Predicting Russell 3000 Market States with Neural Networks

ABSTRACT

In this study, we implement a neural network approach to predicting market states and a trading strategy to demonstrate our results. We approach the prediction problem in two parts: in the first part, we classify the training data into bear, bull, and static states. In the second, we train a neural network and generate predictions. We used widely accepted market definitions and a non-parametric, rule-based classification algorithm by Lunde and Timmerman to classify bear and bull markets. In order to fit the description of the problem, in addition to the bear and bull regimes, we include a static regime by loosening up the bear market definitions.

Keywords: Neural Network, Market States Prediction

INTRODUCTION

Market State Identification

The problem of classifying market states with rule based methods has been around for some time. The earliest approach is the Dow Theory (circa 1930s), which was essentially the grandfather of all market regime identification approaches. Dow Theory defined the concept of a market trend as a lasting period where the market exhibited consistent qualities that can be defined by some rule. But more recently (in the last handful of decades) algorithmic and more detailed and statistically complex approaches have emerged such as Bry and Boschan (1971), and more notably, Lunde and Timmermann (2004), and Hanna, A. (2018), which we heavily drew insight from.

Principles for bear/bull classification

As mentioned in [5], which examined an assortment of market state identification methodologies, there are a few principles that seem to be shared:

1. Bull and bear phases alternate.
2. Bull (bear) markets exhibit a significant rise (fall) in prices between the start and end.
3. The prices over each phase should be bounded by the values achieved at the phase endpoints.
4. A small change in parameterization should not fundamentally alter the phase dates.
5. Extending the time series should not fundamentally alter the phase dates

For our purposes, since we are adding a 'static' state; our definition of which will be revealed soon; these principles will be less strictly followed, but they are noteworthy nonetheless for insight on the problem of classifying states into bear/bull.

Lastly, to provide the reader with more insight, here are some broad definitions of bear and bull markets as motivation defined by Spearandeo (1990).

Bull market: A long-term upward price movement characterized by a series of higher intermediate highs interrupted by a series of higher intermediate lows.
Bear market: A long-term downtrend characterized by lower intermediate lows interrupted by lower intermediate highs.

The Lunde and Timmerman Algorithm

For our project, we used the Lunde and Timmerman (LT) algorithm from [2] to identify bull/bear periods (see Figure 1 for a sample classification.) The algorithm works around two parameters, λ_{bull} and λ_{bear}, which determine the cutoffs that will trigger a change in market state. Essentially, bear and bull states are defined when the cutoffs are met, and are allowed to continue until the next cutoff is met. For example, in figure 1, there is a long bull period from 2003 until roughly 2007, which only ended when the financial

crisis hit and the market became bear by dropping under the threshold λ_{bear} = 20%. The bull period return is allowed to go much higher than 20%, instead of being cutoff exactly at 20%.

Figure 1: the grey stripes indicate bull market periods of R 3000 according to the LT algorithm.

Neural Networks

Neural networks are a deep learning model, connected by artificial neurons, most of which are estimates or approximations of nonlinear functions. Neural networks are an adaptive system with feedforward network and backward propagation, that is, they can continuously update the internal structure according to external information. As such, they are good at prediction by 'learning' the data efficiently. This approach is still relatively recent (last 10 years,) and straightforward, so we though it would be interesting to use in our study. We did not use recurrent neural networks due to the fact that our features were not sequential. Specifics will be elaborated later.

METHODOLOGY

Data Sourcing

For our data, we referenced [1] and [4] for some insight on what features to include in our project. For example, as mentioned in [1], two of the strongest predictors of bear and bull markets include the 10-year treasury yield minus the 3-month treasury bill and the short-term returns. In total, we considered 13 features, with daily data from 2001/01/04 to 2021/12/30 (See Data Cleaning below for details.) For our data, we obtained them from either FRED or the Bloomberg Terminal.

Below is a summary of the features we considered:

1. Crude Oil Prices from West Texas Intermediate (DCOILWTICO)
2. Goldman Sachs Commodity Index (S&P_GSCI_PX_LAST)
3. The Economic Policy Uncertainty Index (USEPUINDXD)
4. Difference between 3-month Treasury bill and 3 month LIBOR (TEDRATE)
5. Federal Funds Effective Rate (DFF)
6. 10 year Treasury Yield Minus 3 month Treasury Bill (T10Y3M)
7. Bloomberg High Yield Bond Index (BBG_US_HIGH_YIELD_BOND_INDEX_PX_LAST)
8. Bloomberg Corporate Bond Index (BBG_US_CORP_BOND_INDEX_PX_LAST)
9. 3 Month Treasury Bill Secondary Market Rate (3M_TREASURY_DTB3)
10. Russell 3000 (R_3000_Close)
11. VIX Closing Prices (VIXCLS)
12. J.P. Morgan Emerging Markets Bond Index (JPM_EMBI_GLOBAL_PX_LAST)
13. Morgan Stanley Capital International World Market Index (MSCI_WORLD_US_NET_RETURN_PX_LAST)

Data Cleaning

Below are the steps for our data cleaning:

1. Got rid of monthly data, and made sure all data were daily.
2. Forward filled all erroneous data (N/As and '.'s).
3. Used an inner merge.
4. Took the daily returns of features that did not show stationarity.
5. Standardized all the features.

Approach

Our approach can be split into two components. The first is to classify the data, and the second is to use the data to train a neural network and predict market states.

For classifying the market states, we took an approach similar to the Top Down (TD) method mentioned in [5]. This is because like the TD method, we believe that we could perform additional, deeper examination of the classified bear/bull states to look for a 'static' market state. This approach is also the reason we chose daily data as opposed to monthly data like many other studies. Additionally, having a third market state will naturally result in shorter lengths for the bull and bear periods. Consequently, we believe that in this case, the case where bear and bull markets are 'refined,' daily data would serve better for prediction than monthly data.

We chose to use the LT algorithmic approach because the LT algorithm is non parametric: it only requires a very simple classification rule. On the other hand the PS algorithm requires a lot of parametrization, such as lengths of windows, cycles, and periods to define.

We decided to loosely define the 'static' period as a period where the market exhibits qualities that are neither bear nor bull. The decision to filter out 'static' periods from both bear and bull were dependent on the idea that bear states are high mean low variance, while bull states are low mean high variance. We could not find a widely accepted definition of a 'static' market, thus our interpretation is that states with bullish returns but low variance would be considered 'static'.

Instead of first coming up with our own definitions with unsupervised learning like [4] which used gaussian mixture models, we sought more solid, grounded approaches in hopes of obtaining better results with our neural network. After all, a wise man once said that with trash data comes trash results.

Implementation

First, we broadly outlined the bear and bull trends with the Lunde and Timmerman (LT) bear/bull market classification algorithm from [2], which was conveniently included in the R package bbdetection. For our approach, we set $\lambda_{bear} = \lambda_{bull} = 20$ or 20%.

Then, in each of the bear/bull periods we have identified, 'static' states were filtered out from the bull classifications by removing the low volatility periods of the LT algorithm classifications, and thus creating points for the static states. The 63 day or 3-month volatility returns were used based on the general notion that bull markets were, in general, much longer than bear markets, and bear markets are at least 3 months long. We believed that if the data labels were too sparse, (i.e. like a barcode) the neural would have a hard time detecting patterns. See Figure 2a, Figure 2b, and Figure 2c below.

We then fit a neural network with 13 inputs, 200 nodes for the first hidden layer, 50 nodes for the second layer, and 3 output layers. For predicting market states, we used a fully connected neural network, with stochastic gradient descent as the optimization method. We also used 10% of the training data up to 2018 as our validation set.

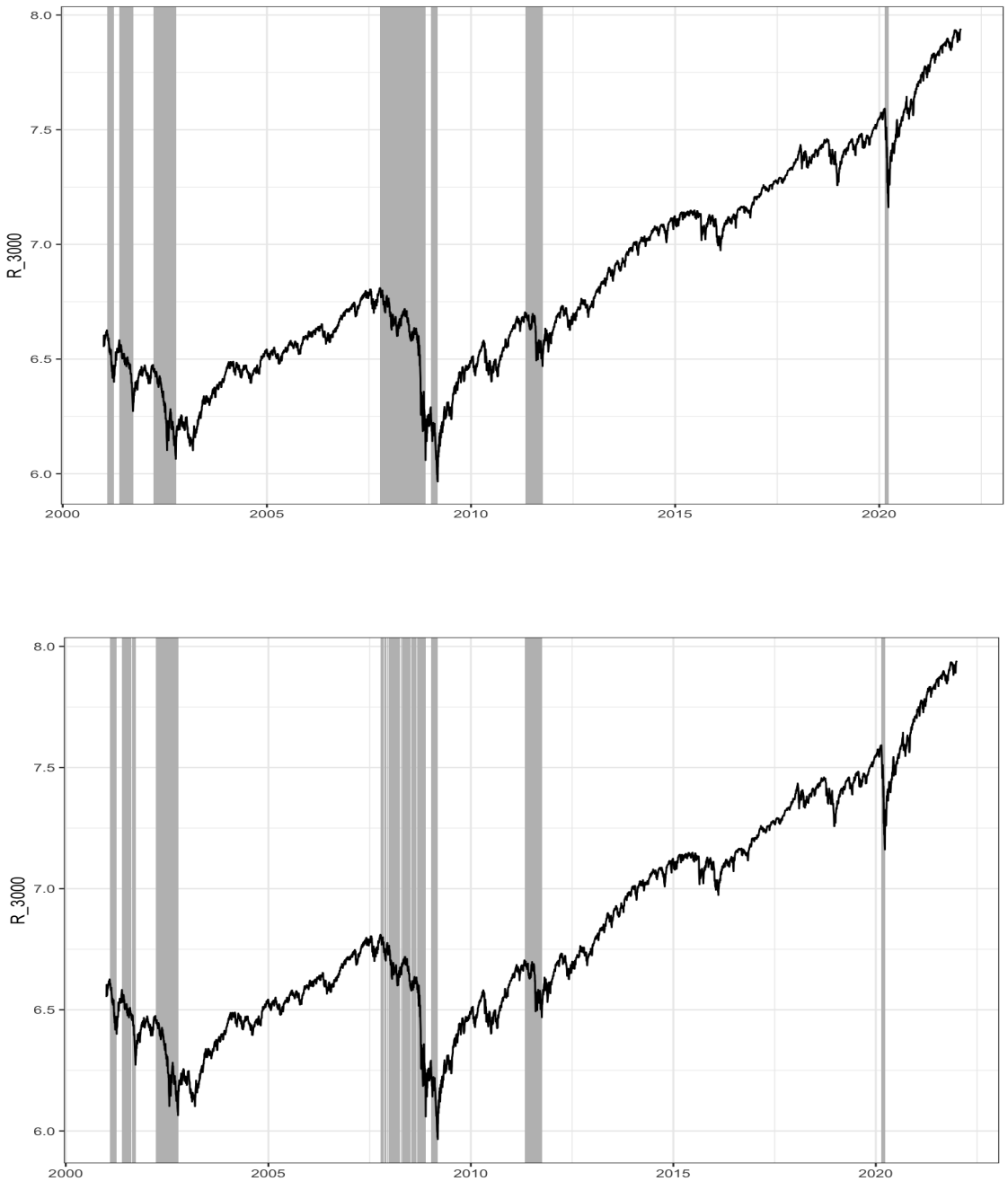

Figure 2a: A comparison of before and after bear market classifications

(The shaded areas represent bull markets. Top: The LT algorithm classifications before taking into account static volatility returns. Bottom: The LT algorithm classifications after excluding static volatility returns.)

Figure 2b: A graph of the bull markets in our dataset according to the LT algorithm.

Lastly, for our trading strategy, we shorted 50% on the Russell 3000 index when the market is bear, longed the Russell 3000 index when the market is bull, and held while the market was static.

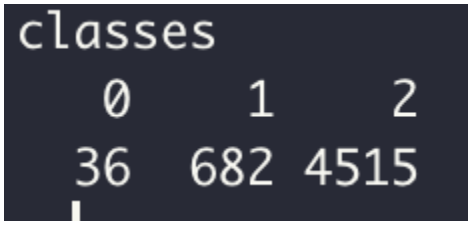

Figure 2c: A table of the labels in our total dataset. 0 = static, 1 = bear, 2 = bull.

Why Neural Networks?

Out of the recommended approaches, we chose Neural Networks because of their prediction effectiveness. While rule-based models are superior for in-sample identification, they are inferior for out-of-sample forecasting. We wanted to remedy this with the predictive power of the neural networks' hidden layers. Further, unlike a Hidden Markov Approach, which would require assumptions, we do not require any assumptions. And lastly, we thought that explainability did not matter as much as effectiveness for the sake of this problem.

While HMM and KF approaches might fit the problem better, including classification along with prediction, in general, they do not predict better than neural networks. HMMs can be used to predict hidden states, but only of the kind that the forward model is expecting. On the other hand, networks can be used to predict a not yet observed state, e.g. future states for which predictors are available.

RESULTS

In all honesty, our results were not very impressive. With the definition of static markets being low volatility bear low return bull, our results were not able to beat the market. Trading returns were very volatile, with an average validation set error of around 60%, and our trading results were around 20% positive returns. We believed that this was because our labels were too sparsely separated, which resulted in the neural network not being able to learn the bear bull and static patterns successfully. In an attempt to test for a better result, we tried a redefinition of static markets.

Redefining Static Markets

After redefining our static markets as just low volatility bear markets, we were able to successfully fit a neural network with average validation set error of around 90%, and our trading results were able to surpass the market cumulative returns, albeit being inconsistent across fitted neural networks. See Figure 3a, Figure 3b, and Figure 3c for our best run.

```
epoch: 96,  loss: 0.7154220342636108,   train acc: 0.9698162729658785,   valid acc: 0.9186046511627903
epoch: 97,  loss: 0.7081066966056824,   train acc: 0.9769028871391071,   valid acc: 0.9162790697674414
epoch: 98,  loss: 0.6913440823554993,   train acc: 0.9797900262467182,   valid acc: 0.9232558139534881
epoch: 99,  loss: 0.6596572995185852,   train acc: 0.9811023622047237,   valid acc: 0.9162790697674417
epoch: 100, loss: 0.6610328555107117,   train acc: 0.9713910761154849,   valid acc: 0.9116279069767439
```

Figure 3a: A table of final training and validation accuracies over 100 epochs

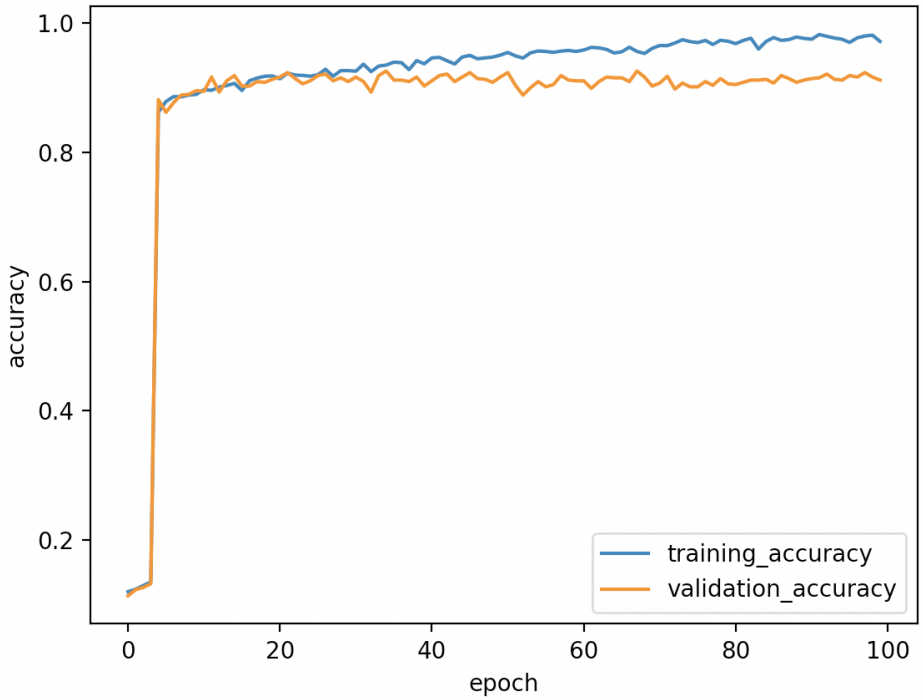

Figure 3b: A table of final training and validation accuracies over 100 epochs

Our trading strategy was very simple: we shorted in bull markets, longed in bear markets, and held in static markets. Due to our definition and the way we incorporated the 'static' market, our returns unsurprisingly resembled the Russell 3000 market returns. See Figure 3c.

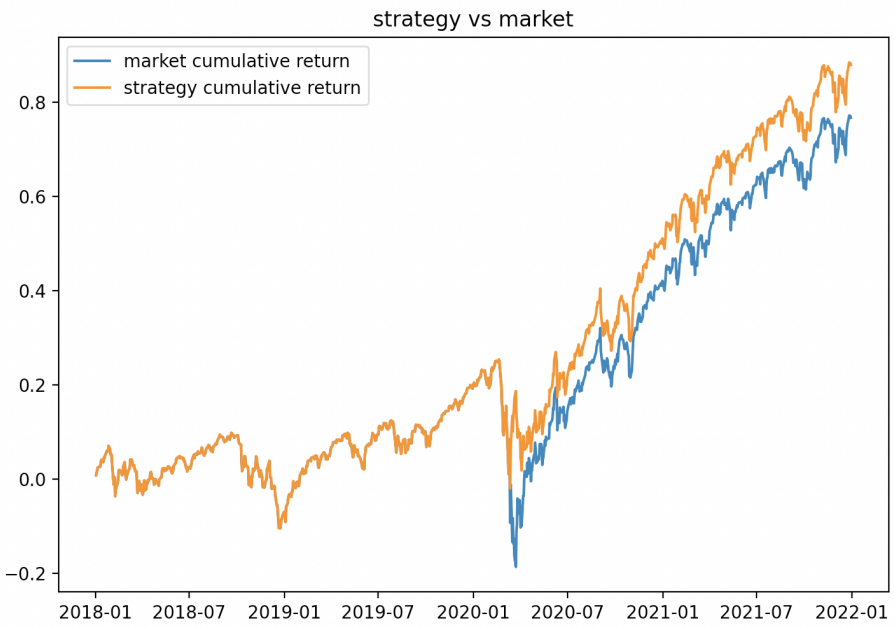

Figure 3c: Returns of our trading strategy returns compared with market returns

CONCLUSION

We built a methodology to predict the states of the US equity market by using artificial neural networks. 13 main features which capture the trend of global markets from 2001 to 2017 are introduced for training the neural network. The training accuracy and validation accuracy show that the model fits the feature in the way we expected and has relatively robust predictability. We then used this neural network to predict the market states from 2018 to 2021 and implement the pre-designed trading strategy based on our prediction. The result shows that the cumulative return of our strategy is higher than buy-and-hold.

Given our results, we can confidently say that neural networks are effective at predicting bear/bull market states. But the incorporation of other variables like static markets needs to be better thought out. Through our redefinition of static markets, it seems that the neural networks have a hard time learning patterns if the bear bull static market data are sparsely separated. We were only able to obtain the results we got by minimizing the existence of static market data.

Potential Problems and Improvements

The shortfalls of the LT algorithm are that varying the values of bull and bear thresholds could cause the timing of bull and bear markets to change in violation of principle P4. For example, as λbear decreases, smaller market falls could be designated as bears. For this project, we used λbear = λbull = 20, or 20%.

Overall, the static period was the hardest to define. And given our definition, it was also hard to predict with the neural network. This could be remedied by using some unsupervised learning method to define the static periods more rigorously. The 'discreteness' of the classifications of each day gets in the way.
We note that for our approach, the general idea is that we first define a set of labels with some method, then apply a neural network over it. We would recommend trying different definitions of labels to experiment with efficiency in the future.

REFERENCES

1. Nyberg, Henri. "Predicting bear and bull stock markets with dynamic binary time series models." *Journal of Banking & Finance* 37.9 (2013): 3351-3363.
2. Lunde, Asger, and Allan Timmermann. "Duration dependence in stock prices: An analysis of bull and bear markets." *Journal of Business & Economic Statistics* 22.3 (2004): 253-273.
3. Alex, Botte, and Doris Bao. "A Machine Learning Approach to Regime Modeling." *Two Sigma*, 10.6 (2021).
4. Hanna, Alan J. "A top-down approach to identifying bull and bear market states." *International Review of Financial Analysis* 55 (2018): 93-110.
5. Pagan, Adrian R., and Kirill A. Sossounov. "A simple framework for analyzing bull and bear markets." *Journal of applied econometrics* 18.1 (2003): 23-46.