Stock price movement prediction and strategy on 2016 market

a "showcase" project

Yiteng Tian
yiteng.tian@gmail.com

This is a mini project to model the stock price open-to-close direction within one trading day. I applied machine learning technique to build the predictive model using historical OHLC (open, high, low, close) data from Yahoo Finance for nine ETFs: XLE, XLU, XLK, XLB, XLP, XLY, XLI, XLV, and SPY. And by this predictive model, I am able to develop the Buy/Sell strategy to apply on 2016's market and test its profitability.

Source code (in python) is available at https://github.com/tianyiteng/predictive-modeling

Model Construction and Training:

This predictive model is constructed by three sub-models. To start the research on the movement of one day's stock price, I used the features from: a. self historic performance; b. periodic oscillation of the market; c. cross correlation of other ETFs' data, to build three sub-models. The respective techniques I have used to build these classifiers are summarized in Fig. 1.

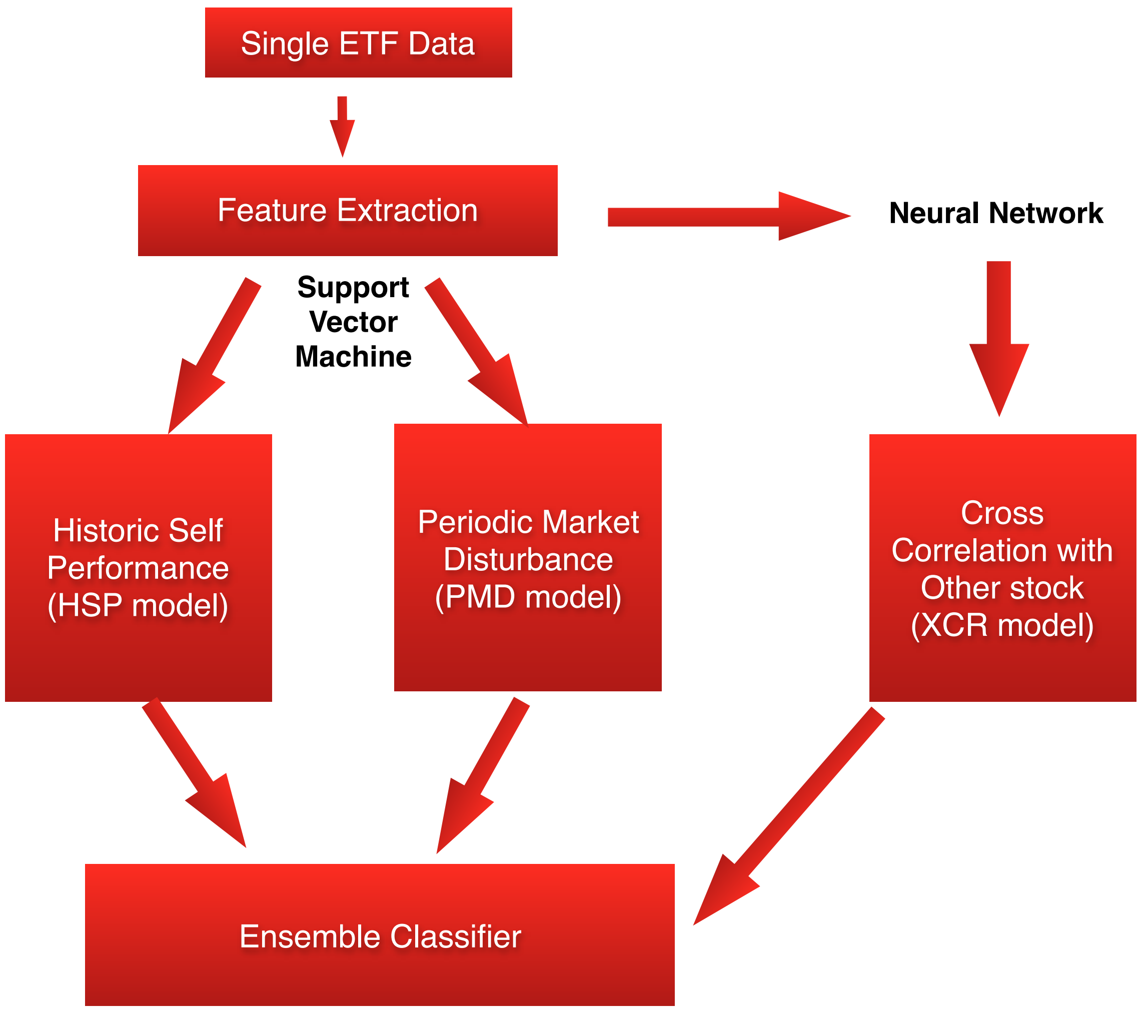
Single ETF Data
Feature Extraction
Neural Network
Support Vector Machine
Historic Self Performance (HSP model)
Periodic Market Disturbance (PMD model)
Cross Correlation with Other stock (XCR model)
Ensemble Classifier

Fig. 1

The historic self performance (call it HSP model) is the most weighted in this predictive model. To train this classifier, I used the following features:

1. the last two day's open, high, low, close, volume data, normalized by the average of recent 30 market days. These features are floating numbers within (0~1)
2. the five day's movement of close data. Use the 3-bits binary int to describe the movement. For example recent movement is (down, down, up, up, down), denoted by 00110 (bin), 7 (dec).

For in-sample objects, I've calculated the historical Buy/Sell signal as 1 and 0 class. And I've used the rising/declining percent (0.5 percent as level increment) as the weights to indicate the strong or weak Buy/Sell signal.

After extracting these features and transform them into machine recognizable data, apply the machine learning technique of Support Vector Machine (SVM) to do fitting and predicting. Due to its none linearity, I've used the radial basis function (RBF) as my kernel function. Splitting the data into 9:1 sub dataset for training and testing purpose, I was able to find out the proper kernel coefficient to avoid over fitting. Since at this step, the model only care about for single ETF, we can repeat the same process to all 9 ETFs' data.

The periodic market disturbance (call it PMD model) takes the seasonal performance of the market in to consideration. To train this classifier, I used the following features:

1. The movement from opening to closing, for same day last week (same weekday).
2. The movement of stock close price for consecutive 20 market days back to 63 market days (3 months, one season) ago. In other words, if taking today's date as day 0, the price movement from day -73 to day -53 is our first feature. To denote this movement, I used the same method mentioned in the last model.
3. The movement of stock close price for consecutive 20 market days back to 252 market days (1 year) ago. In other words, if take today's date as day 0, the price movement from day -262 to day -242 is our second feature.

Since the periodic market movement is usually directly related to the stock price movement, I still chose to use SVM to build classifier for this sub-model. Similar to avoid over-fitting, I split the data into 9:1 sub dataset for training and testing purpose, to find out the proper number of neighbors. And also use the rising/declining percent (0.5 percent as level increment) as the weights to indicate the strong or weak Buy/Sell signal.

The last sub-model cross correlation of other ETFs' data (call it XCR model) takes account of the influence of other stocks, since these ETFs are all sector-level and market-level within S&P500. To train this classifier, I did the following steps:

1. For every ETFs, get the features included in the first two sub-models, and build a Python dictionary.
2. Use the DictVectrizer class from sklearn library, to transform the lists of feature-value mappings to vectors.

Due to the input's high degree of freedom, I used neural network model Multi-layer Perceptron classifier, a supervised neural network model from scikit-learn toolbox, to build classifier for this sub-model.

After training these three models, I combine them linearly to build the overall full Three-Step Classifier, and generate the strategy for 2016 market. The performance are described in Evaluation section.

Evaluation

After training the classifier and build the three-step model, I tested my strategy on 2016 market. The performance can be show in Fig. 2. It describe the cumulative daily portfolio P&L's versus time through out 2016. At the end of 2016, the cumulative P&L for SPY-Long-Only strategy is 1.36 dollars and for ALL-Long-Only strategy is 1.65 dollars. The strategy using the previous model gives 2.30 dollars.

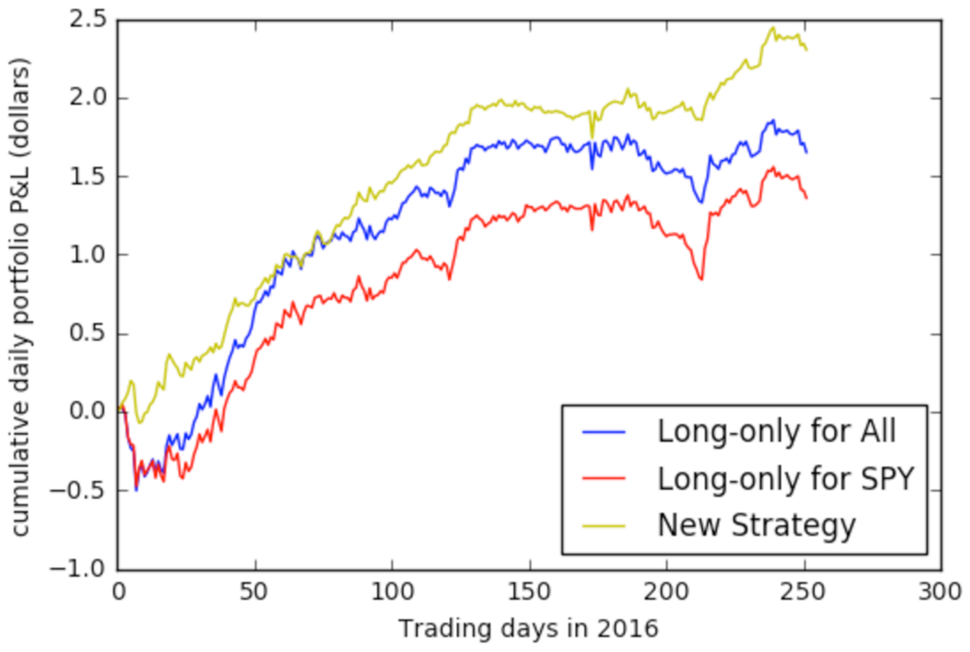

Fig. 2

The daily performance and annualized sharpe ratio is shown in Fig. 3

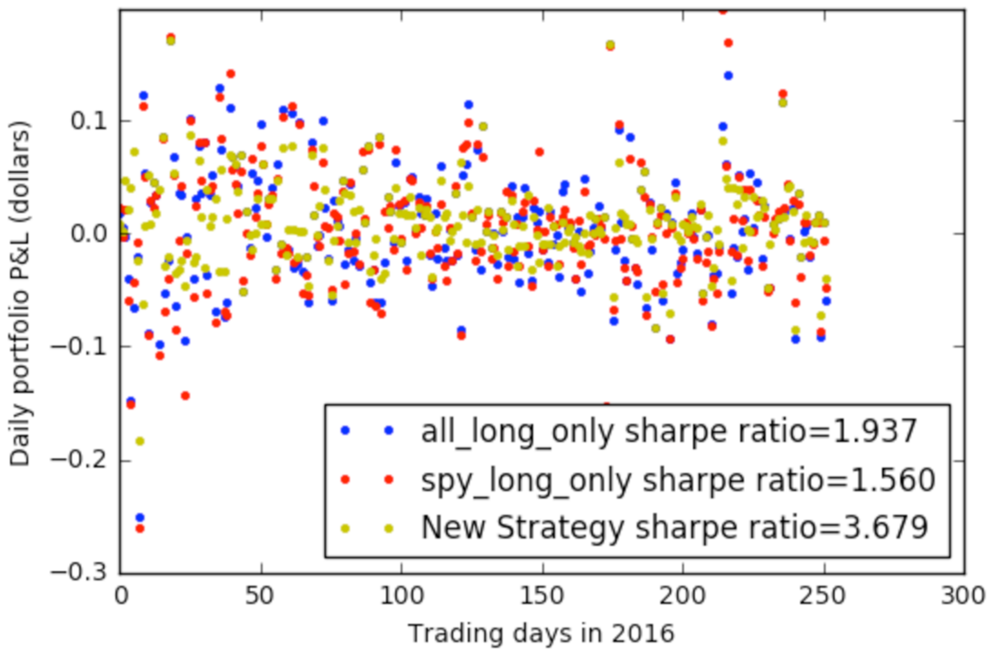

Fig. 3