ADVANCED OPTIMIZATION TECHNIQUES IN DYNAMIC PORTFOLIO STRATEGIES, PAIR TRADING,AND CARBON DIOXIDE EMISSION MODELING

by

JINHUI LI

A thesis submitted in conformity with
the requirements for the degree of
Doctor of Philosophy
Graduate Department of Mathematics
University of Toronto

ABSTRACT

This thesis explores advanced optimization modeling techniques with applications in three areas relevant to sustainable finance: dynamic portfolio construction through market state classification, pair trading strategies in financial markets, and carbon dioxide emission modeling. We develop a clustering algorithm for Bayesian Markov Switching Models to classify markets into volatility-based states, optimizing asset allocation strategies by identifying distinct market regimes. This approach incorporates equal-weighted investment, minimum variance, maximum diversification, and equal risk contribution strategies, providing evidence of superior risk-adjusted returns compared to static strategies. We introduce a Multi-modal Temporal Relation Graph Learning (MTRGL) framework that integrates time-series and categorical data through a dynamic graph and a memory-enhanced dynamic graph neural network to identify time-dependent correlations among financial instruments. This approach reframes pair trading as a temporal graph link prediction problem, outperforming traditional methods in empirical tests. We conducted a convergence analysis of the MTRGL framework to evaluate the convergence rate and complexity of the model, ensuring that it efficiently reaches stable solutions while demonstrating robustness in identifying optimal correlated pairs. Finally, we establish an optimization framework incorporating Support Vector Machine (SVM) regression and Principal Component Regression (PCR) to analyze socioeconomic and environmental factors affecting carbon dioxide emissions, refining emission models, and informing sustainability policies.

Our findings underscore the effectiveness of optimization modeling in forecasting carbon dioxide emissions, enhancing pair trading reliability, and optimizing investment strategies, thereby advancing environmental and quantitative finance.

Advanced Optimization Techniques in Dynamic Portfolio Strategies, Pair
Trading,and Carbon Dioxide Emission Modeling

Jinhui Li
Doctor of Philosophy
Graduate Department of Mathematics
University of Toronto

2024

To my parents and my sister,
for their endless love and support.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to those who have made this thesis possible.I am profoundly grateful to Dr. Luis Seco for their invaluable guidance, unwavering support, and mentorship throughout this research journey.I would like to extend my thanks to Prof. Vardan Papyan, Prof.Almut Burchard and Dr.Hamid Arian for their critical feedback and constructive suggestions. I would also like to thank University of Toronto for its supporting and financial assistance. I owe a debt of gratitude to my friends Clarence Su and Jack Ma for their encouragement and unwavering belief in me. To my family, especially my mom Liangzhen Ji and my dad Xudong Li, thank you for your love, patience, and understanding.To all those mentioned above and those who remain unnamed, your support has made this journey both possible and worthwhile.

PUBLICATIONS

Chapter 4 is based on my publication [1]

CONTENTS

1 INTRODUCTION WITH LITERATURE REVIEW

1.1 MARKET CLASSIFICATION AND DYNAMIC PORTFOLIO MANAGEMENT

The dynamic nature of financial markets poses significant challenges for investors. Traditional static asset allocation strategies often fail to adapt to changing market conditions, leading to suboptimal performance. Recent advances in financial research emphasize market classification and dynamic portfolio management to address this issue.

Market classification involves segmenting the market into distinct states based on characteristics such as volatility and returns. Methods such as K-means clustering and Markov switching models are commonly used for this purpose [2]. These models help identify different market regimes, enabling investors to adjust their strategies accordingly.

Markov switching models assume that the market can be in one of several states, with transitions governed by a Markov process. This framework allows estimating transition probabilities and forecasting future market states, providing valuable information for investment decisions [3].

Dynamic portfolio management leverages market classification to adjust asset allocations in real-time. Techniques such as mean variance optimization [4], equal risk contribution [5], minimum variance [6], and maximum diversification [7] have been extended to dynamic contexts, demonstrating improved performance.

This study focuses on dynamic investment strategies that use machine learning and statistical modeling to optimize portfolio management under varying market conditions. At the core of this research is the integration of Bayesian Markov Switching Models (BMSM) with traditional portfolio optimization techniques to adaptively allocate assets in response to market volatility.

K-Means Clustering for Market Segmentation

K-means clustering is utilized to classify the market into ten distinct volatility-based states. This unsupervised learning algorithm partitions a set of n observations into k clusters by minimizing the within-cluster sum of squares (WCSS). Mathematically, WCSS is defined as:

$$\text{WCSS} = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2, \tag{1.1}$$

where C_i is the set of observations in cluster i, and μ_i is the centroid of cluster i. The algorithm iteratively refines the cluster centroids until convergence, optimizing the segmentation of market states based on volatility.

Portfolio Optimization Methods

The study evaluates four classical portfolio optimization strategies:

EQUALLY-WEIGHTED INVESTMENT: Allocates equal weights to all assets. The weight w_i for each asset i is:

$$w_i = \frac{1}{n}, \tag{1.2}$$

where n is the total number of assets.

MINIMUM VARIANCE PORTFOLIO: Minimizes the portfolio's variance, defined as:

$$\sigma_p^2 = \mathbf{w}^T \Sigma \mathbf{w}, \tag{1.3}$$

subject to $\sum_{i=1}^{n} w_i = 1$ and $w_i \geq 0$.

MAXIMUM DIVERSIFICATION: Maximizes the diversification ratio:

$$\text{DR}(\mathbf{w}) = \frac{\sum_{i=1}^{N} w_i \sigma_i}{\sqrt{\mathbf{w}^T \Sigma \mathbf{w}}}, \tag{1.4}$$

where σ_i is the volatility of asset i.

EQUAL RISK CONTRIBUTION (ERC): Balances the risk contributions of assets:

$$\text{TRC}_i = w_i (\Sigma \mathbf{w})_i, \tag{1.5}$$

with the goal of equalizing total risk contributions across assets.

Dynamic Portfolio Strategy

The dynamic portfolio strategy leverages the Bayesian Markov transition matrix to optimize asset allocations by predicting market state transitions. The transition matrix P is constructed using Bayesian inference, where each element P_{ij} represents the probability of transitioning from state i to state j.

BAYESIAN ESTIMATION OF TRANSITION PROBABILITIES: The transition probabilities are estimated using a Bayesian approach, which incorporates prior beliefs about the state transitions and updates these beliefs based on observed data. The Dirichlet distribution is used as the conjugate prior for the multinomial transition probabilities, allowing the estimation process to incorporate both prior information and new observations effectively.

$$P_{ij} \sim \text{Dirichlet}(\alpha_{ij} + N_{ij}), \tag{1.6}$$

where α_{ij} are hyperparameters representing prior beliefs about the transitions, and N_{ij} are the observed counts of transitions from state i to state j.

GIBBS SAMPLING FOR POSTERIOR INFERENCE: Gibbs sampling is employed to generate samples from the posterior distribution of the transition probabilities. This iterative process allows for the estimation of the transition matrix by sequentially sampling each element P_{ij} conditioned on the current values of the other elements:

$$P_{ij}^{(t+1)} \mid \cdots \sim \text{Dirichlet}(\alpha_{ij} + N_{ij} + \sum_{k \neq i,j} P_{ik}^{(t)}), \tag{1.7}$$

ensuring convergence to the true posterior distribution over iterations.

DYNAMIC ASSET ALLOCATION: The dynamic asset allocation process involves using the estimated transition matrix to adjust portfolio weights based on the predicted probabilities of market state transitions. The asset allocation vector $\mathbf{w}(t)$ at time t is determined by:

$$\mathbf{w}(t) = P(t) \cdot \mathbf{w}_{\text{previous}}, \tag{1.8}$$

where $\mathbf{w}_{\text{previous}}$ is the asset allocation from the previous time period and $p(t)$ is the transition matrix. This process allows the portfolio to adapt dynamically to changing market conditions, optimizing the expected returns and risk profile based on the probabilistic forecasts provided by the Bayesian Markov model.

By incorporating Bayesian inference, the dynamic portfolio strategy achieves a more responsive and informed approach to asset allocation, leveraging probabilistic insights into market dynamics to enhance investment performance.

1.2 PAIR TRADING

Pair trading is a market-neutral trading strategy that aims to capitalize on the relationship between two related financial instruments, such as stocks, commodities, or currencies. The basic premise is to go long on one asset and short on another, assuming that the relationship between the two will revert to its historical mean over time. In recent years, the integration of Artificial Intelligence (AI) into the financial sector has revolutionized traditional trading strategies, including pair trading. Using machine learning algorithms, traders and financial institutions can now more accurately identify and execute pair trading opportunities with improved precision and efficiency.

The first meaningful academic paper on pair trading was written by Evan Gatev, William N. Goetzmann, and K. Geert Rouwenhorst. [8] In their paper "Pairs Trading: Performance of a Relative Value Arbitrage Rule", the authors test the profitability of a Wall Street investment strategy called pairs trading, which involves matching stocks into pairs based on their historical price movements and trading them based on the expectation that prices will converge. The authors find that pair trading yields average annualized excess returns of up to 11 percent for self-financing portfolios of pairs, which exceed conservative transaction cost estimates. Profits are attributed to temporary mispricing of close substitutes and are not caused by simple mean reversion. The paper also explores the risk characteristics of pairs trading and finds that the strategy is low exposed to systematic risk factors. The results suggest that pair trading is a profitable investment strategy that can be used to exploit temporary mispricing in the market.

Daniel Herlemont in his well-known paper "Pairs Trading, Convergence Trading, Cointegration" discusses pairs trading and cointegration in financial assets [9] It introduces the concept of mean reversion and explains how pairs trading can be used to take advantage of deviations from the historical mean. This paper also discusses optimal convergence trading and the use of the Ornstein-Uhlenbeck process to model mean reversion. This explains the Dickey Fuller test and its variants for testing the stationarity of variables. The document also mentions the variance ratio test and the concept of error correction models. In general, the document provides an overview of the different techniques and tests used in pairs trading and cointegration analysis.

Lastly, since artificial intelligence is significantly changing the financial industry, pair trading with machine learning is very popular nowadays. Han et al. [10] introduce CREDIT, a novel reinforcement learning-based approach to pair trading, a financial strategy aimed at mitigating market risks by trading two correlated assets. Traditional reinforcement learning methods face challenges in this domain due to the complexity of the trading environment and the need for long-term reasoning. CREDIT addresses these issues by incorporating a bidirectional Gated Recurrent Unit (GRU) and a temporal attention mechanism, enabling the model to capture long-term asset price patterns effectively. In addition, the method features a unique risk-aware reward system that balances both profit and associated trading risks. Empirical tests reveal that CREDIT outperforms existing methods, demonstrating significant profits over a five-year period using US stock data.

Temporal Graph Neural Networks (TGNN)

Temporal Graph Neural Networks (TGNNs) are advanced neural network architectures designed to manage dynamic graphs where both the structure and features evolve over time. This makes them particularly effective for applications in financial markets, such as pair trading, where temporal dependencies and the evolution of relationships are crucial.

MEMORY-BASED TGNNS (MTGNNS) incorporate a memory module to maintain the historical context of node interactions. This is achieved through components like the encoder, which processes event-based messages and updates memory states using a Gated Recurrent Unit (GRU), and the decoder, which predicts temporal links. The message module computes messages for nodes in an event $e_{ij}(t)$ as follows:

$$m_i(t) = \mathrm{msg}(s_i(t^-), s_j(t^-), e_{ij}(t), \psi(t - t_i')), \tag{1.9}$$

$$m_j(t) = \mathrm{msg}(s_j(t^-), s_i(t^-), e_{ij}(t), \psi(t - t_j')), \tag{1.10}$$

where $\psi(\cdot)$ is a time encoding function. The memory update is given by:

$$s_i(t) = \mathrm{mem}(m_i(t), s_i(t^-)). \tag{1.11}$$

The embedding module aggregates neighborhood information through:

$$z_i^{(l)} = \mathrm{mlp}^{(l)}(z_i^{(l-1)} \,||\, \tilde{z}_k^{(l)}), \tag{1.12}$$

$$\tilde{z}_k^{(l)} = \mathrm{mha}^{(l)}(q_i^{(l)}, K_i^{(l)}, V_i^{(l)}). \tag{1.13}$$

Temporal link prediction is performed by the decoder:

$$\hat{p}_{ij}(t) = \sigma(\mathrm{MLP}(z_i(t^-) \,||\, z_j(t^-))). \tag{1.14}$$

Training employs a binary cross-entropy loss combined with contrastive learning:

$$L = -\sum_{e_{ij}(t) \in E} \left[\log \hat{p}_{ij}(t) + \log(1 - \hat{p}_{ik}(t)) \right]. \tag{1.15}$$

Contrastive Learning

Contrastive learning is a self-supervised approach that enhances the extraction of meaningful representations by contrasting positive and negative samples. The core idea is to learn an embedding space where positive pairs (similar samples) are closer together and negative pairs (dissimilar samples) are further apart.

Mathematically, given a set of samples, we define positive pairs (x, x^+) and negative pairs (x, x^-). The goal is to minimize the distance between positive pairs and maximize the distance between negative pairs in the embedding space. This can be formulated using a contrastive loss function, such as the InfoNCE loss:

$$L = -\log \frac{\exp(\mathrm{sim}(h_x, h_{x^+})/\tau)}{\sum_{x^-} \exp(\mathrm{sim}(h_x, h_{x^-})/\tau)}, \tag{1.16}$$

where $\mathrm{sim}(h_x, h_{x'})$ is a similarity function (e.g., cosine similarity) between embeddings h_x and $h_{x'}$, and τ is a temperature parameter that controls the smoothness of the distribution.

The objective is to optimize the encoder to produce embeddings that maintain these relationships, which helps in capturing the intrinsic structures of the data. In TGNN, this is particularly useful for learning robust node representations that capture temporal dependencies.

Contrastive learning enhances the robustness and generalization of the learned embeddings, providing superior feature extraction capabilities. By contrasting samples, the model becomes more adept at identifying subtle but significant features that distinguish different states or behaviors in the data.

Introduction of Our Model

The proposed Multi-modal Temporal Relation Graph Learning (MTRGL) framework integrates high-dimensional feature data with time series data to identify temporal correlations among financial entities. Key features include dynamic graph construction, which constructs graphs by segmenting time series into intervals, with entities represented as nodes. The memory-based dynamic graph neural network captures historical interactions through a memory module, while contrastive learning distinguishes meaningful patterns using positive and negative samples.

MTRGL adapts to changing market conditions and improves the prediction accuracy of pair-trading opportunities, utilizing both feature and temporal data. The integration of TGNNs and contrastive learning within this framework provides a powerful tool for understanding and exploiting temporal correlations in financial markets. A detailed convergence analysis of the MTRGL framework was conducted to assess its convergence rate and computational complexity. The analysis confirms that the model efficiently stabilizes to optimal solutions, even in high-dimensional and dynamic environments, ensuring reliable identification of trading opportunities.

1.3 carbon dioxide emission forecast models

In the battle against climate change, accurately modeling carbon dioxide emissions is crucial for shaping effective policies and advancing sustainable development. This study leverages the analytical capabilities of Support Vector Regression (SVR) and Principal Component Regression (PCR) to examine the impact of key socioeconomic and environmental factors on carbon dioxide emissions using a dataset spanning from 1992 to 2019 across 62 countries. Our data, sourced from the World Bank [11] and NationMaster [12], provides a comprehensive basis for investigating the dynamics of global emissions.

Previous studies have explored a variety of machine learning techniques for carbon emission modeling. Kavoosi et al. [13] utilized a genetic algorithm (GA) for forecasting emissions. Sun [14] employed an optimized grey forecasting model for China's emissions, while Abdel [15] developed an artificial neural network (ANN) model to predict carbon emissions. Additionally, Kaboli et al. [16] estimated energy usage using adaptive neuro-fuzzy inference systems (ANFIS), support vector regression (SVR), and other methods. These studies highlight the diverse approaches available for tackling the complexities of emission forecasting.

Our methodology begins with extensive data preprocessing to ensure the integrity and compatibility of the dataset with the SVR and PCR algorithms. This involves standardizing the data to a uniform scale and verifying the stationarity of each component, thereby setting the stage for accurate analysis.

A critical aspect of our methodology is the hyperparameter tuning of the SVR model, which involves selecting the optimal kernel function, regularization parameter C, and kernel-specific parameters to enhance model accuracy and adaptability. Mathematically, SVR aims to minimize the following objective function:

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \max(0, |y_i - (\mathbf{w} \cdot \mathbf{x}_i + b)| - \epsilon), \tag{1.17}$$

where $\mathbf{w}$ is the weight vector, b is the bias term, ϵ is the margin of tolerance, and C is the regularization parameter.

Principal Component Regression (PCR) is employed to reduce dimensionality and multicollinearity among predictor variables. PCR utilizes Principal Component Analysis (PCA) to transform the original variables into a set of uncorrelated principal components. The PCA process involves solving the eigenvalue problem for the covariance matrix Σ of the data:

$$\Sigma\mathbf{v}_i = \lambda_i\mathbf{v}_i, \tag{1.18}$$

where $\mathbf{v}_i$ are the eigenvectors (principal components) and λ_i are the eigenvalues. The principal components are ordered by the magnitude of their eigenvalues, with larger eigenvalues corresponding to components that explain more variance. The PCR model is constructed as:

$$Y = \mathbf{Z} \cdot \boldsymbol{\beta} + \epsilon, \tag{1.19}$$

where Y is the dependent variable, $\mathbf{Z}$ represents the principal components, $\boldsymbol{\beta}$ is the coefficient vector, and ϵ is the error term.

To evaluate the models, we utilize Permutation Importance, a technique that quantifies the impact of each feature on model predictions. The permutation importance PI_j of a feature j is calculated as:

$$PI_j = \frac{1}{N}\sum_{i=1}^{N} \left(\text{error}_{i,\text{original}} - \text{error}_{i,\text{permuted}}\right), \tag{1.20}$$

where $\text{error}_{i,\text{original}}$ is the model's prediction error using the original dataset, and $\text{error}_{i,\text{permuted}}$ is the error after randomly permuting the values of feature j. This technique helps identify the most influential factors affecting emissions.

Our study assesses the precision of SVR and PCR models by comparing predicted emissions with actual figures and provides insights into the relative influence of various factors. This approach not only refines carbon dioxide volume predictions but also equips policymakers with actionable insights for designing effective emission reduction strategies.

In the context of machine learning applications for carbon emission modeling, studies such as those by Mehdizadeh and Movagharnejad [17] have demonstrated the accuracy of SVR compared to semi-empirical models. Lu et al. [18] applied neural networks for transportation-related emissions, while Wang et al. [19] highlighted SVM's efficacy in time-series predictions. Our study contributes to this growing body of literature by demonstrating the effectiveness of SVR and PCR in providing stable and reliable forecasts.

By blending sophisticated SVR and PCR techniques with an in-depth analysis of a broad spectrum of influencing factors, this research aims to illuminate the intricacies of carbon emissions, fostering an enriched understanding that supports the journey toward global sustainability.

2 PRELIMINARIES

In all of my work, many machine learning algorithms have been used. The core mathematics of these algorithms are linear optimization and probability theory, for example, support vector regression, stochastic gradient descent, Markov chain, etc. I will elaborate on these concepts and the mathematics behind them.

2.1 SUPPORT VECTOR REGRESSION

Support Vector Machine (SVM) is a powerful supervised learning algorithm that aims to find the optimal hyperplane in a high-dimensional feature space. It can be used for classification, regression, and outlier detection.

Training points beyond the margin contribute little to the cost function for Support Vector Classification (SVC), while samples whose prediction is close to their target are ignored by the cost function for Support Vector Regression (SVR). Although the logic is different, they lead to a similar result: the prediction results depend only on a subset of the training data.

SVR can handle both linear and non-linear relationships, making it well suited for predicting carbon volume based on multiple factors. By maximizing the margin between the hyperplane and the support vectors, SVR seeks to find the best fit for the data, ensuring accurate predictions.

The primary objective of SVR is to find a function $f(x)$ that approximates the target variable y as closely as possible. The function $f(x)$ is defined as

$$f(x) = \langle w, x \rangle + b, \tag{2.1}$$

where $\langle w, x \rangle$ is the dot product between the weight vector w and the feature vector x, and b is the bias term.

SVR uses an ϵ-insensitive loss function, which means the errors within a certain margin are ignored. The loss function L is defined as:

$$L(y, f(x)) = \max(0, |y - f(x)| - \epsilon). \tag{2.2}$$

The optimization problem in SVR is to minimize the following objective function:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \tag{2.3}$$

subject to the constraints:

$$y_i - \langle w, x_i \rangle - b \leq \epsilon + \xi_i, \tag{2.4}$$

$$\langle w, x_i \rangle + b - y_i \leq \epsilon + \xi_i^*, \tag{2.5}$$

$$\xi_i, \xi_i^* \geq 0. \tag{2.6}$$

SVR can also be extended to solve nonlinear problems by applying the "kernel trick," which involves mapping the input features into a higher-dimensional space. The kernel function $K(x, x')$ replaces the dot product $\langle x, x' \rangle$ in the optimization problem.

Commonly used kernel functions include:

- Linear: $K(x, x') = \langle x, x' \rangle$,
- Polynomial: $K(x, x') = (1 + \langle x, x' \rangle)^d$,
- Radial Basis Function (RBF): $K(x, x') = \exp(-\gamma||x - x'||^2)$.

To solve this constrained optimization problem, we introduce Lagrange multipliers $\alpha_i, \alpha_i^*, \beta_i, \beta_i^*$ and form the Lagrangian:

$$\begin{aligned}\mathcal{L}(w, b, \xi, \xi^*, \alpha, \alpha^*, \beta, \beta^*) = {} & \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \\ & - \sum_{i=1}^{n}\alpha_i(\epsilon + \xi_i - y_i + \langle w, x_i \rangle + b) \\ & - \sum_{i=1}^{n}\alpha_i^*(\epsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b) \\ & - \sum_{i=1}^{n}\beta_i\xi_i - \sum_{i=1}^{n}\beta_i^*\xi_i^*. \end{aligned} \tag{2.7}$$

To find the optimal w and b, we set the derivatives of the Lagrangian with respect to w, b, ξ, and ξ^* to zero:

$$\frac{\partial \mathcal{L}}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)x_i, \tag{2.8}$$

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0, \tag{2.9}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \Rightarrow C - \alpha_i - \beta_i = 0, \tag{2.10}$$

$$\frac{\partial \mathcal{L}}{\partial \xi_i^*} = 0 \Rightarrow C - \alpha_i^* - \beta_i^* = 0. \tag{2.11}$$

Thus in the dual form, particularly when using kernel methods, the formula can be expressed as:

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) K(x_i, x) + b, \tag{2.12}$$

where one can solve the bias term b by any support vector x_i with

$$b = y_i - \sum_{j=1}^{n} (\alpha_j - \alpha_j^*) K(x_i, x_j). \tag{2.13}$$

2.2 stochastic gradient descent

Stochastic Gradient Descent (SGD) is an optimization algorithm commonly used in machine learning and deep learning to minimize the loss function. Unlike traditional Gradient Descent, which uses the entire dataset to compute the gradient at each iteration, SGD randomly selects a subset of the data at each step. This makes SGD faster and more suitable for large-scale datasets.

Let $J(\theta)$ be the objective function (often called the loss function in machine learning) that we want to minimize. The function is defined as:

$$J(\theta) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \theta)), \tag{2.14}$$

where N is the number of data points, L is the loss for a single data point, y_i is the true label, x_i is the feature vector, and θ are the parameters we want to optimize.

Gradient Descent Update Rule

In traditional Gradient Descent, the update rule for the parameters θ is:

$$\theta_{\text{new}} = \theta_{\text{old}} - \alpha \nabla J(\theta), \tag{2.15}$$

where α is the learning rate, and $\nabla J(\theta)$ is the gradient of the loss function with respect to θ.

Stochastic Gradient Descent Update Rule

In Stochastic Gradient Descent, instead of using the entire dataset to compute $\nabla J(\theta)$, we randomly select a single data point (x_i, y_i) or a mini-batch of data points to estimate the gradient. The update rule becomes:

$$\theta_{\text{new}} = \theta_{\text{old}} - \alpha \nabla L(y_i, f(x_i; \theta)). \tag{2.16}$$

Advantages and Disadvantages

- **Advantages**: Faster convergence per iteration, ability to escape local minima for nonconvex functions, and suitability for large-scale datasets.
- **Disadvantages**: More noise in the gradient estimation, which may lead to oscillations and instability.

Hyperparameters

- **Learning Rate (α)**: Controls the step size in the parameter space.
- **Batch Size**: The number of samples used to estimate the gradient in each iteration.

2.3 PERMUTATION IMPORTANCE

To determine the relative importance of factors, we employ the importance of permutation as a feature selection technique. Permutation Importance measures the impact of permuting the values of each feature on the model's performance. By ranking the features through this process, we gain insights into the factors that have the most significant impact on the volume of carbon dioxide. This combined approach of SVR and Permutation Importance allows us to make accurate predictions while identifying the key drivers behind carbon emissions.

Given a predictive model M trained on tabular data D consisting of n features, and the performance score S of model M evaluated on dataset D.

Theorem 2.1. *For each feature j in the dataset D, the importance value I_j is determined by the difference in the performance of the model when the feature j is used normally versus when the values of the feature j are randomly permuted. This importance value is calculated as follows:*

$$I_j = S - \frac{1}{L} \sum_{l=1}^{L} S_j^l, \tag{2.17}$$

where S_j^l is the performance score of model M on dataset D_j^l, which is derived by randomly shuffling the values of feature j in D for the l-th permutation, and L is the total number of permutations.

Interpretation: The importance value I_j quantifies the contribution of feature j to the predictive accuracy of model M. A positive value of I_j indicates that the predictive accuracy decreases when the values of feature j are permuted, suggesting that feature j holds predictive power within model M. Conversely, a non-positive I_j suggests that feature j does not contribute meaningfully to the model's performance, or the model is insensitive to the order of data points in feature j.

2.4 principal component analysis (pca)

Principal Component Analysis (PCA) is a foundational technique in statistical analysis, introduced by Karl Pearson in 1901, and later developed independently by Harold Hotelling in the 1930s. PCA serves as a powerful tool for dimensionality reduction, data compression, and feature extraction. By transforming a set of possibly correlated variables into a set of linearly uncorrelated variables called principal components, PCA simplifies the complexity of high-dimensional datasets while preserving their essential structures.

Objectives of PCA

1. **Dimensionality Reduction**: PCA reduces the number of variables in a dataset while retaining the maximum amount of variability present in the original data. This is achieved by identifying the main components that capture the most significant patterns in the data.

2. **Data Compression**: By transforming the data into a smaller set of principal components, PCA effectively compresses the data, reducing storage requirements, and enhancing computational efficiency.

3. **Feature Extraction**: PCA identifies the most informative features (principal components) that explain the largest variance in the data, facilitating better insights and more effective modeling.

Mathematical Foundation

PCA is rooted in linear algebra and involves several key mathematical steps:

1. **Standardization**: Given a dataset $\mathbf{X}$ with n observations and p variables, the data is standardized to have a mean of zero and a standard deviation of one:
$$\mathbf{X}_{\text{scaled}} = \frac{\mathbf{X} - \mu_{\mathbf{X}}}{\sigma_{\mathbf{X}}}, \tag{2.18}$$
where $\mu_{\mathbf{X}}$ is the mean vector and $\sigma_{\mathbf{X}}$ is the standard deviation vector of $\mathbf{X}$.

2. **Covariance Matrix Calculation**: The covariance matrix $\mathbf{C}$ of the standardized data is computed:
$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}_{\text{scaled}}^{\top}\mathbf{X}_{\text{scaled}}, \tag{2.19}$$
The covariance matrix captures the pairwise covariances between the variables, reflecting how each variable varies with respect to the others.

3. **Eigen Decomposition**: The covariance matrix $\mathbf{C}$ is decomposed into its eigenvalues λ_i and eigenvectors $\mathbf{e}_i$:
$$\mathbf{C}\mathbf{e}_i = \lambda_i \mathbf{e}_i, \tag{2.20}$$
The eigenvectors $\mathbf{e}_i$ (principal components) represent the directions of maximum variance in the data, while the eigenvalues λ_i indicate the magnitude of the variance along these directions.

4. **Principal Components**: The original data is projected onto the principal components to form the principal component scores $\mathbf{Z}$:
$$\mathbf{Z} = \mathbf{X}_{\text{scaled}}\mathbf{E}, \tag{2.21}$$
where $\mathbf{E} = [\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_p]$ is the matrix of eigenvectors. Each row of $\mathbf{Z}$ represents the coordinates of the original data point in the new principal component space.

5. **Selection of Principal Components**: To achieve dimensionality reduction, a subset of the principal components is selected based on the explained variance. The explained cumulative variance $\sum_{i=1}^{k} \lambda_i / \sum_{i=1}^{p} \lambda_i$ is used to determine the number of principal components k that capture a specified proportion (for example, 90%) of the total variance.

Interpretation of Principal Components

Each principal component is a linear combination of the original variables, with coefficients known as loadings:

$$\mathrm{PC}_i = a_{i1}X_1 + a_{i2}X_2 + \cdots + a_{ip}X_p, \tag{2.22}$$

where a_{ij} are the elements of the eigenvector $\mathbf{e}_i$. The loadings indicate the contribution of each original variable to the principal component. A high absolute value of a loading implies a strong influence of the corresponding variable on the principal component. Positive loadings indicate a direct relationship, while negative loadings indicate an inverse relationship.

Applications of PCA

PCA is utilized in various domains, including:

- **Finance**: For portfolio optimization, risk management, and asset pricing by identifying uncorrelated factors that influence market movements.
- **Biology**: For analyzing gene expression data, image processing, and ecological studies to uncover patterns and structures in high-dimensional biological data.
- **Engineering**: For fault detection, signal processing, and control systems to simplify complex system behavior.
- **Social Sciences**: For survey data analysis, psychology, and market research to identify underlying factors that explain responses and behaviors.

Advantages of PCA

- **Reduction of Overfitting**: By reducing the number of dimensions, PCA helps mitigate overfitting in predictive models, leading to more robust and generalizable results.
- **Improved Visualization**: PCA enables the visualization of high-dimensional data in two or three dimensions, making it easier to identify patterns, clusters, and outliers.
- **Enhanced Computational Efficiency**: Reducing the number of dimensions decreases the computational complexity of subsequent

analyses, making PCA a valuable preprocessing step for machine learning algorithms.

- **Reduction of Collinearity**: PCA transforms correlated variables into uncorrelated principal components as the eigenvectors are orthogonal, effectively reducing multicollinearity in the data.

In summary, PCA is a fundamental technique for simplifying complex datasets while preserving their essential characteristics. By transforming correlated variables into a set of uncorrelated principal components, PCA provides a clear and concise representation of the data, facilitating deeper insights and more effective modeling. Incorporating PCA into your research methodology can significantly enhance the interpretability and performance of your analyses.

2.5 MARKOV MODELS AND CONVERGENCE

Markov models are fundamental tools in stochastic processes, widely used for modeling systems that transition between states probabilistically. These models are characterized by the memoryless property, where the future state depends only on the current state, not on the sequence of events that preceded it.

Markov Chains and Markov Matrices

A Markov chain is a sequence of random variables $\{X_t\}$ that transition between states in a state space $\{1, 2, \ldots, K\}$ with the Markov property:

$$\Pr(X_{t+1} = j \mid X_t = i, X_{t-1}, \ldots, X_0) = \Pr(X_{t+1} = j \mid X_t = i). \tag{2.23}$$

The transitions are governed by a Markov matrix $P = [P_{ij}]$, where P_{ij} represents the probability of transitioning from state i to state j:

$$P_{ij} = \Pr(X_{t+1} = j \mid X_t = i). \tag{2.24}$$

The state distribution vector $\pi(t)$ evolves as:

$$\pi(t+1) = \pi(t)P. \tag{2.25}$$

A steady-state distribution π satisfies:

$$\pi = \pi P, \tag{2.26}$$

which means π is the eigenvector of P corresponding to the eigenvalue 1.

Markov Switching Models

Markov switching models (MSMs) extend Markov chains by allowing the observed process to switch between different regimes or states. Each state has its own distinct characteristics, often modeled by different statistical distributions.

An MSM consists of an observed time series $\{y_t\}$ and an unobserved state variable $\{s_t\}$ that follows a Markov chain. The state variable s_t takes values in $\{1, 2, \ldots, K\}$, where K is the number of regimes. The observed series y_t depends on the state s_t:

$$y_t \mid s_t = k \sim f(y_t \mid \theta_k), \tag{2.27}$$

where θ_k are the parameters associated with state k.

The transition probabilities of the state variable are given by the Markov matrix:

$$\Pr(s_{t+1} = j \mid s_t = i) = P_{ij}. \tag{2.28}$$

In mathematical terms, the joint distribution of the observed data and the state sequence is:

$$\Pr(y_1, \ldots, y_T, s_1, \ldots, s_T) = \Pr(s_1) \prod_{t=2}^{T} \Pr(s_t \mid s_{t-1}) \prod_{t=1}^{T} \Pr(y_t \mid s_t). \tag{2.29}$$

Bayesian Markov Switching Models

Bayesian Markov Switching Models (BMSMs) integrate Bayesian inference into the Markov Switching framework, providing a probabilistic approach to parameter estimation and state prediction. The BMSM is defined by the following components:

PRIOR DISTRIBUTIONS: Prior distributions are assigned to the model parameters, capturing our initial beliefs before observing the data. For example, the transition probabilities P_{ij} can be assigned a Dirichlet prior, and the state-specific parameters θ_k can be given appropriate prior distributions:

$$P_{ij} \sim \text{Dirichlet}(\alpha_{ij}), \quad \theta_k \sim p(\theta_k \mid \eta_k), \tag{2.30}$$

where P_{ij} represents the probability of transitioning from state i to state j, and $\text{Dirichlet}(\alpha_{ij})$ reflects our prior belief about these probabilities with parameters α_{ij}. The state-specific parameters θ_k, which may define characteristics like means and variances in each state, have prior distributions conditioned on hyperparameters η_k.

The Dirichlet distribution is parameterized by $\alpha = (\alpha_1, \ldots, \alpha_K)$ and has the following mean and variance:

$$\text{Mean:} \quad \frac{\alpha_i}{\sum_{k=1}^{K} \alpha_k}, \tag{2.31}$$

$$\text{Variance:} \quad \frac{\alpha_i(\sum_{k=1}^{K} \alpha_k - \alpha_i)}{(\sum_{k=1}^{K} \alpha_k)^2(\sum_{k=1}^{K} \alpha_k + 1)}. \tag{2.32}$$

LIKELIHOOD: The likelihood function captures the probability of the observed data given the states and model parameters. For a given state sequence $\{s_t\}$, the likelihood is:

$$\Pr(y_1, \ldots, y_T \mid s_1, \ldots, s_T, \theta) = \prod_{t=1}^{T} f(y_t \mid \theta_{s_t}), \tag{2.33}$$

where y_t is the observed data at time t, and $f(y_t \mid \theta_{s_t})$ is the likelihood of observing y_t given the state-specific parameters θ_{s_t}.

POSTERIOR DISTRIBUTION: Using Bayes' theorem, the posterior distribution of the parameters and states is obtained by combining the priors and the likelihood:

$$\begin{aligned}\Pr(\theta, s_1, \ldots, s_T \mid y_1, \ldots, y_T) \propto{}& \Pr(y_1, \ldots, y_T \mid s_1, \ldots, s_T, \theta) \\ &\times \Pr(s_1, \ldots, s_T \mid P) \Pr(P) \Pr(\theta),\end{aligned} \tag{2.34}$$

where $\Pr(P)$ is the prior probability of the transition matrix P, reflecting beliefs about state transitions, and $\Pr(\theta)$ is the prior distribution over state-specific parameters θ, encoding prior knowledge about these parameters.

GIBBS SAMPLING: Gibbs sampling, a Markov Chain Monte Carlo (MCMC) method, is commonly used to sample from the posterior distribution. This involves iteratively sampling the state sequence and model parameters from their conditional posterior distributions.

- Sample the state sequence s_t given the data and parameters:

$$\Pr(s_t \mid y_1, \ldots, y_T, \theta, P) \propto \Pr(y_t \mid \theta_{s_t}) \Pr(s_t \mid s_{t-1}, P). \tag{2.35}$$

- Sample the parameters θ_k given the states and data:

$$\Pr(\theta_k \mid s_t = k, y_1, \ldots, y_T) \propto \prod_{t: s_t = k} f(y_t \mid \theta_k) p(\theta_k \mid \eta_k). \tag{2.36}$$

- Sample the transition probabilities P_{ij} given the state sequence:

  $$P_{ij} \mid s_1, \ldots, s_T \sim \text{Dirichlet}(\alpha_{ij} + n_{ij}), \tag{2.37}$$

  where n_{ij} is the number of transitions from state i to state j.

Gibbs sampling effectively generates samples from the posterior distribution, particularly when the full posterior is complex. The samples can then be used to approximate the posterior mean, variance, and other statistical properties.

ADVANTAGES OF BAYESIAN MARKOV SWITCHING MODELS: Bayesian Markov Switching Models offer several significant advantages over traditional approaches. Firstly, they facilitate the incorporation of prior knowledge through prior distributions, allowing researchers to integrate existing expertise or empirical evidence into the model. This integration can substantially enhance estimation accuracy, particularly in scenarios where data is sparse or limited.

Moreover, Bayesian inference provides a comprehensive framework for quantifying parameter uncertainty. By generating credible intervals for model parameters, Bayesian methods offer a nuanced measure of uncertainty, which is often absent in frequentist approaches. This aspect of Bayesian analysis ensures that estimates are accompanied by a quantifiable level of confidence, thereby enhancing the robustness of the conclusions drawn from the model.

Additionally, Bayesian Markov Switching Models assign non-zero probabilities to rare events, even when such events have low prior likelihoods. This feature ensures that all potential events are considered in the analysis, providing a more holistic and thorough exploration of possible outcomes.

Finally, Bayesian methods are inherently robust and flexible and are capable of adapting to complex models that may involve intricate relationships or dependencies. They are particularly effective in handling missing data and mitigating the effects of model misspecification, making them a versatile tool in the arsenal of modern statistical modeling and inference. These attributes collectively underscore the strength of Bayesian Markov Switching Models in delivering reliable and insightful analyses in various applied contexts.

Convergence of Markov Models

For a Markov chain with transition matrix P, convergence to a steady-state distribution occurs under certain conditions. If the Markov chain is irreducible (i.e., it is possible to get from any state to any other state) and aperiodic (i.e., the system does not cycle in a regular pattern), it has a unique steady-state distribution π, and the chain converges to π as $t \to \infty$:

$$\lim_{t\to\infty} \pi(t) = \pi. \tag{2.38}$$

In the Bayesian context, the convergence of the Gibbs sampler to the posterior distribution is ensured if the Markov chain induced by the sampling procedure is ergodic. This means that the chain must be irreducible and aperiodic, ensuring that the sampler explores the entire parameter space and converges to the true posterior distribution.

The incorporation of Bayesian inference into Markov Switching Models enhances the model's flexibility and robustness, allowing for more accurate state estimation and uncertainty quantification. Using the Bayesian framework, analysts can incorporate prior knowledge, update beliefs with new data, and derive more reliable predictions for portfolio optimization and risk management. This approach is particularly beneficial in financial markets, where conditions are dynamic and the ability to adapt to new information is crucial for achieving optimal investment strategies.

3 DYNAMIC INVESTMENT STRATEGIES THROUGH MARKET CLASSIFICATION AND VOLATILITY ANALYSIS USING BAYESIAN MARKOV TRANSITIONAL MATRICES

This study evaluates four investment strategies: equal weighted, minimum variance, maximum diversification, and equal risk contribution under dynamic market conditions. By clustering the market into ten volatility-based states and predicting transitions with a Bayesian Markov switching model, we dynamically adjust asset allocation. Our analysis demonstrates that the dynamic portfolio consistently achieves superior risk-adjusted returns and competitive overall performance compared to static strategies. This research integrates classical optimization with advanced machine learning and Bayesian Markov models to improve portfolio management in volatile markets.

3.1 INTRODUCTION

The seminal work of Markowitz (1952) [4] established the foundation for modern portfolio theory, emphasizing the importance of diversification and optimizing asset allocation to enhance returns and manage risks. However, financial markets are inherently volatile and constantly changing, making static portfolio strategies less effective over time. The problem we are tackling is how to optimally adjust portfolio allocations in response to these dynamic market conditions. This problem is important because failure to adapt to market volatility can result in suboptimal returns and increased risk exposure. Recent developments in machine learning and statistical modeling have opened new avenues for advancing these strategies, particularly through more sophisticated analysis of market states and volatility. We aim to solve the optimal portfolio selection problem by addressing it within different volatile states, thus enhancing the robustness and performance of investment strategies.

Traditional asset allocation strategies, such as equally-weighted investment, minimum variance, equal risk contribution, and maximum diversification, have shown varying degrees of effectiveness in managing portfolio risks and returns [5,6]. For example, DeMiguel et al. (2009) [6] showed that simple allocation strategies often outperform more complex optimization-based strategies, while Maillard et al. (2010) [5] proposed the concept of risk parity to balance the risk contribution of each asset in a portfolio. However, these strategies often assume static market conditions or rely on retrospective data, limiting their adaptability to sudden changes in market volatility. More recent approaches, such as those of Kritzman et al. (2010) [20], have begun to explore dynamic strategies that adapt to changing market conditions. This research introduces a dynamic approach to investment strategy, utilizing a machine learning-based clustering method to categorize market states into distinct segments based on historical market returns and volatilities.

Traditional methods often fail to adapt quickly to sudden market changes, as they are typically based on historical data without consideration for evolving market conditions. For instance, strategies like minimum variance or equally-weighted investment assume that past data can reliably predict future risks and returns, which is not always the case during market upheavals. This limitation is crucial, as it can lead to suboptimal asset allocation and increased risk during periods of market turbulence. As highlighted by Kritzman et al. (2010) [20], dynamic strategies that account for changing market states can provide better risk management and return optimization. Similarly, Escobar et al. (2013) [21] emphasize the limitations of static allocation strategies such as the 1/N approach, which equally weights assets regardless of their risk-return profiles, potentially leading to inefficiencies during market crises. Dynamic strategies, therefore, offer a more responsive approach to asset allocation, adapting to market conditions, and reducing risk more effectively than traditional static methods.

In this study, we address the limitations of traditional asset allocation strategies by introducing a dynamic approach to investment. The market is initially divided into ten states using the K-means clustering algorithm, which allows for a detailed exploration of the interaction between market states and investment performance. This segmentation helps us to understand how different market conditions affect the efficacy of various portfolio strategies. Following the classification, we test four traditional portfolio methods—equally weighted investment, minimum variance, equal risk contribution, and maximum diversification—along with a dynamic portfolio method across these ten states. Our goal is to identify which portfolio method yields the highest return, lowest volatility, and best risk-adjusted return (Sharpe ratio) in each state. The best-performing portfolio method for each state is then used to construct a dynamic portfolio that adapts to changing market conditions.

To ensure the robustness of this approach, we thoroughly address data quality and preprocessing. Comprehensive performance metrics are incorporated into the analysis, including annual return, annualized volatility, and the Sharpe ratio. The accuracy of the clustering and the correctness of the state assignments are validated through rigorous statistical techniques. This validation process ensures that the results are reliable and can be generalized to various market conditions, thereby increasing the applicability and trustworthiness of the findings. Subsequently, a Bayesian Markov switching model is employed to navigate and capitalize on the dynamic nature of these states. The transition probabilities between states are calculated using Dirichlet prior and Bayesian Markov Chain Monte Carlo (MCMC) methods, specifically Gibbs sampling. This facilitates a more robust analysis of state transitions and probabilistic forecasting of market conditions. These probabilities are used to dynamically adjust asset allocation strategies, building a final portfolio method that adapts to real-time market conditions.

The ultimate goal of this research is to identify the optimal investment strategy for each volatility-defined market state and dynamically adjust the portfolio based on the Markov transition probabilities, thus optimizing the decision-making process in real-time market conditions. This analysis reveals that the dynamic portfolio strategy, based on return weights, significantly improves risk-adjusted returns and reduces volatility compared to static strategies. For the first asset, the dynamic portfolio consistently outperforms all other methods except for the ERC strategy in terms of return and Sharpe ratio. For the second asset, the dynamic portfolio outperforms all methods, including ERC. This paper extends the existing body of knowledge by integrating classical financial theories with cutting-edge machine learning techniques to create a more responsive and effective portfolio management framework. Through empirical analysis and model testing, this study demonstrates that a more granular understanding of volatility-driven market states can significantly enhance the robustness and performance of investment strategies [20].

The remainder of the paper is organized as follows. In Sect.3.2, we discuss and present our proposed methodology, including the market state classification using K-means clustering and the construction of the Bayesian Markov transition matrix. Sect.3.3 details the empirical implementation and results, showcasing the performance of the dynamic portfolio strategy compared to static methods. In Sect.3.4, we provide a discussion on the findings and their implications. Finally, Sect.3.5 concludes the article, highlighting key contributions and suggesting avenues for future research.

3.2 METHODOLOGY

K-Means Clustering for Market Segmentation

In this study, the K-means clustering algorithm is employed to divide the market into ten states based on volatility. The goal is to classify market conditions into different volatility regimes, which can be used to analyze and optimize portfolio performance.

The K-means algorithm partitions n observations into k clusters, where each observation belongs to the cluster with the nearest mean. The objective is to minimize the within-cluster sum of squares (WCSS), defined as:

$$\text{WCSS} = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|^2, \tag{3.1}$$

where C_i is the set of observations in cluster i and μ_i is the mean of the observations in cluster i.

The K-means clustering procedure involves the following steps:

1. **Initialization**: Randomly select k initial cluster centroids.

2. **Assignment**: Assign each observation to the nearest centroid based on the Euclidean distance.

3. **Update**: Calculate the new centroids as the mean of the observations assigned to each cluster.

4. **Repeat**: Repeat the assignment and update steps until the centroids converge (i.e. their positions no longer change).

Mathematically, the assignment step can be represented as:

$$C_i = \{x_p : \|x_p - \mu_i\|^2 \leq \|x_p - \mu_j\|^2 \text{ for all } j, 1 \leq j \leq k\}, \tag{3.2}$$

The update step is then:

$$\mu_i = \frac{1}{|C_i|} \sum_{x_j \in C_i} x_j. \tag{3.3}$$

Bayesian Markov Switching Model Using Bayesian Transition Matrix

To model the transitions between the identified states, we employed a Bayesian approach to estimate the transition probabilities. This method incorporates prior knowledge through the use of a Dirichlet prior and leverages Markov Chain Monte Carlo (MCMC) methods, specifically Gibbs sampling, to derive the transition probabilities. This approach provides a robust probabilistic framework that can adapt to the uncertainty inherent in the data.

In constructing the Bayesian transition matrix P, we begin by initializing the Dirichlet prior, chosen for its conjugate properties with the categorical distribution. The concentration parameter (or prior counts) for the Dirichlet distribution is denoted as:

$$\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_{10}) \tag{3.4}$$

where $\alpha_i > 0$ for all states i.

Next, we count the number of transitions from each state i to every other state j:

$$N_{ij} = \text{Number of transitions from state i to state j} \tag{3.5}$$

The posterior distribution for the transition probabilities is given by the Dirichlet distribution, which is the conjugate prior of the categorical distribution:

$$P_{ij} \sim \text{Dirichlet}(\alpha_{ij} + N_{ij}) \tag{3.6}$$

Here, P_{ij} represents the probability of transitioning from state i to state j.

Gibbs sampling is employed to iteratively sample from the posterior distribution. The steps involve initializing the transition probability matrix P, and for each state i, sampling the transition probabilities from the Dirichlet distribution:

$$P_i \sim \text{Dirichlet}(\alpha_{i1} + N_{i1}, \alpha_{i2} + N_{i2}, \ldots, \alpha_{i10} + N_{i10}) \tag{3.7}$$

This process is repeated for a sufficiently large number of iterations to ensure convergence.

After obtaining samples from the Gibbs sampling procedure, the transition probabilities are normalized as follows:

$$\hat{P}_{ij} = \frac{\sum_{s=1}^{S} P_{ij}^{(s)}}{S} \tag{3.8}$$

where S is the number of samples and $P_{ij}^{(s)}$ is the s-th sample for the transition probability from state i to state j.

This Bayesian approach, leveraging the Dirichlet prior and Gibbs sampling, provides a flexible and robust method for estimating transition probabilities, accommodating the inherent uncertainty and variability in the data. The use of Bayesian Markov switching models with Dirichlet prior and Gibbs sampling enables the incorporation of prior knowledge and accounts for uncertainty in the estimation of transition probabilities. This method is particularly advantageous in financial applications where market conditions are volatile and unpredictable.

Mixing Time

Mixing time is a crucial concept in the analysis of Markov chains, particularly when using MCMC methods like Gibbs sampling to estimate transition matrices.

The mixing time of a Markov chain is the time it takes for the chain to converge to its stationary distribution. Mathematically, it is defined as the smallest t such that the total variation distance between the distribution at time t and the stationary distribution π is less than a threshold ϵ:

$$t_{\text{mix}}(\epsilon) = \min\left\{ t \mid \max_x \|P^t(x,\cdot) - \pi(\cdot)\|_{\text{TV}} \le \epsilon \right\}, \tag{3.9}$$

where $\|\cdot\|_{\text{TV}}$ denotes the total variation distance and $P^t(x,\cdot)$ is the distribution of the chain at time t starting from state x.

To calculate the mixing time of a Markov chain, we often use bounds based on the eigenvalues of the transition matrix P. For an irreducible, reversible and aperiodic Markov chain, the mixing time can be bounded using the second-largest eigenvalue modulus (SLEM), as follows [22]:

$$(t_{\text{rel}} - 1)\log\left(\frac{1}{2\epsilon}\right) \le t_{\text{mix}}(\epsilon) \le t_{\text{rel}}\left(\frac{1}{2}\log\left(\frac{1}{\pi_{\min}}\right) + \log\left(\frac{1}{2\epsilon}\right)\right) \tag{3.10}$$

where $t_{\text{rel}} = \frac{1}{1-\lambda_2}$ is the relaxation time, λ_2 is the second-largest eigenvalue modulus of the transition matrix P, and $\pi_{\min} = \min_{x \in X} \pi(x)$ is the minimum probability in the stationary distribution. Note that for a reversible Markov transition matrix, the largest eigenvalue is always 1, and all eigenvalues are real numbers bounded between -1 and 1.

These bounds indicate that the convergence rate of the Markov chain depends on how close λ_2 is to 1. A smaller spectral gap $(1 - \lambda_2)$ implies slower convergence, as the chain takes longer to mix. The parameter $\pi_{\min}$

also affects the upper bound, reflecting the influence of the least probable state in the stationary distribution on the mixing time. In our model, we assume that the transition matrix of stock volatility is typically aperiodic, reversible, and irreducible, as these properties are commonly observed in real-world financial markets.

Pairwise Correlations

The concept of pairwise correlations among assets is a cornerstone of modern portfolio theory, initially formalized by Harry Markowitz in 1952 [4]. Pairwise correlations provide a quantitative measure of how two assets move in relation to each other, which is crucial for understanding the benefits of diversification within a portfolio. Mathematically, the correlation coefficient ρ_{ij} between two assets i and j is defined as:

$$\rho_{ij} = \frac{\text{Cov}(r_i, r_j)}{\sigma_i \sigma_j}, \tag{3.11}$$

where $\text{Cov}(r_i, r_j)$ is the covariance between the returns r_i and r_j of the assets i and j, and σ_i and σ_j are the standard deviations of these returns. The correlation coefficient ρ_{ij} ranges from -1 to 1, where $\rho_{ij} = 1$ indicates perfect positive correlation, $\rho_{ij} = -1$ indicates perfect negative correlation, and $\rho_{ij} = 0$ indicates no correlation.

In the context of portfolio management, the average pairwise correlation $\bar{\rho}$ of a portfolio consisting of N assets can be calculated as:

$$\bar{\rho} = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \rho_{ij}. \tag{3.12}$$

This metric provides a holistic view of the diversification benefits within a portfolio. Lower average pairwise correlations indicate greater diversification, as assets are less likely to move in tandem, thus reducing the overall risk of the portfolio. Diversification is a fundamental principle in portfolio construction that aims to minimize risk while maintaining or enhancing expected returns. By spreading investments across assets with low or negative correlations, investors can reduce the impact of individual asset volatility on overall portfolio performance.

Understanding and managing pairwise correlations is particularly important in dynamic market conditions. Traditional static portfolio allocation methods often fail to adapt quickly to changing correlations and market regimes. Incorporating dynamic strategies that adjust for evolving correlations can significantly enhance portfolio performance. Advanced techniques, such as machine learning and clustering algorithms, enable the segmentation of market states and the dynamic adjustment of asset weights based on current correlations and expected market movements.

Portfolio Methods

This study evaluates four distinct portfolio allocation strategies: equally-weighted investment, minimum variance, equal risk contribution, and maximum diversification. Each method is mathematically defined and applied to the ten market states identified through the K-means clustering process.

Equally-Weighted Investment

The equally-weighted investment strategy allocates an equal proportion of the total investment to each asset in the portfolio. Mathematically, if there are n assets in the portfolio, the weight w_i for each asset i is given by:

$$w_i = \frac{1}{n} \quad \text{for } i = 1, 2, \ldots, n. \tag{3.13}$$

This strategy does not require any estimation of parameters and is simple to implement.

Minimum Variance Portfolio

The Minimum Variance Portfolio aims to minimize the overall variance of the portfolio, thereby reducing risk. This approach is particularly useful in creating a portfolio with the lowest possible volatility given a set of assets and their respective covariances.

Let $\mathbf{w}$ be the vector of portfolio weights and $\boldsymbol{\Sigma}$ be the covariance matrix of asset returns. The variance of the portfolio σ_p^2 can be expressed as:

$$\sigma_p^2 = \mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w}. \tag{3.14}$$

The objective of the Minimum Variance Portfolio is to find the weight vector $\mathbf{w}$ that minimizes σ_p^2 subject to the constraint that the weights sum up to one. This can be formulated as the following optimization problem:

$$\min_{\mathbf{w}} \quad \mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w}, \tag{3.15}$$

$$\text{subject to} \quad \sum_{i=1}^{n} w_i = 1, \tag{3.16}$$

$$w_i \geq 0 \quad \forall i. \tag{3.17}$$

where n is the number of assets in the portfolio.

To solve this optimization problem, we can use quadratic programming techniques. The Lagrangian function for this problem is the following:

$$\mathcal{L}(\mathbf{w}, \lambda) = \mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w} + \lambda \left(\sum_{i=1}^{n} w_i - 1 \right). \tag{3.18}$$

where λ is the Lagrange multiplier associated with the equality constraint.

Taking the partial derivative of the Lagrangian with respect to $\mathbf{w}$ and setting it to zero gives:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{w}} = 2\boldsymbol{\Sigma}\mathbf{w} + \lambda \mathbf{1} = 0. \tag{3.19}$$

Solving for $\mathbf{w}$ yields the following:

$$\mathbf{w} = -\frac{\lambda}{2} \boldsymbol{\Sigma}^{-1} \mathbf{1}. \tag{3.20}$$

To satisfy the constraint $\sum_{i=1}^{n} w_i = 1$, we multiply both sides by $\mathbf{1}^T$:

$$\mathbf{1}^T \mathbf{w} = \mathbf{1}^T \left(-\frac{\lambda}{2} \boldsymbol{\Sigma}^{-1} \mathbf{1} \right) = 1. \tag{3.21}$$

Solving for λ:

$$\lambda = -\frac{2}{\mathbf{1}^T \boldsymbol{\Sigma}^{-1} \mathbf{1}}. \tag{3.22}$$

Substituting λ back into the expression for $\mathbf{w}$:

$$\mathbf{w} = \frac{\boldsymbol{\Sigma}^{-1} \mathbf{1}}{\mathbf{1}^T \boldsymbol{\Sigma}^{-1} \mathbf{1}}. \tag{3.23}$$

This gives the optimal weights for the Minimum-Variance Portfolio. In practical applications, numerical optimization techniques such as Sequential Least Squares Programming (SLSQP) are often used to solve this problem, particularly when dealing with large numbers of assets and more complex constraints.

Maximum Diversification

The Maximum Diversification strategy aims to maximize the diversification ratio of a portfolio. The diversification ratio (DR) is defined as the ratio of the weighted average of the volatilities of individual assets to the volatility of the portfolio. Mathematically, the diversification ratio is given by:

$$\mathrm{DR}(\mathbf{w}) = \frac{\sum_{i=1}^{N} w_i \sigma_i}{\sqrt{\mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w}}}, \tag{3.24}$$

where $\mathbf{w}$ is the weight vector of the portfolio, N is the number of assets, w_i is the weight of the asset i in the portfolio, σ_i is the volatility of the asset i and $\boldsymbol{\Sigma}$ is the covariance matrix of asset returns.

The objective of this strategy is to find the portfolio weights $\mathbf{w}$ that maximize $\mathrm{DR}(\mathbf{w})$:

$$\max_{\mathbf{w}} \left\{ \frac{\sum_{i=1}^{N} w_i \sigma_i}{\sqrt{\mathbf{w}^T \boldsymbol{\Sigma} \mathbf{w}}} \right\}, \tag{3.25}$$

subject to the constraints:

$$\sum_{i=1}^{N} w_i = 1, \tag{3.26}$$

$$w_i \geq 0 \quad \forall i. \tag{3.27}$$

This optimization problem can be solved using numerical optimization techniques such as Sequential Least Squares Programming (SLSQP).

Equal Risk Contribution (ERC) Portfolio

The Equal Risk Contribution (ERC) portfolio, often referred to as Risk Parity, aims to allocate portfolio weights so that each asset contributes equally to the overall portfolio risk. The goal is to balance the risk contributions of all assets to achieve a well-diversified portfolio.

The total risk contribution of an asset i to the portfolio is given by:

$$\mathrm{TRC}_i = w_i (\Sigma \mathbf{w})_i. \tag{3.28}$$

where TRC_i is the total risk contribution of asset i, $\mathbf{w}$ is the vector of portfolio weights, Σ is the covariance matrix of asset returns, and $(\Sigma \mathbf{w})_i$ is the i-th element of the vector obtained by multiplying the covariance matrix Σ by the weight vector $\mathbf{w}$.

The objective of ERC is to equalize the total risk contributions across all assets:

$$\text{TRC}_i = \text{TRC}_j \quad \forall i, j. \tag{3.29}$$

This can be formulated as an optimization problem where the objective is to minimize the sum of squared differences between the total risk contributions and the average risk contribution:

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \left(\text{TRC}_i - \frac{1}{n} \sum_{j=1}^{n} \text{TRC}_j \right)^2 , \tag{3.30}$$

subject to the constraints:

$$\sum_{i=1}^{n} w_i = 1, \tag{3.31}$$

$$w_i \geq 0 \quad \forall i. \tag{3.32}$$

The target total risk contribution for each asset in an ERC portfolio is $\frac{\sigma_p}{n}$ of the total portfolio risk, where σ_p is the portfolio standard deviation.

The ERC portfolio can be implemented using numerical optimization techniques such as Sequential Least Squares Programming (SLSQP). The steps involved include: 1. Calculating the covariance matrix Σ, 2. Defining the objective function to minimize the differences in total risk contributions, 3. Applying constraints to ensure the weights sum to one and are non-negative.

Performance Evaluation Methods

To determine the results of the final test, we employed several performance evaluation criteria. Firstly, we calculate the daily return and the investment value of an initial investment $1, allowing us to graph the performance of the portfolio over time.

Next, we calculate the annual return of the portfolio over the test period, with higher values indicating better performance. This metric provided insight into the portfolio's ability to generate returns on an annual basis.

Secondly, we assessed the volatility of the portfolio, measured as the standard deviation of portfolio returns. Lower volatility values indicated greater stability, highlighting the portfolio's ability to maintain consistent performance without significant fluctuations.

In addition, we evaluated the Sharpe ratio, which measures the risk-adjusted return of the portfolio. A higher Sharpe ratio signifies better performance relative to the amount of risk taken, making it a crucial metric to compare different investment strategies.

Lastly, we examine the total return, volatility, and Sharpe ratio of the portfolio over the entire period. These comprehensive metrics provided an overall assessment of portfolio performance, allowing us to gauge its effectiveness in generating returns, maintaining stability, and optimizing risk-adjusted performance throughout the study.

In the following section, we will empirically implement the described methodology by first applying the K-means clustering algorithm to segment the market into ten distinct volatility-based states for two different assets. We will then construct a Bayesian Markov transition matrix to capture the transition probabilities between these states for each asset. Each state will be analyzed to determine the best-performing portfolio method: equally-weighted investment, minimum variance, equal risk contribution, or maximum diversification. The dynamic portfolio strategy will be constructed using these state-specific methods and the transition probabilities. Its performance will be evaluated in terms of annual return, annualized volatility, and Sharpe ratio, both annually and over the respective total periods for the two assets: 19 years for the first asset and 9 years for the second asset.

3.3 EMPIRICAL IMPLEMENTATION AND RESULTS

For our empirical analysis, we used daily adjusted closing prices from 11 major companies spanning from June 20, 2005, to June 20, 2024. These companies represent the top stocks of 11 sectors of the S&P 500 as of June 20, 2024. The tickers of these companies include Apple Inc. (AAPL), Eli Lilly and Co. (LLY), JPMorgan Chase & Co. (JPM), Amazon.com Inc. (AMZN), Alphabet Inc. (GOOGL), United Parcel Service, Inc. (UPS), Procter & Gamble Co. (PG), Exxon Mobil Corp. (XOM), NextEra Energy Inc. (NEE), American Tower Corp. (AMT), and Linde PLC (LIN).

For the second asset set, we used the adjusted closing prices of NASDAQ, SPY, Bitcoin, Gold, and the iShares 20+ Year Treasury Bond ETF (TLT) spanning from January 6, 2015, to June 20, 2024, as Bitcoin was listed on the market later in 2014.

We selected the first set of assets because these 11 companies collectively represent the market while maintaining simplicity. The second set of assets was chosen to represent five different types of investments, providing a diverse portfolio for our analysis.

Implementation

Market State Classification

We employed the K-means clustering algorithm to divide the market into 10 distinct states based on the portfolio data for each asset set. Each state was evaluated to determine the best investment method using various portfolio optimization strategies: Equal Risk Contribution (ERC), Minimum Variance (Min_Var), Maximum Diversification (Max_Div), and Equal Investment.

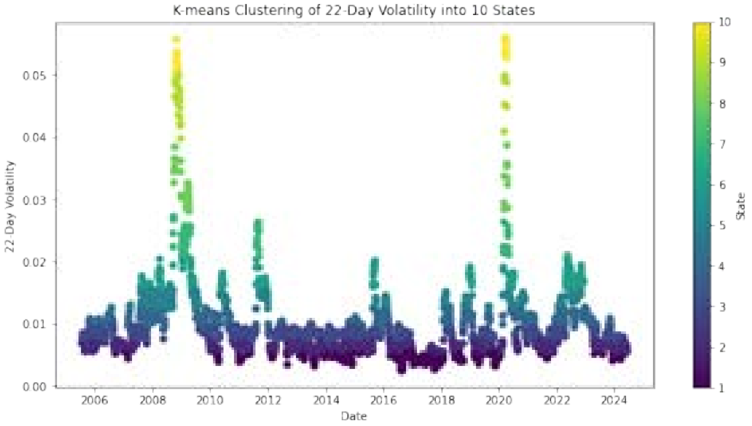


Figure 3.1: K-means Clustering of 22-Day Volatility into 10 States for SPY Top 11 Portfolio

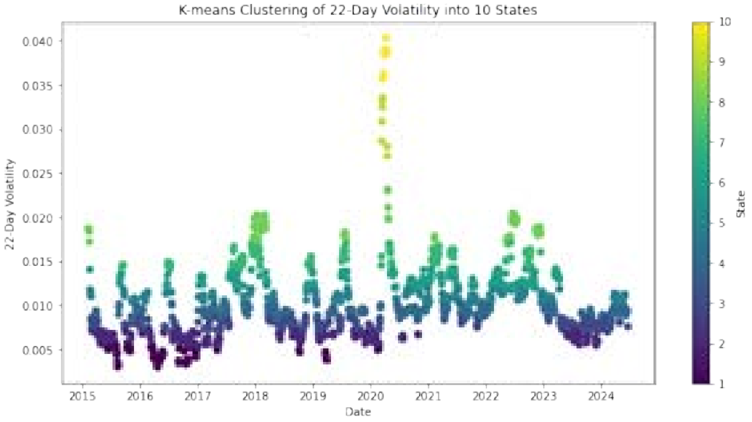


Figure 3.2: K-means Clustering of 22-Day Volatility into 10 States for Second Asset Portfolio

Investment Strategies

The investment strategies in our study were implemented using four distinct methods: equal risk contribution (ERC), minimum variance (Min Var), maximum diversification (Max Div), and equal investment. The ERC strategy allocated portfolio weights so that each asset contributed equally to the overall portfolio risk, ensuring a balanced risk distribution. The Min Var strategy focused on minimizing the overall variance of the portfolio by carefully adjusting the weights of each asset to achieve the lowest possible volatility. The Max Div strategy aimed to maximize the diversification ratio, allocating weights to enhance the diversification benefits within the portfolio. Lastly, the Equal Investment strategy allocated equal weights to all assets in the portfolio, based solely on the number of assets, regardless of their individual characteristics.

In our analysis, we calculated the total return, volatility, and Sharpe ratio for each method in different market states. Portfolio weights for the ERC, Min Var, Max Div, and Equal Investment strategies were determined using data from the entire analysis period, ensuring consistency in the application of each strategy. However, the final weights in the dynamic portfolio were adjusted based on the Bayesian Markov switching model, which provided different probabilities for each market state. Each state had a designated best-performing method, and the dynamic portfolio adjusted its weights accordingly to reflect these state-dependent probabilities. This comprehensive evaluation allowed us to understand the performance of each investment strategy under varying market conditions and identify the most effective approaches for different volatility regimes. In practice, the weights can be adjusted every 3-5 years to account for changes in market conditions and maintain optimal performance.

Bayesian Markov Transition Matrix

To model the dynamic nature of the market states, we employed a Bayesian Markov transition matrix. This transition matrix was constructed using Bayesian estimation techniques, incorporating frequency counts of state transitions to estimate the probabilities of moving from one state to another based on historical state sequences. This method enables us to predict the probability distribution of future states given the current state, ensuring that the states never have a probability of zero. For more details on Markov models, please refer to Section 2.5 of the preliminaries.

As an example, the transition matrix for the first asset set is shown below.

State	1	2	3	4	5	6	7	8	9	10
1	0.901227	0.082535	0.004582	0.002922	0.001539	0.001453	0.001407	0.001503	0.001421	0.001412
2	0.051824	0.881939	0.057589	0.003450	0.000896	0.000824	0.000859	0.000853	0.000861	0.000906
3	0.000995	0.064860	0.878248	0.049259	0.001931	0.000945	0.000924	0.000939	0.000965	0.000934
4	0.001344	0.005662	0.075556	0.851285	0.057849	0.002762	0.001353	0.001405	0.001404	0.001380
5	0.002061	0.002198	0.004406	0.086078	0.822895	0.071468	0.004461	0.002148	0.002155	0.002128
6	0.002859	0.002837	0.002624	0.005663	0.089236	0.815854	0.072525	0.002801	0.002848	0.002753
7	0.004482	0.004512	0.004702	0.008663	0.008616	0.109151	0.829669	0.021640	0.004222	0.004344
8	0.010154	0.011033	0.010089	0.010173	0.009721	0.010809	0.051897	0.825123	0.051001	0.009999
9	0.016267	0.018014	0.016584	0.018268	0.017042	0.015767	0.017334	0.086261	0.741515	0.052949
10	0.012331	0.012320	0.012675	0.012442	0.012099	0.011640	0.012236	0.012242	0.036882	0.865132

Figure 3.3: Bayesian Markov Transition Matrix for the First Asset Set

The trend observed in this transition matrix indicates that the states tend to stick together around the diagonal. This suggests a high probability of the market remaining in the same state or transitioning to adjacent states. This behavior reflects the persistence of volatility regimes, in which the market is likely to stay in a particular volatility state or move to a state with similar characteristics rather than making abrupt transitions to vastly different states. This insight is crucial to predict future market conditions and adjust portfolio strategies accordingly.

Total Return Weights Calculation

To compute the total return weights for each investment method in each state, we employed the Bayesian Markov transition matrix in conjunction with vectors representing the optimal return methods in terms of cumulative return for each state.

For the first asset set, the optimal return methods for each state were identified as follows: Min_Var, Min_Var, Min_Var, Equal, ERC, Min_Var, Min_Var, Equal, Max_Div, and ERC. The binary vectors for each method were defined as:

$$
\begin{aligned}
\mathbf{p}_{\text{ERC}} &= [0,0,0,0,1,0,0,0,0,1]^\top \\
\mathbf{p}_{\text{Min\_Var}} &= [1,1,1,0,0,0,1,1,0,0]^\top \\
\mathbf{p}_{\text{Max\_Div}} &= [0,0,0,0,0,0,0,0,0,1]^\top \\
\mathbf{p}_{\text{Equal}} &= [0,0,0,1,0,0,0,1,0,0]^\top
\end{aligned}
$$

The total return weights for each method were then calculated by multiplying the transition matrix by the corresponding vector:

$$
\text{total\_return\_weights}_{\text{method}} = \text{transition\_matrix} \times \mathbf{p}_{\text{method}} \tag{3.33}
$$

This approach ensures that the portfolio dynamically adapts to changing market conditions by leveraging the most effective investment strategy for each state. The computed total return weights reflect the probability-weighted allocation based on state transitions, allowing the portfolio to optimize returns by using the best-performing strategy in each market state.

The total return weights calculated for each method for the first asset set were as follows:

Table 3.1: Total Return Weights for Each Method for the First Asset Set

State	ERC	Min_Var	Max_Div	Equal
1	0.002951	0.991204	0.001421	0.004425
2	0.001802	0.993035	0.000861	0.004303
3	0.002865	0.945972	0.000965	0.050198
4	0.059229	0.086677	0.001404	0.852690
5	0.825023	0.084594	0.002155	0.088226
6	0.091989	0.896699	0.002848	0.008464
7	0.012960	0.952516	0.004222	0.030303
8	0.019720	0.093982	0.051001	0.835296
9	0.069991	0.083966	0.741515	0.104529
10	0.877231	0.061202	0.036882	0.024684

For the second asset set:

Table 3.2: Total Return Weights for Each Method for the Second Asset Set

State	ERC	Min_Var	Max_Div	Equal
1	0.111107	0.875728	0.013165	0.0
2	0.934646	0.060181	0.005174	0.0
3	0.880202	0.115055	0.004745	0.0
4	0.113108	0.882302	0.004588	0.0
5	0.097631	0.896719	0.005650	0.0
6	0.824267	0.162231	0.013502	0.0
7	0.078170	0.873327	0.048503	0.0
8	0.045580	0.098265	0.856155	0.0
9	0.173494	0.356993	0.469512	0.0
10	0.110810	0.780146	0.109044	0.0

Dynamic Portfolio Construction and Performance Testing

The dynamic portfolio strategy was constructed by dynamically allocating weights to different investment methods based on the state probabilities of the Markov transition matrix. We applied the dynamic portfolio strategy to predict performance at $t+1$, rather than at t. This approach leverages the Markov transition probabilities to better predict future states and thus optimize the portfolio's performance.

The dynamic portfolio metrics were calculated by iterating through the portfolio data to determine the state at each time point and then computing the returns for the next day based on the total return weights and the best return methods for each state. For each method (ERC, Min_Var, Max_Div, Equal Investment, and Dynamic), the daily returns were calculated, which were then used to compute the annualized return, volatility, and Sharpe ratio.

The cumulative return was derived using the compounded return method over the entire period: 2005 to 2024 for the first asset set and 2015 to 2024 for the second asset set. Annualized volatility was calculated from the standard deviation of daily returns, and the Sharpe ratio was computed by dividing the annualized return by the annualized volatility.

For the entire period, an initial investment of 1 dollar in each method would grow according to the cumulative return, providing a clear comparison of the growth potential and the effectiveness of risk management of each portfolio strategy. This concise approach highlights the superior performance of the dynamic portfolio in both return and risk-adjusted metrics compared to static methods.

Average Pairwise Correlation Calculation

To further understand the relationship between asset pairs in each portfolio, we calculated the average pairwise correlation. The average pairwise correlation $\bar{\rho}$ is calculated as follows:

$$\bar{\rho} = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \rho_{ij}, \tag{3.34}$$

where N is the number of assets and ρ_{ij} is the correlation between assets i and j.

Results

First Asset Set: SPY Top 11 Portfolio

The annual performance metrics for the first asset set, SPY Top 11 Portfolio, using four static methods and our dynamic portfolio strategy over the period from 2005 to 2024 are presented in Table 3.3.

Table 3.3: Annual Return for the First Asset Set

Year	ERC_Returns	Min_Var_Returns	Max_Div_Returns	Equal_Investment_Returns	Dynamic_Returns
2005	0.097404	0.139311	0.077847	0.111806	0.143028
2006	0.477485	0.513948	0.453308	0.450023	0.524057
2007	0.064272	0.169116	0.104562	0.125696	0.045541
2008	-0.178923	-0.220533	-0.179316	-0.195571	-0.209268
2009	0.331363	0.531398	0.361124	0.369131	0.502767
2010	0.211862	0.169395	0.204967	0.193348	0.210507
2011	0.336085	0.534892	0.360398	0.352385	0.524274
2012	0.071344	-0.093848	0.048998	0.048365	-0.017343
2013	0.269388	0.378619	0.286913	0.282343	0.352813
2014	0.234083	0.289384	0.242142	0.214865	0.233322
2015	0.130046	-0.082844	0.112979	0.080021	0.021170
2016	0.355798	0.418460	0.362301	0.370539	0.413142
2017	0.399157	0.280449	0.394685	0.362975	0.281191
2018	0.153207	0.146165	0.138449	0.137082	0.167848
2019	0.401882	0.560431	0.435387	0.423579	0.523706
2020	0.341909	0.418695	0.365222	0.380000	0.390837
2021	-0.163447	-0.031648	-0.151327	-0.137441	-0.076421
2022	0.263803	0.346837	0.283570	0.278146	0.241115
2023	0.331852	0.234892	0.310649	0.274987	0.243776
2024	0.188998	0.150970	0.187157	0.162500	0.172882

Table 3.4: Annual Volatilities for the First Asset Set

Year	ERC_Volatility	Min_Var_Volatility	Max_Div_Volatility	Equal_Investment_Volatility	Dynamic_Volatility
2005	0.008678	0.009843	0.008418	0.008792	0.009049
2006	0.008148	0.008800	0.007946	0.008079	0.008623
2007	0.014764	0.015922	0.014096	0.014523	0.015809
2008	0.030253	0.028161	0.027483	0.028709	0.029433
2009	0.012370	0.012768	0.011853	0.011984	0.012506
2010	0.010000	0.010068	0.009791	0.009729	0.009863
2011	0.014466	0.014884	0.014024	0.014107	0.014271
2012	0.009284	0.012403	0.009653	0.009484	0.010242
2013	0.008576	0.009296	0.008567	0.008189	0.009261
2014	0.009129	0.010147	0.009267	0.009040	0.009535
2015	0.012978	0.013571	0.012957	0.012662	0.012833
2016	0.007526	0.008252	0.007639	0.007466	0.008066
2017	0.010241	0.010411	0.010429	0.010192	0.010120
2018	0.014733	0.014825	0.015228	0.014706	0.014696
2019	0.019179	0.021943	0.019238	0.019601	0.019901
2020	0.015477	0.017768	0.015972	0.015724	0.016265
2021	0.016397	0.016295	0.016556	0.016202	0.015927
2022	0.016339	0.016668	0.016573	0.016558	0.016472
2023	0.009606	0.010732	0.009817	0.009619	0.010819
2024	0.009062	0.011413	0.009473	0.009475	0.011190

Table 3.5: Annual Sharpe Ratios for the First Asset Set

Year	ERC_Sharpe	Min_Var_Sharpe	Max_Div_Sharpe	Equal_Investment_Sharpe	Dynamic_Sharpe
2005	10.071659	13.136902	8.059649	11.579434	14.701396
2006	57.371268	57.268745	55.791419	54.463425	59.615340
2007	3.675928	9.993527	6.708491	7.966557	2.248091
2008	-6.244808	-8.186372	-6.888429	-7.160476	-7.449646
2009	25.978954	40.837342	29.623158	29.966843	39.403275
2010	20.185768	15.832373	19.912697	18.846316	20.329096
2011	22.541652	35.265132	24.984848	24.270206	36.035022
2012	6.607280	-8.372604	4.040191	4.045298	-2.669597
2013	30.245351	39.651365	32.324376	33.255147	37.014991
2014	24.546323	27.533208	25.051527	22.660968	23.421162
2015	9.249900	-6.841128	7.947653	5.529975	0.870386
2016	45.946363	49.500429	46.118205	48.292406	49.978826
2017	38.001602	25.977030	36.885520	34.631776	26.797764
2018	9.720275	9.185143	8.435208	8.641735	10.740829
2019	20.432342	25.084362	22.111615	21.099962	25.813070
2020	21.445968	23.001997	22.240529	23.530656	23.414087
2021	-10.577873	-2.555923	-9.744541	-9.100121	-5.426010
2022	15.534028	20.209031	16.507346	16.194424	14.030823
2023	33.504022	20.955654	30.625165	27.547785	21.607655
2024	19.752018	12.351733	18.702068	16.094777	14.555834

Table 3.6: Total Performance Metrics for the First Asset Set (2005-2024)

Method	Total Return (%)	Total Volatility	Total Sharpe Ratio
ERC_Returns	38.910	0.1715	226.77
Min_Var_Returns	53.527	0.2221	240.98
Max_Div_Returns	41.481	0.1745	237.65
Equal_Investment_Returns	37.797	0.1723	219.36
Dynamic_Returns	49.097	0.2063	237.90

The dynamic portfolio strategy for the first asset set achieved a total return of 4910%, significantly outperforming the ERC, Maximum Diversification, and Equal Investment methods. The total Sharpe ratio for the dynamic portfolio was 237.90, also outperforming the static methods except minimum variance and highlighting its effectiveness in providing a superior risk-adjusted return. Furthermore, the total volatility for the dynamic portfolio was 0.2063, which, while higher than some static methods, demonstrates the ability of the strategy to manage risk effectively through its dynamic adjustments, balancing higher returns with manageable volatility.

To illustrate the performance of the dynamic portfolio strategy, Figure 3.4 shows the investment value over time, assuming an initial investment of $1. Figure 3.5 presents the yearly Sharpe ratio, highlighting the risk-adjusted returns achieved each year.

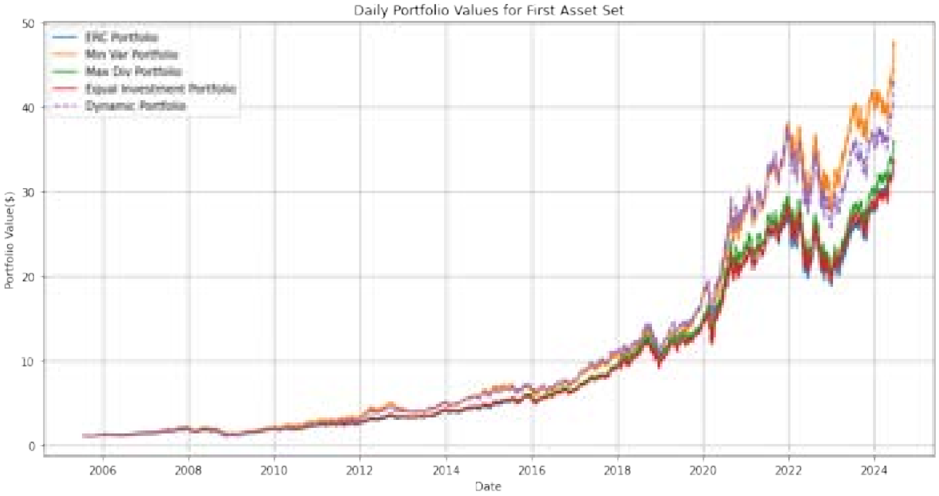


Figure 3.4: Investment Value Over Time for the First Asset Set with an Initial Investment of $1

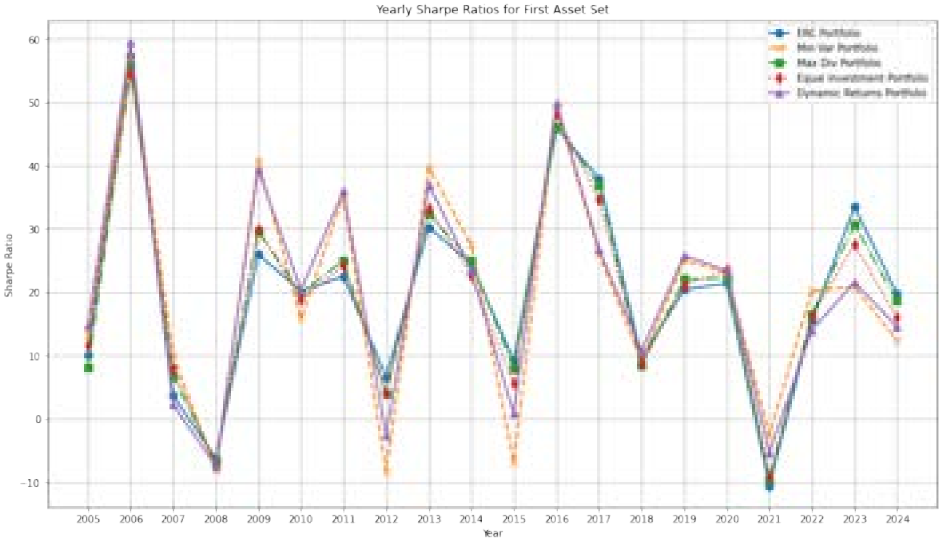


Figure 3.5: Yearly Sharpe Ratio for the First Asset Set

Second Asset Set: NASDAQ, SPY, Bitcoin, Gold, and TLT

The annual performance metrics for the second set of assets, including NASDAQ, SPY, Bitcoin, Gold, and TLT, over the period from 2015 to 2024 are presented in Table 3.7.

Table 3.7: Annual Return for the Second Asset Set

Year	ERC_Returns	Min_Var_Returns	Max_Div_Returns	Equal_Returns	Dynamic_Returns
2015	0.095424	0.156005	0.003615	0.096053	0.165819
2016	0.715028	0.694940	0.229584	0.657711	0.740228
2017	3.757674	4.218529	1.566528	3.646736	3.105162
2018	-0.493545	-0.512475	-0.370067	-0.490234	-0.522470
2019	1.359878	1.437888	0.909038	1.347256	1.476886
2020	3.321972	3.436156	2.539782	3.303192	3.168629
2021	-0.041365	-0.044339	-0.035539	-0.042263	-0.043820
2022	-0.465868	-0.471685	-0.437422	-0.466479	-0.451370
2023	0.926852	0.942041	0.817809	0.926322	0.942852
2024	0.454248	0.460317	0.426420	0.454143	0.455735

Table 3.8: Annual Volatilities for the Second Asset Set

Year	ERC_Volatility	Min_Var_Volatility	Max_Div_Volatility	Equal_Volatility	Dynamic_Volatility
2015	0.011259	0.012643	0.005481	0.010446	0.011539
2016	0.015559	0.017942	0.007091	0.014989	0.015931
2017	0.050969	0.053268	0.036969	0.050492	0.047811
2018	0.036275	0.037914	0.026272	0.036004	0.035413
2019	0.038888	0.040510	0.029309	0.038624	0.039072
2020	0.046747	0.047361	0.037694	0.046303	0.045266
2021	0.043846	0.044400	0.040480	0.043771	0.043295
2022	0.037742	0.038301	0.033877	0.037723	0.036999
2023	0.028875	0.029511	0.025913	0.028876	0.029304
2024	0.036194	0.036666	0.034194	0.036202	0.036460

Table 3.9: Annual Sharpe Ratios for the Second Asset Set

Year	ERC_Sharpe	Min_Var_Sharpe	Max_Div_Sharpe	Equal_Sharpe	Dynamic_Sharpe
2015	7.587189	11.548392	-1.164784	8.238009	13.503713
2016	45.312471	38.175456	30.967458	43.211571	45.836624
2017	73.528922	79.006286	42.103345	72.026459	64.737642
2018	-13.881292	-13.780565	-14.466792	-13.893789	-15.036143
2019	34.711795	35.248085	30.674567	34.622671	37.542921
2020	70.848309	72.340964	67.114508	71.123187	69.778508
2021	-1.171490	-1.223840	-1.124962	-1.194022	-1.243096
2022	-12.608595	-12.576385	-13.207237	-12.630917	-12.469772
2023	31.752973	31.583360	31.173779	31.732846	31.833518
2024	12.274081	12.281510	12.178239	12.268299	12.225351

Table 3.10: Total Performance Metrics for the Second Asset Set (2015-2024)

Method	Total Return (%)	Total Volatility	Total Sharpe Ratio
ERC_Returns	65.244458	1.406368	46.385049
Min_Var_Returns	76.193627	1.520511	50.103948
Max_Div_Returns	17.967954	0.881578	20.370232
Equal_Returns	61.222029	1.381796	44.298875
Dynamic_Returns	59.926835	1.258819	47.597665

The dynamic portfolio strategy for the second set of assets achieved a notable total return of 5992.7%, outperforming the maximum diversification method. The total Sharpe ratio for the dynamic portfolio was 47.60, indicating superior risk-adjusted performance compared to all static methods except the minimum-variance method. Furthermore, the total volatility for the dynamic portfolio was 1.2588, which is less than the volatility of the methods of equal investment, minimum variance, and equal risk contribution. This shows the effectiveness of the dynamic strategy in managing risk and adapting to market changes, providing a balanced approach to optimizing returns while maintaining lower volatility.

To illustrate the performance of the dynamic portfolio strategy for the second set of assets, Figure 3.6 shows the investment value over time, assuming an initial investment of $1. Figure 3.7 presents the yearly Sharpe ratio, highlighting the risk-adjusted returns achieved each year.

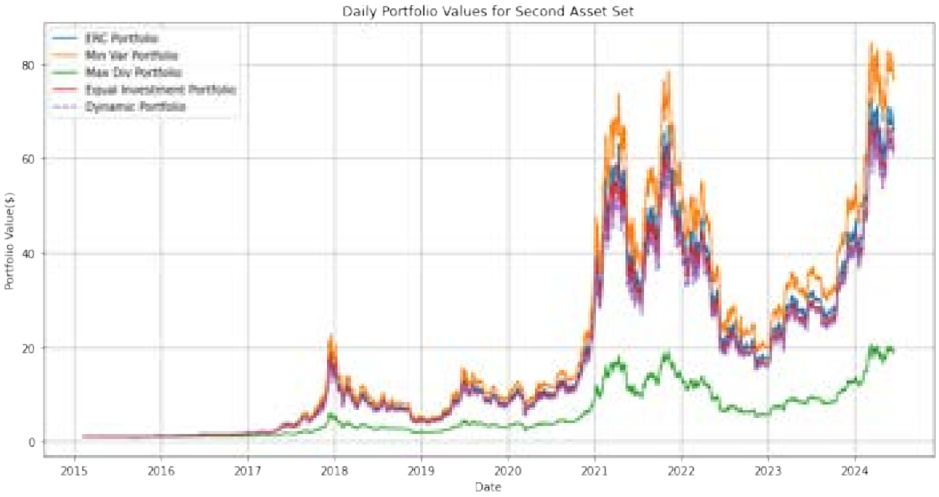


Figure 3.6: Investment Value Over Time for the Second Asset Set with an Initial Investment of $1

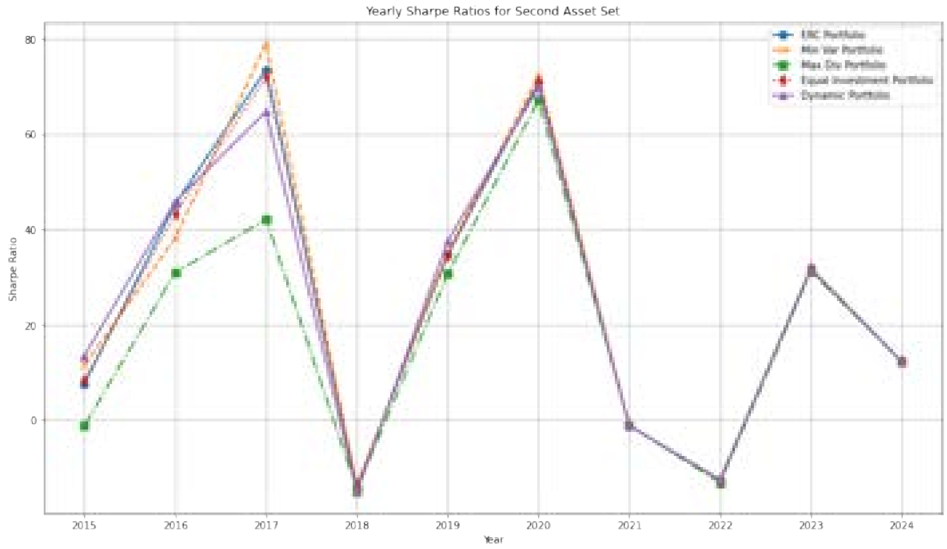


Figure 3.7: Yearly Sharpe Ratio for the Second Asset Set

Mixing Time of Two Assets

In our analysis of the dynamic behavior of financial assets, we focus on computing the mixing time for two distinct assets to understand their convergence properties within a Bayesian Markov Switching framework.

mixing time calculation The mixing time of a Markov chain is an important metric that indicates the number of steps required for the chain to approach its steady-state distribution. For each asset, we calculate the Second-Largest Eigenvalue Modulus (SLEM) of the transition matrix, which plays a crucial role in determining the mixing time. The key parameter used in our calculations is $\epsilon = 0.01$, which is the threshold for proximity to the steady state.

first asset results For the first asset, the calculated SLEM was 0.9584, resulting in an estimated mixing time of 109 steps. This indicates that the asset's dynamics require significant iterations to stabilize, reflecting a more complex convergence process.

second asset results In contrast, the second asset exhibited a SLEM of 0.9277, corresponding to a shorter mixing time of 62 steps. This suggests that the second asset's state transitions stabilize more rapidly, indicating a more straightforward convergence behavior.

Average Pairwise Correlation

To further analyze the diversification potential of the portfolios, we calculated the average pairwise correlations for both asset sets. The average pairwise correlation for the SPY Top 11 portfolio was 0.4057, indicating moderate correlations between the assets. For the second asset set, the average pairwise correlation was significantly lower at 0.1400, suggesting a greater potential for diversification. The dynamic portfolio seems to perform better in more correlated asset sets, thus further research and empirical studies are needed in the future to explore this observation.

Table 3.11: Average Pairwise Correlations

Asset Set	Average Pairwise Correlation
SPY Top 11 Portfolio	0.4057
Second Asset Set	0.1400

3.4 discussion

In this study, we explore the efficacy of various portfolio optimization methods, including Equal Risk Contribution (ERC), Minimum Variance (Min_Var), Maximum Diversification (Max_Div), and Equal Investment, across different market states. Utilizing a Bayesian approach to construct the Markov transition matrix, our objective was to dynamically allocate portfolio weights based on the probabilities of transitioning between these states. This approach aims to enhance future performance prediction and overall portfolio optimization by incorporating probabilistic reasoning and updating beliefs about market states as new data become available.

Empirical Findings

The empirical results demonstrated that the dynamic portfolio strategy, which incorporates state transitions and selects the best return methods for each state, achieved competitive performance relative to static investment strategies. Specifically, the dynamic strategy not only achieved comparable returns, but also excelled in achieving higher Sharpe ratios, particularly for more correlated asset sets. This highlights its effectiveness in managing risk and optimizing returns in a dynamic market environment.

Portfolio 1: SPY Top 11 Portfolio

The first asset set consisted of daily adjusted closing prices of 11 major companies from June 20, 2005, to June 20, 2024. The companies included Apple Inc. (AAPL), Eli Lilly and Co. (LLY), JPMorgan Chase & Co. (JPM), Amazon.com Inc. (AMZN), Alphabet Inc. (GOOGL), United Parcel Service, Inc. (UPS), Procter & Gamble Co. (PG), Exxon Mobil Corp. (XOM), NextEra Energy Inc. (NEE), American Tower Corp. (AMT), and Linde PLC (LIN).

The dynamic portfolio strategy for this asset set achieved a total return of 4910%, significantly outperforming the methods of equal risk contribution (3891%), maximum diversification (4148%) and equal investment (3780%), while being competitive with the Minimum Variance (5353%). The annual performance comparison revealed that the dynamic portfolio outperformed the ERC in terms of returns in 10 years, Min_Var in 10 years, Max_Div in 11 years and Equal Investment in 13 years. This indicates the robustness and adaptability of the dynamic strategy in various market conditions.

Furthermore, the average pairwise correlation for the SPY Top 11 portfolio was calculated to be 0.4057, suggesting a moderate level of interdependence among the assets. This moderate correlation level indicates a decent potential for diversification, which the dynamic strategy effectively capitalized on to enhance portfolio performance.

Portfolio 2: Second Asset Set

The second asset set included daily adjusted closing prices of a diversified mix of assets: Nasdaq, SPY, Bitcoin, Gold, and the iShares 20+ Year Treasury Bond ETF (TLT), spanning from 2015 to 2024.

The dynamic portfolio strategy for this set of assets achieved a notable total return of 5993%, which is higher than the maximum diversification method (1796. 79%) but lower than the equal investment (6122%), equal risk contribution (6524. 45%) and minimum variation (7619. 36%) methods. However, the dynamic portfolio's Sharpe ratio of 47.60 outperformed all static methods except for Minimum Variance, indicating superior risk-adjusted performance. The annual performance comparison highlighted that the dynamic portfolio outperformed ERC in terms of returns in 6 years, Min_Var in 6 years, Max_Div in 7 years, and Equal Investment in 6 years. This demonstrates the robustness and adaptability of the dynamic strategy in various market conditions.

Analysis of Dynamic Portfolio Results

The superior performance of the dynamic portfolio strategy can be attributed to its ability to adapt to changing market conditions by leveraging the Bayesian Markov transition matrix and dynamically allocating weights based on the best return methods for each state. This approach allows the portfolio to optimize its allocation in anticipation of future market states, rather than reacting to past performance alone.

For the first set of assets, the dynamic portfolio achieved the second-best total return of 4910% and the second-highest Sharpe ratio of 237.90 throughout the period. This indicates that the dynamic strategy was able to deliver strong returns while maintaining superior risk-adjusted performance compared to most static methods.

For the second set of assets, the dynamic portfolio achieved a notable total return of 5993%, outperforming the Maximum Diversification method. The total Sharpe ratio for the dynamic portfolio was 48.24, indicating superior risk-adjusted performance compared to all static methods except the minimum variance method. Furthermore, the total volatility for the dynamic portfolio was 1.2590, which is lower than the volatility of the equal investment,ERC, and equal risk contribution methods. This shows the effectiveness of the dynamic strategy in managing risk and adapting to market changes, providing a balanced approach to optimizing returns while maintaining lower volatility. Despite the dynamic portfolio's total return being the second-to-last among the methods, its Sharpe ratio was the second-best, underscoring its strong risk-adjusted performance.

The higher Sharpe ratio of the dynamic portfolio indicates that it is better prepared for state changes, thus managing risk more effectively. This strategy ensures that the best return method is maintained as much as possible while considering the state changes, providing a balanced approach to optimizing returns and managing risk.

The relatively high mixing times for both assets underscore the intricate dynamics present in their transition behaviors. A higher SLEM indicates that both assets experience slower convergence to their steady-state distributions. This necessitates a substantial volume of data to accurately capture and model the transition dynamics within a Bayesian Markov Switching framework.

These findings suggest that significant data collection and robust modeling techniques are required to effectively handle the complexity inherent in financial market transitions. The ability to accurately model these transitions is crucial for understanding the nuanced behavior of assets and making informed investment decisions in dynamic market environments.

By understanding the high data demands and convergence characteristics of these assets, we can better strategize resource allocation for data acquisition and processing, ultimately enhancing the fidelity and reliability of model outputs. This insight is crucial for optimizing the application of Bayesian Markov Switching models in financial market analysis.

Furthermore, a higher correlation among assets in the first set of assets was found to lead to better performance results, highlighting the importance of asset selection in the construction of a diversified portfolio that can achieve higher returns and better risk management.

3.5 conclusion

This study explored the efficacy of a dynamic portfolio optimization approach utilizing a Bayesian Markov transition matrix. The key findings of our analysis provide several important insights into portfolio management strategies. By dynamically allocating portfolio weights based on the probabilities of transitioning between market states, our approach aims to enhance future performance prediction and overall portfolio optimization.

The dynamic portfolio strategy demonstrated robust performance across both asset sets. For the first set of assets, it achieved a total return of 4910%, significantly outperforming several static methods. For the second set of assets, the dynamic strategy achieved a total return of 5993%, showing superior performance compared to the maximum diversification method, although it was slightly less than the equal investment, equal risk contribution and minimum variance methods. These results highlight the effectiveness of the dynamic strategy in adapting to market changes and optimizing returns.

In addition, the dynamic strategy consistently delivered the second highest Sharpe ratio compared to static methods for the first set of assets, indicating better risk-adjusted performance. For the second set of assets, the dynamic strategy achieved the second highest Sharpe ratio, underscoring its robustness in managing risk and adapting to state changes. Although its return was second to last.

Although promising, our study identifies several areas for future research and improvement. Incorporating additional macroeconomic and financial factors could refine state classification and enhance the predictive power of the Bayesian Markov transition matrix, improving portfolio diversification and performance.

Exploring alternative models, such as Hidden Markov Models (HMM) or Regime-Switching Models, might capture market dynamics more accurately, leading to better investment strategies. Expanding the analysis to include a broader range of assets, such as European bonds, commodities, and other cryptocurrencies, could further diversify the portfolio and improve risk management. Furthermore, incorporating more portfolio methods could help build a more sophisticated and robust dynamic portfolio, enhancing its ability to adapt to varying market conditions and optimize performance.

Developing more real-time implementation and testing frameworks is crucial to assess the practical applicability of the dynamic portfolio strategy in live trading environments. This would help evaluate the strategy's performance under actual market conditions and its responsiveness to market changes.

Several limitations should be acknowledged. The study's reliance on historical data assumes that past market behavior will repeat, which may not always hold true. The Bayesian Markov transition matrix and portfolio optimization methods are based on specific assumptions that may not capture all market dynamics and investor behavior. Additionally, the computational intensity of the dynamic strategy and the exclusion of transaction costs and other practical constraints may impact real-world performance.

In conclusion, while our study demonstrates the potential benefits of a dynamic portfolio approach, addressing its limitations and exploring the identified areas for future research could further enhance its effectiveness and applicability. The dynamic portfolio strategy shows significant promise in improving returns and sharpe ratios through better future state prediction and adaptation, but further research and refinement are needed to enhance its practical applicability across diverse market conditions.

4 DYNAMIC GRAPH-BASED TEMPORAL CORRELATION ANALYSIS FOR PAIR TRADING

In our research, we explore the confluence of deep learning technologies and their applications in financial markets, focusing particularly on pair trading strategies. Pair trading is a cornerstone in the field of quantitative finance and offers an ideal setting to implement cutting-edge deep learning methods. One of the key challenges in pair trading is identifying time-dependent correlations among different financial instruments, which requires the effective amalgamation of various types of data, also known as modalities. To address this complexity, we introduce a new framework called Multi-modal Temporal Relation Graph Learning (MTRGL). This framework consists of two primary elements: a dynamic graph that integrates both time series data such as price movements and categorical data such as industry sectors. It also employs a memory-enhanced dynamic graph neural network for its neural architecture. This setup reframes the issue of identifying temporal correlations as a temporal graph link prediction problem, an approach that has shown empirical benefits. Real-world data tests confirm that MTRGL consistently outperforms existing methods, highlighting its potential to improve the reliability and precision of automated pair trading systems.

4.1 INTRODUCTION

Pair trading is a fundamental investment tactic that capitalizes on correcting pricing imbalances between closely linked assets or financial markets. As noted by Gatev et al. (2006), this strategy focuses on identifying transient discrepancies in prices, allowing traders to strategically position themselves to profit from an anticipated price convergence. This method involves simultaneously taking long and short positions in correlated assets or markets. A long position occurs when an investor purchases an asset and anticipates an increase in its value. In contrast, a short position is adopted when an investor sells an asset they have borrowed, predicting a drop in its price, with the intention of repurchasing it at a lower cost later. This dual strategy enables traders to benefit from expected price adjustments over time. A graphical representation of the pair trading strategy is shown in Figure [4.1]. Pair trading plays a significant role in financial investments, as it capitalizes on market inefficiencies and erroneous asset pricing. Using these fleeting price variances, traders have the potential to make profits while the inherent value of the assets reverts to normal. In addition, pair trading contributes to market efficiency by narrowing price gaps and enhancing price discovery mechanisms. The skilled identification and exploitation of these price differences can lead to better returns and more effective risk management in investment portfolios.

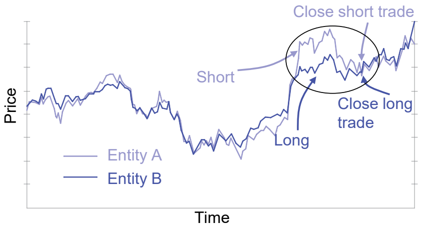


Figure 4.1: An illustration showing the Pair Trading Strategy involves two correlated entities, labeled A and B, whose prices generally move in tandem. The highlighted section marks a brief phase where the prices of A and B diverge, possibly due to market volatility or inefficiencies. In this phase of divergence, a trader can exploit the situation by taking a long position on entity B (betting its price will rise) and a short position on entity A (betting its price will fall). The strategy aims to profit from this divergence when the price movements of A and B realign.

The efficacy of a pair trading strategy fundamentally hinges on the accurate identification of temporal correlations between varied financial entities, such as assets or markets. This strategy aims to identify pairs that exhibit a high degree of correlation, often demonstrated by synchronized price movements that periodically converge or diverge. However, discerning these temporal correlations within the dynamic landscape of financial markets is a complex task, given the market's ever-changing nature and the vast array of potential pairs. This complexity requires the use of advanced quantitative analysis, data mining techniques, and sophisticated statistical methods to uncover patterns, correlations, and complex interdependencies. The transient nature of these relationships, which can evolve due to changing market conditions, regulatory changes, and macroeconomic factors, adds to the challenge.

Traditionally, the identification of temporal correlations among financial entities was a manual process, conducted by teams of experts in financial institutions who closely analyzed market behaviors. These traditional methods focused on developing statistical techniques to identify temporal correlations, but such methods are static and limited to basic data analysis such as price trends. The rise of machine learning has ushered in a new era, highlighting its potential in identifying crucial temporal correlations for pair trading strategies. Machine learning algorithms have shown superiority in processing large datasets, identifying nonlinear relationships, and detecting complex dependencies, often outperforming traditional statistical methods and human capabilities. Despite the promising potential of machine learning to identify correlated assets and markets for pair trading, the field remains underexplored, which calls for thorough research and development to fully harness its capabilities.

The application of machine learning to identify temporal correlations in pair trading presents significant challenges. The first challenge is the reliance of machine learning models on high-quality, diverse feature-rich data. Although the financial sector provides ample data like stock prices, the simplicity of such time series data may limit the effectiveness of advanced machine learning models, which thrive on processing complex, multidimensional inputs. To effectively leverage machine learning in pair trading, it is crucial to develop innovative methods that integrate additional information from various sources, enriching the learning process. The second challenge is the dynamic, ever-changing nature of financial markets, influenced by factors like economic policies, geopolitical events, and market sentiment. These changes can significantly alter asset pair relationships, making them volatile and time sensitive. Machine learning models must therefore be designed to continuously adapt and learn from these changes, which requires sophisticated algorithms and models.

In this study, we explore machine learning applications in the context of pair trading. We introduce a new framework, the Multi-Modal Temporal Relation Graph Learning (MMTRGL), which uniquely combines high-dimensional feature data and time-series data to identify temporal correlations among financial entities. This approach utilizes temporal graph learning, a method that has recently shown promising results. The contributions of this paper are threefold:

We examine the challenges and requirements of applying machine learning to identify temporal correlations in pair trading, emphasizing the importance of integrating information from various sources.

We present MMTRGL, a novel framework that seamlessly assimilates information from different modalities, bridging the gap between temporal correlation identification and temporal graph learning. This framework includes a mechanism for constructing a dynamic graph that incorporates both time series data (like price trends) and discrete feature information (such as sector classifications), and a neural model equipped with a memory-based dynamic graph neural network, effective in temporal graph learning. MMTRGL adeptly tackles the challenges of applying machine learning in pair trading.

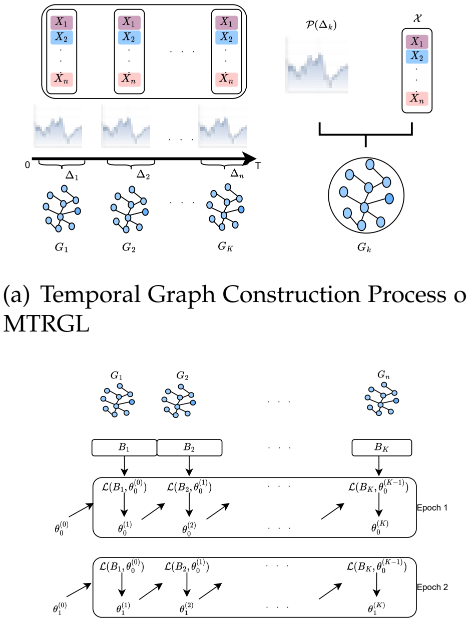
Through empirical analysis using real-world data, we show that MMTRGL consistently outperforms existing benchmark methods in identifying and inferring temporal correlations. This highlights its potential in detecting pair trading opportunities and exploiting pricing anomalies. In addition, an ablation study assesses the impact of excluding feature information on entities and their higher-order structural relationships. The results validate the significance of this information in inferring temporal correlations, confirming MMTRGL's effectiveness, and offering insights for future machine learning applications in finance.

4.2 RELATED WORK

The field of machine learning, particularly neural network-based models, has significantly benefited from advances in computational resources. These models have demonstrated remarkable proficiency in deriving insights from complex datasets, including those involving images, languages, and networks. The financial markets, with their intricate and dynamic characteristics, are particularly well suited for the application of machine learning techniques. The potential for financial benefits and growth has attracted considerable attention from the research community to this area. For further details on this topic, the reader is directed to the work of Ozbayoglu (2020) [ozbayoglu2020deep].

Much of the existing research in this domain has focused on stock price data, which represents a relatively simplistic form of time-series data. This focus has limited the ability to fully exploit the potential of machine learning in uncovering significant patterns within these markets.

In more recent developments, large language models like ChatGPT [chen2023chatgpt] have been used to extract features from financial news, which aids the learning process. However, the integration of these features with stock price data often involves a basic concatenation approach, which presents challenges in effectively combining different types of information for financial market analysis. This paper addresses such challenges by focusing on the identification of temporal correlations for pair trading. We introduce MTRGL, a novel approach that provides a more sophisticated method for merging multimodal data, thereby revealing deeper structural insights into financial markets.



(a) Temporal Graph Construction Process of MTRGL

(b) Training Pipeline of MTRGL

Figure 4.2: Depiction of Temporal Graph Construction and Training Methodology in MTRGL. Figure 4.2(a) illustrates the segmentation of time series data from $[0, T]$ into segments $\Delta_1, ..., \Delta_n$ and their integration with feature data to form a series of temporal investment graphs $G_1, ..., G_n$. In contrast, Figure 4.2(b) visualizes the training mechanics of MTRGL, where the graphs are processed in batches for model refinement. The event batch losses B_i, represented as $\mathcal{L}(B_i, \theta_k^{(j)})$, are computed using parameters $\theta_k^{(j)}$, derived from the k-th epoch and j-th iteration. The parameters obtained from the last iteration of an epoch are used as starting points for the next epoch, indicated by $\theta_k^{(K-1)} = \theta_{k+1}^{(0)}$.

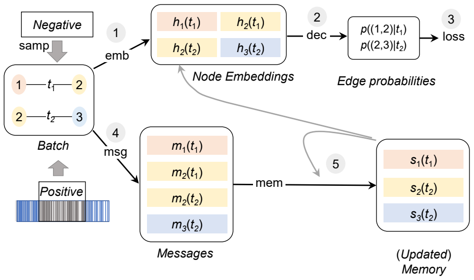


Figure 4.3: Depiction of Memory Update Mechanics in MTRGL. Each graph, labeled as G_i, undergoes processing as an event batch. This process not only updates the model but also rejuvenates the memory for subsequent batches.

4.3 METHODOLOGY

Let us define $\mathcal{A} = A_1, A_2, ..., A_n$ as a set of n entities under scrutiny, representing companies, assets, or markets. Each entity A_i is associated with a dynamic feature vector $X_i(t) = (x^{(1)}(t), x^{(2)}(t), ..., x^{(m)}(t))$, where $x^{(j)}(t)$ denotes the j-th feature of entity A_i at time t. $\mathcal{X}(t)$ symbolizes the aggregate of feature vectors for $\mathcal{A}$ at time t. Furthermore, $P_i(t) : t \mapsto \mathbb{R}$ represents the trading price time series for entity A_i, with $P_i([0, t])$ denoting the time series information for entity A_i within the interval $[0, t]$. The set $\mathcal{P} = P_i(t)$ symbolizes the time series associated with the entities in $\mathcal{A}$. Additionally, $\mathcal{P}([0, t]) = P_i([0, t]) | P_i(.) \in \mathcal{P}$ denotes the collection of time series information for all entities within the interval $[0, t]$.

A key element in pair trading is the identification of temporal correlations among entities based on their historical and feature data. To formalize, let $S(P_i(t), P_j(t))$ represent a predefined measure of correlation between two time series, such as the normalized historical difference (NHD) according to [23], and let γ be a predetermined threshold. The objective is to identify pairs of entities A_i, A_j within the set $\mathcal{A}$, with their feature vectors $\mathcal{X}$ and time series data $\mathcal{P}([0, T])$, such that $S(p_i([T, T + \delta]), p_j([T, T + \delta])) \geq \gamma$ for a certain $\delta > 0$.

Overview

This document introduces MTRGL, a novel framework designed to pinpoint correlated pairs as a temporal graph prediction challenge. This method integrates both time series data and specific features of companies into a cohesive graph structure. Initially, MTRGL assembled a sequence of graphs, representing each entity as a node, and temporal information as dynamic links. This methodology offers an elaborate sequential portrayal of each entity's historical and current data. Moreover, MTRGL utilize a custom-designed memory-based dynamic graph neural network model, proven effective in temporal graph inference tasks. The primary aspect of MTRGL is its graph training phase, which uses a contrastive learning technique for model training via the prediction of future inter-company edges. This training strategy enhances the model's forecast accuracy for potential inter-company relations through an evolving sequence of temporal graphs, resulting in precise and durable future projections. The design of the MTRGL framework amalgamates time series and feature data, presenting a distinct model capable of handling the dynamic nature of the involved data. Further sections elaborate on each algorithmic element in detail.

Temporal Graph Construction

Given data within the time frame $[0, T]$, it is initially partitioned into K smaller intervals, $\Omega = \Delta_{i i=1,...,K}$. We postulate that each interval Δ_i is uniform in size, $|\Delta_i| = \delta = \frac{T}{K}$, simplifying Ω as $[0, \delta), [\delta, 2\delta), ..., [(K-1)\delta, K\delta)$.

Assuming $S(P_i(t), P_j(t)) \mapsto [0, 1]$ as a measure function (for simplicity, we use the NHD) for correlating two time series, and γ as a preset threshold, we construct the k-th temporal graph G_k corresponding to the interval Δ_k as follows.

- For each entity A_i, we introduce a vertex v_i into the graph.
- An edge $e_{ij}(t)$ is formed between vertices v_i and v_j if $S(P_i(\Delta_k), P_j(\Delta_k)) \geq \gamma$.
- We assign a timestamp $t = (k - \frac{1}{2})\delta$ to the aforementioned edge.

The entire process for constructing the temporal graph is encapsulated in Algorithm **??**. The sequence of temporal graphs thus formed is denoted as $\mathcal{G} = G_{i i=1,...,K}$, and the vertex set (entities) within $\mathcal{G}$ is represented as $\mathcal{V} = 1, ..., n$. In alignment with dynamic graph learning conventions, we refer to each edge in $\mathcal{G}$ as an event.

Advantage: Transforming temporal correlation analysis into a temporal graph learning framework offers substantial benefits, as supported by extensive research and literature. The temporal graph prediction issue has been extensively explored, providing a solid base for our approach. Recent developments, especially in temporal graph neural networks, have shown significant efficacy in temporal graph learning challenges. By translating the task of identifying correlated pairs into the domain of temporal graph learning, we achieve seamless integration of both time series and feature data for each trading entity, while also leveraging their interrelations. This dual advantage allows for more comprehensive and accurate interpretations of dynamic market data, thereby enhancing our ability to make more effective and precise predictions in pair trading contexts.

Temporal Graph Neural Networks

Temporal graph neural networks (TGNNs) have demonstrated their strength as neural models, particularly in predicting temporal graph behavior. A specific category, Memory-based TGNNs (MTGNNs), has outperformed their memory-less counterparts [24]. A notable characteristic of MTGNNs is their incorporation of a memory module, which acts as a filter, continuously refining data from both new and historical graph events. Consequently, MTGNNs efficiently grasp extensive dependencies and deliver superior performance across various dynamic graph tasks [25].

In our research, we adopt and customize MTGNNs as our neural model. Following the guidelines in [25] [26], our MTGNN is structured with an encoder-decoder configuration. The encoder in MTGNN consists of three key modules: **m**sg (message), **m**em (memory), and **e**mb (embedding). The encoder's output is then inputted into a decoder (here, a basic two-layer MLP) to perform the inference task. The subsequent sections provide detailed descriptions of each module within the MTGNN.

Encoder

The MTGNN encoder contains three modules: message, memory, and embedding, each described separately for clarity. Figure 4.3 shows the interaction of data among these modules.

MESSAGE Each node i involved in an event (edge) generates a message to update its memory state. For an interaction event $e_{ij}(t)$ between a source node i and a target node j at time t, two messages are formulated:

$$\begin{aligned} m_i(t) &= \mathrm{msg}(s_i(t^-), s_j(t^-), e_{ij}(t), \psi(t - t'_i)), \\ m_j(t) &= \mathrm{msg}(s_j(t^-), s_i(t^-), e_{ij}(t), \psi(t - t'_j)). \end{aligned} \tag{4.1}$$

Here, $s_i(t^-)$ and $s_j(t^-)$ represent the memory states of nodes i and j just before the event at time t. msg$(.)$ is the message function and t'_j signifies the timestamp of the last event involving node j. The time encoding function $\psi(.)$ [27] transforms the time interval into a d-dimensional vector. We opt for the widely used identity message function that outputs the concatenation of input vectors [25] [26].

MEMORY The memory state of a node is refreshed with each event involving that node:

$$s_i(t) = \mathrm{mem}(m_i(t), s_i(t^-)). \tag{4.2}$$

In scenarios where interaction events include two nodes i and j, the memories of both nodes are updated following the event. mem denotes a learnable memory update function. In our implementation, we utilize the gated recurrent unit (GRU) [28]. The memory module serves as an iterative filter, assimilating information from both new and historical temporal graph data. Hence, it capably captures long-range dependencies.

EMBEDDING The objective of the embedding module in the temporal domain is to produce representations $z(t^-)$ right before the occurrence of the subsequent event at any given time t. For this purpose, we utilize a temporal graph attention network with L layers to accumulate information about the neighborhood.

The enhancement of the memory vector of a specific entity, say A_i, is carried out by combining it with the corresponding node feature to form $z_i^{(0)}(t) = s_i(t) + X_i(t)$. This fusion allows the model to benefit from the up-to-date memory state $s_i(t)$ and the feature of the time-variant nodes $X_i(t)$. Following this, for every layer in the range $1 \leq l \leq L$, neighborhood data is consolidated using multi-head attention [27], as detailed in the following equation:

$$\begin{aligned}
z_i^{(l)} &= \text{mlp}^{(l)}(z^{(l-1)}||\tilde{z}k^{(l)}), \\
\tilde{z}k^{(l)} &= \text{mha}^{(l)}(q^{(l)}i(t), K^{(l)}i(t), V^{(l)}i(t)), \\
q^{(l)}(t) &= zi^{(l-1)}||\psi(0), \\
K_i^{(l)}(t) = V^{(l)}i(t) &= \begin{bmatrix} z_i^{(l-1)}||e\pi_i(1)(t\pi_i(1))||\psi(t - t\pi_i(1)) \\ ... \\ z_i^{(l-1)}||e\pi_i(N)(t\pi_i(N))||\psi(t - t\pi_i(N)) \end{bmatrix}.
\end{aligned} \tag{4.3}$$

In these formulas, $||$ represents the operation of vector concatenation, $\text{mlp}(.)$ are single-layer feedforward networks with a dimensionality of d, and $\text{mha}(.)$ stands for multi-head attention functions with queries $q(.)$, keys $K(.)$, and values $V(.)$. The receptive field of each node i is confined to its most recent N events, denoted as $\pi_i = e_{\pi_i(1)}(t_{\pi_i(1)}), e_{\pi_i(1)}(t_{\pi_i(2)}), ..., e_{\pi_i(N)}(t_{\pi_i(N)})$, where π indicates a permutation, and $\pi_i(.)$ symbolizes the temporal neighbors of node i.

Decoder

Our approach simplifies the identification of temporal correlations to a task of predicting temporal links. The decoder calculates the probability of an event $e_{ij}(t)$ based on the pre-event representations $z_i(t^-)$ and $z_j(t^-)$ generated by the encoder. This computation is performed using a two-layer MLP followed by the sigmoid function $\sigma(.)$ as shown below:

$$\hat{p}_{ij}(t) = \sigma(\text{MLP}(z_i(t^-)||z_j(t^-))). \tag{4.4}$$

4.4 MODEL TRAINING AND INFERENCE

For training the model, we utilize binary cross-entropy as the loss function and apply a contrastive learning approach:

$$\mathcal{L} = -\sum_{e_{ij}(t)\in\mathcal{E}} [\log \hat{p}_{ij}(t) + \log(1 - \hat{p}_{ik}(t))], \tag{4.5}$$

wherein a random negative destination node is selected as k, and $\hat{p}_{ik}(t)$ is computed in a similar fashion using $z_i(t^-)$ and $z_k(t^-)$. This method helps the neural model in assimilating contrastive signals extracted from the graphs formed.

In addition, a batch temporal training technique is adopted for efficiency. Consecutive events are grouped into a temporal batch, allowing simultaneous processing of events within that batch. For nodes involved in numerous events in a single batch, we apply the most recent message aggregator, which retains only the latest message for each node in the batch, according to the practices of [25]. To prevent information leakage, where the data of a batch might influence its own event prediction, we employ a lag-one scheme, using the temporal batch B_{i-1} to update the memory state and create embeddings to predict B_i.

Theorem 4.1 (Multi-modal Temporal Relation Graph Construction). *Given a set of entities $\mathcal{A} = \{A_1, ..., A_n\}$, a number of time intervals K, a threshold γ for correlation and a measure of covariance $S(.,.)$, there exists a sequence of temporal graphs $\mathcal{G} = \{G_1, ..., G_K\}$ that represent the relationships between entities over time.*

Proof.

1. **Initialization:**
    - Define an empty list $\mathcal{G}$ to store the constructed graphs.
    - Divide the set $\mathcal{A}$ into K intervals $\{\Delta_1, ..., \Delta_K\}$.
2. **Graph Construction:**
    - For each interval $\Delta_i \in \{\Delta_1, ..., \Delta_K\}$:
        - Initialize graph G_i.
        - Create a vertex v_j for each entity $A_j \in \mathcal{A}$ with feature X_j.
        - Create an edge $e_{ij}(t)$ between vertices v_i and v_j if and only if $S(P_i(\Delta_k), P_j(\Delta_k)) \geq \gamma$.
        - Assign $t = (k - \frac{1}{2})\delta$ and append G_i to $\mathcal{G}$.
3. **Return:**

- The sequence of temporal graphs $\mathcal{G}$ represents the multi-modal temporal relation graph constructed over the specified intervals and correlation threshold.

Q.E.D. □

Theorem 4.2 (Training Procedure for Temporal Graph-based Model). *Given a temporal graph sequence $\mathcal{G} = \{G_1, ..., G_K\}$, a number of epochs T, and initial memory state S_0 and model parameter $\theta_0^{(0)}$, a neural model can be trained using contrastive learning.*

Proof.

1. **Initialization:**
   - Initialize memory vectors $S_0 \leftarrow 0$.
   - Initialize model parameters $\theta_0^{(0)} \leftarrow 0$.
2. **Epoch Iteration:**
   - For each epoch $t = 1$ to T:
     - For each graph $G_i \in \{G_2, ..., G_K\}$:
       - ∗ Define positive batch $B_i^+ \leftarrow \mathcal{E}_i$.
       - ∗ Sample negative events to create negative batch B_i^-.
       - ∗ Combine batches to form $\bar{B}_i = B_i^- \cup B_i^+$.
       - ∗ Use the temporal batch from the previous iteration to update the memory and embedding:
         - · $\bar{B}_{i-1} \leftarrow$ Temporal batch from last iteration.
         - · $M_i = \text{msg}(S_{i-1}, \bar{B}_{i-1})$ (Compute messages for the events in the batch).
         - · $S_i = \text{mem}(S_{i-1}, M_i)$ (Update memory with the computed message).
         - · $H_i = \text{emb}(S_i, A_i)$ (Compute the embedding).
       - ∗ Compute the loss $\mathcal{L}(H_i, B_i)$ as defined in Equation (4.5) and update the model parameter using a training algorithm (e.g., backpropagation and Adam).
3. **Return:**
   - The trained neural model captures the temporal dynamics and relationships encoded in the sequence of temporal graphs $\mathcal{G}$.

Q.E.D. □

4.5 CONVERGENCE ANALYSIS

In this section, we analyze the convergence properties of the minibatch stochastic gradient descent (SGD) algorithm applied to our model. We utilize a temporal graph neural network model, which involves layers of graph attention and a two-layer MLP for decoding. Our analysis assumes that the model's loss function can be expressed as a sum of convex functions and is $L_{\max}$-smooth. Most lemmas and theorems were adapted from [29], and we have revised and tailored them for this thesis.

Minibatch SGD Algorithm

Minibatch SGD is a variant of the gradient descent algorithm that balances the benefits of stochastic and batch gradient descent by using small, random subsets of the data for each parameter update. Here is the basic algorithm:

Algorithm 1 Minibatch Stochastic Gradient Descent

```
Initialize parameters $x^0$.
for each iteration $t = 0, 1, 2, \ldots, T-1$ do
    Sample a minibatch $B_t$ from the data.
    Compute the gradient estimate: $\nabla f_{B_t}(x^t) = \frac{1}{|B_t|} \sum_{i \in B_t} \nabla f_i(x^t)$.
    Update the parameters: $x^{t+1} = x^t - \eta_t \nabla f_{B_t}(x^t)$.
end for
return $x^T$.
```

Assumptions

To analyze the convergence of minibatch SGD, we make the following assumptions:

Definition 4.3 (Convexity)**.** A function $f(x)$ is convex if for all x, y and $\theta \in [0,1]$, the following holds:

$$f(\theta x + (1-\theta)y) \leq \theta f(x) + (1-\theta) f(y). \tag{4.6}$$

This implies that the function lies below the line connecting any two points on its graph, indicating that there are no local minima outside the global minimum.

Definition 4.4 ($L_{\max}$-Smoothness)**.** A function $f(x)$ is $L_{\max}$-smooth if for all x, y, the following holds:

$$\|\nabla f(x) - \nabla f(y)\| \leq L_{\max} \|x - y\|. \tag{4.7}$$

This condition ensures that the gradient does not change too rapidly, providing an upper bound on the Lipschitz constant of the gradient.

Notice that while the binary cross-entropy loss function in equation (4.5) is convex with respect to the model's parameters in a simple single layer setup, adding multiple layers and increasing the input dimensions of the loss function result in a non-convex optimization landscape. Thus we need the convex assumptions.

Remark 4.5 (Minibatch Distribution). We impose that the batches B are sampled uniformly among all subsets of size b in $\{1, \dots, n\}$. This means each batch is sampled with probability

$$\frac{1}{\binom{n}{b}} = \frac{(n-b)!b!}{n!}. \tag{4.8}$$

The expected minibatch gradient is given by

$$\mathbb{E}[\nabla f_B(x)] = \frac{1}{\binom{n}{b}} \sum_{B \subset \{1,\dots,n\}, |B|=b} \nabla f_B(x), \tag{4.9}$$

where

$$\nabla f_B(x) = \frac{1}{b} \sum_{i \in B} \nabla f_i(x). \tag{4.10}$$

Each sample i appears in $\binom{n-1}{b-1}$ batches. Therefore, the expected gradient is:

$$\mathbb{E}[\nabla f_B(x)] = \frac{1}{\binom{n}{b}} \sum_{i=1}^{n} \nabla f_i(x) \cdot \binom{n-1}{b-1} \cdot \frac{1}{b} \tag{4.11}$$

$$= \frac{1}{b} \cdot \frac{\binom{n-1}{b-1}}{\binom{n}{b}} \sum_{i=1}^{n} \nabla f_i(x) \tag{4.12}$$

$$= \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x) = \nabla f(x). \tag{4.13}$$

Thus, the expected minibatch gradient equals the full gradient, $\nabla f(x)$.

Lemmas and Theorems

Definition 4.6 (Minibatch Gradient Noise)**.** Let Assumption (Sum of $L_{\max}$-Smooth) hold, and let $b \in \{1, \dots, n\}$. We define the minibatch gradient noise as

$$\sigma_b^* \triangleq \inf_{x^* \in \arg\min f} \operatorname{Var}[\nabla f_B(x^*)], \tag{4.14}$$

where B is sampled according to the distribution specified in Remark 1 .

Lemma 4.7. *If $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth, then for all $x, y \in \mathbb{R}^d$,*

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2. \tag{4.15}$$

Proof. Let $x, y \in \mathbb{R}^d$ be fixed. Define $\varphi(t) := f(x + t(y - x))$. Using the Fundamental Theorem of Calculus, we have

$$f(y) = f(x) + \int_0^1 \langle \nabla f(x + t(y - x)), y - x \rangle \, dt. \tag{4.16}$$

This can be rewritten as

$$\begin{aligned} f(y) = f(x) &+ \langle \nabla f(x), y - x \rangle \\ &+ \int_0^1 \langle \nabla f(x + t(y - x)) - \nabla f(x), y - x \rangle \, dt. \end{aligned} \tag{4.17}$$

Applying the Cauchy-Schwarz inequality to the integrand gives

$$\begin{aligned} f(y) \leq f(x) &+ \langle \nabla f(x), y - x \rangle \\ &+ \int_0^1 \|\nabla f(x + t(y - x)) - \nabla f(x)\| \|y - x\| \, dt. \end{aligned} \tag{4.18}$$

Using the L-smoothness condition $\|\nabla f(x + t(y - x)) - \nabla f(x)\| \leq Lt\|y - x\|$, we have

$$\begin{aligned} f(y) \leq f(x) &+ \langle \nabla f(x), y - x \rangle \\ &+ \int_0^1 Lt\|y - x\|^2 \, dt. \end{aligned} \tag{4.19}$$

Evaluating the integral, we get

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2}\|y - x\|^2. \tag{4.20}$$

□

Lemma 4.8 (Convex and L-Smooth Function Property). *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex and L-smooth function. Then, for all $x, y \in \mathbb{R}^d$, we have*

$$\frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2 \leq f(y) - f(x) - \langle \nabla f(x), y - x \rangle.$$

Proof. To prove the inequality, fix $x, y \in \mathbb{R}^d$ and start by using the convexity and smoothness of f to write, for every $z \in \mathbb{R}^d$,

$$f(x) - f(y) = f(x) - f(z) + f(z) - f(y). \tag{4.21}$$

Using the first-order condition for convexity and lemma 4.7, we have:

$$f(x) - f(z) \leq \langle \nabla f(x), x - z \rangle, \tag{4.22}$$

and

$$f(z) - f(y) \leq \langle \nabla f(y), z - y \rangle + \frac{L}{2}\|z - y\|^2. \tag{4.23}$$

Substituting these into the equation gives:

$$f(x) - f(y) \leq \langle \nabla f(x), x - z \rangle + \langle \nabla f(y), z - y \rangle + \frac{L}{2}\|z - y\|^2. \tag{4.24}$$

To obtain the tightest upper bound, minimize the right-hand side with respect to z, by differentiating and setting the gradient to zero, which gives:

$$z = y - \frac{1}{L}(\nabla f(y) - \nabla f(x)). \tag{4.25}$$

Substituting this z back, we have:

$$f(x) - f(y) \leq \langle \nabla f(x), x - z \rangle + \langle \nabla f(y), z - y \rangle + \frac{L}{2}\|z - y\|^2. \tag{4.26}$$

This simplifies to:

$$= \langle \nabla f(x), x - y \rangle - \frac{1}{L}\|\nabla f(y) - \nabla f(x)\|^2 + \frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2, \tag{4.27}$$

which further simplifies to:

$$= \langle \nabla f(x), x - y \rangle - \frac{1}{2L}\|\nabla f(y) - \nabla f(x)\|^2. \tag{4.28}$$

Thus, the inequality is proved. □

Definition 4.9. Let Assumption (Sum of $L_{\max}$–Smooth) hold, and let $b \in \{1, \ldots, n\}$. We say that f is L_b-smooth in expectation if for all $x, y \in \mathbb{R}^d$,

$$\frac{1}{2L_b}\mathbb{E}\left[\|\nabla f_B(y) - \nabla f_B(x)\|^2\right] \leq f(y) - f(x) - \langle \nabla f(x), y - x \rangle,$$

where B is sampled according to Remark 1.

Lemma 4.10. *Let Assumptions (Sum of $L_{\max}$-Smooth) and (Sum of Convex) hold. Then for all $x, y \in \mathbb{R}^d$, the function f is L_b-smooth in expectation over minibatches, such that*

$$\frac{1}{2L_b}\mathbb{E}_{B_t}\left[\|\nabla f_{B_t}(y) - \nabla f_{B_t}(x)\|^2\right] \leq f(y) - f(x) - \langle \nabla f(x), y - x\rangle, \quad (4.29)$$

where B_t is a minibatch sampled uniformly from the dataset and

$$L_b = \frac{n(b-1)}{b(n-1)}L + \frac{n-b}{b(n-1)}L_{\max}. \quad (4.30)$$

Proof. See Proposition 3.8 in [30]. □

Lemma 4.11. *If Assumptions (Sum of $L_{\max}$-Smooth) and (Sum of Convex) hold, then for every $x \in \mathbb{R}^d$ and every $x^* \in \arg\min f$, we have*

$$\frac{1}{2L_b}\mathbb{E}\left[\|\nabla f_{B_t}(x) - \nabla f_{B_t}(x^*)\|^2\right] \leq f(x) - \inf f,$$

where L_b is the smoothness constant, and B_t represents a minibatch sampled from the dataset.

Proof. To prove the lemma, apply the L_b-smoothness condition for the function f using minibatches. Let $y = x^*$, where x^* is a minimizer of f, so that $f(x^*) = \inf f$ and $\nabla f(x^*) = 0$. By the lemma 4.10, we have:

$$\frac{1}{2L_b}\mathbb{E}_{B_t}\left[\|\nabla f_{B_t}(y) - \nabla f_{B_t}(x)\|^2\right] \leq f(y) - f(x) - \langle \nabla f(x), y - x\rangle. \quad (4.31)$$

Substituting $y = x^*$, we get:

$$\frac{1}{2L_b}\mathbb{E}_{B_t}\left[\|\nabla f_{B_t}(x^*) - \nabla f_{B_t}(x)\|^2\right] \leq f(x^*) - f(x) - \langle \nabla f(x), x^* - x\rangle. \quad (4.32)$$

Since $\nabla f(x^*) = 0$, the inner product $\langle \nabla f(x), x^* - x\rangle$ becomes zero:

$$\frac{1}{2L_b}\mathbb{E}_{B_t}\left[\|\nabla f_{B_t}(x^*) - \nabla f_{B_t}(x)\|^2\right] \leq f(x^*) - f(x). \quad (4.33)$$

Recognizing that $f(x^*) = \inf f$, this simplifies to:

$$\frac{1}{2L_b}\mathbb{E}\left[\|\nabla f_{B_t}(x) - \nabla f_{B_t}(x^*)\|^2\right] \leq f(x) - \inf f. \quad (4.34)$$

Thus, the expected squared norm of the gradient difference over a minibatch is bounded by the difference between the function value at x and its minimum. □

Lemma 4.12. *Let Assumptions (Sum of $L_{\max}$-Smooth) and (Sum of Convex) hold. Then*

$$\mathbb{E}\left[\|\nabla f_B(x)\|^2\right] \leq 4L_b(f(x) - \inf f) + 2\sigma_b^*, \tag{4.35}$$

where B is a minibatch sampled uniformly from the dataset.

Proof. Start by expressing the squared norm of the minibatch gradient as:

$$\|\nabla f_B(x)\|^2 = \|\nabla f_B(x) - \nabla f_B(x^*) + \nabla f_B(x^*)\|^2 . \tag{4.36}$$

Applying the inequality $(a+b)^2 \leq 2a^2 + 2b^2$, this becomes:

$$\|\nabla f_B(x)\|^2 \leq 2\|\nabla f_B(x) - \nabla f_B(x^*)\|^2 + 2\|\nabla f_B(x^*)\|^2 . \tag{4.37}$$

Taking the expectation over the minibatch B, we obtain:

$$\mathbb{E}_B\left[\|\nabla f_B(x)\|^2\right] \leq 2\mathbb{E}_B\left[\|\nabla f_B(x) - \nabla f_B(x^*)\|^2\right] + 2\mathbb{E}_B\left[\|\nabla f_B(x^*)\|^2\right] . \tag{4.38}$$

Using lemma 4.11 for minibatches, we have:

$$\mathbb{E}_B\left[\|\nabla f_B(x) - \nabla f_B(x^*)\|^2\right] \leq 2L_b(f(x) - \inf f). \tag{4.39}$$

The second term $\mathbb{E}_B\left[\|\nabla f_B(x^*)\|^2\right]$ represents the variance of the gradient noise at the minimizer, which is σ_b^*.

Thus, substituting these back gives:

$$\mathbb{E}_B\left[\|\nabla f_B(x)\|^2\right] \leq 4L_b(f(x) - \inf f) + 2\sigma_b^*. \tag{4.40}$$

This completes the proof. □

Theorem 4.13. *Let Assumptions (Sum of $L_{\max}$-Smooth) and (Sum of Convex) hold. Consider $(x_t)_{t\in\mathbb{N}}$ a sequence generated by the (MiniSGD) algorithm, with a sequence of step sizes satisfying $0 < \gamma_t \leq \frac{1}{4L_b}$. It follows that for every $T \geq 1$, $x^* \in \arg\min f$, and $\bar{x}_T \overset{\text{def}}{=} \frac{1}{\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t x_t$,*

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\sum_{t=0}^{T-1}\gamma_t} + \frac{2\sigma_b^*\sum_{t=0}^{T-1}\gamma_t^2}{\sum_{t=0}^{T-1}\gamma_t}. \tag{4.41}$$

Proof. Let $x^* \in \arg\min f$, so we have $\sigma_b^* = \mathbb{V}[\nabla f_B(x^*)]$. Start by analyzing the behavior of $\|x_{t+1} - x^*\|^2$. By developing the squares, we obtain

$$\|x_{t+1} - x^*\|^2 = \|x_t - x^*\|^2 - 2\gamma_t \langle \nabla f_{B_t}(x_t), x_t - x^* \rangle + \gamma_t^2 \|\nabla f_{B_t}(x_t)\|^2. \tag{4.42}$$

Taking the expectation conditioned on x_t, using the convexity of f and lemma 4.12, we can write

$$\begin{aligned} \mathbb{E}\left[\|x_{t+1} - x^*\|^2 \mid x_t\right] = \|x_t - x^*\|^2 &+ 2\gamma_t \langle \nabla f(x_t), x^* - x_t \rangle \\ &+ \gamma_t^2 \mathbb{E}\left[\|\nabla f_{B_t}(x_t)\|^2 \mid x_t\right]. \end{aligned} \tag{4.43}$$

Using the first order condition of convexity of f, this becomes

$$\leq \|x_t - x^*\|^2 - 2\gamma_t (f(x_t) - \inf f) + \gamma_t^2 \mathbb{E}\left[\|\nabla f_{B_t}(x_t)\|^2 \mid x_t\right]. \tag{4.44}$$

Applying Lemma 4.12, we have

$$\leq \|x_t - x^*\|^2 + 2\gamma_t (2\gamma_t L_b - 1)(f(x_t) - \inf f) + 2\gamma_t^2 \sigma_b^*. \tag{4.45}$$

Given that $\gamma_t \leq \frac{1}{4L_b}$, it follows

$$\leq \|x_t - x^*\|^2 - \gamma_t (f(x_t) - \inf f) + 2\gamma_t^2 \sigma_b^*. \tag{4.46}$$

Rearranging and taking the expectation, we get

$$\begin{aligned} \gamma_t \mathbb{E}\left[f(x_t) - \inf f\right] \leq \mathbb{E}&\left[\|x_t - x^*\|^2\right] \\ &- \mathbb{E}\left[\|x_{t+1} - x^*\|^2\right] + 2\gamma_t^2 \sigma_b^*. \end{aligned} \tag{4.47}$$

Summing over $t = 0, \ldots, T-1$ and using telescopic cancellation gives

$$\sum_{t=0}^{T-1} \gamma_t \mathbb{E}\left[f(x_t) - \inf f\right] \leq \|x_0 - x^*\|^2 - \mathbb{E}\left[\|x_T - x^*\|^2\right] + 2\sigma_b^* \sum_{t=0}^{T-1} \gamma_t^2. \tag{4.48}$$

Since $\mathbb{E}\left[\|x_T - x^*\|^2\right] \geq 0$, dividing both sides by $\sum_{t=0}^{T-1} \gamma_t$ gives:

$$\frac{1}{\sum_{t=0}^{T-1} \gamma_t} \sum_{t=0}^{T-1} \gamma_t \mathbb{E}\left[f(x_t) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\sum_{t=0}^{T-1} \gamma_t} + \frac{2\sigma_b^* \sum_{t=0}^{T-1} \gamma_t^2}{\sum_{t=0}^{T-1} \gamma_t}. \tag{4.49}$$

Finally, define $\bar{x}_T \overset{\text{def}}{=} \frac{1}{\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t x_t$ and use the convexity of f together with Jensen's inequality to conclude

$$\begin{aligned}\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] &\leq \mathbb{E}\left[\frac{1}{\sum_{t=0}^{T-1}\gamma_t}\sum_{t=0}^{T-1}\gamma_t(f(x_t) - \inf f)\right] \\ &\leq \frac{\|x_0 - x^*\|^2}{\sum_{t=0}^{T-1}\gamma_t} + \frac{2\sigma_b^*\sum_{t=0}^{T-1}\gamma_t^2}{\sum_{t=0}^{T-1}\gamma_t}.\end{aligned} \tag{4.50}$$

□

Theorem 4.14. *Let Assumptions (Sum of $L_{\max}$-Smooth) and (Sum of Convex) hold. Consider $(x_t)_{t\in\mathbb{N}}$ a sequence generated by the (MiniSGD) algorithm, with a sequence of constant step sizes $\gamma_t \equiv \gamma \leq \frac{1}{4L_b}$. It follows that for every $T \geq 1$, $x^* \in \arg\min f$ and $\bar{x}_T \overset{\text{def}}{=} \frac{1}{T}\sum_{t=0}^{T-1} x_t$,*

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\gamma T} + 2\gamma\sigma_b^*. \tag{4.51}$$

In particular, if for a fixed horizon $T \geq 1$ we set $\gamma = \frac{\gamma_0}{\sqrt{T}}$ for some $\gamma_0 \leq \frac{1}{4L_b}$, then

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\gamma_0\sqrt{T}} + \frac{2\gamma_0\sigma_b^*}{\sqrt{T}} = O\left(\frac{1}{\sqrt{T}}\right). \tag{4.52}$$

Proof. This result is a direct consequence of Theorem 4.13. By setting the step size $\gamma_t = \gamma$, the summations become:

$$\sum_{t=0}^{T-1}\gamma_t = \gamma T \quad \text{and} \quad \sum_{t=0}^{T-1}\gamma_t^2 = \gamma^2 T. \tag{4.53}$$

Substituting these into the bound from Theorem 4.13:

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\gamma T} + \frac{2\sigma_b^*\gamma^2 T}{\gamma T} = \frac{\|x_0 - x^*\|^2}{\gamma T} + 2\gamma\sigma_b^*. \tag{4.54}$$

Now, consider setting $\gamma = \frac{\gamma_0}{\sqrt{T}}$. Then the bound becomes:

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \frac{\|x_0 - x^*\|^2}{\gamma_0\sqrt{T}} + \frac{2\gamma_0\sigma_b^*}{\sqrt{T}}. \tag{4.55}$$

This shows that the expected difference between the function value at $\bar{x}_T$ and the infimum decreases at a rate of $O\left(\frac{1}{\sqrt{T}}\right)$. This means that as the number of iterations T increases, the expected error reduces proportionally to $1/\sqrt{T}$, implying that to halve the error, the number of iterations needs to be quadrupled. □

Corollary 4.15. *($O(1/\varepsilon^2)$ Complexity). Consider the setting of Theorem 6.9. For every $\varepsilon > 0$, we can guarantee that*

$$\mathbb{E}\left[f(\bar{x}_T) - \inf f\right] \leq \varepsilon \tag{4.56}$$

provided that

$$\gamma = \frac{\gamma_0}{\sqrt{T}}, \quad \gamma_0 = \min\left(\frac{1}{4L_b}, \frac{\|x_0 - x^*\|}{\sqrt{2\sigma_b^*}}\right), \tag{4.57}$$

and

$$T \geq \left(\frac{\|x_0 - x^*\|}{\sqrt{\sigma_b^*}} + \|x_0 - x^*\|^2 L_b\right)^2 \frac{1}{16\varepsilon^2}. \tag{4.58}$$

Proof. This result is a direct consequence of Theorem 6.9 and Lemma A.1 from [29], with $A = \|x_0 - x^*\|^2$, $B = 2\sigma_b^*$, and $C = 4L_b$. □

The $O(1/\varepsilon^2)$ complexity indicates that the number of iterations T required for the algorithm to reach an expected error less than ε in the function value is inversely proportional to the square of ε. This implies that as the desired accuracy ε decreases, the number of iterations needed increases quadratically.

4.6 experimental analysis

This section provides a detailed examination of the method we have proposed, addressing primarily two essential research questions. **Q1**: What is the comparative performance of our method compared to existing benchmarks to automatically identify temporal correlations between entities? **Q2**: Does the inclusion of multimodal data contribute to an improved solution?

Dataset and Preparation

We conducted our evaluation using financial data available from Yahoo Finance and Naver Finance. Our focus was on three specific indices: the Korea Composite Stock Price Index (KOSPI), the Standard & Poor's 500 Index (S&P 500), and the Heng Seng Index (HSI), which provided a varied analysis in different markets. The study intentionally concentrates on the period before the pandemic, precisely the daily closing prices from 2015 to 2019, to avoid the unpredictable impacts caused by the COVID-19 pandemic. Variables such as market capitalization and sector data were included in the feature vectors for each entity.

The data were segregated into training, validation, and testing sets in a chronological manner to ensure a thorough evaluation. The division was in a 60/20/20 (%) ratio. The first three years were utilized for training, the subsequent year for validation, and the last year as a test set. This arrangement simulates the sequential nature of financial data in a realistic evaluation setting.

Comparative Methods

We compared our method with four distinct approaches, which served as our experimental benchmarks. The first uses a basic 2-layer Multi-Layer Perceptron (MLP), relying exclusively on static features of entities. The second employs long-short-term memory (LSTM) [31], using historical pricing data to detect correlations. The third transforms time-series data into a time-frequency domain, applying a Convolutional Neural Network (CNN) for learning processes [32]. Furthermore, a conventional statistical approach based on cointegration (COINT) [31] is included for comparison. All machine learning techniques (ours included) were trained until they converged or reached a maximum of 50 epochs, using Adam optimizer with a learning rate of 0.01 and a weight decay of 0.0001.

Evaluation Metrics

We utilized two primary metrics for performance evaluation: average precision (AP) and mean absolute percentage error (MAPE). AP is commonly used for models that predict categorical outcomes, indicating the accuracy of correctly identified correlated pairs in our scenario. However, MAPE assesses the accuracy in predicting the actual value of the correlation, providing insights into the model's effectiveness in predicting the degree of correlation between identified pairs.

Findings

Effectiveness Evaluation

We initially assessed the effectiveness of our proposed method, MTRGL. We opted for a one-month interval for both training and analysis phases, which allowed us to capture both short-term fluctuations and long-term trends. The findings of this evaluation are shown in Table 4.1, with values averaged from five separate runs. The entries in bold represent the highest performance levels. The symbol ↑ indicates that higher values indicate

Table 4.1: Performance comparison of MTRGL with baselines.

	KOSPI		**S&P 500**		**HSI**	
	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓
COINT	55.8 ± 0.5	43.6 ± 1.2	56.6 ± 0.9	40.2 ± 1.3	51.4 ± 0.7	32.8 ± 1.2
MLP	48.2 ± 0.4	45.2 ± 1.6	46.8 ± 0.3	43.2 ± 1.4	44.2 ± 0.4	34.4 ± 1.3
CNN	62.7 ± 0.4	32.8 ± 1.3	64.8 ± 0.3	37.2 ± 1.5	64.2 ± 0.8	25.8 ± 1.4
LSTM	61.4 ± 0.3	30.6 ± 1.5	65.6 ± 0.5	34.8 ± 1.3	61.5 ± 0.5	23.3 ± 1.2
MTRGL (ours)	**72.8 ± 0.4**	**24.2 ± 1.0**	**74.2 ± 0.4**	**27.8 ± 1.3**	**69.8 ± 0.7**	**16.8 ± 1.4**

Table 4.2: Performance of MTRGL w./w.o feature information

	KOSPI		**S&P 500**		**HSI**	
	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓
MTRGL-one-hot	63.2 ± 0.9	32.2 ± 1.2	64.3 ± 0.2	34.6 ± 1.5	60.8 ± 0.5	21.6 ± 1.9
MTRGL	**72.8 ± 0.4**	**24.2 ± 1.0**	**74.2 ± 0.4**	**27.8 ± 1.3**	**69.8 ± 0.7**	**16.8 ± 1.4**

better results, while ↓ indicates that lower values are preferable. As depicted in Table 4.1, MTRGL significantly surpasses existing benchmarks in forecasting future correlations among entities.

Component Analysis

An ablation study was conducted to confirm the design choices of MTRGL, focusing on the use of feature data and structural details of the constructed graph. The influence of feature data is detailed in Table 4.2. In the MTRGL-one-hot variant, feature vectors are substituted with unique one-hot identifiers for each entity. The outcomes indicate a considerable decline in performance when feature data are omitted. The impact of structural details is summarized in Table 4.3. In the MTRGL-edgeless variant, the memory module is excluded, relying solely on the memory vector of each entity for the decoder input. This eliminates the concept of temporal neighborhood and, consequently, the structural details. As shown in Table 4.3, removing structural details also results in a reduced performance for MTRGL.

Table 4.3: Performance of MTRGL w./w.o structural information

	KOSPI		**S&P 500**		**HSI**	
	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓	AP(%) ↑	MAPE ↓
MTRGL-edgeless	57.2 ± 1.1	36.2 ± 2.4	58.0 ± 0.9	38.2 ± 2.1	53.6 ± 0.8	26.2 ± 1.4
MTRGL	**72.8 ± 0.4**	**24.2 ± 1.0**	**74.2 ± 0.4**	**27.8 ± 1.3**	**69.8 ± 0.7**	**16.8 ± 1.4**

4.7 conclusion

In this paper, we embark on an exploration of the application of deep learning in the realm of pair trading, a well-regarded quantitative investment strategy. This journey has led to the creation of a unique approach MTRGL, explicitly designed to amalgamate descriptive and time series data, thus optimizing the process of discerning temporal correlations in pair trading. Our empirical evidence shows that MTRGL is highly effective in automatically identifying correlated pairs, exceeding the performance of traditional baselines that rely exclusively on descriptive or time-series data.

A crucial aspect of our approach is the convergence analysis of the deep learning model. By analyzing the convergence behavior, we ensure that the model reliably finds optimal solutions and is robust against variations in input data. The convergence rate indicates how quickly our model can reach a stable solution, providing insights into its efficiency and reliability in identifying trading opportunities.

However, a caveat of our work is that it focuses solely on discerning temporal correlation among entities, without considering the strategy to position the identified correlated pair. This leaves room for future research, where we aim to extend our findings and build a comprehensive machine learning system for pair trading.

Moreover, the novel integration of multi-modal information in our approach extends beyond the scope of this study and pair trading. Its potential impact is significant for other quantitative finance-related problems, and we look forward to seeing how this innovation could transform these areas in the future.

5 OPTIMIZATION AND PERMUTATION ANALYSIS OF CARBON DIOXIDE EMISSION MODELS USING ADVANCED REGRESSION TECHNIQUES

This paper presents a comprehensive study leveraging Support Vector Machine (SVM) regression and Principal Component Regression (PCR) to predict carbon dioxide emissions in a global data set of 62 countries. Our objective is to understand the factors that contribute to carbon dioxide emissions employing permutation importance to identify the most predictive elements. The analysis provides country-specific emission estimates, highlighting diverse national trajectories and pinpointing areas for targeted interventions in climate change mitigation and sustainable development.

The study aims to support policy making with accurate predictions of carbon dioxide emissions, providing nuanced information for formulating effective strategies to address climate change. By combining detailed country-level emission estimates with a broader analysis of contributing factors, this research improves the precision and relevance of policy interventions, contributing significantly to global environmental sustainability efforts.

5.1 INTRODUCTION

Accurate prediction of carbon dioxide emissions is crucial for shaping effective policies and advancing sustainable development. This study utilizes Support Vector Machine (SVM) regression and Principal Component Regression (PCR) on a dataset spanning from 1992 to 2019 across 62 countries to analyze the impact of ten socioeconomic and environmental factors on carbon dioxide emissions. These factors include population, surface area, total consumption of fossil fuel, electricity production, GDP, urban population, construction value, manufacturing, number of livestock and agricultural gross production.

Extensive data preprocessing was undertaken to standardize the data and ensure their stationarity, preparing them for accurate analysis. Hyperparameter tuning was performed on the SVM regression model to optimize its predictive performance. The PCR method was used to address multicollinearity among predictor variables, which improved the robustness of our regression analysis.

The core of this study lies in the evaluation of the predictive accuracy of both the SVM and PCR models. By comparing predicted emissions against actual figures, we assess the models' precision and use Permutation Importance to determine the relative influence of the examined factors. This approach not only refines the predictions of the emission, but also provides policymakers with essential insights to devise effective emission reduction strategies.

In addition to SVM, PCR serves as a complementary technique that transforms correlated variables into uncorrelated principal components, reducing multicollinearity and providing stable estimates of regression coefficients. This dual approach enhances the reliability of our findings and offers a comprehensive understanding of the factors that influence carbon dioxide emissions.

By integrating advanced SVM regression techniques, PCR, and Permutation Importance analysis, this research aims to illuminate the complex dynamics of carbon emissions. The goal is to support global sustainability efforts by providing policymakers with informed and actionable insights into the determinants of carbon dioxide emissions and the effectiveness of potential mitigation strategies.

5.2 literature review

In recent years, carbon dioxide estimation and prediction have seen a large increase in the use of machine learning techniques.

Kavoosi et al. [13] forecasted carbon dioxide emissions using a genetic algorithm (GA). To predict carbon dioxide emissions in China, Sun [14] employed an optimized grey forecasting model based on Harmony Search. Abdel [15] presented an Artificial Neural Network model (ANN) to forecast time series of carbon dioxide emissions.

The adaptive neuro-fuzzy inference system (ANFIS), ANN, support vector regression (SVR), gene expression programming (GEP), particle swarm optimization (PSO) and backtracking search algorithm (BSA) were used by Kaboli et al. [16] to estimate electrical energy usage. Lu et al. [18]

applied a three-layer perceptron neural network to predict transportation-related CO2 emissions. Gholizadeh and Sabzi [33] estimated the sorption of CO2 emissions using ANFIS and ANN techniques. Norhayati and Rashid [34] used real data from a facility involved in the burning of medical waste to use the ANFIS model to assess CO2 emissions. The CO2 emissions of expanding megacities have also been predicted by Zhang et al. [35] using the XGBoost model.

When predicting the solubility of various solutes in supercritical carbon dioxide, Mehdizadeh and Movagharnejad [17] claimed that SVM is more accurate than semi-empirical models. De Paz [36] suggested that the prediction of the carbon dioxide exchange rate has been effectively resolved due to the structural risk minimization principle of SVM [37]. Wang et al. [19] indicated that SVM is an effective time series prediction method in machine learning. High-accuracy predictions using SVM were achieved by Saleh et al. [36], which can provide information on carbon dioxide emissions.

Mardani et al. [38] reviewed related works on the nexus between carbon dioxide emissions and economic growth from 1995 to 2017. Subsequently, Zheng et al. [39] used the logarithmic mean Divisia index (LMDI) to assess seven socioeconomic factors that have changed CO2 emissions. The expanded STIRPAT decomposition model, the Tapio decoupling model, and the grey relation analysis were all used by Dong et al. [40] to examine the connections between CO2 emissions, industrial structure, and economic growth.

Using a quantile regression model and path analysis of inter-provincial panel data from 2008 to 2017, Zheng et al. [41] examined the impact of renewable energy generation on CO2 emissions. According to the findings, CO2 emissions are less directly impacted by renewable energy, but are reduced by energy intensity and GDP per capita. Abbasi et al. [42] used frequency domain causality (FDC) models and unique dynamic ARDL simulations to examine environmental factors affecting China's CO2 emissions from 1980 to 2018.

Siqin et al. [43] explored the relationship between CO2 emissions, urbanization, and industrial structure using panel econometric approaches. Zhang et al. [35] found that population, land area, and GDP continue to be the main drivers of CO2 emissions when many factors work synergistically.

Furthermore, Wang et al. [44] discussed the influence factors and forecast of carbon emission in China, emphasizing the importance of structural adjustments to achieve emission peak. This study highlights the role of regional industrial structure, energy intensity, and economic development in shaping emission trajectories.

Although various machine learning models have been extensively used in carbon dioxide prediction, the application of Principal Component Regression (PCR) has also gained traction for its ability to handle multicollinearity among predictors. PCR transforms the correlated variables into uncorrelated principal components, which are then used in regression analysis. This method provides stable and reliable regression coefficients, which improves the precision of carbon dioxide emission forecasts.

5.3 FEATURES

Features play a pivotal role in predictive modeling, especially when the goal is to understand complex phenomena such as carbon volume. In this section, we discuss the significance of each feature in relation to carbon volume, detail the quantification methods, and specify the range of values. All data were sourced from the World Bank [11] and NationMaster database. [12]

1. **Total Fossil Fuel Consumption (GWh)**
   - **Importance**: Critical for assessing a nation's carbon dioxide emissions footprint.
   - **Quantification**: Combined fossil fuel energy consumption across all sectors annually.
   - **Range**: From hundreds to hundreds of thousands of GWh.
   - **Unit**: Gigawatt-hours (GWh).
2. **GDP (US $)**
   - **Importance**: Indicates economic activity level, correlating with carbon dioxide emissions.
   - **Quantification**: Annual GDP in purchasing power parity or nominal values.
   - **Range**: From billions to trillions.
   - **Unit**: US Dollars (USD).
3. **Population**
   - **Importance**: Directly influences carbon dioxide emissions through increased demand for energy and services.
   - **Quantification**: Annual population figures.
   - **Range**: From thousands to billions.
   - **Unit**: Individuals.

4. **Urban Population**
   - **Importance**: Urbanization increases energy consumption and carbon dioxide emissions.
   - **Quantification**: Number of individuals living in urban areas.
   - **Range**: From thousands to hundreds of millions.
   - **Unit**: Individuals.
5. **Electricity Production (GWh)**
   - **Importance**: Source and scale of electricity production impact carbon dioxide volumes.
   - **Quantification**: Total annual electricity production.
   - **Range**: From thousands to billions of GWh.
   - **Unit**: Gigawatt-hours (GWh).
6. **Surface Area (Square KM)**
   - **Importance**: Land use and size of a country influence carbon dioxide emissions.
   - **Quantification**: Total land area.
   - **Range**: From small areas to vast expanses.
   - **Unit**: Square kilometers (km^2).
7. **Construction Value (US $)**
   - **Importance**: Reflects construction activity levels, tied to urban development and emissions.
   - **Quantification**: Annual financial value of construction.
   - **Range**: Varied, dependent on development stage.
   - **Unit**: US Dollars (USD).
8. **Manufacturing (US $)**
   - **Importance**: Measures industrial production, a significant factor in energy use and emissions.
   - **Quantification**: Monetary value of manufactured goods annually.
   - **Range**: Reflects industrial capacity.
   - **Unit**: US Dollars (USD).
9. **Number of Livestock (Heads)**

- **Importance**: Agricultural activity level indicator, contributing to methane and carbon dioxide emissions.
- **Quantification**: Total count of livestock.
- **Range**: Varies widely.
- **Unit**: Heads.

10. **Agriculture Gross Production (million US $)**
    - **Importance**: Economic output of agriculture, influencing land use and emissions.
    - **Quantification**: Economic value of agricultural production.
    - **Range**: Based on productivity and market value.
    - **Unit**: Million US Dollars (USD).

Before implementing any procedures on our dataset, it is crucial to pre-process the raw feature data. The first step in this process is standardization, in which we rescale each feature so that it has a mean of 0 and a standard deviation of 1. This is performed irrespective of the country or year to which the data pertain. Following standardization, our initial plan was to apply differencing to achieve stationarity in the data for more reliable forecasting. However, upon conducting the Augmented Dickey-Fuller (ADF) test, we ascertained that our data are already stationary, rendering the differencing step unnecessary.

In detail, data stationarity implies a constant mean and variance over time, alongside a consistent covariance between different time intervals within the time series. This property of stationarity ensures the independence of data, a pivotal assumption for many statistical models. The Augmented Dickey-Fuller (ADF) test is a well-established statistical test for determining the stationarity of a given time series. It extends the Dickey-Fuller test by incorporating lagged differences to account for autocorrelation in the data.

The mathematical representation of the ADF test is as follows:

$$\Delta y_t = \alpha + \beta t + \gamma y_{t-1} + \delta_1 \Delta y_{t-1} + \delta_2 \Delta y_{t-2} + \ldots + \delta_p \Delta y_{t-p} + \varepsilon_t,$$

where y_t represents the time series data, α is a constant, βt is the coefficient for a time trend, γ is the coefficient for y_{t-1}, the lagged value of the time series, δ_i are the coefficients for the lagged differences, p is the number of lags included in the model, and ε_t is the error term.

The coefficient γ is the focus of the test because it is associated with the lagged value of the time series y_{t-1}. The null hypothesis H_0 posits that if $\gamma = 0$, it indicates that the time series has a unit root and is nonstationary.

The alternative hypothesis posits that if $\gamma < 0$, it suggests that the time series does not have a unit root and is stationary.

5.4 METHODOLOGY

In this section, we outline the methodology employed to predict carbon volume and investigate the importance of factors that contribute to the prediction. We utilize a support vector machine (SVM) as the primary machine learning technique and apply Permutation Importance as a feature selection method to identify the most important factors. To address multicollinearity in our data, we incorporate Principal Component Analysis (PCA) and Principal Component Regression (PCR) to reduce dimensionality and ensure stable estimates of regression coefficients.

Support Vector Regression (SVR)

Support Vector Machine (SVM) is a powerful supervised learning algorithm that aims to find the optimal hyperplane in a high-dimensional feature space. For predicting carbon dioxide volume based on multiple factors, we employ support vector regression (SVR), which handles both linear and non-linear relationships. The primary objective of SVR is to find a function $f(x)$ that approximates the target variable y as closely as possible, defined by $f(x) = \langle w, x \rangle + b$.

SVR uses a ϵ insensitive loss function, $L(y, f(x)) = \max(0, |y - f(x)| - \epsilon)$, to minimize the impact of errors within a certain margin. The optimization problem involves minimizing the following.

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

subject to the constraints outlined in Chapter 2, Section 2.1.

The SVR can also be extended to solve non-linear problems using kernel functions, with common choices being Linear, Polynomial, and Radial Basis Function (RBF) kernels. For more details, please refer to Chapter 2, Section 2.1 of the Preliminaries.

Permutation Importance

To determine the relative importance of factors, we employ Permutation Importance as a feature selection technique. Permutation Importance measures the impact of permuting the values of each feature on the model's performance. By ranking the features through this process, we gain insights into the factors that have the most significant impact on the volume of carbon dioxide. This combined approach of SVR and Permutation Importance allows us to make accurate predictions while identifying the key drivers behind carbon emissions.

The process of calculating permutation importance and its interpretation is shown in Theorem 2.1 in the Preliminaries.

Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms a large set of correlated variables into a smaller set of uncorrelated variables known as principal components. The primary goal of PCA is to capture as much variance as possible with the fewest number of principal components. This process involves the following steps:

1. **Standardization**: The data is standardized to have a mean of zero and a standard deviation of one. 2. **Covariance Matrix Computation**: The covariance matrix of the standardized data is computed. 3. **Eigen Decomposition**: The eigenvalues and eigenvectors of the covariance matrix are calculated. The eigenvalues represent the variance captured by each principal component, while the eigenvectors represent the directions of the principal components. 4. **Principal Components Selection**: The principal components are selected based on their eigenvalues, typically retaining enough components to explain a desired percentage of the total variance (e.g., 90%).

Mathematically, if $\mathbf{X}$ is the standardized data matrix, then the covariance matrix $\mathbf{C}$ is given by:

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X},$$

The eigenvalues λ_i and eigenvectors $\mathbf{v}_i$ are obtained by solving:

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i.$$

The principal components are then given by:

$$\mathbf{Z} = \mathbf{X}\mathbf{V},$$

where $\mathbf{V}$ is the matrix of selected eigenvectors.

PCA transforms correlated variables into uncorrelated principal components, effectively reducing multicollinearity in the data. This transformation is particularly useful in regression analysis, where multicollinearity can lead to unstable estimates of regression coefficients. For more details on the methodology, see Section 2.4 in the preliminaries.

Principal Component Regression (PCR)

Principal Component Regression (PCR) combines PCA and multiple linear regression. The steps involved in PCR are:

1. **PCA on Predictor Variables**: Perform PCA on the predictor variables to obtain the principal components. 2. **Selection of Principal Components**: Select a subset of principal components that explain a sufficient amount of variance. 3. **Regression Analysis**: Use the principal components selected as predictors in a multiple linear regression model to predict the response variable.

The mathematical formulation of PCR is as follows:

1. Let $\mathbf{X}$ be the matrix of predictor variables and $\mathbf{Y}$ be the response variable. 2. Perform PCA on $\mathbf{X}$ to obtain the principal components $\mathbf{Z}$. 3. Select the first k principal components $\mathbf{Z}_k$ that explain a significant portion of the variance. 4. Fit a linear regression model:

$$\mathbf{Y} = \mathbf{Z}_k\mathbf{B} + \mathbf{E},$$

where $\mathbf{B}$ is the vector of regression coefficients and $\mathbf{E}$ is the error term.

PCR effectively addresses multicollinearity by using principal components, which are orthogonal to each other, thereby providing stable estimates of the regression coefficients.

5.5 IMPLEMENTATION

In this section, we outline the implementation details of Principal Component Regression (PCR) and compare its performance with the Support Vector Machine (SVM) regression model.

Principal Component Regression (PCR)

To implement PCR, we first perform PCA on the predictor variables to reduce their dimensionality. The principal components are then used as predictors in a linear regression model.

Model's Fine-Tuning

Hyperparameter tuning is a critical step in the machine learning pipeline, ensuring that the model is optimized for the given data, thereby improving generalization on unseen datasets. To fine-tune our Support Vector Regressor (SVR) model, we embarked on a systematic exploration of the hyperparameter space using Grid Search coupled with 5-fold cross-validation.

Hyperparameter Space

The following hyperparameters were considered:

- **Kernel**: Determines the type of hyperplane used to separate the data. We experimented with `linear`, `poly`, and `rbf`.
- **C (Regularization Parameter)**: This parameter trades off correct classification of training examples against maximization of the decision function's margin. We tested values in the range [0.1,1,10,100,1000,2000,3000,4000,5000].
- **Gamma**: Defines how far the influence of a single training example reaches. We tried both `scale` and `auto`.
- **Degree**: The degree of the polynomial kernel function (`poly`). Evaluated degrees included 2, 3, and 4, although this parameter is disregarded if the kernel isn't polynomial.

Results

The Grid Search, combined with 5-fold cross-validation, revealed the optimal hyperparameters for our dataset as:

- Kernel: `poly`
- C: 2000
- Gamma: `auto`
- Degree: 2

This configuration was determined to offer the most promising performance, balancing the bias-variance trade-off and potentially delivering superior results on unseen data.

Principal Component Regression (PCR) Performance

After performing PCA on the predictor variables and selecting the first three principal components, we used these components as predictors in a linear regression model. The results of the PCR model are as follows:

- **Mean Squared Error**: 0.08226991002545095
- **R-squared**: 0.9431715925806526
- **Regression Coefficients**: [[0.36246693 -0.00369905 0.05123607]]

The explained variance ratios for the selected principal components are:

- **PC1**: 0.68652744
- **PC2**: 0.15766246
- **PC3**: 0.08149733

The principal component loadings for the first three principal components are:

Feature	PC1	PC2	PC3
Population	0.288659	-0.469154	0.090625
Surface Area	0.230376	-0.082302	-0.825116
GDP	0.321209	0.391463	0.141022
Total Fossil Fuel Consumption	0.363920	0.075377	-0.016215
Urban Population	0.344907	-0.313251	0.084428
Electricity Production	0.259688	0.438979	-0.340387
Agriculture Gross Production	0.325330	-0.231430	0.242083
Manufacturing	0.345919	0.227373	0.260098
Construction Value	0.339340	0.306241	0.161256
Number of Livestock	0.317710	-0.352596	-0.124784

Table 5.1: Principal Component Loadings

Principal component regression (PCR) effectively addresses the issue of multicollinearity by transforming the original predictor variables into orthogonal principal components, thereby providing stable and reliable regression coefficients.

5.6 performance metrics of regression models

In this section, we present the performance metrics of our regression models, trained and tested with data from numerous countries to explore the dynamics between carbon dioxide emissions and a variety of socioeconomic indicators. We have scrutinized two critical metrics for assessing model performance: R-squared and mean squared error (MSE) for an 80/20 training/testing split, along with cross-validated scores to ensure robustness.

Support Vector Regression (SVR) Model

Table 5.2: Performance Metrics for SVR Model (80/20 Training and Testing Ratio)

Metric/Training-Testing Split	**80%/20%**
R-squared Score	0.9895
Mean Squared Error	0.0152

Observations from the metrics reveal exceptionally high R-squared values for the 80/20 split, averaging at 0.9895. Such strong R-squared values demonstrate the excellent fit of the model to the data set, indicating that the socioeconomic factors considered possess significant predictive power for carbon dioxide emissions. The uniformity of these high R-squared scores in various splits further attests to the stability and reliability of the model.

MSE scores, indicative of the average squared differences between the observed and predicted values by the model, are comparatively low, reaffirming the precision of the model in forecasting carbon dioxide emissions and highlighting its accuracy.

The SVR model demonstrates formidable predictive strength, as evidenced by high R-squared and low MSE scores, underscoring its utility in capturing the complex interplay between carbon dioxide emissions and socioeconomic factors. The performance metrics suggest areas for further methodological refinement, particularly in exploring different data partitioning strategies to enhance prediction accuracy and mitigate overfitting risks.

To illustrate the prediction ability of the SVR model more clearly, Figure 1 shows the prediction results versus the actual value of the 80/20 training and testing split.

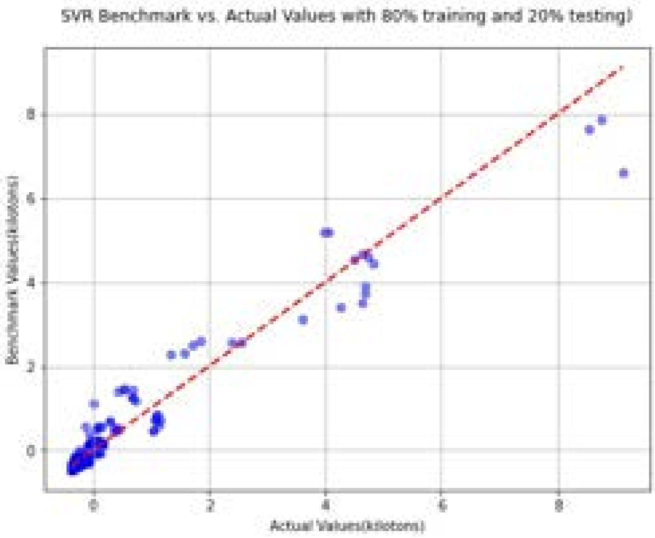


Figure 5.1: SVR Model prediction vs. actual value with 80% training and 20% testing

From the graph, it is evident that, despite some outliers, most points are closely aligned with the diagonal line, which represents the accuracy of the prediction of 100%. Thus, the prediction power of the SVR model is reliable.

Principal Component Regression (PCR) Model

To validate the robustness of our PCR model, we employed a k-fold cross-validation. The cross-validated R-squared and MSE scores for the PCR model are presented below:

Table 5.3: Cross-Validation Performance Metrics for PCR Model

Metric	**Score**
Cross-Validated R-squared Scores Mean R-squared	[0.9428, 0.8421, 0.9174, 0.9215, 0.8827] 0.9013
Cross-Validated MSE Scores Mean MSE	[-0.0828, -0.0858, -0.0981, -0.0927, -0.0711] 0.0861

The cross-validation results indicate a strong average R-squared score of 0.9013, confirming the reliability of the PCR model across different folds of the dataset. The consistent MSE values further underscore the predictive accuracy and robustness of the model.

To illustrate the prediction ability of the PCR model more clearly, Figure 2 shows the prediction results versus the actual value of the 80/20 training and testing split.

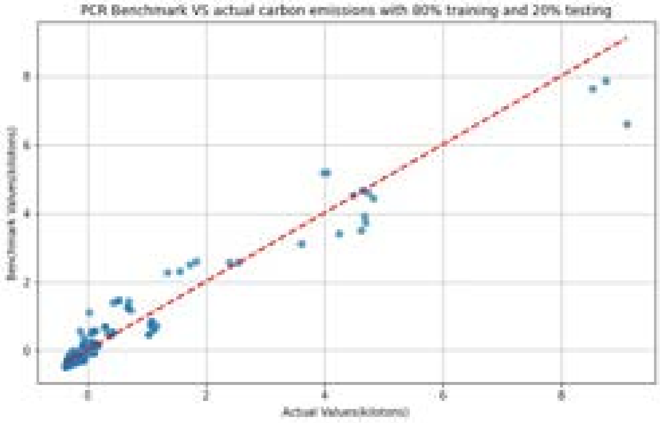


Figure 5.2: PCR Model prediction vs. actual value with 80% training and 20% testing

From the graph, it is evident that, despite some outliers, most points are closely aligned with the diagonal line, which represents the accuracy of the prediction of 100%. Thus, the prediction power of the PCR model is reliable.

Next, we employ the permutation importance technique to assess the importance of different factors in our SVR model. As mentioned previously, permutation importance is a model-agnostic feature importance technique that evaluates the impact of individual features on the model's predictions. It does so by comparing the performance of the model in the original dataset with its performance in terms of the R-squared score after shuffling each feature.

Table 5.4: Feature Importance Ranking for 80%/20% Training/Testing Split

Rank	Feature	Importance Value (%)
1	Total Fossil Fuel Consumption (GWh)	36.8243
2	GDP (US $)	13.0691
3	Population	8.3565
4	Urban Population	6.8323
5	Electricity Production (GWh)	2.1137
6	Surface Area (Square KM)	1.7779
7	Construction Value (US $)	1.3473
8	Manufacturing (US $)	1.2966
9	Number of Livestock (Heads)	0.4121
10	Agriculture Gross Production (million US $)	0.2432

From the analysis of feature importances for the 80/20 training/testing split in the SVR model, we observe the dominant influence of 'Total Fossil Fuel Consumption (GWh),', which highlights its pivotal role in predicting carbon dioxide emissions. This consistency underscores the direct impact of a nation's total fossil fuel usage on its carbon dioxide footprint.

The terms 'population' and 'GDP (US$)' also consistently rank highly, reinforcing the significant roles these factors play in the estimation of carbon dioxide emissions. The steadfast positions of these features illustrate the undeniable correlation between population size, economic output, and carbon dioxide emissions. These elements serve as fundamental drivers in the models, emphasizing the interplay between demographic scale, economic activity, and environmental impact.

Although the 'urban population' consistently emerges as a crucial factor, its ranking varies slightly, indicating its significant but fluctuating impact on the predictions of carbon dioxide emissions. This fluctuation suggests that while urbanization is a key determinant of carbon dioxide emissions, its relative influence can be modulated by other socioeconomic factors.

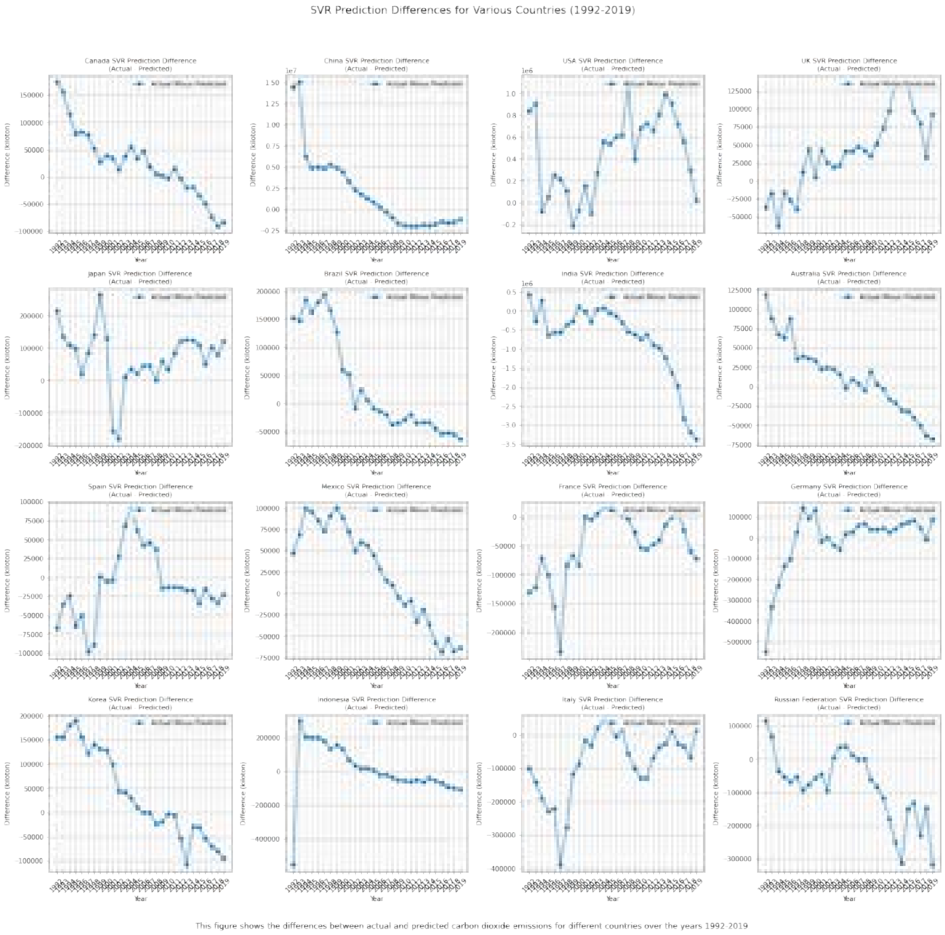
'Electricity production (GWh)' and 'Surface area (KM square)' exhibit variable importance, reflecting the nuanced relationship these factors have with carbon dioxide emissions. Electricity production, in particular, showcases how energy generation methods and efficiency levels can significantly influence a country's carbon dioxide footprint.

Interestingly, "Construction Value (US $)" and "Manufacturing (US $)" show a noteworthy presence, pointing to the considerable effect of the industrial sector on carbon dioxide emissions. These factors highlight the environmental cost of industrial and construction activities, underscoring the need for sustainable practices in these areas.

Less prominently ranked features such as “Number of Livestock (Heads)” and “Agriculture Gross Production (million US $)” still contribute valuable information, suggesting the role of the agricultural sector in carbon dioxide emissions. Although these factors are lower, they underscore the broader spectrum of contributors to a nation’s carbon dioxide emissions, from agriculture to industrial production.

The observed variations in feature importance shed light on the complex interdependencies among socioeconomic, demographic, and environmental factors in the carbon dioxide emission dynamics. These insights call for a deeper exploration into how these variables interact to shape global carbon dioxide emission profiles, providing valuable guidance for targeted policy and intervention strategies to mitigate environmental impact.

Comparing Countries' Performance



This figure shows the differences between actual and predicted carbon dioxide emissions for different countries over the years 1992-2019

Figure 5.3: Difference of carbon dioxide emission by SVR between actual value and model prediction results

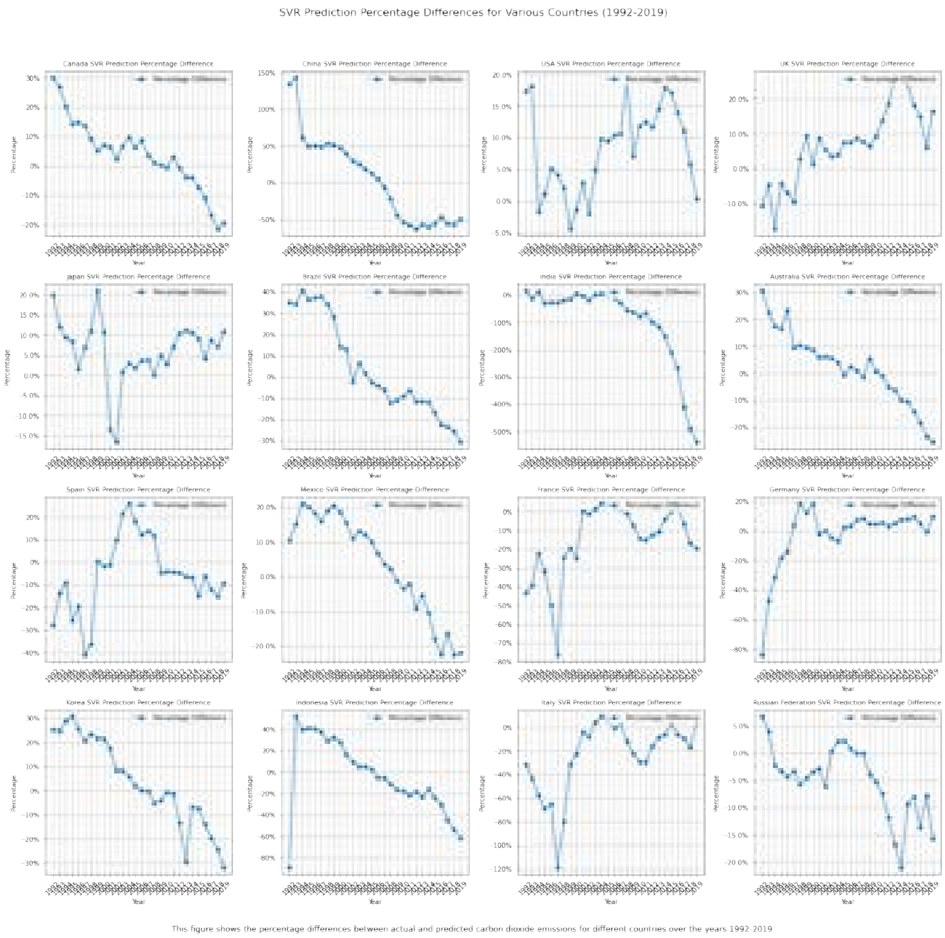
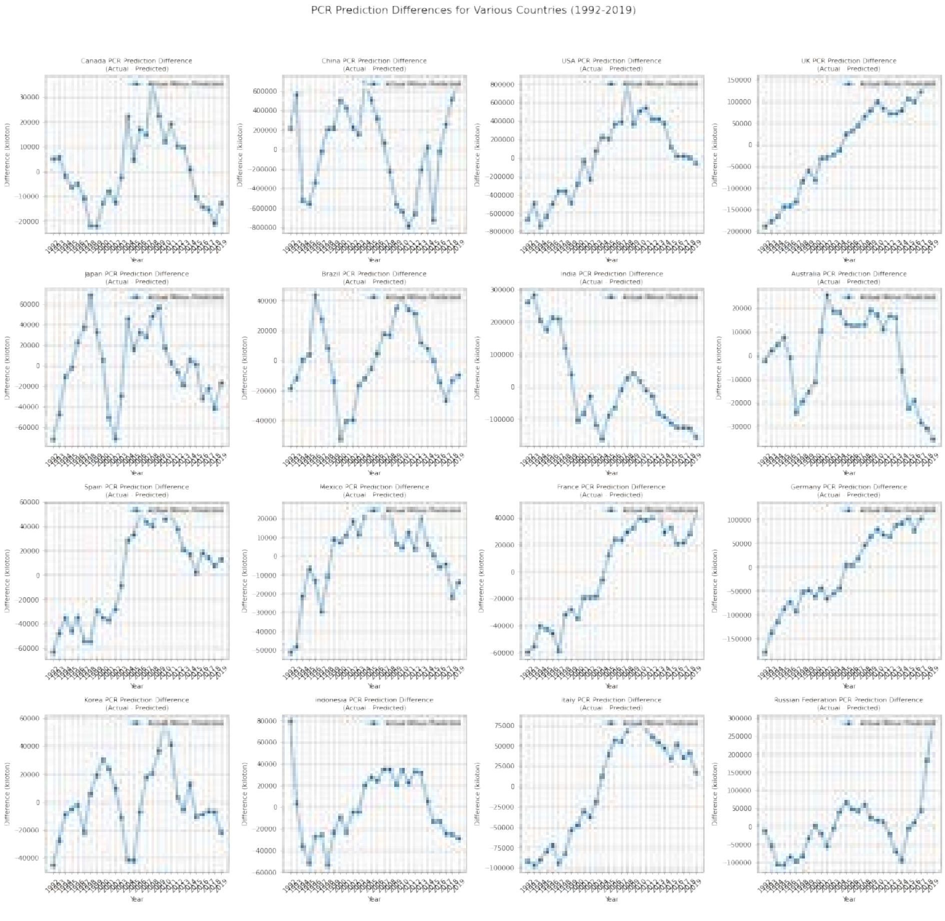


Figure 5.4: Percentage difference of carbon dioxide emission by SVR between actual value and model prediction results



This figure shows the differences between actual and predicted carbon dioxide emissions for different countries over the years 1992-2019

Figure 5.5: Difference of carbon dioxide emission by PCR between actual value and model prediction results

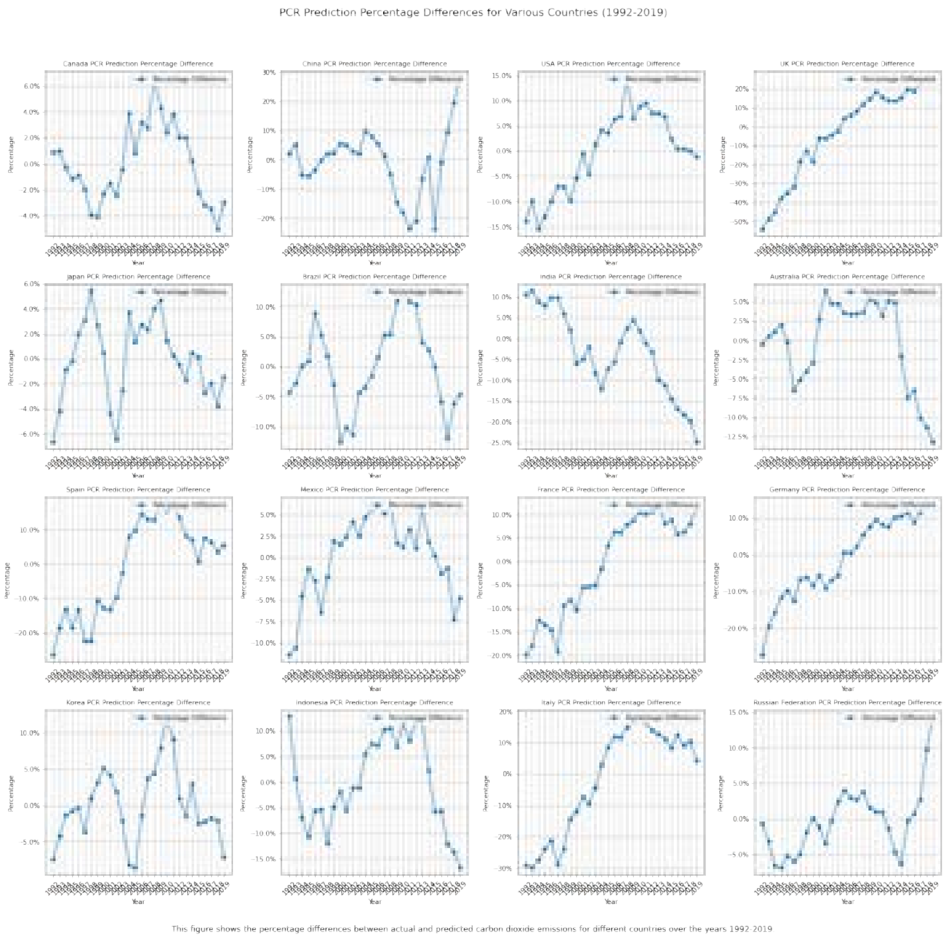


Figure 5.6: Difference of carbon dioxide emission by PCR between actual value and model prediction results

In our quest to create a predictive model with wide applicability, we dedicated 80% of a comprehensive dataset from 1992 to 2019, covering 62 countries, to training. This decision was instrumental in enhancing the model's ability to predict carbon dioxide emissions globally. Our focus on 16 major economies yielded two series of graphs that not only displayed the raw differences between predicted and actual carbon dioxide emissions but also contextualized these differences relative to actual emissions.

Training the model with a comprehensive global dataset allows us to project the world's carbon emission expectations onto individual countries. This approach enables us to assess whether a country's performance aligns with global standards, providing a benchmark for evaluating national efforts in emission reduction.

Principal Component Regression (PCR) Analysis

Graphical analyses, encapsulated in Figures 5.5 and 5.6, reveal a spectrum of accuracy across countries, reflected in underestimations and overestimations by the model. This granularity uncovers trends and deviations, highlighting the distinct environmental trajectories of each nation.

In the UK, the model's predictions shift from negative to positive differences, indicating a trend from overestimation to underestimation in recent years. This suggests evolving emission factors not fully captured by the model.

Canada exhibits a distinctive pattern of prediction differences. The early years show an overestimation, followed by a notable downward spike, a subsequent upward spike, and another decline. This fluctuation indicates significant variability in emission factors and the impact of environmental policies. The early overestimations may suggest an initial overestimation of emission levels, while the downward and upward spikes reflect periods of effective policy implementation and possible lapses or changes in industrial activity.

In the USA, the predictions show a general increasing trend with significant positive spikes, indicating a consistent underestimation. Notable years like 2009 and 2016 highlight possible optimistic evaluations of environmental efforts or technological advancements.

China presents a mixed picture. The early years show large positive differences, followed by a decline and recent negative differences. This reflects the complexities of forecasting in an evolving industrial landscape with rapid industrialization periods and successful emissions reduction initiatives.

Support Vector Regression (SVR) Analysis

Similar to the PCR approach, we employed SVR to predict carbon dioxide emissions using the same comprehensive dataset. This method provided another perspective on the prediction accuracy in different countries. By analyzing the differences and percentage differences between the actual and predicted emissions, we can evaluate the performance of the model and the specific nuances it captures for each country.

The graphical analyses in Figures 5.3 and 5.4 highlight the prediction discrepancies for the 16 major economies. For example, the SVR model tends to consistently underestimate emissions in the USA, indicating potential gaps in capturing the complexities of its emissions profile. For China and Canada, the model starts with an underestimation, but the trend shows a decline in the differences, suggesting that these countries' environmental policies are progressively improving their emission profiles.

In Australia, the differences show a clear downward trend from positive to negative, indicating that the SVR model increasingly overestimates emissions over time. Conversely, Germany exhibits an upward trend in differences, suggesting that the model increasingly underestimates emissions, indicating potential issues in capturing recent increases in emissions or changes in industrial activity.

The performance of the SVR model varies significantly across different countries, reflecting the diverse environmental and industrial landscapes. These insights underscore the importance of using multiple models to accurately capture the complexities of global carbon dioxide emissions.

Conclusion

The results of the PCR and SVR analyses provide valuable insights into the accuracy and nuances of the carbon dioxide emission predictions for major economies. Both models show a mix of underestimations and overestimations, highlighting the complexities of forecasting emissions on a global scale.

Both PCR and SVR analyses reveal consistent underestimations in the USA, indicating that the country is producing more carbon dioxide than the models predict. In China, both models show early underestimations followed by overestimations, reflecting the country's rapid industrialization and subsequent emission reduction initiatives.

The PCR model shows a shift from overestimation to underestimation in the UK, whereas the SVR model does not capture this trend as clearly. Canada's distinctive pattern of early overestimations and later fluctuations is more pronounced in the PCR analysis compared to the SVR analysis.

The differences between PCR and SVR predictions may stem from the inherent nature of each model. PCR, which reduces dimensionality, might miss some nuanced variations captured by SVR's flexibility in handling non-linear relationships. The varying economic and environmental policies across countries also contribute to the differences observed in the model predictions.

In general, using both PCR and SVR provides a more comprehensive understanding of carbon dioxide emission trends, helping to identify areas where models need adjustment and where policy impacts are most significant. Using world data, these models allow for benchmarking a country's carbon performance against global standards, thus offering a clear perspective on how each country measures up on a global scale. This is the most significant aspect of the research, as it highlights the relative success or shortcomings of national policies in the context of worldwide emission trends.

5.7 DISCUSSION

This research aimed to develop a comprehensive model for predicting carbon dioxide emissions by leveraging advanced machine learning techniques and a broad range of socioeconomic and environmental variables. By integrating Support Vector Regression (SVR), Principal Component Regression (PCR), and Permutation Importance, we sought to capture the intricate relationships between these factors and carbon dioxide emissions.

The SVR model demonstrated robust predictive capabilities, achieving an exceptionally high R-squared value of 0.9895 and a low Mean Squared Error (MSE) of 0.0152. These metrics indicate the model's excellent fit to the dataset, suggesting that the selected socioeconomic factors possess significant predictive power for carbon dioxide emissions. The consistent performance across different data splits further attests to the stability and reliability of the model.

In parallel, the PCR model effectively addressed the issue of multicollinearity by transforming correlated variables into orthogonal principal components. This approach yielded stable and reliable regression coefficients, with a strong mean R-squared of 0.9013 across cross-validation folds. The performance of the PCR model reinforces the findings from the SVR model, providing complementary insights into the dynamics of carbon dioxide emissions.

Hyperparameter tuning using Grid Search and 5-fold cross-validation played a critical role in optimizing the SVR model. The optimal configuration, a polynomial kernel with specific parameters, struck a balance between bias and variance, ensuring the generalization of the model in unseen data. This rigorous approach underscores the importance of systematic hyperparameter optimization in machine learning workflows.

The permutation importance analysis highlighted the most significant predictors of carbon dioxide emissions, with "Total Fossil Fuel Consumption (GWh)," GDP, and population size emerging as key factors. This model-agnostic technique validated our choice of predictors and provided a detailed map of leverage points for targeted and effective environmental strategies.

Training on a global dataset allowed us to benchmark countries' carbon emission performances against world standards. Graphical analyses revealed varied trends: the UK consistently underestimated emissions, possibly due to evolving emission factors, while Canada showed overestimations, suggesting effective environmental policies. The consistent underestimations of the USA indicate higher emissions than predicted, while China's mixed results reflect its rapid industrialization and emission reduction efforts.

Overall, using both PCR and SVR provides a comprehensive understanding of carbon dioxide emission trends, helping to identify areas for model adjustment and significant policy impacts. These models offer valuable insights for benchmarking national carbon performance against global standards, highlighting the importance of continuous data enhancement for improved accuracy and reliability. This multifaceted approach is crucial for the development of effective and informed environmental strategies that contribute to global sustainability efforts.

5.8 CONCLUSION

This study explores the global dynamics of carbon dioxide emissions using a dataset from 62 countries, highlighting the impact of socioeconomic and environmental variables such as Electricity Production and Urban Population. The comprehensive dataset and advanced machine learning techniques demonstrate the potential for adaptive and evolving environmental studies.

The Support Vector Regression (SVR) model achieved high predictive accuracy with an R-squared value of 0.9895 and a low Mean Squared Error (MSE) of 0.0152, affirming the robustness of our methodology through rigorous data preprocessing and hyperparameter tuning. The Principal Component Regression (PCR) model addressed multicollinearity and provided stable and reliable regression coefficients, complementing the SVR findings.

Country-specific analyses revealed unique emission trajectories. The consistent underestimations of the USA indicate higher-than-expected emissions, suggesting potential gaps in capturing the complexities of its emission profile. In contrast, China's early underestimations followed by overestimations reflect the country's rapid industrialization and subsequent emission reduction initiatives. Canada's emission patterns showed early overestimations followed by a decline, indicating the effectiveness of its environmental policies over time.

The permutation importance analysis highlighted key predictors of carbon dioxide emissions, such as Total Fossil Fuel Consumption, GDP, and population size, validating our choice of predictors and providing a detailed map of leverage points for targeted environmental strategies.

In general, this research underscores the necessity of a multifaceted approach that combines SVR and PCR, offering a robust analytical framework to understand the intricate relationships between socioeconomic factors and carbon dioxide emissions. This comprehensive perspective is crucial for developing more informed and effective strategies for global environmental sustainability.

BIBLIOGRAPHY

[1] Junwei Su, Shan Wu, and Jinhui Li. MTRGL: Effective Temporal Correlation Discerning through Multi-modal Temporal Relational Graph Learning. In *2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, Seoul, Korea, 2024. Institute of Electrical and Electronics Engineers.

[2] J. D. Hamilton. A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica*, 57(2):357–384, 1989.

[3] Andrew Ang and Geert Bekaert. International asset allocation with regime shifts. *Review of Financial Studies*, 15(4):1137–1187, 2002.

[4] H. Markowitz. Portfolio selection. *Journal of Finance*, 7(1):77, 1952.

[5] S. Maillard, T. Roncalli, and J. Teiletche. The properties of equally weighted risk contribution portfolios. *Journal of Portfolio Management*, 36(4):60, 2010.

[6] V. DeMiguel, L. Garlappi, and R. Uppal. Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *Review of Financial Studies*, 22(5):1915–1953, 2009.

[7] Yves Choueifaty and Yves Coignard. Toward maximum diversification. *The Journal of Portfolio Management*, 34(4):40–51, 2008.

[8] Evan Gatev, William N. Goetzmann, and K. Geert Rouwenhorst. Pairs trading: Performance of a relative value arbitrage rule. *Yale ICF Working Paper No. 08-03*, February 2006.

[9] Daniel Herlemont. Pairs trading, convergence trading, cointegration. 01 2003.

[10] Weiguang Han, Jimin Huang, Qianqian Xie, Boyi Zhang, Yanzhao Lai, and Min Peng. Mastering pair trading with risk-aware recurrent reinforcement learning. *arXiv preprint arXiv:2304.00364*, 2023. Available at arXiv: `https://arxiv.org/abs/2304.00364`.

[11] World Bank. Carbon volume and its related factors. World Bank, 1992–2019.

[12] NationMaster. Global statistics and country comparisons. NationMaster, Accessed 2023.

[13] H Kavoosi, MH Saidi, M Kavoosi, and M Bohrng. Forecast global carbon dioxide emission by use of genetic algorithm (ga). *International Journal of Computer Science Issues (IJCSI)*, 9(5):418, 2012.

[14] Wei Sun, Jingmin Wang, and Hong Chang. Forecasting carbon dioxide emissions in china using optimization grey model. *J. Comput.*, 8(1):97–101, 2013.

[15] Abdel Karim Baareh. Solving the carbon dioxide emission estimation problem: An artificial neural network model. 2013.

[16] S Hr Aghay Kaboli, A Fallahpour, J Selvaraj, and NA Rahim. Long-term electrical energy consumption formulating and forecasting via optimized gene expression programming. *Energy*, 126:144–164, 2017.

[17] Bahman Mehdizadeh and Kamyar Movagharnejad. A comparison between neural network method and semi empirical equations to predict the solubility of different compounds in supercritical carbon dioxide. *Fluid Phase Equilibria*, 303(1):40–44, 2011.

[18] Xiangyong Lu, Kaoru Ota, Mianxiong Dong, Chen Yu, and Hai Jin. Predicting transportation carbon emission with urban big data. *IEEE Transactions on Sustainable Computing*, 2(4):333–344, 2017.

[19] Wenjian Wang, Changqian Men, and Weizhen Lu. Online prediction model based on support vector machine. *Neurocomputing*, 71(4-6):550–558, 2008.

[20] M. Kritzman, S. Page, and D. Turkington. In defense of optimization: The fallacy of 1/n. *Financial Analysts Journal*, 66(2), 2010.

[21] Marco Escobar, Michael Mitterreiter, David Saunders, Luis Seco, and Rudi Zagst. Market crises and the 1/n asset-allocation strategy. *Journal of Investment Strategies*, 2(4):83–107, 2013.

[22] David A. Levin, Yuval Peres, and Elizabeth L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society. American Mathematical Society, Providence, RI, 2009.

[23] Evan Gatev, William N Goetzmann, and K Geert Rouwenhorst. Pairs trading: Performance of a relative-value arbitrage rule. *The Review of Financial Studies*, 19(3):797–827, 2006.

[24] Farimah Poursafaei, Shenyang Huang, Kellin Pelrine, , and Reihaneh Rabbany. Towards better evaluation for dynamic link prediction. In *Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2022.

[25] Yao Zhang, Yun Xiong, Yongxiang Liao, Yiheng Sun, Yucheng Jin, Xuehao Zheng, and Yangyong Zhu. Tiger: Temporal interaction graph embedding with restarts, 2023.

[26] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. Temporal graph networks for deep learning on dynamic graphs. *arXiv preprint arXiv:2006.10637*, 2020.

[27] Da Xu, Chuanwei Ruan, Evren Korpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. *arXiv preprint arXiv:2002.07962*, 2020.

[28] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[29] Guillaume Garrigos and Robert M. Gower. *Handbook of Convergence Theorems for (Stochastic) Gradient Methods*. Université Paris Cité and Sorbonne Université, CNRS Laboratoire de Probabilités, Statistique et Modélisation, F-75013 Paris, France, January 2023. Center for Computational Mathematics, Flatiron Institute, Simons Foundation, New York.

[30] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik. Sgd: General analysis and improved rates. In *International Conference on Machine Learning*, pages 5200–5209, 2019.

[31] Simão Moraes Sarmento and Nuno Horta. Enhancing a pairs trading strategy with the application of machine learning. *Expert Systems with Applications*, 158:113490, 2020.

[32] Bendong Zhao, Huanzhang Lu, Shangfeng Chen, Junliang Liu, and Dongya Wu. Convolutional neural networks for time series classification. *Journal of Systems Engineering and Electronics*, 28(1):162–169, 2017.

[33] Farshad Gholizadeh and Fatemeh Sabzi. Prediction of co2 sorption in poly (ionic liquid) s using ann-gc and anfis-gc models. *International Journal of Greenhouse Gas Control*, 63:95–106, 2017.

[34] I Norhayati and M Rashid. Adaptive neuro-fuzzy prediction of carbon monoxide emission from a clinical waste incineration plant. *Neural Computing and Applications*, 30:3049–3061, 2018.

[35] Jianxun Zhang, He Zhang, Rui Wang, Mengxiao Zhang, Yazhe Huang, Jiahui Hu, and Jingyi Peng. Measuring the critical influence factors for predicting carbon dioxide emissions of expanding megacities by xgboost. *Atmosphere*, 13(4):599, 2022.

[36] Chairul Saleh, Nur Rachman Dzakiyullah, and Jonathan Bayu Nugroho. Carbon dioxide emission prediction using support vector machine. In *IOP Conference Series: Materials Science and Engineering*, volume 114, page 012148. IOP Publishing, 2016.

[37] Milan Protić, Shahaboddin Shamshirband, Dalibor Petković, Almas Abbasi, Miss Laiha Mat Kiah, Jawed Akhtar Unar, Ljiljana Živković, and Miomir Raos. Forecasting of consumers heat load in district heating systems using the support vector machine with a discrete wavelet transform algorithm. *Energy*, 87:343–351, 2015.

[38] Abbas Mardani, Dalia Streimikiene, Fausto Cavallaro, Nanthakumar Loganathan, and Masoumeh Khoshnoudi. Carbon dioxide (co2) emissions and economic growth: A systematic review of two decades of research from 1995 to 2017. *Science of the total environment*, 649:31–49, 2019.

[39] Jiali Zheng, Zhifu Mi, D'Maris Coffman, Stanimira Milcheva, Yuli Shan, Dabo Guan, and Shouyang Wang. Regional development and carbon emissions in china. *Energy Economics*, 81:25–36, 2019.

[40] Biying Dong, Xiaojun Ma, Zhuolin Zhang, Hongbo Zhang, Ruimin Chen, Yanqi Song, Meichen Shen, and Ruibing Xiang. Carbon emissions, the industrial structure and economic growth: Evidence from heterogeneous industries in china. *Environmental Pollution*, 262:114322, 2020.

[41] Huanyu Zheng, Malin Song, and Zhiyang Shen. The evolution of renewable energy and its impact on carbon reduction in china. *Energy*, 237:121639, 2021.

[42] Kashif Raza Abbasi, Muhammad Shahbaz, Jinjun Zhang, Muhammad Irfan, and Rafael Alvarado. Analyze the environmental sustainability factors of china: The role of fossil fuel energy and renewable energy. *Renewable Energy*, 187:390–402, 2022.

[43] Zhuoya Siqin, Dongxiao Niu, Mingyu Li, Hao Zhen, and Xiaolong Yang. Carbon dioxide emissions, urbanization level, and industrial structure: Empirical evidence from north china. *Environmental Science and Pollution Research*, 29(23):34528–34545, 2022.

[44] Qing Wang and Hong Yang. Influence factors and forecast of carbon emission in china: structure adjustment for emission peak. *IOP Conference Series: Earth and Environmental Science*, 113(1):012197, 2018.

COLOPHON

This thesis was typeset using the typographical look-and-feel `classicthesis` developed by André Miede and Ivo Pletikosić.

The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*".

Here you can insert things like "Figures were created with..."

[Insert version number/description, if you want]