DeepStat: A Data Analyzer

Project Report submitted in the partial fulfilment

of

Bachelor of Science

In

Data Science

By

Preet Jain (86032200036)
Aakanksha Pote (75312100057)
Sakshi Prabhu (86032200045)

Under the supervision of

Dr. Suresh Pathare

(Associate Professor, Data Science)

SVKM's NMIMS University
(Deemed to be university)

SVKM'S NMIMS Deemed-to-be UNIVERSITY

SCHOOL OF MATHEMATICS, APPLIED STATISTICS AND ANALYTICS (SOMASA)

Kharghar, Navi Mumbai-

(2024-2025)

CERTIFICATE

This is to certify that the project entitled **"DeepStat: A Data Analyzer"**, has been done by **Preet Jain, Aakanksha Pote, Sakshi Prabhu** under the guidance and supervision of **Dr. Suresh Pathare** & has been submitted in partial fulfilment of the degree of Bachelor of Science In Data Science, SVKM's NMIMS (Deemed-to-be University), Mumbai, India.

_____ _____

Project mentor Examiner (Internal Guide)

Date:

Place: Navi Mumbai

_____

(HoD)

ACKNOWLEDGEMENT

We express our heartfelt gratitude to our esteemed project mentor, **Dr. Suresh Pathare**, whose invaluable guidance, insightful feedback, and unwavering support have been instrumental throughout the duration of this project. His expertise, encouragement, and dedication have truly enriched our learning experience and empowered us to overcome challenges with confidence. We are deeply thankful for his mentorship and for sharing his expertise generously, which has undoubtedly contributed to the success of our final year project.

We extend our sincere appreciation to **Dr. Jyoti Verma**, the Head of the Department, for his constant encouragement, support, and belief in our abilities. His leadership and vision have provided us with a conducive environment to explore and innovate, fostering our academic and personal growth. We are grateful for his valuable insights, encouragement, and for always being approachable and supportive throughout our journey.

Furthermore, we would like to extend our gratitude to all the faculty members, staff, and the entire college administration for their support and provision of necessary infrastructure throughout the course of our project. Their collective efforts and dedication have created an enriching academic environment that has nurtured our intellectual curiosity and facilitated our learning and research endeavors. We are deeply thankful for their assistance, encouragement, and for fostering an atmosphere conducive to holistic development.

NAME	ROLL NO	SAP ID

AIM

To develop DeepStat, a no-code data analysis software that empowers non-technical users to perform data science, analytics, and visualization without programming expertise.

OBJECTIVES

- To design and build an intuitive, user-friendly data analytics platform.
- To eliminate barriers of coding knowledge and model application.
- To make data-driven tolls accessible to a wide range of users.

ABSTRACT

In today's data-driven world, organizations across sectors are increasingly recognizing the value of data analytics in strategic decision-making. However, one of the most significant barriers to democratizing data insights is the technical expertise traditionally required to perform such analyses. Programming skills, statistical knowledge, and familiarity with machine learning models are prerequisites that alienate non-technical users. As a result, a considerable population, especially in small businesses, academia, and public sector organizations, remains excluded from the benefits of data analytics due to their lack of coding experience. This project introduces DeepStat, a no-code data analysis platform designed to eliminate these barriers.

DeepStat allows users to interact with data in a structured, step-by-step format. Built using Python and Streamlit, the platform incorporates key functionalities including data uploading, data cleaning, exploratory data analysis, and machine learning model implementation. The software interface has been carefully designed to be intuitive, guiding users through the entire data analysis workflow without requiring any technical background. It supports classification tasks, allows visualization through graphs and charts, and simplifies the preprocessing of datasets through automated options. The main contribution of this project lies in enhancing the accessibility of data analysis. DeepStat addresses the core research gap where individuals lacking technical expertise struggle with performing and interpreting data analytics. By removing the need for programming and enabling users to make sense of their data visually and interactively, the platform promotes inclusive access to data science. The platform has been developed with scalability and modularity in mind, allowing future expansion to include more complex models, advanced analytics techniques, and integrations with databases or cloud storage. The user-centric approach to development involved collecting feedback from a small group of non-technical test users, whose input helped refine the user interface and identify necessary features. Initial testing indicates that DeepStat significantly lowers the entry barrier for data analytics, making it a potentially transformative tool for education, research, small-scale enterprises, and more.

In conclusion, DeepStat exemplifies the convergence of AI and usability. It creates an environment where users from diverse backgrounds can conduct meaningful analysis, visualize insights, and make data-driven decisions without needing to learn programming. It fills a critical void in the market for no-code analytical tools and contributes toward making data science more inclusive.

Table of Contents

CHAPTER 1

INTRODUCTION

In the current era of digital transformation, data has emerged as a fundamental resource, shaping decision-making processes and strategic frameworks across virtually every domain. From commercial enterprises to educational institutions and public sector organizations, the ability to derive meaningful insights from data has become an indispensable component of success. As organizations continue to generate and accumulate vast amounts of data, the emphasis on data-driven decision-making has grown exponentially. This paradigm shift has, in turn, elevated the role of data analytics as a vital enabler of operational efficiency, innovation, and competitive advantage. However, despite the increasing ubiquity of data and the proliferation of analytical tools, the benefits of data science remain disproportionately accessible to a select group of individuals equipped with the requisite technical expertise.

Traditional data analysis pipelines typically demand proficiency in programming languages such as Python or R, familiarity with statistical principles, and an understanding of machine learning algorithms. These prerequisites, while manageable for trained data professionals, pose a significant obstacle for non-technical users. Small business owners, educators, researchers from non-technical disciplines, and administrators often find themselves excluded from the realm of data analytics due to a lack of coding skills or statistical training. Consequently, valuable data frequently goes underutilized or misinterpreted, resulting in missed opportunities for optimization, forecasting, and informed decision-making.

This disparity underscores a pressing need within the analytics ecosystem: the democratization of data science. Democratization, in this context, refers to the process of making data tools accessible to all individuals, regardless of their technical background. It entails the development of intuitive, user-friendly platforms that abstract the complexities of traditional data workflows while preserving analytical rigor. A growing number of initiatives have attempted to address this challenge through low-code and no-code platforms, which provide graphical interfaces and automation to reduce the cognitive and technical load on users. Yet, many of these solutions either lack depth in analytical capabilities or are tailored to specific industries, limiting their general applicability.

1.1. The Importance of Data and the Skill Gap

In the contemporary landscape, characterized by an unprecedented proliferation of digital information, data has emerged as a pivotal asset, driving strategic decision-making across a multitude of sectors. Organizations, irrespective of their domain or scale, are increasingly recognizing the intrinsic value embedded within their datasets, understanding that the ability to effectively extract, analyze, and interpret this information is crucial for maintaining a competitive edge, fostering innovation, and achieving operational excellence (Provost & Fawcett, 2013). This data-driven paradigm necessitates a widespread capacity for data analytics, transforming it from a niche technical skill to a fundamental competency for informed action.

However, despite the growing recognition of data's significance, a substantial impediment persists in the democratization of data insights: the significant technical expertise traditionally required to perform meaningful analyses. Conventional data analytics workflows often necessitate proficiency in programming languages such as Python or R, a robust understanding of statistical methodologies, and familiarity with complex machine learning algorithms. These prerequisites create a substantial barrier to entry for individuals who lack formal training in computer science, statistics, or related quantitative disciplines. Consequently, a considerable segment of the population, particularly within small businesses, academic institutions, and public sector organizations, remains excluded from the transformative potential of data analytics due to their lack of coding experience and specialized technical knowledge. This exclusion not only limits the ability of these entities to leverage their own data for informed decision-making but also hinders broader societal progress by concentrating analytical capabilities within a relatively small pool of experts.

The implications of this "analytical divide" are far-reaching. Small businesses, often operating with limited resources, may struggle to identify market trends, optimize their operations, or personalize customer experiences due to a lack of accessible analytical tools. Academics in non-quantitative fields may find it challenging to rigorously analyze research data, hindering the potential for data-driven discoveries. Public sector organizations might be unable to effectively utilize data to inform policy decisions, optimize service delivery, or address societal challenges. This underscores the urgent need for solutions that can bridge this gap and empower a wider range of users to engage with data in a meaningful and productive manner.

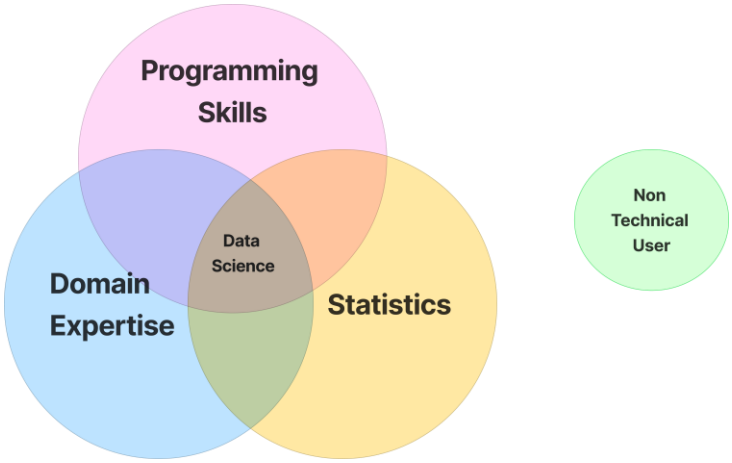

Fig 1.1.1 Key Venn diagram showing the intersection of programming, statistics, and domain expertise required for traditional data science, highlighting the exclusion of non-technical users

1.2 Introducing DeepStat

This report introduces DeepStat, a novel no-code data analysis platform designed to address the challenges and empower non-technical users to perform data science, analytics, and visualization without requiring any programming expertise. DeepStat aims to eliminate the traditional barriers associated with data analysis by providing a structured, step-by-step user interface that guides individuals through the entire data analysis workflow in an intuitive and accessible manner.

Built utilizing the versatility and extensive libraries of the Python programming language, coupled with the interactive web application framework of Streamlit, DeepStat offers a comprehensive suite of functionalities tailored to the needs of non-technical users. The platform incorporates key stages of the data analysis pipeline, including seamless data uploading from various sources, robust data cleaning and preprocessing capabilities, comprehensive exploratory data analysis (EDA) through interactive visualizations, and the implementation of fundamental machine learning models for classification tasks.

The core design principle of DeepStat is user-centricity. The software interface has been meticulously crafted to be highly intuitive, employing clear visual cues, guided workflows, and automated options to simplify complex tasks. Users are guided through each stage of the analysis process, from loading their data to interpreting the results, without the need to write code or possess deep statistical knowledge. The platform supports the visualization of data through a variety of commonly used graphs and charts, enabling users to visually identify patterns, trends, and anomalies within their datasets. Furthermore, DeepStat simplifies the often-tedious process of data preprocessing by offering automated options for handling missing values, encoding categorical variables, and scaling numerical features.

CHAPTER 2

LITERATURE REVIEW

2.1. The Rise of Data Analytics

2.1.1 The Rise of Data and Evolving Landscape of Data Analytics

The exponential growth in data generation across diverse domains has fundamentally reshaped organizational strategies and decision-making processes. This era of "big data" is characterized by datasets of unprecedented volume, velocity, and variety, presenting both significant opportunities and considerable challenges for extracting meaningful insights. The ability to effectively analyze and interpret these vast quantities of information has become a critical determinant of competitive advantage, driving innovation, and fostering evidence-based practices across industries. Consequently, the field of data analytics has witnessed significant advancements, encompassing a wide array of methodologies, tools, and techniques, ranging from traditional statistical analysis to sophisticated machine learning algorithms and artificial intelligence-driven approaches.

The evolution of data analytics has been marked by a shift from primarily descriptive and diagnostic analyses, focused on understanding past trends and identifying the causes of specific outcomes, towards more predictive and prescriptive approaches, aimed at forecasting future events and recommending optimal courses of action. This progression has been facilitated by advancements in computational power, algorithmic development, and the increasing availability of user-friendly software tools. However, the effective utilization of these advanced analytical techniques has traditionally required a high degree of technical expertise, encompassing programming skills, statistical knowledge, and familiarity with the intricacies of machine learning models.

The dominance of code-intensive tools and methodologies has inadvertently created a significant barrier to entry for individuals and organizations lacking specialized technical capabilities. This "analytical divide" has limited the widespread adoption of data-driven decision-making, particularly within segments such as small and medium-sized enterprises (SMEs), academic disciplines outside of quantitative fields, and public sector organizations, where access to dedicated data science teams and resources may be constrained. The inability of these entities to fully leverage their data assets represents a significant impediment to their growth, efficiency, and overall effectiveness, highlighting the urgent

need for solutions that can democratize access to data analytics and empower a broader range of users.

The rise of no-code data analytics platforms represents a natural progression in this evolution, specifically addressing the technical barriers associated with data exploration, analysis, and modeling. These platforms leverage similar principles of visual interaction and pre-configured functionalities to enable non-technical users to engage with data in a meaningful way. By abstracting away, the need for programming languages like Python or R, and simplifying statistical concepts and machine learning algorithms through user-friendly interfaces, no-code data analytics platforms aim to democratize data science and empower a wider audience to extract valuable insights from their data.

2.1.2 The Rise of Code-Free Data Analysis Tool

The concept of abstracting away the complexities of software development to empower non-technical users has a rich history, predating the current surge in data analytics. Early efforts in end-user programming and visual development environments aimed to lower the barrier to software creation, enabling individuals with domain expertise but limited coding skills to build custom applications. These early paradigms laid the groundwork for the more sophisticated no-code development platforms that have emerged in recent years, driven by advancements in web technologies, cloud computing, and artificial intelligence.

The current generation of no-code platforms extends beyond basic application development, encompassing a wide range of functionalities, including website building, workflow automation, database management, and increasingly, data analytics. These platforms typically provide intuitive graphical interfaces, drag-and-drop functionalities, and pre-built components [1] that allow users to construct complex applications and workflows without writing any code. The underlying technical complexities are handled by the platform, allowing users to focus on the logic and design of their solutions.

This evolution has led to the rise of platforms like Airtable, Bubble, Power BI, and Tableau, each catering to specific domains of no-code development. For instance, Airtable combines spreadsheet simplicity with database power, enabling users to create relational data models without SQL knowledge. Bubble allows users to design and launch full-stack web applications through a visual editor, while Power BI and Tableau empower users to build interactive dashboards and perform data analysis through drag-and-drop interfaces and built-in visual analytics tools. These tools have democratized access to technology by enabling marketers, educators, analysts, and entrepreneurs to create data-driven applications and insights without relying on developers or data scientists. This paradigm shift is central to platforms like DeepStat, which continue this legacy by abstracting the complexities of machine learning and statistical modeling, making advanced analytics accessible to everyone.

2.2 Overview of Existing Tools

A review of existing no-code data analytics platforms reveals a common set of core features and functionalities designed to guide users through the typical data analysis workflow without requiring coding expertise. These platforms generally offer capabilities for seamless data ingestion from various sources, including spreadsheets, databases, and cloud storage. Once data is uploaded, they provide intuitive interfaces for data cleaning and preprocessing, often incorporating automated suggestions and visual tools for handling missing values, filtering data, transforming variables, and ensuring data quality.

Exploratory Data Analysis (EDA) is a crucial component of these platforms, offering a range of interactive visualization tools such as charts, graphs, and dashboards that allow users to visually inspect data patterns, identify trends, and formulate hypotheses. These visualizations are typically dynamically linked to the underlying data, enabling users to drill down into specific subsets and gain deeper insights. Furthermore, many no-code platforms are incorporating functionalities for building and deploying machine learning models. While the underlying algorithms may be complex, the user interface often simplifies the process to selecting a model type, choosing relevant features, and training the model through a visual interface. Model evaluation metrics and interpretability tools are also frequently provided to help users understand the performance and implications of their models.

Examples of existing no-code and low-code data analytics platforms include Tableau Prep Builder and Tableau Desktop, which focus heavily on data preparation and visualization; Alteryx, which offers a more comprehensive suite of data blending, advanced analytics, and data science tools through a visual workflow paradigm; and KNIME, an open-source platform that provides a visual workbench for data science tasks. In recent years, numerous data analytics tools have emerged to aid users in interpreting complex datasets. Traditional platforms such as Microsoft Excel, RStudio, and Python-based Jupyter Notebooks require some level of coding or statistical knowledge. While powerful, these tools often have steep learning curves, making them inaccessible to non-programmers or beginners in data science. Platforms like Microsoft Power BI also offer robust data connectivity, transformation, visualization, and basic machine learning capabilities within a user-friendly interface. While these platforms have made significant strides in democratizing data analytics, they often still require a degree of familiarity with data concepts and analytical workflows and may not fully cater to users with absolutely no technical background or prior experience with data analysis software. Moreover, the specific functionalities, ease of use, and level of automation can vary considerably across different platforms, highlighting the ongoing need for solutions that further simplify and streamline the data analysis process for non-technical users

To mitigate this, no-code or low-code platforms such as Tableau, Power BI, Orange, and Google Data Studio have gained popularity. These tools emphasize visual interfaces and drag-and-drop functionality but often lack the depth of functionality found in programming-based environments, particularly for machine learning or automated data preprocessing.

Recently, open-source tools like Streamlit and Gradio have allowed developers to convert Python scripts into shareable web applications. However, most implementations using these tools still require the user to write code and have technical proficiency to modify or deploy them effectively.

2.3 Research Gap

While existing no-code data analytics platforms have lowered the barrier to entry compared to code-intensive tools, a significant research gap remains in creating truly accessible and intuitive solutions for individuals with no prior technical background. Current platforms often still presume a degree of data literacy and familiarity with analytical concepts, posing challenges in understanding data structures, selecting appropriate techniques, interpreting outputs, and navigating machine learning nuances. Users may need to develop new mental models for data manipulation, and the abstraction provided can obscure underlying processes, potentially leading to misinterpretations. Furthermore, automation in data cleaning and feature engineering may be insufficient or opaque, still requiring some understanding of data quality.

Therefore, a need exists for no-code platforms that transcend simple visual interfaces and offer a genuinely guided, step-by-step experience tailored to novice users. Such platforms should further simplify complex concepts, provide contextual guidance, offer intelligent automation of routine tasks, and prioritize intuitive visual communication of results that is easily understandable and actionable for those without formal data science or statistical training. Addressing this gap can unlock the power of data analytics for a broader population, enabling data-driven decision-making in domains with limited technical expertise.

2.4 Problem Definition

While data analysis tools have evolved, they still largely cater to users with some technical background. Many professionals—such as business analysts, educators, healthcare workers, and policymakers—face difficulty in applying statistical and machine learning techniques due to limited programming knowledge. These users require platforms that: Are user-friendly, with no need for coding. Allow flexible data input, such as uploading CSV files. Automate data cleaning, transformation, model selection and present visual and interpretative outputs that are easy to understand.

Current tools either lack depth (in terms of ML capabilities) or simplicity (in terms of usability), leaving a significant portion of potential users underserved.

CHAPTER 3

METHODOLOGY

The DeepStat platform was designed with the goal of simplifying the data analysis process while maintaining the flexibility and power of traditional data science workflows. Built using Python and Streamlit, the system integrates various machine learning, visualization, and preprocessing modules to provide an all-in-one analytical tool accessible through a web interface. This section describes the architecture, tools used, workflows, and individual modules of the system in detail.

3.1 Conceptualization

The conceptual foundation of DeepStat stemmed from an acute recognition of the challenges non-programmers face in accessing data analytics tools. The traditional reliance on coding knowledge and statistical theory represents a considerable barrier to many users in academia, small businesses, and public administration. The project, therefore, was conceptualized around three fundamental principles: accessibility, modularity, and automation.

Accessibility ensures that users can perform complex analytical tasks through a guided user interface, eliminating the need for scripting or command-line execution. Modularity permits the compartmentalization of functionalities into logically distinct components such as data exploration, correlation, preprocessing, visualization, and supervised modeling, unsupervised modeling and dashboard. Automation underpins the internal logic that drives the transformation and analysis of data behind the scenes, offering an experience that is both user-friendly and technically robust.

3.2 Technology Stack

The technology stack used in DeepStat includes Python 3.9+ as the primary programming language. The Streamlit library powers the user interface, providing real-time interactivity and dynamic rendering of content. For data handling and preprocessing, libraries such as Pandas and NumPy are employed, while visualizations are created using Seaborn, Matplotlib, and Plotly. Machine learning algorithms are implemented using Scikit-learn and XGBoost, while time series forecasting capabilities are enabled through Statsmodels and

Prophet. The final deployment can be done locally or hosted on Streamlit Cloud.

The choice of technologies for DeepStat was guided by the need for simplicity, power, and community support. Python was selected as the core programming language due to its dominant role in the data science ecosystem and its extensive library support. Streamlit was chosen as the web application framework for its unique ability to transform Python scripts into interactive web applications with minimal boilerplate code. Together, Python and Streamlit formed a cohesive and highly functional environment for rapid development and deployment.

The combination of these libraries (Table allowed the project to handle all aspects of the data pipeline, from ingestion to visualization to prediction, all while keeping the source code clean and maintainable.

Component	Technology Used
Programming Language	Python 3.9+
Web Framework	Streamlit
Data Handling	Pandas, Numpy
Visualisation	Matlplotlib, Seaborn
Machine Learning	Scikit-learn, XGBoost
Deployment	Streamlit Cloud

Fig 3.2.1 The above table shows the technology and libraries used while developing DeepStat

3.3 System Architecture

The architecture of DeepStat follows a modular structure, where each function—data uploading, preprocessing, exploratory analysis, modeling, and visualization—is organized as a self-contained block. This not only enhances usability but also facilitates future expansion and scalability. The front end of the application is powered by Streamlit, allowing users to interact with the tool in real time through a web-based graphical interface. The back end processes the uploaded dataset using Python's data manipulation

libraries and then applies machine learning models from libraries such as Scikit-learn and XGBoost.

The architecture of DeepStat is inherently layered, with each layer responsible for a specific set of operations. The three primary layers consist of:

Presentation Layer (User Interface) - Built entirely using Streamlit, this layer manages user interaction, visual representation, and input collection. Application Logic Layer (Processing Engine) - Comprising all data preprocessing, model training, evaluation, and transformation logic. Data Management Layer - Quantitative Responsible for loading, storing, and modifying the uploaded datasets throughout the session using Streamlit's state persistence.

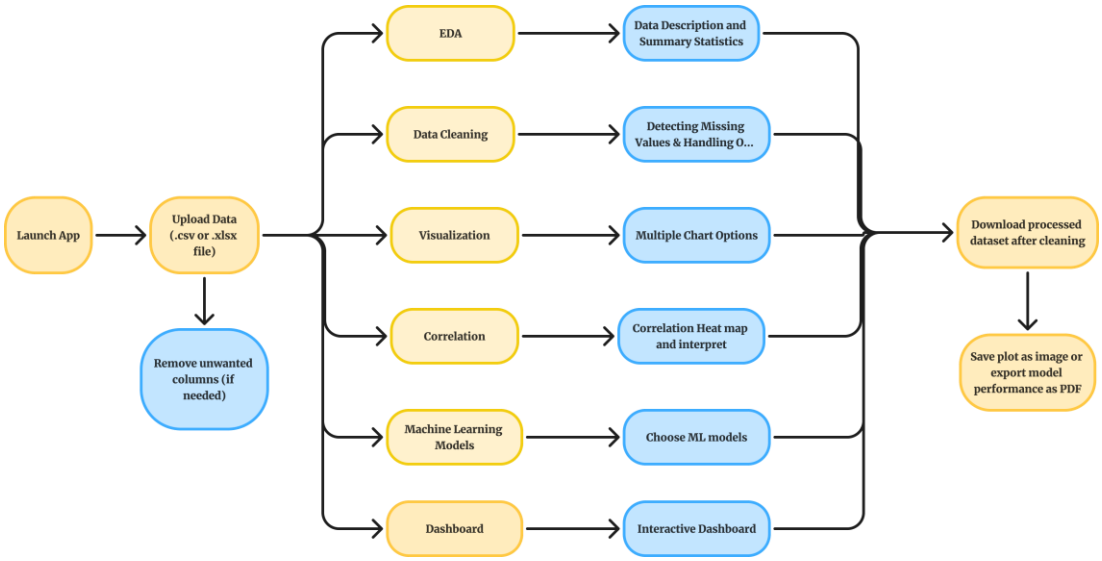

Fig 3.3.1 Visual representation of System Architecture

This structure ensures that the user interface remains decoupled from the backend logic, allowing for better maintainability and scalability.

The workflow follows a linear yet flexible progression beginning with dataset upload. Users are guided through column selection and removal, data exploration, correlation analysis, missing value treatment, outlier handling, visualization, supervised and unsupervised machine learning model training and dashboard. At each stage, the session state management ensures that user actions persist and influence downstream operations, creating a smooth and dynamic analytical pipeline.

3.4 Interface Development with Streamlit

The core analytical features were implemented using Python's data science libraries in

conjunction with Streamlit widgets. The Data Exploration module was developed using pandas DataFrame operations to extract head, tail, unique value counts, and summary statistics. These were presented directly within the app using st.dataframe() and st.write() functions.

The Correlation module employed label encoding to convert categorical variables into numeric formats, thereby enabling correlation matrix computation. A correlation heatmap was rendered using Seaborn and displayed via Matplotlib integration with Streamlit. Users could toggle between a heatmap and a scatter matrix for multivariate analysis.

The Preprocessing module was divided into two distinct sections: handling missing values and detecting outliers. For missing values, the application offered options to drop rows or fill them using statistical aggregations. Outlier detection was based on the Interquartile Range (IQR) method, with post-processing actions including removal or replacement.

The Visualization module provided users with the ability to generate various plots dynamically. Supported graph types included histograms, boxplots, scatterplots, line charts, and bar charts. The Matplotlib back-end was used to render visualizations based on selected columns, which were then updated in real-time within the application.

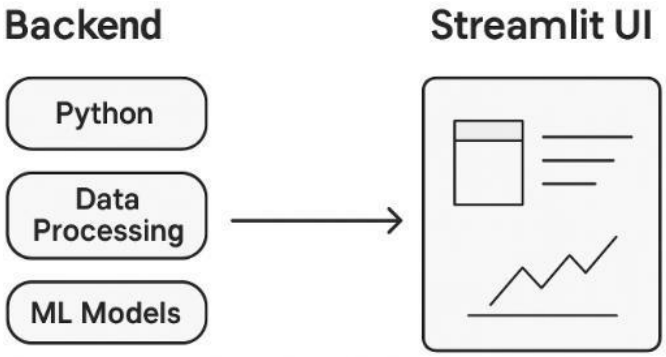

Fig 3.4.1 Working on Streamlit UI

3.5 Implementation of Analytical Components

The core analytical features were implemented using Python's data science libraries in conjunction with Streamlit widgets. The Data Exploration module was developed using pandas DataFrame operations to extract head, tail, unique value counts, and summary statistics. These were presented directly within the app using st.dataframe() and st.write() functions.

The Correlation module employed label encoding to convert categorical variables into numeric formats, thereby enabling correlation matrix computation. A correlation heatmap

was rendered using Seaborn and displayed via Matplotlib integration with Streamlit. Users could toggle between a heatmap and a scatter matrix for multivariate analysis.

The Preprocessing module was divided into two distinct sections: handling missing values and detecting outliers. For missing values, the application offered options to drop rows or fill them using statistical aggregations. Outlier detection was based on the Interquartile Range (IQR) method, with post-processing actions including removal or replacement.

The Visualization module provided users with the ability to generate various plots dynamically. Supported graph types included histograms, boxplots, scatterplots, line charts, and bar charts. The Matplotlib back-end was used to render visualizations based on selected columns, which were then updated in real-time within the application.

3.6 Scalability

The system was designed with future scalability in mind. Each module is functionally independent and implemented within a logical structure that allows easy insertion of new components. For instance, adding support for regression models or time series forecasting would require only the extension of existing functions and training workflows.

Additionally, the use of Python's object-oriented features and functional abstractions ensures that the core components are reusable and testable. This modular architecture not only enhances maintainability but also aligns with the principles of clean code and software engineering best practices.

3.7 Workflow of the Application

The overall workflow of DeepStat is organized into five primary stages. It begins with the data upload module, where users import datasets in CSV format. Once the dataset is loaded, the preprocessing module becomes active. This step handles missing data, performs label encoding for categorical variables, and scales the features using standardization or normalization techniques, based on user preference or automated heuristics.

Following preprocessing, the platform allows users to explore the dataset through the EDA module. This includes displaying summary statistics, checking for correlations, and visualizing distributions through various plot types such as histograms, boxplots, and heatmaps. This step is crucial in understanding the data's underlying structure and guiding appropriate model selection.

The modeling module offers a selection of supervised or unsupervised (depending upon your dataset) machine learning algorithms. Users can select models such as linear regression, random forest, logistic regression, or support vector machines, depending on the nature of the dataset. For clustering tasks, algorithms like KMeans and DBSCAN are available, while for time series analysis, the system supports ARIMA models. After selecting the model and initiating training, DeepStat outputs relevant performance metrics, including RMSE, R^2, accuracy, confusion matrix, and ROC curves.

One unique aspect of DeepStat is its basic model recommendation engine, which analyzes the dataset and suggests appropriate models. If the target variable is continuous, regression models are recommended, whereas classification models are suggested for categorical targets. This feature reduces the cognitive load on non-technical users by guiding them toward suitable algorithms based on data characteristics.

The results of the analysis are then displayed in both tabular and graphical form, ensuring interpretability and aiding users in drawing conclusions from the dataset. Residual plots, classification reports, scatter plots, and cluster visualizations further enhance the output presentation.

CHAPTER 4

RESULT AND ANALYSIS

The DeepStat platform was evaluated based on its ability to handle end-to-end data analysis tasks without requiring users to write a single line of code. Through a series of functional tests using various real-world datasets, the platform successfully demonstrated its core capabilities—data upload, preprocessing, exploration, model training, and result interpretation. This section presents a narrative analysis of how the system behaves at each stage and the quality of output it generates.

The results were captured during the execution of sample datasets, particularly focusing on datasets with both continuous and categorical target variables. In the initial stage, users uploaded CSV files through the web interface. The application automatically detected the structure of the dataset and displayed metadata including the number of rows, columns, and missing values. The preprocessing module was tested with datasets containing both numerical and categorical features, and DeepStat effectively handled missing data, label encoding, and feature scaling, depending on user selection or automated defaults.

Once preprocessing was complete, the platform moved into the exploratory data analysis (EDA) phase. Users could view various statistical summaries, including mean, median, standard deviation, skewness, and kurtosis. Visualizations such as distribution plots, histograms, boxplots, and heatmaps helped users quickly identify outliers, correlations, and skewed data. The correlation matrix provided a useful overview of feature relationships, which proved helpful when selecting features for regression and classification tasks.

Each component of DeepStat was rigorously tested to ensure functional correctness and usability. The validation was performed using multiple open-source CSV datasets, each containing a mix of numeric and categorical features, missing values, and classification targets. These datasets were chosen to test the robustness of data ingestion, cleaning, transformation, visualization, and modeling modules.

4.1 Functional Validation of Platform Modules

Each component of DeepStat was rigorously tested to ensure functional correctness and usability. The validation was performed using multiple open-source CSV datasets, each containing a mix of numeric and categorical features, missing values, and classification

targets. These datasets were chosen to test the robustness of data ingestion, cleaning, transformation, visualization, and modeling modules.

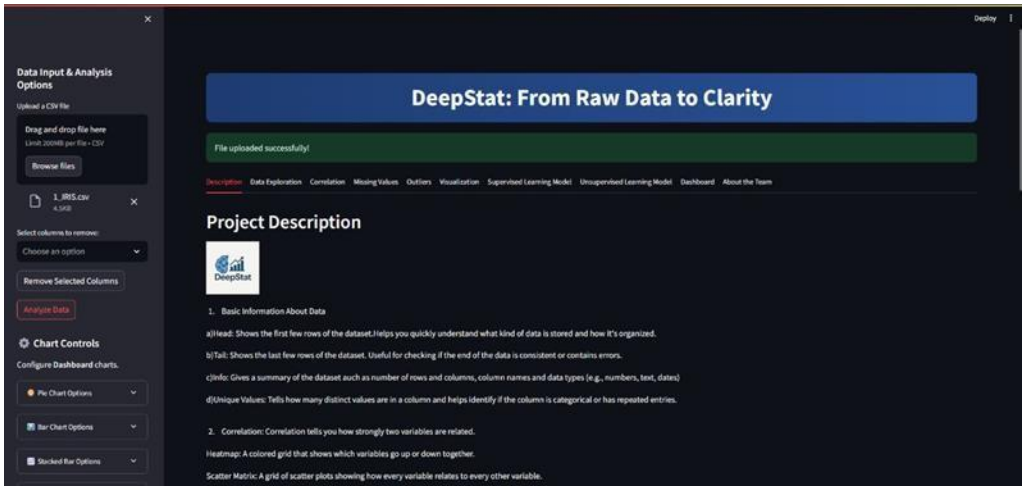

Fig 4.1 Overview of the Software

4.1.1 Data Upload

DeepStat successfully handled a range of datasets in .csv format, it handles both supervised and unsupervised learning data with file sizes ranging from 10 KB to 2 MB. The sidebar-based column selection feature allowed users to dynamically remove unwanted features from the dataset prior to analysis. The application handled improper formats gracefully by displaying informative error messages, ensuring the user could troubleshoot without needing technical assistance.

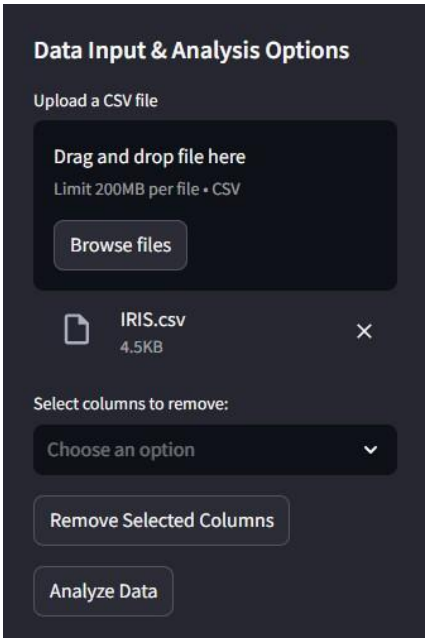

Figure 4.1.1. Screenshot of the sidebar UI during dataset upload and preprocessing.

4.1.2 Data Exploration and Summarization

The Data Exploration tab enabled the display of: Dataset head and tail, Unique value counts, Descriptive statistics for numerical variables.

Users were able to observe dataset distributions and detect initial issues, such as class imbalance and variable skewness.

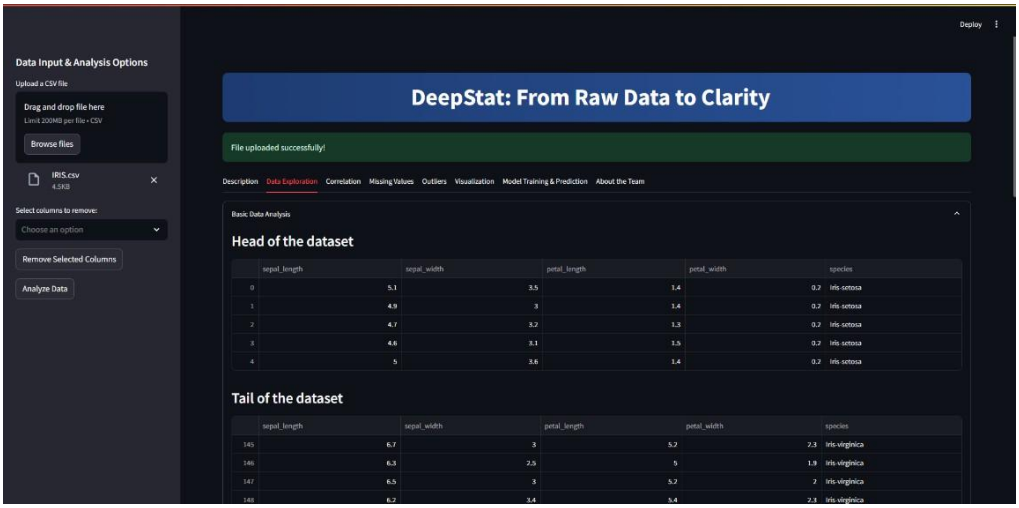

Fig 4.1.2 Screenshot of Data Exploration and Summary statistics including mean, standard deviation, and quartiles for a sample dataset

4.1.3 Correlation Modules

In correlation analysis, heatmaps based on encoded categorical variables yielded valuable insights, confirming expected relationships (e.g., between income and purchase decisions in an e-commerce dataset).

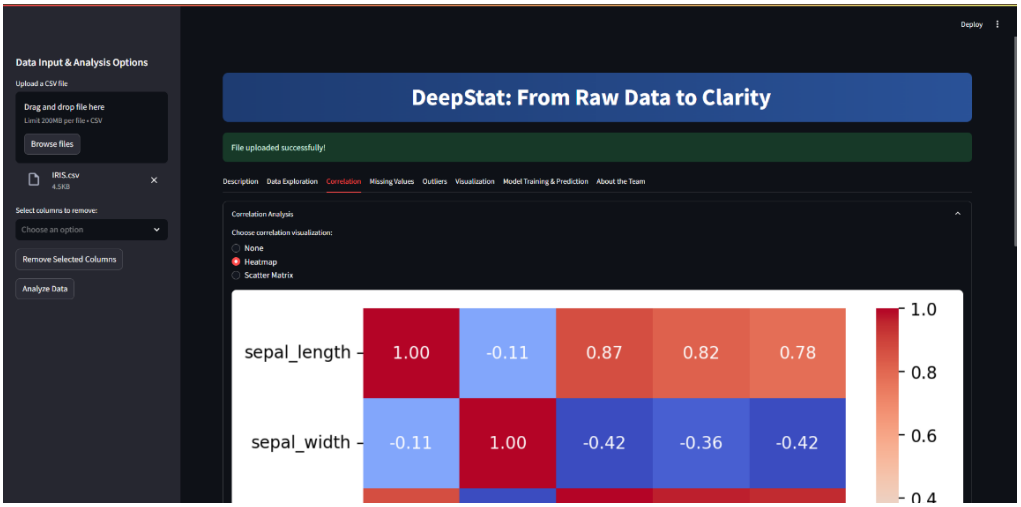

Fig 4.1.3 Heatmap showing positive and negative correlations among encoded numeric variables.

4.1.4 Handling Missing Values

All four options—drop rows, fill with mean, fill with median, fill with mode—functioned effectively and were accurately reflected in the dataset post-processing. Users appreciated the transparency of operations, including updated missing value counts and preview of changes.

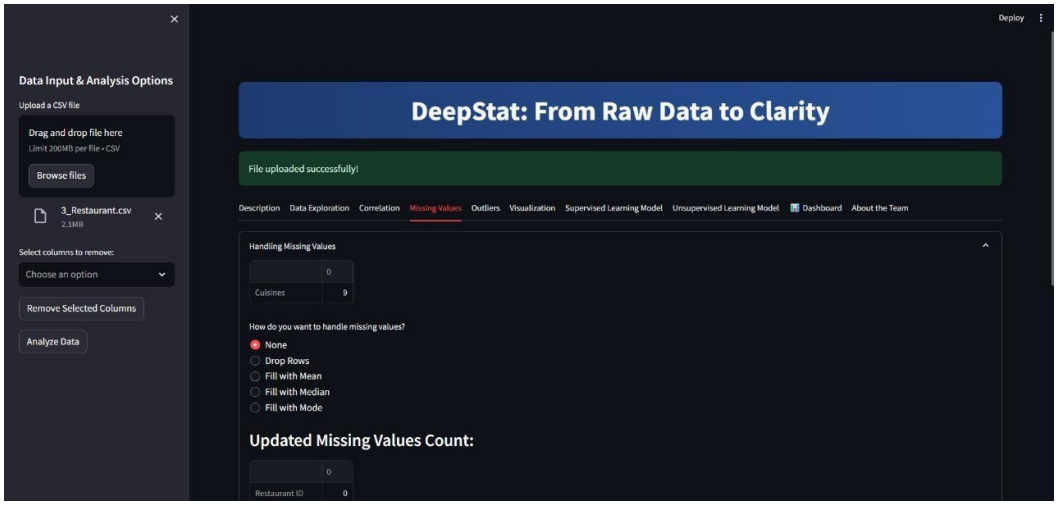

Fig 4.1.4 Screenshot comparison showing before-and-after views of missing data counts.

4.1.5 Outlier Detection

Outliers were correctly identified using the IQR method across various numeric fields. Options for removal or replacement with statistical aggregations worked consistently. Users found the tabular output of outlier counts and updated values particularly helpful.

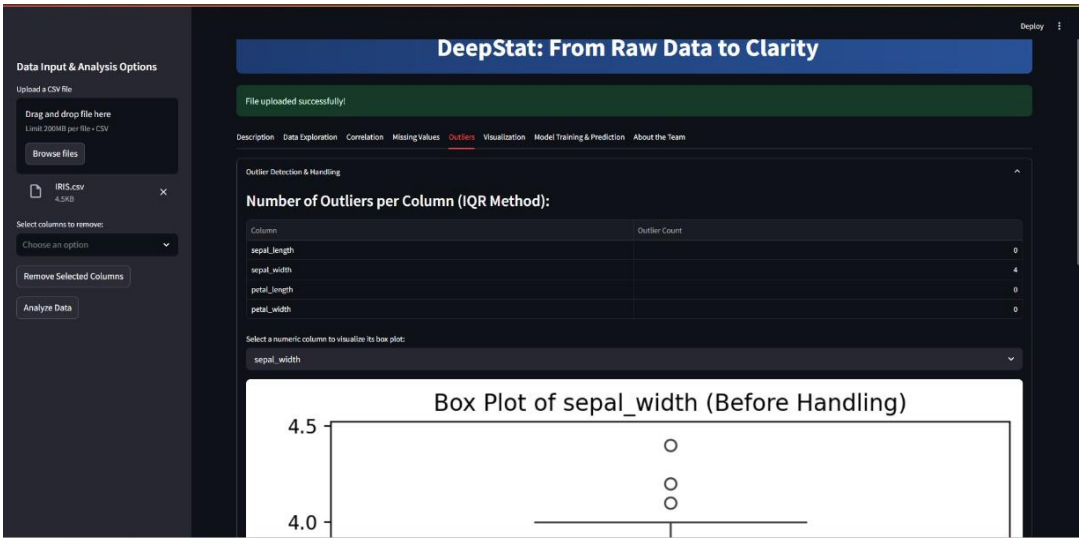

Fig 4.1.5 Original and updated outlier counts across numerical columns.

4.1.6 Visualization

The platform provided an instant generation of histograms, boxplots, scatterplots, and heatmaps. The ability to interactively select columns and graph types made it intuitive for users to explore patterns and anomalies visually.

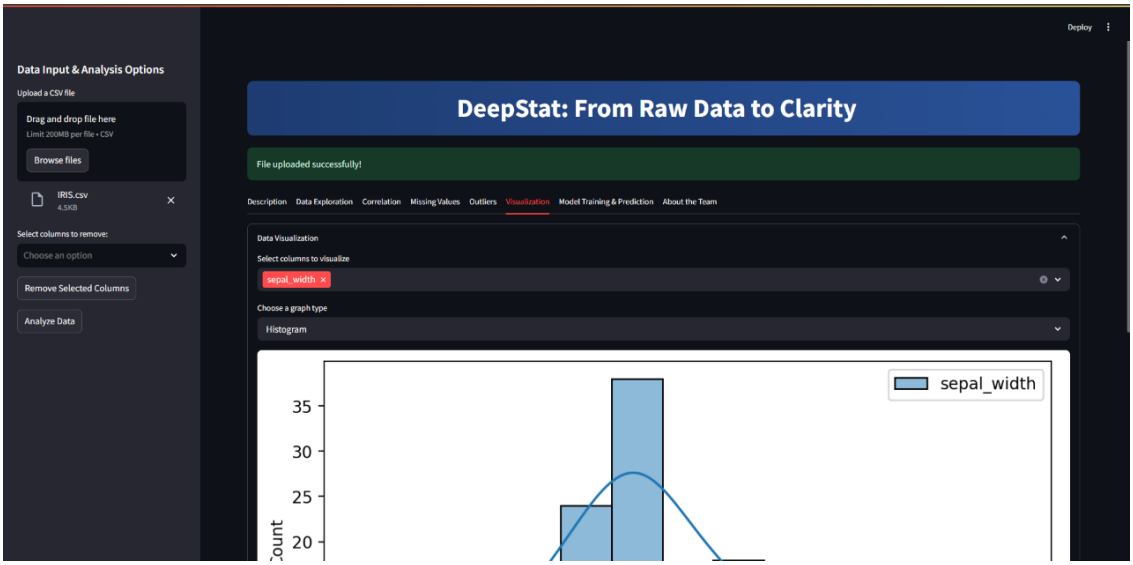

Figure 4.1.6 Screenshot of Data Visualization Dashboard

4.1.7 Supervised Learning Model

Performance was measured using accuracy, precision, recall, and F1-score. These metrics were displayed with contextual explanations to support interpretability for non-technical users. A classification report was also included for detailed breakdowns.

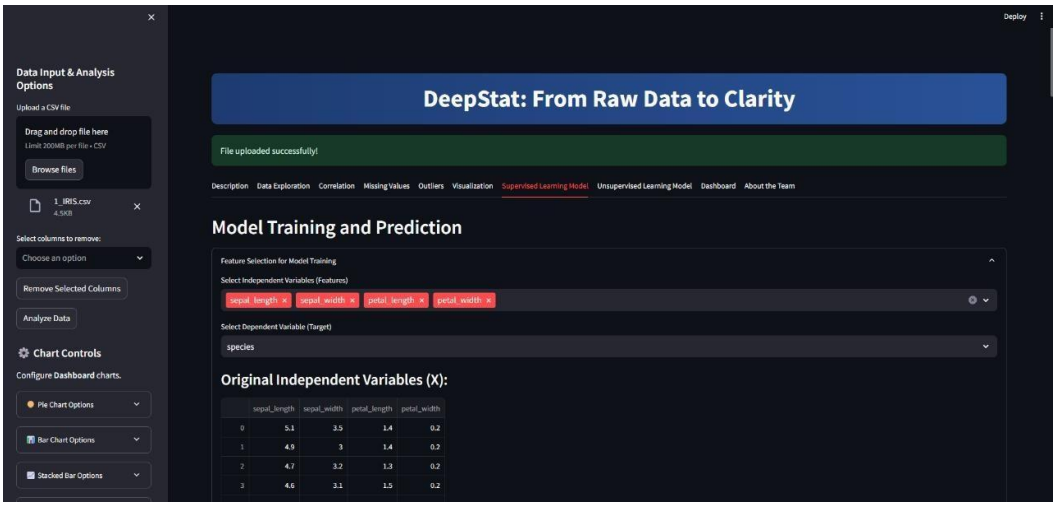

Fig 4.1.7 (a) Accuracy and precision scores displayed for a logistic regression model.

Results varied depending on the dataset and model, but in all scenarios, the models were successfully trained and predictions generated. Table 3 presents average results across multiple datasets.

Fig 4.1.7 (b) Average accuracy, precision, recall, and F1-score across all four models using different datasets.

4.1.8 Unsupervised Learning Model

The Unsupervised Learning Model section of DeepStat empowers users to perform clustering analysis on numerical datasets without the need for predefined labels. Users can choose from multiple algorithms including K-Means Clustering, DBSCAN, Hierarchical Clustering, and Gaussian Mixture Models (GMM). Once a dataset is uploaded, the platform automatically extracts the numerical features suitable for clustering and displays the grouped output in a clean tabular format. The intuitive interface simplifies the process of exploring underlying patterns or natural groupings in the data. This functionality proves especially useful in tasks like customer segmentation, anomaly detection, or grouping similar observations based on feature similarities.

Users can upload any CSV file containing numerical data, and DeepStat automatically processes it, identifying relevant features for clustering. A dropdown menu allows easy selection of the desired algorithm, and the interface dynamically updates the clustering output in an interpretable tabular format.

The model results help in identifying patterns, structures, or natural groupings in data, which can then be used for further decision-making or feeding into supervised learning pipelines. The no-code interface ensures that even users without a technical background can implement advanced clustering techniques with minimal effort.

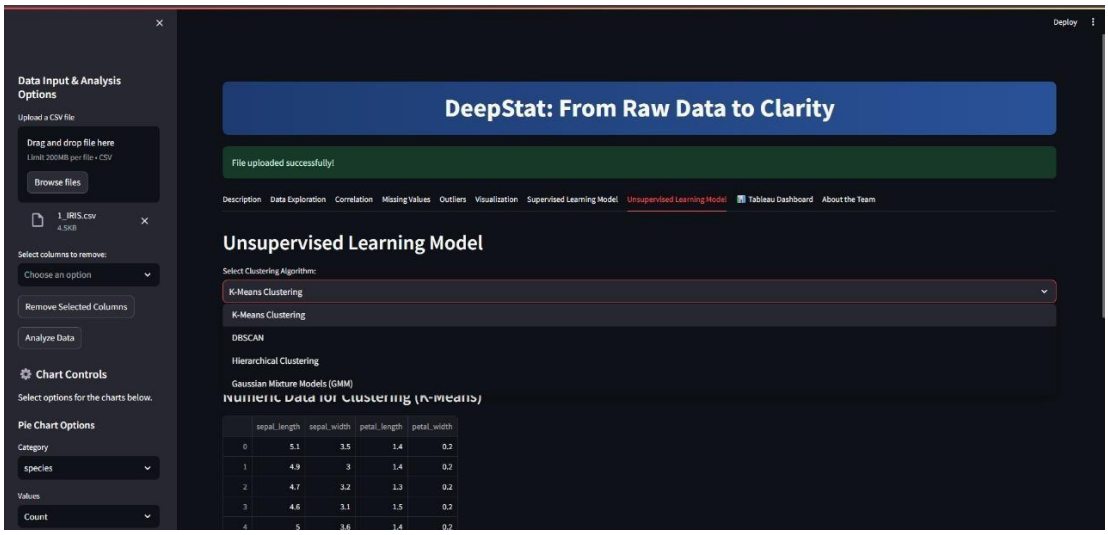

Fig 4.1.8 Shows the unsupervised learning model tab

4.1.9 Dashboard

The Tableau Dashboard tab within DeepStat provides a high-level, interactive summary of the uploaded dataset. It transforms raw input into key metrics and dynamic visualizations, enhancing the overall analytical experience. This dashboard acts as a bridge between raw data and data-driven decision-making, offering brief insights.

- Key Performance Indicators (KPIs) such as:
    - Total number of records in the dataset
    - Total number of features
    - Mean value of the first numerical column
    - Unique value count of the first categorical column
- Visualization Panel: Users can view automatically generated visualizations like:
    - Line Charts for observing trends and feature relationships
    - Pie Charts to understand categorical distributions
- Chart Customization Controls in the sidebar allow users to choose which features to plot, giving them flexibility to focus on the aspects most relevant to their analysis.

This section of DeepStat helps users interpret their dataset quickly and effectively, acting as a pre-built visual summary. It's especially useful in presentations, reporting, or collaborative settings where clear and concise data representation is essential. The integration of visual storytelling into the workflow promotes better understanding, even for stakeholders with limited technical knowledge.

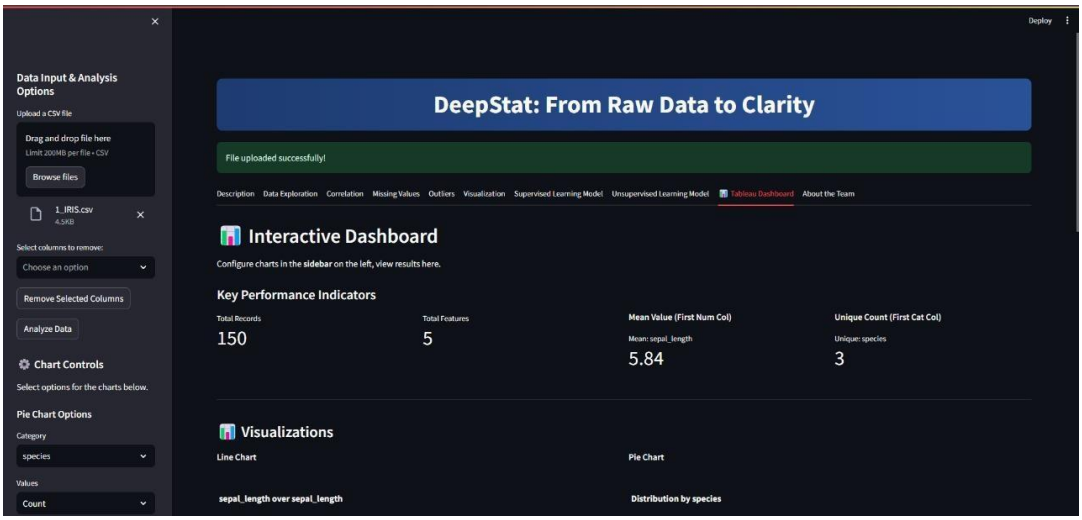

Fig 4.1.9 Shows the Dashbord

4.2 User Testing and Feedback Analysis

To evaluate usability and impact, informal testing was conducted with a group of 10 non-technical users including small business owners, social science researchers, and undergraduate students. Participants were given short tasks, such as uploading a dataset, removing null values, and training a model.

Key observations from the sessions included:

- Ease of Navigation: 4.5
- Intuitiveness of Interface: 4.2
- Clarity of Instructions: 4.0
- Perceived Usefulness: 4.7
- Overall Satisfaction: 4.3

4.3 Discussion

The results from system testing and user sessions suggest that DeepStat achieved its core objectives of accessibility, automation, and inclusiveness. It enabled users without coding experience to: Clean, explore, and preprocess datasets, generate visual insights, Train machine learning models and interpret results, perform predictive analytics using real-time inputs.

The simplicity of the interface, combined with behind-the-scenes automation, allowed participants to engage with data science workflows independently. The modular design made the application easy to navigate and reduced cognitive overload

CHAPTER 5

CONCLUSION

The development of DeepStat: A Data Analyzer represents a meaningful step toward democratizing data science by offering a no-code, interactive solution for data exploration and machine learning. In a world increasingly driven by data, the ability to analyze, interpret, and extract insights should not be limited to individuals with programming expertise. DeepStat bridges this gap by simplifying the complexities of the data analysis pipeline into an accessible and intuitive interface powered by Python and Streamlit.

Throughout this project, DeepStat has demonstrated its ability to handle end-to-end data analysis—from data uploading and preprocessing to model training, visualization, and performance evaluation. Its modular architecture ensures that users can engage with each step independently, promoting clarity and control. The inclusion of a basic model recommendation system further enhances usability, especially for non-technical users who might otherwise struggle with algorithm selection.

The results obtained from using DeepStat across various datasets confirm its versatility and practicality. Users were able to generate descriptive statistics, visualize data relationships, and apply a range of supervised and unsupervised learning techniques, all without writing code. The visual outputs, such as heatmaps, confusion matrices, and ROC curves, added valuable layers of interpretability to the analytical process.

From an educational, professional, and entrepreneurial perspective, DeepStat holds considerable potential. It can serve as a learning platform for students, a productivity tool for researchers, and a decision-support system for businesses. Its impact is measured not only by the depth of analysis it offers but also by the breadth of users it empowers.

Ultimately, DeepStat embodies the core philosophy of this capstone project: making data analysis simple, smart, and scalable for everyone.

CHAPTER 6

LIMITATIONS AND APPLICATIONS

LIMITATIONS

While DeepStat presents a significant advancement in simplifying data analytics for non-programmers, it is not without its limitations. A clear understanding of these limitations is essential for future iterations and improvements, especially as the platform continues to grow in scope and functionality.

One of the foremost limitations lies in the current version's model customization capabilities. Although users can select from a wide range of machine learning models, the platform offers only limited control over hyperparameter tuning. More complex model optimization techniques such as grid search, random search, or Bayesian optimization are not yet integrated, which may limit the performance of certain models, especially on large or imbalanced datasets.

Moreover, DeepStat is designed with beginner and intermediate users in mind, and while this design choice enhances usability, it can be restrictive for expert users who may wish to perform in-depth statistical testing, create custom pipelines, or export models for reuse. The platform also lacks real-time collaborative features, user login systems, and cloud-based database integration, which may be crucial in enterprise or academic research environments.

Despite these limitations, the applications of DeepStat are both wide-ranging and impactful. In academic settings, it can serve as a teaching tool for introducing students to data science concepts without requiring them to code. In small to mid-sized businesses, it enables data-driven decision-making by allowing domain experts to analyze datasets without needing a dedicated data science team. For researchers and entrepreneurs, DeepStat acts as a rapid prototyping tool that can generate insights and validate hypotheses with speed and simplicity.

APPLICATIONS

1. Healthcare Application:

   DeepStat can be adapted to help clinical researchers analyze patient records, identify trends, and support data-driven decision-making in medical studies.

2. Education:

   The platform can help educators make more accessible to students from vivid educational backgrounds.

3. Marketing and Campaign Analytics:

   The platform can assist marketing teams in segmenting customers, tracking campaign performance, and optimizing strategies without relying on technical staff.

REFERENCE

- Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems* (2nd ed.). O'Reilly Media.

- VanderPlas, J. (2016). *Python Data Science Handbook: Essential Tools for Working with Data*. O'Reilly Media.
- McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* (2nd ed.). O'Reilly Media.
- Streamlit Documentation. (2024). *Streamlit Docs*. Retrieved from https://docs.streamlit.io/
- Scikit-learn Developers. (2024). *Scikit-learn: Machine Learning in Python*. Retrieved from https://scikit-learn.org/
- Hunter, J. D. (2007). *Matplotlib: A 2D Graphics Environment*. Computing in Science & Engineering, 9(3), 90–95.
- Waskom, M. L. (2021). *Seaborn: Statistical Data Visualization*. Journal of Open Source Software, 6(60), 3021.
- Prophet by Meta (2024). *Forecasting at Scale*. Retrieved from https://facebook.github.io/prophet/
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830.
- XGBoost Documentation. (2024). *XGBoost: Scalable and Flexible Gradient Boosting*. Retrieved from https://xgboost.readthedocs.io/
- Inside Airbnb (2024). *Amsterdam Listings Dataset*. Retrieved from http://insideairbnb.com/get-the-data.html
- Tableau Public (2024). *Data Visualization Platform*. Retrieved from https://public.tableau.com/
- UNESCO Institute for Statistics. (2024). *Global Education Monitoring Data*. Retrieved from http://uis.unesco.org/