# Project Overview

Nexa Fund is a new decentralized crowdfunding platform that allows **users to create various types of fundraising campaigns** and enables **donors/backers to contribute** based on their interests, budgets, and causes they care about.

The goal of this project is to build a **smart recommendation system** that can:

# • Recommend campaigns to donors

• Find **similar donors** for profiling, clustering, or insights

This system is critical to ensuring donors are shown campaigns that **align with their values, budgets, and preferences** even when there's no prior user interaction history.

**Approach & ML Techniques Used**

# TF-IDF Embedding

* Donor and campaign text data (bios, interests, tags, descriptions) are processed using **TF-IDF** to convert them into meaningful vector representations.
* TF-IDF captures **word importance** and gives each profile a semantic footprint without needing large pre-trained models.

# Interaction Matrix Creation

* A **synthetic interaction matrix** is generated using cosine similarity between donor and campaign embeddings.
* This matrix simulates partial interactions and is used as input to the recommendation engine.

# NMF (Non-negative Matrix Factorization)

* NMF is applied on the interaction matrix to learn **latent donor and campaign features**.
* It helps uncover hidden patterns and predict interest scores for donor–campaign pairs, even if the donor has no prior engagement.

# Features

* **Donor Recommendations:** Top campaign suggestions are returned for each donor using NMFpredicted scores.
* **Similar Donors Finder:** For any donor, the system retrieves a list of most similar donors using their NMF-derived latent profiles.
* **FastAPI Integration:** A lightweight RESTful API is provided to query recommendations or similar profiles via endpoints.
* **Modular Architecture:** Code is split cleanly into:

o embed\_utils.py: Text + numeric vector generation o ml\_recommender\_scalable.py: Main ML logic (NMF) o app.py: FastAPI backend o donors\_updated.csv & campaigns\_updated.csv: Mock datasets

# Project Strengths

* **Cold Start Ready**: Since TF-IDF uses profile text, new donors and campaigns can be recommended immediately.
* **Interpretable**: NMF + TF-IDF is easy to explain to stakeholders and visualize.
* **Lightweight & Fast**: Requires no GPU, large language models, or external APIs.
* **Scalable**: Ready to plug into live data sources and track real interactions in future.

**Working Example**

**Recommending Campaigns to a Donor**

**🔸 Donors:**

| **Donor ID** | **Bio** | **Interests** |
| --- | --- | --- |
| D1 | I love helping children and education | education, kids |
| D2 | Passionate about health and wellness | health, hospitals |
| D3 | Tech enthusiast and startup backer | AI, blockchain |

**🔸 Campaigns:**

| **Campaign ID** | **Title** | **Description** |
| --- | --- | --- |
| C1 | Help Kids Learn | Support educational supplies for children |
| C2 | Hospital Beds Needed | Donate to hospitals lacking basic equipment |
| C3 | Blockchain for Charity | A decentralized platform to fundraise |

**🔹 STEP 1: Create TF-IDF Vectors**

**Text Fields Combined (Donors):**

* D1: "I love helping children and education education kids"
* D2: "Passionate about health and wellness health hospitals"
* D3: "Tech enthusiast and startup backer AI blockchain"

**Text Fields Combined (Campaigns):**

* C1: "Support educational supplies for children"
* C2: "Donate to hospitals lacking basic equipment"
* C3: "A decentralized platform to fundraise blockchain"

After applying **TF-IDF**, each profile gets a vector like this (simplified example):

| **Entity** | **TF-IDF Vector (keywords)** |
| --- | --- |
| D1 | [0.6 (education), 0.6 (children), ...] |
| C1 | [0.7 (education), 0.5 (children), ...] |
| D2 | [0.8 (health), 0.4 (hospital), ...] |
| C2 | [0.6 (hospital), 0.5 (equipment), ...] |
| D3 | [0.6 (blockchain), 0.4 (startup), ...] |
| C3 | [0.6 (blockchain), 0.5 (fundraise), ...] |

**🔹 STEP 2: Compute Cosine Similarity**

We compute how similar each donor is to each campaign using **cosine similarity**.

| **Donor → Campaign** | **Similarity Score** |
| --- | --- |
| D1 → C1 | 0.92 ✅ |
| D1 → C2 | 0.10 |
| D1 → C3 | 0.05 |
| D2 → C1 | 0.10 |
| D2 → C2 | 0.88 ✅ |
| D2 → C3 | 0.04 |
| D3 → C1 | 0.05 |
| D3 → C2 | 0.04 |
| D3 → C3 | 0.90 ✅ |

This forms the **interaction matrix**:

|  | **C1** | **C2** | **C3** |
| --- | --- | --- | --- |
| D1 | 0.92 | 0.10 | 0.05 |
| D2 | 0.10 | 0.88 | 0.04 |
| D3 | 0.05 | 0.04 | 0.90 |

**🔹 STEP 3: Apply NMF to Learn Latent Factors**

NMF breaks this into two hidden matrices:

* **W (Donor Latent Matrix):** Each donor becomes a vector of *interests* (e.g., education, health, tech)
* **H (Campaign Latent Matrix):** Each campaign becomes a vector of *topics*.

Let’s say after NMF:

| **Donor** | **[Education Score, Health Score, Tech Score]** | | |
| --- | --- | --- | --- |
| D1 | [0.95, 0.05, 0.01] | | |
| D2 | [0.02, 0.96, 0.01] | | |
| D3 | [0.01, 0.01, 0.97] | | |
| **Campaign** | | **[Education, Health, Tech]** |
| C1 | | [0.9, 0.1, 0.1] |
| C2 | | [0.2, 0.8, 0.1] |
| C3 | | [0.1, 0.1, 0.9] |

**🔹 STEP 4: Make Predictions**

Now we multiply donor and campaign latent vectors to get **predicted interest scores**:

**Example: Donor D1**

[0.95, 0.05, 0.01] ×

[ C1: [0.9, 0.1, 0.1] → 0.86 ✅

C2: [0.2, 0.8, 0.1] → 0.15

C3: [0.1, 0.1, 0.9] → 0.10 ]

So D1 gets campaign recommendation:

✅ C1 (Help Kids Learn) with predicted score: 0.86

**✅ Final Recommendations:**

| **Donor** | **Top Recommended Campaign** |
| --- | --- |
| D1 | C1 (Education for Kids) |
| D2 | C2 (Hospital Equipment) |
| D3 | C3 (Blockchain Platform) |

**🔚 Summary**

* **TF-IDF** captures what each person/campaign talks about.
* **NMF** learns “themes” (like Education, Health, Tech) from those descriptions.
* You predict **which campaign matches a donor**, even if they haven’t donated before.