STEVENS INSTITUTE OF TECHNOLOGY

FINANCIAL ENGINEERING

QUANTIFYING AND INCREASING DIVERSIFICATION IN INVESTMENT PORTFOLIOS

AUTHOR:

SHAANTANU BHAKUNI (10433501)

SHREYAS JAIKUMAR (10441437)

QIKAI YAN (10440218)

SUPERVISOR:

DR. CRISTIAN HOMESCU

May 15, 2020

Abstract

In this paper, as in accordance with the title, we intricately quantify and diversify financial assets by implementing various deep learning algorithms, machine learning models, certain variations of cluster analysis and by also adopting factor-based approaches to more effectively measure and manage quantification and diversification in multi-asset portfolios. Additionally, we have implemented certain models for the purpose of Risk-Management.

1 Introduction

As we know, multi-asset investment solutions have become increasingly popular among sophisticated institutional investors focusing on efficient harvesting of risk premia across and within asset classes. However, one key challenge in the construction of diversified multi-asset portfolio strategies is that even a seemingly well-balanced allocation to many asset classes can eventually translate into a portfolio with a very concentrated set of underlying risk exposures.

After having read through this paper i.e., ***"Proverbial Baskets Are Uncorrelated Risk Factors! A Factor-Based Framework for Measuring and Managing Diversification in Multi-Asset Investment Solutions" by (Martellini, Lionel; Milhau, Vincent., 2017)*** we came to understand that by following a factor-based investment approach (ENB, PCA and MLT), it will lead to a better structuration of the investment process in the design of the multi-asset investment solutions.

The second paper which proved to be of paramount importance for our project is ***"Minimum Risk Versus Capital and Risk Diversification Strategies for Portfolio Construction" by (Francesco, Cesarone; Stefano, Colucci., 2017).*** In this paper, the authors have proposed extensive empirical analysis in three categories of portfolio selection models with various objectives i.e., Minimization of Risk, Maximization of Capital Diversification and Uniform Distribution of Risk Allocation

This was also a relevant reference i.e., ***"Portfolio Asset Allocation Optimization with Deep Learning" by (Marcos Lopez de Prado., 2017)*** wherein we came to understand the concepts of Equal Risk Parity and Hierarchical Risk Parity in the portfolio construction methodologies.

This paper i.e., ***"Measuring Portfolio Diversification based on Optimized Uncorrelated Factors" by (A. Meucci; A. Santangelo; R. Deguest., 2013)*** proved to be very useful for us wherein it helped in providing an in-depth understanding of the quantification procedure by helping us to compute the Principal Components Torsion Matrix and the Minimum Linear Torsion Matrix, which as specified earlier is an integral part for quantification and diversification analysis.

Also, ***"Financial Time Series Forecasting with Deep Learning: A Systematic Literature Review" by (Omer Berat Sezera; M. Ugur Gudeleka; Ahmet Murat Ozbayoglua., 2019)*** proved to be very useful for us wherein it helped in providing an in-depth understanding of Recurrent Neural Networks

In the rest of the report we show our methodology, portfolio construction along with quantification and diversification analysis in Section **2**. We highlight our results obtained, numerical outputs and the various graphical illustrations in Section **3**. Finally, we state our conclusion along with future improvements in Section **4**.

2 Methodology

2.1 DIVERSIFICATION PORTFOLIOS CONSTRUCTION

Input Data

- Daily Close Prices of Individual Stocks pertaining to the S&P500 Index
- Daily Returns of Individual Stocks pertaining to the S&P500 Index
- (NxN) Co-Variance Matrix
- (NxN) Torsion Matrix

All Data related information are obtained from Yahoo Finance and Bloomberg

A EQUAL RISK PARITY PORTFOLIO

This is a specific type of risk parity approach to portfolio construction wherein the goal is to have assets that contribute equally to the overall volatility of the portfolio.
It comprises of various methodologies as discussed in [Cesarone; Colluci., 2018] but we chose to use hierarchical clustering method to distribute equal risk to the asset, taking Value at Risk(VaR) and log returns as risk distributing metrics.
This type of portfolio is constructed by implementing the following steps:

- From a universe of assets, a distance matrix is formed based on the correlation of assets based on a rolling window of VaR or any other metric chosen.
- By using this distance matrix, the assets are hierarchically clustered into a tree
- A cutting level is decided and then the number of clusters are formed
- Equal weights are distributed amongst each cluster and then again equal weights are distributed amongst the sub groups till every asset is allocated a weight

B HIERARCHICAL RISK PARITY PORTFOLIO

Hierarchical Risk Parity applies modern mathematics (graph theory and machine learning techniques) to build a diversified portfolio based on the information contained in the covariance matrix. However, unlike quadratic optimizers, Hierarchical Risk Parity does not require the invertibility of the covariance matrix. In fact, Hierarchical Risk Parity can compute a portfolio on an ill-degenerated or even a singular covariance matrix, an impossible feat for quadratic optimizers. Monte Carlo experiments show that Hierarchical Risk Parity delivers lower out-of-sample variance than CLA, even though minimum-variance is CLA's optimization objective. Hierarchical Risk Parity also produces less risky portfolios out-of-sample compared to traditional risk parity methods. This particular type of portfolio is constructed by implementing the following steps:

- From a universe of assets, a distance matrix is formed based on the correlation of assets.
- By using this distance matrix, the assets are hierarchically clustered into a tree
- Within each branch of the tree, we then form the Minimum Variance Portfolio (normally between just two assets)
- Finally, we iterate over each level wherein we optimally combine the sub-portfolios at each node.

C PORTFOLIO ASSET ALLOCATION OPTIMIZATION USING DEEP LEARNING

The previous discussed algorithm for portfolio construction are quite known in literature but offers little flexibility. Here we discuss a completely new, assemble methodologies using modern techniques from data science and numerical finance techniques. The Portfolio thus constructed are highly diversified and can further be optimized for adjusting to an investor's changing risk appetite based on personal goals in life.

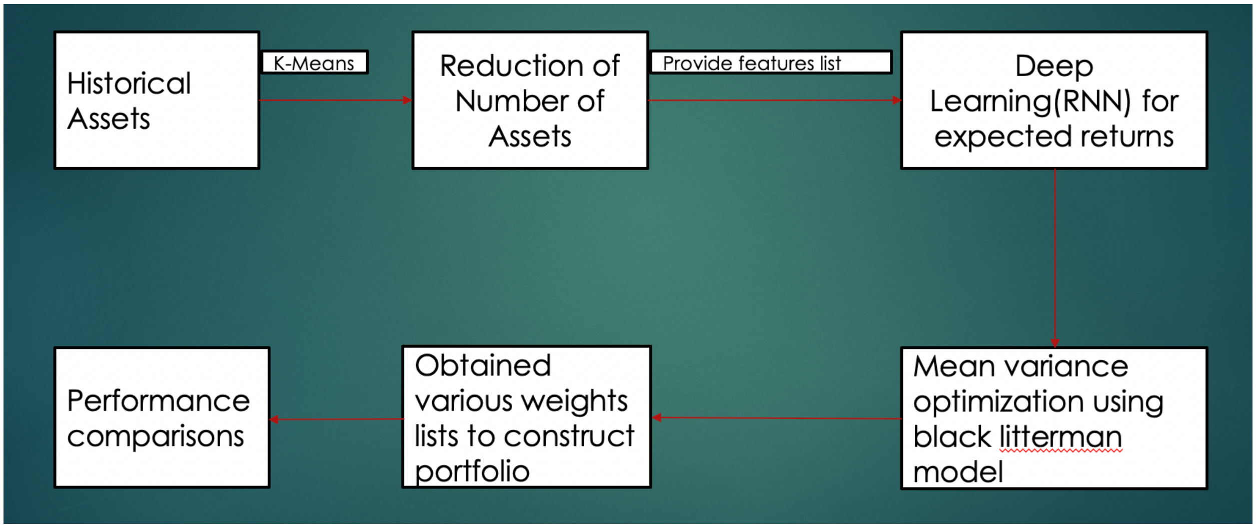


Figure 1) Flow Chart Representation of Portfolio Asset Allocation Optimization Using Deep Learning

Figure 1 Overview:

- Time series of assets are manipulated to be symmetric and used in K-means clustering
- Elbow methods is used to give the optimal number of clusters and the assets closest to the centroid of individual clusters are chosen as the assets that we will proceed with.
- In those selected assets we perform a deep learning algorithm(RNN) to get the expected returns for the last year of our analysis (2019). We can use a user defined feature list as input of RNN.
- The expected returns list of those assets are passed as an input in Black Litterman algorithm which gives room for investors' views. Thus, the algorithm becomes a combination of both machine learning and mean variance optimization.
- Various variations of the mean-variance optimization like Minimum Volatility, Maximum Sharpe Ratio can thus be used.

Now, we will discuss each component of the Portfolio Asset Allocation Optimization Using Deep Learning in detail.

C.1 k-MEANS CLUSTERING

k-means clustering is a method of vector quantization, originally from signal processing, that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi Cells. It is popular for cluster analysis in data mining. k-means clustering minimizes within-cluster variances (squared Euclidean distances), but not regular Euclidean distances, which would be the more difficult Weber problem: the mean optimizes squared errors, whereas only the geometric median minimizes Euclidean distances. For instance, better Euclidean solutions can be found using k-medians and k-medoids. The problem is computationally difficult (NP-hard); however, efficient heuristic algorithms converge quickly to a local optimum. These are usually similar to the expectation-maximization algorithm for mixtures of Gaussian distributions via an iterative refinement approach employed by both k-means and Gaussian mixture modeling. They both use cluster centers to model the data; however, k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes. The algorithm has a loose relationship to the k-nearest neighbor classifier, a popular machine learning technique for classification that is often confused with k-means due to the name. Applying the 1-nearest neighbor classifier to the cluster centers obtained by k-means classifies new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm. k-means clustering is rather easy to apply to even large data sets, particularly when using heuristics such as Lloyd's algorithm. It has been successfully used in market segmentation, computer vision, and astronomy among many other domains. It often is used as a preprocessing step for other algorithms, for example to find a starting configuration. In cluster analysis, the k-means algorithm can be used to partition the input data set into k partitions (clusters). However, the pure k-means algorithm is not very flexible, and as such is of limited use (except for when vector quantization as above is actually the desired use case). In particular, the parameter k is known to be hard to choose (as discussed above) when not given by external constraints. Another limitation is that it cannot be used with arbitrary distance functions or on non-numerical data. For these use cases, many other algorithms are superior.

RELATIONSHIP BETWEEN k-MEANS CLUSTERING AND PRINCIPAL COMPONENT ANALYSIS

The relaxed solution of *k*-means clustering, specified by the cluster indicators, is given by principal component analysis (PCA). The intuition is that k-means describe spherically shaped (ball-like) clusters. If the data has 2 clusters, the line connecting the two centroids is the best 1-dimensional projection direction, which is also the first PCA direction. Cutting the line at the center of mass separates the clusters (this is the continuous relaxation of the discrete cluster indicator). If the data have three clusters, the 2-dimensional plane spanned by three cluster centroids is the best 2-D projection. This plane is also defined by the first two PCA dimensions. Well-separated clusters are effectively modelled by ball-shaped clusters and thus discovered by k-means. Non-ball-shaped clusters are hard to separate when they are close. For example, two half-moon shaped clusters intertwined in space do not separate well when projected onto PCA subspace. k-means should not be expected to do well on this data. It is straightforward to produce counterexamples to the statement that the cluster centroid subspace is spanned by the principal directions.

C.2 RECURRENT NEURAL NETWORKS

A Recurrent Neural Network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. The term “recurrent neural network” is used indiscriminately to refer to two broad classes of networks with a similar general structure, where one is finite impulse and the other is infinite impulse. Both classes of networks exhibit temporal dynamic behavior. A finite impulse recurrent network is a directed acyclic graph that can be unrolled and replaced with a strictly feedforward neural network, while an infinite impulse recurrent network is a directed cyclic graph that cannot be unrolled. Both finite impulse and infinite impulse recurrent networks can have additional stored states, and the storage can be under direct control by the neural network. The storage can also be replaced by another network or graph, if that incorporates time delays or has feedback loops. Such controlled states are referred to as gated state or gated memory, and are part of long short-term memory networks (LSTMs) and gated recurrent units. This is also called Feedback Neural Network. Basic RNNs are a network of neuron-like nodes organized into successive layers. Each node in a given layer is connected with a directed (one-way) connection to every other node in the next successive layer. Each node (neuron) has a time-varying real-valued activation. Each connection (synapse) has a modifiable real-valued weight. Nodes are either input nodes (receiving data from outside the network), output nodes (yielding results), or hidden nodes (that modify the data enroute from input to output). For supervised learning in discrete time settings, sequences of real-valued input vectors arrive at the input nodes, one vector at a time. At any given time-step, each non-input unit computes its current activation (result) as a nonlinear function of the weighted sum of the activations of all units that connect to it. Supervisor-given target activations can be supplied for some output units at certain time steps. For example, if the input sequence is a speech signal corresponding to a spoken digit, the final target output at the end of the sequence may be a label classifying the digit. In reinforcement learning settings [*Omer Berat Sezera; M. Ugur Gudeleka; Ahmet Murat Ozbayoglua.,*] no teacher provides target signals. Instead a fitness function or reward function is occasionally used to evaluate the RNN's performance, which influences its input stream through output units connected to actuators that affect the environment. This might be used to play a game in which progress is measured with the number of points won. Each sequence produces an error as the sum of the deviations of all target signals from the corresponding activations computed by the network. For a training set of numerous sequences, the total error is the sum of the errors of all individual sequences.

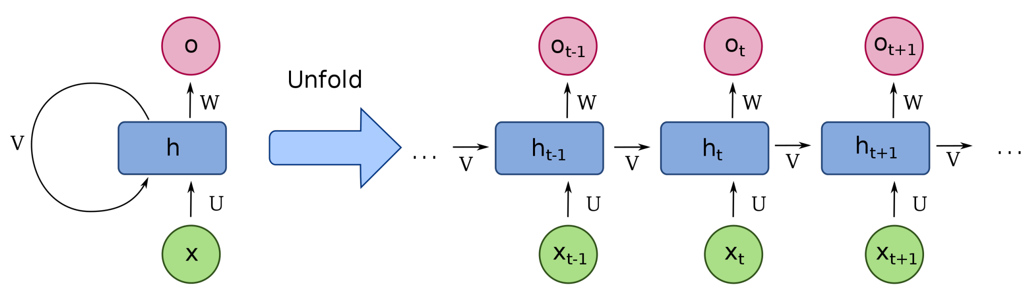


Figure 2) Fully Recurrent Neural Network (RNN)

Figure 2 represents the schematic sketch of a Fully Recurrent Neural Network (RNN) wherein it has only One Unit. From bottom to top, the three states are:

- Input State
- Hidden State and
- Output State.
- U, V, W are the weights of the network

C.3 REGIME SHIFT MODELS

The autoregressive model specifies that the output variable depends linearly on its own previous values and on a stochastic term (an imperfectly predictable term). The stock price has a constant intercept (mean value), slope against previous value, and error term related to variance. This approach assumes that the characteristics of the time series (mean and variance) stay the same during the whole time period under consideration but that is usually not the case. A time series can change behavior completely from one period to the next due to some structural changes. For example, a stock price series can change its behavior drastically from trending to volatile after a policy or macroeconomic shock. Regime shift models address this gap in basic time series modelling by segregating the time series into different “states”. These models are also widely known as state-space models in time series literature. Financial markets can change their behavior abruptly. The behavior of stock prices from one period to the next can be drastically different. So many people try to play the market and fail spectacularly – I’m sure you’ve read about a few examples in the news. When trying to model these prices, we can see that the mean, variance and correlation patterns of stocks can vary dramatically. This, in financial markets, is usually triggered by fundamental changes in macroeconomic variables, policies or regulations. For financial models, these models are of immense importance as someone participating systematically in the financial market needs to adapt their trading style and choice of instruments based on the market regimes to get a consistent performance. Choosing the right set of assets or sectors in a particular regime can provide significant alpha to a systematic investor. Similarly, for active traders, the optimal stop-loss limits and the choice of technical indicators can vary from one regime to another. There are three types of models that are popularly used:

C.3.1 THRESHOLD MODELS

In mathematical or statistical modeling a threshold model is any model where a threshold value, or set of threshold values, is used to distinguish ranges of values where the behavior predicted by the model varies in some important way. Certain types of regression model may include threshold effects.

Figure 3) NIFTY Index Closing Prices from 01-01-2011 to 01-01-2020 along with 200 Days Moving Average

As mentioned, **Figure 3** represents the NIFTY Index Closing Prices from 01-01-2011 to 01-01-2020 along with 200 Days Moving Average. An observed variable crossing a threshold triggers a regime shift. For example, the prices moving below the 200-day moving average trigger a 'bearish regime' or a downtrend.

C.3.2 PREDICTIVE MODELS

Predictive modeling uses statistics to predict outcomes. Most often the event one wants to predict is in the future, but predictive modelling can be applied to any type of unknown event, regardless of when it occurred. For example, predictive models are often used to detect crimes and identify suspects, after the crime has taken place. In many cases the model is chosen on the basis of detection theory to try to guess the probability of an outcome given a set amount of input data, for example given an email determining how likely that it is spam. Models can use one or more classifiers in trying to determine the probability of a set of data belonging to another set. Depending on definitional boundaries, predictive modelling is synonymous with, or largely overlapping with, the field of machine learning, as it is more commonly referred to in academic or research and development contexts. When deployed commercially, predictive modelling is often referred to as predictive analytics. Predictive modelling is often contrasted with causal modelling/analysis. In the former, one may be entirely satisfied to make use of indicators of, or proxies for, the outcome of interest. In the latter, one seeks to determine true cause-and-effect relationships. This distinction has given rise to a burgeoning literature in the fields of research methods and statistics and to the common statement that "correlation does not imply causation". Predictive modeling in trading is a modeling process wherein the probability of an outcome is predicted using a set of predictor variables. Predictive models can be built for different assets like stocks, futures, currencies, commodities etc. Predictive modeling is still extensively used by trading firms to devise strategies and trade. It utilizes mathematically advanced software to evaluate indicators on price, volume, open interest and other historical data, to discover repeatable patterns. Data scientists can use prediction methods like machine learning algorithms to take macroeconomic variables like GDP, unemployment, long term trends, bond yields, trade balance, etc. as input and predict the next period risk. When the model predicts a high-risk number, the market is in a risky regime. When the model predicts a low-risk number, the market is in a trending regime.

C.3.3 MARKOV SWITCHING AUTOREGRESSIVE MODELS

The steady-state equation of an asset price is defined as an autoregressive process. This is characterized by the intercept μ, autocorrelation β, and volatility σ, of the process specific to the market regime

$R_t = \mu + \beta \times R_t\text{-}1 + \sigma\epsilon$

The governing dynamics of the underlying regime, S_t, are assumed to follow a homogenous first-order Markov chain. Here, the probability of the next state depends only on the present state. We can estimate the transition probabilities from one such state to the next through a Maximum Likelihood Estimator. It attempts to find the parameter values that maximize the likelihood function.

C.4 BLACK LITTERMAN MODEL

In finance, the **Black–Litterman Model** is a mathematical model for portfolio allocation developed in 1990 at Goldman Sachs by Fischer Black and Robert Litterman, and published in 1992. It seeks to overcome problems that institutional investors have encountered in applying modern portfolio theory in practice. The model starts with an asset allocation based on the equilibrium assumption (assets will perform in the future as they have in the past) and then modifies that allocation by taking into account the opinion of the investor regarding future asset performance. Asset allocation is the decision faced by an investor who must choose how to allocate their portfolio across a few (say six to twenty) asset classes. For example, a globally invested pension fund must choose how much to allocate to each major country or region. In principle Modern Portfolio Theory (the mean-variance approach of Markowitz) offers a solution to this problem once the expected returns and covariances of the assets are known. While

Modern Portfolio Theory is an important theoretical advance, its application has universally encountered a problem: although the covariances of a few assets can be adequately estimated, it is difficult to come up with reasonable estimates of expected returns. Black–Litterman overcame this problem by not requiring the user to input estimates of expected return; instead it assumes that the initial expected returns are whatever is required so that the equilibrium asset allocation is equal to what we observe in the markets. The user is only required to state how his assumptions about expected returns differ from the markets and to state his degree of confidence in the alternative assumptions. From this, the Black–Litterman method computes the desired (mean-variance efficient) asset allocation. In general, when there are portfolio constraints - for example, when short sales are not allowed the easiest way to find the optimal portfolio is to use the Black–Litterman model to generate the expected returns for the assets, and then use a mean-variance optimizer to solve the constrained optimization problem.

2.2 METHODS OF QUANTIFYING DIVERSIFICATION

A EFFECTIVE NUMBER OF UNCORRELATED BETS

Here we review the Effective Number of Bets approach in [Meucci., 2009], using a notation more suitable for the generalizations to follow.
Consider an arbitrary portfolio which gives rise to a yet to be realized projected return R. In asset-based portfolio management, a portfolio is a combination of n correlated assets (stocks, options, bonds, futures, ...), and the portfolio return is a weighted average of the return of each asset:

$$R = \sum_{n=1}^{\bar{n}} w_n R_n$$

where w_n represents the weight of the n^{th} asset in the portfolio.

More in general, in factor-based portfolio management, and in factor-based risk budgeting and risk parity, a portfolio is a combination of k correlated factors, such as momentum, value, etc. Then, the portfolio return is a combination of the factor returns

$$R = \sum_{k=1}^{\bar{k}} b_k F_k, \tag{1}$$

where b_k represents the exposure of the portfolio to the k-th factor.

Typically, but not necessarily k (the number of factors) is much smaller than n (the number of assets). Furthermore, clearly, asset-based portfolio management represents a special case of factor-based management (1), where n = k, the factors are the asset returns F_n = Rn and the exposures are the portfolio weights $b_n = w_n$.
Let us assume for now an ideal, apparently non-realistic scenario, where we can express the portfolio return as a combination of k Bets, or uncorrelated factors.
To measure diversification via the Effective Number of Bets, we need to express our portfolio returns as a combination of uncorrelated terms

$$R = \sum_{k=1}^{\bar{k}} \mathring{b}_k \mathring{F}_k, \tag{2}$$

Then, we can compute the Diversification Distribution, namely true relative contributions to total risk from each bet

$$p_k \equiv \frac{\mathbb{V}\{\mathring{b}_k \mathring{F}_k\}}{\mathbb{V}\{R\}}, \qquad k = 1, \ldots, \bar{k}, \tag{3}$$

where V denotes the variance. Notice that, as for any distribution, the masses pk sum to one and are non-negative p_k

$$\sum_{k=1}^{\bar{k}} p_k = 1, \qquad p_k \geq 0, \quad k = 1, \ldots, \bar{k}, \tag{4}$$

The Diversification Distribution (3) provides a detailed picture of the portfolio concentration structure. A portfolio is well diversified among the k factors, and thus achieves risk parity, if the masses p_1,…., p_k are equal, or equivalently if the Diversification Distribution is uniform. To quantify diversification precisely, we use the exponential of the entropy, a tool from information theory, that measures the uniformity of a distribution. Accordingly, we define the Effective Number of Bets as follows

$$\mathbb{N} \equiv e^{-\sum_{k=1}^{\bar{k}} p_k \ln p_k}. \tag{5}$$

B PRINCIPAL COMPONENT ANALYSIS

Given a collection of points in two, three, or higher dimensional space, a "best fitting" line can be defined as one that minimizes the average squared distance from a point to the line. The next best-fitting line can be similarly chosen from directions perpendicular to the first. Repeating this process yields an orthogonal basis in which different individual dimensions of the data are uncorrelated. These basis vectors are called principal components, and several related procedures principal component analysis (PCA). PCA is mostly used as a tool in exploratory data analysis and for making predictive models. It is often used to visualize genetic distance and relatedness between populations. PCA is either done by singular value decomposition of a design matrix or by doing the following 2 steps:

- Calculating the data covariance (or correlation) matrix of the original data
- Performing eigenvalue decomposition on the covariance matrix

Usually the original data is normalized before performing the PCA. The normalization of each attribute consists of mean centering – subtracting each data value from its variable's measured mean so that its empirical mean (average) is zero. Some fields, in addition to normalizing the mean, do so for each variable's variance (to make it equal to 1); see z-scores. The results of a PCA are usually discussed in terms of component scores, sometimes called factor scores (the transformed variable values corresponding to a particular data point), and loadings (the weight by which each standardized original variable should be multiplied to get the component score). If component scores are standardized to unit variance, loadings must contain the data variance in them (and that is the magnitude of eigenvalues). If component scores are not standardized (therefore they contain the data variance) then loadings must be unit-scaled, ("normalized") and these weights are called eigenvectors; they are the cosines of orthogonal rotation of variables into principal components or back. PCA is the simplest of the true eigenvector-based multivariate analyses. Often, its operation can be thought of as revealing the internal structure of the data in a way that best explains the variance in the data. If a multivariate dataset is visualized as a set of coordinates in a high-dimensional data space (1 axis per variable), PCA can supply the user with a lower-dimensional picture, a projection of this object when viewed from its most informative viewpoint. This is done by using only the first few principal components so that the dimensionality of the transformed data is reduced. PCA is closely related to factor analysis. Factor analysis typically incorporates more domain specific assumptions about the underlying structure and solves eigenvectors of a slightly different matrix. PCA is also related to canonical correlation analysis (CCA). CCA defines coordinate systems that optimally describe the cross-covariance between two datasets while PCA defines a new orthogonal coordinate system that optimally describes variance in a single dataset. Robust and L1-norm-based variants of standard PCA have also been proposed. The principal components approach provides a formal set of uncorrelated factors from which to compute the Effective Number of Bets. However, it presents several problems.

- First, the principal components bets tend to be statistically unstable, especially those relative to the lowest eigenvalues.
- Second, the principal components bets are not invariant under simple scale transformations, as provided by a diagonal matrix d with positive entries.
- Third, the principal components bets are not unique. As discussed in [Deguest et al., 2013], if e_k is one of the k eigenvectors, so is its opposite e_k, and thus there are basically 2^k possible combinations of principal components bets.
- Fourth, the principal components bets are in general not easy to interpret, and hence disconnected from the decision process. In particular, in a dynamic setting, the meaning of the PCA factors changes from one date to another date, except possibly for the very first few factors.
- Fifth, the principal components bets give rise to counter-intuitive results.

C MINIMUM LINEAR TORSION BETS

The Effective Number of Bets approach to risk budgeting and risk parity builds on the uncorrelated decomposition (2). Hence, unlike the standard approach to risk parity based on marginal contributions to risk, the Effective Number of Bets approach highlights the contributions from truly separate sources of risk. However, if the uncorrelated portfolio decomposition (2) is achieved via the principal components bets, we obtain suboptimal results for the several reasons highlighted previously. Fortunately, the principal components bets are not the only zero-correlation transformation of the original factors F that allows to express the portfolio as in the uncorrelated decomposition (2). There exist several alternative factor rotations, or torsions F =t*F , of the original factors F, that are uncorrelated, and that are represented by a suitable k * k decorrelating torsion matrix t. For instance, one could think of independent component analysis, see [Back and Weigend, 1997], or more simply we could use the lower-triangular Cholesky decomposition. However, such transformations display the same problems as the principal component approach. Most notably, the resulting uncorrelated factors F are not interpretable, as in general they bear no relationship with the original factors F that are used to manage the portfolio. Therefore, we propose a natural, interpretable definition for the de-correlating transformation and the resulting uncorrelated factors: we choose the minimum torsion linear transformation that least disrupts the original factors F. More precisely, among all the torsions t that ensure that the new factors are uncorrelated, we select the Minimum-Torsion transformation, i.e. the transformation that minimizes the tracking error with respect to the original factors. Then, the Minimum-Torsion Bets and exposures allow us to compute the Minimum-Torsion Diversification Distribution and the Effective Number of Minimum-Torsion Bets, as follows

$$\boldsymbol{p}_{MT}(\boldsymbol{b}) = \frac{(\mathring{\boldsymbol{t}}_{MT}^{\prime -1}\boldsymbol{b}) \circ (\mathring{\boldsymbol{t}}_{MT}\boldsymbol{\Sigma}_F\boldsymbol{b})}{\boldsymbol{b}'\boldsymbol{\Sigma}_F\boldsymbol{b}} \quad \Rightarrow \quad \mathbb{N}_{MT}(\boldsymbol{b}) = e^{-\boldsymbol{p}_{MT}(\boldsymbol{b})'\ln \boldsymbol{p}_{MT}(\boldsymbol{b})}$$

The Minimum-Torsion Bets address all the problems of the principal components bets. In particular, the normalization in the tracking error allows us not to worry about non-homogenous factors F measured in completely different units, such as interest rates and implied volatilities. In other words, unlike in the case of uncorrelated bets defined by the principal components, for the Minimum-Torsion Bets the following diagram holds

$$\begin{tikzcd} \boldsymbol{F} \arrow[r, "\text{MT decorrelation}"] \arrow[d, "\text{scaling}"'] & \mathring{\boldsymbol{F}}_{MT} \arrow[d, "\text{scaling}"] \\ \boldsymbol{dF} \arrow[r, "\text{MT decorrelation}"] & (\mathring{\boldsymbol{dF}})_{MT} = \boldsymbol{d}\mathring{\boldsymbol{F}}_{MT} \end{tikzcd}$$

3 Outputs, Numerical Results & Graphical Illustrations

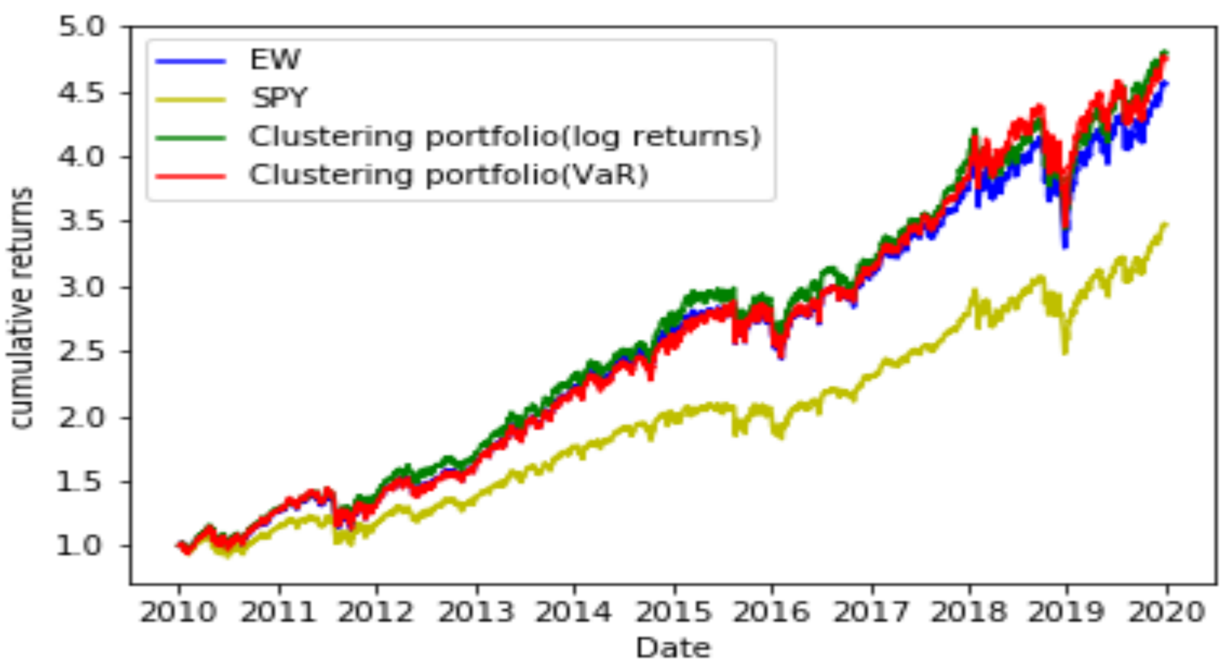


Figure 4) Graphical Plot Representing the Yearly Returns for the Various Types of Portfolios Constructed

It is evident from **Figure 4** that the Cumulative Returns obtained from the 3 constructed portfolios (i.e., the Equally Weighted Portfolio(EW), the Clustered Portfolio based on Log Returns and the Clustered Portfolio based on VaR [Value at Risk]) is much greater than the Cumulative Returns obtained from S&P 500 Index over a 10-year time frame. But on detailed analysis it was found that the more sophisticated algorithms as mentioned in [Cesarone; Colluci., 2018] should be used to get better diversified portfolios.

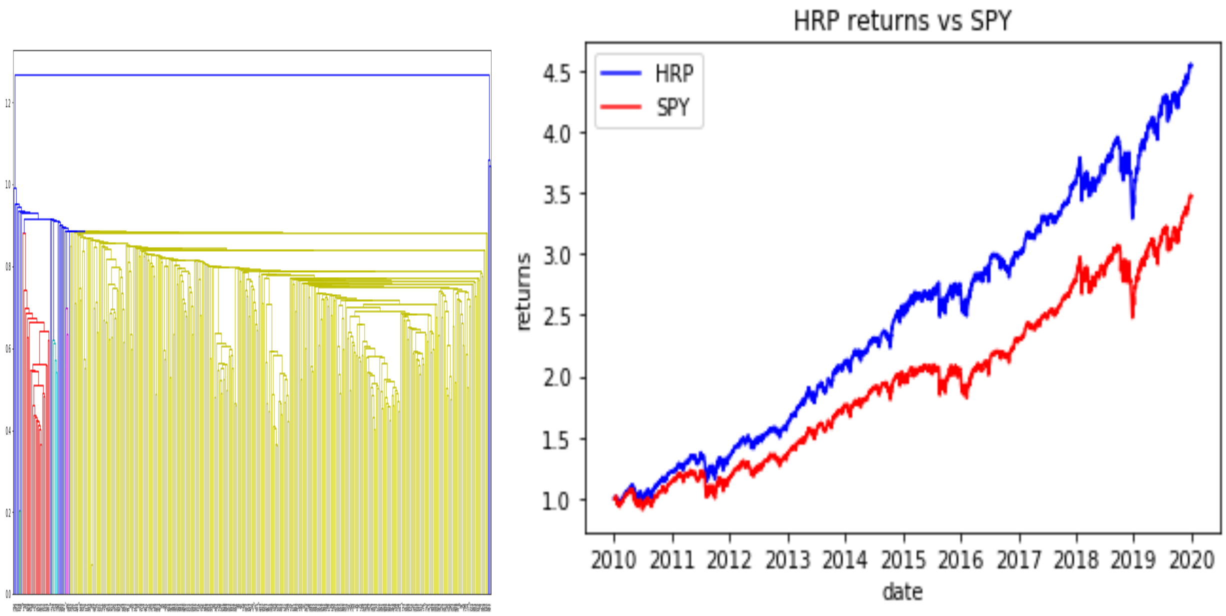


Figure 5) Graphical Plots representing the Yearly Returns for the Constructed HRP Portfolio vs S&P 500

It is evident from **Figure 5** that the Cumulative Returns obtained from the Hierarchical Risk Parity Portfolio (HRP) is much greater than the Cumulative Returns obtained from S&P 500 Index over a 10-year time frame. This algorithm specializes in giving better out of sample results and is more interpretable and fast.

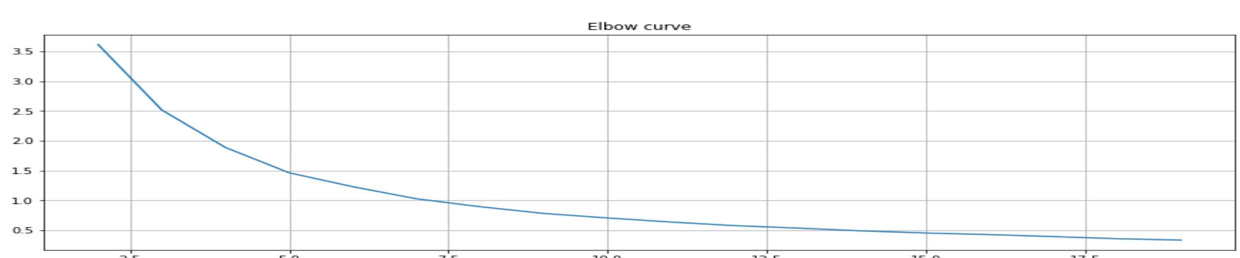


Figure 6) Elbow method shows the optimal number of cluster to be 7 for K-Means clustering

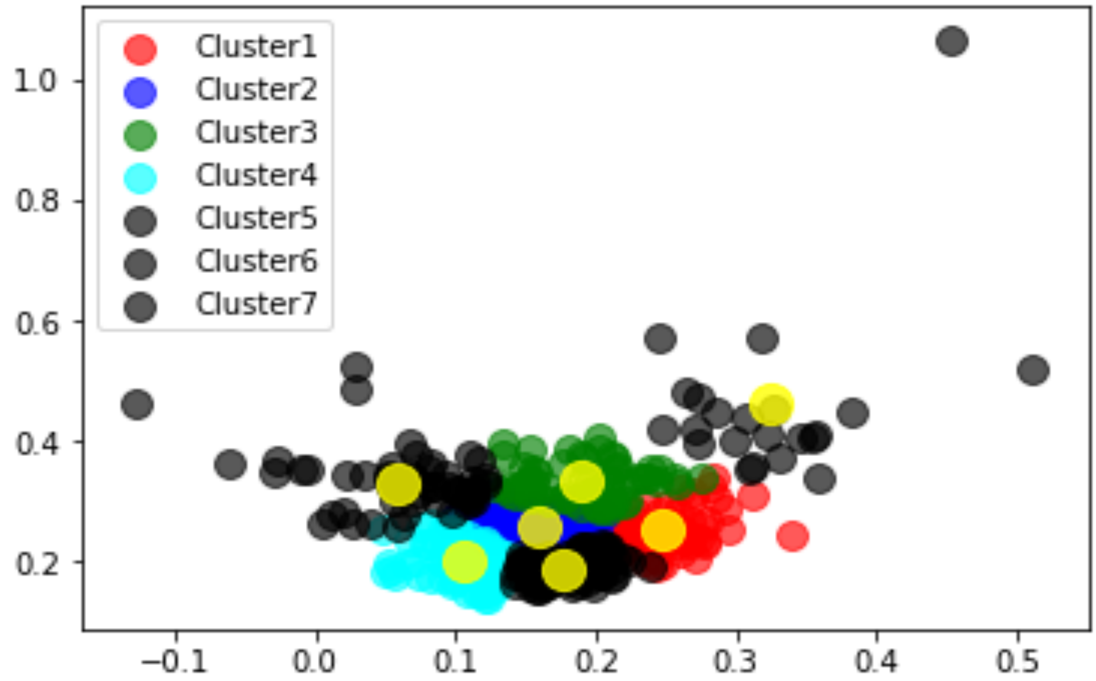


Figure 7) Output Plot obtained through the K-Means Cluster Analysis for the Sample Assets

Looking at **Figure 7**, we can infer the clusters formed. The yellow points are the centroid of the cluster and the asset closest to it has been used in our analysis

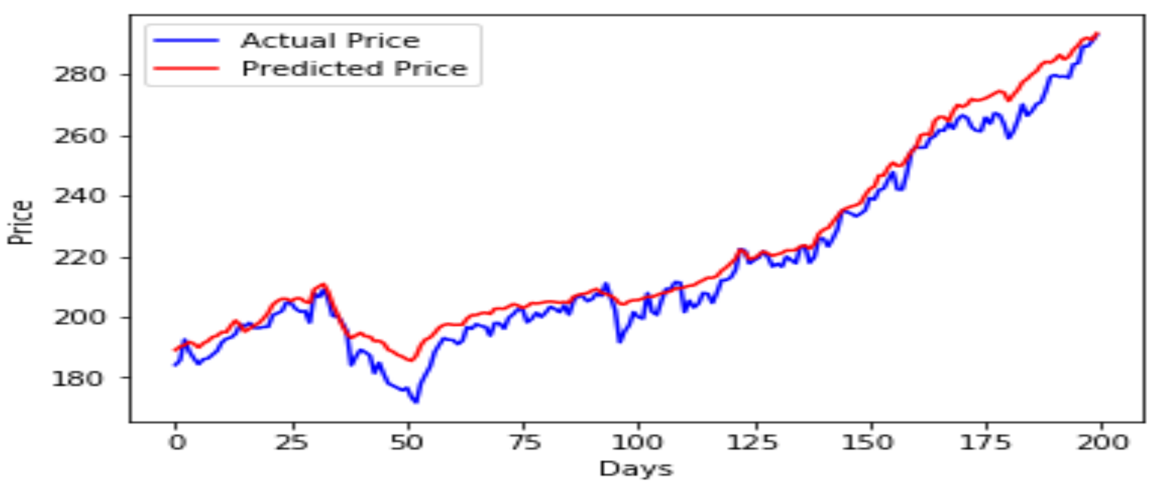


Figure 8a) Graphical Plot representing the Actual Prices along with the Predicted Prices obtained by employing RNN for the asset “APPLE”

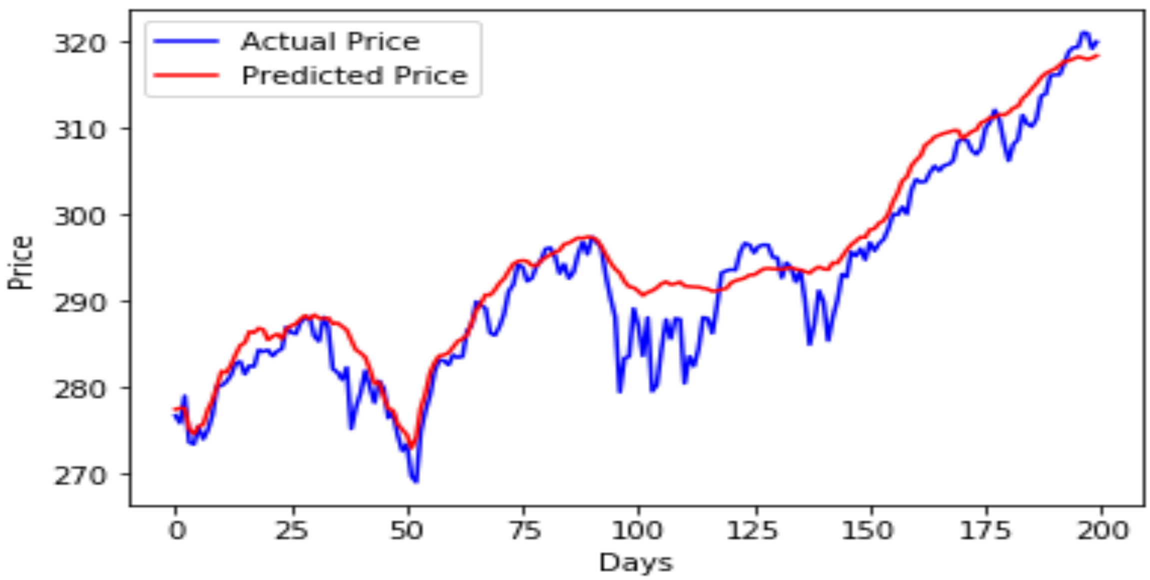


Figure 8b) Graphical Plot representing the Actual Prices along with the Predicted Prices obtained by employing RNN for the index "S&P 500".

After looking at **Figures 8a and 8b**, we can almost surely infer that by using Recurrent Neural Networks (RNN), it proves to be very useful in Time Series Prediction mainly because of the fact that RNN remembers each and every information over the respective time frame. Also, after completing each iteration i.e., epochs, the corresponding model alters and learns.

Index	Maximum Sharpe Portfolio	Maximum Quadratic Utility Portfolio	SPY
Returns	0.244694	0.614169	0.303
Volatility	0.205636	0.688899	0.125
Sharpe Ratio	1.09399	0.862491	2.358

Table 1) The above table shows the final values obtained through Portfolio Asset Allocation Optimization Using Deep Learning

Table 1 Overview

For the above type of portfolios, we used a universe of **423 Stocks** in a time period from **01-01-2010** to **01-01-2020**. K-means clustering was done on Yearly Data and RNN was done on a rolling window of 3 years to predict the future one year. ***Regime Probabilities*** were given as Input in the RNN algorithm. Two types of variation of Mean-variance optimization is used namely ***Maximum Sharpe Ratio*** and ***Maximum Quadratic Utility Portfolio***. It can be inferred that this portfolio construction is good at providing significant returns based on diversification and also ***optimal Sharpe Ratios*** in both types of constructed portfolios, keeping in mind that year 2019 was susceptible to ***Random Shocks*** in the stock market. Similarly, we can also find Minimum Volatility, Maximum Risk Aversion types of portfolio.

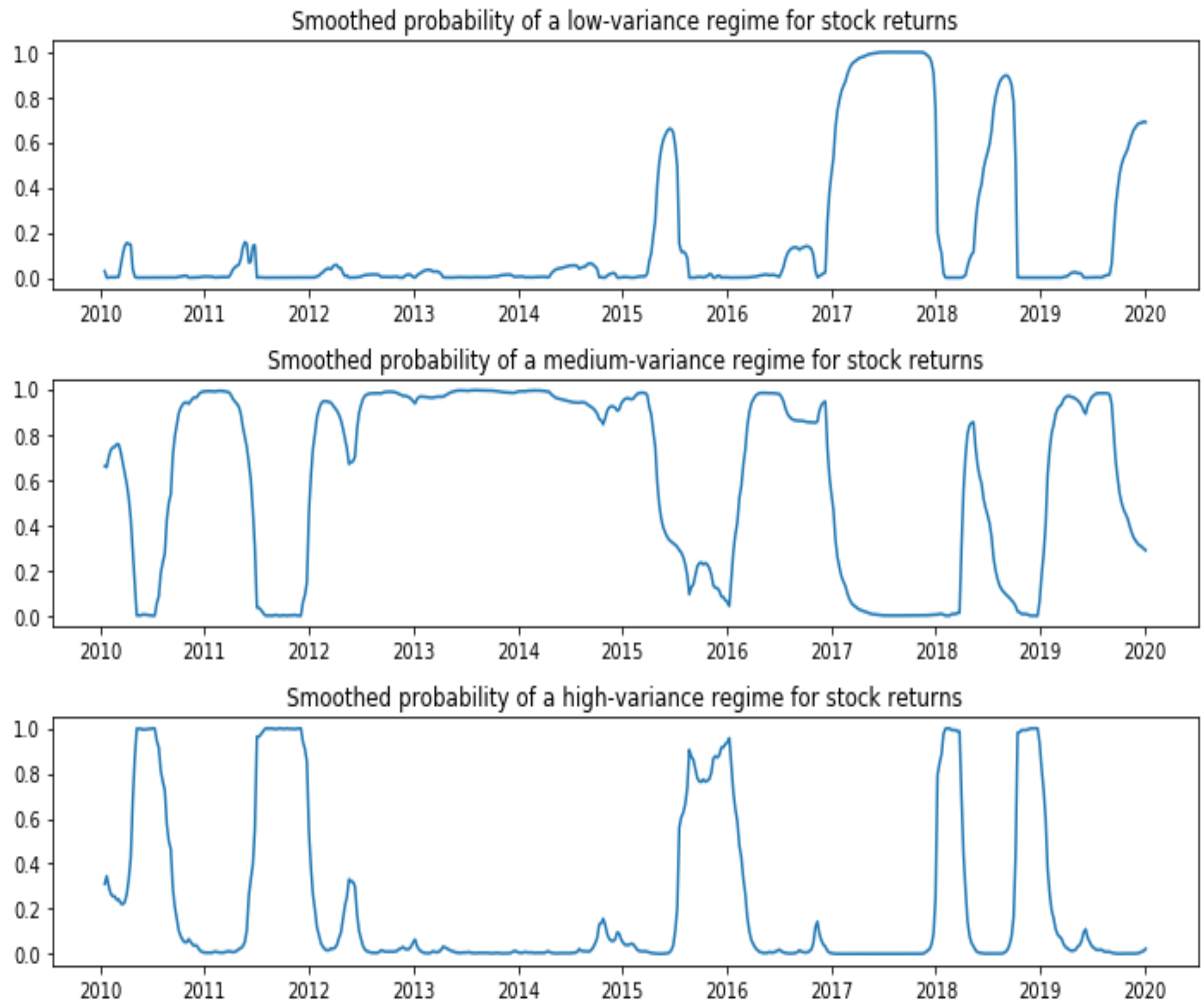


Figure 9) Output Plots obtained by employing Markov Switching Autoregressive Models

By looking at **Figure 9**, we can infer that depending on the various regimes pertaining to the risk criteria [i.e., low variance (low-risk), medium variance (medium-risk) and high variance (high-risk)], we obtain different Probability values thereby highlighting the purpose of Regime Shift Models

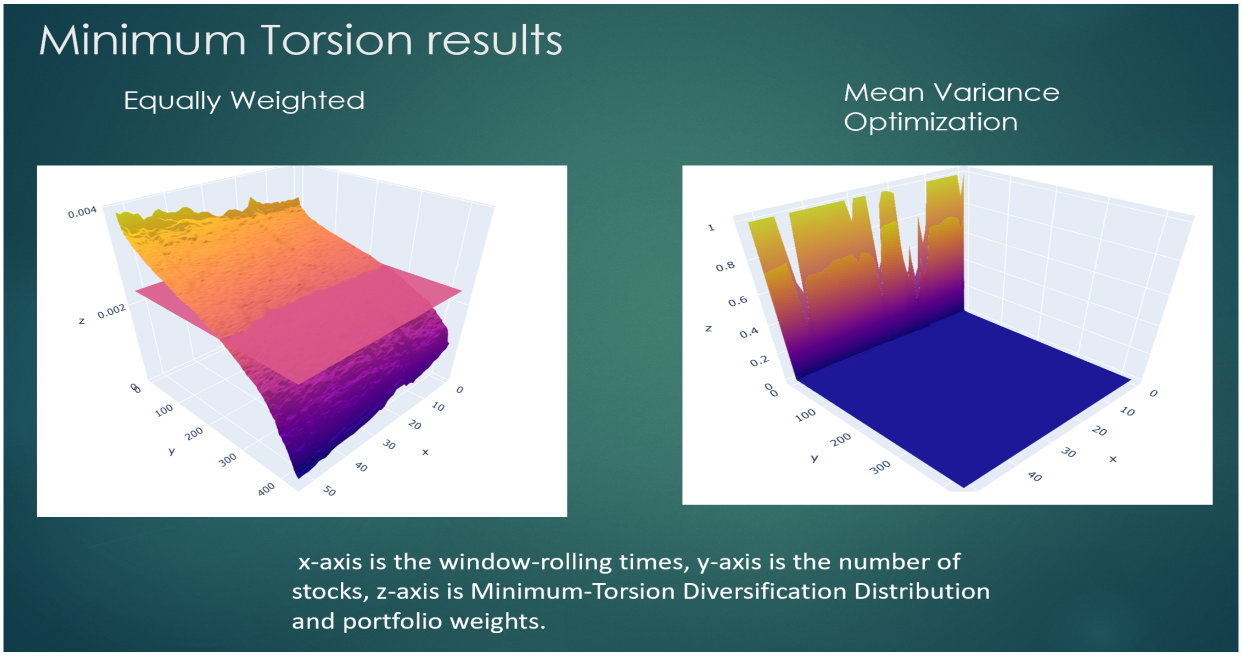


Figure 10a) 3-D Plot representations of Minimum Linear Torsion results in both Equally Weighted and Mean Variance Optimization Portfolios

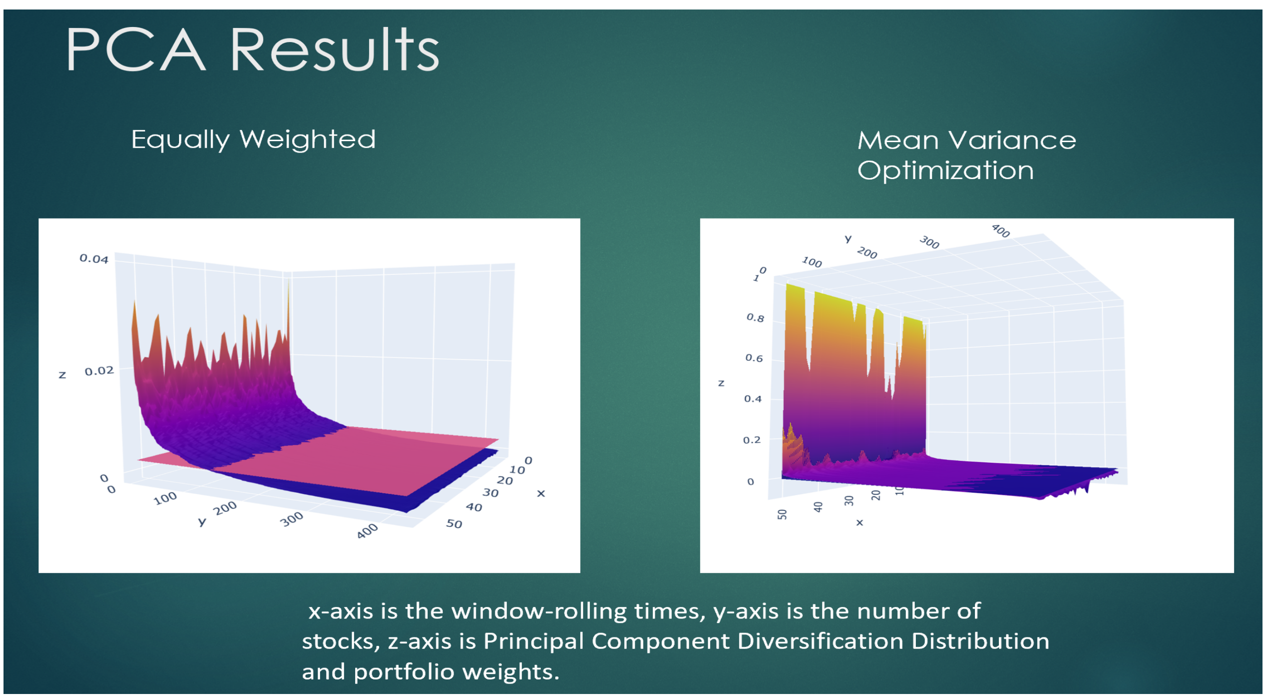


Figure 10b) 3-D Plot representations of Principal Component Analysis results in both Equally Weighted and Mean Variance Optimization Portfolio

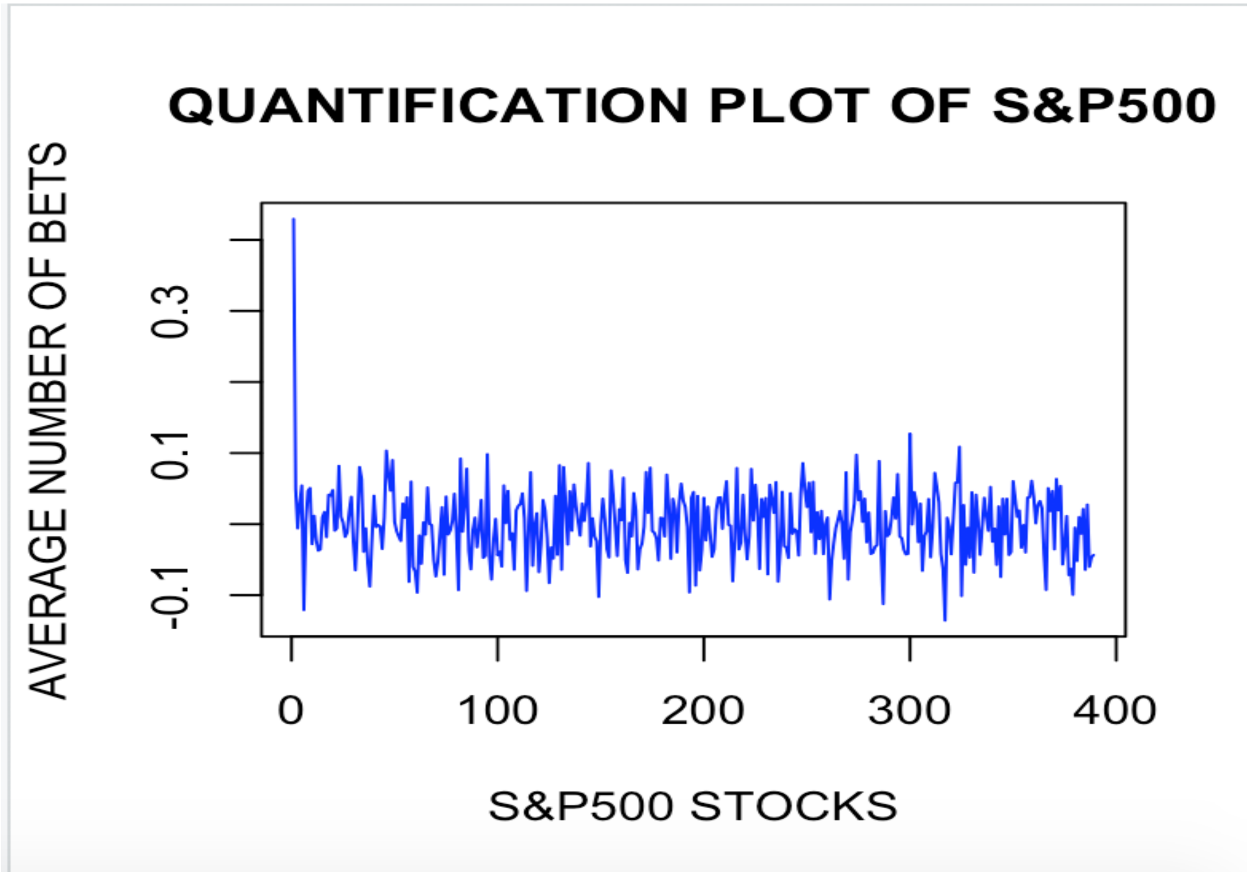


Figure 11) 2-D Plot Representing the Average Number of Bets for all the 393 Stocks pertaining to the S&P 500 Index

Figure 11 shows the Average Number of Bets value for all the 393 Stocks pertaining to the S&P 500 Index wherein the values range from **-0.15** to **+0.4.** In general, we can Normalize the results and have a range from 0 to +1 but, the reason as to why we have considered negative values is to highlight a potential investor if he/she has to go either **Underweight** or **Overweight** in the respective Asset

4 Conclusion

Quantification and Diversification are one of the most important concepts in the Financial World wherein, it is often said that Diversification is the only free lunch in Finance. We have successfully created various customizable portfolio techniques which aims at creating portfolio weights based on the risk appetite of investors. We have also constructed certain unique methodologies which helps in quantifying the diversification measures. Regarding future prospects, by employing certain alterations, our project can also be useful in Goal Based Investing wherein different portfolios are needed with respect to changing goals of a retail investor. Moreover, we have also highlighted and have also elaborated on certain Risk-Management techniques which can be employed for various purposes.

5 Appendix

A Risk-Management Techniques

Aaxj	Acim	AA	A	Abb	Abt	B	BB	Bas	C	CA	CC
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
50.722	58.8390	21.365	39.979	16.293	40.918	33.595	9.1350	1470.60	49.935	26.598	5.4028
50.821	58.1280	20.398	39.842	16.004	40.908	33.408	8.8600	1419.30	49.670	26.333	5.3435
49.833	57.5810	18.946	40.018	15.735	40.566	33.155	8.7400	1333.80	48.948	25.757	5.0477
48.444	55.9310	18.197	38.317	15.437	39.593	31.960	8.0000	1100.10	46.448	24.917	4.6830
48.041	56.5430	17.757	37.916	15.167	38.764	31.587	7.7100	1088.70	45.052	24.889	4.7323
48.071	56.1910	17.603	37.278	15.167	38.822	31.881	7.5150	1014.60	45.754	24.747	4.3180
48.189	55.9010	16.018	37.523	15.493	39.517	32.293	7.3800	1054.50	45.784	25.284	3.8548
47.631	55.2640	15.688	36.216	15.158	38.634	30.990	7.0800	1014.60	44.133	24.889	3.2534
48.180	56.3200	15.931	36.952	15.558	39.422	31.431	7.3200	1048.80	44.319	25.558	3.4605
46.340	54.6990	15.183	36.463	14.935	38.885	30.833	6.9850	986.10	41.478	25.057	3.9238
47.230	54.4900	14.831	36.550	15.121	38.779	30.617	6.7500	1105.80	40.960	24.653	3.5394
46.037	53.1750	14.831	36.608	14.963	38.328	30.726	6.9000	1083.00	39.543	24.398	3.5984
46.291	54.2390	15.601	36.579	14.963	37.867	30.225	6.8800	1259.70	39.211	24.717	3.5688
47.553	54.8560	15.117	37.315	15.381	38.395	30.882	7.0100	1322.40	40.100	25.247	3.7463
46.946	54.4830	14.963	36.952	15.177	38.040	30.100	6.8200	1179.90	38.624	24.794	3.4407
47.416	54.7930	15.710	36.856	15.502	38.520	30.941	7.0200	1162.80	39.552	25.360	3.2829
46.966	54.2770	15.314	36.403	15.716	38.817	30.696	6.8800	1145.70	39.612	26.191	3.0759
47.319	55.0490	15.404	35.480	15.743	35.210	30.754	6.8700	1214.10	39.494	26.633	3.0956
48.668	56.0080	16.041	36.991	16.088	36.305	31.832	7.1200	1311.00	41.635	27.137	3.8845
48.228	56.5140	15.865	37.031	16.013	36.880	31.528	7.0700	1282.50	41.539	26.882	3.8548
47.045	55.1900	15.337	36.423	15.586	36.333	30.518	6.9100	1174.20	39.524	26.098	3.6083
47.807	55.2880	16.685	36.540	16.023	36.467	30.568	7.1800	1191.30	39.465	26.323	3.9436
47.974	54.9190	18.364	36.766	16.395	36.458	30.774	7.4000	1214.10	39.885	26.588	4.3379
47.592	55.2600	17.944	35.411	16.367	35.882	30.333	7.0900	1174.20	38.976	26.750	4.1112

Table 2) Sample Dataset

I choose four models to predict one asset's price by the others using the data sample above. Here are the results that the numbers of variables in each model, so I choose BIC as the model.

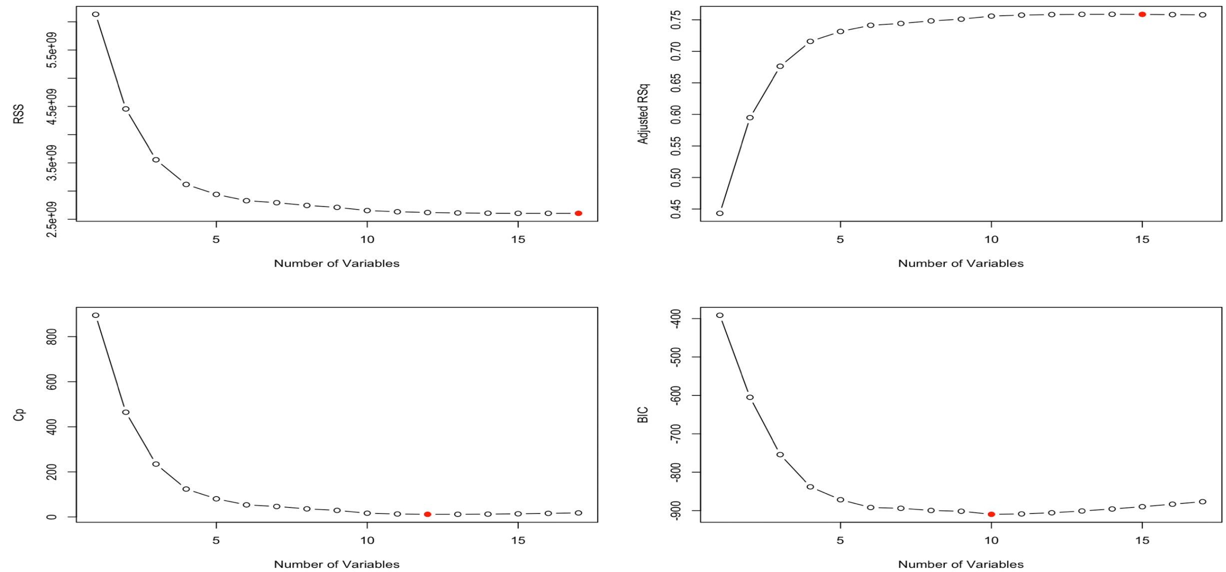
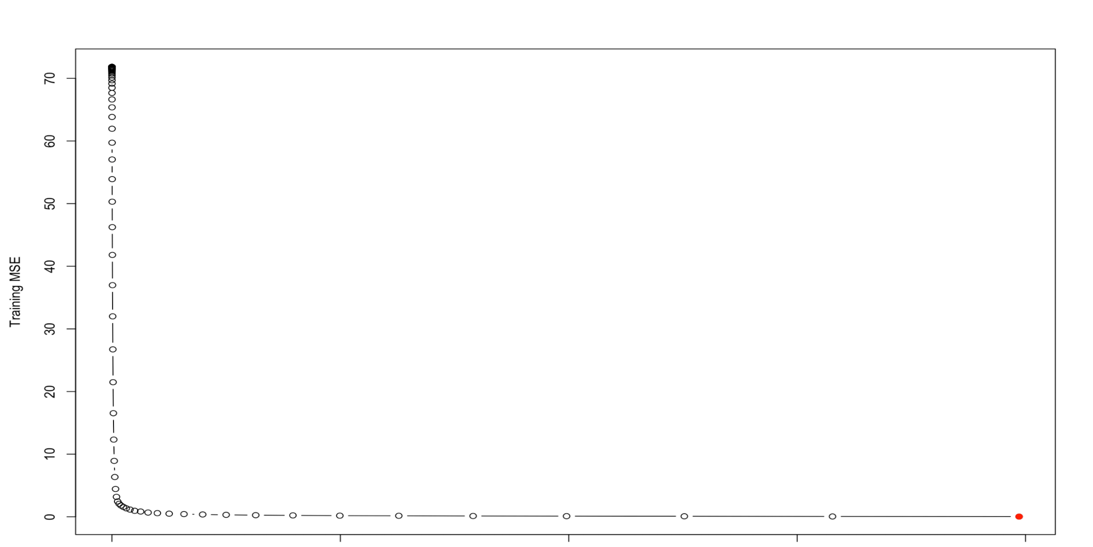


Figure 11a) results about the variables' number for four models

I split the data into training and testing sets by using boosting to predict the ETF. I perform boosting on the training set with 1000 trees for a range of values of the shrinkage parameter and produce two plots about training set MSE and testing set MSE with different parameters. Then I fit the model using liner regression and tree regression techniques and compare the MSE of those techniques to the results of these boosted trees.



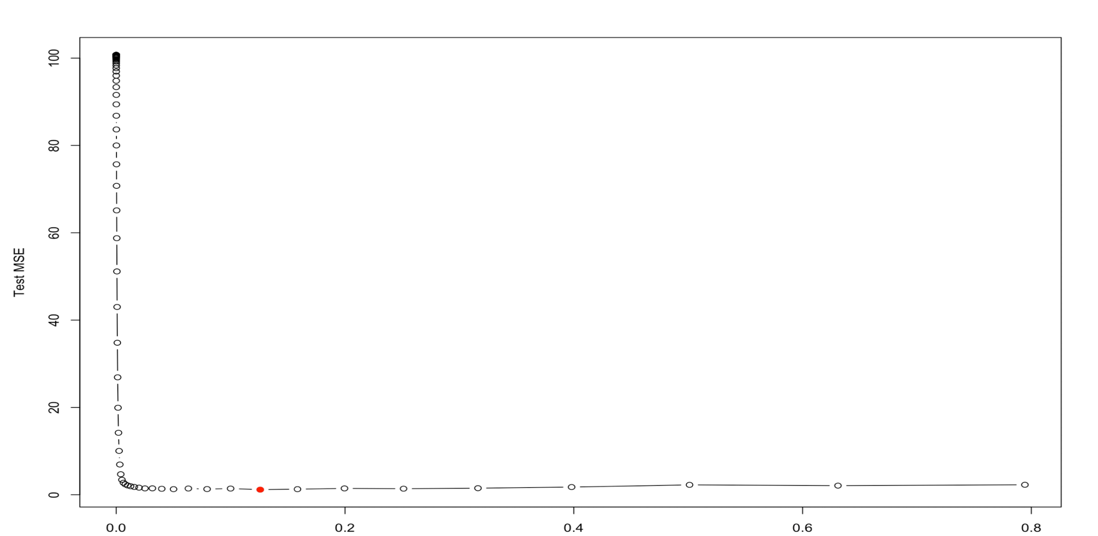
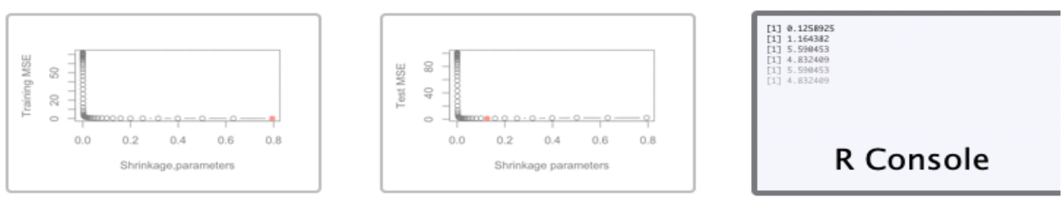


Figure 11b) charts about the MSE for test set and training set.




```
[1] 0.1258925
[1] 1.164382
[1] 5.590453
[1] 4.832409
[1] 5.590453
[1] 4.832409
```

Figure 11c) results about the MSE.

Here is part of the code and I find that the accuracy reaches 99 percent.

```
`{r}
TF2
= tune.svm(Acim_clust~Aaxj+AA+A+Abb+Abt+B+BB+Bas+C+CA+CC
               Acim_data,cost=c(0.001,0.01,0.1,1,5,10,100,
stSvm<-a$best.model
mmary(BestSvm)
```

```
ll:
est.svm(x = Acim_clust ~ Aaxj + AA + A + Abb + Abt + B +
   cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100, 1000), scal


arameters:
  SVM-Type:  eps-regression
SVM-Kernel:  radial
      cost:  1
     gamma:  0.09090909
   epsilon:  0.1


umber of Support Vectors:  390
```

````
```{r}
real <- Acim_data$Acim_clust
pred = predict(BestSvm,type='response',Acim_data[,c(-2,-13)])
prod =ifelse(pred>=0.5,1,0)
resul = data.frame(real,prod)
Acim_cof_matrix = table(resul)
Acim_cof_matrix
```
````

```
    prod
real   0   1
   0 271   0
   1   1 198
```

````
```{r}
n=length(real)
Acim_Accuracy=(Acim_cof_matrix[1,1]+Acim_cof_matrix[2,2])/n
Acim_Accuracy
```
````

```
[1] 0.9978723
```

````
```{r}
Acim_Recall=Acim_cof_matrix[2,2]/(Acim_cof_matrix[2,2]+Acim_cof_
Acim_Recall
```
````

```
[1] 0.9949749
```

Figure 12) part code of the predicting model and the accuracy result.

I am going to test the return of assets under the systematic risk. I use the principal component analysis and Linear weighted method as my methodology. These are the indicators I have choose from different periods.

name	meaning	name	meaning
GDP	the total monetary or market value of all the finished goods and services produced	Loan-deposit Ratio	Loan to deposit ratio
M2/M1	the rate between M2 to M1	IMPORT	Growth rate of import
TED	the difference between the interest rates on interbank loans and on short-term U.S. government debt	EXPORT	Growth rate of export
Stock Price	Average price earnings ratio of stock market	Exchange Rate	the value of RMB to Dollar
Real Estate Value	Growth rate of total value in real estate	Deficit Ratio	Rate between debt to GDP

Table 3) index I choose to do the test

I do the Bartlett's test to see if it is available to the factor analysis.

KMO and Bartlett's Test

KMO		0.741
Bartlett test of septicity	Approximate chi square	6863.740
	Freedom	378
	Saliency	0.000

Table 4) Bartlett's test results.

The results show that the KMO is 0.741, which is greater than 0.7. Bartlett's test results are significant, so it can be concluded that the selected series of indicators are suitable for factor analysis. In the next step, we use the Principal components analysis to find the index we need.

index	eigenvalue	Variance contribution	Total Variance contribution
1	9.984	35.657	35.657
2	4.422	15.794	51.451
3	2.808	10.029	61.480
4	2.066	7.380	68.860
5	1.950	6.963	75.824
6	1.435	5.125	80.948
7*	1.208	4.314	85.262
8	0.786	2.807	88.069
9	0.619	2.211	90.280
10	0.530	1.891	92.172

Table 5) the cumulative variance contribution of the factors.

According to the results of the chart, the cumulative variance contribution of the first seven factors reached 85.262%, which can basically reflect the expression of the original data. Next, we use Linear weighted method to calculate the financial systemic risk index as following function.

$$SRI = \frac{9.984 * F_1 + 4.422 * F_2 + 2.808 * F_3 + 2.066 * F_4 + 1.950 * F_5 + 1.435 * F_6 + 1.208 * F_7}{9.984 + 4.422 + 2.808 + 2.066 + 1.950 + 1.435 + 1.208}$$

Here are the results.

Feb-05	-0.4239	Jun-09	0.4621	Oct-13	0.3434
Mar-05	-0.6712	Jul-09	0.5863	Nov-13	0.3302
Apr-05	-0.6240	Aug-09	0.5525	Dec-13	0.4174
May-05	-0.5970	Sep-09	0.4184	Jan-14	0.2259
Jun-05	-0.5910	Oct-09	0.4305	Feb-14	0.2716
Jul-05	-0.5557	Nov-09	0.2081	Mar-14	0.2786
Aug-05	-0.6096	Dec-09	-0.1756	Apr-14	0.2077
Sep-05	-0.5696	Jan-10	-0.5364	May-14	0.1716
Oct-05	-0.5661	Feb-10	-0.5870	Jun-14	0.1793
Nov-05	-0.4617	Mar-10	-0.7207	Jul-14	0.2287
Dec-05	-0.4051	Apr-10	-0.6705	Aug-14	0.2642
Jan-06	-0.5827	May-10	-0.7843	Sep-14	0.2074
Feb-06	-0.6373	Jun-10	-0.6493	Oct-14	0.2297
Mar-06	-0.6815	Jul-10	-0.6315	Nov-14	0.3520
Apr-06	-0.6160	Aug-10	-0.6473	Dec-14	0.3992
May-06	-0.6152	Sep-10	-0.4934	Jan-15	0.4343
Jun-06	-0.5605	Oct-10	-0.4791	Feb-15	0.1957

Jul-06	-0.5162	Nov-10	-0.5849	Mar-15	0.5750
Aug-06	-0.6188	Dec-10	-0.2154	Apr-15	0.4964
Sep-06	-0.6072	Jan-11	-0.5228	May-15	0.4071
Oct-06	-0.5227	Feb-11	-0.3109	Jun-15	0.3704
Nov-06	-0.4617	Mar-11	-0.6994	Jul-15	0.4683
Dec-06	-0.2689	Apr-11	-0.5896	Aug-15	0.5456
Jan-07	-0.5025	May-11	-0.4457	Sep-15	0.6727
Feb-07	-0.4253	Jun-11	-0.1666	Oct-15	0.6479
Mar-07	-0.2796	Jul-11	-0.2147	Nov-15	0.6737
Apr-07	-0.2312	Aug-11	-0.3582	Dec-15	0.5456
May-07	-0.3725	Sep-11	-0.2094	Jan-16	0.7661
Jun-07	-0.3207	Oct-11	-0.1994	Feb-16	0.8696
Jul-07	-0.3410	Nov-11	-0.1332	Mar-16	0.6938
Aug-07	0.1452	Dec-11	-0.0225	Apr-16	0.7489
Sep-07	0.0786	Jan-12	0.2345	May-16	0.7038
Oct-07	0.0708	Feb-12	-0.2972	Jun-16	0.7934
Nov-07	-0.2393	Mar-12	-0.1169	Jul-16	0.7935
Dec-07	0.0785	Apr-12	0.0835	Aug-16	0.9169
Jan-08	-0.6046	May-12	-0.0827	Sep-16	0.7428
Feb-08	-0.5964	Jun-12	0.0494	Oct-16	0.7048
Mar-08	-0.8448	Jul-12	0.1832	Nov-16	0.6304
Apr-08	-0.6793	Aug-12	0.2427	Dec-16	0.7211
May-08	-0.8838	Sep-12	0.1855	Jan-17	0.4125
Jun-08	-0.7890	Oct-12	0.1703	Feb-17	0.4824
Jul-08	-0.8308	Nov-12	0.2430	Mar-17	0.4604
Aug-08	-0.6973	Dec-12	0.1462	Apr-17	0.4918
Sep-08	-0.6390	Jan-13	0.0787	May-17	0.5005
Oct-08	-0.4162	Feb-13	0.3775	Jun-17	0.4357
Nov-08	-0.1397	Mar-13	0.2339	Jul-17	0.4321
Dec-08	-0.1246	Apr-13	0.2333	Aug-17	0.4915
Jan-09	0.3968	May-13	0.3948	Sep-17	0.4590
Feb-09	0.4897	Jun-13	0.7329	Oct-17	0.4829
Mar-09	0.4363	Jul-13	0.3291	Nov-17	0.4249
Apr-09	0.4806	Aug-13	0.3213	Dec-17	0.4963
May-09	0.5015	Sep-13	0.3428		

Table 5) the result of the stress index.

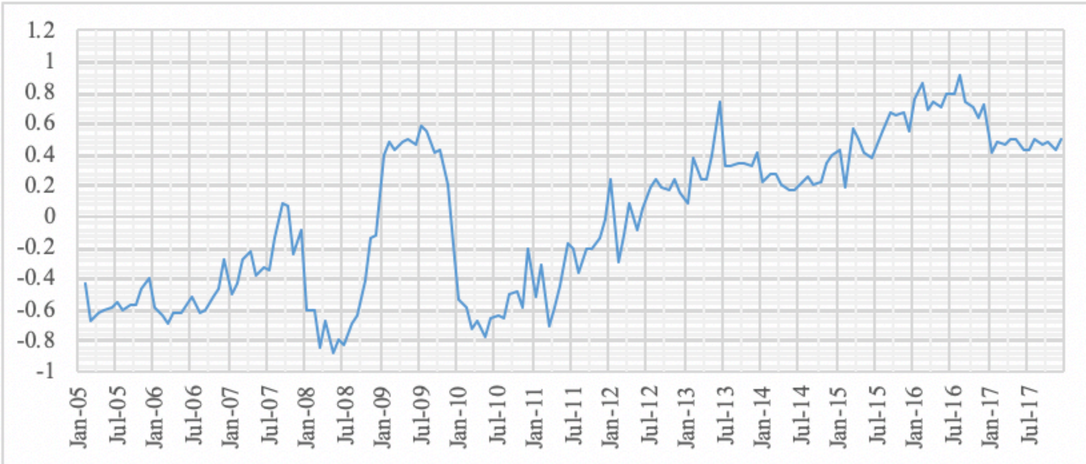


Figure 13) chart that the stress index change.

Finally, I do the regression between the risk index and the return of the important assets.

	Price Index return of variable	beta	t	Saliency	R^2
SRI	S&P 500	4.091	2.712	0.008	0.052
	USD/RMB	0.452	3.386	0.001	0.082
	Nonferrous Metals	5.827	2.524	0.014	0.044
	precious metal	5.543	2.228	0.028	0.033
	Agriculture products	3.253	1.723	0.088	0.017
	Commodity index	2.402	2.534	0.013	0.044

Table 6) the regression between stress index and the return of the assets.

According to the regression results in the above table, it can be concluded that the change of SRI can explain the change of the return rate of financial asset indexes such as stock index, foreign exchange, non-ferrous metals, precious metals and commodity index under 5% significance level. Under 10% significance level, SRI has influence on the yield of agricultural products index and the yield of bond index.

B R Code Listing

```
#INSTALL AND ACTIVATE THE RESPECTIVE LIBRARIES
library(BatchGetSymbols)
library(tseries)
library(zoo)
library(quantmod)
library(fBasics)
library(zoo)
library(PerformanceAnalytics)
library(tidyr)
library(reshape)
library(TTR)
library(BBmisc)
effective.bets <- function(b, Sigma, Tm)
  {

  ## this funciton computes the Effective Number of Bets and the Diversificat
ion distribution
  ## see A. Meucci, A. Santangelo, R. Deguest - "Measuring Portfolio Diversif
ication Based on Optimized Uncorrelated Factors" to appear (2013)
  ##
  ## Last version of code and article available at http://symmys.com/node/599

  ##
  ## INPUTS
  ##  b     : [vector] (n_ x 1) exposures
  ##  Sigma : [matrix] (n_ x n_) covariance matrix
  ##  t     : [matrix] (n_ x n_) torsion matrix
  ##
  ## OUTPUTS
  ##  enb : [scalar] Effetive Number of Bets
  ##  p   : [vector] (n_ x 1) diversification distribution

  p = mrdivide(mldivide(t(Tm),b) * (Tm %*% Sigma %*% b) , ( t(b) %*% Sigma %*
% b ) );
```

```
  enb = exp(-sum(p*log(1+(p-1)*(p>1e-5))));


  list(enb=enb,p=p)
}
torsion <- function(Sigma, model="pca", method="approximate", max_niter=10000
L) {


  ## this funciton computes the Principal Components torsion and the Minimum
Torsion for diversification analysis
  ## see A. Meucci, A. Santangelo, R. Deguest - "Measuring Portfolio Diversif
ication Based on Optimized Uncorrelated Factors" to appear (2013)


  ## Last version of code and article available at http://symmys.com/node/599


  ## INPUTS
  ##  Sigma     : [matrix] (n_ x n_) covariance matrix
  ##  model     : [string] choose between 'pca' and 'minimum-torsion' model
  ##  method    : [string] choose between 'approximate' and 'exact' method fo
r 'minimum-torsion' model
  ##  max_niter : [scalar] choose number of iterations of numerical algorithm
  ## OUTPUTS
  ## t : [matrix] (n_ x n_) torsion matrix


  if(model=="pca") {
    ## R already sorts these
    Tm <- prcomp(Sigma,center=FALSE,scale=FALSE)$rotation
    flip <- Tm[1,] < 0
    Tm[,flip] <- -Tm[,flip]


    ## PCA torsion, transpose
    Tm = t(Tm)
  } else if(model=='minimum-torsion') {
    ## alt
    rho <- cov2cor(Sigma)


    ## Correlation matrix
```

```
    sigma = diag(Sigma)^(1/2)
    rho = diag(1/sigma) %*% Sigma %*% diag(1/sigma)
    rho.norm = sqrtm(rho) ## Riccati root of C

    if(method=='approximate') {
      Tm = mrdivide(diag(sigma),rho.norm) %*% diag(1/sigma)
    } else if(method=='exact') {
      n_ = ncol(Sigma)

      ## initialize
      d = rep(1, n_)
      f = rep(0, max_niter)
      for(i in 1:max_niter) {
        U = diag(d) %*% rho.norm %*% rho.norm %*% diag(d)
        u = sqrtm(U)
        q = solve(u) %*% (diag(d) %*% rho.norm)
        d = diag(q %*% rho.norm)
        pi_ = diag(d) %*% q ## perturbation
        f[i] = norm(rho.norm - pi_, 'F')

        if (i > 1 && abs(f[i] - f[i-1])/f[i]/n_ <= 10^-8) {
          f = f[1:i]
          break
        } else if (i == max_niter && abs(f[max_niter] - f[max_niter-1])/f[max
_niter]/n_ > 10^-8) {
          warning("number of max iterations reached: n_iter = ", max_niter)
        }
      }
      ##cat(i)
      x = mrdivide(pi_,rho.norm)
      Tm = diag(sigma) %*% x %*% diag(1/sigma)
    }
  }
  Tm
}
```

```
SPY <- sp500hst

SPY <- SPY[,c(1:2, 6)]

#SPY.MODIFIED <- split(SPY, SPY$Ticker)

SPY.SYMBOLS <- unique(SPY[,2])

SPY.SYMBOLS <- as.character(SPY.SYMBOLS)

print(SPY.SYMBOLS)

SPY.MODIFIED1 <- SPY.MODIFIED[,c(2:779)]

SPY.MODIFIED1 <- SPY.MODIFIED1[c(T,F)]

SPY.MODIFIED1 <- cbind(SPY.MODIFIED[,c(1)], SPY.MODIFIED1)

#colnames(SPY.MODIFIED1) <- c(SPY.SYMBOLS2)

SPY.MODIFIED1 <-  xts(SPY.MODIFIED1[,-1], order.by=as.Date(SPY.MODIFIED1[,1],
"%m/%d/%Y"))

SPY.RETURNS <- diff(SPY.MODIFIED1, arithmetic=FALSE ) - 1

SPY.RETURNS <- SPY.RETURNS[-1,]

SPY.COVARIANCE <- cov(SPY.RETURNS)

SPY.COVARIANCE[is.na(SPY.COVARIANCE)] <- 0

TORSIONSPY.PCA <- torsion(SPY.COVARIANCE, model="pca", method="approximate",
max_niter=10000L)

TORSIONPCA.NORMALISED <- normalize(TORSIONSPY.PCA, method = "standardize", ra
nge = c(0, 1), margin = 1L, on.constant = "quiet")

TORSIONPCANORMALISED.MEAN <- colMeans(TORSIONPCA.NORMALISED)

TORSIONSPY.PCAMEAN <- colMeans(TORSIONSPY.PCA)

print(TORSIONPCANORMALISED.MEAN)

plot(1:389, TORSIONPCANORMALISED.MEAN , xlab = "S&P500 STOCKS", ylab = "AVERA
GE NUMBER OF BETS", type = "l", col = "blue", main = "QUANTIFICATION PLOT OF
S&P500")

TORSIONPCA.NORMALISED1 <- (TORSIONSPY.PCA[2][1:389] - colMins(TORSIONSPY.PCA[
1:389])) / (colMaxs(TORSIONSPY.PCA[1:389]) - colMins(TORSIONSPY.PCA[1:389]))
```

C Python Code Listing

Portfolio Asset Allocation Optimization Using Deep Learning:
https://github.com/sbhakuni/Quantitative-portfolio-construction-methods/blob/master/shaantanu_portfolio.ipynb

Hierarichal Risk Parity Portfolio Code: https://github.com/sbhakuni/Quantitative-portfolio-construction-methods/blob/master/HRP%20portfolio.ipynb

https://github.com/sbhakuni/Quantitative-portfolio-construction-methods/blob/master/ENB shaantanu.py

6 References

"Proverbial Baskets Are Uncorrelated Risk Factors! A Factor-Based Framework for Measuring and Managing Diversification in Multi-Asset Investment Solutions" by (Martellini, Lionel; Milhau, Vincent., 2017)

"Minimum Risk Versus Capital and Risk Diversification Strategies for Portfolio Construction" by (Francesco, Cesarone; Stefano, Colucci., 2017).

"Portfolio Asset Allocation Optimization with Deep Learning" by (Marcos Lopez de Prado., 2017)

"Measuring Portfolio Diversification based on Optimized Uncorrelated Factors" by (A. Meucci; A. Santangelo; R. Deguest., 2013)

"Financial Time Series Forecasting with Deep Learning: A Systematic Literature Review" by (Omer Berat Sezera; M. Ugur Gudeleka; Ahmet Murat Ozbayoglua., 2019)

"A First Application of Independent Component Analysis to Extracting Structure from Stock Returns" by (Andrew D. Back; Andreas S. Weigend.,1997)

Internet Sources: GitHub, Wikepedia, Investopedia, SSRN, Journal of Finance, Journal of Operations Research