Using Topological Data Analysis to Create Systematic Equity Strategies

Aaron Kreiner
Center for Data Science
New York University
ak6817@nyu.edu

Travis Luo
NYU Shanghai
New York University
113794@nyu.edu

Abstract

We use topological data analysis (TDA) to create a systematic investment strategy from 2015-2021. We source close prices from around 8000 stocks from the NASDAQ and NYSE exchange to generate portfolios every month from 2005-2021. We find a 90x return during this period using our framework, which beats the SP500 and other models.

1 Introduction

The equity market consists of thousands of stocks that deliver gains or losses to an investor depending on market conditions. Systematic equity investing is the process of using signals on market related data to predict individual stock returns. These signals, depending on strength, are used to create a portfolio of stocks for a given time period. For our project, we consider several thousand equities and their associated time series for adjusted close prices. We feed these close prices into a topological indicator which gives a prediction of whether the individual stock price will substantially fall or rise at a given date. For each date, we long the stocks with the highest rise probability and short the stocks with the highest fall probability. Our final product would be a backtested systematic strategy that shows how much money we would have made if we had used the topological indicator to make investments. This problem has important relevance. Equity price data is free, meaning that if we find strong returns under our strategy anyone can use our model to generate sustainable investments. We could potentially open up world class strategies to those who can not afford to pay large data fees. Second, very few people have found strong returns from using price alone. Due to the highly non-linear nature of the indicator, we see strong potential on the individual stock level to do quite well. In this sense, we are proposing something novel. Third, no one in the literature has tried to apply this indicator on the single stock level, which is another facet that makes our research unique. For these reasons, our project is both novel and highly relevant.

2 Related Work

In this section, we present a short synopsis of the relevant works for our project. There are thousands of published academic papers that look at systematic equity strategies. We therefore only look at papers that apply topological methods to predict stock movements. Nugroho et al calibrates a topological framework to predict severe market crashes. They find that a topological framework vastly outperforms a simple baseline on the large index level. Their main result is shown below Prabowo (2021).

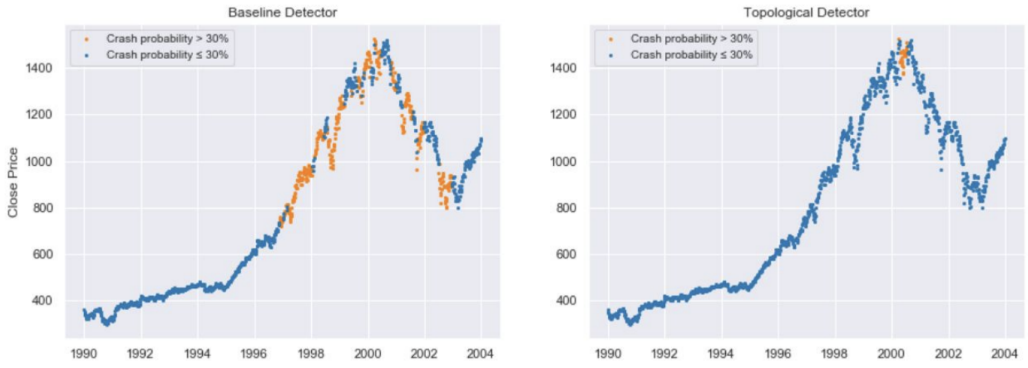

Figure 1: The orange points represent where the crash probabilities are high for the SP500. We see a topological indicator is able to find the point where a crash happens better than a simple baseline.

Most articles only focus on the application of topological data analysis (TDA) methods to global indexes, such as the SP500. We extend the literature by creating a framework to apply it to the single stock level with superior results. The next section describes our theoretical algorithm used to calculate the probabilities, which will be used to decide which stock to buy at a given time.

3 Problem Definition and Algorithm

3.1 Task

In this section, we describe how we find the probability of financial crisis using time series analysis methods.

First, we start with applying Kalman Filter to the stock price in our selection. According to Bjork (2009), the stock price follows a Wiener process with drift, it is a dynamic process

$$X_t - X_0 = \mu \cdot t + \sigma \cdot (W_t - W_0) \tag{1}$$

$$X_t = X_0 + \mu t + \sigma W_t \tag{2}$$

where W is a Wiener process.

Under the influence of many factors, e.g. the psychology of the market, we believe that the market close price follows a kind of stochastic process with random noise.

$$X_t = X_0 + \mu t + \sigma W_t + w_t \tag{3}$$

where w_t is independent white noise

Then Kalman filters are ideal for dynamic systems which are continuously changing. It can reduce the noise of the data we observed from the stock market and make predictions on its actual value.

Following Shumway and Stoffer (2005), We have Latent variable:

$$z_n = Az_{n-1} + w \tag{4}$$

and Observed variable:

$$x_n = Cz_n + v \tag{5}$$

where the Gaussian noise terms are

$$w \sim N(0, \Gamma), \quad v \sim N(0, \Sigma), \quad z_0 \sim N(\mu_0, \Gamma_0)$$

Kalman filter will first do forecasting, which is prediction given latent space parameters

$$z_n^{pred} \sim N(\mu_n^{pred}, V_n^{pred}) \tag{6}$$

$$\mu_n^{pred} = A\mu_{n-1} \tag{7}$$

$$V_n^{pred} = AV_{n-1}A^T + \Gamma \tag{8}$$

then it will do innovation, which is correction from observation

$$\mu_n^{innov} = \mu_n^{pred} + K_n(x_n - C\mu_n^{pred}) \tag{9}$$

$$V_n^{innov} = (I - K_n C)V_n^{pred} \tag{10}$$

Kalman gain matrix:

$$K_n = V_n^{pred} C^T (CV_n^{pred} C^T + \Sigma)^{-1} \tag{11}$$

Filtering is a forward process, which gives us predictions on the actual data. Here we will also apply the backward method, smoothing, which will give us the final estimates on latent variables with lower variance.

$$\mu_n^{smooth} = \mu_n^{innov} + J_n(\mu_{n+1}^{smooth} - A\mu_n^{innov}) \tag{12}$$

$$V_n^{smooth} = V_n^{innov} + J_n(V_{n+1}^{smooth} - V_{n+1}^{pred})J_n^T \tag{13}$$

$$J_N = V_n^{innov} A^T (V_{n+1}^{pred})^{-1} \tag{14}$$

After we have all the smoothing data, we regard this as the actual price of our selected stocks and make it as the input data to the Topological Data Analysis framework.

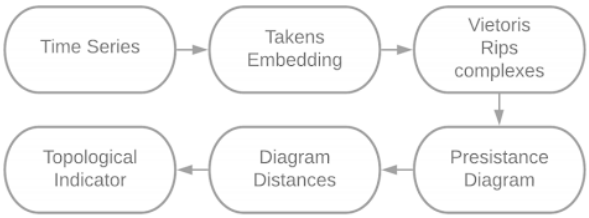

Figure 2: TDA pipeline

We first construct the Takens' embedding, the parameters we choose here are embedding dimension d, and time delay τ. For each time $t_i \in (t_0, t_1, \dots)$, we collect the values of the data x at d distinct times, evenly spaced by τ and starting at t_i, and present them as a vector with d entries, namely:

$$X_{t_i} = (x_{t_i}, x_{t_i+\tau}, ..., x_{t_i+(d-1)\tau}) \tag{15}$$

Then choose a window size w, apply the sliding window procedure to the full time series, that will give us a time series of point clouds, each one is a sliding window, namely:

$$W_t = (X_{t_i}, X_{t_i+1}, ..., X_{t_i+w-1}) \tag{16}$$

After generating a time series of point clouds, the next step is entering persistent homology, which looks for topological features. According to Gideaand and Katz (2018), all topological features that emerge from the data will be kept, and assigned 'weights' according to their persistence. For example, a feature will initially not be observed, then will appear, and after a range of values of the parameter will disappear again.

Before the procedure of computing persistent homology, we first need to the construct a filtration of simplicial complexes. Here for each distance $\epsilon > 0$, we define the Vietoris-Rips simplicial complex $R(X, \epsilon)$:

for each $k = 0, 1, 2, ...$, a k-simplex of vertices $\{x_{i_1}, ..., x_{i_k}\}$ is part of $R(X, \epsilon)$ if and only if the mutual distance between any pair of its vertices is less than ϵ, that is,

$$d(x_{i_j}, x_{i_l}) < \epsilon, \text{ for all } x_{i_j}, x_{i_l} \in \{x_{i_1}, ..., x_{i_k}\} \tag{17}$$

$R(X, \epsilon)$ form a filtration: $R(X, \epsilon) \subseteq R(X, \epsilon')$, whenever $\epsilon < \epsilon'$. Then for each complex we can compute the k-dimensional homology $H_k(R(X, \epsilon))$. The property of $R(X, \epsilon)$ ensures for the homology that $H_k(R(X, \epsilon)) \subseteq H_k(R(X, \epsilon'))$ (for $\epsilon < \epsilon'$).

After having all properties above, we can encode all the information in a persistence diagram P_k. According to Gideaand and Katz (2018), a persistence diagram consists of:

- For each k-dimensional homology class α one assigns a point $z_\alpha = (b_\alpha, d_\alpha) \in \mathbb{R}^2$ together with its multiplicity $\mu_\alpha(b_\alpha, d_\alpha)$;
- In addition, P_k contains all points in the positive diagonal of $\mathbb{R}^2$. These points represent all trivial homology generators that are born and instantly die at every level; each point on the diagonal has infinite multiplicity.

Given two windows and their corresponding persistence diagrams, we can calculate a variety of distance metrics. When considering a metric, according to Berwald et al. (2018), there are two closely related options: the bottleneck distance, and the p-th Wasserstein distance $(p1)$. Here, we choose the p^{th} Wasserstein distance as the metric for the persistence diagrams by considering the meaning of the points. Because the p^{th} Wasserstein distance measures the similarity between two persistence diagrams using the sum of all edges lengths (instead of the maximum). It allows to define sophisticated objects such as barycenters of a family of persistence diagrams. And it is defined as:

$$W_p = d_p(X, Y) = \inf_{\phi: X \to Y} (\sum_{a \in X} ||a - \phi(a)||_q^p)^{1/p} \tag{18}$$

However, if we continue to follow Gideaand and Katz (2018), we will know that, such metric space (P, W_p) is not complete. In order to study time series of persistence diagrams with statistical tools, we have to modify the structure of the space of persistence diagrams based on persistence landscapes. We first define a piecewise linear function:

$$f_{(b_\alpha, d_\alpha)} = \begin{cases} x - b_\alpha, & \text{if } x \in (b_\alpha, \frac{b_\alpha + d_\alpha}{2}] \\ -x + d_\alpha, & \text{if } x \in (\frac{b_\alpha + d_\alpha}{2}, d_\alpha) \\ 0, & else \end{cases} \tag{19}$$

To a persistence diagram P_k, we associate a sequence of functions $\lambda = (\lambda_k)_{k \in \mathbb{N}}$, where $\lambda_k : \mathbb{R} \to [0, 1]$:

$$\lambda_k(x) = k - max\{f_{(b_\alpha, d_\alpha)}(x) | (b_\alpha, d_\alpha) \in P_k\} \tag{20}$$

where the k-max denotes the k-th largest value of a function. $\lambda_k(x) = 0$ if the k-th largest value does not exist. And we can see this persistence landscapes actually form a subset of the Banach space $L^p(\mathbb{N} \times \mathbb{R})$.

We will use these persistence landscape distances as our topological features. In order to make it meaningful in a statistical sense, we transform the distance into probability by simply normalizing a time series of distance values. Thus, the probability can be obtained by:

$$prob = \frac{\mu_{roll} - \min \lambda(n)_{roll}}{\max \lambda(n)_{roll} - \min \lambda(n)_{roll}} \tag{21}$$

where μ_{roll} is a rolling mean value on the landscape distances, and $\max \lambda(n)_{roll}, \min \lambda(n)_{roll}$ are rolling max or min value of the landscape distances.

3.2 Algorithm

All of the stock price data we're going to use are downloaded from Yahoo Finance and are reformatted properly before all the jobs begin.

After finishing a universe selection by picking a minimum of 1000 Stocks for each month that match the SP500 return out of all the stocks that are from NYSE/NASDAQ during that month. (the detail of this algorithm will be discussed in **4.1**)

We first run the Kalman filter and smoothing to the data we have for each month to get the "no-noise data". The aim of this step is trying to make the stock price fit the dynamic process we defined in (2) as much as possible. So that, we can say the stock price follows

$$X_{t_i} - X_{t_j} \sim \mathcal{N}(\mu(t_i - t_j), \sigma^2(t_i - t_j)),$$
at least for t_i, t_j close enough (e.g. one month)

the signal we found using TDA would be more robust and consistent.

Once we have the data after preprocess, they will be used as the input of our TDA probability function. Following the pipeline of the TDA method we've mentioned above, we can get the probability for each stock in each month.

We summarize these steps in Algorithm 1, 2.

Algorithm 1: An algorithm to perform Kalman filter

Data: raw stock price df_price
Assume: (i) each month has 30 days (ii) we have all the data before each time point
for *each month* t **do**
  for *each stock* s *in the Universe Selection* U **do**
    stock_price: $x\_t = df\_price.loc[s, :t]$;
    $model = KalmanFilter(n\_dim\_obs = 1, n\_dim\_state = 1)$;
    smoothed_price: $sx\_t = model.smooth(x\_t.reshape(-1, 1))$;
    use sx_t as an input to perform **Algorithm 2**: $prob = Algorithm2(sx\_t)$;
  end
end

Algorithm 2: A sub algorithm to perform TDA

Data: smoothed price sx_t obtained from **Algorithm 1**
Initialize: window_size w
Initialize: embedding_dimension d, embedding_time_delay τ;
(i) resample sx_t into $24H$ data $price\_resampled$ using pad() method for fill_na;
(ii) construct takens' embedding and sliding window:

$$embedder = ts.SingleTakensEmbedding(d, \tau), \tag{22}$$
$$price\_embedded = embedder.fit\_transform(price\_resampled), \tag{23}$$
$$sliding\_window = ts.SlidingWindow(size = window\_size), \tag{24}$$
$$price\_embedded\_windows = sliding\_window.fit\_transform(price\_embedded); \tag{25}$$

(iii) get persistence diagrams:

$$VR = hl.VietorisRipsPersistence(n\_jobs = -1) \tag{26}$$
$$diagrams = VR.fit\_transform(price\_embedded\_windows) \tag{27}$$

(iv) get landscape distances between persistent diagrams by using PairwiseDistance:

$$landscape\_hom\_der = HomologicalDerivative(PairwiseDistance) \tag{28}$$
$$landscape\_dists = landscape\_hom\_der.fit\_transform(diagrams) \tag{29}$$

(v) choose the probability as the topological indicator in this TDA, get a time series of probability by normalizing the distances we get from step **(iv)**:

$$probs = get\_probability(landscape\_dists) \tag{30}$$

(vi) return the latest probability of this stock $probs[t]$

Based on the probability we have for each stock in each month, we pick out top 10 stocks with highest probabilities to construct our portfolio and long them in the following month. We set up the back testing platform, and compare our cumulative portfolio return to the SP500 return.

4 Experimental Evaluation

4.1 Data

We plan to use the close prices for several thousand equities from 2005-2021 that span the NYSE and NASDAQ Exchange. All of the data is available through the Yahoo Finance API and is free. First, we

look at a plot of the number of stock tickers available every day. The gaps are around major holidays and weekends when there are inconsistencies in reporting.

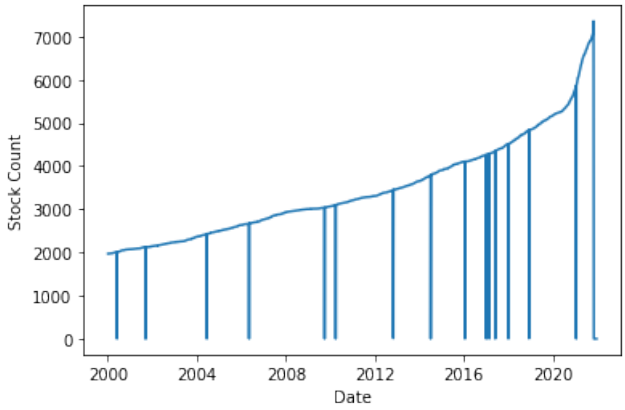

Figure 3: Number of Stocks Reporting Per Day

However, the stock universe is biases because yahoo finance drops de-listed tickers. Therefore, the stocks that remain suffer from survivorship bias. In other words, in prior periods we know these stocks will remain on the index, and will likely have positive returns. To correct for this bias, we implement an optimization method that makes the average return of the stock universe match the return of the SP500 every month. To make this concrete, suppose the SP500 has return spr_t in month t. Then let each stock i have return $r_{i,t}$. We then introduce a binary variable $u_{i,t}$ which determines if stock i is in the new stock universe at time t. The optimization objective function can then be represented as:

$$\min \text{spr}_t - \frac{\sum_i u_{i,t} * r_{i,t}}{\sum_i u_{i,t}} \tag{31}$$

A constraint is added so there are at least 1000 stocks in the universe. This ensures we have enough stocks to create a robust topological indicator. We then solve this optimization problem every month to determine the universe. The figure below shows the number of stocks that make the final universe every month.

Next, we check that the new universe perfectly matches the SP500 returns. We see in the plot below that the returns month over month are identical.

Thus, we have created an unbiased universe with respect to returns. We will now test in the following sections whether our stock picking strategy consistently outperforms the SP. This is a litmus test for whether our strategy is learning new information from the market.

4.2 Methodology

Stock price is the only data we are going to use in this project. Because we believe that, this time series data will have all the information we need for analysis in our case.

Because in our project, what we need to predict is the probability for a increase in the stock price for one stock in the following month. So we will use all the stock price data we have for a stock before time point t, which is the time point for turnover. We apply the same algorithm to all the stock in our selection for each month between 2005 and 2021. (Here we are using each 30 days for one month for convenience, instead of one natural month.) After that we choose the top 10 with highest probabilities to construct our portfolio.

We will evaluate our model performance by comparing the return of our portfolio with the return of SP500. Since the selection is set up to match the SP500 return, if the portfolio return has a significant improvement than the SP500 return, we can say that our model performs very well on picking out stocks that will increase in the following month.

In our Algorithm 2, the parameters we need to define are window size w, embedding_dimension d and embedding_time_delay τ. We set $w = 30$ which is equal to the time interval for turnover, so that we can have a consistent probability results. The larger the window size is chosen, the longer period there would be for the persistence diagrams and probabilities dependent, and also longer program run time. And we set embedding_dimension $d = 3$ and embedding_time_delay $\tau = 2$, based on the parameter selection in Gideaand and Katz (2018).

4.3 Results

First, we show Apple's transformed TDA probabilities from 2018-2021. We use this chart to show some of the fundamentals of our trading strategy. Because we are picking the stocks with the highest probabilities, we only care about the behavior of the probability estimates when they are above 0.3 or 0.4 (when they would rank as one of the highest probabilities in the universe).

Figure 6: TDA probability before and after Kalman filter

Now, we move to showing our returns results. We find that under our algorithm, we generate consistently good returns. The first plot shows a histogram of monthly returns for the SP500. The second plot shows a histogram of monthly returns for our TDA strategy. Finally, the third plot shows our backtested returns from 2005-2021.

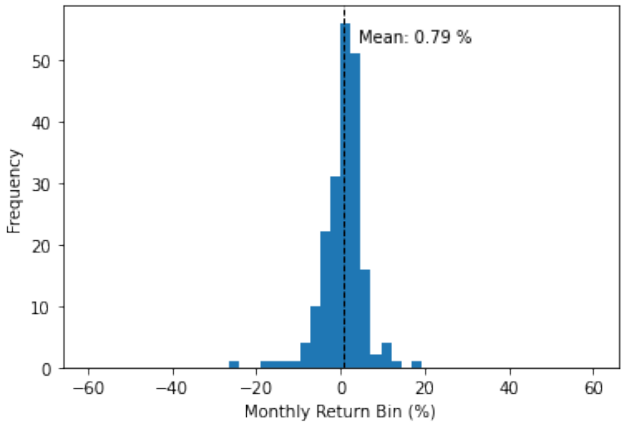

Figure 7: Histogram of SP500 Returns

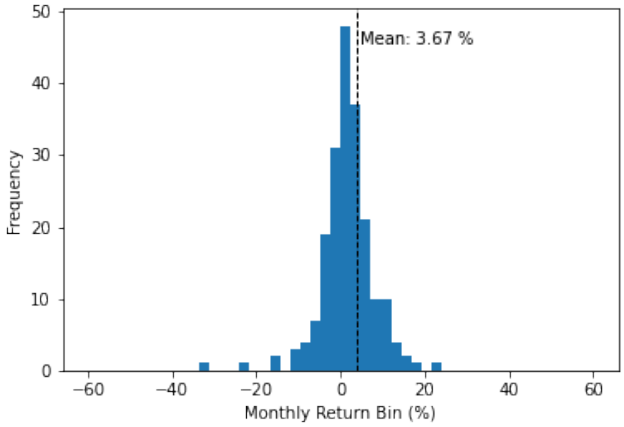

Figure 8: Histogram of TDA Portfolio Returns

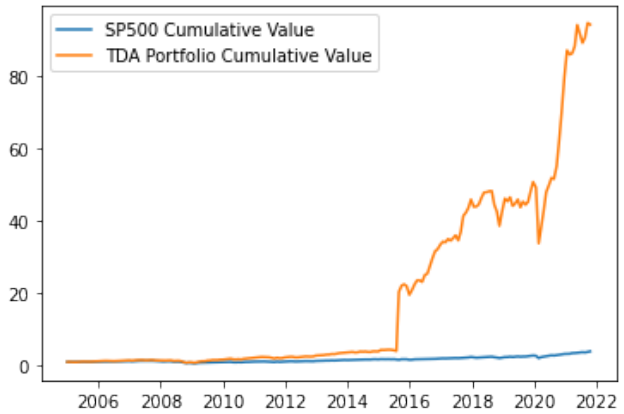

Figure 9: Cumulative return for one dollar of the TDA portfolio vs the SP portfolio

4.4 Discussion

First, based on the first plot, we note that the Kalman filter produces better results when looking at when the probabilities rise above 0.3. This represents when the TDA value is high, and when it would be likely that Apple would be picked as part of the ten stock portfolio. We see this superior performance in mid-2018 and mid-2020, when the probabilities predicted the correct movement (or a substantial rise in value).

We see from the histograms that our strategy has a higher mean return than the SP500, but has a higher variance which means it takes on more risk to generate superior returns. We see our strategy makes over ninety times the investment from 2005-2021. However, our strategy consistently gets those returns with a low overall correlation to the market, which is attractive to an investor who wants to beat the market regardless of market conditions. We also do this with only price data, which is free, which means anyone can run our strategy and generate superior returns. Few strategies in the literature do so without fundamental data, but we find this framework works well with only fitting on prices.

Our method works well because during crises and upswings the financial time series strongly depends on the state of the market. The empirical analysis shows that the time series of the Lp-norms exhibit strong growth around the primary peak emerging during the beginning of an upswing. This empirical observation leads us to believe that TDA serves as a good framework to predict single stock substantial rises in value. Also, we see an increased persistence of loops appearing in point clouds as the market undergoes transition from the ordinary to the “cooled” state. Remarkably, the variability of the time series of the Lp-norms of persistence landscapes, quantified by the variance and the spectral density

at low frequencies, demonstrates a strong rising trend for almost all of the key periods where the market substantially picked up in value. This observation also applied on the single stock level, where the LP Norm would pick up on volatilities right before a stock would rise in value.

5 Conclusions

-Conclusions

We use topological data analysis (TDA) to create a systematic investment strategy from 2015-2021. In our approach, we study the topological features appearing in the sliding windows, persistence diagrams. And how the modified structure of the distance metrics, the persistence landscapes, can perform as a more statistical tool to present us the topological features with a good stability. We also improve the performance by implementing other time series analysis methods we learned from the class by considering the meanings and features of our data set. We source close prices from around 8000 stocks from the NASDAQ and NYSE exchange to generate portfolios every month from 2005-2021. We find a 90x return during this period using our framework, which beats the SP500 and other models.

-Future Work

By analyzing our probability results against the stock price, we found that, the probability doesn't alert the increase of the stock price early enough. Sometimes the stock price has already increased for some days, until the signal appears. So, it would be an improvement for this model if we can move our signal a little bit forward or reconstruct the signal by adding the future signal as an embedding. Then it would be very important for us to make predictions on the stock price.

As shown in Figure 10, after analyzing the daily log return of stock price, we can find that, it follows a Gaussian distribution very well, which is a good news for us. Since the stock price itself, it's a very messy dynamic system, it's very hard to make prediction on it purely based on the previous stock price. So we can apply HMM method to a time series of log return, try to make predictions on the log return in the future days, e.g. 10 days.

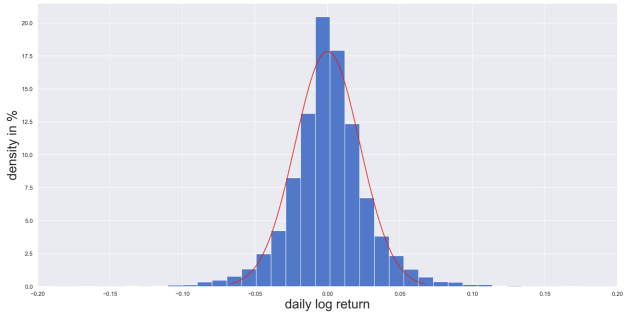

Figure 10: Daily log return density plot of Apple

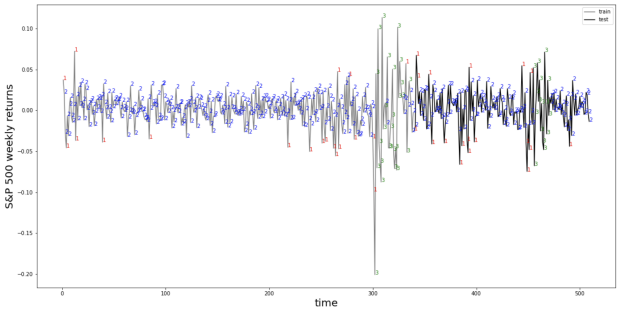

Figure 11: Hidden states tag for SP500 log return

We can implement this by using the similar method we have done in Homework 3, which is to the SP500 data. We first fit the HMM model with stock price we have before time point t. Then make predictions on the log return in the future given the initial state which is the log return on t. After that, we can compute the stock price using the log return, and run our algorithm get the new probability. We can use this probability as a supplementary signal which can help us make better selection for the stocks to construct our portfolio.

References

Berwald, J., Gottlieb, J., and Munch, E. (2018). Computing wasserstein distance for persistence diagrams on a quantum computer. *ArXiv*, abs/1809.06433.

Bjork, T. (2009). *Arbitrage Theory in Continuous Time*. Oxford University Press.

Gideaand, M. and Katz, Y. (2018). Topological data analysis of financial time series: Landscapes of crashes. *Physica A: Statistical Mechanics and its Applications*, 491:820–834.

Prabowo, N. (2021). With topological data analysis, predicting stock market crashes. *International Journal of Informatics and Information System*.

Shumway, R. H. and Stoffer, D. S. (2005). *Time Series Analysis and Its Applications (Springer Texts in Statistics)*. Springer-Verlag.

6 Student contributions

Aaron Kreiner: Universe Selection, Initial TDA Indicator, Paper

Travis Luo: Kalman Filter, TDA Tuning, Paper