Machine Learning Applied on Portfolio Optimization

Eric Wang

October 4, 2022

1 Introduction

This project is built with python and all of my codes can be accessed through my github. In this project, instead of using global Mean-Variance Optimization(MVO) strategy with a fixed window for re-allocating assets, I use a more flexible strategy to estimate expected return, covariance matrix and determine when to adjust the portfolio weights.

I try several unsupervised learning models to analyze the market regimes, and I find that K-means clustering with technical indicators outperforms the other models. To minimize the risk of my portfolio, I use all S&P 500 component stocks as my stock pool. However, a very large dataset is required to estimate the expected return and covariance matrix of all 500 stocks, thus I use Support Vector Machine(SVM) to create a subset of stocks for investing before re-allocating my portfolio. I also use a shrinkage estimator for covariance matrix to make sure that it's positive definite such that optimization problem can be solved. At the end, I choose an proper optimization strategy to build my portfolio, results of backtesting shows that this strategy is practicable.

2 Methodology Discussion

2.1 Data Collection

I use yfinance to download all the data of stocks in S&P 500 [1]. The SPDR S&P 500 trust(SPY) was launched on 1993, by downloading all stocks data after 1993 and dropping stocks in which there're more than 5% undefined numbers, I use 426 stocks for analysis and backtesting.

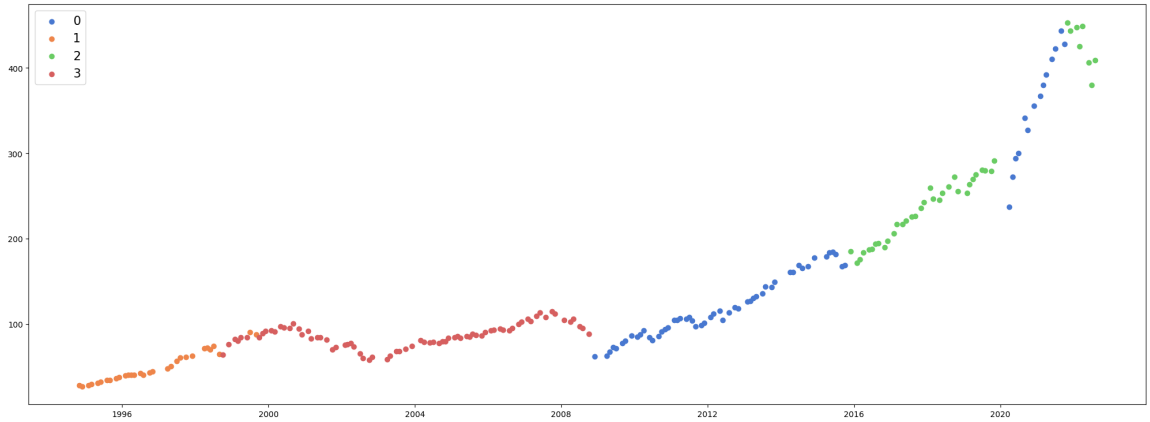
I use Alpha Vantage to download data of economic indicators. And I use following indicators in my analysis: Federal Funds Rate, Consumer Price Index (CPI), Inflation Expectation, Consumer Sentiment and Unemployment.

For technical indicators of stocks, most of them, including Simple Moving Average (SMA), Standard Deviation, Exponential Moving Average (EMA), High minus Low and Close minus Open, are calculated by pandas. Indicators about momentum such as Relative Strength Index (RSI), parabolic SAR (SAR), Average Directional Movement index (ADX) are calculated via TA-lib.

2.2 Market Regime Detection

Upon the materials we talk about on class, a natural idea to use a Hidden Markov Model(HMM) with economic indicators to detect the regime of market. However, I try to change the number of states and shrink some inputs, the result always makes no sense as it groups by time and not the market regimes, for example, see figure 1. I think there're two reasons, the dataset is relatively small because most economic indicators are reported monthly, and economic are slow to react compared with the equity market change.

[1] We can easily use `read_html()` in pandas to get all tickers in S&P 500



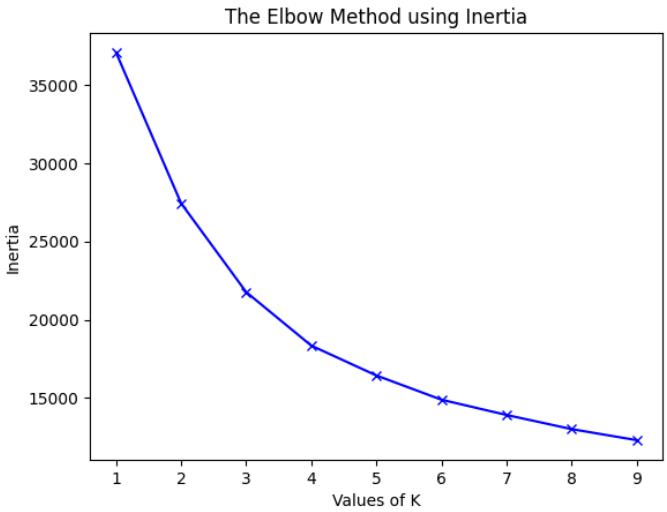
Ffigwr 1: Using HMM to detect Market Regime

Cont (2000) proposes that high-volatility events of stocks tend to cluster in time. And according to an article from Two Sigma, I use GMM to estimate the market regime. The idea is that returns of financial assets do not always follow a normal distribution, GMM fits various Gaussian distributions to capture different parts of the asset's return distribution, and each of those distributions would have its own properties. I use returns of SPY and its moving average to fit GMM and get best result assuming there're 4 underlying clusters, see figure 2. Although the figure gives a appropriate classification, it's hard to explain green points(state 2) after year 2020.

Ffigwr 2: Using GMM to detect Market Regime

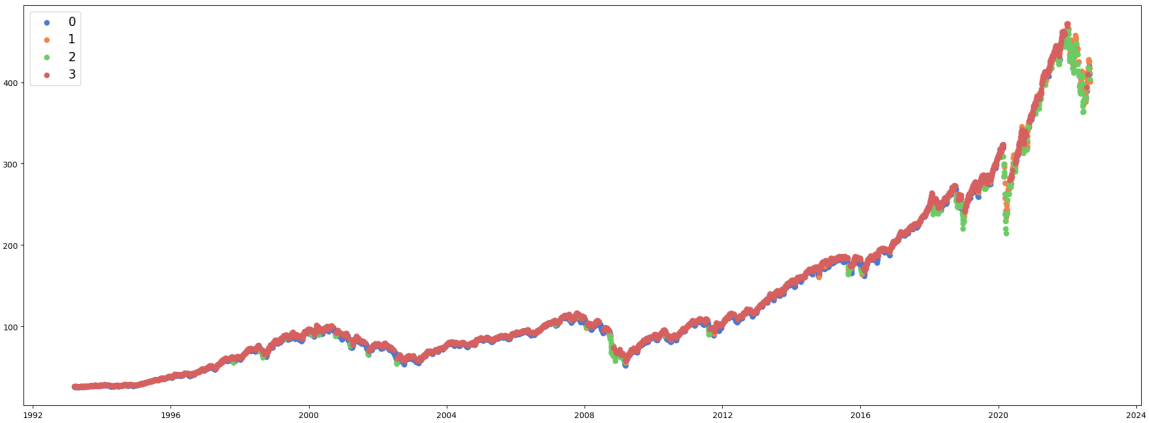
Since we know that high-volatility events of stocks tend to cluster in time, another way is to use K-means clustering to classify market regimes. In my test, using SMA, standard deviation,

High minus Low, Close minus Open and RSI gives us best result. In each test, I use Elbow plot to determine the number of clusters K, then use K to fit the model. Take my final result as an example, to determine the number of clusters(K), I first create a Elbow plot, see figure 3.

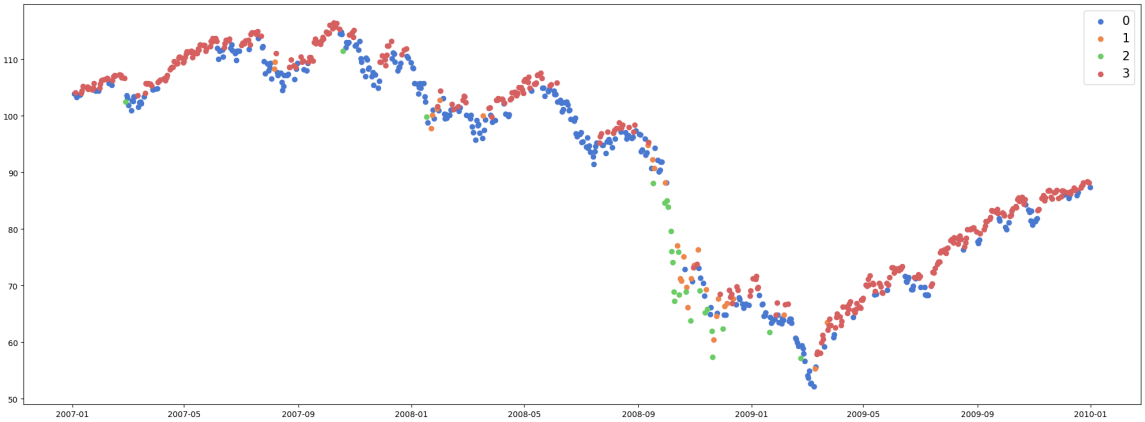


Ffigwr 3: Elbow plot of K-means

From the figure, we choose to use 4 clusters to fit K-means model, and K-means fits the data perfectly, see figure 4. Figure 5 show a closer result of financial crisis, we find that there's always huge jump around green points (state 2).

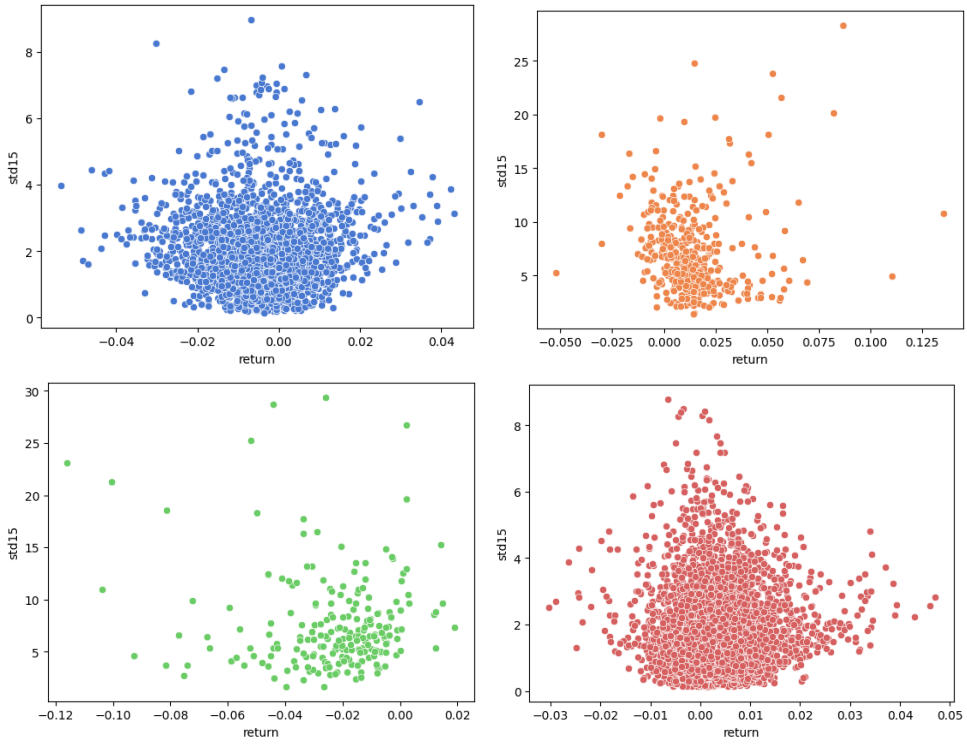


Ffigwr 4: K-means clustering result



Ffigwr 5: K-means clustering result during Crisis

Figure 6 is a figure of 4 states' returns versus standard deviation. It shows that K-means shows four states, where blue points and red points have low standard deviation, orange points and green points have relatively higher standard deviation. And by analyzing their returns, we can conclude that yellow points and green points suggest bull and bear market. Red and blue ones suggest that we are at calm market, but if it's red, the market is more possible to go up. Table 1 verifies our analysis, in which mean and std are calculated by returns of SPY.



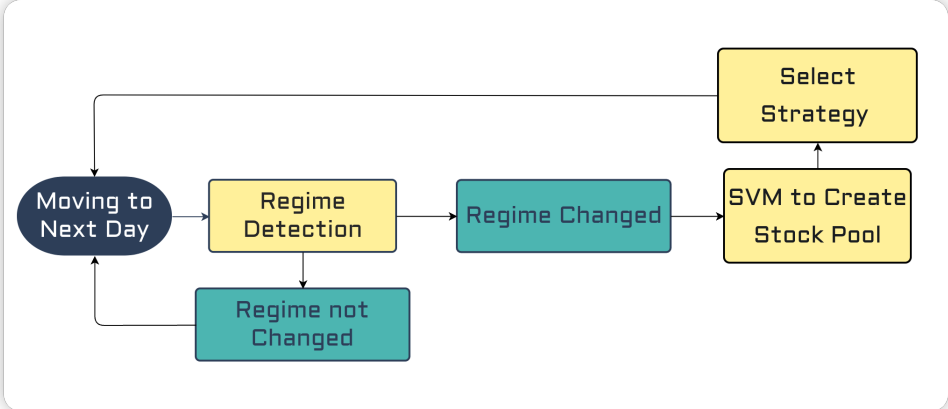
Ffigwr 6: 4 regimes

2.3 SVM on Stock Selection

To shrink the number of stocks we invest in, I simply use all the technical indicators to fit the model. The input is all the indicators and output is to determine if the SMA(30 days) two weeks later will be larger than toady's SMA. I test 50 stocks and the prediction accuracy is around 85%. It's not very high but it's enough for me to get a smaller stock pool.

3 Backtesting

In this section, I will do my backtesting based on the research of Chapter 2. The process of my backtesting is first implementing K-means to detect our current regime, if regime changed, I use SVM to test each stock and select stocks worth investing into my stock pool, then choose an appropriate strategy based on current regime, see the flow chart.



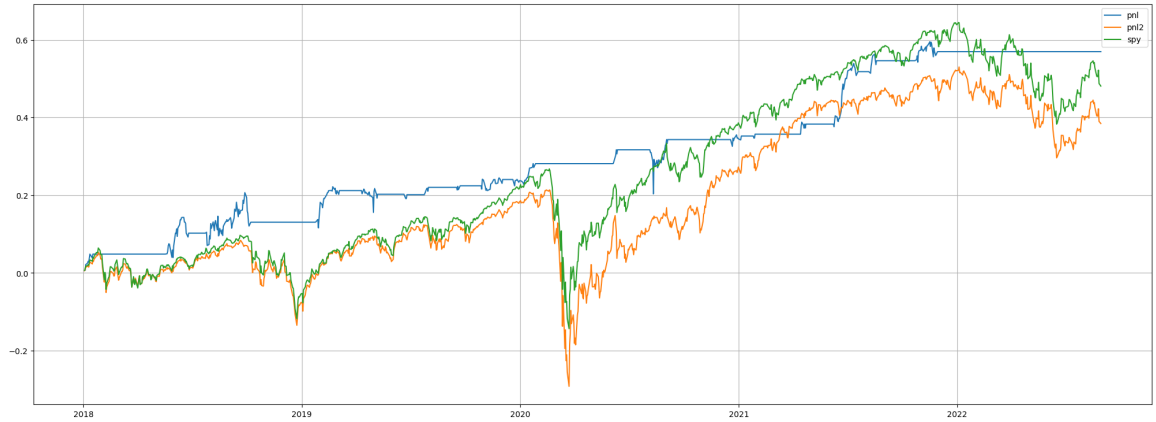
Ffigwr 7: Flow Chart

In my implementation, I use PyPortfolioOpt built by Martin (2021) to estimate expected return and covariance matrix with constant correlation shrinkage method. A simple trading strategy is used in my backtesting:

1. If there's a bear market detected within past 2 weeks, I just empty my portfolio.
2. If there's a clam market detected but with a negative return, I use mean-variance optimization with a relatively high risk-averse parameter.

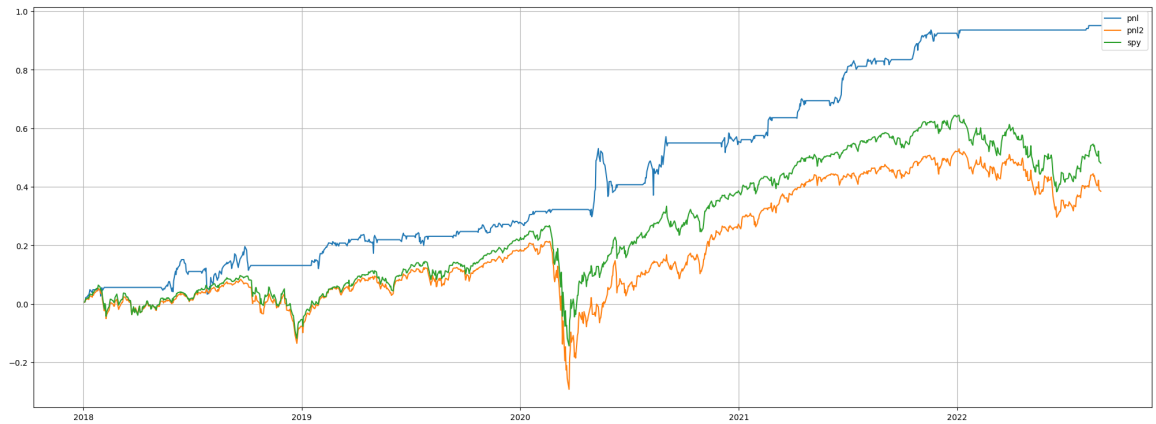
3. In other cases, I use maximize the sharpe ratio given sum of all weights equals to 1.

I set the transaction fee to be 1%, results are shown in figure 8. Blue line is my portfolio, green is SPY and orange is equal-weighted portfolio. It doesn't outperform SPY a lot, but we notice that when SPY drops a lot, P&L of our portfolio doesn't change because we sell all the stocks in advance.



Ffigwr 8: Backtesing Result

Indeed, if I relax some constraints of my trading strategy, for example, I only empty my portfolio when there's a bear market detected in 10 days. The performance of my portfolio will be much more better, which benefits from K-means clustering gives me a good result of detecting market regime, see figure 9. It not only performs better than SPY in bull or calm markets but also avoids huge drops during bear markets.



Ffigwr 9: Backtesing Result 2

4 Conclusion

In my project, I try several ways to detect the regimes of market and find that K-means produces a best result. Based on my market regime model, I built my optimization strategies, results show that they are feasible and efficient.

At the end, there're many future works to do:

- In the beginning, I want to build a factor model using auto-encoders as Gu et al. (2021) proposed in his article. However, it requires me a lot of prerequisite knowledge about asset pricing to replicate and modify his method so I give up this idea. And I believe it's interesting to be implemented and the covariance matrix estimation would be more accurate.
- The SVM method to determine whether to invest a stock is superficial, and there're many other methods to be investigated. Once I find a more efficient method, the performance of my backtesting result will be better.
- Besides, the trading strategy I used in my project can also be improved to get a better result. I think it's also possible to use a deep learning method here, for TD learning or policy gradient.

5 Appendix

State	Count	Mean	Std
0	2428.0	-0.004335	1.946686
1	394	0.013764	7.281018
2	218	-0.023388	7.635582
3	4379	0.002953	1.759820

Table 1: Summary of 4 states

References

Cont, R. (2000). Empirical properties of asset returns: stylized facts and statistical issues. *QUANTITATIVE FINANCE*, 14.

Gu, S., B. Kelly, and D. Xiu (2021, May). Autoencoder asset pricing models. *Journal of Econometrics 222*(1), 429–450.

Martin, R. A. (2021). Pyportfolioopt: portfolio optimization in python. *Journal of Open Source Software 6*(61), 3066.