LOW LEVEL DESIGN DOCUMENT

MUSHROOM CLASSIFICATION

AAKASH BHUTE

CONTENTS

Abstract

This study focuses on the classification of mushrooms into two categories, namely Poisonous and Edible, using a machine learning model. It aims to determine the significant features that play a crucial role in predicting the edibility or toxicity of mushrooms. Mushrooms have been consumed since ancient times and are highly regarded for their nutritional value. They are low in calories, carbohydrates, fats, and sodium, while being cholesterol-free. Mushrooms offer essential nutrients such as selenium, potassium, riboflavin, niacin, Vitamin D, proteins, and fiber. They have a rich history as a food source and are also recognized for their healing properties in traditional medicine. Various health benefits and potential disease treatments have been associated with mushrooms, including their anticancer and antitumor properties. Moreover, mushrooms exhibit antibacterial effects, enhance the immune system, and assist in lowering cholesterol levels. Furthermore, mushrooms are a valuable source of bioactive compounds. Throughout this machine learning analysis, we will identify the key features that determine whether a mushroom is poisonous or edible.

1. Introduction

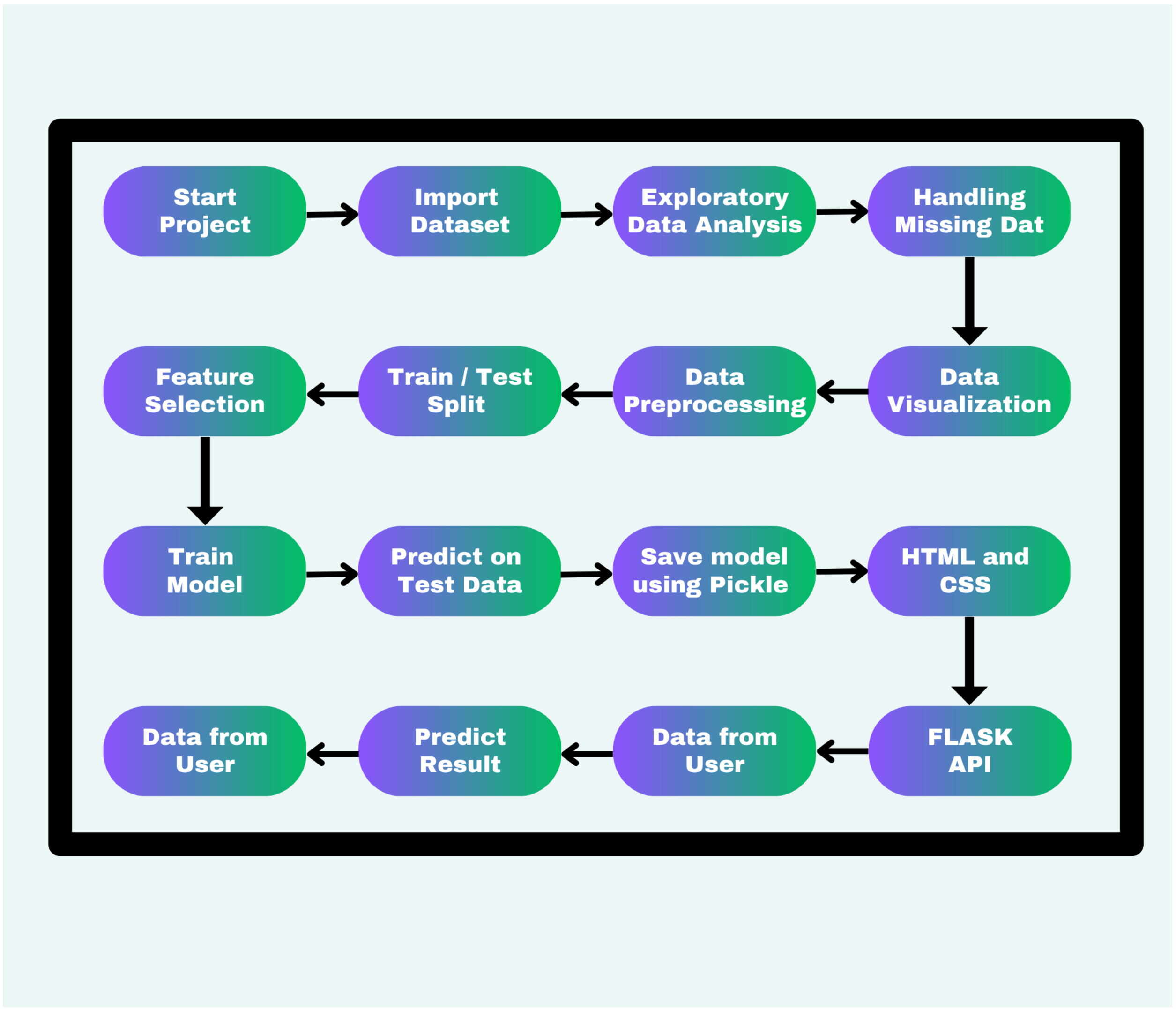
1.1 What is a Low-Level Design Document?

The objective of a Low-Level Design Document (LDD) is to provide an internal logical design of the program code specifically for the Food Recommendation System. The LDD outlines detailed class diagrams that illustrate the methods and relationships between classes, as well as program specifications. It further explains the various modules involved, enabling programmers to directly translate the document into program code.

1.2 Scope

Low-level design (LLD) is a systematic approach to designing software components, involving a step-by-step refinement process. It encompasses the design of data structures, software architecture, source code, and performance algorithms. The process begins with defining the data organization during the requirement analysis phase and subsequently refining it during the data design work.

2. Architecture



3. Architecture Description

This project is designed to make an interface for the user to predict whether a mushroom is poisonous or not.

3.1 Data Collection

The data utilized for this project is sourced from the Kaggle Dataset, and the corresponding URL for accessing the dataset is provided below:

Dataset: https://www.kaggle.com/datasets/uciml/mushroom-classification.

3.2 Data Description

The dataset comprises descriptions of fictional samples representing 23 species of mushrooms with gills. These species are derived from the Agaricus and Lepiota families of mushrooms, as documented in The Audubon Society Field Guide to North American Mushrooms (1981). Each species is categorized as either definitely edible, definitely poisonous, or of unknown edibility and not recommended. The latter category, which includes mushrooms of unknown edibility, has been merged with the poisonous category.

3.3 Exploratory Data Analysis

The dataset is provided in a CSV file format. It consists of 8,124 rows and 23 columns. All the columns in the dataset are categorical variables. The target column contains two classes: 'p' for poisonous and 'e' for edible. It is worth noting that the dataset exhibits a roughly equal distribution of counts between the poisonous and edible classes, indicating a balanced dataset.

3.4 Handling Missing Data

At first, we observed that there were no missing/null values in the dataset. However, if you go through the data description (check the link) you will find that the missing values in one column are replaced with "?". There are 2480 missing values in 'stalk-root' column. First, we will replace these values with np.nan so that we can handle missing data. We will impute the missing values in 'stalk-root' column using sklearn SimpleImputer with strategy='most_frequent'.

3.5 Data Visualization

Initially, no missing or null values were apparent in the dataset. However, upon reviewing the data description (please refer to the provided link), it was discovered that missing values in one column are denoted by "?". Specifically, the 'stalk-root' column contains 2480 missing values. To address this, the missing values will be replaced with np.nan, allowing for proper handling of missing data. The subsequent step involves imputing the missing values in the 'stalk-root' column using the SimpleImputer module from the sklearn library, utilizing the 'most_frequent' strategy.

3.6 Data Preprocessing

To begin, we removed the 'veil-type' column from the dataset because it had the same value across all the data points, making it uninformative for determining the mushroom's class. Afterwards, we transformed our target column by mapping 'poisonous' to 0 and 'edible' to 1. To convert the categorical values into numerical ones, we employed a Label Encoder. Finally, we scaled the data to ensure that all features were on the same scale.

3.7 Feature Selection

Once the data was divided into training and testing sets, we employed the SelectKBest method with a score_func of chi2. This allowed us to determine the most significant features related to the target column. From the initial 21 columns, we identified that 12 columns were crucial for training our model.

3.8 Model Training & Evaluation

For model training, we utilized the XGBClassifier. This model demonstrated remarkable speed in comparison to other models. Additionally, it achieved outstanding accuracy of 100% on both the training and testing data sets, which is highly favorable for the goals of our project.

3.9 Model Deployment

We developed a webpage using HTML and CSS, and subsequently built a Flask web application. Initially, we conducted testing on our local machine to ensure its functionality. Later, we deployed our model. During testing, we provided various input combinations and observed that the predicted outputs were accurate. We encountered no issues with the application as it operated smoothly and without any problems.