Enhanced DAV Project: Automated EDA, Profiling, Visualization, and Interactive Q&A Framework using Python

Team 10

Department of Computer Science, KLH Aziz Nagar

`2310030411@klh.edu.in`

Abstract

This project presents an automated Exploratory Data Analysis (EDA) framework implemented in Python that integrates automated profiling, visualizations, and an interactive question-answer module. The system demonstrates the EDA process using a customer feedback dataset, producing summaries, plots, and profiling reports. The main goal is to minimize repetitive work and help users, especially non-programmers, explore and understand datasets easily.

1 Introduction

Exploratory Data Analysis (EDA) is one of the key stages in understanding a dataset before applying any machine learning model. Performing EDA manually is time-consuming and often repetitive. Automating this process not only saves time but also ensures consistency and reproducibility.

This project presents a framework that performs EDA automatically in a single pipeline. It includes file reading, data profiling, visualization, basic cleaning, and an interactive Q&A feature to make data exploration simple and efficient.

2 Related Work

Existing automated EDA tools like **ydata-profiling (formerly Pandas Profiling)**, **Sweetviz**, and **AutoViz** simplify data exploration but lack features such as an integrated question-answer system or combined visualization modules.

Our framework enhances these tools by integrating automated profiling, statistical visualizations, and an optional AI-based Q&A module, all in one unified structure.

3 Methodology

3.1 Overall Pipeline

The framework follows these main stages:

1. Upload and read the dataset.
2. Display a basic data summary.
3. Generate an automated profiling report.
4. Create visualizations for better understanding.
5. Clean data automatically.
6. Provide an interactive Q&A interface.

Each component is implemented as a Python function, which can be reused independently or as a full pipeline.

3.2 Implementation Environment

The project was developed in Python 3 and tested on Google Colab and Jupyter Notebook. Libraries used include:

- **pandas, numpy** – for data manipulation
- **matplotlib, seaborn, plotly** – for static and interactive visualizations
- **ydata-profiling** – for automated dataset reports
- **openai** – for the interactive Q&A feature

4 Dataset

A sample dataset named `Customer_Feedback.csv` is used for testing. It contains 1,000 records with the following fields: `Customer_ID, Product, Rating, Age_Group, Region, Feedback_Score, Date`.

This dataset was chosen to demonstrate various EDA tasks like summary statistics, correlation, and trend visualization.

5 Code Overview

The main functions of the framework are:

- `upload_file()` – handles file uploads.
- `read_uploaded_file()` – reads CSV/Excel/JSON formats.
- `basic_summary()` – displays shape, types, and missing values.
- `profiling_report()` – creates an HTML summary report.
- `generate_visualizations()` – produces charts and plots.
- `simple_clean()` – handles missing data and duplicates.
- `interactive_qa()` – for user-based dataset queries.

6 Experiments and Results

The framework was applied to the customer feedback dataset to test its functionalities.

6.1 Basic Summary

The dataset includes 1,000 rows and 7 columns. Descriptive statistics show key patterns and help detect outliers in numeric columns like *Rating* and *Feedback Score*. **Why this is useful:** Basic summaries help analysts quickly understand data structure and potential issues before deeper analysis.

6.2 Correlation Heatmap

The correlation heatmap reveals relationships between numerical features. A strong positive correlation was found between *Rating* and *Feedback Score*, meaning customers who give higher ratings often provide higher feedback scores. **Why this is useful:** It helps identify which variables move together and can guide feature selection for predictive models.

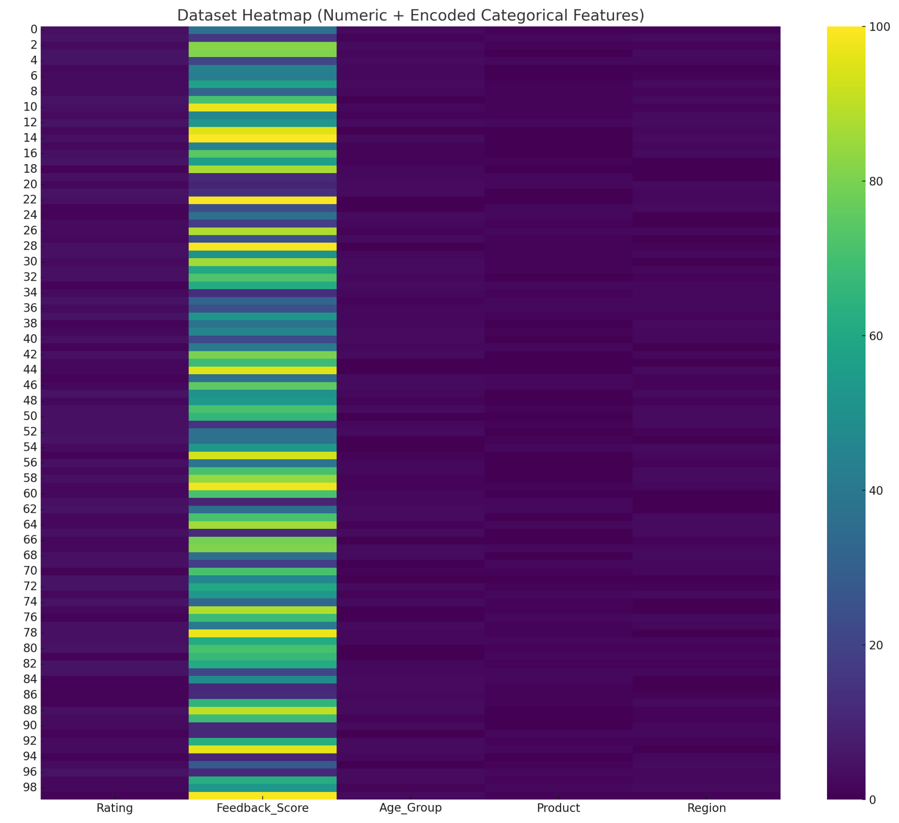

Figure 1: Correlation heatmap between numeric variables.

6.3 Distribution of Ratings

The histogram below shows that most users gave 4 or 5-star ratings, suggesting overall satisfaction with products. **Why this is useful:** Visualizing distributions allows detection of skewness, peaks, and outliers in customer ratings.

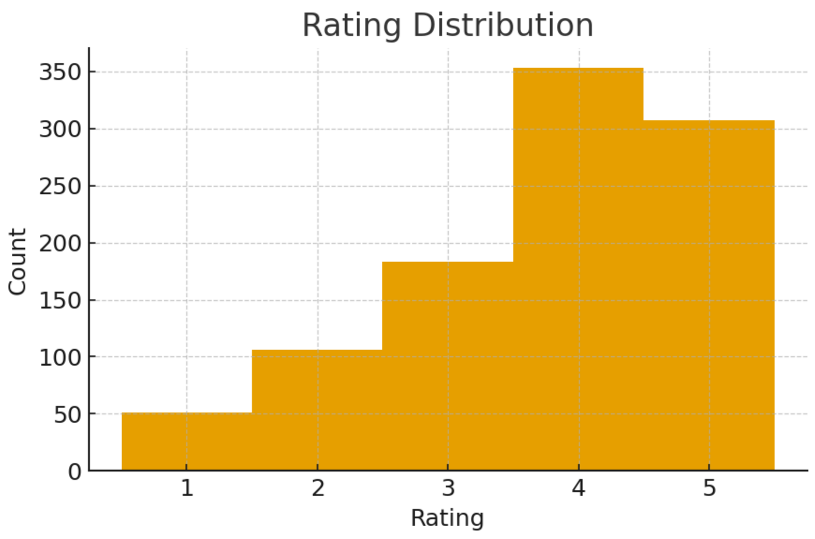

Figure 2: Histogram of customer ratings.

6.4 Top Products

This bar chart identifies the most popular items among users, showing that Product A and Product B received the highest engagement. **Why this is useful:** Such plots help businesses understand product popularity and make better marketing decisions.

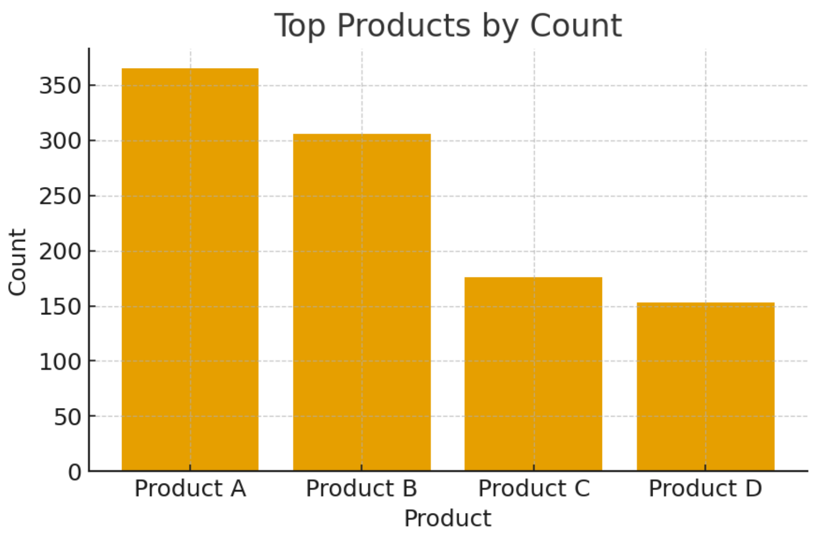

Figure 3: Top products by count.

6.5 Time Series Analysis

The time series plot shows average feedback scores over time. Patterns indicate that user feedback fluctuates with time, which could point to seasonal demand changes. **Why this is useful:** Time-based visualizations help track performance trends and detect seasonal behavior.

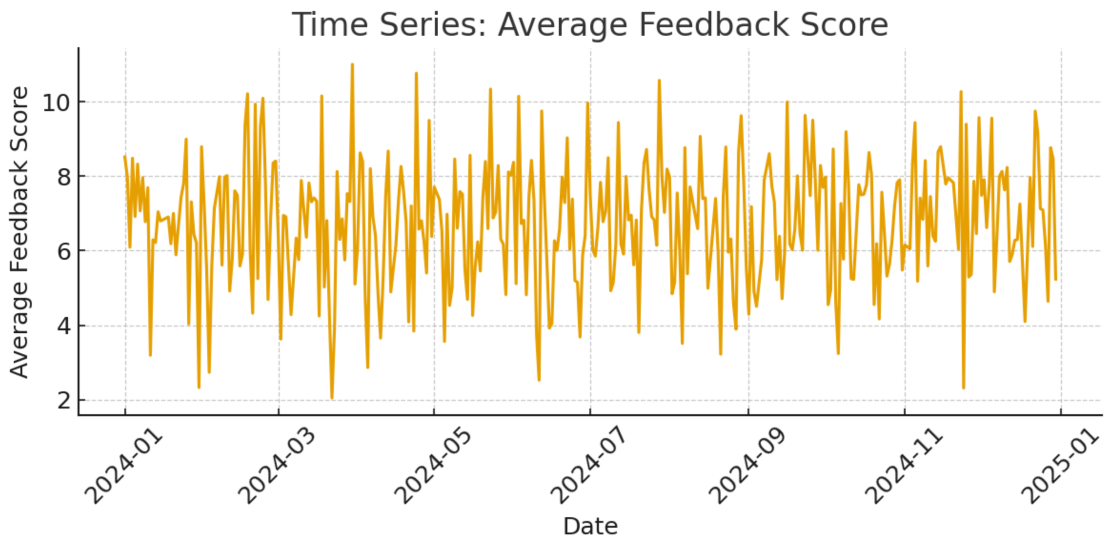

Figure 4: Average feedback score over time.

6.6 Scatter Matrix

The scatter matrix displays pairwise relationships between numeric columns, highlighting clusters and variable distributions. **Why this is useful:** It gives a clear visual of how numerical features relate to each other and helps detect correlations or unusual data points.

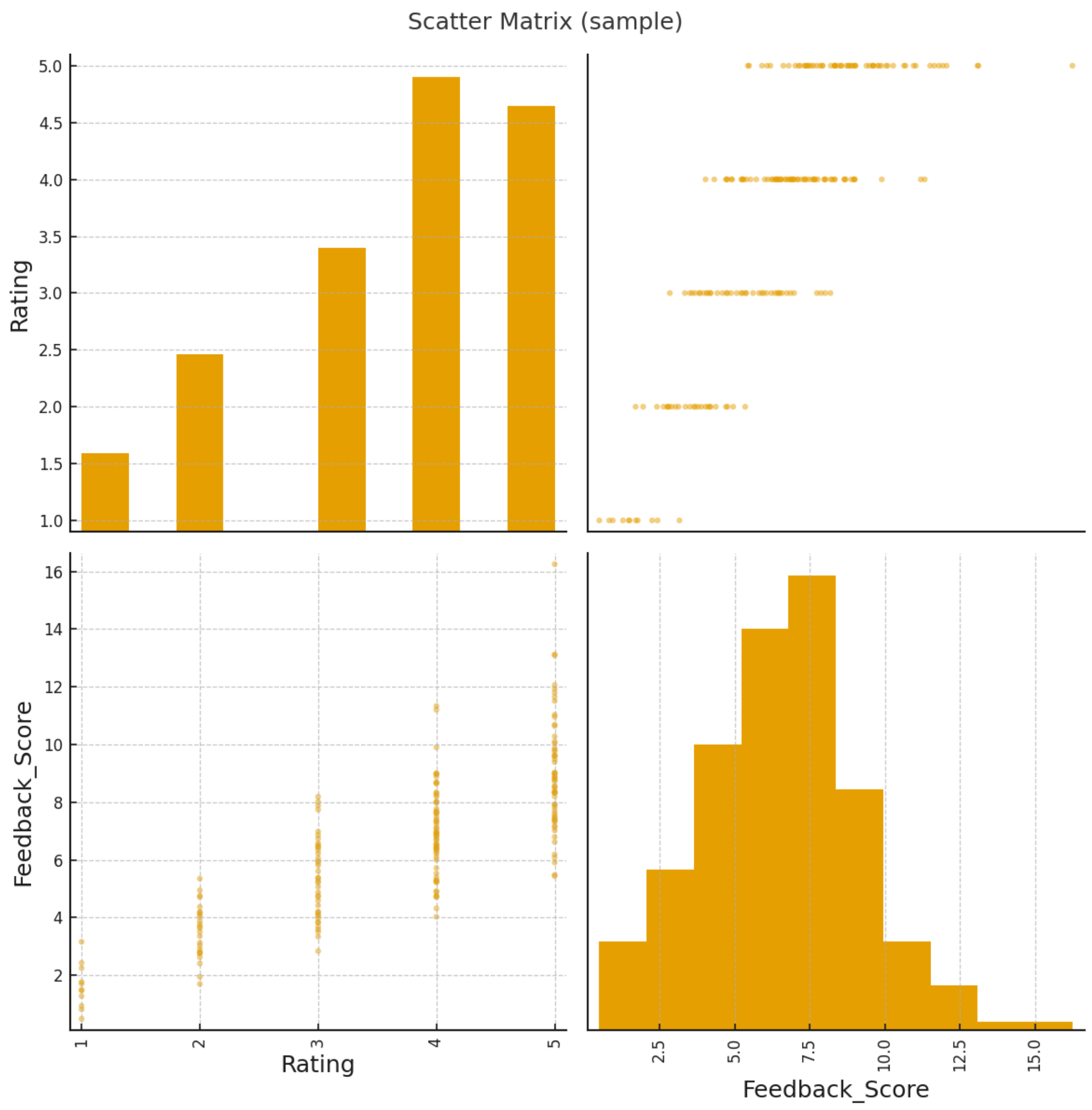

Figure 5: Scatter matrix showing pairwise numeric relationships.

7 Use Cases and Applications

- **Education:** Helps students visualize EDA processes and interpret data easily.
- **Business Analytics:** Enables quick exploration of customer and product feedback.
- **Healthcare:** Can be adapted for patient data exploration and anomaly detection.

8 Conclusion and Future Work

The automated EDA framework successfully simplifies the process of understanding and analyzing data by combining profiling, visualization, and interactive querying in a single system.

By automating key stages of EDA, it helps users uncover insights efficiently, reducing manual workload and promoting consistent analysis across different datasets. This approach also encourages non-technical users to explore data without deep programming knowledge, making it highly valuable for academic, business, and research domains.

The project highlights the importance of integrating automation and AI in modern data analytics workflows. Although the current version primarily focuses on static datasets, future developments will aim to enhance scalability, include real-time data handling, and incorporate adaptive AI-based question-answering modules. Expanding this framework with Streamlit-based dashboards, automated feature engineering, and domain-specific large language models will make it more powerful and user-friendly. Ultimately, this system can evolve into a comprehensive analytical assistant that bridges data science and human understanding seamlessly.

References

1. J. W. Tukey, “Exploratory Data Analysis,” Addison-Wesley, 1977.
2. YData Profiling Documentation – `https://ydata-profiling.ydata.ai/docs/`
3. Wes McKinney, “Python for Data Analysis,” O’Reilly Media, 2018.
4. Seaborn Documentation – `https://seaborn.pydata.org/`
5. Plotly Graphing Library – `https://plotly.com/python/`
6. M. Waskom et al., “Seaborn: Statistical Data Visualization,” Journal of Open Source Software, 2021.
7. D. Grus, “Data Science from Scratch: First Principles with Python,” O’Reilly Media, 2019.
8. J. VanderPlas, “Python Data Science Handbook,” O’Reilly Media, 2016.
9. F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” JMLR, 2011.
10. A. Géron, “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow,” O’Reilly Media, 2022.
11. OpenAI API Documentation – `https://platform.openai.com/docs/`

12. Pandas Documentation – `https://pandas.pydata.org/docs/`

13. NumPy Documentation – `https://numpy.org/doc/`

14. M. Harris, “Automating Data Exploration with Python,” Towards Data Science, 2021.

15. Streamlit Documentation – `https://docs.streamlit.io/`