Machine Learning Engineer Nanodegree

Capstone Proposal

Chitrang Patel July 14th, 2019

Implementation of the Recurrent Neural Network for Investment Trading

Domain Background

Technical analysis is a trading discipline employed to evaluate investments and identify trading opportunities by analyzing statistical trends gathered from trading activity, such as price movement and volume. Technical trading focuses on patterns of price movements, trading signals and various other analytical charting tools to evaluate a security's strength or weakness[1]. Computorization of the exchanges began in the early 1970s, with some landmarks being the introduction of the New York Stock Exchange's “designated order turnaround” system (DOT, and later SuperDOT), which routed orders electronically to the proper trading post, which executed them manually[2].

Computing already revolutionized financial trading once, it facilitated enormous numbers of calculations in a fraction of a second, and to track markets that shift in light speed. Now AI trading systems are poised to foster a second wave of innovation, one that will be the most significant transformation in finance history[3]. Recent advances in Deep Learning for financial trading have shown to outperform human traders. It is incredibly valuable for ameture investors!

This project uses Recurrant Neural Nets (RNNs), to predict stock prices. This technique is actively being studied and used in time series forecasting situations and also to real-world trading platforms. This project will use the Keras[4] library to build a RNN to predict closing value of stocks using historical data. It will show visualizations of the optimal parameters and the data vs predictions.

Problem Statement

The main task for this project is to accurately predict the future closing value of a given stock across a given period of time in the future. Almost all stocks have a large range of historical daily closing price values along with other important features like the opening, high and low prices and the volume of stocks. In principle the model should be trainable over a long period of time to develop a model. I can then determine the performance of this predicted model by back testing for their against a large period of time. For this, the plan is to use a Keras RNN model that is trainable across a time series[5] of adjusted closing stock price values. These model predicted values will be compared to the actual values. I will use this model on a few stocks such as (AAPL, MSFT, AMZN, GOOGL, GOLD, NVDA, BBRY, SGBX).

Datasets and Inputs

All the input data for this project will come from Yahoo Finance[6]. To do this, I will use a python package Yahoo Historical[7] for reproducability. The input to the program will be a stock ticker symbol for which we want to

predict the closing value. We will pull the historical data from the API to train the model: - Daily Adjusted Closing Price (target value that we are trying to predict) - Daily Volume (level of trade activity and market momentum for the specific stock) - Daily Adjusted Closing Price of S\&P 500 trust Exchange-Traded-Fund.

We will compute roling means to smooth out the noise: - Day Rolling Mean of the Daily Adjusted Closing Price of the target stock. - Day Rolling Mean of the Daily Adjusted Closing Price of S\&P 500 trust Exchange-Traded-Fund. I might add more inputs if needed while developing the actual model.

Solution Statement

I will be using the Keras library that utilizes the TensorFlow backend. I plan to use a Recurrant Neural Net with a Long Short-Term Memory model which is capable of learning from time series data. The bonus is that it is also supported by Pandas DataFrame which is the output of the python yahoo-historical API's library. I will try and improve upon the RNN usinf Ensemble techniquesand tuning various hyperparameters that go along with it. The performance of the model will measured as follows: - Compare the predicted stock price in comparison to both the actual price - Compare the predicted stock price in comparison to the benchmark model's (Linear Regression model) predicted price.

Benchmark Model

I plan to use a Linear Regression model from Scikit-Learn[8] as its primary benchmark. I will also compare it to the techniques using Support Vector Machine Regression[9]. There have been extensive studies and models using these techniques that I can compare the performance of my model to.

Evaluation Metrics

The evaluation metric will be the mean squared difference between the predicted and actual values of the target stock. This will also be repeated with the benchmark model.

Project Design

This project will be implemented using the Keras/TensorFlow library, specifically the Recurrant Neural Networks using a Long Short-Term Memory model to analyze the Time Series nature of our data.

I will take the following steps:

- Set up the project and install necessary libraries.
    - GitHub (https://github.com/chitrangpatel/RNNIT)
    - Incorporate required Libraries (Keras, Tensorflow, Pandas, Numpy, Matplotlib, Seaborn, yahoo-historical)
    - Git project organization
- Extract and preprocess the Data
    - Use Yahoo Historical API to request stock data
    - Load the data into a Pandas Dataframe

    - Scale data for regularization.
    - Calculate the rolling means of several inputs.
    - Split the data for training, validaiton and testing (60, 20 and 20 percent, respectively) across all models

- Develop the linear regression benchmark model using Scikit-Learn.

- Develop the RNN (LSTM) using Keras.
- Improve the RNN model using ensemble techniques.
- Evaluate the model using mean squared error with respect to the actual data and te benchmark model.

References

1. Technical Analysis Definition
2. Algorithimic Trading
3. The future of trading belong to Artificial Intelligence
4. Keras // Deep Learning library for Theano and TensorFlow
5. Time Series Prediction With Deep Learning in Keras
6. Yahoo Finance
7. Yahoo Historical // Python API
8. Scikit-Learn
9. Predicting Stock Price Direction using Support Vector Machines