CS446 Project Progress Report[[1]](#footnote-2)\*

|  |
| --- |
| **Machine Learning Based MACD Divergence Trading Strategy** |
|  |
|  |
| Qiasheng(Huck) Zou, Yiwei Zhuang |
|  |
|  |
|  |
|  |
|  |

Introduction

The goal of this project is to build an automated-trading system that outputs buying, selling or holding signal for each stock included in the current S&P500 index on each trading day. Our trading strategy will be based on a technical trading pattern called MACD divergence. A MACD divergence has two kinds: bullish and bearish. If a MACD diagram shows a bullish pattern, we would predict the stock price will go up and a bearish pattern would mean the stock price will go down. Since the decision process for a human trader who uses this strategy involves two basic steps:

1. recognize there is a MACD divergence pattern occurring,

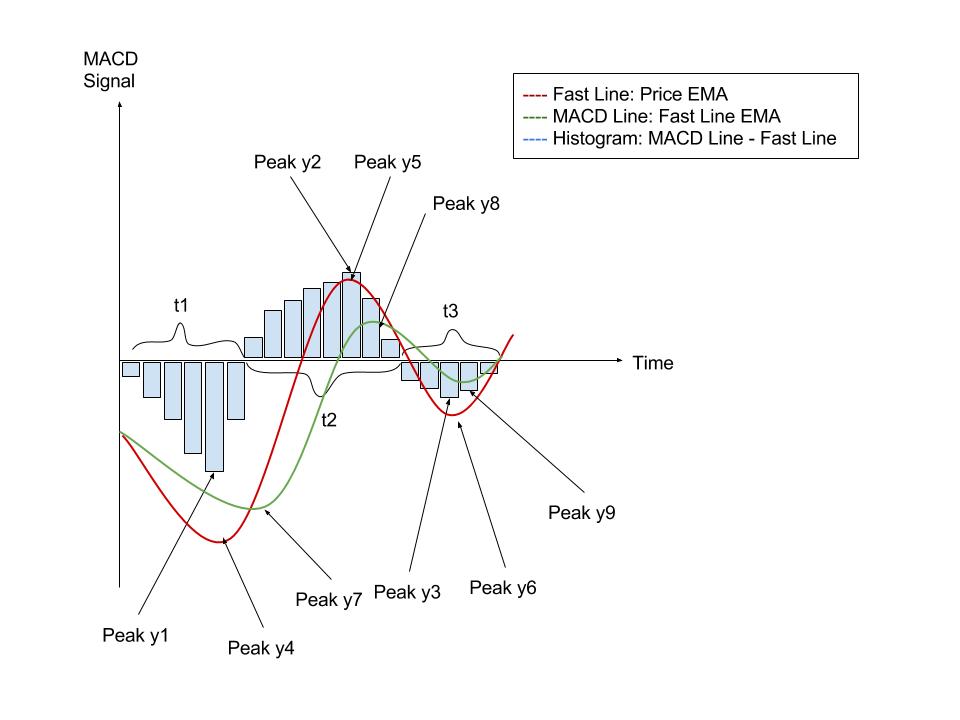
2. combine other information such as price levels, trading volume, price exponential moving averages and so on,

we decided to simulate a human trader’s decision process and train two classifiers corresponding to each of the task.

For this intermediate project report, we will focus on discussing the approaches that we have taken to train a classifier that recognizes a MACD divergence pattern, which is the first task involved in a human trader’s decision process.

To start, we will first give a more formal definition of our machine learning problem. Then, we will discuss our raw market data collection process and data labeling process. After that, we will discuss the machine learning algorithms we have experimented and present the best result we have obtained. Lastly, we will briefly discuss what we will work on in the future.

Problem Definition

For our work up to now, we have been focusing on recognizing a MACD pattern like the one shown below.

The diagram above shows a MACD bullish pattern. If we flip the bullish pattern about x-axis, then we will get a bearish pattern. Note for each MACD pattern, we divide the pattern into three phases based on the sign of the histograms. For example, in the pattern shown above, phase 1 is negative, phase 2 is positive and phase 3 is negative.

A list of features we are currently using (notations are labeled in the diagram above) to identify the pattern is:

|  |  |  |
| --- | --- | --- |
|  | Feature | Value |
| 1 | Sign of the first phase | Negative: 0,  Positive: 1 |
| 2 | y1/y2 | decimal |
| 3 | y1/y3 | decimal |
| 4 | y2/y3 | decimal |
| 5 | y4/y5 | decimal |
| 6 | y4/y6 | decimal |
| 7 | y5/y6 | decimal |
| 8 | y7/y8 | decimal |
| 9 | y7/y9 | decimal |
| 10 | y8/y9 | decimal |
| 11 | t1 | integer |
| 12 | t2 | integer |
| 13 | t3 | integer |

Based on the pattern, here is a list of labels for all data samples:

|  |  |
| --- | --- |
| Label | Value |
| Neutral | 0 |
| Bullish | 1 |
| Bearish | 2 |

With the features and labels defined above, we would like to train a classifier that is able to correctly classify a given MACD pattern into one of the three types: neutral, bullish, and bearish.

Data Collection Process

With Yahoo Finance API, we retrieved the daily stock price data from Jan. 1ST 2000 to Aug. 20TH 2016, for all stocks included in the current S&P500 Index and stored them in a MySQL database. We then cleaned the data by backward filling[[2]](#footnote-3) the missing or invalid values. To make our training data more representative, we picked a few frequently traded stocks from different industries:

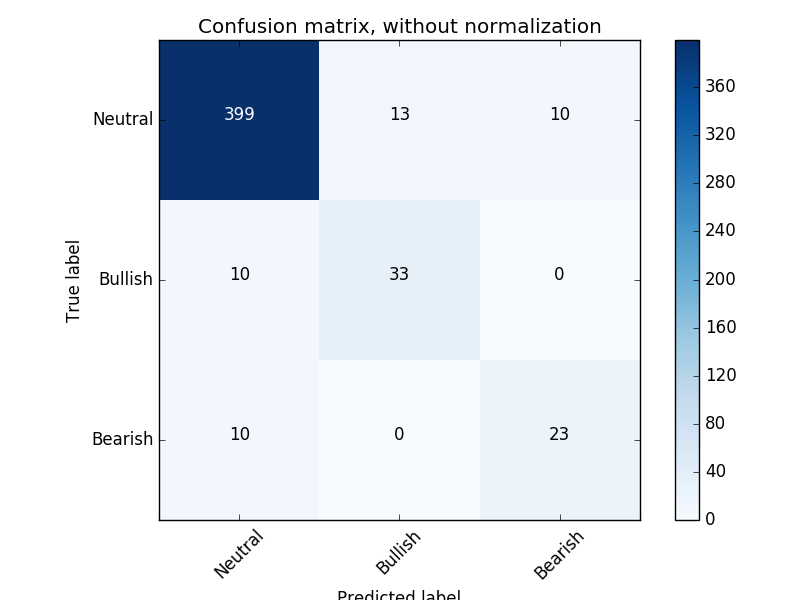
|  |  |  |
| --- | --- | --- |
| Industry | Company Name | Stock Symbol |
| Tech | Apple | AAPL |
|  | Facebook | FB |
| E-commerce | Amazon | AMZN |
| Food | Chipotle | CMG |
|  | Coca-cola | KO |
| Clothing | Under Armor | UA |
|  | Nordstorm | JWN |
| Auto | Ford | F |
| Banking | JP Morgan | JPM |

After getting the raw daily price data for the stocks selected above, we then calculated the MACD indicator for each stock. To generate training samples, we divided the entire MACD indicator time series into smaller pieces that consist of three phases[[3]](#footnote-4). Therefore, each stock could generate hundreds of three-phase pieces. After that, we plotted a graph for each piece like the one shown at the beginning. These pieces were our training samples. To label them, we came up with a set of labeling instructions (please see the attachment at the end) and labeled each sample according to the rules. We had three people to label the data. To test if the labeling was consistent, we ran Kappa tests among the labeling participants and obtained Kappa scores above 0.8 for each pair of participants. Therefore, we concluded that our labeling was consistent and good for training a classifier.

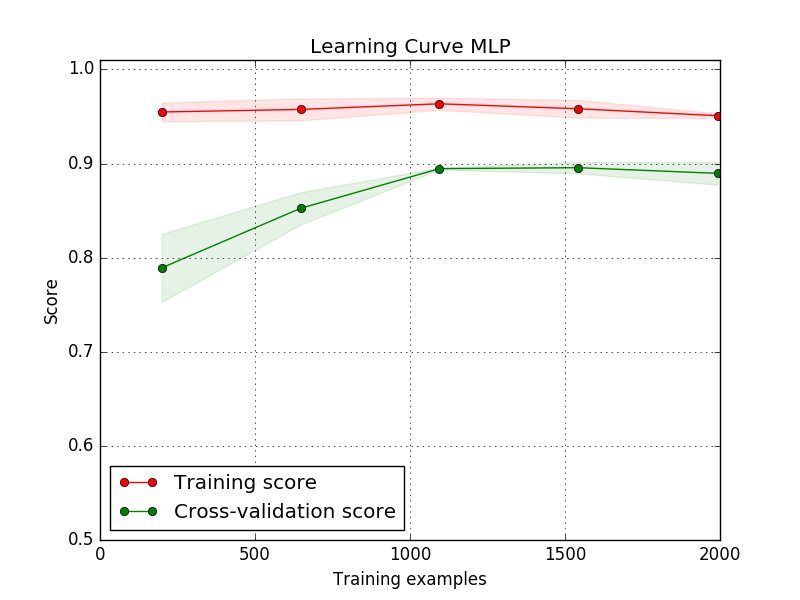
Training Methods and Results

We trained our samples with a multilayer perceptron package in Python. We ran a neural network algorithm with different parameter settings for 10 rounds and calculated the average accuracy at the end. For each round, we randomly selected 80% of the samples as training data and 20% of the samples as testing data. After the model selection process, we obtained the following optimal set of parameters that yield the highest overall accuracy.

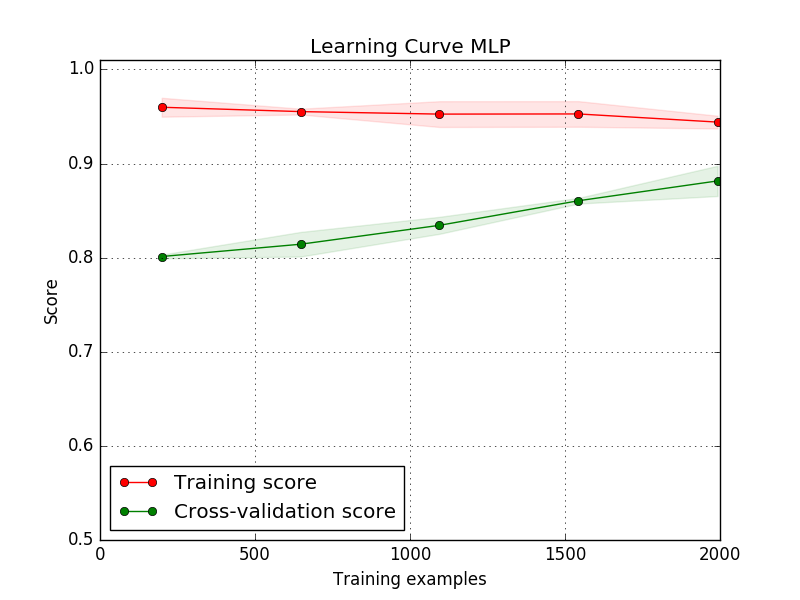
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Training  Objective | Solver | Activation | Hidden, Layers | Accuracy |
| Highest  Overall Acc. | lbfgs | logistic | (10,) | 89.6% |

To see more clearly the classification accuracy for each label, we also plotted the confusion matrix.

We can see that since 85% of our samples are labeled as zero, a trivial classifier that outputs zero for all samples would obtain an accuracy of 85%. Therefore, the performance of the classifier we have trained is not particularly impressive. Nevertheless, from the confusion matrix above, we can also see that our classifier actually never predicts a bullish pattern as a bearish pattern and vice versa. It means that if we use the classification result as an indicator for buying and selling stocks, we will not short stocks when bullish pattern occurs and vice versa. Therefore, it is still very possible to build a profitable trading algorithm with it.

To improve the accuracy further, we tried to test whether or not more samples would help us with our classification accuracy. We plotted a learning curve graph for our neural network model.

From the graph we can see that, our neural network model stops improving after 1000 training samples. Therefore, it is not possible to improve the accuracy with more samples in this case.

Then we thought of a more complex neural network model might be able to help because with more hidden layers, our model will be more expressive. We ran a new model with the same configuration as our previous model except that there are two hidden layers with 10 and 5 neurons respectively.   
From the graph we can see that this new model actually shows some potential for improvements with more samples. Therefore, we might consider to label more samples in the future and train a more complex model to obtain better accuracy.

Future Work

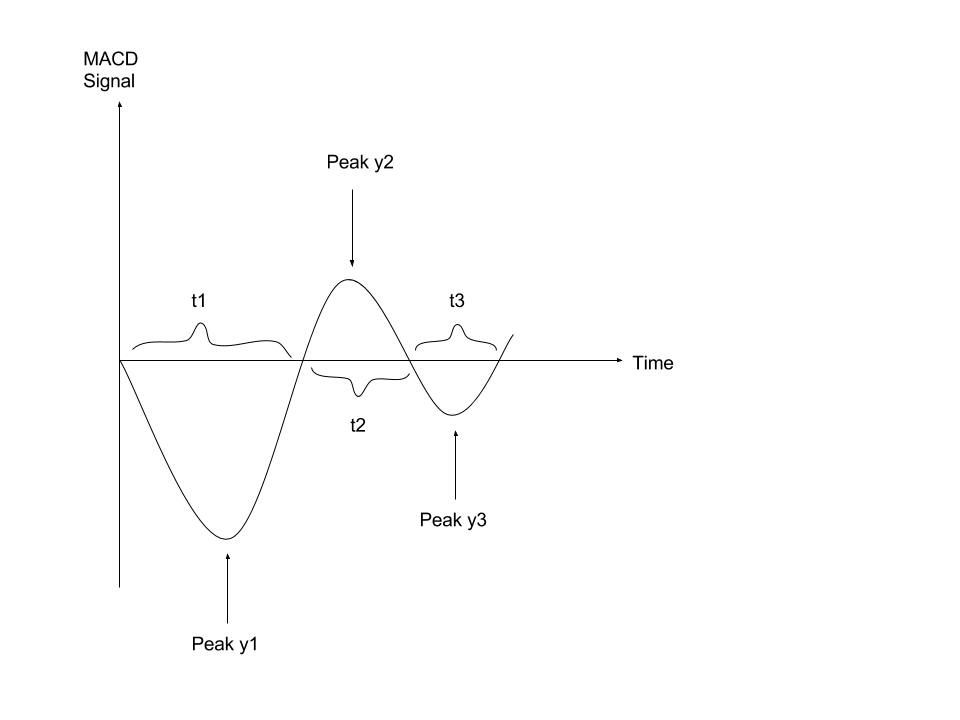
Since the pattern of MACD divergence does not occur frequently for each stock. Eighty-five percent of the samples are labeled as neutral. In our future work, we will add in more labels that are 1s and 2s in the samples.

Up to now, we have completed our first step of building the automated trading system. Our next step is to combine the outputs from the classifier we have obtained with some additional features, such as price levels, volumes, different moving averages, and so on, to train another neural network model that gives us buying, selling, or holding signals. That work will be included in our final report.

Also, we will try other learning algorithms such as decision trees and SVMs to see whether we can obtain better performance.

Appendix A.

Labeling Instructions (short version):



1. For all three lines: histogram, EMA line, and signal line, we are looking for the patterns like the one shown above.
2. To label the data as 1:
   1. |y1| > |y2|. That is |y1/y2| the bigger the better
   2. |y1| > |y3|. That is |y1/y3| the bigger the better
   3. t1+t2+t3 should not be too small.
   4. t1 the longer the better.
   5. t2 should be from 10 to 30
   6. t3 the smaller the better.
3. To label the data as 2: simply flip the diagram about x-axis and apply the same rule as for labeling as 1.
4. All other patterns that don’t belong to the description above should be labeled as 0.

1. \* This report is only a progress report based on what has accomplished so far, and does not contain the full implantation of the trading strategy. [↑](#footnote-ref-2)
2. Backward filling means whenever there is an invalid value in a given time series, we will use the next valid value in the time series to replace the invalid value. [↑](#footnote-ref-3)
3. Each phase is a histogram as shown in the example diagram at the beginning. The diagram consists of three phases (three histograms). Note for each training sample, the three phases can be either +-+ or -+-. [↑](#footnote-ref-4)