**Machine Learning-Based Diagnosis of Lumpy Skin Disease**

**Rakshitha S, Prajwal KM, Mir Mohammed Medhi,Keneeth santhis**

Department of artificial intelligence and machine learning, The Oxford College of Engineering ,Banglore, Karnataka, India

**ABSTRACT:**

Lumpy Skin Disease (LSD) is a highly contagious viral infection that significantly impacts cattle and poses a threat to the global livestock industry. Traditional diagnostic methods are often slow and prone to inaccuracies, which delay timely interventions and exacerbate the spread of the disease. This research introduces a novel machine-learning-based diagnostic system leveraging Convolutional Neural Networks (CNNs) and MobileNet architectures to detect LSD efficiently and accurately. The system utilizes image-based analysis of cattle skin to classify them as either affected or healthy, reducing reliance on subjective or time-consuming laboratory tests. This innovative approach offers a scalable and effective solution for veterinary practices and livestock management, helping to mitigate economic losses and enhance animal health outcomes.

**INTRODUCTION :**

Lumpy Skin Disease (LSD) has become a growing concern in the livestock industry due to its rapid spread and economic implications. It is caused by the Lumpy Skin Disease Virus (LSDV), a member of the *Poxviridae* family, which manifests as nodules on the skin of cattle.

LSD leads to reduced milk and meat production, abortions in pregnant cattle, and, in severe cases, sterility or death. The disease is enzootic in various parts of Africa, but its recent emergence in previously unaffected regions has highlighted the need for efficient control measures. Traditional diagnostic methods rely heavily on clinical observations and laboratory tests, which are time-intensive and require specialized expertise. As a result, there is often a delay in identifying and isolating infected animals, leading to the rapid spread of the disease within cattle populations.

Machine learning has emerged as a transformative tool in disease diagnostics, offering the potential for rapid, accurate, and scalable solutions. This research focuses on utilizing machine learning models, particularly CNNs and MobileNet, to automate the diagnosis of LSD using image-based data. By integrating advanced algorithms with robust datasets, the proposed system aims to address the limitations of traditional diagnostic approaches.

The motivation for this research stems from the urgent need to improve livestock health management practices and reduce the economic burden caused by LSD outbreaks. Leveraging the power of machine learning, this study proposes a diagnostic system that can be easily deployed in real-world settings, providing farmers and veterinarians with a reliable tool for early disease detection.

**Literature Review**

The application of machine learning in agriculture and animal health has gained significant traction in recent years. Researchers have explored various algorithms to address challenges in disease detection, yield prediction, and environmental monitoring. For example:

* Gupta et al. (2022) demonstrated the efficacy of CNNs in identifying diseases in paddy crops, achieving high accuracy in detecting conditions like Rice Tungro and Brown Spot.
* Singh et al. (2022) utilized deep learning models to detect apple leaf diseases, achieving an accuracy of 99.2% with a dataset of over 50,000 images.
* In the context of LSD, Afshari Safavi (2022) explored the role of geospatial and meteorological features in predicting disease outbreaks. Using an Artificial Neural Network (ANN) model, the study achieved 97% accuracy in forecasting LSD occurrences. However, this approach focused on environmental factors rather than direct diagnosis through visual inspection of cattle.

Traditional machine learning algorithms, such as Random Forest and Decision Trees, have also been employed for animal disease diagnostics but often lack the robustness required for complex image-based tasks. This research builds on these advancements by combining the feature extraction capabilities of CNNs with the efficiency of MobileNet. The proposed system not only addresses the challenges of diagnosing LSD but also provides a scalable solution that can be extended to other diseases in livestock.

**Methodology**

The proposed system employs a combination of machine learning and deep learning algorithms to diagnose LSD. The architecture is designed to handle image-based data, enabling rapid and accurate classification of cattle as either healthy or affected by the disease.

The dataset used for this study is sourced from Kaggle and includes labeled images of healthy and LSD-affected cattle.

Key Stages:

1. Data Preprocessing:
   * Resizing: Ensures all input images have a consistent size using techniques such as nearest neighbor interpolation.
   * Normalization: Adjusts pixel values to a common scale (0–1), reducing biases from varying lighting conditions.
   * Augmentation: Enhances dataset diversity with random rotations, flips, and brightness/contrast adjustments.
2. Image Quality Enhancement:
   * Denoising: Reduces noise to preserve critical features.
   * Sharpening: Enhances edges and details for better feature detection.
3. Model Architecture:
   * CNNs: Extract hierarchical features from images, focusing on relevant areas while ignoring noise.
   * MobileNet: Lightweight yet accurate, ensuring efficient processing even on lower-resolution images.
4. Handling Aspect Ratios:
   * Implements padding to retain image information while achieving uniform input dimensions.

The training phase involves fine-tuning the CNN and MobileNet models using a labeled dataset. Evaluation metrics such as accuracy, precision, recall, and F1-score are used to assess performance. The final model is validated on a separate test dataset to ensure reliability in real-world applications.

**System Implementation:**

The system comprises several modules:

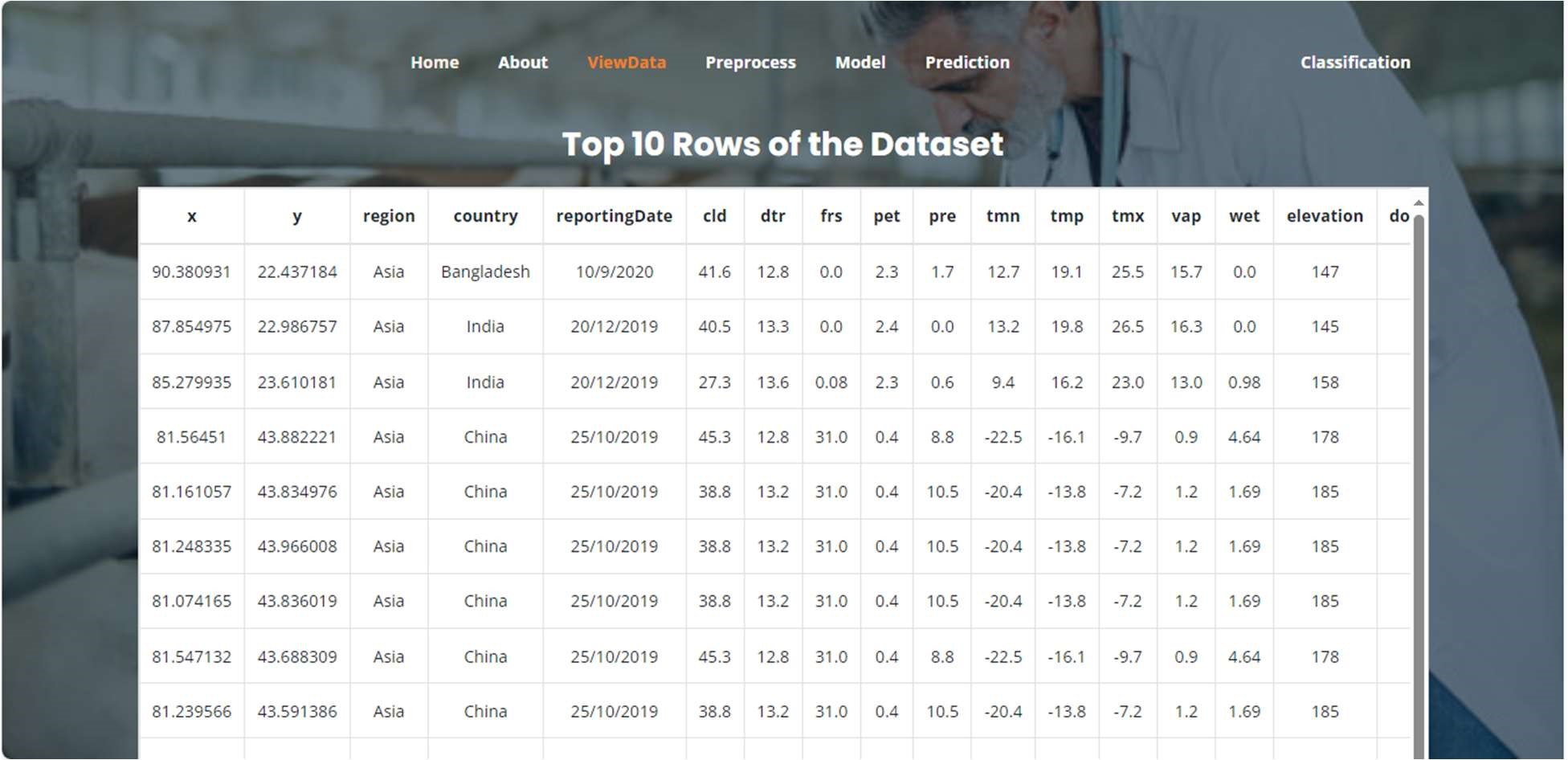
1. Data Upload Module: Users can input cattle images and associated clinical data.
2. Preprocessing Module: Cleans and transforms input data for analysis.
3. Diagnostic Engine:
   * Image-based classification using CNNs and MobileNet.
   * Feature-based predictions using Naive Bayes and Extra Trees.
4. User Interface: Displays results and actionable insights for veterinarians and farmers.



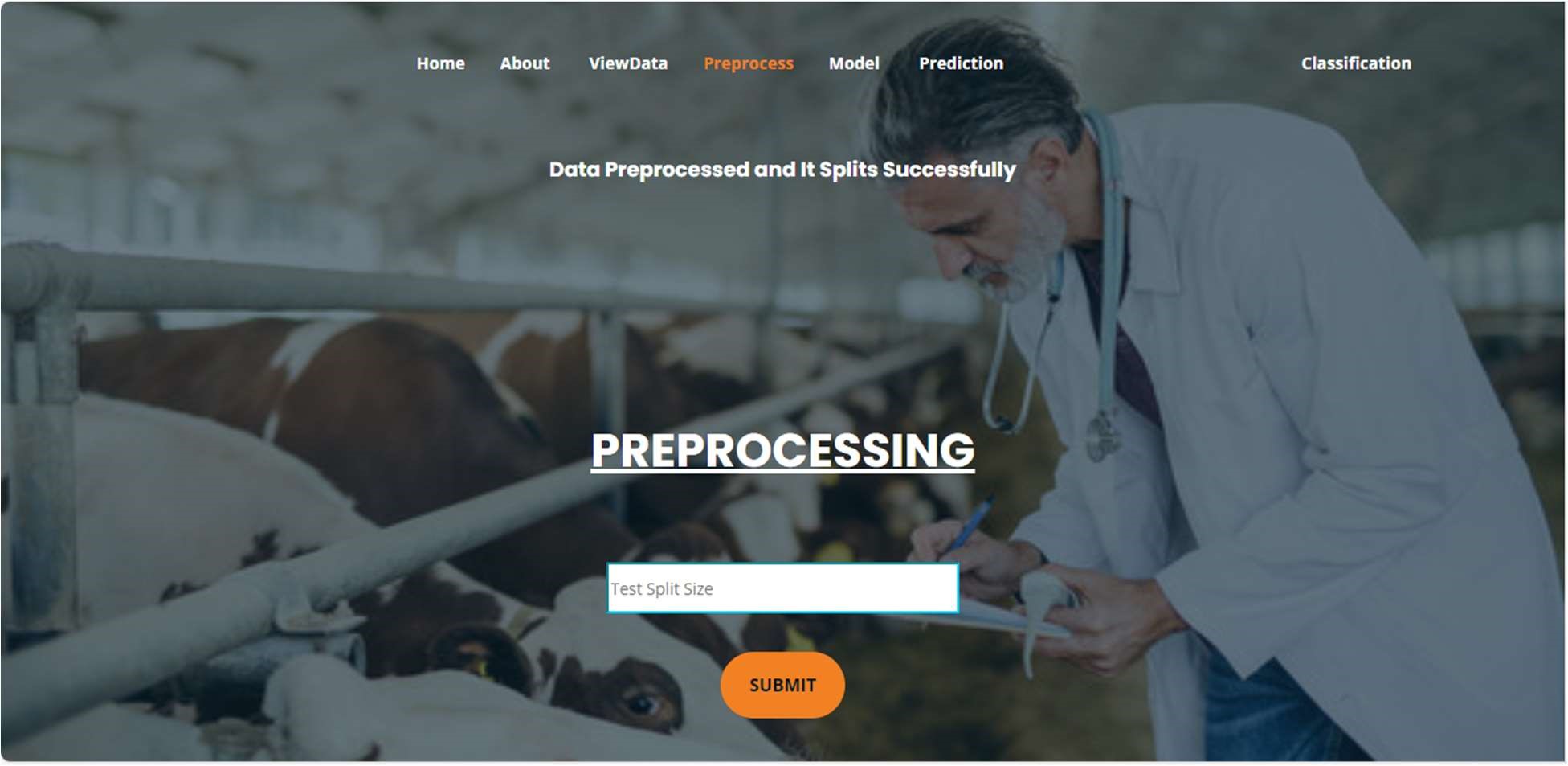
ABOUT PAGE: Here we can read about our project.



View Data: In the Viewdata page, users can view the skin dataset.



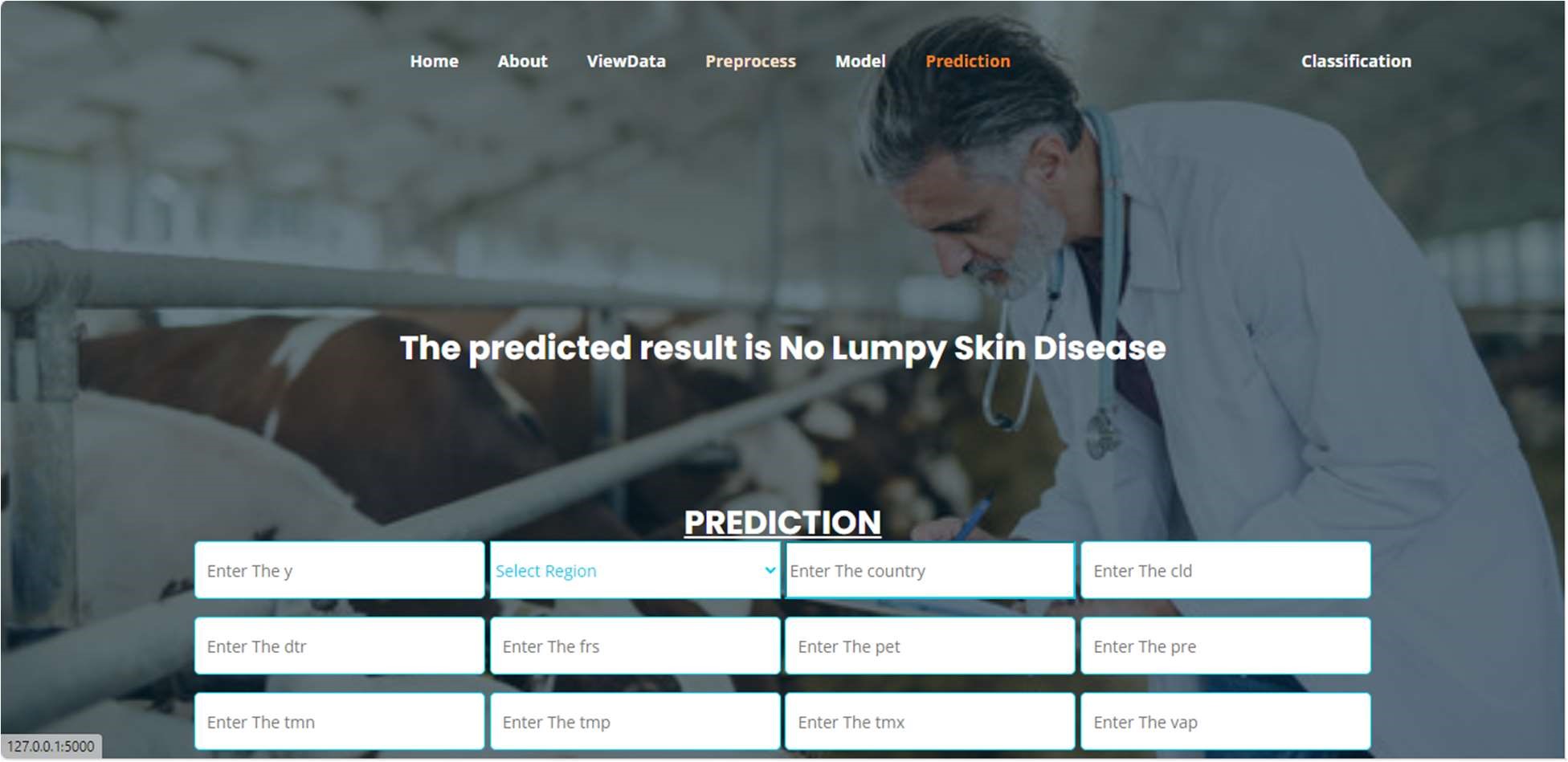
Pre-process: Here we can pre-process and split our data into train and test.



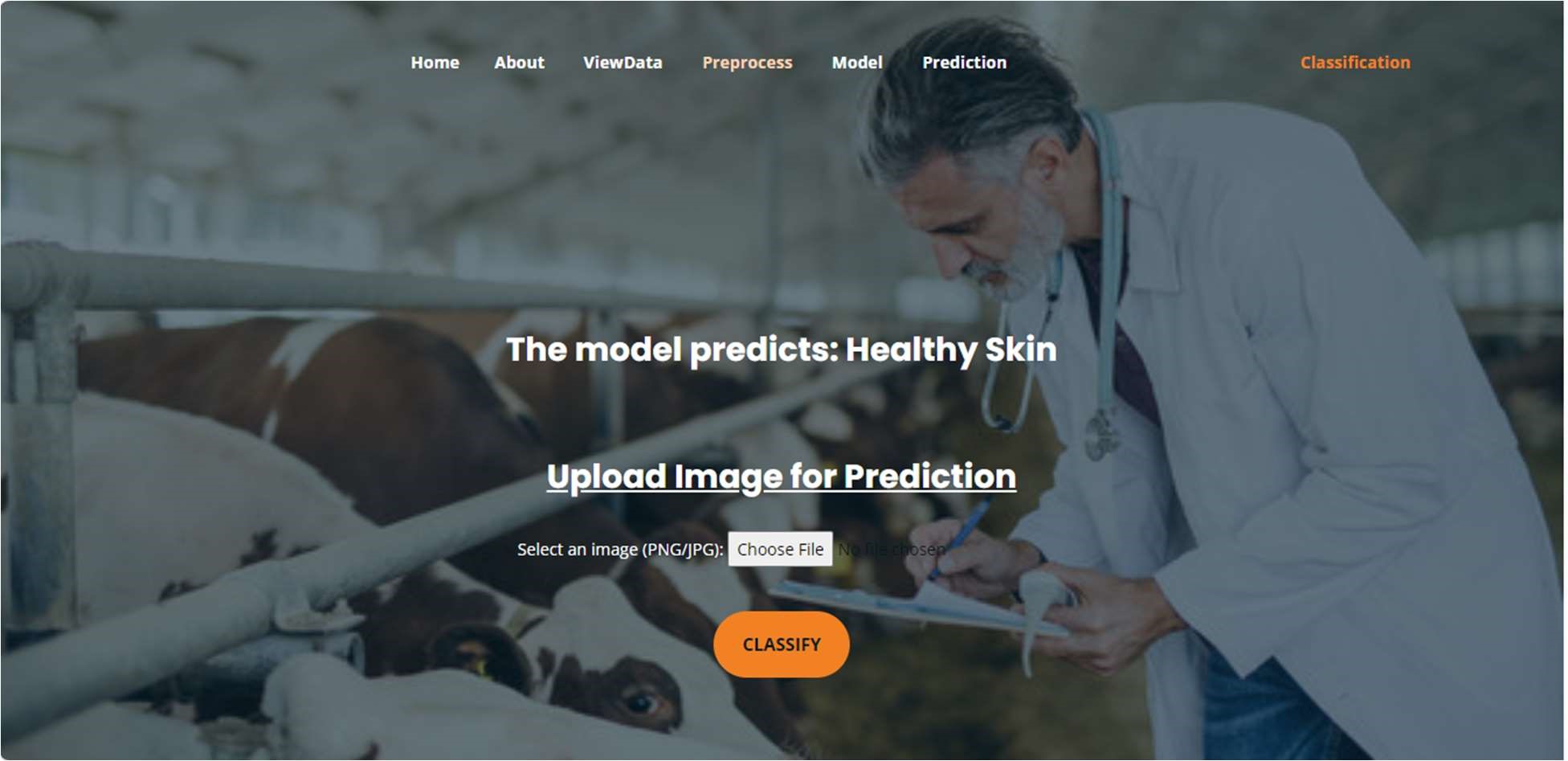
Model: Here we train our data with different ML algorithms.



Prediction: This page show the result of the user given input data.



Classification: This page show the result of the user given image.



# SYSTEM STUDY AND TESTING

**Feasibility Study**The feasibility of the project is analyzed in this phase, and a business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis, the feasibility study of the proposed system is carried out to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

* Economical Feasibility
* Technical Feasibility
* Social Feasibility

**Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of funds that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system is well within the budget, and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

**Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or no changes are required for implementing this system.

**Social Feasibility**

This aspect of the study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system; instead, they must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make them familiar with it. Their level of confidence must be raised so that they are also able to make some constructive criticism, which is welcomed, as they are the final users of the system.

**Tests**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

**Types of Tests**

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g., components in a software system or—one step up—software applications at the company level interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.