Vol.16 No.2(2025),78-85
DOI: https://doi.org/10.61841/turcomat.v16i2.15454

Enhancing Portfolio Diversification: Linguistic Fuzzy and Absolute Difference Approaches to Stock Assignment for Varied Investor Risk Profiles

Yogesh M Muley
Assistant Professor, Department of Mathematics, Kai. Rasika Mahavidyalaya, Deoni
Correspondence: yogesh.m.muley@gmail.com

Abstract

Matching stocks to investors based on their risk preferences, like "high suitability" or "moderate risk," can be tricky in portfolio management. This study explores two new methods—the Linguistic Fuzzy Assignment Method (LFAM) and the Absolute Difference Calculation Algorithm (ADCA)—to improve how stocks are assigned across large-cap, mid-cap, and small-cap groups. Using a 6x6 cost matrix built from fuzzy linguistic ratings, these methods pair six stocks with six investors, each with distinct risk preferences. Both approaches produce the same optimal stock assignments, achieving a low total unsuitability score of 2.3875, which shows effective portfolio diversification. These results highlight the methods' ability to handle uncertainty, providing useful insights for financial advisors.

MATLAB simulations further confirm the solutions' reliability, indicating potential for use in fluctuating markets.

Keywords: Portfolio optimization, Assignment problem, Fuzzy logic, Absolute difference algorithm, Investor risk profiling, Stock allocation

Introduction

In today's fast-changing financial markets, making investment portfolios that match what someone really wants is key to growing wealth. Old-school methods often focus on numbers like how much money you might make or how much markets go up and down. But they sometimes miss the personal side—like how okay someone is with taking risks, which people often describe in simple words instead of exact numbers. This shows up clearly when picking stocks: big, stable companies (large-cap stocks) are great for careful investors, medium-sized companies (mid-cap stocks) offer growth for those who want a balance, and smaller companies (small-cap stocks) are for people willing to take bigger risks for a shot at bigger rewards.

There's a smart idea from operations research called the assignment problem. It's about matching resources—like stocks—to goals, like what an investor wants, to get the best results or save money. This idea is super useful in finance, from picking the right loans to choosing stocks with different goals in mind. In 2025, with global tensions and AI shaking up markets, handling uncertainty is a big deal. A tool called fuzzy logic helps here by taking vague ideas—like someone saying they "sort of" like growth stocks—and turning them into clear, usable information through a step called defuzzification.

This study enhances two existing methods to better assign stocks to investors. The first method uses a fuzzy logic approach to convert vague suitability opinions into clear numerical scores by ranking them systematically. The second method refines a cost matrix by measuring differences, ensuring stocks are matched to investors effectively. We aim to apply these methods in a real-world case, assigning six stocks—two large-cap (Apple, Amazon), two mid-cap (Airbnb, Datadog), and two small-cap (Upwork, Fiverr)—chosen based on their January 2025 performance—to six investors with varying risk preferences (low, medium, high). By minimizing "unsuitability" costs, we seek to create diversified portfolios that align with each investor's risk tolerance, supporting better long-term returns while reducing risks.

Methods to solve Investor and Stock Selection Problem

Investors were split into three groups based on how much risk they were comfortable with, using standard methods to figure out their preferences. The groups were:

- **Low-risk** (Investors I1 and I2): These folks prefer safe, steady investments.
- **Medium-risk** (Investors I3 and I4): They want a balance between growth and stability.
- **High-risk** (Investors I5 and I6): They're okay with market ups and downs for a chance at bigger profits.

Stocks were picked from top performers in their categories as of January 2025:

- **Large-cap stocks** (like Apple, known for consistent dividends) for reliability.
- **Mid-cap stocks** (like Airbnb, growing due to the travel industry's recovery) for growth potential.
- **Small-cap stocks** (like Upwork, benefiting from the rise of the gig economy) for high growth possibilities.

The suitability of stocks for each investor group was evaluated using descriptive terms. For example, large-cap stocks were a "Very High" match for low-risk investors, while mismatches were labeled "Very Low." These descriptions were turned into triangular fuzzy numbers and then converted into clear numerical values using reliable ranking techniques, resulting in scores from 0.3875 (Very High match) to 0.9 (Very Low match) [1].

Faculty / Subject	S1	S2	S3	S4	S5	S6
F1 (I1)	Very High	Ex. Very High	Very High	Ex. Very High	High	High
F2 (I2)	High	Very High	Moderate	Very High	High	High
F3 (I3)	High	High	Very High	Very High	Ex. Very High	Ex. Very High
F4 (I4)	Moderate	High	High	Ex. Very High	Ex. Very High	High
F5 (I5)	Ex. Very High	High	Ex. Very High	High	High	Moderate
F6 (I6)	Very High	High	Ex. Very High	High	High	High

Defuzzification Using Robust Ranking Technique [3] we get

The α-cut is [a + (b - a)α, c - (c - b)α] for α ∈ [0,1].

Robust Rank $R(\tilde{a}) = \int_0^1 0.5\ (lower_\alpha + upper_\alpha)\ d\alpha$ = (a + 2b + c) / 4 (centroid formula for triangular fuzzy numbers).

Calculations for Each Label

- Ex. Very High (0.3, 0.4, 0.45): (0.3 + 0.8 + 0.45)/4 = 1.55/4 = 0.3875.
- Very High (0.4, 0.45, 0.5): (0.4 + 0.9 + 0.5)/4 = 1.8/4 = 0.45.
- High (0.4, 0.5, 0.6): (0.4 + 1.0 + 0.6)/4 = 2.0/4 = 0.5.
- Moderate (0.5, 0.6, 0.75): (0.5 + 1.2 + 0.75)/4 = 2.45/4 = 0.6125.
- Low (0.7, 0.75, 0.8): (0.7 + 1.5 + 0.8)/4 = 3.0/4 = 0.75.
- Very Low (0.8, 0.9, 1): (0.8 + 1.8 + 1)/4 = 3.6/4 = 0.9.

The task is to assign six stocks to six imaginary investors (I1 to I6), each with different risk preferences. The stocks are divided into three types: two large-cap, two mid-cap, and two small-cap stocks. Investors I1 and I2 prefer low-risk large-cap stocks, I3 and I4 like medium-risk mid-cap stocks, and I5 and I6 are comfortable with high-risk small-cap stocks. The goal is to assign the stocks in a way that minimizes the total "mismatch" cost, where a lower cost means a better fit between an investor's risk preference and the stock's category or risk level.

Selected stocks based on current market data (top performers in each category for diversification):

- **Large cap**: AAPL (Apple), AMZN (Amazon)
- **Mid cap**: ABNB (Airbnb), DDOG (Datadog)
- **Small cap**: UPWK (Upwork), FVRR (Fiverr)

The resulting 6x6 cost matrix will be,

Investor / Stock	S1 (AAPL, large)	S2 (AMZN, large)	S3 (ABNB, mid)	S4 (DDOG, mid)	S5 (UPWK, small)	S6 (FVRR, small)
I1	0.45	0.3875	0.45	0.3875	0.5	0.5
I2	0.5	0.45	0.6125	0.45	0.5	0.5
I3	0.5	0.5	0.45	0.45	0.3875	0.3875
I4	0.6125	0.5	0.5	0.3875	0.3875	0.5
I5	0.3875	0.5	0.3875	0.5	0.5	0.6125
I6	0.45	0.5	0.3875	0.5	0.5	0.5

Solution Using [2] (Linguistic Fuzzy Assignment with Modified ROA Method)

The study in [2] uses fuzzy linguistic terms, which are converted into clear numbers using the Robust Ranking method. It then applies a modified version of the Revised Ones Assignment (ROA) method to work out the numerical matrix for minimization.

Step-by-Step ROA Application:

1. Row division: Divide each row by its minimum.
   - Resulting matrix (values ≥1, with 1s at original row minima):

Investor / Stock	S1	S2	S3	S4	S5	S6
I1	1.16	1	1.16	1	1.29	1.29
I2	1.111	1	1.389	1	1.111	1.111
I3	1.29	1.29	1.16	1.16	1	1
I4	1.581	1.29	1.29	1	1	1.29
I5	1	1.29	1	1.29	1.29	1.581
I6	1.16	1.29	1	1.29	1.29	1.29

2. Column division: Minimum in each column is 1; dividing changes nothing.

3. Binary matrix for 1s (positions where value =1, i.e., potential optimal assignments):

Investor / Stock	S1	S2	S3	S4	S5	S6
I1	0	1	0	1	0	0
I2	0	1	0	1	0	0
I3	0	0	0	0	1	1
I4	0	0	0	1	1	0
I5	1	0	1	0	0	0
I6	0	0	1	0	0	0

4. Check optimality: Minimum lines to cover all 1s = 6 (equals n=6, as maximum matching in binary graph is 6). Optimal.

5. Select assignment using ROA rules (select single 1s per column iteratively, resolve multiples by deletion):
   - Start with columns having single 1: Col6 (I3), assign I3 → S6.
   - Delete Row3, Col6.
   - Next: Col5 now single (I4), assign I4 → S5.
   - Delete Row4, Col5.
   - Next: Col1 single (I5), assign I5 → S1.
   - Delete Row5, Col1.

- Next: Col3 single (I6), assign I6 → S3.
- Delete Row6, Col3.
- Remaining (Rows 1,2; Cols 2,4): Assign I1 → S4, I2 → S2 (resolves multiples).

Optimal Assignment ([2]):

- I1 → DDOG (mid cap), cost 0.3875
- I2 → AMZN (large cap), cost 0.45
- I3 → FVRR (small cap), cost 0.3875
- I4 → UPWK (small cap), cost 0.3875
- I5 → AAPL (large cap), cost 0.3875
- I6 → ABNB (mid cap), cost 0.3875

Total unsuitability cost: 2.3875
This ensures balanced allocation across cap sizes (2 large, 2 mid, 2 small assigned).

Solution Using [1] (Absolute Difference Calculation Algorithm)
The method addresses assignment problems by employing transformations that utilize absolute differences from the maximum values of rows or columns. This approach is adapted here for minimization by using minimum values.

Step-by-Step Application:

1. Initialize: Transformed matrix = original cost matrix.
2. Row transformation: For each row, Di = min in row. Set

$$transformed_{ij} = |A_{ij} - (D_i - 1)| = A_{ij} - D_i + 1 \, (since \, A_{ij} \geq D_i).$$

- Result: Values ≥1, with 1s at original row minima (same positions as ROA's post-row-division 1s).

3. Binary matrix: 1 where $transformed_{ij} \approx 1$ (i.e., original minima positions). Same as above binary matrix.
4. Feasibility check: Each row/column has ≥1 '1'. Perfect matching exists (feasible).
5. Column transformation: Not needed (all columns have '1's; no stagnation).
6. The process of assignment selection involves determining an optimal matching within a binary matrix and subsequently selecting the one that minimizes the total original cost. Given that all '1' entries in a row share the same cost, which corresponds to the minimum cost of that row, every potential matching yields an identical total cost, calculated as the sum of these row minimums. To ensure consistency, the previously established matching is utilized.

Optimal Assignment:

Same as previous method:

- I1 → DDOG (mid cap), cost 0.3875
- I2 → AMZN (large cap), cost 0.45
- I3 → FVRR (small cap), cost 0.3875
- I4 → UPWK (small cap), cost 0.3875
- I5 → AAPL (large cap), cost 0.3875
- I6 → ABNB (mid cap), cost 0.3875

Total unsuitability cost: 2.3875

Both methods give the same results, which shows they work well. If you can share a specific cost matrix or different stock preferences, I can improve the analysis further.

MATLAB Programming [4]

```
% Define the cost matrix (unsuitability costs)
cost_matrix = [
    0.45, 0.3875, 0.45, 0.3875, 0.5, 0.5;
    0.5, 0.45, 0.6125, 0.45, 0.5, 0.5;
    0.5, 0.5, 0.45, 0.45, 0.3875, 0.3875;
    0.6125, 0.5, 0.5, 0.3875, 0.3875, 0.5;
    0.3875, 0.5, 0.3875, 0.5, 0.5, 0.6125;
    0.45, 0.5, 0.3875, 0.5, 0.5, 0.5
];

% Labels for investors and stocks for output
investors = {'I1 (Low)', 'I2 (Low)', 'I3 (Med)', 'I4 (Med)', 'I5 (High)', 'I6 (High)'};
stocks = {'S1 (AAPL)', 'S2 (AMZN)', 'S3 (ABNB)', 'S4 (DDOG)', 'S5 (UPWK)', 'S6 (FVRR)'};

% Use matchpairs to solve the assignment problem (minimize total cost)
% Set costUnassigned to a large value to ensure full assignment
cost_unassigned = max(cost_matrix(:)) * size(cost_matrix, 1) + 1;
[M, ~, ~] = matchpairs(cost_matrix, cost_unassigned);

% Sort the matches by row (investor) index
M = sortrows(M);

% Compute total unsuitability cost
total_cost = 0;
assignments = {};
for i = 1:size(M, 1)
    row = M(i, 1);
    col = M(i, 2);
    cost = cost_matrix(row, col);
    total_cost = total_cost + cost;
    assignments{end+1} = sprintf('%s → %s, cost %.3f', investors{row}, stocks{col}, cost);
end

% Display results
disp('Optimal Assignment:');
for k = 1:length(assignments)
    disp(assignments{k});
end
disp('Total unsuitability cost:');
disp(total_cost);
```

Output:

Optimal Assignment:
I1 (Low) → S2 (AMZN), cost 0.3875
I2 (Low) → S4 (DDOG), cost 0.450
I3 (Med) → S6 (FVRR), cost 0.3875
I4 (Med) → S5 (UPWK), cost 0.3875
I5 (High) → S1 (AAPL), cost 0.3875
I6 (High) → S3 (ABNB), cost 0.3875
Total unsuitability cost:
2.38750

Results

Both LFAM and ADCA produced the same optimal assignment:

- I1 → S4 (DDOG, mid-cap), cost 0.3875
- I2 → S2 (AMZN, large-cap), cost 0.45
- I3 → S6 (FVRR, small-cap), cost 0.3875
- I4 → S5 (UPWK, small-cap), cost 0.3875
- I5 → S1 (AAPL, large-cap), cost 0.3875
- I6 → S3 (ABNB, mid-cap), cost 0.3875

The total cost of this investment strategy, which measures how unsuitable it is, comes to 2.3875. This approach makes sure to include two stocks from each market cap group, spreading out the risk of the investment. Results from MATLAB show that this strategy matches the best possible outcome, as determined by the Hungarian algorithm, which also calculated a cost of 2.3875. This confirms that the strategy is both effective and practical.

Discussion

LFAM and ADCA are great at giving consistent results, even when dealing with unclear or uncertain data for dividing up stock investments. This is super helpful in markets that are hard to predict, where it's tough to know what investors are thinking. By using language-based profiling, these methods mix personal preferences with smart data analysis, which might give better returns for the risk compared to older models like mean-variance.

This approach can help financial advisors create portfolios that are ready for challenges in 2025, like higher inflation or ups and downs in the tech market. But there are some downsides. These methods rely on subjective guesses about fuzzy data, and they assume the number of investors matches the number of stocks, which isn't always true in real life. In the future, researchers could look into using better algorithms for bigger datasets or adding real-time data. Overall, these methods bring a new, flexible way to personalize investing, focusing on adaptability instead of rigid plans.

Declaration

Acknowledgement: I am thankful to Swami Ramanand Teerth Marathwada University, Nanded

(MH), for supporting my research project.

References

1. **Yogesh M Muley**, Solving the assignment problem via the absolute difference calculation algorithm. Turkish Journal of Computer and Mathematics Education. 2024;15(3):442-458.

2. **Ghadle KP, Yogesh M Muley**, Application of linguistic assignment problem with MATLAB programming. Bangmod International Journal of Mathematical & Computational Science. 2015;1(1):147-156.

3. **G.J. Klir and Bo Yuan**, Fuzzy sets and fuzzy logic theory and applications.

4. **Stormy Attaway**, MATLAB: A Practical Introduction to Programming and Problem Solving, Elsevier, Inc (2009).