iNeuron

Architecture

Mushroom Classification Application

By

ANNAMALAI SUBRAMANI

Revision Number: 1.0
Last date of revision: 29/01/2024

Document Version Control

Date Issued	Version	Description Author
28/01/2024	2	Initial HLD - V1.0 Annamalai

Contents

Abstract

This document presents the architectural framework for the Mushroom Classification machine learning project, offering a detailed guide to the underlying structure and implementation intricacies of the system. The architectural overview encompasses key design decisions and specifications critical for the successful development and deployment of the Mushroom Classification application.

Providing insights into the system architecture, the document elucidates the relationships and interactions among various components. It delves into the chosen classification algorithm, its architectural nuances, and the pivotal input features. Moreover, the document furnishes a comprehensive understanding of model training methodologies and validation techniques, ensuring the predictive accuracy of the system.

API design considerations form a vital aspect of this architectural documentation, covering endpoint definitions, input validation procedures, error handling mechanisms, and robust security measures, including token authentication. Deployment strategies, along with environment specifications and containerization details, are meticulously outlined to facilitate a seamless and efficient implementation process.

The document also addresses testing methodologies, encompassing both unit and integration levels, to ensure the reliability of both the machine learning model and API functionalities. Logging mechanisms and monitoring metrics are defined to enable effective debugging and continuous assessment of the system's health.

A strong emphasis is placed on documentation planning, covering code documentation with a focus on docstrings and comments. Additionally, a comprehensive user guide is provided to assist both developers and end-users in navigating the system, fostering a smooth and user-friendly interaction.

Security measures are integrated to address data privacy concerns, outlining strategies to safeguard sensitive data. Model explainability techniques are incorporated to enhance transparency in predictions, providing stakeholders with a clear understanding of the Mushroom Classification system.

This abstract serves as a succinct reference, encapsulating the essential features of the architectural design document for stakeholders involved in the evolution of the Mushroom Classification system.

1 Introduction

1.1 Purpose and Significance of Architecture Document?

This architecture document serves the purpose of offering a comprehensive overview of the design and structure integral to the Mushroom Classification system. Its primary aim is to articulate the significance of the system's architecture, providing clear guidance to both the development team and stakeholders. By delving into key architectural decisions, the document ensures a shared understanding of the system's blueprint, fostering effective collaboration and informed decision-making throughout the development process.

1.2 Scope

The scope of this architecture document encompasses a thorough exploration of the fundamental design elements, inter-component relationships, and the overall structure defining the Mushroom Classification system. Tailored for a diverse audience including stakeholders and developers, it furnishes a detailed insight into the architectural decisions shaping the system. Proposed for approval by higher management, this document is designed to be a guiding resource across the development lifecycle, facilitating a cohesive and well-informed approach to system architecture.

2 Technical specifications

2.1 Dataset

The dataset for Mushroom Classification is a comprehensive collection of features related to mushroom characteristics. It encompasses details such as cap shape, cap color, gill color, odor, etc. The data is sourced from [provide dataset link] and is structured to facilitate the development of predictive models for classifying mushrooms into edible or poisonous categories.

- Data Source: [Provide dataset link]
- Dataset Size:
    - Original dataset: 8314 rows, 23 columns

This dataset structure mirrors the requirements for machine learning models and is conducive to training accurate classifiers.

2.2 Predicting Mushroom Edibility

The system for predicting mushroom edibility utilizes a user-friendly interface for input and prediction. The following steps outline the process:

- User Interface:
    - The system presents a user-friendly form with fields representing various mushroom attributes (e.g., cap shape, cap color, gill color).
    - Users fill in the required information for the mushroom in question.
- Prediction Process:
    - Upon confirmation, the system processes the input data through the selected machine learning model.
    - The prediction output is displayed, indicating whether the model predicts the mushroom as edible or poisonous.

This prediction system is designed to be intuitive and accessible, allowing users to quickly assess the edibility of a mushroom based on its features.

Model Training and Evaluation

- Machine Learning Algorithm: [Specify the chosen algorithm, e.g., Decision Trees, Random Forest]
- Training Set:
    - Total records: 8314
- Testing Set:
    - Total records: [Number of records in the testing set]

The chosen machine learning model is trained on the provided dataset, and its performance is evaluated using appropriate metrics such as accuracy, precision, recall, and F1-score.

2.3 Logging

1. User Activity Logging:
Ensure comprehensive logging of all user activities within the system.

2. Dynamic Logging Identification:
The system intelligently determines the appropriate steps in the workflow where logging is necessary.

3. System Flow Logging:
Enable logging for every step and aspect of the system's flow to capture detailed information.

4. Flexible Logging Methods:
Developers have the option to employ various logging methods, including database logging or file logging.

5. Optimized System Performance:The system should maintain optimal performance even when employing extensive logging methods, such as database or file logging.

6. Logging for Debugging:
Emphasize the importance of logging for debugging purposes; it is a mandatory practice to facilitate issue identification and resolution.

2.4 Database

1. Data Storage:

The system must store every mushroom classification request in the database, ensuring a format that facilitates easy model retraining.

2.User Interaction:

Users interact with the system by selecting the mushroom classification option and providing the necessary information.

3.Comprehensive Data Storage:

The system systematically stores all user-provided data or received information related to mushroom classification in the chosen database.

- The dataset obtained from Kaggle serves as a valuable resource for model training and testing, with 8,314 rows and 23 columns.

2.5 Deployment

Deployment, in the context of software development, refers to the process of making a software application available and operational in a specific environment. It involves transferring a developed and tested software system from a staging or development environment to a production environment, where it becomes accessible to end-users. In the case of this particular project, AWS (Amazon Web Services) was utilized for the deployment, leveraging its cloud infrastructure and services to host, scale, and manage the application in the production environment.

3 Technology stack

The technology stack for the mushroom classification system is carefully chosen to ensure a seamless and user-centric experience. It leverages a dynamic combination of front-end and back-end technologies, along with a reliable database and cloud deployment for scalability.

Front End	HTML/CSS
Backend	Python
Deployment	AWS

4 Model training/ validation workflow

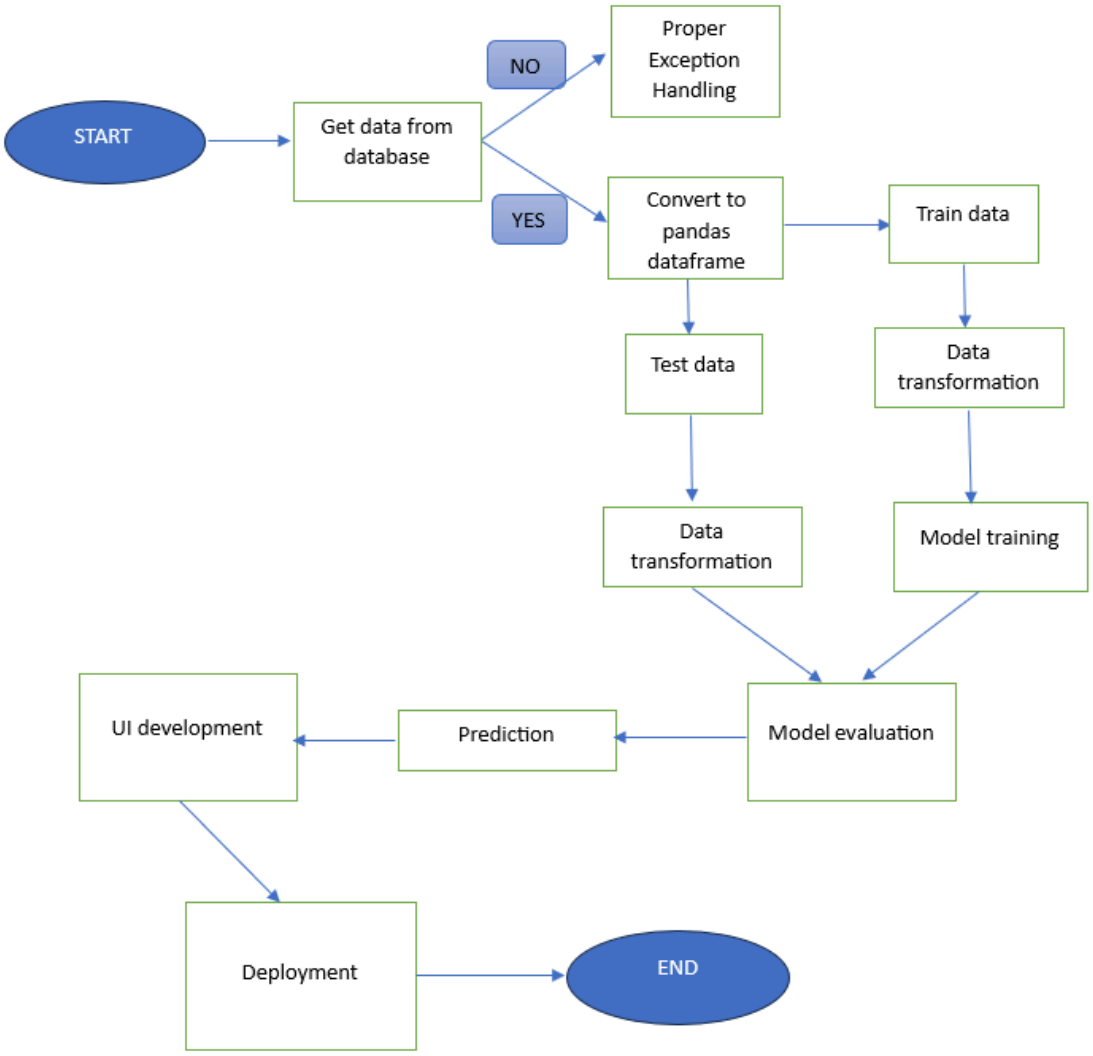
The model training and validation workflow for mushroom classification involves several critical steps to ensure the effectiveness of the system in distinguishing between edible and poisonous mushrooms. The process is designed to handle dataset preparation, feature selection, model training, and validation. Here is an overview of the key steps:

1.Model Training:

A classification model, implemented using Python and popular machine learning libraries like scikit-learn, is trained on the labeled training data. The model learns patterns and relationships within the dataset to distinguish between edible and poisonous mushrooms.

2.Performance Assessment:

The trained model's performance is assessed using the test dataset. Metrics such as accuracy, precision, recall, and F1 score are employed to evaluate how well the model generalizes to new, unseen mushroom data.



5 User I/O workflow

The User I/O workflow for the mushroom classification system is designed to provide an intuitive and user-friendly experience. The system incorporates HTML and CSS for the front-end, ensuring an engaging interface. Here's a detailed overview of the workflow:

1.User Interface:

The system features an HTML and CSS-based user interface that allows users to interact seamlessly. The design prioritizes clarity and ease of use.

2.User Input:

Users initiate the classification process by selecting the mushroom classification option. They provide relevant information about the mushroom attributes, such as cap shape, cap color, gill size, and habitat.

3.Exception Handling:

The system incorporates robust exception handling to address potential errors or incomplete information in the user input. Users are guided to rectify any issues for successful processing.

4.Data Validation:

The provided user input undergoes validation to ensure that it aligns with the expected format and required attributes for mushroom classification.

5.Model Prediction:

The validated user input is then processed through the pre-trained supervised learning classification model, implemented using Python and machine learning libraries. The model predicts whether the mushroom is edible or poisonous based on the input attributes.

6.Results Presentation:

The prediction results, along with any additional insights or information, are presented to users through a clear and visually appealing interface. This enhances the overall user experience and facilitates a better understanding of the classification outcome.

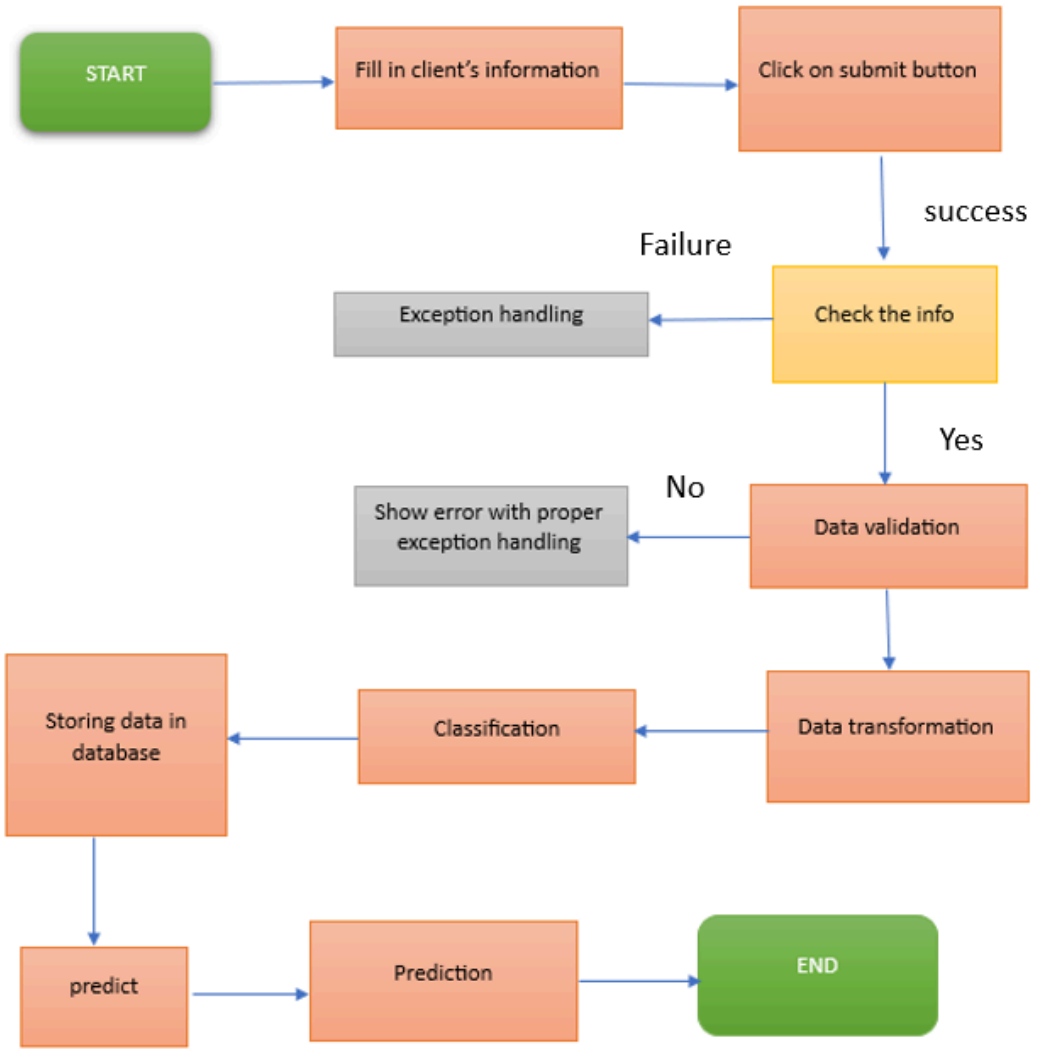
7.User Interaction Feedback:

The system provides feedback to users regarding the classification outcome, ensuring transparency and building user confidence in the reliability of the predictions.

8.User-Friendly Interface:

Throughout the workflow, the emphasis is on maintaining a user-friendly interface, making it easy for users to navigate, input data, and interpret the results.

.



6 Conclusion

In conclusion, the Architecture Document lays out a detailed plan for the Mushroom Classification System, encompassing system structure, algorithmic choices, API considerations, deployment strategies, testing approaches, and security measures. This guide serves as a crucial reference, ensuring clarity and reliability in mushroom classification predictions. With a focus on user-centric design and meticulous testing, the document provides a solid foundation for developers and stakeholders. The commitment to algorithm transparency and secure design contributes to a trustworthy and effective mushroom classification system. As a guiding resource, this

document facilitates an informed and cohesive approach to the ongoing development and refinement of the Mushroom Classification System.