STAT 613 001:Machine learning for finance application: Exchange Market Index Replication

Xiaoqian Chen (xc33)

May, 2020

Contents

1 Survey

1.1 Background

Fund managers and investors are increasingly adopting passive investing strategies. Passive investing is a buy-and-hold strategy that involves buying an asset (or set of assets) to hold it for a long-term horizon.[5]3. Index investing is probably the most popular passive investment strategy. Since a market index cannot be directly traded, fund managers have two ways, Full Replication and Sampling Replication.

Full Replication is buying all of the securities that make up the index, that is usually for large and liquid indexes like the S&P 500 or Russell 3000. Full Replication might not be the best way to deliver the returns of the index, but not all markets are as liquid as the companies in the S&P 500, and some indexes include thousands of constituents, such as NASDAQ. It can achieve an exact index replication, but it entails significant costs and may be difficult to manage. Also, it may include illiquid stocks, increasing the risk.

Therefore, some fund managers prefer Sampling Replication, buying the securities in an index that provide the most representative sample of the index based on correlations, exposure and risk. It is a common practice to create a well-diversified portfolio of securities that best replicate the index performance. In this case, the tracking error is arguably greater than under full replication, but partial replication reduces transaction costs and tends to avoid illiquid stocks. The main advantages of these passive strategies are the low fees and simplicity, due to the lack of active management, but they usually entail small returns.

The problem is how to replicate an index with a small subset of stocks belonging to such an index with minimal tracking error. In Statistical machine learning, it is called Dimensionality Reduction, which seeks for a different, smaller dataset that uses significantly fewer features or observations to represent the original information.There are several methods to reduce dimension. We choose the standard supervised method principal component analysis and the unsupervised autoencoders neural networks.

- Principal component analysis (PCA): Finds the linear transformation that captures most of the variance in the existing dataset;
- Autoencoders: Uses neural networks to compress data non-linearly with minimal loss of information.

1.2 Index replication

The main idea of index replication is come from [1], there are two conceptual approaches we can choose from, when aiming to replicate (or approximate) a stock index through a subset of stocks.

- Identify a small group of stocks which historically have given a performance very similar to that of the observed index.
- Identify a small group of stocks which historically have represented an over-proportionally large part of the total aggregate information of all the stocks the index comprises of.

The method is rank all stocks by their proximity to the auto-encoded information and create an equally weighted

portfolio from the auto-encoder basis of the top leading stocks. [1] replicate S&P500 index, we would like to check if this method works for larger exchange market, NASDAQ composite has about 5 times stocks as S&P500.

The bottleneck structure of an auto-encoder creates a compressed set of information from which all stocks are re-created (through linear and non-linear relationships). we can compare the result's we've obtained with respect to Principal Component Analysis (PCA) since this is a standard method for dimensionality reduction in linear space.

Hence, the NASDAQ composite will be replicated by a subset of stocks is identified by using an autoencoder and PCA. We will focus on creating and training a autoencoder to minimize the reconstruction error and how it can be used for tracking a market index. Special attention will be paid to comparison between machine learning method autoencoders and statistical method PCA.

A stock index is a measurement that represents the aggregated value of a collection of stocks that describes a particular market or a segment of it. The value of a stock market index is computed from the prices of stocks that compose it according to some weighting schema. The two main approaches are as follows:

- Price-weighting: It is computed as a weighted average of the stock prices, such as in the case of Dow Jones Industrial Average (DJIA).
- Capitalization-weighting: Here, stocks are weighted based on their total market capitalization, which refers to the total number of common shares outstanding multiplied by the price per share, such as SP500 and NASDAQ Composite.

There are several other schemes, including the equal-weighting of stocks, which assigns the same weight to all components in the portfolio. As mentioned earlier, it is not possible to directly invest in a market index. To gain access to the selected index, we may construct a diversified portfolio of stocks that tracks its performance by following the value of the index, called a tracking portfolio. Even the NASDAQ composite is Capitalization-weighting, we would like to explore how the simple equal-weighted tracking portfolio will perform.

1.3 Unsupervised Algorithms:Principal Component Analysis

Principal Component Analysis (PCA), invented in 1901 by Karl Pearson [4], finds new features that reflect directions of maximal variance in the data while being mutually uncorrelated, or orthogonal. PCA finds principal components as linear combinations of the existing features and uses these components to represent the original data. It reduces dimensionality by projecting the original data into the principal component space. The number of components is a hyperparameter that determines the target dimensionality and needs to be equal to or smaller than the number of observations or columns, whichever is smaller. The number of components that capture, for example, 95% of the original variation relative to the total number of features provides an insight into the linearly-independent information in the original data.

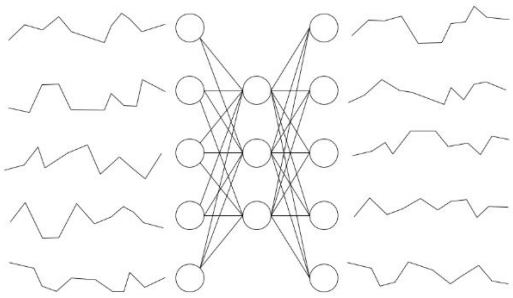
The procedure first moves the origin of the axes to the average point of the data, i.e. the mean. Subsequently, it determines a first axis in order to maximize the variance of the data projected on it. Then, it determines a second axis, orthogonal to the first, so as to maximize the variance of the data projected on it. Following the same principle, it determines a third orthogonal to the first and second, and so on, until all axes are determined.

Given a matrix of N-case training data A $(N \times p)$, reduced system $A_{reduced}(N \times q)$ with q $(q \leq p)$ principal components with the following PCA algorithm.

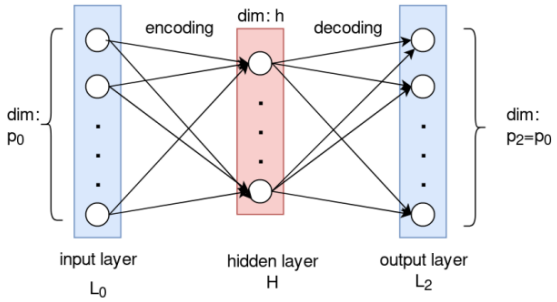
1. Mean normalize $A = A - mean(A)$.
2. Compute the covariance matrix of the mean normalized data, $\Sigma = \frac{1}{N-1} A A^T$.
3. Find the eigenvectors and eigenvalues of Σ.
4. Sort the eigenvectors in descending eigenvalue order as $s_1, s_2, \ldots, s_p$, where $s_1 \geq s_2 \geq \ldots s_p$.
5. Let $A_{reduced}$ be new matrix with the first q column vectors. $A_{reduced}$ is the new basis for our system.

1.4 Supervised Algorithms: Autoencoders

We are going to look at a representation of time series data given by an autoencoder (AE) [2]in a latent space with reduced dimensionality. This allows us to take into account the overall market. In this case, at each time t, the price of an asset is assumed as the $p(t) = f(t, M, A, T)$ function of the overall market M, asset-specific factors A, and trading activity T. There is a clear cross-influence among assets that are non-linear that depends on the market and the specific sector the asset belongs to. The purpose of an autoencoder is to learn the latent representation that minimizes the reconstruction error. Thus, an AE can be used to capture price movements in a smaller non-linear latent space and use this representation in order to limit price components that are specific to a particular asset. In other terms, we aim to gather reconstructed time-series data for each stock that passes through a latent space represented by the hidden layer, which is the "core" of the autoencoder. We expect that the resulting series will be more affected by the other stocks in the market (M) but still take the asset-specific dynamics of the value (A , T) into account.



(a) The idea behind an AE working on time series data



(b) MLP architecture of a 3 layers Autoencoders

Figure 1: Architecture of Autoencoders networks

The Figure1 shows the idea behind an autoencoder working on time series data. At each time t, the autoencoder maps the price movements given as input to the latent space, and from there it provides a reconstructed version that, due to the lower dimensionality of the representation, is depending on non-linear relations between the asset prices.

There is the simplest form of an auto-encoder, which is a feed-forward, non-recurrent 3 layers fully-connected network, having an input layer, an output layer and one hidden layers connecting them. Neurons between two adjacent layers are fully pairwise connected, but neurons within a single layer share no connections. There is

only one hidden layer with dimension h. The dimension of the output layer is equal to the dimension of input layer. The response function σ is non-linear transformations of weighted data, which applied for all neurons in the hidden layer and output layer, such as sigmoid function, RELU function. Deep Autoencoders extend this shallow architecture by stacking multiple layers for encoding and decoding.

Each neuron implements a parametric mathematical function, generally taking the following form:

$$f(x) = \sigma(\sum_{i=1}^{h} W_i x_i + b_i) \tag{1}$$

The Rectified Linear Unit(ReLU) is very commonly used activation function in deep learning models. The function returns 0 if it receives any negative input, but for any positive value x it returns that value back. We use the ReLU activation function in all hidden layers of the autoencoders implemented here. Besides accelerating the training process, the ReLU function helps to avoid the vanishing gradient problem when training deep architectures since it doesn't cause a small derivative.

$$RELU(x) = max\{0, x\} \tag{2}$$

The mean square error (MSE) was used as loss function and RELU was used as response function. The main goal is to minimize the result of sum of MSE for training set to get a adapted network. In an autoencoder, we set the target values as $Y_i = X_i$. $\hat{Y}_i$ is the output of the autoencoders.

$$MSE = \sum_{i=1}^{N} (Y_i - \hat{Y}_i)^2 \tag{3}$$

2 experimental

In this section, I create and train a Autoencoders to identify the most communal features of the stocks composing the NASDAQ composite index, to replicate such an index. The strategy consists of creating an equal-weighted portfolio with those stocks whose reconstruction error after passing through the autoencoder are lower, assuming that they are the most representative stocks of the market index. Also, we compare the result's we've obtained with respect to Principal Component Analysis (PCA) since this is a standard method for dimensionality reduction.[3]

2.1 Data exploration and preparation

The NASDAQ Composite is a stock market index of the common stocks and similar securities listed on the NASDAQ stock market. Along with the Dow Jones Industrial Average and S&P 500 it is one of the three most-followed indices in US stock markets. I am interested in sampling replication, NASDAQ has about 2,500 stocks, that much more than S&P 500.

The composition of the NASDAQ Composite is heavily weighted towards information technology companies. The NASDAQ-100, whose components are a subset of the NASDAQ Composite's, accounts for over 90% of the NASDAQ Composite's movement. The NASDAQ-100 is made up of 103 equity securities issued by 100 of the largest non-financial companies listed on the NASDAQ stock market. It is a modified capitalization-weighted index and does not have any financial companies. NASDAQ-100 is considered as the best portfolio with 100

stocks that could capture NASDAQ Composite, which is for evaluating self-built portfolio.

The data resource from Yahoo! Finance historical daily price in NASDAQ stock market, from January 1, 2015, to March 31, 2020, for a total of 5 years and 3 months. I only consider the stocks whose data is available for such a period of interest (no missing data), apart from the index. There are total 2,157 stocks daily price and two market Index, NASDAQ-100 and NASDAQ Composite. Stocks data has a shape of $1,298 \times 2,157$ since there are 2,157 series of 1,298 trading days corresponding with the assets forming the NASDAQ composite. Similarly, the index data is a series of 1,298 samples along one single dimension.

Data preparation is a necessary step before training any neural network. It usually includes data splitting and scaling. First, we need to split our data into two sets: one for training the model and another for back testing purposes. Price data from January 1, 2015 , to January 1, 2019 is set as training set $(1,000 \times 2,157)$ and the rest part is test set $(298 \times 2,157)$.

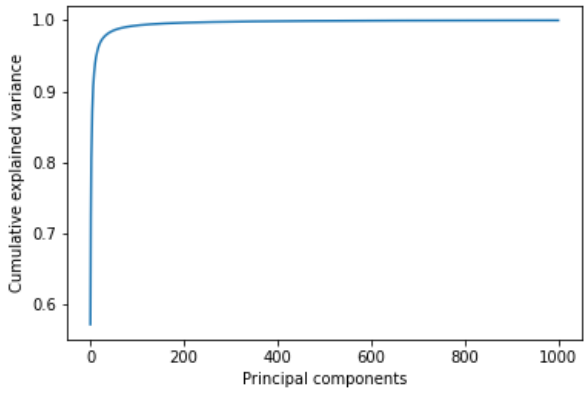
Next, we scale our data to adjust each feature to a given range and hence make the convergence of the neural network model easier since the network parameters are initialized. Input features is re-scaled to the range [0,1] by (Min-Max) normalization.

$$X_{norm} = \frac{X - min(X)}{max(X) - min(X)} \tag{4}$$

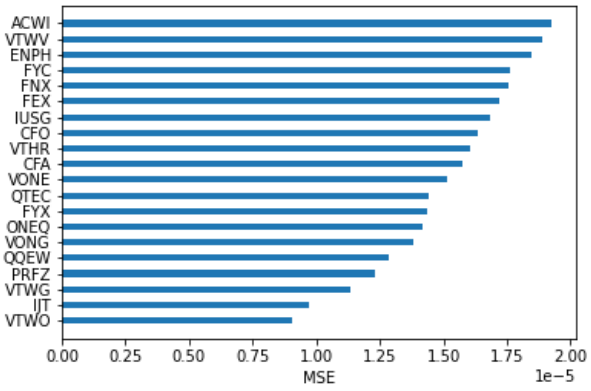
2.2 Implement:PCA and Autoencoders

Principal component analysis (PCA) is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components. The number of components is account as guide for the number of hidden layers in Autoencoders.

Firstly, Figure 2(a) plotted the cumulative explained variance by varying the number of components, confirming that 50 components is enough to cover the variance of the data at 95%. 50 components are used to reconstruct the time series, just like they are for the autoencoder. Figure 2(b) shows the top 20 stocks with a lower MSE.



(a) The cumulative explained variance in terms of the number of principal components

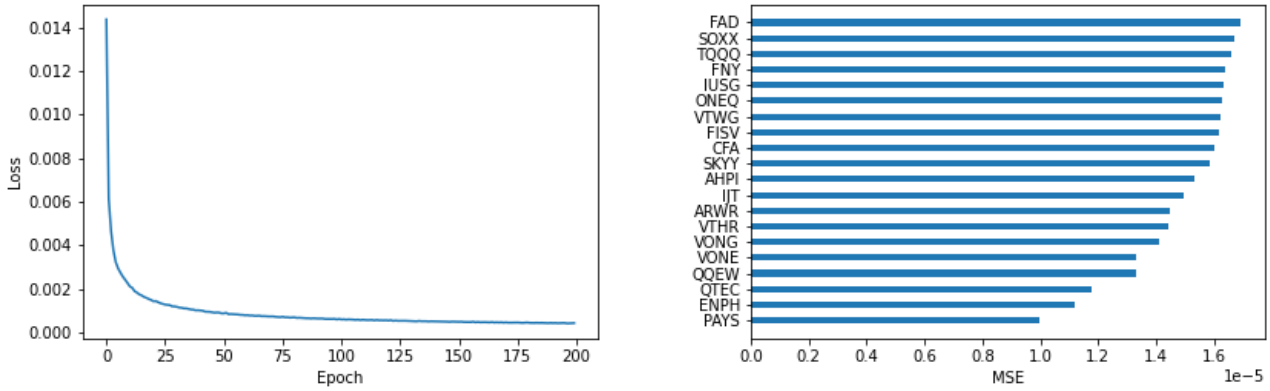


(b) Top 20 stocks with a lower MSE

Figure 2: Principal component analysis

Next, I will focus on creating and training a Autoencoders to minimize the reconstruction error and how it can be used for tracking a market index. Firstly, design the multilayer perception architecture for autoencoder. Start with the simplest 3-layers architecture and the hidden layer size is 50. Then extend this shallow architecture by increasing stacking multiple layers for encoding and decoding. After several implements, the 5 layers autoencoder has the best performance, that is Input(2,157)- Dense(200)-Dense(50)-Dense(200)-Output(2,157). Deeper autoencoders does not tend to have better performance. Deep architectures can exponentially reduce the computation cost and the amount of data required for training. Also, the latent representations learned by a deep autoencoders are relatively robust and useful compared to a shallow autoencoders. Note that it does not mean deeper mean better, it may bring overfitting. Also, when we design neural networks, the simpler model with less parameters is pursued in practice.

The batch size is set as 1, and randomly shuffle the training samples before each epoch. Random shuffle could speed up the convergence of the network and avoid overfitting. Mean square error (MSE) is as the loss function to measure the difference between the reconstructed input and the original one and the adaptive learning rate optimization algorithm (Adam) optimizer.



(a) Loss function for 5-layers autoencoders

(b) Top 20 stocks with a lower MSE

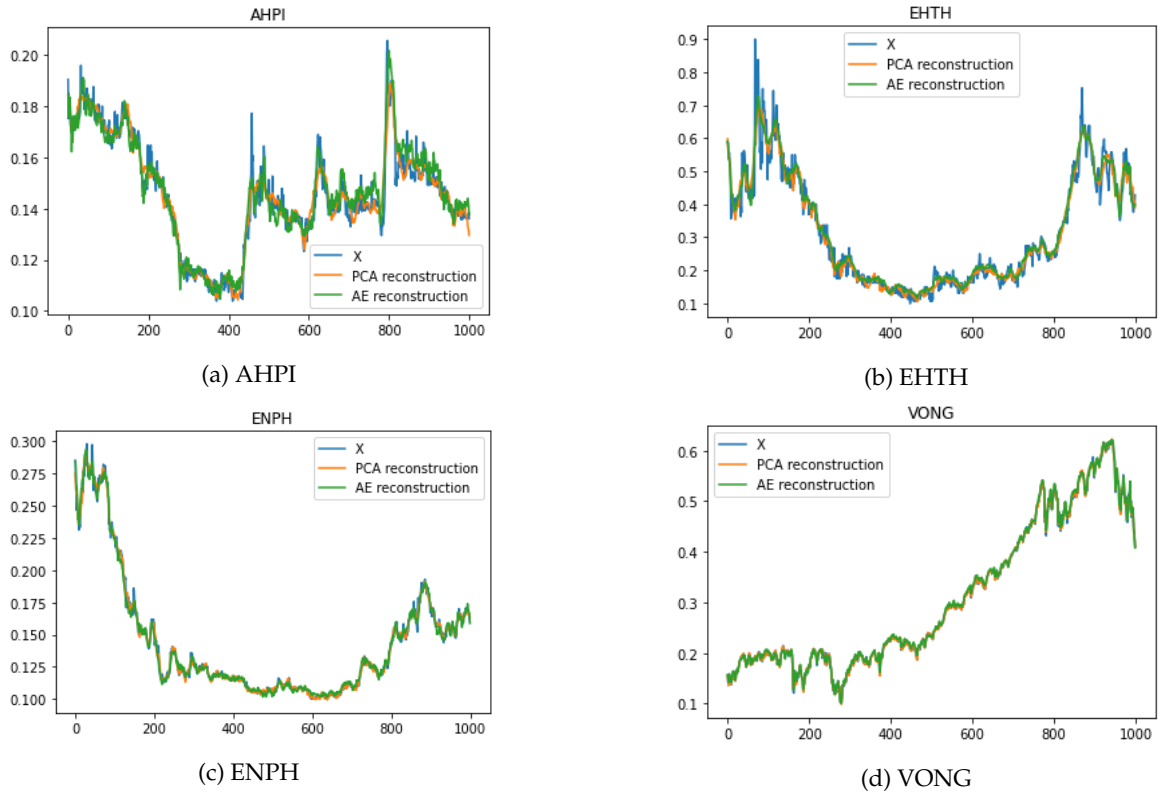
Figure 3: Training of 5-layers Autoencoders

Figure 3 shows loss curve in the training with respect to epochs, we can see that the model rapidly converges over the first 20 epochs and then the loss decreases slowly until reaching a plateau, indicating the convergence of the model. The training MSE: 0.000210.

Randomly chosen two stocks symbol AHPI and EHTH reconstruction offered by PCA(components= 50) and the autoencoders is showed in Figure 4.(a) and (b) reconstruction have low error, missing some extreme movement. The reconstruction by the autoencoder of top 20 is generally closer to the index since it is affected more by the other stocks, while the PCA is closer to the original price series. Both are highly correlated to the other series. (c) and (d) are from the top 20 lowest MSE stocks, that are almost perfectly reconstitute price trend.

Compare the top 20 lowest MSE reconstruction stocks, there are 11 common stocks for both PCA and AE, that are ENPH, IUSG, VTHR, CFA, VONE, QTEC, ONEQ, VONG, QQEW, VTWG and IJT. What surprise is that none of these 11 stocks belongs to NASDAQ-100 components. Further look at these componies, 10 out of 11 stocks are in financial section, except from ENPH. Such that, ONEQ is Fidelity NASDAQ Composite Index Tracking Stock Fund. VONG is Vanguard Russell 1000 Growth Index Fund ETF Shares. QQEW is First Trust NASDAQ-100

Equal Weighted Index Fund.This result shows that PCA and autoencoders have capability to pick up the most significant factors for index price trend. However, some fund managers dislike too heavy financial sector in the portfolio.



(a) AHPI

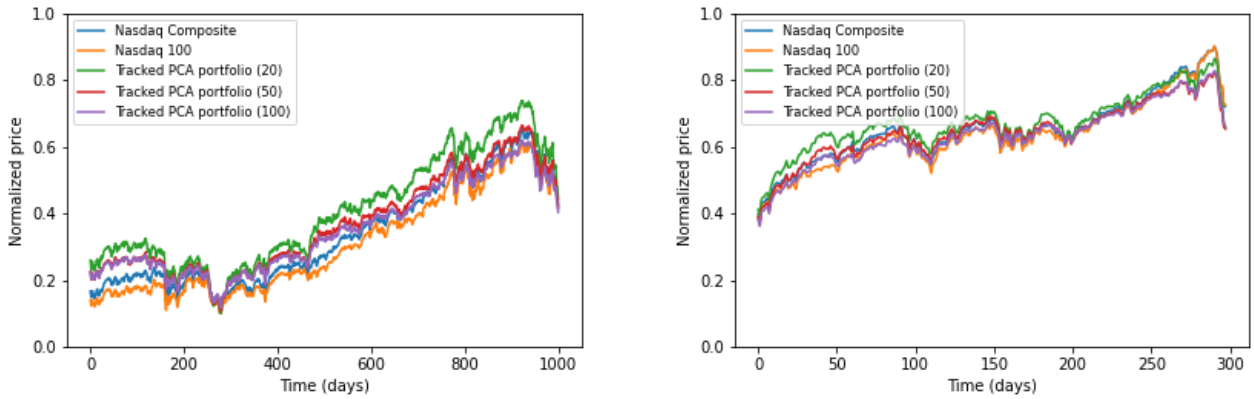
(b) EHTH

(c) ENPH

(d) VONG

Figure 4: Reconstruction

Now it is time to replicate the NASDAQ-composite index through a portfolio containing a small group of stocks. The basic idea is to identify the components that better represent the aggregate information of all of the stocks composing the index. We pick up the top 20, 50, 100 leading stocks by PCA and autoencoders, then create equal-weighting portfolio.

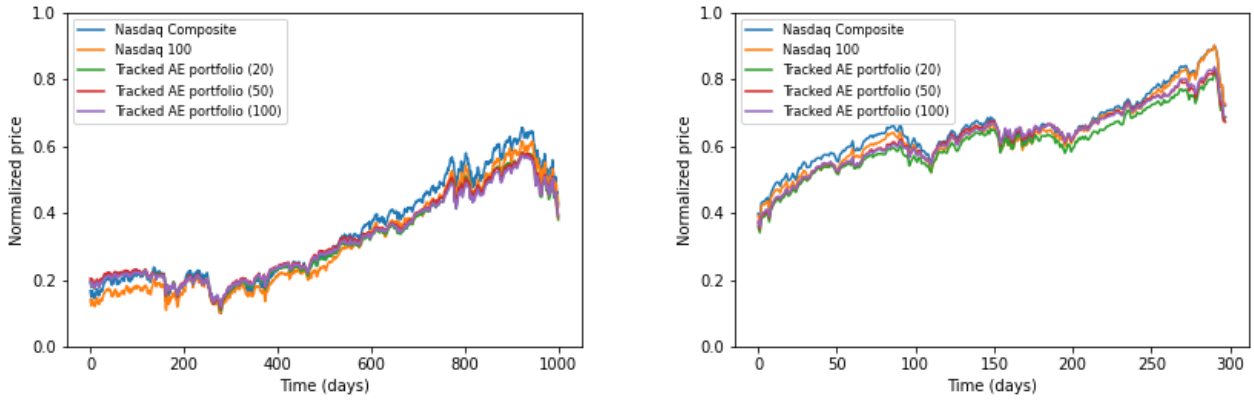


(a) in-sample

(b) out-of-sample

Figure 5: Tracked PCA portfolio

From Figure 5, we could observe that the tracked portfolio with 20, 50, 100 stocks picked by PCA. Three tracked portfolios all has good performance, highly correlated to the NASDAQ-composite index, correlation coefficient of portfolio (in-sample) is 99.69%. Ever the out-of-sample performance of tracked portfolios give correlation coefficient of portfolio with 20 stocks (out-of-sample) is 97.69%. There is not necessary to increase the number of components in portfolio. Three tracked portfolios all outperform NASDAQ-100 index,even higher than NASDAQ-composite index, and the 20 stocks portfolio has the highest price.



(a) in-sample

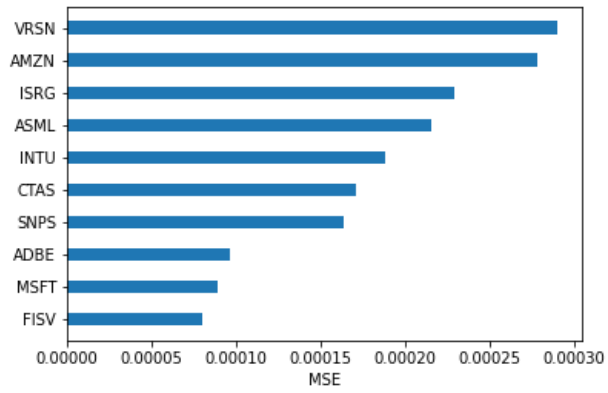
(b) out-of-sample

Figure 6: Tracked AE portfolio

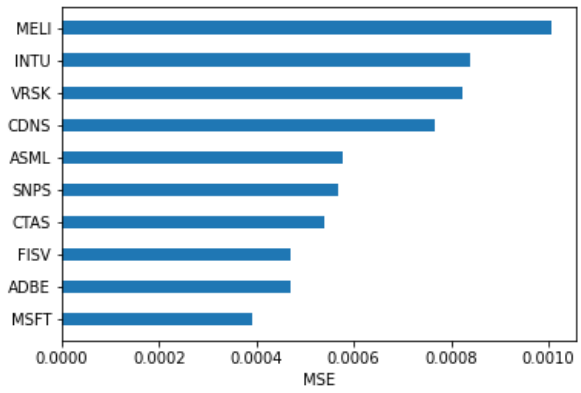
Portfolios created by autoencoders has similar performance, highly correlated to NASDAQ-composite index with at least 99.70% correlation coefficient. The price of all three portfolio are in almost same trend, slightly lower than NASDAQ-composite index. The price difference in terms of components numbers is minor, and 20 stocks portfolio shows lowest average price in out-of-sample.

2.3 Focus on non-financial sector

Traditionally, technology stocks have made up a large portion of the NASDAQ composite, with larger technology companies especially influencing the total composite index. The NASDAQ-100 index is made up of 100 the largest non-financial companies listed on the NASDAQ stock market. The NASDAQ-100 is very focused on the technology sector, and is a widely-held tracking index for futures, options and exchange-traded fund trading. Fund managers care more about technology sector than finance sector in NASDAQ market. From last subsection, PCA and AE have successfully replicate the NASDAQ composite index, with essential financial stocks. Furthermore, we would like to focus on non-financial stocks in NASDAQ, which means the no more single stocks track index trend. In this subsection, we will track the NASDAQ composite index from list of NASDAQ-100 components without financial companies.

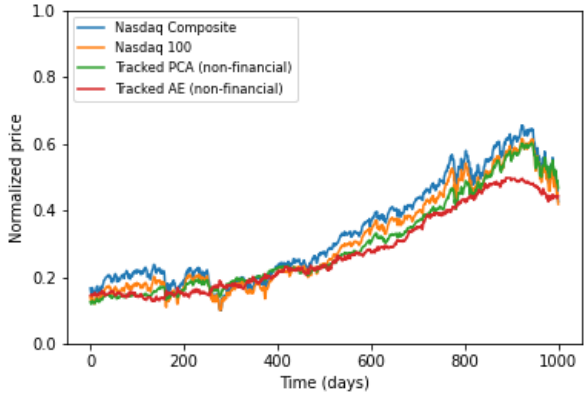


(a) PCA

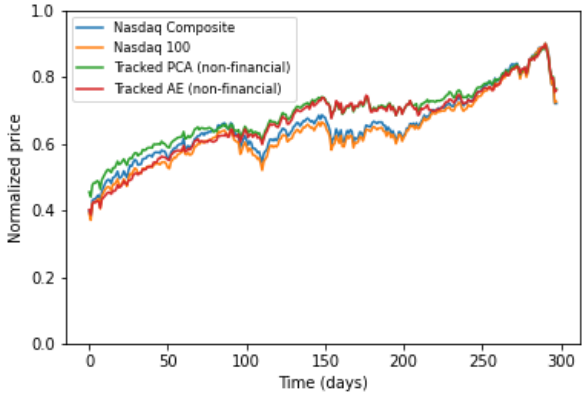


(b) AE

Figure 7: Top 10 non-financial stocks with lowest MSE



(a) in-sample



(b) out-of-sample

Figure 8: Tracked PCA and AE portfolio with 10 non-financial stocks

Only with the largest 100 non-financial companies, AE equal-weighted replication with 10 stocks has correlation coefficient 97.41% in sample and 93.61% out of sample. PCA equal-weighted replication with 10 stocks is closer relative to NASDAQ-composite, 98.12% in sample and 96.53% out of sample. Overall, PCA portfolio has higher price than AE portfolio, which still lower than NASDAQ-100 and NASDAQ-composite in sample.

3 Conclusion

This project utilized PCA and AE to identify the most communal features of the stocks composing the NASDAQ-Composite index, to replicate such an index. As a statistical unsupervised algorithm, PCA finds principal components as linear combinations of the existing features and uses these components to represent the original data. AE can be used to capture price movements in a smaller nonlinear latent space with minimizing the reconstruction error, and use this representation in order to limit price components that are specific to a particular asset.

Both of PCA and AE are able to find the most significant stocks for the price movement of NASDAQ-Composite index. When we consider the all 2,157 stocks without missing data in the past 1,298 trading days, PCA and AE reduced dimension to about 20 essential stocks, of which equal-weighted portfolio have large than 97% correlation with NASDAQ-Composite index in and out of sample. Among these top 20 stocks, there are 11 common stocks for PCA and AE. In addition, 10 out of 11 these common stocks are financial sectors, tracking index for futures, options and exchange-traded fund trading. Such that ONEQ (Fidelity NASDAQ Composite Index Tracking Stock Fund), tracks NASDAQ Composite Index. When replicating NASDAQ-Composite index with all stocks , PCA and AE identify a large part of financial stocks which historically have given a performance very similar to that of the observed index.

If we focus on the 100 largest non-financial stocks in NASDAQ market, 10 stocks portfolio is good enough to represent an over-proportionally large part (correlation coefficient $\geq$ 93%) of the total aggregate information of all the stocks the index comprises of. The 10 equal-weighted stocks portfolio with PCA algorithm perform better than AE, shows overall higher price and larger correlation coefficient.

This project is an interesting financial application with machine learning. A large NASDAQ market with about 2,500 stocks, would represented by about 10 technology stocks or 20 stocks with financial sector. Before implements, I guess that AE may outperform PCA as AE has more powerful expression in nonlinear latent space. We learn that more complicated algorithm does not means better in practice.

Remark (Data resource and Codes)**.** *Price data is gather from Yahoo! Finance and the list of symbols is download from Wikipedia. The code is trained with Python3. Dataset and codes are available at: https://github.com/Chenxq0709/IndexReplication*

References

[1] J. Heaton, N. Polson, and J. Witte. Deep learning for finance: deep portfolios: J. b. heaton, n. g. polson and j. h. witte. *Applied Stochastic Models in Business and Industry*, 33, 10 2016.

[2] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[3] Krish Naik. *Hands-On Python for Finance: A practical guide to implementing financial analysis strategies using Python*. Packt Publishing Ltd, 2019.

[4] Karl Pearson. Principal components analysis. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 6(2):559, 1901.

[5] Niklas Piispanen, Riku Sundqvist, Risto Vuotila, and Ville Matilainen. Applications of deep learning in finance. *Emerging Technology Adoption and Use*, page 4.