



Stock Trading with Machine Learning

Daniel Hunegnaw , Kai Luo | EE P 596 Advanced Machine Learning | 02-20-22

UW EE 596: Stock Trading with Machine Learning

Contents

Introduction.....	2
Project Structure.....	2
Folder structure	2
Results.....	4
Performance baseline	4
Performance EMA+ML	6
Performance STL+ML.....	8
Contribution	10

UW EE 596: Stock Trading with Machine Learning

Introduction

In this report, we will present the implementation of the baseline cross moving average strategy for stock trading.

Project Structure

The project is implemented in python using visual studio code. A private repository is created in github. We have planned to make the repository public after the project is completed. Fig 1 below show the current structure of the project. By the end of the project, it might be updated

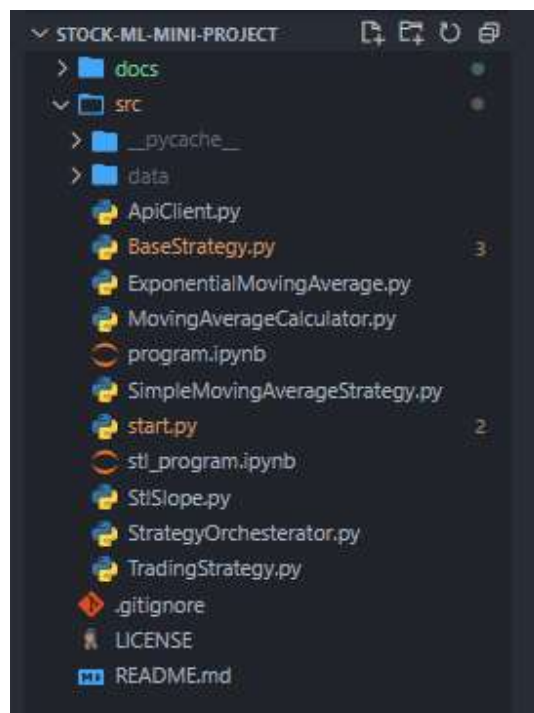


Fig 1. Project Folder Structure

Folder structure

`stock-ml-mini-project`: The root folder of the project

`docs`: where documentations and reports of the project reside

`src`: where the source code of the project resides

UW EE 596: Stock Trading with Machine Learning

`data`: is a folder where a csv file of stocks is saved. After the initial call to the ALPACA API, the data for each stock is cached into this folder to avoid hitting the API every time. If fresh data is needed, delete all the csv files in this folder

`ApiClient.py`: a utility class used to call the ALPACA API

`BaseStrategy.py`: a base class which currently contains three functions for:

`calculate_profit`: to calculate columns for profit for buy and hold, and for cross moving average strategy

`_plot`: private function for plotting dataframe data, moving averages, buy and sell signals

`_generate_signal_position`: for generating buy and sell signals based on moving averages


`MovingAverageCalculator.py`: contains functions for calculating Simple and Exponential Moving Average

`ExponentialMovingAverageStrategy.py`: contains for generating data for exponential moving average based strategy

`SimpleMovingAverageStrategy.py`: contains function for generating data for exponential moving average based strategy

`program.ipynb` : a Jupiter file for running the strategies (simple and moving averages) for the last 365 days for Stocks ("FB","MSFT","NFLX","AMD","GOOG"). To run the project using the `program.ipynb`, you need to provide values for `Api_key` and `secret_key` variables.

`start.py` : this does the same task as in `program.ipynb` but you can use this to run it from the command line using the command as in the fig 2



```
PS D:\UW\EE 596\stock-ml-mini-project> python .\src\start.py
```

fig 2. Running the project from command line

UW EE 596: Stock Trading with Machine Learning

Results

The following are results of the execution of the Exponential Moving Average based strategy.

Performance Metrics

1) crossing Exponential moving average without machine learning

To measure the performance of the crossing Exponential moving average, a buy and hold strategy for the same period used.

Log returns (in %) of the buy-and-hold strategy for the last 365 days was calculated and log returns of the crossing exponential moving average strategy was calculated.

Based on the log returns, the crossing exponential moving average gives better performance (at least reduces the loss) 80% of the time among the 5 stocks used for testing. However, for one stock [AMD] (36.5% vs 20.7%) as shown in Fig 6, the performance is lower. For one stock (NFLX), the loss was not avoided but greatly reduced in comparison to the buy-and-hold strategy (-20.8% vs -6.9%)



Fig 3 Facebook (FB), the last 365 days baseline EMA performance

UW EE 596: Stock Trading with Machine Learning

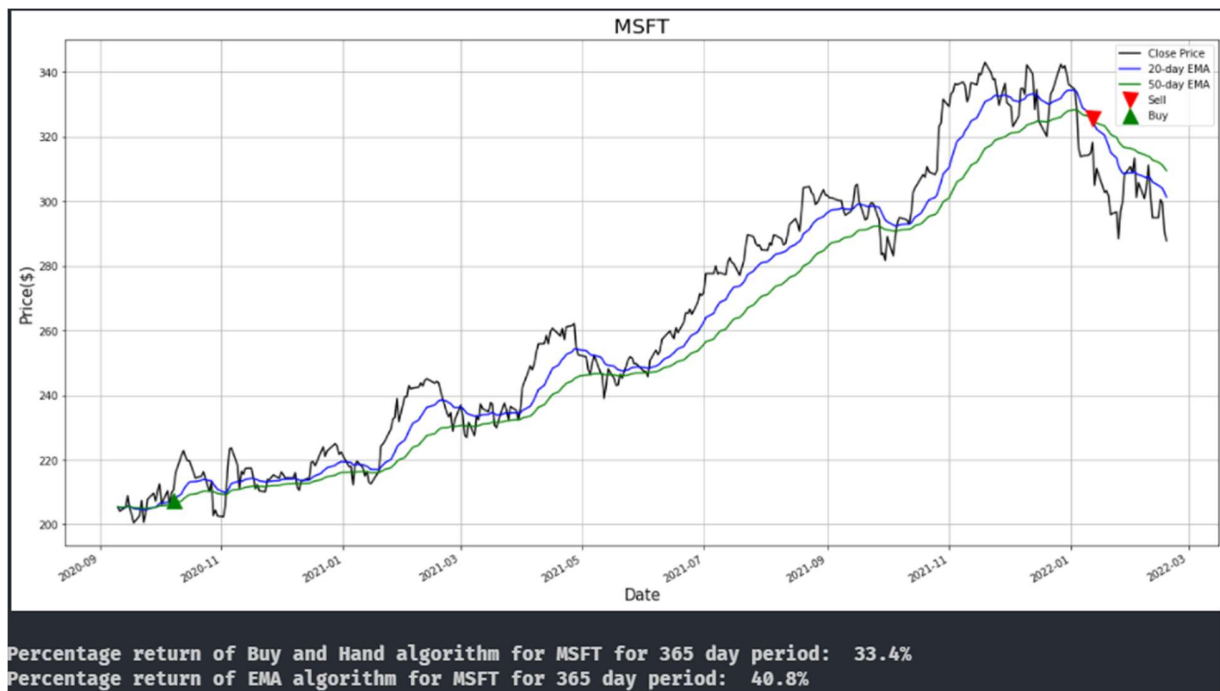


Fig 4 Microsoft (MSFT), the last 365 days baseline EMA performance

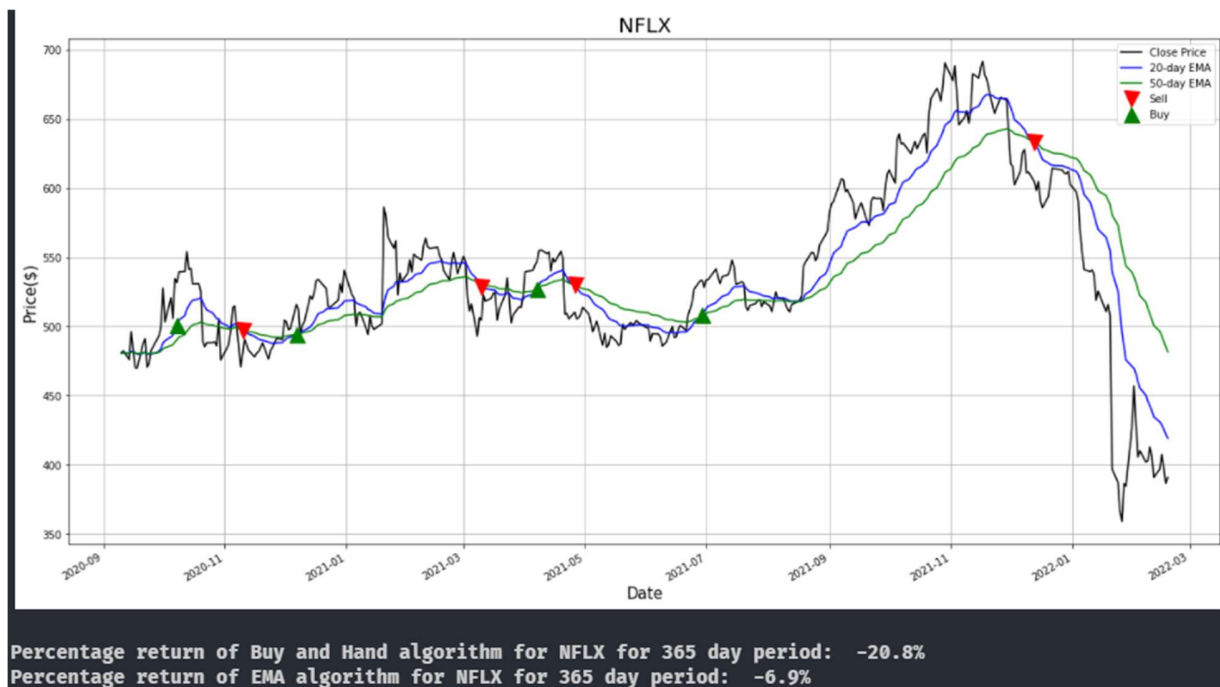


Fig 5 Netflix (NFLX), the last 365 days baseline EMA performance

UW EE 596: Stock Trading with Machine Learning

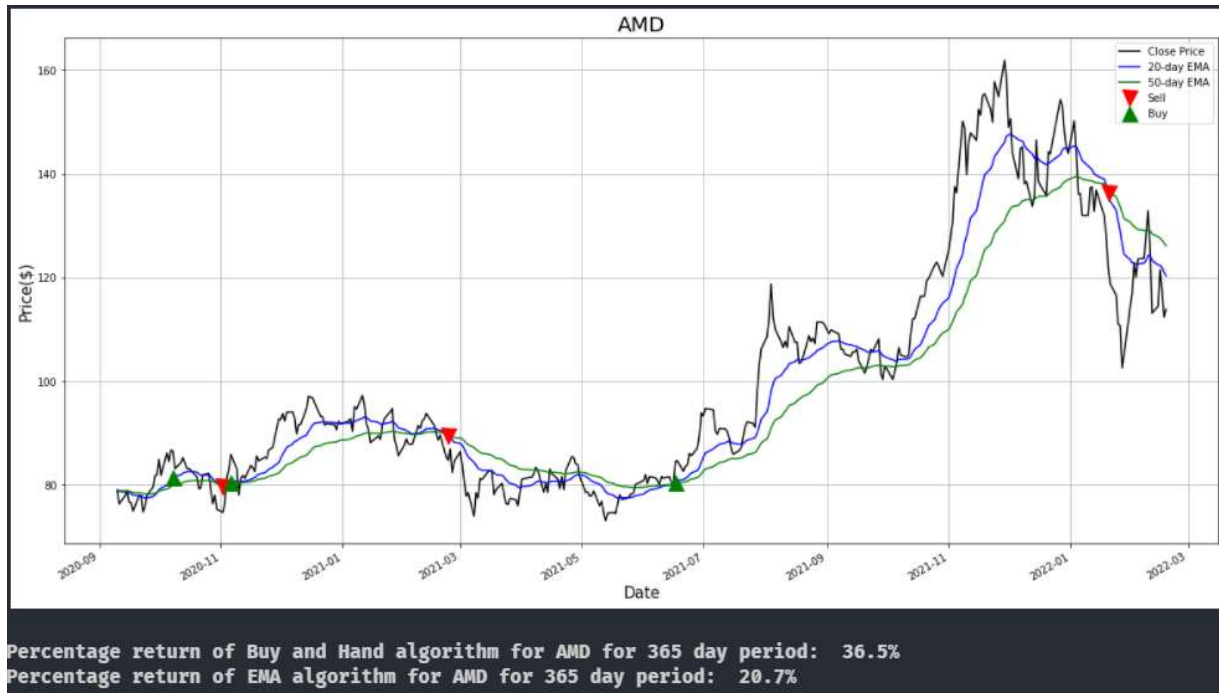


Fig 6 AMD (AMD), the last 365 days baseline EMA performance



Fig 7 Google (GOOG), the last 365 days baseline EMA performance

UW EE 596: Stock Trading with Machine Learning

2) machine learning model Using Exponential moving average

First, we using EMA to find all of sharp edge mark as buy and sell point. Fill 0 in between as holding point. The result list will be labels for supervised learning model.

Second, using all kind the prices for the day as features. Scaling inputs to the range -1 and 1.

Third, split the data set to train and test (backtesting) at 7 and 5. Then doing training and predict with MLP.

Multi-layer perceptron (MLP) is a supervised learning algorithm of learning function. It can learn nonlinear function approximators for classification or regression. It differs from logistic regression in that there can be one or more nonlinear layers, called hidden layers, between the input layer and the output layer.

The result is no very good, this strategy misses highest selling point. There still have a lot of things need to adjust like collect features and change model.

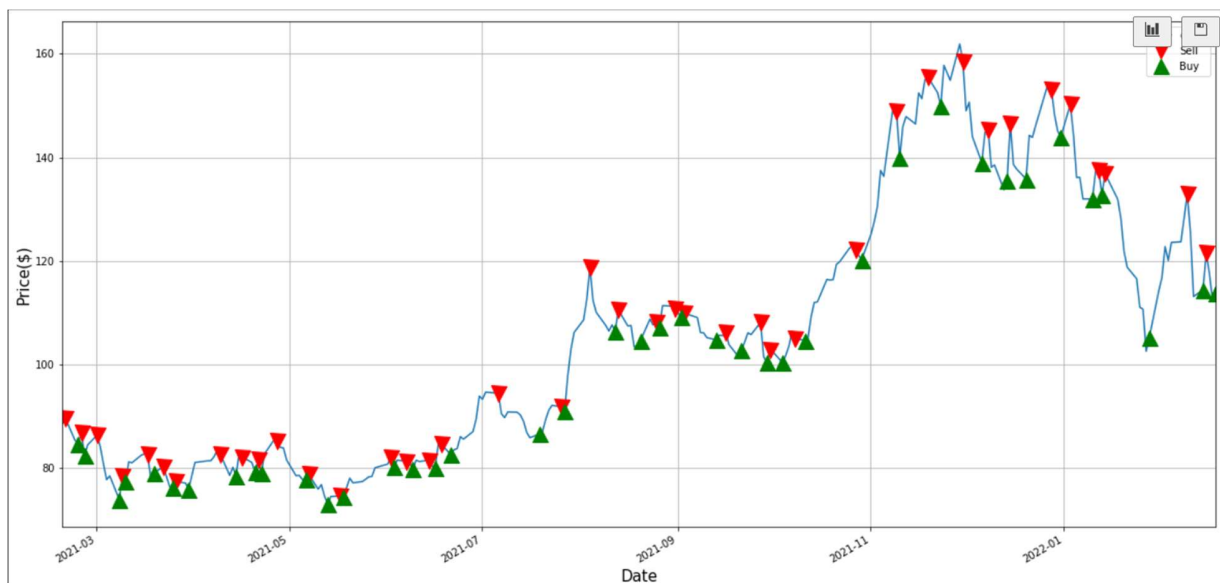


Fig 8 AMD, the collection of buy/sell points

```
Principal: $ 100000
Profit: $ 19038
That's 1.1903849789018022 %
```

Fig 9 Profit

UW EE 596: Stock Trading with Machine Learning

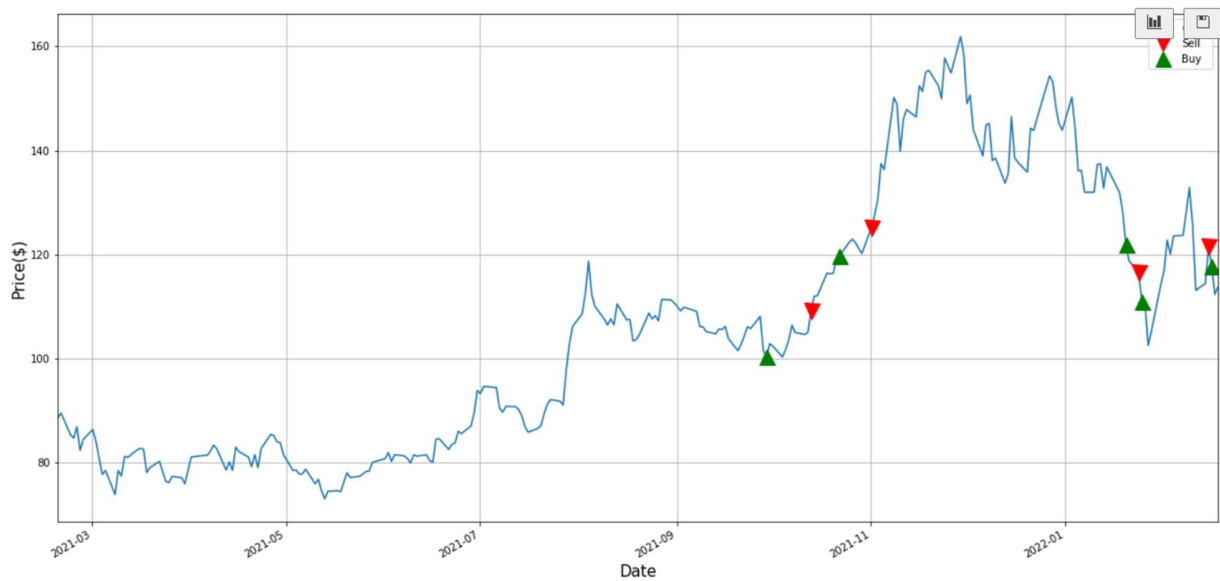


Fig 10 Strategy prediction

3) machine learning model Using STL slope

First, we use the slope of residual from STL to find all of buy and sell point. Fill 0 in between as holding point. The result list will be labels for supervised learning model.

Second, using all features from STL for all the prices for the last day as features. Scaling inputs to the range -1 and 1.

Third, split the data set to train and test (backtesting) at 7 and 5. Then doing training and predict with linear SVC(SVM).

The result is better than MLP, but still not as good as baseline. This strategy misses lowest selling point. There still have a lot of things need to adjust like collect features and change model.

UW EE 596: Stock Trading with Machine Learning

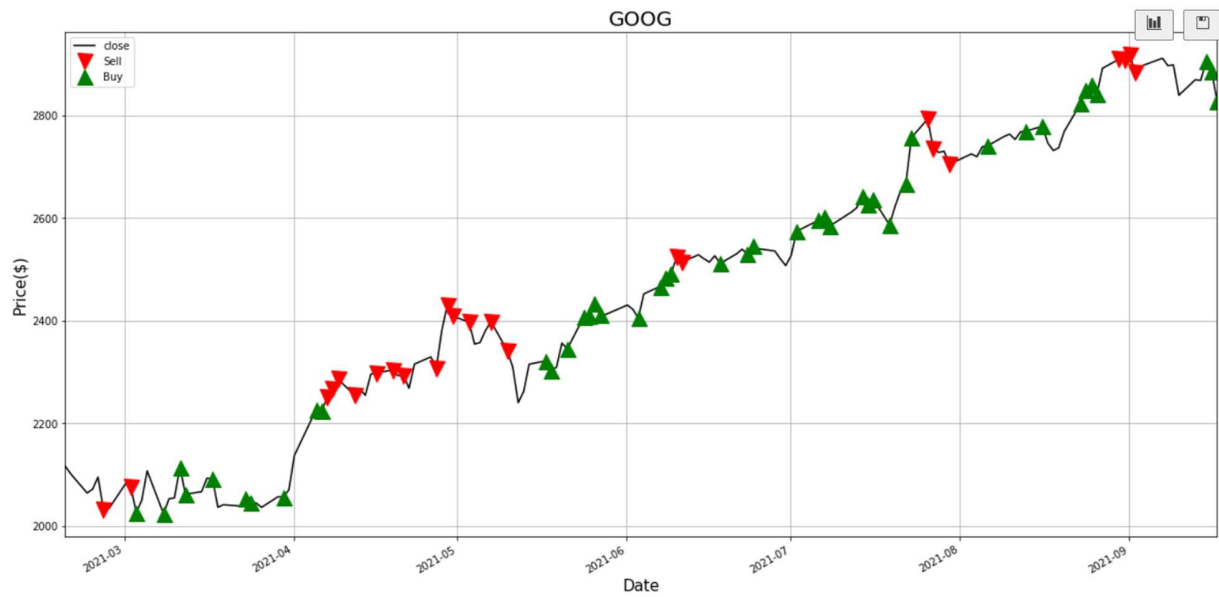


Fig 11 GOOG, the collection of buy/sell points

```
Principal: $ 100000  
Profit: $ 8063  
That's 1.0806303608812513 %
```

Fig 12 Profit

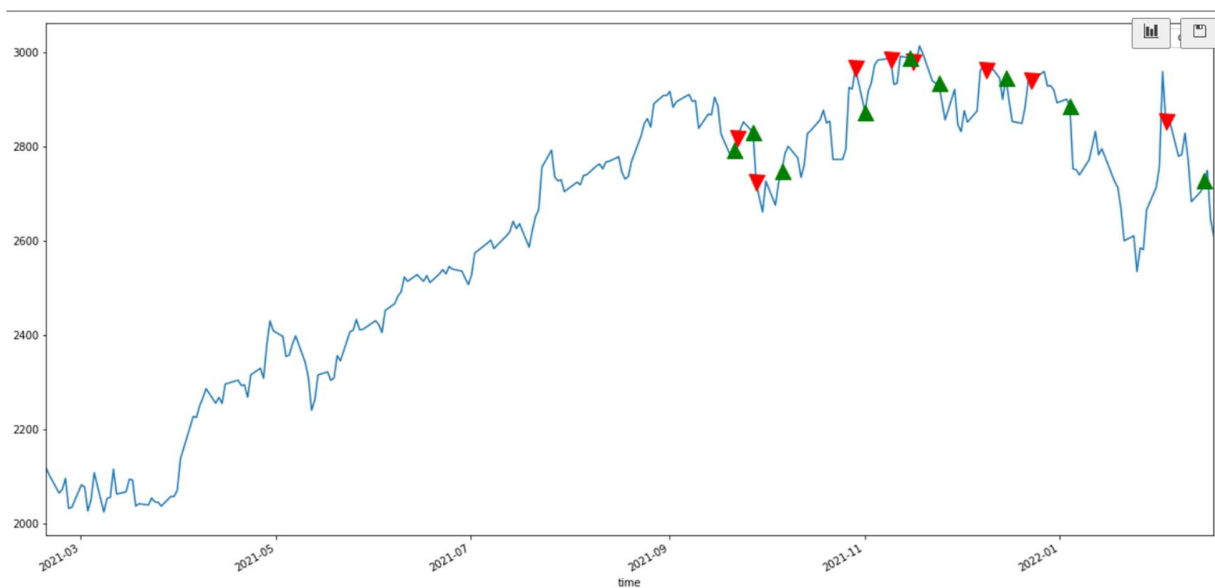


Fig 13 Strategy prediction

UW EE 596: Stock Trading with Machine Learning

Contribution of each team member :	Daniel Hunegnaw	50%
	Kai Luo	50%