*Udacity Capstone Project*

**Project Proposal**

# Deep Learning for Small Cap Stock Sentiment Analysis and Trading

***[Fredrik Hellander]***

***[fredrik@hellander.sel]***

## TABLE OF CONTENTS

**1. INTRODUCTION** .......... **1**
1.1 PROBLEM STATEMENT .......... 1
1.2 DATASET AND INPUTS .......... 2
**2. PROPOSED TECHNICAL APPROACH** .......... **3**
2.1 SOLUTION STATEMENT .......... 3
2.2 BENCHMARK MODEL .......... 4
2.3 EVALUATION METRICS .......... 4
2.4 PROJECT DESIGN .......... 4
**3. REFERENCES** .......... **5**

# 1. INTRODUCTION

In the domain of finance and trading data is available in abundance and quantitative and statistical analysis has been employed successfully for a long time. Today a majority of financial transactions are, at some level, executed by software and algorithms. High frequency trading robots are exploiting incorrect pricing of assets to make millions of small arbitrage trades each day, human brokers are all but replaced by software which executes given sell or buy orders to the best price given the current market condition, and hedge funds and investor build models to predict price movements.

Machine learning would seem to be an ideal fit for an industry which revolves around data but the evidence is conflicting, many attempts have been made to apply neural networks for predicting stock-market prices with limited success [1]. The rather pessimistic results from academic investigations and literature reviews is perhaps not surprising; stock markets are noisy and stochastic in nature and finding relevant patterns is not easy. Furthermore, they are also extremely competitive and short term speculative trading is in some sense a zero-sum game, to find successful trading algorithm one has to outsmart many other strategies and there is evidence that analyzing time-series of price data is insufficient [2]. However, machine learning is extensively used and given the competitiveness of the markets one could suspect that successful algorithms would not get published out of fear of making them obsolete.

Many algorithm and trading strategies trade 'momentum', they try to find stocks in a positive or negative trend and bet that this development will continue for a period of time. From my personal experience companies with a small market valuation (Small Cap and Micro Cap) exhibit more prominent trend patterns, which also last longer, than more widely traded stocks. Small Cap stocks are also more influenced by the online sentiment around the stock, i.e. how many google searches, twitter and forum post made each day. The hypothesis of this project is that time-series of price data and meta data regarding online sentiment can be used to train a small neural network to predict the price movements of Small Cap stocks and implement a successful trading strategy.

## 1.1 PROBLEM STATEMENT

This project aims to predict the price changes of Small Cap stocks from time-series of price data and online meta data. Given the historical data an estimate of the next day's gain or loss should be estimated and the validity and usefulness of the predictions will be evaluated by implementing the model in a trading strategy.

## 1.2 DATASET AND INPUTS

Historical stock prices are available for free at Yahoo! Finance [3] and can be accessed through a Python API. Daily information regarding opening price (OPEN), closing price (CLOSE), intraday high (HIGH), intraday low (LOW), number of shares traded (VOLUME) and stock price adjusted for stock splits and dividends (ADJUSTED CLOSE). From these basic quantities, many indicators can be calculated, such as simple moving averages, volatility, MACD, RSI and other technical indicators used to forecast stock prices.

The focus of this project will be Swedish Small Cap stocks. Historical data is available for several years trading history and many hundreds of companies in this category, which if relevant makes it possible to train complex deep neural networks.

The historical stock price data will be combined with online metadata to further trying to capture the sentiment and momentum of a stock. Data publicly available from Google Trends [4] can give insight into the number of google searches that has been made for a stock or keyword, the number of post per day on the public free stock forum Placera [5] can also be used as an indication of interest in a stock, as can keyword statistics from Twitter available free at Tweetchup [6].

The different data sources will be stored in a SQL database and evaluated with machine learning techniques with the goal of training a neural network using relevant inputs.

What seems to be a common approach in machine based trading algorithms is to take very long time series of historic stock prices to get enough training data. However, it is my understanding that stock pattern changes through time and patterns change and break down. Thus, an algorithm training on very old datasets has a limited ability to predict future behavior.

Instead the strategy in this project will be to use relative new stock and price data. For a specific stock all available data from e.g. 2 years will be used and make up about 10% of the total training data. The remaining 90% constitutes of random sampling from a group of stock peers. This approach will hopefully teach the algorithm current and relevant general trading patterns with a bias towards the specific stock price is predicted for .

## 2. PROPOSED TECHNICAL APPROACH

### 2.1 SOLUTION STATEMENT

Given the input data the preliminary plan is to train a neural network to create a regression model predicting the change in stock price the following day. The reason for using a neural network in a regression application is that the vector of inputs will be large and its desirable to have a model flexible enough to learn features to ignore and features to focus on and furthermore learn many different types of patterns that might not always be present.

An input vector $X$ consisting of historic price data and complied historical metadata is fed to the network. The network consists of one or more hidden layers, the output layer is a single node summing all signals output by the previous layer and denoted $\hat{Y}$, the network estimate of the next day's price movement. In training the network output $\hat{Y}$ is compared to the actual price change $Y$ and the root-means-squared-error is calculated as the error and backpropagated to make the network learn.

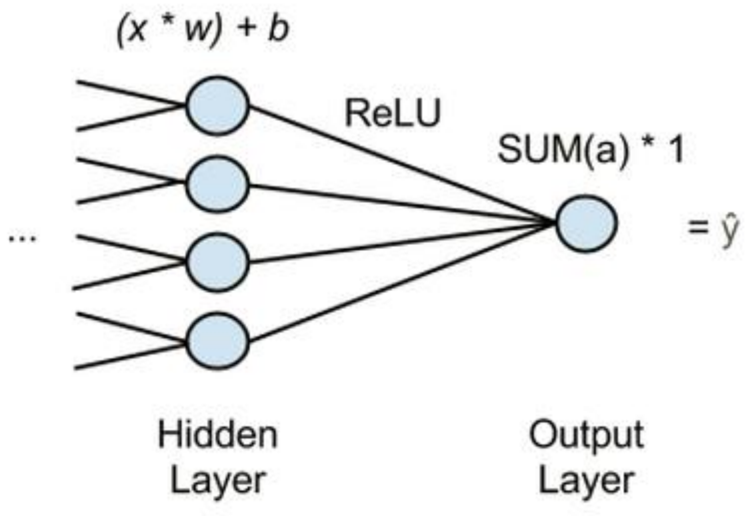

**Figure 1 – Illustration of Neural Network with regression output**

A neural network model is built for several stocks and the trading algorithm will simply buy or sell the stocks with the largest predicted price moves.

## 2.2 BENCHMARK MODEL

The trading model will be back-tested against unseen data and its returns will be compared to an equally weighted basket of the same stocks it is trading during the same period. This will tell us if the performance of the algorithm is better than simply buying a holding the same stocks and if the modelling effort is giving any additional benefit.

E.g. if the algorithm is trading Apple, Tesla and GM during a 6-month period the benchmark to compare it to would be the return of portfolio of 1/3 Apple, 1/3 Tesla and 1/3 GM bought at the start of the period and sold at the end

## 2.3 EVALUATION METRICS

The main evaluation metric to evaluate the algorithm against the benchmark will be the return on capital during the period. From the return and volatility several other performance metrics can be calculated and used.

The alpha value of the algorithm can be calculated as the difference in absolute return between the algorithm and the benchmark.

The sharp ratio can be compared between the benchmark and the algorithm. This is the annualized return divided by the annualized standard deviation (volatility) to compare risk adjusted performance.

Jensen index can be calculated as the difference in return between the algorithms return and the return predicted by the Capital Asset Pricing Model.

## 2.4 PROJECT DESIGN

The project is planned according to this theoretical workflow:

1. Data acquisition from Yahoo Finance, Google Trends, Twitter and Forums
2. Data storage and organization in SQL database
3. Generating secondary data structures such as moving averages and technical indicators
4. Preliminary data and pattern exploration using linear regressions and heatmaps
5. Building and training simple neural network, evaluate R^2 score.
6. Hyperparameter exploration and improvement of neural network
7. Back-testing and evaluation against benchmark
8. Implementation of algorithm into web-based portfolio interface

## 3. REFERENCES

[1]

Fernando Fernández-Rodríguez, Christian González-Martel and Simón Sosvilla-Rivero. On the profitability of Technical Trading Rules based on Artificial Neural Networks: Evidence from the Madrid Stock Market. Elsevier 2000

[2]

Evan Hurwitz, Tshilidzi Marwala,State of the Art Review for Applying Computational Intelligence and Machine Learning Techniques to Portfolio Optimisation. arXiv:0910.2276

[3]

https://www.finance.yahoo.com

[4]

https://trends.google.com/trends/

[5]

https://www.avanza.se/placera/forum.html

[6]

https://tweetchup.com/