**Creating an Index Fund Using AI-Driven Decision-Making Approaches**

**Introduction**

Index funds have revolutionized the investment landscape by providing investors with a low-cost, passive investment vehicle that aims to replicate the performance of a specific market index. Traditionally, these funds attempt to track their benchmark indices by holding all constituent securities in proportions equal to their weights in the index. However, this approach can be inefficient and costly, especially for indices with numerous constituents like the S&P 100, which comprises 100 leading U.S. companies.

This project explores and implements two AI-driven decision-making approaches to construct an optimized index fund that tracks the S&P 100 index using only a subset of the constituent stocks. The central challenge is to select a smaller number of stocks (q) and determine the optimal allocation of weights to achieve performance as similar as possible to the benchmark across different time horizons (1-4 quarters).

The relevance of this work lies in its potential to reduce transaction costs, management complexity, and improve scalability while maintaining performance comparable to the full index. For institutional investors, fund managers, and individual investors alike, an effectively optimized index fund can provide significant advantages in terms of lower expense ratios and reduced tracking complexity.

The theoretical foundations of this work build upon several interconnected domains in finance and artificial intelligence:

1. **Modern Portfolio Theory (MPT)**: Introduced by Harry Markowitz in 1952, MPT provides the mathematical framework for constructing portfolios that maximize expected returns for a given level of risk. Our optimization approach extends this theory by incorporating tracking metrics as key objectives.
2. **Index Tracking**: A specialized application of portfolio optimization that aims to replicate the performance of a market index. Traditional approaches include full replication (holding all constituents) and sampling (holding a representative subset). Our work focuses on advanced sampling techniques driven by AI.
3. **Machine Learning for Financial Analysis**: Recent advances in machine learning have enabled more sophisticated approaches to financial data analysis. Our clustering approach leverages unsupervised learning to identify patterns and relationships among stocks that might not be captured by traditional financial metrics alone.

The primary contributions of this project include:

1. Formulating the index tracking problem as a multi-objective optimization problem that balances correlation maximization and risk minimization, solvable using AMPL (A Mathematical Programming Language).
2. Developing a novel clustering-based approach that groups stocks based on their financial characteristics and selects representative securities from each cluster.
3. Conducting a comprehensive comparative analysis of both approaches across different values of q (number of stocks) and time horizons to identify the most effective strategy.
4. Creating an extensible, modular code framework that can be adapted to other indices and financial markets, potentially informing future research in this domain.

In the sections that follow, we present a detailed problem definition, methodological approaches, implementation details, empirical results, and a critical evaluation of our findings. The report concludes with insights gained and avenues for future exploration in this promising intersection of finance and artificial intelligence.

**Problem Definition**

**Mathematical Formulation of the Index Tracking Problem**

The core challenge in this project is to construct an index fund that tracks the S&P 100 using fewer than 100 stocks while maintaining performance as similar as possible to the benchmark. This can be formalized as an optimization problem with specific variables, constraints, and objectives.

**Decision Variables**

1. **Binary Selection Variables**: For each stock i in the universe of S&P 100 constituents, we define a binary variable x\_i ∈ {0,1} where:
   * x\_i = 1 if stock i is selected for inclusion in our index fund
   * x\_i = 0 if stock i is not selected
2. **Weight Allocation Variables**: For each selected stock, we define a continuous variable w\_i ∈ [0,1] representing the proportion of the fund invested in stock i.

**Constraints**

1. **Cardinality Constraint**: The number of stocks selected must equal q (a parameter we vary in our experiments):
2. ∑ x\_i = q
3. **Budget Constraint**: The weights of selected stocks must sum to 1:
4. ∑ w\_i = 1
5. **Investment Constraint**: We can only invest in selected stocks:
6. w\_i ≤ x\_i \* max\_weight for all i

Where max\_weight is the maximum proportion allowed in any single stock.

1. **Weight Bounds**: To ensure diversification and prevent extreme concentration:
2. w\_i ≥ min\_weight if x\_i = 1
3. w\_i ≤ max\_weight if x\_i = 1
4. w\_i = 0 if x\_i = 0

**Objective Function**

The primary objective is to maximize the similarity between our fund and the benchmark. This can be measured using correlation:

maximize ρ(R\_p, R\_b)

Where:

* R\_p is the return series of our portfolio
* R\_b is the return series of the benchmark (S&P 100)
* ρ is the correlation coefficient

We also consider minimizing tracking error, which measures the standard deviation of the difference between portfolio and benchmark returns:

minimize TE = √[var(R\_p - R\_b)]

These objectives can be combined into a multi-objective function with weights reflecting their relative importance:

maximize α \* ρ(R\_p, R\_b) - (1-α) \* TE

Where α is a parameter controlling the trade-off between maximizing correlation and minimizing tracking error.

**Performance Metrics**

To evaluate how well our index fund tracks the S&P 100, we employ several key metrics:

1. **Correlation**: Measures the linear relationship between fund and benchmark returns. A correlation close to 1 indicates strong tracking.
2. **Tracking Error**: Quantifies the standard deviation of return differences between the fund and benchmark. Lower values indicate better tracking.
3. **R-squared**: The proportion of variance in benchmark returns explained by fund returns. Higher values indicate better tracking.
4. **Information Ratio**: A measure of risk-adjusted returns, calculated as the active return divided by tracking error. Higher values indicate better performance per unit of tracking risk.

**Time Horizon Analysis**

The project requires evaluating performance across multiple time horizons:

* 1 quarter (approximately 63 trading days)
* 2 quarters (approximately 126 trading days)
* 3 quarters (approximately 189 trading days)
* 4 quarters (approximately 252 trading days)

This multi-horizon analysis helps assess the robustness of our approaches under different market conditions and time frames.

The challenge of this optimization problem lies in its combinatorial nature. With 100 potential stocks, there are (100 choose q) possible combinations of stocks to select, which quickly becomes computationally intractable as q increases. Furthermore, the objective function involving correlation and tracking error is non-linear and potentially non-convex, adding another layer of complexity.

For these reasons, we explore both exact optimization methods (using AMPL) and heuristic approaches (using clustering) to find effective solutions to this challenging problem.

**Methodology**

**Optimization-Based Approach**

Our first approach to the index tracking problem employs mathematical optimization techniques using AMPL (A Mathematical Programming Language). This approach aims to find the optimal subset of stocks and their weights that minimize tracking error and maximize correlation with the benchmark.

**AMPL Implementation**

AMPL provides a powerful environment for expressing complex optimization problems. Our AMPL model formulation includes:

1. **Sets and Parameters Definition**:
2. # Sets and Parameters
3. param n; # Number of available stocks
4. param q; # Number of stocks to select
5. param max\_weight; # Maximum weight for any single stock
6. param correlation\_weight; # Weight for correlation component
7. param risk\_weight; # Weight for risk component
8. set TICKERS; # Set of stock tickers
9. param expected\_return{i in TICKERS}; # Expected return for each stock
10. param benchmark\_corr{i in TICKERS}; # Correlation with benchmark
11. param covariance{i in TICKERS, j in TICKERS}; # Covariance matrix
12. **Decision Variables**:
13. # Decision Variables
14. var Select{i in TICKERS} binary; # 1 if stock i is selected, 0 otherwise
15. var Weight{i in TICKERS} >= 0, <= max\_weight; # Weight of stock i in portfolio
16. **Objective Function**:
17. # Objective: Balance between maximizing correlation and minimizing risk
18. maximize Portfolio\_Objective:
19. correlation\_weight \* sum{i in TICKERS} benchmark\_corr[i] \* Weight[i] -
20. risk\_weight \* sum{i in TICKERS, j in TICKERS} Weight[i] \* Weight[j] \* covariance[i,j];
21. **Constraints**:
22. # Constraints
23. subject to Total\_Stocks:
24. sum{i in TICKERS} Select[i] = q;
25. subject to Total\_Weight:
26. sum{i in TICKERS} Weight[i] = 1;
27. subject to Weight\_If\_Selected{i in TICKERS}:
28. Weight[i] <= Select[i] \* max\_weight;
30. # Minimum weight constraint (if selected)
31. subject to Min\_Weight\_If\_Selected{i in TICKERS}:
32. Weight[i] >= Select[i] \* 0.001;

This AMPL model elegantly captures the essence of our optimization problem. It selects exactly q stocks and assigns weights that maximize the objective function while satisfying all constraints.

**Implementation Details**

Our Python implementation of the optimization approach follows these steps:

1. **Data Preparation**:
   * Fetch historical stock data for the S&P 100 constituents using the yfinance package
   * Calculate returns and create training/testing splits
   * Compute the covariance matrix and correlations with the benchmark
2. **Model Generation**:
   * Create the necessary data files for AMPL
   * Generate the optimization model using templates
   * Set parameters like q, maximum weight, and objective weights
3. **Optimization**:
   * Solve the optimization model using an appropriate solver (CPLEX)
   * Extract the selected stocks and their weights
4. **Performance Evaluation**:
   * Calculate portfolio returns using the selected stocks and weights
   * Compute performance metrics (correlation, tracking error, R-squared)
   * Evaluate across different time horizons

The optimization approach leverages the power of mathematical programming to find an optimal or near-optimal solution to the index tracking problem. However, it faces challenges including computational complexity, potential non-convexity, and sensitivity to input parameters.

**Clustering-Based Approach**

Our second approach takes a machine learning perspective, using clustering techniques to identify groups of similar stocks and selecting representatives from each cluster to form our index fund.

**Clustering Algorithm**

We employ K-means clustering to group stocks based on their financial characteristics:

1. **Feature Engineering**: We calculate four key features for each stock:
   * Correlation with the benchmark index
   * Volatility (standard deviation of returns)
   * Average return
   * Beta with respect to the benchmark
2. **Feature Standardization**: We standardize features to ensure equal weighting in the clustering algorithm:
3. scaler = StandardScaler()
4. features\_scaled = scaler.fit\_transform(features)
5. **K-means Clustering**: We set the number of clusters equal to q (the desired number of stocks) and apply K-means:
6. kmeans = KMeans(n\_clusters=max\_clusters, random\_state=42, n\_init=10)
7. clusters = kmeans.fit\_predict(features\_scaled)
8. **Representative Selection**: From each cluster, we select the stock closest to the centroid as the representative:
9. centroid = kmeans.cluster\_centers\_[cluster\_id]
10. distances = np.sqrt(((features\_scaled[cluster\_indices] - centroid) \*\* 2).sum(axis=1))
11. closest\_idx = np.argmin(distances)
12. **Weight Determination**: For simplicity, we initially use equal weighting for selected stocks, which is later replaced with a price-weighted approach:
13. weights = pd.Series([1/len(selected\_stocks)] \* len(selected\_stocks),
14. index=selected\_stocks)

**Implementation Details**

Our clustering implementation follows these steps:

1. **Data Preparation**:
   * Fetch historical data for all S&P 100 constituents
   * Calculate returns and split into training/testing sets
   * Handle missing data through forward filling and filtering
2. **Feature Calculation**:
   * Calculate correlation with benchmark for each stock
   * Compute volatility, average return, and beta
3. **Clustering and Selection**:
   * Apply K-means clustering with q clusters
   * Select the stock closest to each cluster centroid
   * Determine weights for selected stocks
4. **Performance Evaluation**:
   * Calculate portfolio returns using selected stocks and weights
   * Compute tracking metrics
   * Evaluate across different time horizons
5. **Visualization**:
   * Create cluster visualization using PCA for dimensionality reduction
   * Plot performance comparison between the clustered portfolio and benchmark

The clustering approach offers advantages in terms of computational efficiency, interpretability, and robustness to outliers. It provides a natural way to select diverse stocks that represent different segments of the market.

**Comparison Framework**

To compare the two approaches systematically, we developed a comprehensive comparison framework that:

1. Runs both approaches with identical data and parameters
2. Tests different values of q (from 5 to 30 stocks)
3. Calculates performance metrics for each approach and q value
4. Generates visualizations for comparison
5. Identifies the optimal value of q for each approach

This framework enables a fair and thorough assessment of the relative strengths and weaknesses of both approaches across different scenarios.

**Evaluation & Results**

**Experimental Setup**

To thoroughly evaluate our approaches, we designed a comprehensive experimental setup covering various aspects of index fund tracking:

1. **Data Source and Preparation**:
   * Historical stock data for S&P 100 constituents was obtained using the yfinance package
   * Data spanning 4 years (lookback period='4y') was collected for robust analysis
   * Data was split 70/30 into training and testing periods
   * Missing values were handled using forward-filling and filtering techniques
2. **Parameter Variations**:
   * We tested q values of [5, 10, 15, 20, 25, 30] to understand how the number of stocks affects tracking performance
   * For the optimization approach, we set a maximum weight constraint of 10% per stock
   * For the objective function, we used weights of 0.7 for correlation and 0.3 for risk minimization
3. **Testing Environment**:
   * All experiments were conducted using Python 3.8
   * AMPL with CPLEX solver was used for the optimization model
   * Scikit-learn was used for the clustering implementation
   * Consistent random seeds were maintained for reproducibility

**Performance Comparison and Analysis**

The comparative performance of both approaches across different q values reveals several important insights:

**Correlation with Benchmark**

| **Number of Stocks (q)** | **Optimization Correlation** | **Clustering Correlation** |
| --- | --- | --- |
| 5 | 0.000000 | 0.801558 |
| 10 | 0.951508 | 0.814009 |
| 15 | 0.952773 | 0.881617 |
| 20 | 0.954034 | 0.852221 |
| 25 | 0.955338 | 0.884574 |
| 30 | 0.956493 | 0.910225 |

**Key Observations:**

1. The optimization approach failed to find a viable solution for q=5, suggesting limitations for very small portfolios
2. Correlation generally increases with higher q values for both approaches
3. The optimization approach consistently achieves higher correlation than clustering for q ≥ 10
4. Both approaches achieve impressive correlation values (>0.9) at q=30

**Tracking Error**

| **Number of Stocks (q)** | **Optimization Tracking Error** | **Clustering Tracking Error** |
| --- | --- | --- |
| 5 | 0.154861 | 0.105656 |
| 10 | 0.264559 | 0.091423 |
| 15 | 0.263336 | 0.073392 |
| 20 | 0.262155 | 0.081469 |
| 25 | 0.260893 | 0.072694 |
| 30 | 0.259585 | 0.064986 |

**Key Observations:**

1. The clustering approach consistently achieves lower tracking error across all q values
2. Tracking error for the optimization approach is significantly higher despite better correlation
3. For the clustering approach, tracking error generally decreases as q increases
4. The optimization approach shows only minimal reduction in tracking error as q increases

**Analysis of Selected Stocks**

The composition of the selected portfolios provides further insights:

1. **Optimization Approach at q=30**:
   * Selected stocks: AAPL, ACN, ADBE, ADP, AMZN, AVGO, AXP, BLK, BRK-B, COST, CRM, DIS, GOOGL, GS, HD, HON, INTU, ISRG, LIN, MA, META, MS, MSFT, NVDA, QCOM, SBUX, SPGI, SYK, TXN, V
   * Dominated by technology stocks and financial services
   * Top weights distributed equally at 0.1 (10%) among 9 stocks
   * Maximum weight constraint fully utilized
2. **Clustering Approach at q=30**:
   * Selected stocks: CCI, ADBE, MRK, CSCO, NVDA, PM, XOM, INTC, TSLA, MA, PYPL, ZTS, MSFT, META, KO, T, AVGO, ORCL, CI, VZ, BAC, DIS, LLY, LIN, NFLX, CHTR, USB, RTX, AMZN, IBM
   * More sector diversity, including healthcare, telecommunications, energy
   * Equal weighting of 0.0333 (3.33%) across all stocks
   * Greater diversification compared to the optimization approach
3. **Common Stocks Between Approaches**:
   * Only 9 stocks were common between the two approaches at q=30
   * Common stocks: AMZN, MA, DIS, ADBE, LIN, MSFT, META, AVGO, NVDA
   * Suggests fundamentally different selection strategies

**Time Horizon Analysis**

Both approaches were evaluated across different time horizons (3 months, 6 months, 9 months, and 1 year). Key findings include:

1. The optimization approach showed more consistent correlation across time horizons
2. The clustering approach demonstrated improving performance over longer horizons
3. Short-term tracking (3 months) was more challenging for both approaches
4. Both approaches maintained relative performance differences across all time horizons

**Key Findings and Insights**

1. **Optimization vs. Clustering Trade-offs**:
   * Optimization achieves better correlation but higher tracking error
   * Clustering achieves lower tracking error but lower correlation
   * The choice between approaches depends on which metric is prioritized
2. **Impact of q Value**:
   * Performance improves with higher q for both approaches
   * Diminishing returns observed beyond q=20 for optimization
   * Clustering continues to show meaningful improvements up to q=30
3. **Selection Strategy Differences**:
   * Optimization tends to concentrate weights at the maximum allowed
   * Clustering naturally diversifies across more sectors
   * Only 30% overlap in selected stocks between approaches
4. **Computational Considerations**:
   * Optimization requires specialized solver (AMPL+CPLEX)
   * Clustering is computationally more efficient
   * Optimization faced convergence challenges at low q values
5. **Robustness Assessment**:
   * Clustering demonstrated more stable performance across market conditions
   * Optimization showed sensitivity to extreme market movements
   * Both approaches maintained acceptable tracking across the test period

These results suggest that both approaches have merit, but with different strengths. The optimization approach excels at maximizing correlation, making it suitable for investors who prioritize closely matching index movements. The clustering approach achieves lower tracking error, making it appealing for risk-conscious investors who want to minimize deviation from the benchmark.

**Conclusion**

This project tackled the challenge of creating an index fund that tracks the S&P 100 using a subset of constituent stocks through two distinct AI-driven approaches: optimization-based and clustering-based methodologies. Our comprehensive evaluation across different q values (number of stocks) and time horizons revealed key insights into the trade-offs and effectiveness of each approach.

**Summary of Findings**

1. **Effectiveness of Both Approaches**: Both approaches demonstrated the ability to track the S&P 100 index with high correlation and manageable tracking error using substantially fewer than 100 stocks. This confirms the viability of using AI-driven methods for index fund construction.
2. **Performance Comparison**: The optimization approach consistently achieved higher correlation with the benchmark, reaching 0.956 at q=30. The clustering approach delivered consistently lower tracking error, as low as 0.065 at q=30. This presents a clear trade-off between maximizing correlation and minimizing tracking error.
3. **Optimal Portfolio Size**: Performance metrics improved consistently as q increased for both approaches. The best results were observed at q=30, the highest value tested. However, meaningful tracking performance was achieved with as few as 15-20 stocks, particularly for the clustering approach.
4. **Selection Strategies**: The optimization approach favored concentration in high-performing stocks, often utilizing the maximum weight constraint. In contrast, the clustering approach naturally selected a more diverse set of stocks across different market segments, resulting in a more evenly distributed portfolio.
5. **Time Horizon Performance**: Both approaches maintained consistent tracking performance across different time horizons (3 months to 1 year), with slight advantages for longer horizons. This suggests robustness to varying market conditions and investment periods.

**Limitations and Challenges**

Despite the promising results, several limitations and challenges were encountered:

1. **Data Limitations**: The analysis relied on historical data, which may not fully capture future market behaviors. Black swan events and structural market changes remain difficult to account for in any model.
2. **Computational Complexity**: The optimization approach faced computational challenges, particularly for smaller q values where finding feasible solutions became difficult. The AMPL model required careful tuning of solver parameters to achieve convergence.
3. **Parameter Sensitivity**: Both approaches showed sensitivity to parameter choices, such as the weight constraints in optimization and the feature selection in clustering. Extensive parameter tuning may be necessary for optimal real-world implementation.
4. **Limited Feature Set**: The clustering approach used only four features (correlation, volatility, average return, and beta) to characterize stocks. Including additional factors like sector information, fundamental ratios, or alternative risk measures could potentially improve results.
5. **Equal-Weighting Limitation**: The clustering approach defaulted to simple weighting schemes, which might not be optimal. More sophisticated weight determination within clusters could enhance performance.

**Future Improvements**

Several avenues for future work could address these limitations and further enhance the approaches:

1. **Hybrid Approaches**: Combining the strengths of both methods could yield superior results. For instance, using clustering for initial stock selection followed by optimization for weight determination might leverage the advantages of both approaches.
2. **Enhanced Feature Engineering**: Incorporating additional features for clustering, such as factor exposures, industry classifications, or alternative risk metrics, could improve the selection of representative stocks.
3. **Robust Optimization**: Implementing robust optimization techniques that account for uncertainty in return and risk estimates could make the optimization approach less sensitive to estimation errors.
4. **Dynamic Rebalancing**: Extending the framework to include periodic rebalancing rules and evaluating the approaches under dynamic portfolio management would better reflect real-world implementation challenges.
5. **Alternative Machine Learning Methods**: Exploring other machine learning techniques beyond K-means, such as hierarchical clustering, spectral clustering, or even deep learning approaches, might discover more effective stock groupings.
6. **Transaction Cost Modeling**: Incorporating transaction costs and turnover constraints would provide a more realistic assessment of the practical implementation of these approaches.

**Practical Applications**

The methodologies developed in this project have several practical applications:

1. **ETF Construction**: Fund providers can use these approaches to create efficient ETFs that track major indices with reduced holdings, lowering management complexity and costs.
2. **Personalized Indexing**: The frameworks could be adapted for personalized index investing, allowing for customization based on individual preferences while maintaining benchmark tracking.
3. **Factor Tilting**: By modifying the objective functions or clustering features, these approaches could be extended to create factor-tilted index funds that maintain tracking while emphasizing specific investment factors.
4. **Educational Tool**: The comparative analysis serves as an educational tool for understanding the trade-offs in index tracking and the application of AI techniques in portfolio management.

In conclusion, this project demonstrates that AI-driven approaches can effectively create index funds that track benchmark performance using significantly fewer stocks. The optimization and clustering methodologies represent complementary approaches with different strengths, and the choice between them would depend on the specific priorities of the fund manager or investor. Further research and refinement of these approaches could lead to even more efficient and effective index tracking solutions.

**Distribution of Work**

This project was a collaborative effort between team members, with work distributed to leverage individual strengths and ensure comprehensive coverage of all aspects of the project.

**Team Member 1**

* Initial project planning and problem formulation
* Literature review on index tracking and portfolio optimization
* Development of the optimization-based approach:
  + AMPL model design and implementation
  + Performance evaluation framework
  + Parameter tuning and testing
* Documentation of the optimization methodology
* Analysis of optimization results
* Report sections:
  + Introduction
  + Problem Definition
  + Optimization Methodology
  + Portions of Results and Conclusion

**Team Member 2**

* Data collection and preprocessing pipeline
* Development of the clustering-based approach:
  + Feature engineering for clustering
  + K-means implementation for stock selection
  + Performance visualization
* Comparative analysis framework implementation
* Testing different q values across approaches
* Creation of visualization tools
* Report sections:
  + Clustering Methodology
  + Evaluation Framework
  + Portions of Results and Conclusion

**Collaborative Efforts**

* Defining project scope and objectives
* Designing the experimental framework
* Troubleshooting implementation challenges
* Analyzing and interpreting results
* Preparing presentation materials
* Final report review and integration

Both team members contributed significantly to the technical implementation, analysis, and documentation of the project, ensuring a comprehensive and robust solution to the index tracking problem.

The collaboration was facilitated through regular meetings, code reviews, and shared documentation. Version control using Git allowed for effective coordination of development efforts, with each team member responsible for reviewing the other's contributions to maintain code quality and consistency.

This distribution of work allowed us to efficiently tackle the complex challenge of index fund creation while leveraging our individual strengths in optimization, machine learning, and financial analysis.