# Introduction

## Background & Aim

Being students of the Master of Quantitative Finance program, my project partner, Joel Cappelli, and I decided to look to the world of finance for inspiration. Neural networks have had numerous implementations within finance, including bankruptcy or failure prediction, and market volatility forecasting. However, we decided to focus our efforts on the challenging yet rewarding area of time series forecasting, specifically of stock market returns.

The aim of our project is to be able to predict stock market returns with an accuracy better than chance. This is one of the more challenging problems in finance and has garnered much research attention over the years. In theory, this would enable us, or any market participant, to make profitable trading decisions and beat average market returns. The techniques we look at are multi-layer feed-forward networks, generalised regression neural networks and probabilistic neural networks.

## Literature Review

In his seminal paper, Fama (1970) developed the efficient-market hypothesis, which states that stocks always trade at their fair value and reflect all available information. This implies that it is impossible to predict market movements with any consistency, since all past publicly available information has already been priced in. Decades later, his work remains highly influential and led to the creation of the index fund, one of the world’s most popular forms of investment, which aims replicate the returns of a stock index rather than try to outperform it through expert selection. The same theory also forms the basis of popular investment advice for institutional and private investors (Malkiel, 2012).

Despite the claims of the efficient market hypothesis, empirical research has shown that markets are not always entirely efficient. Recent works have rejected the random walk model, through the development of a statistical model that is able to predict market returns with results better than chance (Lo & MacKinlay, 1988).

Given the evidence of predictability in stock market returns, numerous models have been applied with varying effectiveness, ranging from regression models (Fama & French, 1993) to support vector machines (Huang, Nakamori, & Wang, 2005). Neural networks have been applied to the forecasting of stock market and exchange rate returns by Enke & Thawornwong (2005), Kaastra & Boyd (1996) and Yu, Lai, & Wang (2008), among others. In all cases, publicly available financial and economic time series are used to predict returns. Enke & Thawornwong point out there is little evidence to support the existence of a linear relationship between the inputs and returns, making the case for a non-linear model. Additionally, neural networks avoid having to pre-specify model parameters, which is particular useful in a domain where little is known about the relationship of the processes being modelled.

Other applications of neural networks within the world of finance have included volatility forecasting (Hamid & Iqbal, 2004) and bankruptcy prediction for credit risk (Atiya, 2001).

## Project Specifications

We decided to base our project on the work of Enke & Thawornwong (2005). Their work has been heavily cited in the literature, and unlike some other papers, they provide a good deal of transparency on the implementation of their neural network. Additionally, they employ a number of techniques and variations to make the project suitably challenging, while still leaving areas for improvement and experimentation.

Instead of forecasting the United States S&P 500 stock index, we will be attempting to apply a neural network to the Australian market, reflected by the S&P/ASX 200 index. This index covers approximately 80% of the entire Australian equity market by capitalisation, and is recognised as the benchmark index for the domestic market.

While Enke & Thawornwong used monthly financial and economic data to predict the next month’s returns, we found early on that the Australian market is much younger and does have the same history as the U.S. market. For example, while the U.S. has many economic indicators dating back to the 1960s and earlier, these indicators are either not available in the Australian market or are only available for the past 20 years or so. Since this would have severely limited the number of data points we have to train with, we elected to use daily financial and econmic data instead. This also meant reconsidering our choice of input variables to ones that better reflect daily fluctuations in the market.

Enke & Thawornwong point out that there is evidence to show trading stratgies guided by forecasting direction (sign) of stock market returns are more effective than those based on the level (value) of returns. We will attempt to replicate their results and develop neural networks to predict both the direction and the level of the S&P/ASX 200 index. We will compare the multi-layer feed-forward neural network to the generalised regression neural network (GRNN) for level estimation, and the probabilistic neural network (PNN) for direction classification.

# Methodology

## System Design

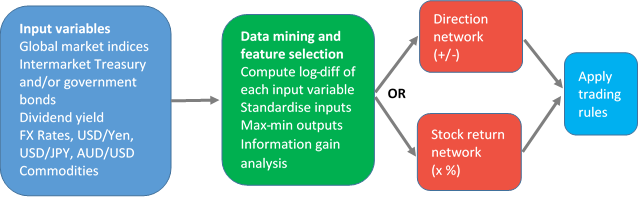


Figure . System design diagram

A high-level diagram of the planned system is shown in Figure 1. The initial step involves collection the market data, including stock market, currency, bond market, commodity and economic indicator time series. From there we need to clean the data, calculate returns and select the variables that are the most useful. With the input data ready, we then feed these into our neural networks for training and testing. As mentioned, we will be using separate networks for forecasting stock market direction and returns. Once we have our trained neural networks, we will evaluate their performance on the test and validation data sets. We consider a number of different performance measures, including prediction accuracy and trading profitability. These steps are described in more detail below.

## Data Selection

Selecting the input variables for our neural networks is an important decision that can greatly affect the results of our predictions. In coming up with an initial list of candidate inputs, we looked to the literature, Enke & Thawornwong (2005) in particular, as well as our own additions based on knowledge of the Australian market. Through this we produced a list of over a hundred possible inputs, using data available from Bloomberg[[1]](#footnote-1), Quandl[[2]](#footnote-2) and the RBA[[3]](#footnote-3). After finding we did not have sufficient sample data for monthly time series forecasting, we made the decision to predict daily price changes instead. We obtained 16 years of daily data, covering 4275 trading days from May 2000 to May 2016. The initial list of variables we used are shown in Table 1.

|  |  |  |
| --- | --- | --- |
| **ID** | **Data Label** | **Description** |
| x1 | S&P500\_DAILY\_PX\_LAST | S&P 500 (U.S.) |
| x2 | STFINL\_DAILY\_PX\_LAST | S&P/TSX Financials Sector Index (Canada) |
| x3 | SHCOMP\_DAILY\_PX\_LAST | Shanghai Composite Index (China) |
| X4 | ASX200\_INDX\_GROSS\_DAILY\_DIV | S&P/ASX 200 Daily Dividends |
| X5 | AUDUSD\_CURRENCY | AUD/USD (Currency) |
| X6 | XAU\_CURRENCY | Gold (Commodity) |
| X7 | CRUDEOIL\_COMMODITY | WTI Crude Oil (Commodity) |
| X8 | 90D\_BANKBILL | Australian 90 Day Bank Bill |
| X9 | OIS\_1M | Australian 1 Month Overnight Index Swap |
| x10 | OIS\_3M | Australian 3 Month Overnight Index Swap |
| x11 | AUD1Y\_SWAP | Australian 1 Year Interest Rate Swap |
| X12 | AUD10Y\_GOVT | Australian Government 10 Year Bonds |
| X13 | USD10Y\_GOVT | United States 10 Year Treasury Note |
| x14 | USDJPY\_CURRENCY | USD/JPY (Currency) |

Table . Initial set of predictor variables

By isolating the predictor inputs with the most information content, we were able to reduce the dimensionality of the problem, with the aim being to reduce overfitting and reduce training time. Less (misleading) data means the accuracy of the model improves and the algorithms are able to train faster. The methods we employed for predictor input selection were:

1. Ensemble trees – model based ranking
2. Univariate input selection
   1. Correlation heat-map
   2. Linear regression coefficient tests

## Ensemble trees – model based ranking

Tree-based methods such as decision trees and random forests can be applied to dimensionality reduction when the relationship between a feature and the response variable is non-linear. To avoid overfitting it is important to ensure the depth of the tree is kept relatively small. Using k-fold cross-validation, we were able to generate a large number of shallow trees and compare the information content of each input variable. Figure 2 shows the average accuracy of each variable across 10 folds of cross-validation.



Figure . Cross-validation results from the random forest classifier

From the results we can see that four variables yielded less than 50% accuracy on the cross-validation tests. This led us to removing 90-day Bank Bills (x8), 1 Month OIS (x9) Australian 10 Year Government Bonds (x12) and U.S. 10 Year Treasury Notes (x13).

## Univariate input selection

Univariate input selection examines each input individually to determine the strength of the relationship of that input with the target variable. Data with very similar trends are also likely to carry very similar information. We used a correlation heat-map, as shown in Figure 5, to identify and remove redundant inputs. Here a threshold of 0.85 was used to remove the *OIS\_3M* variable, as this time series can be adequately represented by the data in *OIS\_1M*.

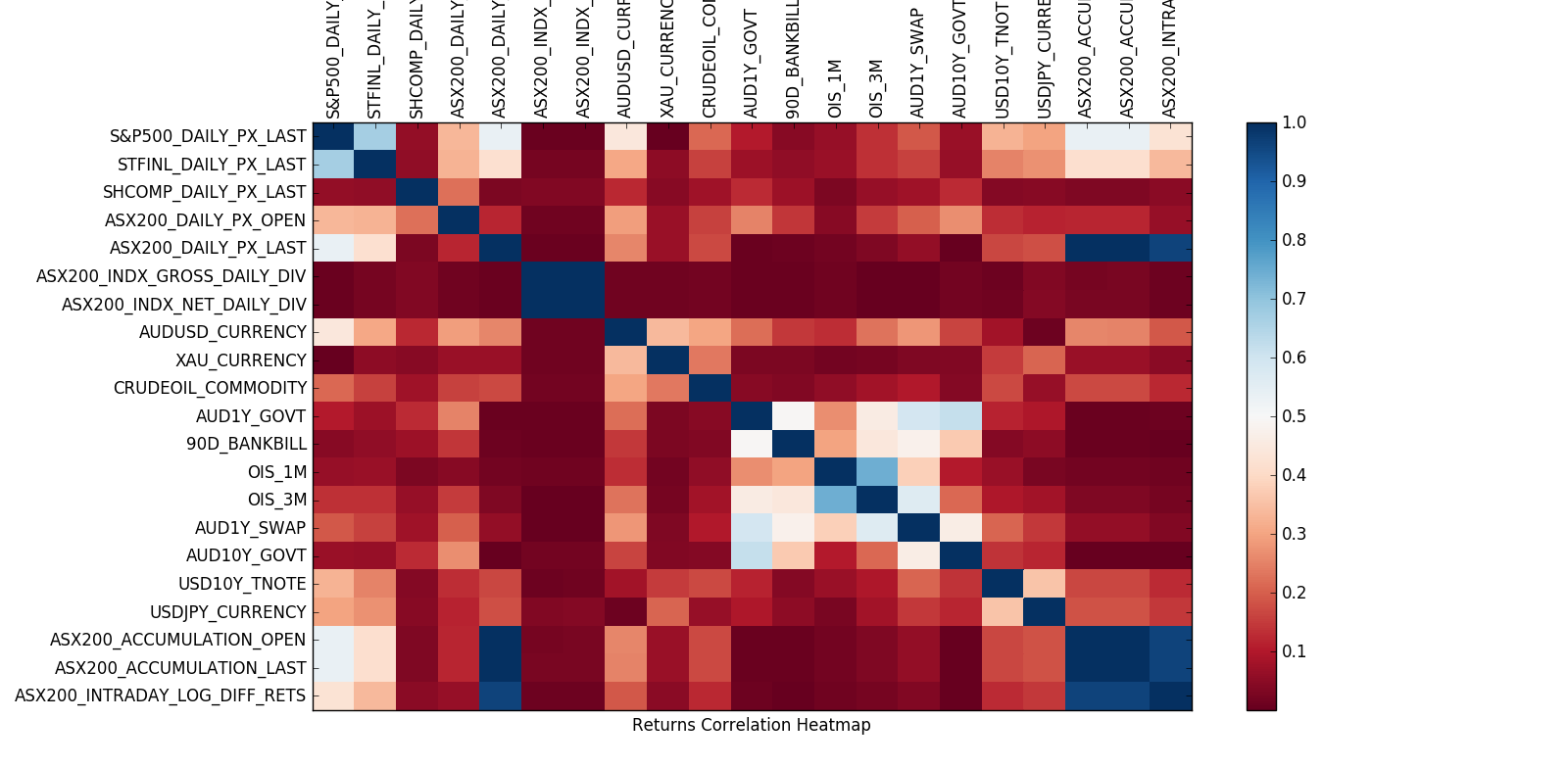


Figure . Input variable correlation heat map

|  |  |  |
| --- | --- | --- |
| **ID** | **Data Label** | **Description** |
| x1 | S&P500\_DAILY\_PX\_LAST | S&P 500 (U.S.) |
| x2 | STFINL\_DAILY\_PX\_LAST | S&P/TSX Financials Sector Index (Canada) |
| x3 | SHCOMP\_DAILY\_PX\_LAST | Shanghai Composite Index (China) |
| X4 | ASX200\_INDX\_GROSS\_DAILY\_DIV | S&P/ASX 200 Daily Dividends |
| X5 | AUDUSD\_CURRENCY | AUD/USD (Currency) |
| X6 | XAU\_CURRENCY | Gold (Commodity) |
| X7 | CRUDEOIL\_COMMODITY | WTI Crude Oil (Commodity) |
| x11 | AUD1Y\_SWAP | Australian 1 Year Interest Rate Swap |
| x14 | USDJPY\_CURRENCY | USD/JPY (Currency) |

Table . Reduced set of predictor variables

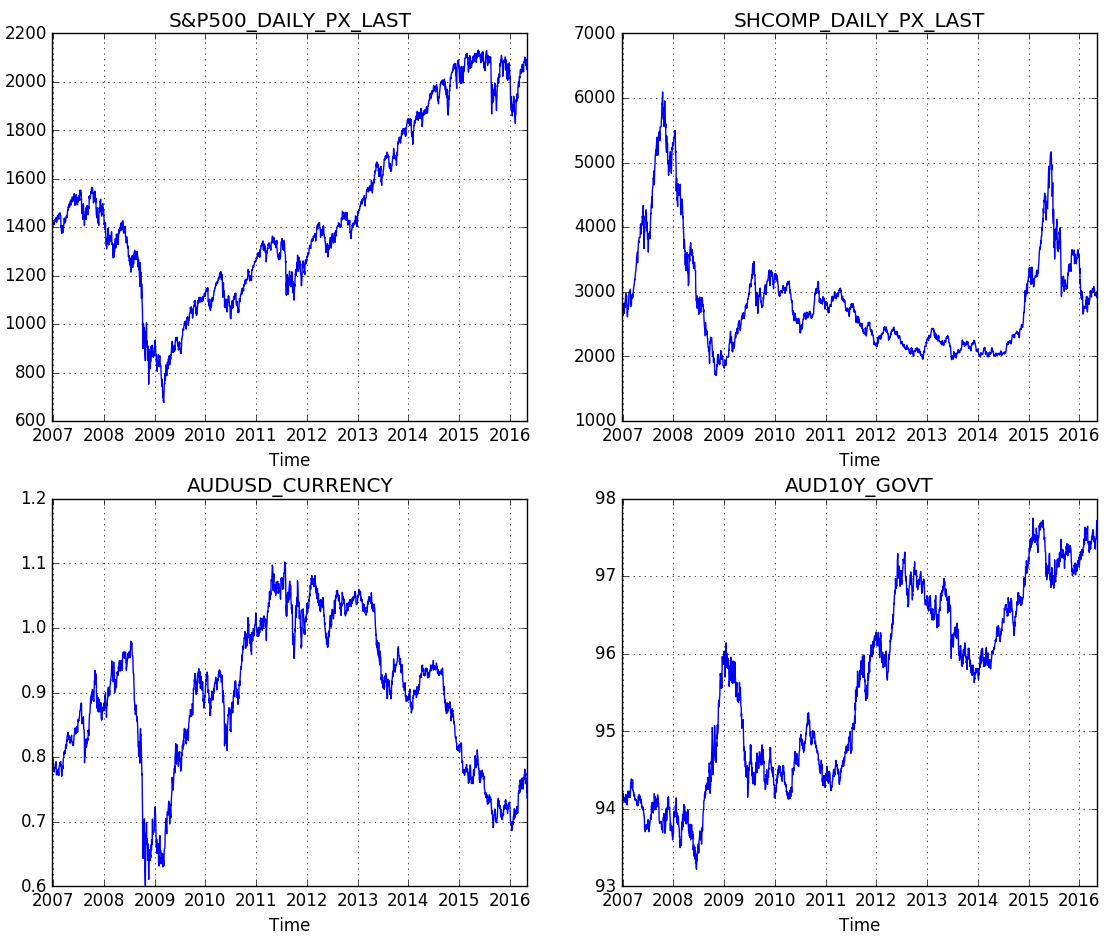


Figure . Sample of four input variables

In predicting the daily price changes of the S&P/ASX 200 index, careful attention had to be given to ensure we weren’t using data that hadn’t been seen at the start of the trading day. Doing so would effectively mean we were looking in the future and would invalidate our test results. When the ASX market closes at 4 PM, it will open up higher or lower at 10 AM the next day due to overnight news and movement in foreign markets. However, even knowing the difference, it would not be possible to profit from it at the start of the day as it would require purchasing (or selling) the index at the previous day’s closing price, which is not possible. For this reason, instead of predicting the log return between daily closing prices, we will look at the log returns from the 10 AM opening price to the 4 PM closing price each day. Our output variable is then the value of these returns for level estimation, and the sign (+/-) for the direction classification.

For most of the input variables, we use the log returns between daily closing prices, except for dividend data where use the actual prices, owing to the data being quite sparse since most days have zero dividends. As we are using the previous day’s financial and economic data to predict what the local market will do today, we apply a *lag* of 1 day to the time series data.

## Implementation

To develop our solution, we elected to use the Python 2.7 programming language. It’s main advantages for our purposes were the speed of development it enables, and the prevalence of open-source libraries for the purposes of machine learning, plotting and data preprocessing. In particlar, we made use of the NumPy scientific computing package, as well as NeuPy for its artificial neural network algorithms.

With our input and output variables selected, we then import these into our system from Excel spreadsheet and adjust them as desribed in section 2.2. The data is split in a ratio of 60%, 20%, and 20% for our training, test and validation sets, respectively. This gives us 10 years for training and 6 years for out-of-sample validation and testing. The training data is then fed into our neural network models. As was done by Enke & Thawornwong (2005), we first attempt to forecast market return levels and direction using multi-layer feed-forward neural networks. We then compare the results to those of a generalised regression neural network (GRNN) for level estimation, and a probabilistic neural network (PNN) for direction classification.

## Multi-layer feed-forward neural network

The feed-forward class of neural networks were the first and simplest type of neural network devies. In this type of network, the training samples are fed simulatenously into an input layer, the weighted outputs of these are fed through one or more hidden layers, whose weighted output are input into the output layer, which produces a prediction for the given sample. While there are a number of learning techniques, the most popular is back-propagation. In this method the output values are compared with the expected values, and the error is fed back through the network and the weights of each neuron are adjusted to reduce the error by some small amount.

We had one input neuron for each input variable, as well as a bias node. The single output neuron represented the daily return for our level estimaton network, or a +1/-1 value for our direction classification network. For our backpropagation learning algorithm, we elected to use the conjugate gradient optimisation method. Why?

Parameters, hidden neurons, how many and why, what activation function, bias. Learning rate , convergenece, epochs, when did we stop learning.

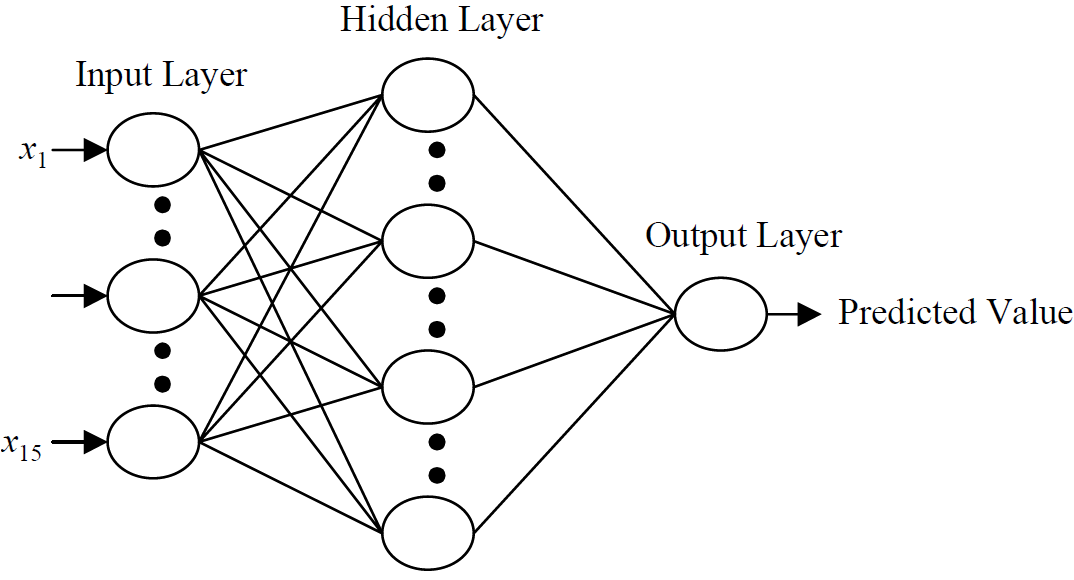


Figure . Multi-layer feed-forward neural network (Enke & Thawornwong, 2005)

## Generalised regression neural network

The generalised regression neural network (GRNN) comprises of four layers, as shown in Figure 5. The first and final layers represent the input and output vectors. The first hidden layer represents the *n* training observations and the two neurons in the second hidden layer represent the numerator (*s*) and denominator (*t*) in the estimation of the conditional expectation E[y|x], the expected output *y* given input *x*.

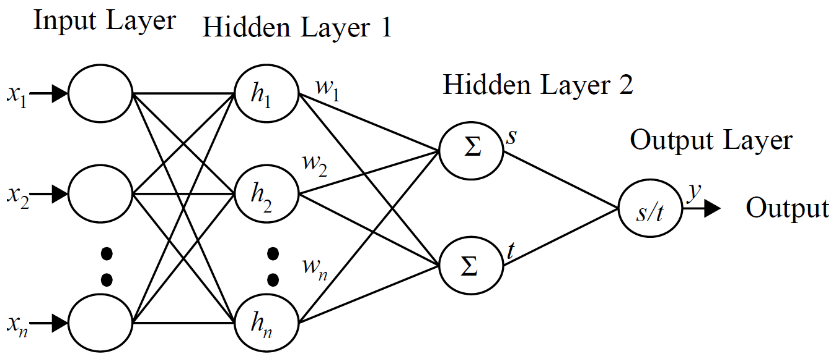
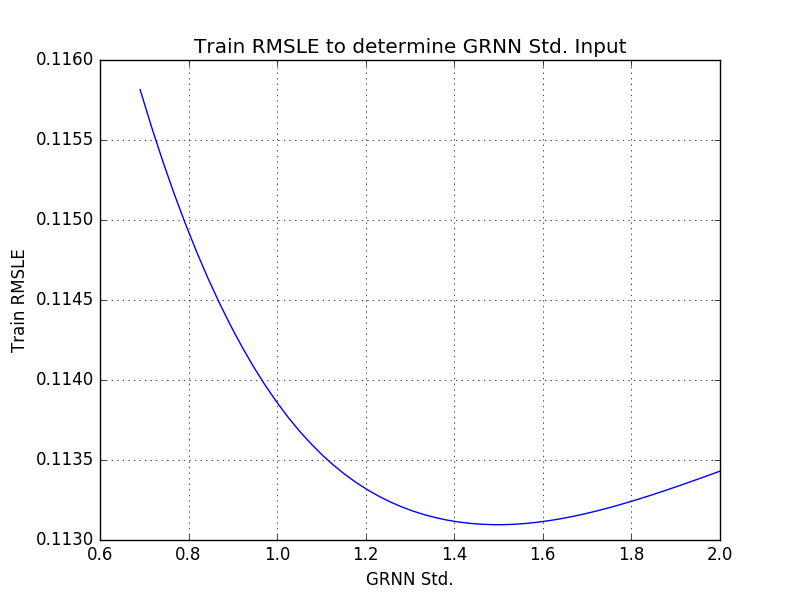


Figure . Generalised regression neural network (Enke & Thawornwong, 2005)

The GRNN can be designed very quickly as it has fewer parameters and does require any early stopping technique during training.

Smoothing parameter for PDF function, what and why



## Probabilistic neural network

The probabilistic neural network (PNN) is predominantly used for classification, as it’s able to find decision boundaries between categories of patterns. The PNN learns from sample data instantly and estimates the probability density functions of the classes learnt. The PNN is based on the theory of Bayesian classification and its network consists of four layers, as shown in Figure 6. The first layer represents the inputs and the number of neurons in the first hidden layer is equal to the number of training samples. When it receives an input vector, the first hidden layer calculates the closeness to the training input vectors. The second hidden layer sums these elements for each class of inputs to produce a vector of probabilities, which are fed into the output layer whose activation function selects the maximum of the probabilities to determine the output class.

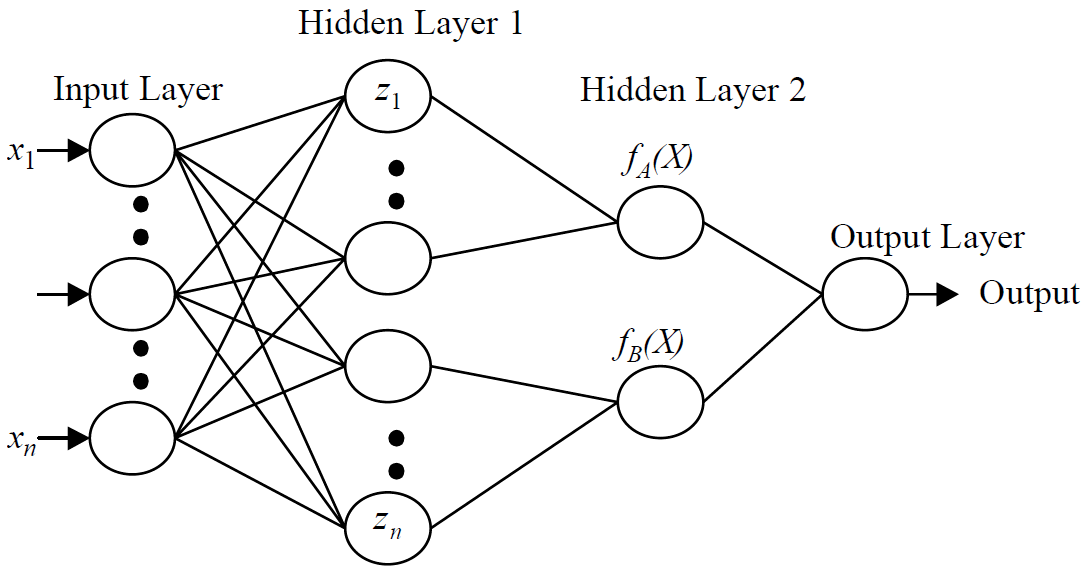
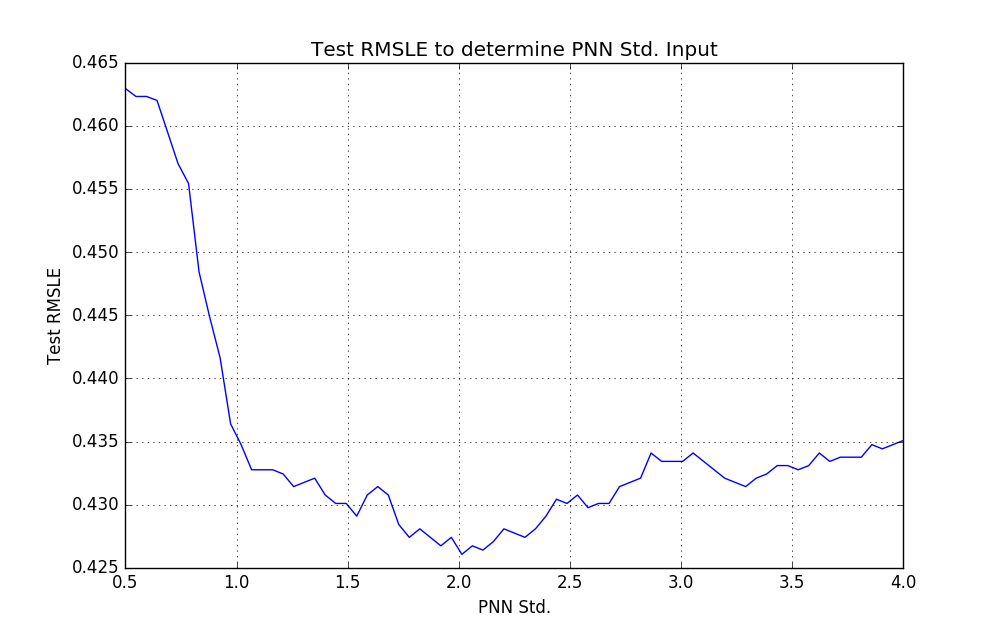


Figure . Probabilistic neural network (Enke & Thawornwong, 2005)

The advantages of the PNN are their ability to learn instantly, and they can be more accurate than multi-layer perceptron networks but this comes at the cost of being slower to classify new cases and requiring more memory to store the model.

As with the GRNN, the design of the PNN is much more straightforward than the multi-layer feed-forward network.

standard deviation for PDF function, what and why, smoothing parameter



## Evaluation Metrics

In order to evaluate our neural networks, we consider three different metrics: root-mean-square error (RMSE), accuracy percentage and trading simulation.

RMSE is a frequently used measure of the differences between values predicted by a given model and the actual values observed. When used for out-of-sample data, the RMSE represents the prediction error and serves as a good measure of accuracy by combining the total error into a single metric. Since it is scale-dependent, we’re able to use to compare different neural networks on the same input variables, but not between different sets of variables.

Measuring accuracy percentage is typically used in classification problems such as our direction classifier neural network, where we can look at the number of samples correctly classified as a portion of total samples. However, by taking the sign of our level estimates, we can also apply this measure to our level estimation neural networks. At a minimum, we’ll be looking for better than chance accuracy (50%) and ideally we’ll be looking to replicate the results of Enke & Thawornwong, with greater than 60% accuracy.

The last metric we look at is a trading simulation, which attempts to replicate how our models might perform in a real-world scenario. We do this by starting with a hypothetical dollar amount and base our buying and selling decisions on the prediction of our neural networks. We’re assuming that it is possible to *short sell*, that is, selling a financial instrument we don’t own and profiting when the market goes down. Our forecasting model is designed to be run at the start of the trading day, where it will provide a prediction on whether the S&P/ASX 200 market is going up or down that day. In this manner, our trading strategy will either buy at the start of the day and sell when it closes, or short sell the market at the open and buy it back at the close of day. It does not hold any position overnight. For the purpose of comparison, we will look at a typical buy and hold strategy, where we purchase the S&P/ASX 200 index and hold it for the duration of the test period. It is worth noting that we ignore taxes and transaction costs in our simulation, whose inclusion would favour the buy and hold strategy.

# Results & Discussion

The predictive performance of the final neural networks were evaluated against the untouched out-of-sample test set. The test set compromised 20% of our data, or roughly x data points.

Level estimation charts

The predictive performance were evaluated using the three metrics described in section 2.3.3.

RMSE table

Accuracy

|  |  |
| --- | --- |
| **Model** | **Accuracy** |
| Level estimation – MLP |  |
| Level estimation – GRNN |  |
| Direction classification - MLP |  |
| Direction classification - PNN |  |

Table . Accuracy

P&L

In our trading simulation, we start with a hypothetical amount of $5,000, which is the level of the S&P/ASX 200 index at the start of the period. In the buy and hold strategy, we simply take a position in the index and hold it until the end, collecting dividends along the way. In doing so, we end up with $20,000 at the end of the period. Compare to other strategies.

# Conclusion

Conclusion

Future directions

# References

Atiya, A. (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on Neural Networks, 12*(4), 929-935. doi:10.1109/72.935101

Enke, D., & Thawornwong, S. (2005). The use of data mining and neural networks for forecasting stock market returns. *Expert Systems with Applications, 29*(4), 927-940. doi:10.1016/j.eswa.2005.06.024

Fama, E. (1970). The Efficient Capital Markets: A Review of Theory and Empirical Work. *Journal of Finance, 25*(2), 383. doi:10.2307/2325486

Fama, E., & French, K. (1993). Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics, 33*(1), 3-56. doi:10.1016/0304-405x(93)90023-5

Hamid, S., & Iqbal, Z. (2004). Using neural networks for forecasting volatility of S&P 500 index futures prices. *Journal of Business Research, 57*(10), 1116-1125. doi:10.1016/s0148-2963(03)00043-2

Huang, W., Nakamori, Y., & Wang, S.-Y. (2005). Forecasting stock market movement direction with support vector machine. *Computers & Operations Research, 32*(10), 2513-2522. doi:10.1016/j.cor.2004.03.016

Kaastra, I., & Boyd, M. (1996). Designing a neural network for forecasting financial and economic time series. *Neurocomputing, 10*(3), 215-236. doi:10.1016/0925-2312(95)00039-9

Lo, A. W., & MacKinlay, A. C. (1988). Stock Market Prices Do Not Follow Random Walks: Evidence from a Simple Specification Test. *Review of Financial Studies, 1*(1), 41-66. doi:10.1093/rfs/1.1.41

Malkiel, B. G. (2012). *A Random Walk Down Wall Street: The Time-Tested Strategy for Successful Investing (Tenth Edition).* New York: W. W. Norton & Company.

Yu, L., Lai, K., & Wang, S. (2008). Multistage RBF neural network ensemble learning for exchange rates forecasting. *Neurocomputing, 71*(16-18), 3295-3302. doi:10.1016/j.neucom.2008.04.029

# Appendix A

A

1. The Bloomberg Terminal system provides access to global historical financial data. [↑](#footnote-ref-1)
2. Quandl is a free online provider of financial market data available from [quandl.com](http://www.quandl.com). [↑](#footnote-ref-2)
3. The Reserve Bank of Australia release a number of economic indicators at [rba.gov.au/statistics/tables](http://www.rba.gov.au/statistics/tables/). [↑](#footnote-ref-3)