**Master thesis presentation notes**

**INTRODUCTION**

The field of machine learning has advanced significantly in recent decades, and, at the same time, computational power has improved to the point where training large machine learning models, such as artificial neural networks, is now accessible. Consequently, there has been a rise in the use of these models within the financial sector.

There are alot of different applications for these machine learning models, but one is training algorithms on historical data to identify patterns that may recur in the future. The rationale behind this approach is that historical data contains structure that will be repeated in the future, meaning that past price developments of an asset hold valuable information for predicting future price developments.

This approach challenges traditional market theories, such as the efficient market hypothesis.

Despite this, price patterns have proved reliable enough to allow multiple investors to reap substantial financial gains and development into organizations with headcounts in the thousands, solely focused on identifying and trading these pricing patterns. One of the best-known such funds is Renaissance Technologies, which bolstered no less than a 66% annualized return during the 30-year span from 1988 to 2018.

**THIS THESIS**

This thesis aims to construct and assess artificial neural network models intended for use in trading algorithms. Given historical returns, the models are trained to forecast the direction of asset price returns the following day. The predicted return directions are then input into a trading algorithm.

The performance of these models are then benchmarked against naive trading strategies and the underlying asset itself.

The underlying assets are government bond contracts, as requested by Handelsbanken Fonder.

**SETUP / THEORY**

So first off, lets very quickly go back to the basics so that we are on the clear of what is actually happening.

The trading algorithms built are taking long or short positions in an asset, that is, it is trading a single asset. The trades are taken based on the output of a neural network trained on that same asset’s historical return data.

When the model output is "up," indicating a positive prediction for the next trading day's return, the trading algorithm will take a long position at the end of the preceding trading day. This position will be held until the model predicts a "down" for the next trading day, at which point the trading algorithm will swap the long position for a short position at the end of the preceding trading day.

Since the neural network model's final node is a sigmoid activation function that generates predictions ranging from 0 to 1, a threshold separating "up" and "down" must be defined. This threshold has been set to 0.5. If the output is precisely 0.5, the position remains unchanged. The choice of 0.5 stems from that the output could be interpreted as a pseudo-probability. A value of 0.5 would imply that the model believes there is an equal likelihood that the next day's return will be positive or negative.

**The models** (keep short and light)

The neural network model structures used were Long Short-Term Memory and  
Gated Recurrent Units. Both models consisted of 4 layers with 128 LSTM or GRU  
nodes, respectively, followed by one dense layer of 32 ReLU nodes, and finally a single  
sigmoid output neuron.

**BENCHMARKING**

In order to benchmark the algorithms, very simple models were implemented and then compared to the neural networks. Specifically, an auto-regressive model of order one, called **Naive**, and one of order 10, called **Trend**.

The Naive model simply takes the most recent return as its prediction, which implies that the direction (up or down) of the last return will be the predicted direction of the next return.

The Trend model's prediction is the weighted average of the last 10 returns, this means that if this weighted average is bigger than zero the model will predict that the next return is positive and vice versa.

**RESULTS**

**Benhmarking**

The Trend model performs the best across almost all measures (excluding TPR, NPR, PPV, and NPV); it has the highest accuracy, Sharpe ratio, Sortino ratio, gross return, and the lowest maximal drawdown.

When the same measures are observed, the GRU model is second best, except for maximal drawdown, where the naive model is slightly better.

**Trading performance - LSTM**

Average trading performance is plotted, LSTM models are found in red. The Trend model in green has the highest gross return over the period. The simple Naive model in purple has the second highest gross return, outperforming the Neural model.

**Trading performance – GRU**

GRU models are seen in red. The Trend model in green has the highest gross return over the period, followed by the Neural model. Note that the Naive and Trend models’ performances are identical to those in the previous Figure, this is expected since these models are deterministic.

**Number of days long vs short**

The table displays the number of trades taken by each model and the percentage of days that the model held a long or short position. When observing the averaged results, you can seen that the Trend model took the least amount of trades, followed by the LSTM model. The number of long and short days are almost the same for the LSTM, GRU, and Naive models; in contrast to the Trend models which on average held a short position 63% of the days. Remeber this!

**CONCLUSION & DISCUSSION**

No definitive conclusions can be drawn regarding the efficacy of neural networks in reliably forecasting expected returns of credit futures and utilizing them in trading algorithms.

While the ANN models proposed in this study did not outperform simple predictors, it is important to note that generalization beyond the specific models and time series examined here is challenging. This is primarily due to the multitude of design options available when building neural networks, including input data selection, neural network type, architecture, regularization approach, and parameter selection, and so on.

It is very much conceivable that, with the appropriate design, an ANN model could provide return estimations to help a trading algorithm so that it outperforms benchmark models.

**About the trend model**

The trend model outperformed the other models tested in this study. And its simplicity makes it enticing to question whether its success was simply circumstantial. While it might be true for the results of this thesis, it is probably not true in general given that multiple studies have reached the conclusion that trend models outperform markets.

But, It is possible that the superior performance of the trend model was due to it holding a short position for 63% of the days during a period when the underlying asset was trending downward. Additionally, the model's accuracy was just 52.4%, which suggests that its impressive results were only specific to that period. As such, it is difficult to draw any firm conclusions about whether the model could consistently produce such performance in other market climates based solely on the results of this study.

The performance of the LSTM models is notably poor, and, in many cases, worse than that of the naive models. One possible reason for this could be observed in the average percentage of days when the models are long versus short. On average, the LSTM models are short for 50.1% of the days, while the naive model is short 53.3% of the days.

Alright, thank you for listening. I’ll be happy to answer any questions :)