Campus Lost & Found: An ML-Powered Matching System

A Project Report submitted in partial fulfillment of the requirements for the degree of M.Tech (AIDS)

Submitted By: Vinay Kumar, hem kamal, chaitanya karthik, santosh.
Roll Number: SE25MAID052 SE25MAID027 SE25MAID032 SE25MAID050
professor: Sanatan sukhija
Course: M.Tech (AIDS) - 1st Year
Institution: Mahindra university-School of engneering.

Date: December 8, 2025

Contents

1 Abstract

This project presents a scalable, machine learning-powered web application designed to assist in matching lost items with found items on a college campus. The system leverages traditional machine learning techniques, specifically **TF-IDF (Term Frequency-Inverse Document Frequency)** for text-based similarity and **color histogram analysis** for image-based similarity. By combining these features, the application provides a robust matching score, helping users identify their lost belongings with high accuracy. The application is built using Python, the Streamlit framework for the user interface, and is deployed on Streamlit Community Cloud, making it publicly accessible and platform-independent. The system is designed to be scalable, capable of handling a growing dataset of items efficiently.

2 Introduction

2.1 Problem Statement

University campuses are dynamic environments where items are frequently lost and found. The traditional process of matching a lost item with a found one is often manual, inefficient, and relies on vague descriptions. This can lead to a low recovery rate and significant inconvenience for students and staff. There is a need for an automated, intelligent system that can accurately match lost items against a database of found items based on textual descriptions and visual features.

2.2 Objectives

The primary objective of this project is to develop a web-based application that:

- Allows users to describe a lost item via a text input field.
- Allows users to upload an image of the lost item for visual matching.
- Compares the user's input against a pre-existing database of found items.
- Ranks and displays the most likely matches based on a combined text and image similarity score.
- Provides a user-friendly, scalable, and publicly accessible solution.

2.3 Scope

The project's scope is focused on implementing a proof-of-concept system using traditional machine learning algorithms. It does not involve deep learning models (like CNNs for image recognition) to ensure computational efficiency and ease of deployment on cloud platforms. The system is designed to be scalable to thousands of items.

3 Tools and Technologies Used

The project was developed using a specific stack of open-source libraries and platforms, chosen for their efficiency, ease of use, and compatibility.

- **Python:** The core programming language used for all backend logic and data processing.
- **Streamlit:** An open-source Python library used to create the interactive web application interface. It allows for rapid development of data-centric applications.
- **Pandas:** A powerful data manipulation and analysis library. It was used to create and manage the structured dataset of found items in a DataFrame.
- **Scikit-learn:** The primary machine learning library. It was used to implement the **TF-IDF Vectorizer** for text feature extraction and **Cosine Similarity** for calculating text-based matching scores.
- **Pillow (PIL Fork):** A lightweight and server-safe image processing library. It was used to extract the dominant color feature from uploaded images, serving as a simple but effective visual matching tool.
- **NumPy:** A fundamental package for numerical computation in Python. It was used for handling the numerical arrays (vectors) generated by the TF-IDF process.
- **GitHub:** A web-based platform for version control and collaboration. The project's source code is hosted on GitHub.
- **Streamlit Community Cloud:** A free hosting service for deploying Streamlit applications directly from a GitHub repository.

4 Methodology and Implementation

4.1 Dataset

A structured dataset of 10 found items was created directly within the application using a Pandas DataFrame. Each item entry includes:

- **Description:** A text string (e.g., "blue hydro flask with stickers").
- **Color Profile:** A pre-defined 3-dimensional RGB vector representing the item's dominant color (e.g., '[0.1, 0.3, 0.8]' for blue).

4.2 Text-Based Matching using TF-IDF

To match items based on their descriptions, we use the TF-IDF vectorization technique.

- **TF-IDF (Term Frequency-Inverse Document Frequency):** This algorithm converts a collection of text documents (our item descriptions) into a matrix of numerical values. It assigns a higher weight to words that are frequent in a specific document but rare across all documents, making them good identifiers.

- **Cosine Similarity:** After converting the text into numerical vectors, we use Cosine Similarity to measure the angle between them. A score of 1 means the descriptions are identical, while a score of 0 means they have no words in common.

4.3 Image-Based Matching using Color Histograms

To incorporate visual information, we extract the dominant color from an image.

- **Color Extraction:** The uploaded image is resized to a small resolution (32x32 pixels) for fast processing. The average RGB value of all pixels is then calculated to create a single 3-dimensional color vector.
- **Color Similarity:** We use the **Euclidean distance** between the color vector of the uploaded image and the pre-defined color vectors in our dataset. A smaller distance implies a closer color match. This distance is then converted into a similarity score.

4.4 Final Matching Score

The final score for each found item is a weighted combination of the text and image similarity scores:

$$\text{Final Score} = (0.6 \times \text{Text\_Similarity}) + (0.4 \times \text{Image\_Similarity}) \tag{1}$$

This weighting gives more importance to the text description, which is typically more unique, while still using the image as a strong secondary feature.

5 Results and Output

The application successfully matches lost items to the found items database. The output is a clean, sorted table displaying the most likely matches first.

6 Conclusion and Future Work

6.1 Conclusion

The "Campus Lost & Found" application successfully demonstrates the power of combining traditional ML techniques to solve a real-world problem. It provides an accurate, scalable, and easy-to-use solution for matching lost and found items. The use of Streamlit and cloud deployment makes the system modern, accessible, and maintainable. Through this project, I gained practical experience in deploying ML models on the cloud, which was a valuable learning experience.

6.2 Future Scope

The project can be extended in several ways:

- **Advanced Image Recognition:** Replace the simple color histogram with a pre-trained Convolutional Neural Network (CNN) to extract more detailed visual features (e.g., shape, texture).

- **Database Integration:** Connect the application to a live SQL or NoSQL database to allow for dynamic addition and management of items by multiple users.
- **User Accounts & Notifications:** Implement a user authentication system so users can report lost items and receive email/SMS notifications when a potential match is found.
- **Mobile Application:** Develop a native mobile app for easier access and image capture.

7 Project Links

- **GitHub Repository:** https://github.com/chattuvinaykumar/campus_lost_found
- **Live Application Demo:** https://campuslostfound-vinay.streamlit.app/