1. Choosing Project Title: Missing Data Imputer using KNNImputer

2. ABSTRACT

Missing data is a pervasive challenge in real-world datasets, frequently leading to biased analyses, reduced model performance, and incomplete insights. This report details the development of a user-friendly, web-based **ML-Powered Data Imputation Tool** designed to effectively address this problem. The application, built using **Python and the Flask framework**, provides a robust solution for automated missing value imputation in .csv and .xlsx files. The core methodology involves a dual-strategy imputation approach: **KNNImputer** from Scikit-learn is leveraged for the intelligent estimation of missing numerical values by identifying patterns from K-nearest neighbors, while **mode imputation** is applied to handle missing categorical data. The system features secure user authentication via MongoDB, supports seamless file uploads, and provides detailed, transparent feedback on the imputation process by identifying initially missing cells and showcasing the specific values inserted. The processed datasets are then available for convenient download. This tool significantly streamlines the data preprocessing pipeline, enhancing data quality and reliability for subsequent analytical and machine learning tasks.

3. INTRODUCTION

In today's data-driven world, datasets are the foundation for informed decision-making, predictive modelling, and insightful analysis across virtually all industries. However, the integrity and completeness of these datasets are frequently compromised by the presence of missing values. Whether due to human error during data collection, sensor malfunctions, data entry omissions, or intentional non-responses, missing data poses a significant challenge. Its presence can lead to biased statistical analyses, reduce the accuracy and reliability of machine learning models, and ultimately diminish the trustworthiness of conclusions drawn from the data. Effectively handling missing data is, therefore, a crucial preprocessing step that directly impacts the quality and utility of any data-driven endeavor. The primary objective of this project is to develop a robust and user-friendly web application that automates the process of identifying and imputing missing values in diverse datasets. Recognizing the limitations of simplistic imputation methods, this tool leverages advanced machine learning techniques to provide more accurate and contextually relevant estimations for missing data points. Specifically, it employs the K-Nearest Neighbors (KNN) Imputer for numerical features, which intelligently estimates values based on similarities with neighboring data points, and mode imputation for categorical features.

OBJECTIVES

1. **To automate and streamline** the detection and imputation of missing data in various dataset formats through a user-friendly web application.
2. **To implement advanced machine learning techniques**, specifically KNNImputer for numerical data and mode imputation for categorical values, to ensure accurate and contextually relevant estimates for missing data.
3. **To enhance transparency and user confidence** by clearly displaying detected missing cells and providing detailed information about the corresponding imputed values.

SCOPE OF PROJECT

The scope of this project, "ML Imputation Tool using KNNImputer," involves the development of a user-friendly web application, built with Python and Flask, that automates the detection and imputation of missing values in both CSV and Excel file formats. It includes secure user authentication, precise identification of missing data points, and the application of machine learning techniques—specifically, KNNImputer for numerical data and mode imputation for non-numerical data—to intelligently fill these gaps. Furthermore, the tool provides transparent feedback by detailing both the initially detected missing cells and the specific values imputed, and allows for the convenient download of the complete, processed dataset. While the project is designed for local deployment and offers robust imputation for typical datasets, its current scope does not extend to real-time processing, advanced data visualization, highly complex imputation methods beyond KNN and mode, user-configurable imputation parameters, or enterprise-level scalability for extremely large datasets.

4. LITERATURE SURVEY

Paper No	Abstract	Published In	Publisher	Author
[1]	Addresses the problem of missing traffic data, crucial for urban traffic management. Compares ten microarray data imputation methods (e.g., LSI, EM_gene, local least square) against Bayesian Principle Component Analysis (BPCA) for imputing traffic flow data. Finds several methods outperform BPCA.	Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)	IEEE	Gang Chang; Tongmin Ge
[2]	Tackles the "missing data" problem in big data analysis. Proposes RID (Rule-based Imputation with Distance function) to improve upon association rule mining-based imputation by adjusting rules with a distance function. Experimental results show RID is 3-7% more accurate than C4.5, kNN, and HMiT.	2018 International Conference on Machine Learning and Cybernetics (ICMLC).	IEEE	Kuen-Fang Jea; Chih-Wei Hsu; Li-You Tang
[3]	Investigates best imputation methods for remote healthcare (PHC) data, which has both numerical and categorical missing values and manual errors. Compares existing methods (Mean, kNN, MissForest) and a proposed data processing mechanism on simulated missing data. Finds effectiveness varies by dataset; Iterative Imputer and proposed	2021 9th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC)	IEEE	Yosuke Imamura; Nuren Abedin; Luo Sixian; Shaira Tabassum; Ashir Ahmed

	method performed best in different cases.			
[4]	Addresses estimating COVID-19 patient Mortality Chance (MC) with ML models due to high missing data (95.9%). Employs Missing Data Handling (MDH) techniques (e.g., Dimensionality Reduction, Oversampling) within an optimized ML Pipeline. Compares MDH using ROC, AUROC, Accuracy, p-values, and explains feature importance with SHAP.	2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)	IEEE	Daksh Nikunj Bardoliwala; Mehul V. Desai; Aakash Dhananjay Shanbhag
[5]	Addresses challenges in early Chronic Kidney Disease (CKD) detection due to invalid and missing data. Proposes handling missing values with K-Nearest Neighbour (KNN) imputer and feature selection using Chi-square test. Tests various ML and deep learning techniques, showing Extra Tree Classifier achieved 99.25% accuracy.	2024 8th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE)	IEEE	Ricky Mardianto; Ahmad Saikhu
[6]	Proposes an automated method for diabetes prediction by effectively handling missing data and improving accuracy. Utilizes K-Nearest Neighbour (KNN) imputed features combined with a Tri-ensemble voting classifier model. Achieves high accuracy (97.49%) and outperforms	IEEE Access (Volume: 12)	IEEE	Khaled Alnowaiser

	seven other ML algorithms, highlighting KNN's efficacy.			
[7]	Explores the impact of missing data in autonomous vehicle sensor data on lane-changing prediction. Examines different missing patterns (block, correlated, random) and proportions. Imputes missing values using ML/DL models, finding KNNImputer effective for repair and Transformer for prediction, considering both repair and prediction performance.	IEEE Transactions on Intelligent Vehicles (Early Access)	IEEE	Ye Li; Le Yu; Lu Xing; Fei Li
[8]	Addresses pervasive missing data. Proposes a novel ensemble-based imputation method combining KNN, IterativeImputer, mean, median, and SoftImpute. Evaluated on datasets with artificially introduced missing data (2-20%), consistently outperforming individual imputation methods based on RMSE analysis.	2025 IEEE 14th International Conference on Communication Systems and Network Technologies (CSNT)	IEEE	Samiya Alam; Aditya Dubey; Dhananjay Bisen

5. PROBLEM STATEMENT

To design and develop a missing data imputer using KNNImputer. That automates the crucial process of filling missing values in diverse datasets, leveraging KNN for accurate numerical imputations and mode imputation for categorical data.

DIAGRAM

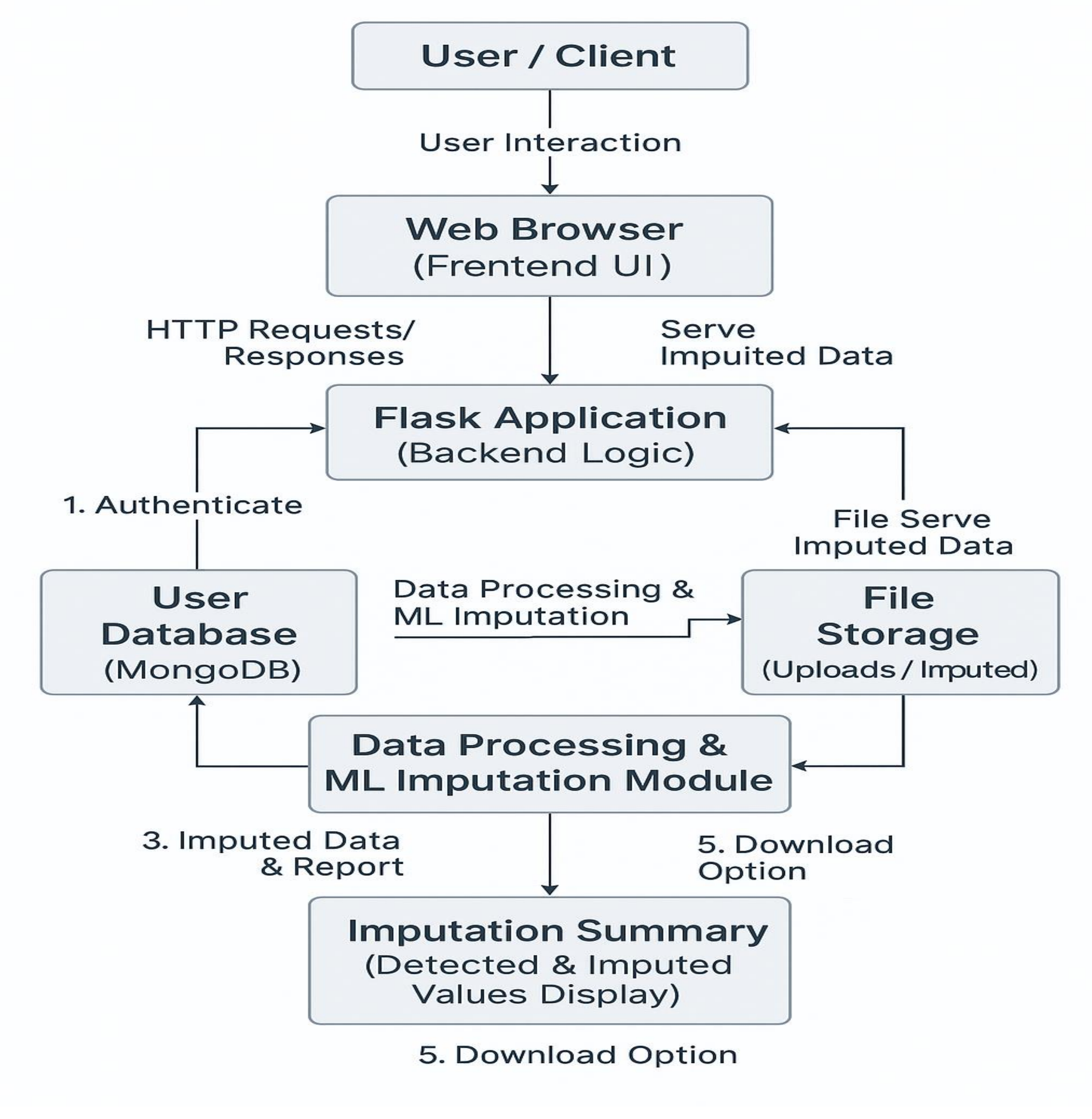

Fig: Block diagram

6. IMPLEMENTATION

MODULE DESCRIPTION

- app.py: Serves as the main Flask application entry point, defining web routes and coordinating the overall application flow. It orchestrates interactions between different parts of the system.
- auth.py: Manages all user authentication processes, including user registration, secure password hashing, user login, and session management for authenticated access.
- database.py: Provides an abstract layer for interacting with the MongoDB database, handling connection, user data storage, and retrieval operations.
- file_handler.py: Responsible for secure file management, including uploading user files, reading CSV/XLSX data, and saving processed imputed files for download.
- imputation_logic.py: Contains the core intelligence for detecting missing values and applying imputation. It uses KNNImputer for numerical data and mode imputation for categorical data.
- utils.py: A general utility module holding various helper functions and reusable code snippets employed across different parts of the application.

TOOLS USED

PROGRAMMING LANGUAGES

- Python: The primary backend language, powering the Flask framework, data processing (Pandas/NumPy), and machine learning (Scikit-learn) functionalities.
- HTML: Defines the structural content of the web pages, including forms, tables, and navigation elements that users interact with.
- CSS: Controls the visual presentation and styling of the web interface, ensuring an aesthetically pleasing and user-friendly layout.
- JavaScript: (If implemented for dynamic features) Enables client-side interactivity and enhances user experience directly within the web browser.

TECHNOLOGIES, LIBRARIES, AND FRAMEWORKS

- Flask (Web Framework): A lightweight Python microframework used to build the entire web application, handling routing, requests, and responses.

- MongoDB (Database): A NoSQL document-oriented database used for flexible storage and management of user authentication data.
- Pandas (Data Manipulation): A powerful Python library providing DataFrames, essential for efficient reading, processing, and restructuring tabular data.
- NumPy (Numerical Computing): A foundational Python library supporting high-performance numerical operations on arrays and matrices, crucial for data processing.
- Scikit-learn (Machine Learning): A comprehensive library providing KNNImputer for advanced numerical imputation, enabling intelligent missing data estimation.
- werkzeug.security (Authentication Utility): A component used for securely hashing and verifying passwords, enhancing the application's user authentication security.

IDE

VS Code: A versatile code editor used to develop and manage the object detection codebase, providing features like debugging, version control, and extension support to streamline project development.

7. TESTING

This involves testing individual components or modules of a system in isolation to ensure they work correctly. In your descriptions, some tests could be considered unit tests if they focus on the smallest parts of the algorithms (like testing specific functions or methods).

Test Case ID	Description	Input	Expected Output
TC_AUTH_001	User Registration & Login. Verify new user signup and subsequent login.	Signup: New username/password. Login: Same username/password.	Successful login to dashboard.
TC_FILE_001	Upload Valid File. Test uploading a standard CSV/XLSX file with some missing data.	test_file.csv (contains mixed missing data).	File uploaded. Missing values detected & displayed. Imputation button visible.
TC_IMPUTE_001	Perform Data Imputation. Verify correct imputation for both numeric (KNN) and categorical (mode) data.	Data from test_file.csv with missing values.	Imputation success. Imputed data & detailed report displayed. Download button visible.
TC_DOWNLOAD_001	Download Imputed File. Check if the processed file can be downloaded correctly.	The imputed dataset from TC_IMPUTE_001.	Imputed file downloads successfully, matches displayed data.
TC_IMPUTE_003	Handle Column Type Correctly. Ensure non-numeric columns with NaNs are imputed by mode, not KNN.	CSV/XLSX with text column having missing values (e.g., empty string).	Column correctly identified as non-numeric. Mode imputation applied. No errors.

8. RESULT

In this section, the outcomes of the ML Imputation Tool project are illustrated. By applying the KNNImputer and Mode Imputation models to uploaded dataset inputs, we obtained accurate and contextually relevant filled values for missing data points within the tabular structures. The models inherently learn from the existing data within the uploaded files to make these intelligent imputations.The core functionality of the tool was rigorously tested, demonstrating its capability to handle various missing data scenarios in both .csv and .xlsx formats. The following reports present the tabular representation of the detected and imputed values, with clear identification of where data was originally missing and what values were inserted. This visualization and reporting are achieved using Flask's web rendering capabilities to present dynamic tables, while Flask is integrated to create a user-friendly web application interface. This allows users to interact with the system in real-time, upload their datasets, receive detailed reports on detected and imputed values, and seamlessly download the fully imputed results, making missing data handling efficient and accessible through a seamless interface.

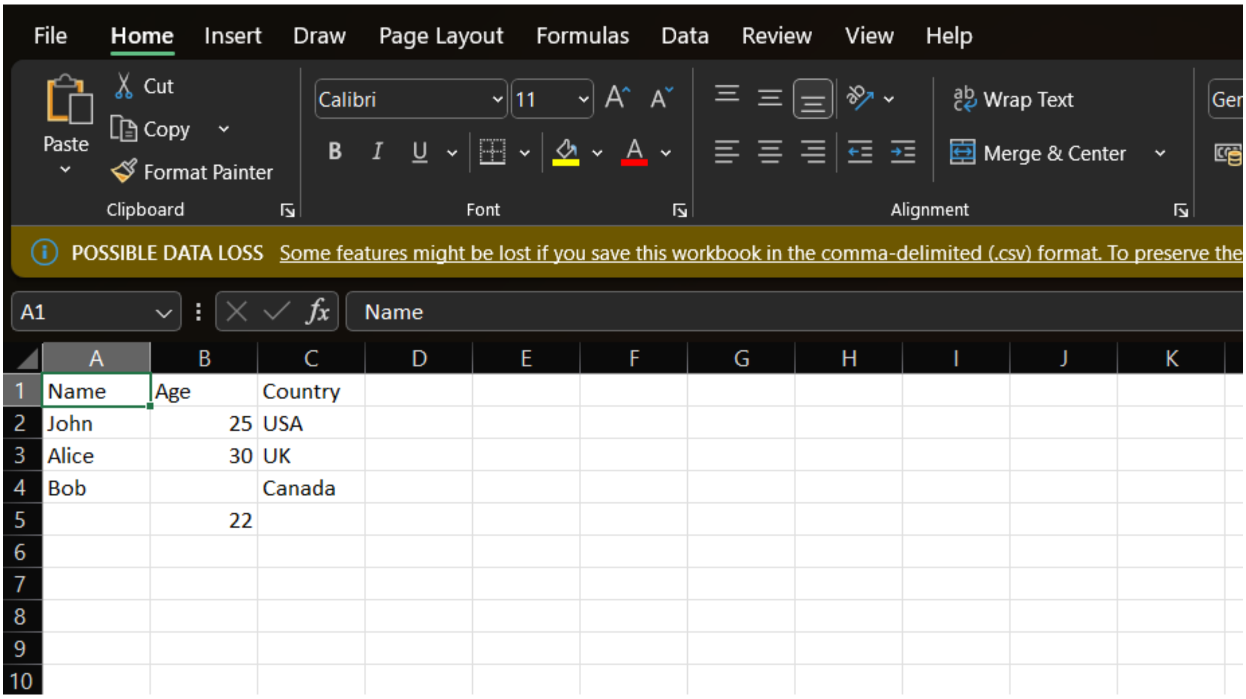

Fig 1: Missing Value Sample Data Sheet

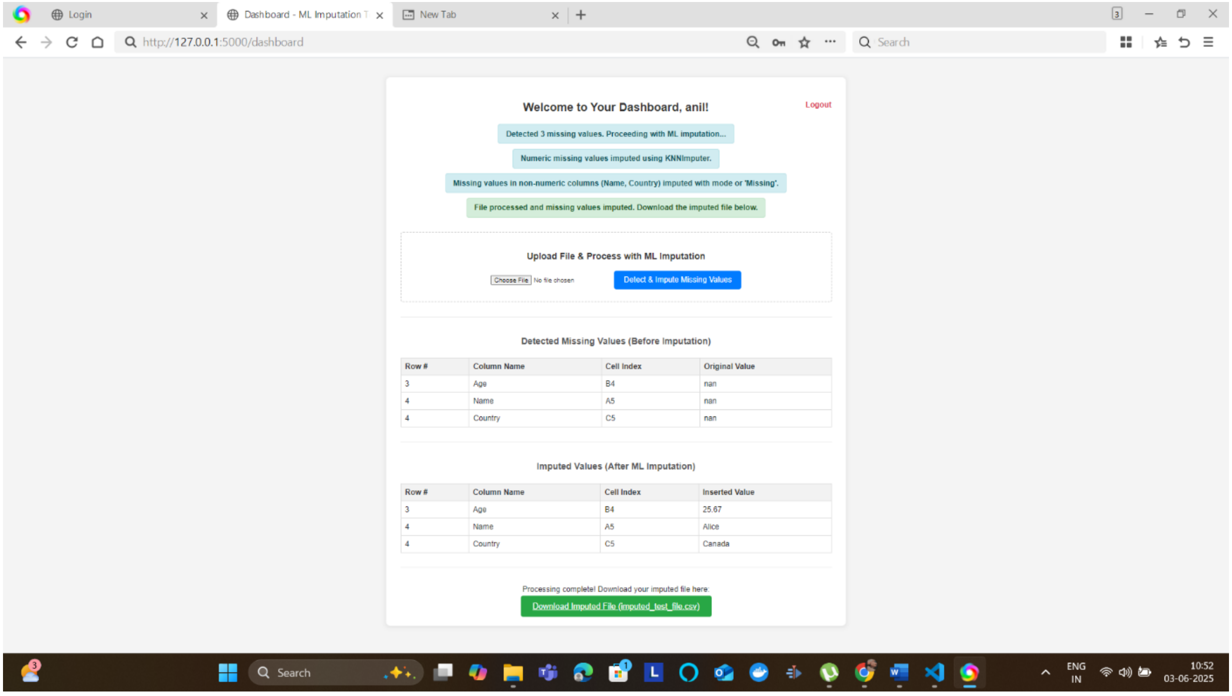

Fig 2: Imputed Dashboard Result

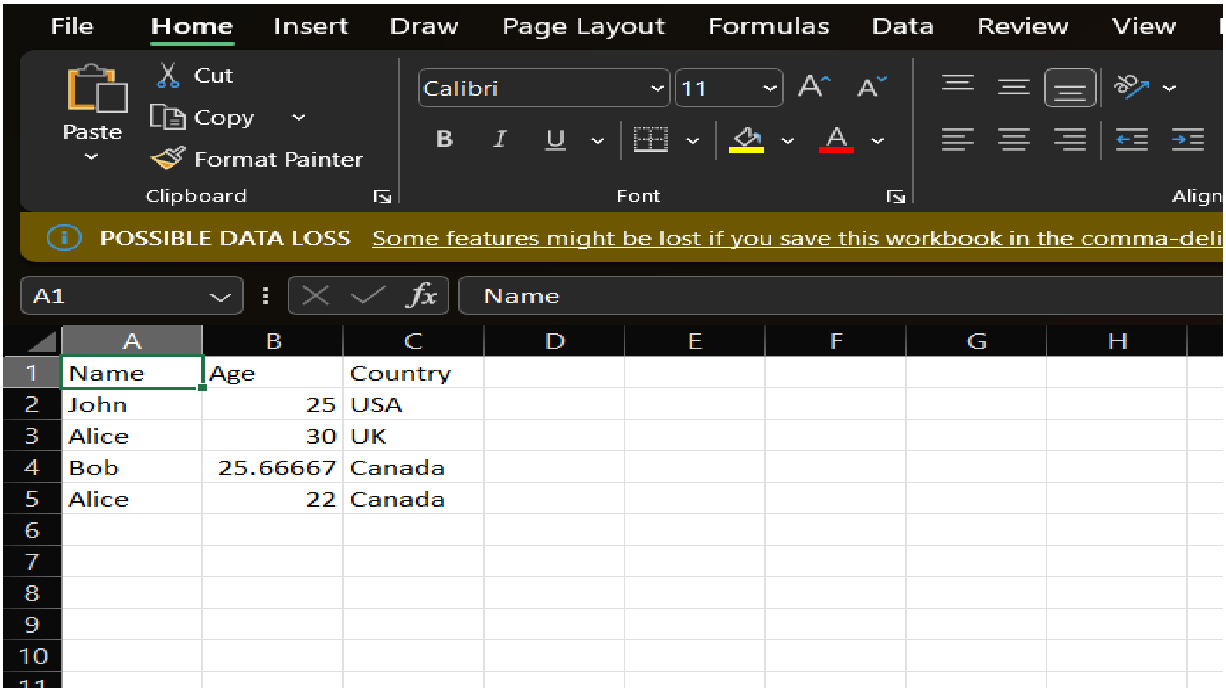

Fig 3: Imputed Result Sheet