Founder Matching: Optimizing Team Formation with Machine Learning

Team Project

December 13, 2025

Contents

1 Introduction

1.1 Project Description and Motivation

Building a successful startup is a high-risk endeavor, and industry data consistently identifies the "team" as a primary determinant of success or failure. The process of finding a suitable co-founder, however, remains largely unstructured, relying on serendipitous networking or inefficient manual filtering. While platforms like YC Match (a "Tinder for founders") exist, they often rely on simple heuristic filters rather than intelligent, data-driven compatibility scoring.

In this project, we aim to conceptualize and build a **Co-Founder Matching System**. Our use-case is a platform that takes a founder's profile—including their skills, personality traits, and startup ideas—and recommends the most compatible partners.

The motivation is to move towards a more nuanced approach that considers:

- **Complementary Skills**: Matching technical founders with business-oriented founders.
- **Behavioral Compatibility**: Aligning risk tolerance, communication styles, and leadership tendencies.
- **Shared Vision**: Matching founders interested in similar problem spaces or industries.

This project applies machine learning techniques to automate and optimize this matching process, aiming to reduce the friction in team formation.

2 Data and Learning Problem

2.1 Dataset Description and Generation

For this project, we required a rich dataset of potential founders. Since public datasets of detailed co-founder profiles are scarce, we constructed a synthetic yet realistic dataset based on real-world social data.

We utilized a seed dataset of social profiles (`Tinder_Data_v3_Clean_Edition.csv`). To transform this into a professional context, we built a data generation pipeline using a Large Language Model (LLM). For each of the approximately 1,200 profiles, we fed the existing biographical data (age, gender, bio) into the LLM with a specific prompt to hallucinate a plausible "Founder Persona."

The resulting `founders_dataset.csv` includes:

- **Professional Attributes**: `preferred_role` (CEO, CTO, COO, etc.), `industry`, `years_of_experience`, `is_technical`, and `education_level`.
- **Skills**: Lists of `tech_stack`, `strengths`, and `weaknesses`.
- **Startup Concept**: `idea_title`, `idea_description`, and `problem_space`.
- **Behavioral Scores**: Numerical scores (scale 1-5 or 0-1) for traits such as `risk_tolerance`, `leadership`, `collaboration_openness`, and `communication_intensity`.

2.2 Data Exploration and Relationships

We expect to see natural correlations in the data that our models can leverage. For instance, "technical" founders (CTOs) likely have high proficiency in specific tech stacks (Python, React) but might self-report lower scores in "sales" or "public speaking" (weaknesses). Conversely, CEOs might index higher on "leadership" and "vision" scores.

We explicitly engineered the generation process to create a balanced distribution of roles (weighted towards CEOs and CTOs) and diverse industries (Fintech, Healthtech, Consumer Social) to simulate a realistic founder pool.

2.3 Challenges with the Data

A significant challenge in this dataset lies in the unstructured text features: `idea_title`, `idea_description`, and `problem_space`. These fields contain full sentences and semantic nuance that are critical for matching founders with a shared vision.

Standard preprocessing techniques like Label Encoding or One-Hot Encoding are unsuitable for these text-heavy features. To fully utilize them, Natural Language Processing (NLP) techniques such as text embeddings (e.g., TF-IDF or BERT) would be required to calculate semantic similarity. While we acknowledge this necessity, extensive NLP implementation is beyond the current scope of this project. For our core models, we primarily rely on the structured categorical data and numerical behavioral scores, treating the text fields as metadata for future expansion.

3 Methodology

To address the matching problem, we propose comparing two distinct machine learning approaches. This comparison allows us to evaluate different paradigms of "team formation":

3.1 Collaborative Filtering (Pair-wise Matching)

This section describes the methodology used to construct a collaborative filtering system for cofounder matching. The system transforms each founder's attributes into numerical representations, computes similarity and complementarity relationships, constructs a combined compatibility matrix, and learns latent archetypes through matrix factorization.

3.1.1 Data Preprocessing and Feature Engineering

Each founder's attributes must first be converted into numerical feature vectors. Features are classified into two categories: (1) similarity features, where alignment increases compatibility (industry, communication style scores, responsiveness, risk tolerance, execution speed, collaboration openness, semantic embeddings from idea text fields), and (2) complementarity features, where differences strengthen a founding team (roles, tech_stack, strengths, weaknesses, preferred roles, experience, education level).

Multi-label variables are encoded as multi-hot vectors. Single-label categories are one-hot encoded. Numerical attributes are standardized. Free-text fields are embedded using a sentence-transformer model to obtain dense semantic representations.

All encoded features are concatenated into a unified matrix X. For similarity computations, each feature vector is L2-normalized. Based on these feature groups, we construct our own "true" compatibility values R_{ij} prior to learning the approximation $\hat{R}_{ij}$ via matrix factorization.

3.1.2 Similarity and Complementarity Computation

Similarity Matrix Similarity is computed across features where alignment matters. Using cosine similarity $\cos(x, y) = (x \cdot y)/(\|x\|\|y\|)$ applied to X_{sim}, we compute the similarity matrix S_{ij}, reflecting alignment in mission, communication style, and idea space.

Complementarity Matrix Complementarity is computed across features where difference is beneficial. For multi-hot vectors, complementarity uses Jaccard distance: $C_{ij} = 1 - J(A, B)$ where $J(A, B) = |A \cap B|/|A \cup B|$. Jaccard ignores co-absence and reflects functional non-overlap. Numerical complementarity uses standardized distances.

Combined Compatibility Matrix Similarity and complementarity are combined with equal weight:

$$R_{ij} = 0.5\, S_{ij} + 0.5\, C_{ij}.$$

3.1.3 Matrix Factorization

Matrix factorization decomposes R into two matrices P and Q, where each founder is represented as a "user" vector $P[i]$ and an "item" vector $Q[j]$. Predicted compatibility is:

$$\hat{R}_{ij} = \mu + b_u[i] + b_i[j] + P[i] \cdot Q[j].$$

Here, μ is the global average, $b_u[i]$ measures how founder i tends to evaluate others, $b_i[j]$ measures how founder j tends to be evaluated, and $P[i] \cdot Q[j]$ models latent interactions.

We minimize:

$$\sum_{i,j} (R_{ij} - \hat{R}_{ij})^2 + \lambda(\|P\|^2 + \|Q\|^2 + \|b_u\|^2 + \|b_i\|^2),$$

using Stochastic Gradient Descent (SGD). Regularization λ prevents overfitting.

3.1.4 Hyperparameter Tuning

The number of latent archetypes A and regularization strength λ are optimized using grid search with cross-validation, selecting the configuration minimizing validation RMSE.

3.1.5 Recommendation Output

After training, the model computes $\hat{R}_{ij}$ for all founder pairs. Ranking these scores produces personalized cofounder recommendations balancing similarity and complementary skill coverage.

3.2 Clustering (Team Formation)

Complementing the pair-wise approach, we use Clustering algorithms (e.g., K-Means or Hierarchical Clustering) to partition the founder pool into groups. This method simulates the formation of co-founding teams (size 2+) by maximizing intra-cluster diversity (ensuring a mix of roles like CEO + CTO) while maintaining compatibility in vision and industry interest.