Low Level Design Document

Project Name: Wheat Kernel Classification

Name: Venkata Phani Srikar Pillalamarri

E-mail: pvpsrikar@gmail.com

Table of Contents

Abstract

Based on a variety of morphological traits, the **Wheat Kernel Classification program** is an advanced tool that helps users determine the class of a wheat kernel (Kama, Rosa, or Canadian). This web application, which was created using Flask, makes predictions in real time using a machine learning model that has been trained. This document offers a thorough low-level design of the program, describing all of the components, how they work together, and the underlying architecture.

User-provided data on characteristics of a wheat kernel, including its **area, perimeter, compactness, length of kernel, width of kernel, asymmetry coefficient,** and **length of kernel groove**, forms the basis of the application's core functionality. After processing, these inputs are transformed into a format that the predictive model can use.

To ensure precise predictions, the **XGBoost classifier** machine learning model makes use of a number of preprocessing procedures. A large dataset of wheat kernel traits was used to train the model, guaranteeing the robustness and accuracy of the predictions.

The technical details of every module, such as **model integration, data preparation,** and **user interface design**, are covered in detail in this document. It also addresses the application's **deployment**, ensuring the system's **scalability, maintainability,** and **user-friendliness**. The purpose of this document is to provide developers with the knowledge and best practices they need to comprehend, manage, and improve the **Wheat Kernel Classification** application.

Chapter -1

Introduction

For the thorough planning and implementation of a software project, a Low-Level Design (LLD) document is essential. It acts as a link between the real coding process and the high-level design, which offers a summary of the system architecture. The LLD document dives deeply into the system's intricacies, providing fine-grained information to guarantee that every component is well understood and capable of being applied correctly and consistently. The LLD document offers a thorough design for every part of the Wheat Kernel Classification application, making the development, testing, and maintenance procedures easier. It describes the application's precise functionality in detail, giving developers a clear path to follow, eliminating confusion and encouraging a well-thought-out implementation plan.

The scope of this LLD document encompasses all technical aspects of the Wheat Kernel Classification application. This includes:

- Detailed descriptions of the user interface components and their interactions.
- Specifics on data preprocessing techniques and transformations.
- Integration of the machine learning model, specifically a XGBoost.
- Implementation details of the predictive algorithms.
- Handling of user inputs and how they are processed for classification.
- Deployment considerations, ensuring the application is scalable and maintainable.
- Error handling and logging mechanisms.
- Security measures to protect user data and application integrity.

This document aims to provide a comprehensive guide that will enable developers to build, test, and deploy the Wheat Kernel Classification application with a high degree of confidence and consistency.

During the development and implementation of the Wheat Kernel Classification application, several challenges have been identified:

- **Processing Capabilities**: The machine learning models employed by the application may require significant processing power for both inference and training. Ensuring optimal performance on limited hardware is essential.
- **Data Privacy**: User inputs must be handled securely to protect privacy and comply with data protection regulations.
- **Model Accuracy**: Ensuring high accuracy, especially in distinguishing between Kama, Rosa and Canadian kernels, is critical. High recall and precision are necessary to avoid misclassification.
- **User Interface**: The UI must be responsive and intuitive, catering to users with varying levels of technical expertise.
- **Deployment Contexts**: The application should be deployable in diverse environments to ensure consistent performance and compatibility across multiple platforms.

Several risks are associated with the development and implementation of the Wheat Kernel Classification application:

- **Model Misclassification**: Incorrect classification of wheat kernels could affect the accuracy of the system and lead to poor decisions in scenarios where this application is used in practical settings. Comprehensive testing and validation of the model are imperative to mitigate this risk.
- **Potential Security Threats**: Like any web application, there is a potential for security breaches. Robust security measures must be implemented to protect user data and maintain application integrity.
- **Performance Issues:** The application may encounter performance bottlenecks, particularly when dealing with a high number of simultaneous users or complex input processing. Optimizing the application for performance is essential.
- **Data Handling Errors:** Improper handling or preprocessing of user inputs could result in inaccurate predictions. Ensuring correct data processing at each step is vital.
- **Deployment Challenges:** Problems may surface during the deployment phase, such as compatibility issues and server configuration errors. A comprehensive deployment plan and rigorous testing can help mitigate these risks.

Chapter - 2

Technical Specifications

2.1 Dataset:

The dataset used in the Wheat Kernel Classification application was provided by the UCI Machine Learning Repository. This dataset contains various traits that describe the physical properties of wheat kernels, which are crucial for distinguishing between different wheat varieties: Kama, Rosa, and Canadian. The following characteristics are included in the dataset:

- Area: Represents the area of the wheat kernel in square units, which helps define its size.
- Perimeter: The total distance around the boundary of the wheat kernel.
- Compactness: Calculated as $\frac{perimeter^2}{area}$, this attribute indicates how compact the kernel is.
- Length of Kernel: Measures the length of the wheat kernel.
- Width of Kernel: The width of the wheat kernel, typically perpendicular to its length.
- Asymmetry Coefficient: This attribute measures the asymmetry or irregularity of the wheat kernel's shape.
- Length of Kernel Groove: Represents the length of the kernel groove, an important feature used to distinguish between different wheat varieties.

Each of these attributes contributes to the effective classification of the wheat kernels into one of the three types: Kama**,** Rosa**,** and Canadian. The machine learning model leverages these features to accurately predict the wheat kernel class based on their physical traits.

2.2 Technology Stack:

The technology stack for the Wheat Kernel Classification application includes a range of tools and frameworks to facilitate development, machine learning, and deployment:

- Python: The primary programming language used for data processing, model training, and application logic.
- Pandas: Utilised for data manipulation and preprocessing tasks.
- Scikit-Learn: Employed for building and evaluating machine learning models.
- NumPy: Essential for numerical operations and data handling.
- Matplotlib/Seaborn: Employed for data visualisation to gain insights during the exploratory data analysis phase.
- Jupyter Notebook: Used for interactive development and experimentation with data and
- Models.

2.3 Predicting Class:

The core functionality of the Wheat Kernel Classification application revolves around predicting whether a kernel is Kama, Rosa or Canadian based on its attributes. To achieve this, a XGBoost classifier machine learning model was employed. This model is chosen for its interpretability and effectiveness in handling categorical data.

Data Collection

A comprehensive dataset of kernel characteristics is utilized for training the machine learning model. This dataset includes features such as Area, Perimeter, Compactness, Length of kernel, Width of kernel, Asymmetry Coefficient, Length of kernel groove attributes that describe the physical properties of kernel.

Data Preprocessing

The preprocessing of the dataset involves several critical steps to ensure it is in the right format for the algorithm:

- **Data Cleaning**: Any missing or inconsistent data is handled appropriately to ensure a clean dataset.
- **Feature Scaling**: Although not always necessary for XGBoost, feature scaling can be applied to normalize the data.

Model Training

The XGBoost is trained using the following steps:

- **Splitting the Data**: The dataset is divided into training and testing sets to evaluate the model's performance.
- **Training the Model**: The decision tree algorithm is trained on the training set, learning the relationships between the features and the target variable (Kama, Rosa and Canadian).
- **Hyperparameter Tuning**: Various hyperparameters of the XGBoost model, such as learning rate, maximum depth, number of estimators, subsample, and colsample_bytree, are tuned to optimize the model's performance. This tuning process helps improve the model's accuracy, prevent overfitting, and ensure that it generalizes well to new data.

Prediction

Once the model is trained, it can be used to classify the type of wheat kernel based on new input data:

- **User Input**: Users input the characteristics of a wheat kernel through the application’s user interface.
- **Data Processing**: The input data is pre-processed in the same way as the training data (scaling and transformation) to ensure consistency.

- **Model Inference**: The processed input data is fed into the trained XGBoost classifier, which outputs a prediction indicating whether the wheat kernel belongs to the 'Kama', 'Rosa', or 'Canadian' class.

Model Evaluation

To ensure the reliability and robustness of the model, several evaluation metrics are used:

- **Accuracy**: Measures the overall correctness of the model's predictions.
- **Precision and Recall**: Evaluates the model's performance, particularly in correctly identifying poisonous mushrooms.
- **Confusion Matrix**: Provides insight into the number of true positive, true negative, false positive, and false negative predictions.

2.4 Deployment:

The Wheat Kernel Classification application was deployed using Flask, a lightweight web framework for Python that facilitates the creation of web applications. Flask is well-suited for this project due to its simplicity and flexibility in integrating with machine learning models and handling web requests.

Deployment Process

1. **Setting Up the Flask Environment**:
    - A Flask environment was established by setting up the necessary dependencies and configuring the application server. This involved installing Flask and any additional libraries required for handling HTTP requests and running the web server.
2. **Creating the User Interface**:
    - An interactive user interface was developed using Flask's templating engine, Jinja2. The interface includes forms for users to input various kernel features such as area, perimeter, compactness, and asymmetry coefficient. The web pages are designed to be user-friendly and responsive, making it easy for users to input the characteristics of a wheat kernel for classification.
3. **Integrating the Machine Learning Model**:
    - The trained XGBoost model, which classifies wheat kernels into one of three categories: 'Kama', 'Rosa', or 'Canadian', was integrated into the Flask application. When users submit their input through the web interface, the data is processed and fed into the model.
    - The model's predictions are then displayed on the web page in real-time, providing users with immediate feedback on the classification based on the inputted kernel attributes.
4. **Handling User Inputs**:

- User inputs are collected from the web forms and processed using Flask's request handling mechanisms. The data is pre-processed (e.g., scaled) to match the requirements of the XGBoost model before making predictions.

5. **Testing and Deployment**:
    - Extensive testing was performed to ensure the application functions correctly across different scenarios and devices. After testing, the application was deployed to a web server, making it accessible to users via a web browser.

By using Flask, the Wheat Kernel Classification application provides a streamlined and efficient platform for users to interact with the XGBoost model, ensuring a smooth experience for classifying wheat kernels based on their characteristics. To track the application's performance and quickly resolve any issues, logging and continuous monitoring were implemented to guarantee an optimal user experience.

Chapter - 3

Proposed System

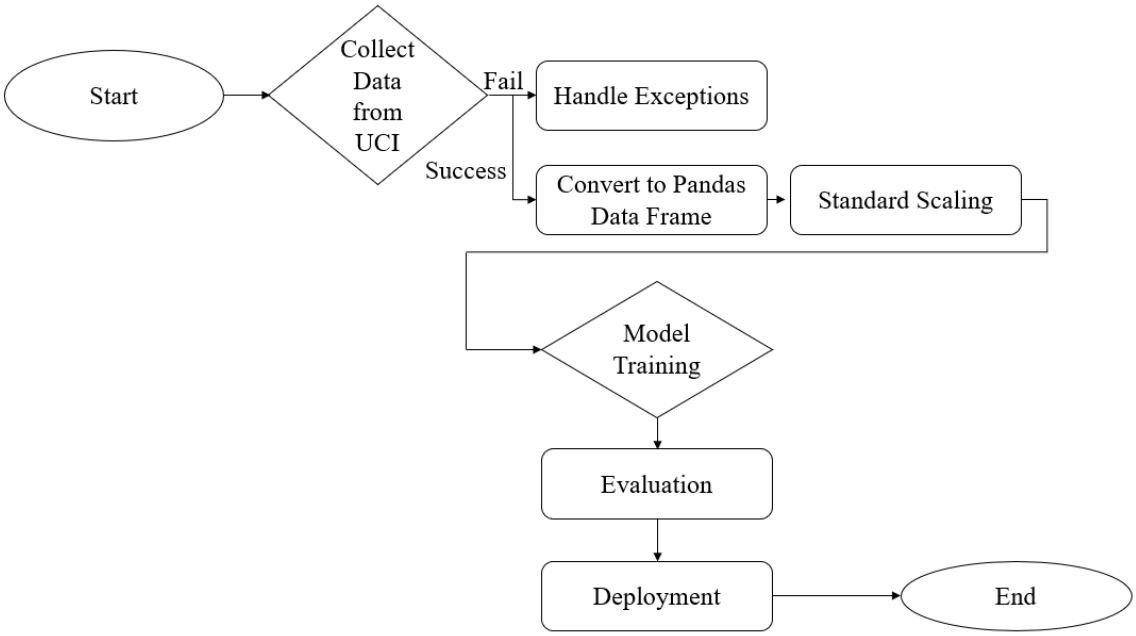
The proposed **Wheat Kernel Classification System** aims to provide an accurate and user-friendly tool for classifying wheat kernels into one of three categories: 'Kama', 'Rosa', or 'Canadian'. This system leverages advanced machine learning techniques to analyze various physical attributes of wheat kernels, such as **area, perimeter, compactness, kernel length, kernel width, asymmetry coefficient, and groove length**. By inputting these attributes, users can obtain reliable predictions regarding the type of wheat kernel, assisting in agricultural research and grain quality analysis.

To achieve this, the system employs a sophisticated **XGBoost model** that effectively captures the patterns and correlations within the data. This model was trained on a comprehensive dataset sourced from the **UCI Machine Learning Repository**, ensuring its robustness and reliability. The XGBoost classifier processes the input attributes and delivers accurate predictions about the wheat kernel type, making it a valuable tool for farmers, agronomists, and researchers.

The system is deployed using **Flask**, a powerful web framework for building interactive web applications. This deployment allows users to access the wheat kernel classification tool through a user-friendly web interface, where they can easily input the relevant kernel attributes and receive instant predictions. The integration of the machine learning model with Flask ensures that the system is both accessible and responsive, offering users a seamless experience. This combination of advanced machine learning techniques and intuitive deployment makes the Wheat Kernel Classification System a valuable tool in the field of agricultural data science.

Chapter - 4

Model Training and Validation Workflow



The provided diagram outlines the workflow of the Mushroom Classification System, detailing the steps from data collection to deployment. Here's an explanation of each step in the process:

1. Start: The process begins with the initiation of the system.
2. Collect Data from UCI Machine Learning Repository: The system first attempts to collect the dataset from UCI. If the data collection fails, the system moves to the "Handle Exceptions" step.
3. Handle Exceptions: Any issues encountered during data collection are managed here, ensuring the system can handle errors gracefully without crashing.
4. Convert to Pandas DataFrame: Upon successfully collecting the data, it is converted into a Pandas DataFrame, which is a tabular data structure suitable for analysis in Python.
5. Standard Scaling: Consider area and perimeter. As they have values in the order of tens and others in the ones, we need to do the standard scaling. In order to bring them on same scale, we will apply the Standard Scaling on the data. Scaling means the mean will be 0 and Standard Deviation will be 1.
6. Model Training: The processed data will be trained on the XGBoost Classifier. This step is crucial as it involves teaching the models to make accurate predictions based on the data.
7. Evaluation: The ensemble model is evaluated using various metrics to assess its performance. This step ensures the model meets the desired accuracy and reliability standards before deployment.
8. Deployment: The final step involves deploying the trained and evaluated model using Flask, making it accessible for real-time predictions.
9. End: The process concludes once the model is successfully deployed and ready for use.

The diagram provides a comprehensive view of the workflow, illustrating the sequence of steps and decision points in the system. This structured approach ensures that each phase of the mushroom classification process is systematically handled, from data collection to model deployment.

Chapter - 5

Interpretation

However, it's important to consider the potential for overfitting, especially since the model achieved perfect scores. Further evaluation on a different validation set or using cross-validation techniques could provide additional insights into the model's generalizability.

To make sure the Wheat Kernel Classification System is accurate and reliable in determining whether a kernel is Kama, Rosa or Canadian, it is essential to assess its performance. The efficacy of the machine learning models included into the system has been evaluated using a number of measures, such as accuracy, precision, recall, F1 score.

The easiest indicator to understand is accuracy, which shows what percentage of all cases were properly identified. It gives a broad idea of how often the model is accurate. But accuracy on its own might be deceptive, particularly if the dataset is unbalanced. A model that consistently predicts "Kama" kernel would do poorly in detecting deadly Kernels, for instance, if the dataset includes a disproportionately larger number of Rosas than Kamas and Canadians. A more complex picture of model performance is provided by precision and recall.

he ratio of actual positive predictions to all of the model's positive predictions is known as precision. In the context of wheat kernel classification, precision indicates the proportion of correctly identified wheat types (Rosa, Kama, or Canadian) among those predicted to be of a particular type. High precision is essential in this application as it reduces false positives, ensuring that the model accurately identifies wheat types, which is crucial for quality assessment and agricultural decision-making.

Conversely, recall is defined as the ratio of all actual positives in the dataset to the genuine positive predictions. In the context of wheat kernel classification, recall assesses how well the model can identify all types of wheat (Rosa, Kama, and Canadian) present in the dataset. High recall means fewer false negatives, ensuring that the majority of wheat types are correctly recognized.

The F1 Score is a single statistic that balances precision and recall, calculated as the harmonic mean of the two. This metric is particularly useful in cases where the dataset is unbalanced, as it incorporates the benefits of both precision and recall into a single value. An F1 score approaching 1 indicates a high-performing model that effectively balances recall and precision.

By utilizing these comprehensive evaluation metrics, the Wheat Kernel Classification System's performance is rigorously assessed, ensuring that it provides accurate and reliable predictions for different wheat types, which are crucial for agricultural decisions.

The XGBoost classifier has performed exceptionally well on the test dataset, achieving 100% accuracy, precision, recall, and F1 score. This suggests that the model is extremely effective at distinguishing between the three types of wheat: Rosa, Kama, and Canadian.

Chapter - 6

Conclusion

The Low-Level Design (LLD) diagram for the Wheat Kernel Classification System provides a detailed and systematic view of the entire workflow, from data collection to deployment. Each step in the diagram is meticulously outlined to ensure clarity and coherence in the process.

The initial phase of collecting data from a specified source and handling exceptions sets a robust foundation for the system. Converting the data into a Pandas DataFrame and performing data visualization allows for a comprehensive understanding of the dataset, while applying label encoding and standard scaling prepares the data for model training.

The model training phase is critical, involving the machine learning algorithm—XGBoost classifier. This algorithm is known for its efficiency and effectiveness in handling classification tasks. XGBoost builds upon the principles of decision trees, using an ensemble approach to improve prediction accuracy and mitigate overfitting. By combining the strengths of individual trees, the model captures various aspects of the data, leading to a more accurate and reliable classification system.

Evaluation of the model using different metrics is crucial to validate its performance and ensure it meets the desired accuracy and reliability standards. Finally, the deployment phase using Flask ensures that the model is accessible for real-time predictions, making it a practical and valuable tool for users.

In summary, the LLD diagram highlights a well-structured and methodical approach to developing the Wheat Kernel Classification System. By breaking down the process into clear, manageable steps, it ensures that each phase is executed effectively, leading to a robust and accurate classification system ready for deployment. This detailed design document serves as a crucial blueprint for developers, providing a clear path from conceptualization to implementation.