Capstone project — Kiem Nguyen
Machine Learning Engineer Nanodegree — January 22nd, 2021

Domain Background:
Trading for profit is a difficult problem to solve. In an efficient market, buyers and sellers would have all the information needed to make a rational trading decision so the stock is should remain at its fair values. The reality is the financial market is not efficient in real life especially now when automated trading allows for thousands of transactions to occur within a nanosecond.

I have a deep passion for applying data science to the financial market. In my previous work, I helped to build an options real-time engine, futures real-time engine, and risk engines for the Chicago Mercantile Exchange so that it now flawlessly handles millions of trades moving billions of dollars every day. My work experience taught me that picking Alpha signal from trading is a very hard task, I would like to use data science and machine learning techniques to solve this problem.

Problem Statement:
Between 2010 and 2020 the S&P 500 has an annual average return of 13.6% in the past 10 years. This project would provide a practical approach to predict future price movements in financial markets based on past returns. The assumption is that certain patterns in financial markets repeat themselves such that past observations can be leveraged to predict future price movements. (*Hilpisch, Yves. 2020. Artificial Intelligence in Finance: A Python-Based Guide.)*

Datasets and Inputs:
I will use pandas-datareader library to extract end-of-day stock pricing data from Yahoo Finance from 2011-2020 from Quandl data source. pandas-datareader library provides functions to extract data from various internet sources in a pandas data frame.

Before using the pandas-datareader library, I will need to create an account on Quandl first and obtain my API key. I will need this API key value for the api_key argument of the pandas-datareader.

Next, I will use matplotlib to explore the movement of prices of a selected tech stock over the year. This would give me a high-level picture of how the stock has been performing for the last 10 years. I will then split the data to train and test set and start to build my models.

Solution Statement:
My solution to this problem would involve multiple sub-tasks below:
- Using matplotlib and plotly libraries to plot and understand the data
- Handling and understanding financial time series data:

    - Calculate and plot short term + long term Simple Moving Average for technical analysis of stocks.

- Using Deep Neural Networks to predict the stock price:
    - Instantiate a sequential model
    - Define the hidden layers and output layer
    - Complies the sequential model object for classification
    - Use Walk Forward Validation to provide the most realistic test harness to evaluate the models. This will result in more models being trained and could provide a more accurate estimate of the model performance on the unseen stock price.
    - Normalizes the features by Gaussian normalization
    - Fit the model to the training data
    - Do inference then transform the prediction into buy (+1) or short (-1) position
    - Calculate the strategy returns given the positions
- Plot and compare the strategy performance to the benchmark performance.

Benchmark Model:
A good benchmark model for this problem would be a backtesting strategy. I would normalize the momentum strategy by taking the mean log return over the last 15, 30, 60, and 120-minute bars to derive the position in the stock. If this value is positive, I would stay long (buy) the stock; if it is negative, I would stay short (sell) the stock.

Next, I would derive the absolute performance for different momentum intervals, create a plot and inspect it.

Evaluation Metrics:
Mean Absolute Error (MAE) is simply the average of the absolute difference between the target value and predicted value. However, MAE does not penalize large errors and there may be outliers on the stock price, so I decided to use Root Mean Squared Error (RMSE) which penalizes larger errors. RMSE is the root of the average of squared residuals.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

Project Design:
A theoretical workflow for approaching a solution to this problem would involve the steps below:

- Retrieve financial data from the data sources
- Using NumPy and Pandas to process trading data and explore vectorization for financial analytics: data analysis, processing data, data loading and storage, data cleaning and preparation, analyzing data, data visualization, etc.
- Using a deep neural network to generate market predictions
- Backtesting algorithms