediction for returns, this model outputs a posterior probability distribution, providing more information about the future. Then, we are able to make a more informed decision when creating our portfolio, **generating positive and significant alpha at the 1% level.** In addition, we achieved a **7% alpha increase** over the HW5 feed forward neural network in-sample, and **a 13% annualized alpha and 1.06 Sharpe Ratio out of sample**.

In this project, we apply the Bayesian Neural Network to predicting factor returns, but this strategy is conceptually sound for applying to any type of returns prediction.

Logic

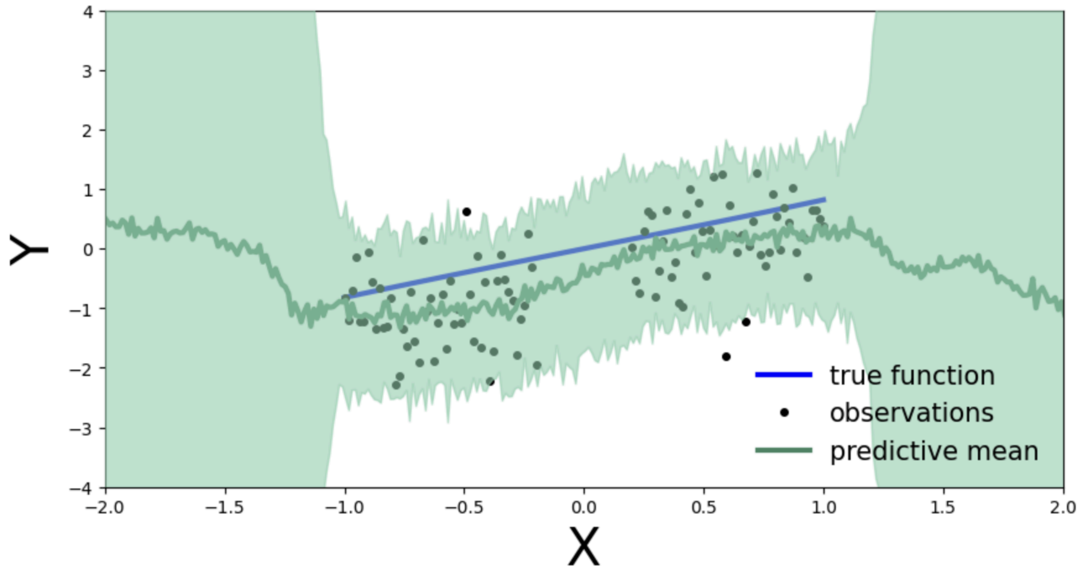
We believe that by predicting a posterior distribution on the underlying stock's price, we will have generated more information than a strategy that just predicts a future price. We can take advantage of this additional information. Rather than predicting just the returns of the market portfolio, we can apply the Bayesian Neural Network to create a more nuanced portfolio. When predicting the returns, we can utilize the standard deviation outputted by the Bayesian neural network to calculate the Sharpe ratio. Then, we can select the asset with the highest Sharpe Ratio in order to choose the best investment choice (adjusted for risk).

Incorporating uncertainty into our strategy can also make the model more robust to market crashes or periods of high volatility. When the market risk is low, then assets will have a higher Sharpe ratio, and our strategy will choose the appropriate asset to invest in. On the other hand, assets with a higher volatility will have a lower Sharpe ratio. So, when the market risk is high, then the Sharpe ratios will be lower in comparison to the Sharpe ratio of the risk-free asset. If the risk-free asset has a higher Sharpe ratio than all other assets, we will shift our allocations towards it. In this sense, by incorporating uncertainty from the Bayesian neural network into our strategy, we are able to implement aspects of the risk parity strategy to build a more robust model that can withstand periods of high volatility and market crashes.

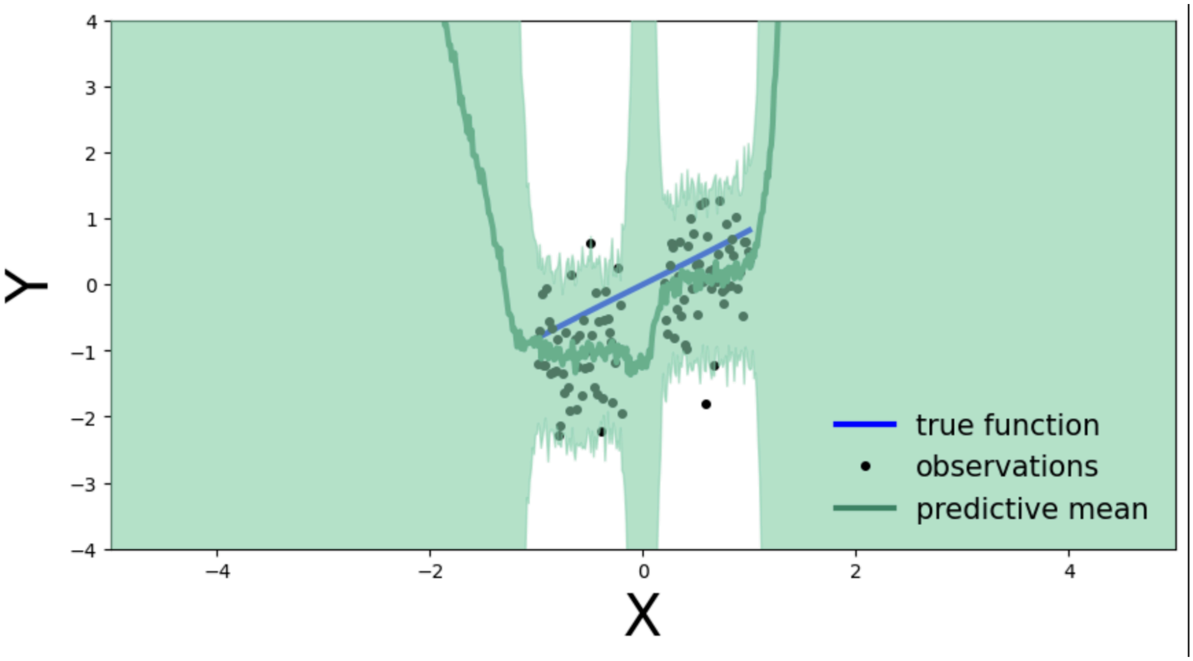
Background: Bayesian Neural Networks

A Bayesian Neural Network (BNN) is an extension of traditional neural networks that incorporates principles from Bayesian inference, allowing for probabilistic interpretation of model parameters. Unlike conventional neural networks that produce point estimates for weights and biases, BNNs treat these parameters as distributions, reflecting uncertainty about their values. The BNNs parameters are typically initialized to gaussian distributions centered at zero, analogous to the random sampling of a zero-mean gaussian to initialize the parameters of a traditional neural network. During training, instead of finding a single optimal set of parameters, BNNs update these distributions to reflect the posterior probability given the data. Analytically computing the posterior is an intractable problem for all but the smallest BNNs, so approximate methods are typically used instead. These include methods like model ensembling, variational inference, or Markov Chain Monte Carlo sampling (Metropolis Hastings Monte Carlo or Hamiltonian Monte Carlo). After training, inference works by repeatedly sampling parameters from the parameter distribution of the BNN to generate an ensemble of traditional neural networks, and performing a forward pass of the input data through each of the generated traditional neural networks. Thus, we obtain a distribution of outputs from a single input, enabling point predictions and uncertainty estimates (by computing the first and second moments) which makes BNNs particularly valuable in finance where understanding the confidence of the model's predictions is crucial. The output distribution tends to have low variance for data points near or similar to the training data. In contrast, it tends to have high variance for data points far from the training data or in regions where the training data is contradictory. This contrasts with traditional neural networks, which might yield overconfident predictions without any indication of uncertainty, potentially leading to riskier decisions when the model is unsure.

Visualizing a Bayesian Neural Network



It is difficult to visualize the inner workings of a neural network based model in the multivariate space of the market data. Thus, above is a simple univariate regression task to better understand how a BNN operates and what type of predictions it makes. 100 [X, Y] training data points were generated with X in the range [-1, -0.2] ∪ [0.2, 1] using a function with added noise. A small, shallow BNN was then trained on this data, and evaluated at 1000 points X in the range [-2, 2]. The dark green line is the expectation of the BNNs predicted distribution and the light green shaded region is the +/- 2 standard deviation bounds at each of the 1000 evaluation points. As desired, the mean line fits the data closely with reasonable bounds in sample and when interpolating between in sample regions (the middle region [-0.2, 0.2] without training data points). Additionally, whereas a traditional neural network would provide baseless predictions outside the range [-1, 1] that often diverge, the BNNs predictions are correctly accompanied with large uncertainty.



If we do not tolerate the low uncertainty for the interpolated region [-0.2, 0.2], we may adjust the BNN hyperparameters to require greater evidence to update the priors of the BNN, resulting in the high uncertainty of the interpolated region seen in the figure above. This demonstrates that the degree to which we induce this high uncertainty in interpolated regions is tunable.

Highest-performing model

Several important design choices went into our best BNN model:

- With the added information of uncertainty predictions, **we enabled the model to choose the factor with the highest expected Sharpe Ratio**, instead of acting on pure returns.
- We also enabled the model to **short factors**, changing the strategy to long or short the highest magnitude return predicted by the BNN. We later upgraded this to running a **long-short Sharpe Ratio strategy** on the factors.
- We increased the width and depth of the fully connected hidden layers of our BNN.
- We found that Hamiltonian Monte Carlo outperformed simpler methods such as ensembling
- We changed from a default tanh to a ReLU activation function because of faster convergence
- We performed a hyperparameter sweep over different prior distributions with which to initialize the BNN weights
- We increased the number of Markov Chain Monte Carlo samples that the BNN runs to generate its predictions, giving a more stable estimate

Backtesting

To backtest this strategy, we started with the Homework 5 file, and switched the feed-forward neural network to a Bayesian neural network. Instead of picking the factor with the highest returns, we looked at the distribution of prices for each factor.

To keep this analysis simple and comparable to Homework 5, we restricted our portfolio to only the factors that we examined previously: the Fama-French 5 factors (Mkt-RF, SMB, HML, RMW, and CMA), momentum (MOM), and the risk-free asset (RF).

Transition from predicting volatility for options trading

Initially, our proposed strategy was to use the predicted distribution of the BNN on SPY returns to better evaluate the fair value of SPY options, and to arbitrage mispricings. Rather than using the Black-Merton-Scholes model, our model would provide us with the information needed to directly calculate the expected value of options by integrating over the payout structure of the option multiplied by the probability distribution of the underlying. Theoretically, this would be more powerful than just using volatility to define a gaussian distribution of the underlying because the BNN could ingest arbitrary data to inform its predictions and the BNN's prediction can have nonzero higher moment unlike a gaussian. However, preliminary results demonstrated that the BNN predictions were not accurate enough to extract meaningful signals to safely trade high market exposure assets such as options. Indeed, fine grained uncertainty prediction in deep learning models is a notoriously difficult task, and is currently an active field of machine learning research.

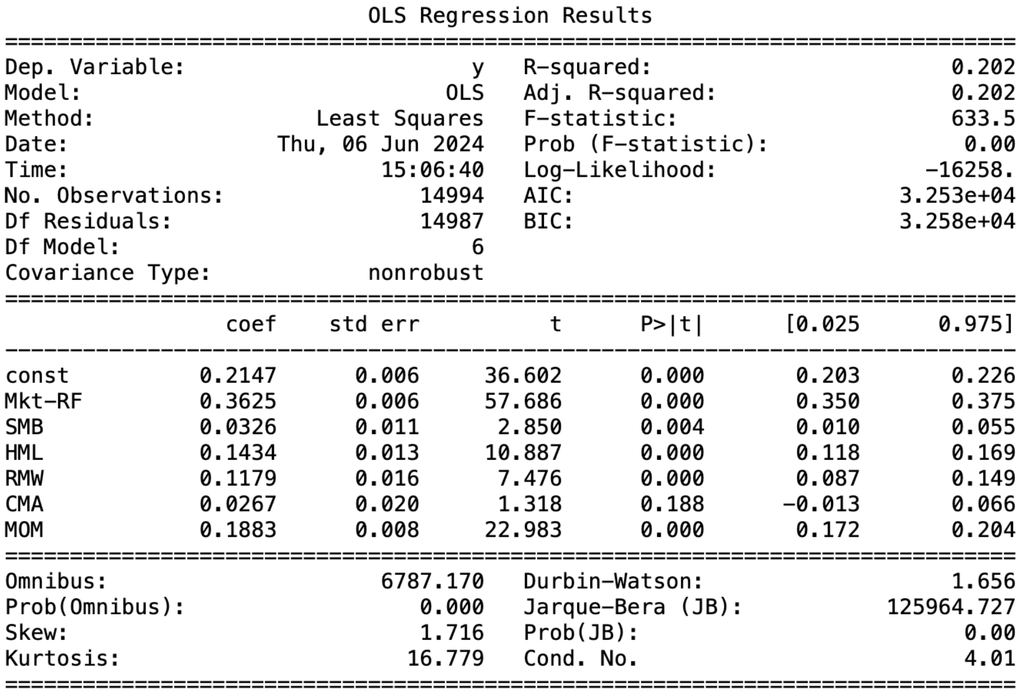
Results

HW5 Benchmark

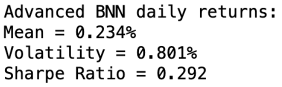
Our submission notebook has visualizations of the best feed forward neural network from Homework 5, which are not included here for brevity. We considered its **14% alpha** and **0.19 Sharpe Ratio** to be the benchmark for the BNN strategies.

Sharpe Ratio BNN

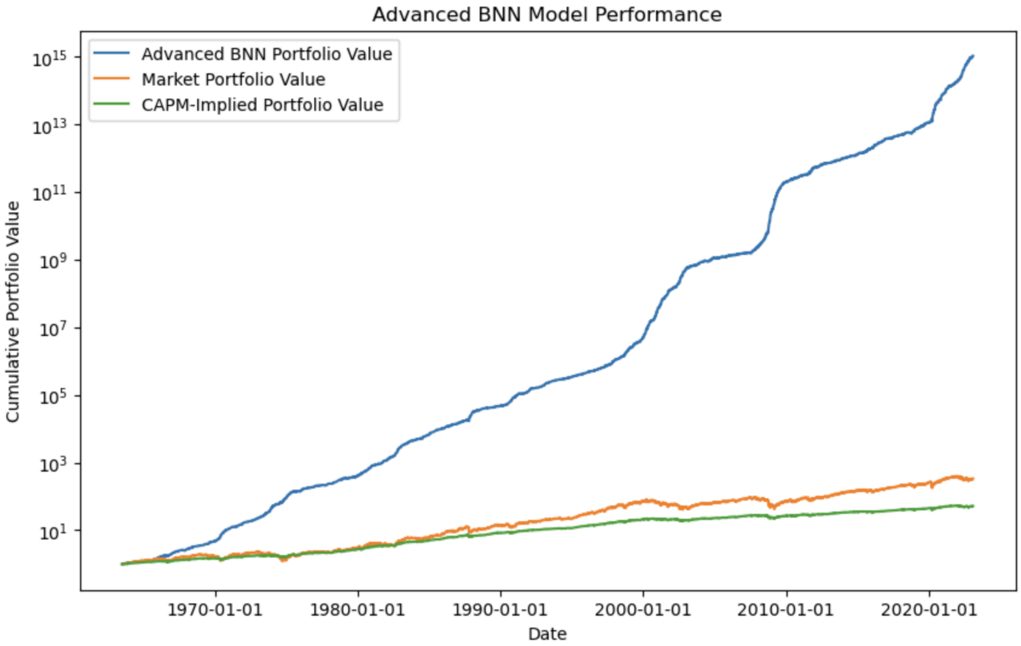
After switching to predicting factor returns using BNNs and taking uncertainty into account, we were able to generate **21% monthly alpha** according to the backtesting framework from HW5, which exceeds the 14% benchmark of our best feed forward neural network model from HW5. Additionally, the strategy has a **high Sharpe Ratio of 0.292.** The best in-sample results are shown below.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.202
Model:                            OLS   Adj. R-squared:                  0.202
Method:                 Least Squares   F-statistic:                     633.5
Date:                Thu, 06 Jun 2024   Prob (F-statistic):               0.00
Time:                        15:06:40   Log-Likelihood:                -16258.
No. Observations:               14994   AIC:                         3.253e+04
Df Residuals:                   14987   BIC:                         3.258e+04
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.2147      0.006     36.602      0.000       0.203       0.226
Mkt-RF         0.3625      0.006     57.686      0.000       0.350       0.375
SMB            0.0326      0.011      2.850      0.004       0.010       0.055
HML            0.1434      0.013     10.887      0.000       0.118       0.169
RMW            0.1179      0.016      7.476      0.000       0.087       0.149
CMA            0.0267      0.020      1.318      0.188      -0.013       0.066
MOM            0.1883      0.008     22.983      0.000       0.172       0.204
==============================================================================
Omnibus:                     6787.170   Durbin-Watson:                   1.656
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           125964.727
Skew:                           1.716   Prob(JB):                         0.00
Kurtosis:                      16.779   Cond. No.                         4.01
==============================================================================
```



```
Advanced BNN daily returns:
Mean = 0.234%
Volatility = 0.801%
Sharpe Ratio = 0.292
```

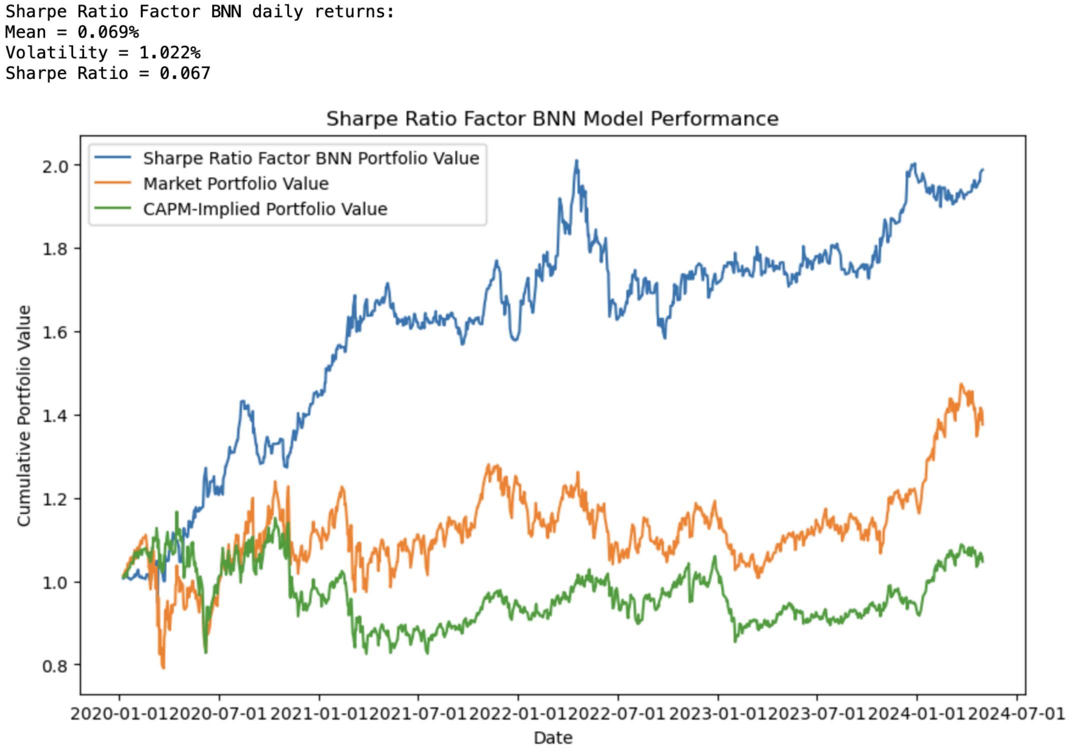


Testing out of sample

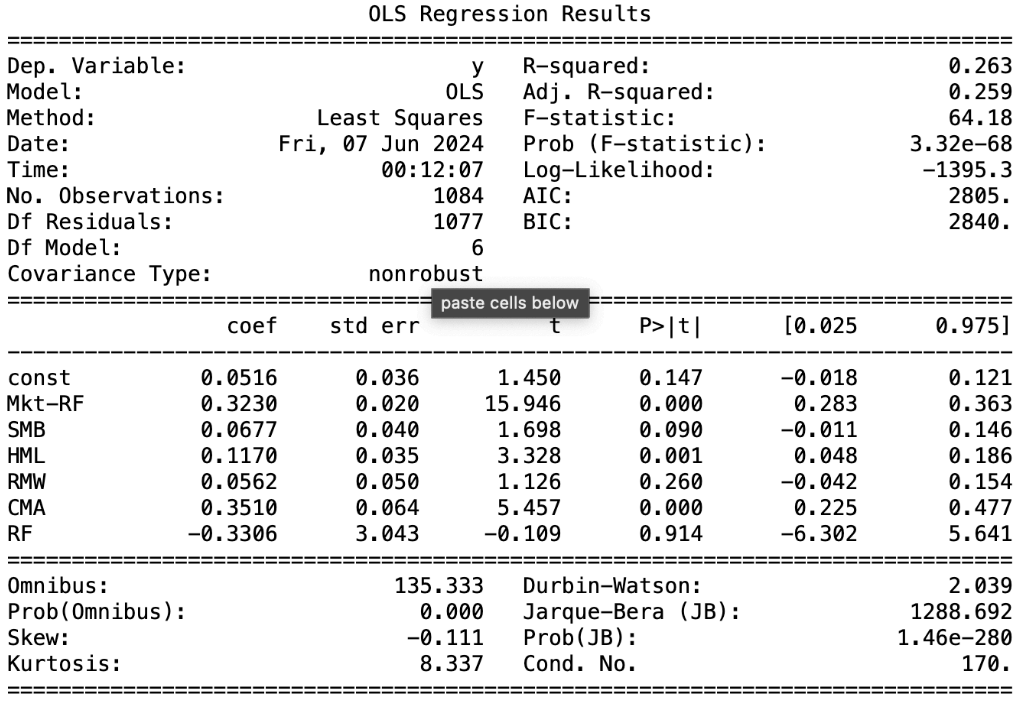
In analyzing these results however, we realized that the backtesting framework from Homework 5 only considers the in-sample performance of the model, training it on the entire dataset from 1960 onwards and then predicting the and evaluating the returns on that exact same dataset. This leads to the unrealistically good performance, both of the HW5 feed forward neural network which matches the sanity check notebook that Professor Sinclair put on Canvas, and the BNN strategies.

To adjust for this, we saved the portion of the dataset that is after March 1st 2020 for evaluating our models. We chose this cutoff intentionally to evaluate the performance of our models during the COVID crash, and found that the strategy continues to generate positive returns during the crash.

As shown on the next page, we achieve **13% annual alpha**, with **18% annualized returns** and a **1.06 annualized Sharpe Ratio**, all **completely out of sample.** The long or short Sharpe Ratio strategy beats the market and the CAPM implied returns, doubling the value of the fund over approximately four years. Most importantly, the exact same BNN model that disregards uncertainty and uses maximum returns instead of Sharpe Ratio fails to beat the market out of sample. **Therefore, we conclude that generated alpha is from the uncertainty component of the BNN as expected, and verify that the large in-sample alphas are robust even in market crashes.**



```
Sharpe Ratio Factor BNN daily returns:
Mean = 0.069%
Volatility = 1.022%
Sharpe Ratio = 0.067
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.263
Model:                            OLS   Adj. R-squared:                  0.259
Method:                 Least Squares   F-statistic:                     64.18
Date:                Fri, 07 Jun 2024   Prob (F-statistic):           3.32e-68
Time:                        00:12:07   Log-Likelihood:                -1395.3
No. Observations:                1084   AIC:                             2805.
Df Residuals:                    1077   BIC:                             2840.
Df Model:                           6
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.0516      0.036      1.450      0.147      -0.018       0.121
Mkt-RF         0.3230      0.020     15.946      0.000       0.283       0.363
SMB            0.0677      0.040      1.698      0.090      -0.011       0.146
HML            0.1170      0.035      3.328      0.001       0.048       0.186
RMW            0.0562      0.050      1.126      0.260      -0.042       0.154
CMA            0.3510      0.064      5.457      0.000       0.225       0.477
RF            -0.3306      3.043     -0.109      0.914      -6.302       5.641
==============================================================================
Omnibus:                      135.333   Durbin-Watson:                   2.039
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1288.692
Skew:                          -0.111   Prob(JB):                    1.46e-280
Kurtosis:                       8.337   Cond. No.                         170.
==============================================================================
```

Risks

While BNNs are an exciting concept to introduce uncertainty measures into machine learning hedge fund strategies, there are still some potential risks. In the logic of our strategy, when the market risk is high, the BNN outputs a high uncertainty for an asset, and the associated Sharpe ratio is low. This makes the choice to invest in the risk-free asset relatively more attractive. So, our strategy will choose to invest in the risk-free asset. The BNN will output a high uncertainty for parameters that it is unfamiliar with (not in the training dataset) or there is minimal data. So, if the market conditions shift to a parameter space that was not included in the training data fed into the neural network, our strategy will continue to stay out of the market and only invest in the risk-free asset. While we believe this risk parity aspect of our strategy to be a strength, it can also hold our strategy back from gaining potential profits. To solve this problem, we can continue to feed in data to update our model and allow it to reduce uncertainty with these unfamiliar parameters in the future.

Another potential risk associated with BNNs, like traditional neural networks, is overfitting to previous market data, especially as we increase the width and depth of the BNN. When a model overfits its training data, it loses the ability to generalize well into the future. So, it is possible that the BNN overfits to past market data, inhibiting its ability to predict returns well in the future.

Fortunately, BNNs are known to be more robust to overfitting than traditional neural networks. This is because BNNs perform marginalization over the weight distributions instead of point estimates (averaging predictions over many possible models) which tends to smooth out extreme predictions that could be a result of overfitting. To further combat the overfitting problem, we altered the variance of the gaussian priors of the BNN parameters, which acts as a form of regularization; a small prior variance would encourage the BNN parameter distributions to be close to zero, analogous to L2 regularization in a traditional neural network. Also, altering the alpha and beta values of the Gamma Distribution provides control over how much our model's posterior is influenced by the evidence (the dataset).

Finally, it became a challenge to choose a suitable time period for backtesting which reflects "typical" market conditions for the future. To protect against the downside that could come with market crashes, we intentionally included the COVID crash in the out-of-sample evaluation set.

Future Work

This project has demonstrated that we can take advantage of the extra uncertainty information that BNNs provide to produce alpha, but there are many exciting next steps.

- To get a more accurate estimate of uncertainty, we can refine the features that are fed into the network, only preserving those that have a high predictive power. For the same reason, we can add additional features to the input space of the neural network. Some potential additional features include idiosyncratic volatility, trading volume, or sentiment.
- Another potential area where Bayesian neural networks may be relevant is trading options. Using the Bayesian neural network's mechanism to produce a probability distribution, we can integrate over the distribution of possible future underlying prices to more accurately determine exactly what an option's price should be today. Then, we will buy the options whose trading prices are lower than our expected price, and sell the options whose trading prices are higher.
- Another area for improvement is to optimize the BNN training process. Because the weights of a BNN are distributions rather than single-point weights, the training process takes much more time and resources to train than a traditional neural network. We can look into methods of optimizing the training time for BNNs.

- BNNs are not the only model capable of estimating the uncertainty of its predictions. The Linearized Laplace Approximation (LLA) is one such method that enables the estimation of the uncertainty of a traditional neural network after it has been trained by using the Laplace kernel to approximate the curvature of the loss landscape, providing a tractable way to quantify uncertainty and make probabilistic predictions. Alternatively, Gaussian Process (GP) models inherently provide uncertainty estimates by defining a prior distribution in function space, and updating the prior given the observed data to produce a posterior distribution in function space. This method is more general than BNNs or LLA, but consequently requires stronger assumptions on the expected structure of the data to produce meaningful predictions.

Resources

- Original project proposal
- Presentation slides
- Git repository
- Demo notebook: Also submitted on Canvas as **BEM114_final_project_working.ipynb** and **BEM114_final_project_working.pdf**
- [Archived] Naive BNN demo notebook