Time Series Neural Network Software Suite & Application Development for Dairy Herd Monitoring

Student Name: Jyothesh Karnam

Student ID: 23032448

MSc in Computer Science 2023-2024

13/09/2024

School of Computer Science and Mathematics

Keele University

Keele

Staffordshire

ST5 5BG

Abstract

The MSc project presents the development of a Time-Series Neural Network Software for Dairy Herd Monitoring. The software aims to predict health issues in dairy cows using Artificial Intelligence (AI) and Machine Learning (ML) models. The system analyses activity and temperature data provided by Keele Harper Adams Veterinary Science School to forecast and detect health alerts in dairy cows. A user-friendly GUI was developed for easy data uploading, model training and result visualization. The system uses machine learning models to improve herd management and incorporates evaluation metrics on the trained models to ensure accurate results. The dairy herd monitoring system aims to assist users in health management of dairy cows.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr Charles Day, for his continuous guidance and insights throughout the development of time-series neural network software suite and app development for dairy herd monitoring MSc project. His guidance and valuable insights were crucial to the development of the software. I am also incredibly thankful to Keele Harper Adams Veterinary Science School for providing the dairy herds datasets, which were essential for developing the dairy herd monitoring system.

Table of Contents

List of Tables

List of Figures

Glossary

- **AI (Artificial Intelligence):** The ability of machines to perform tasks that generally require human intelligence, such as learning and problem-solving.
- **Classification Model:** A machine learning model that categorizes data into predefined classes.
- **Confusion Matrix:** A table used to evaluate the performance of a classification model by comparing actual vs predicted classifications.
- **Graphical User Interface (GUI):** A user interface that allows interaction through graphical elements.
- **Long Short-Term Memory (LSTM):** A type of recurrent neural network that is capable of retaining information for longer periods.
- **Machine Learning (ML):** A subset of AI that uses algorithms which is capable of learning patterns from data to make predictions or decisions.
- **Regression Model:** A type of machine learning model that learns on input data to predict continuous outcomes.
- **Time-Series Data:** data recorded at different time intervals, generally used to analyse trends over time.

Chapter 1 – Introduction

1.1 Introduction

Dairy herd management has evolved remarkably over the years, with new technologies implemented into the systems making them significantly important in animal welfare. With dairy farms on a large scale becoming more common, monitoring the health of dairy cows both individually and collectively is important to ensure their productivity and their lifespan. Traditional monitoring methods for dairy herd involves manual approach, which are labour intensive methods and also prone to human error. These approaches could possibly delay the early signs of illness in dairy herd because they often lack precision, leading to sever health complications in animal welfare (Nalage et al, 2024).

The rise of Artificial Intelligence (AI) and machine learning (ML) have revolutionized prediction tasks by analysing historical data in the recent times. Sharma et al. (2020) highlights how machine learning applications have transformed predictive tasks in agriculture, improving productivity and reducing labour work. By implementing AI/ML models into the dairy her monitoring system, farmers and veterinarians can make decisions based on the early predictions of potential health issues in the dairy herd shown by the machine learning models. This will minimize the labour-intensive tasks and improve dairy herd welfare. The MSc project aims to develop a time-series neural network software suite for dairy herd monitoring to predict the health outcomes in dairy

cows, using activity and temperature sensor data provided by Keele Harper Adams Veterinary Science School.

1.2 Problem Statement

Manual dairy health monitoring of dairy cows id both time-consuming and prone to human error. this method might lead to resulting in delayed responses to health issues in the dairy cows. Without continuous monitoring of the herd, early signs of illness become difficult to detect, leading to health issues in the herd. Specifically in larger herds, manual health monitoring becomes complicated to monitor individual cows and the herd.

An effective system with predictive tools, which can analyse time series sensor data of the dairy herd to forecast the future states and identify the health alerts helps the farmers or veterans to take better decisions to prevent health risks in the dairy herd. The core problem of the dairy herd management is to achieve a higher accuracy of the model predictions. Higher accuracy allows the end user to make informed decisions. This involves in overcoming technical challenges while developing the software such as selecting and developing the right models for time-series based sensor data, ensuring better model accuracy and achieving a balance between predictive performance and computational efficiency.

1.3 Overview of Previous Work

Previous work in animal health monitoring has utilized machine learning techniquies, with a significant focus on developing AI/ML models for predicting health alerts in

livestock. Specifically, the application of Long-Short-Term (LSTM) networks has shown significant progress in health monitoring systems. Keceli et al. (2020) implemented a Bi-directional LSTM based method for predicting calving times using historical time-series data behavioural and activity observations of dairy cows. The study found that LSTM networks, with their ability to analyse sequential data and capture long term dependencies improved the accuracy of calving predictions accurately. Making LSTM networks suitable for dairy herd monitoring.

1.4 Academic and Practical Relevance

From an academic perspective, the project aims to develop a software to contribute to the field of AI in livestock management, specifically in dairy herds. The integration of LSTM models processing the real-world sensor data provided by Keele Harper Adams Veterinary Science School, the software aims to provide better predictions with higher accuracy in identifying health alerts of the dairy cows.

Practically, the dairy herd monitoring software aims to deliver predictive tools that will have direct impact on the dairy management operations. The software helps the users to take better decisions and optimize their strategies in herd management, the system aims to be a practical example of how AI/ML techniques are applied in live stock management.

1.5 Aims and Objectives

Aims: The aim of this project is to develop a software suite that uses time series neural networks such as LSTM models to predict when action should be taken to address the health issues in dairy cows. The software aims to improve the welfare of the dairy herd by providing early detection of health issues based on activity and temperature data.

Objectives:

1. **Data selection and Visualization:** Develop a user-friendly interface that allows the user to upload and select datasets related to activity and temperature of the cows. Visualization tools will be implemented in the software that allows the users to visualize the raw data they have uploaded.
2. **Data Preprocessing:** the data will be pre-processed to improve the performance of machine learning models. Ensuring that the data is clean and ready for LSTM model training and testing. The user should be able to visualize the raw data and pre-processed data in graph format.
3. **Model Training and Prediction:** Train LSTM models using pre-processed time-series data. The models will predict potential health issues based on activity and temperature data. The software provides visualized results for decision making.
4. **Model Management**: Implement features to save and reload trained models. This ensures that models can be reused and fine-tuned over time

1.6 Software Engineering Methodology

This project follows the SCRUM framework, an agile Software Development Life Cycle (SDLC) methodology that prioritizes iterative development, continuous feedback and improvement. Scrum framework is suitable for the development of dairy herd monitoring software due to its sprint-based approach. This allows to prioritize on specific tasks such as GUI development, model training and system integration. The Gantt chart shows a series of sprints. The project begins with project planning and ethics, system design and analysis, followed by software development, AI/ML model

implementation, system integration, poster design, AI/ML evaluation, Software Evaluation, Report writing and Documentation.

1.7 Conclusion

In conclusion, the project explains the importance of machine learning models to develop prediction tools that can predict the dairy cow's health outcomes. By utilizing LSTM models to process historic time-series sensor data, the dairy herd monitoring system aims to provide farmers or veterans to manage the herd more effectively.

Chapter 2 – Literature Review

2.1 Introduction

Dairy farming has seen a consequential shift towards technology-driven livestock management practices, with AI/ML techniques emerging as a crucial tool for enhancing herd health and productivity. Specifically, the application of LSTM models to predict health issues in live stock management based on the historical time series data has emerged as an effective approach because of its prediction capabilities. The aim of this chapter is to review relevant literature on time series data analysis, the use of Recurrent Neural Networks and LSTMs and their application in health monitoring.

2.2 Time-Series Data in Dairy Cow Health Monitoring

The time series data provided by Keele Harper Adams veterinary science school provides important information related to daily activity levels and temperature data of the dairy herd. Monitoring their activity levels and temperature data with the help of AI/ML models can help predict potential health problems in the dairy heard before they

occur. As Sherstinsky (2020), explains LSTM models, with their ability to retain long-term dependencies are suitable for this type of data analysis. By identifying the changes in behaviour and temperature over longer periods, LSTM models are capable of predicting and providing early warnings of potential health issues in the dairy herd.

2.3 Recurrent Neural Networks and LSTM Models

Recurrent Neural Networks (RNNs) have shown their ability to process sequential data where the outputs are significantly affected based on the order of inputs, but traditional RNNs face challenges to retain information over longer sequences because of the vanishing gradient problem and this limits the traditional RNNs effectiveness in capturing long-term dependencies (Sherstinsky 2020). In health monitoring systems, this issue could be particularly relevant when used traditional RNNs to predict health issues because the changes in cows' health might evolve over longer periods. The vanishing gradient problem in traditional RNNs reduces the ability to capture relevant information of data over time which could possibly lead to potential lapses in prediction accuracy.

A specialized form of RNNs named Long Short-Term Memory (LSTM) networks were developed to solve the issue faced by traditional RNNs by introducing memory cells that can store information for longer periods as these memory cells, controlled by gates, enable LSTM networks to retain information for longer timeframes (Sherstinsky, 2020). This makes LSTM models particularly suitable for analysing time series data and has the potential to predict the health issues in dairy cows when it is integrated into dairy herd monitoring system.

2.4 Application of LSTM in Health Monitoring

LSTM network's ability to capture long-term dependencies in time series data have proven to be efficient in health monitoring. Zhao et al. (2016) showcased the success of LSTM modes in machine health monitoring by using raw sensor data from a CNC milling machine to predict the tool wear. Their study showed that LSTM models had outperformed traditional models such as Multi-Layer Perceptron (MLP) and Support Vector Regression (SVR) by predicting conditions without domain knowledge because LSTMs are capable of learning the patterns accurately.

Similarly, LSTM models can be applied in dairy herd monitoring system by giving the provided activity and temperature data as an input to the LSTMs to detect early health issues in dairy cows. LSTM's ability to simulate long-term patterns, timely actions can be made to improve animal welfare.

2.5 Model Training and Prediction

The process of training LSTM models for dairy herd monitoring will involve in using pre-processed activity and temperature data. Once the model is trained, the LSTM models will be capable of predicting and classifying health alerts based on the historical time-series data they have been trained on. LSTMs are capable of handling multiple parallel streams of data (Brownlee, 2020). LSTM network's ability to handle multiple parallel streams of data and their efficiency in capturing long term dependencies make them a suitable for detecting anomalies in dairy health.

2.6 Conclusion

This chapter reviews the advancements made in health monitoring utilizing LSTM models which are well suited to train on historical time series data. It highlights the relevance of these models in dairy herd monitoring and management, explaining how

suitable the LSTM models are based on their ability to predict health issues based on sensor data. This chapter concludes that machine learning models, specifically LSTM models are suitable compared to traditional health monitoring techniques.

Chapter 3 – System Design and Analysis

3.1 Introduction

This chapter delves into the core design and analysis of the dairy herd monitoring software system. Beginning with outlining the user requirements, both functional and non-functional, explores the evolution of user interface and backend approach. Explains the initial design of the system and identifying its limitations and discuss the improvements made to enhance the user experience and improve performance.

3.2 User Requirements Table

3.2.1 Functional Requirements

Upload data and Visualize features	The system should provide the user to upload datasets, allow the user to visualize raw and pre-processed data.
Data Preprocessing	The system should pre-process raw uploaded data (activity and temperature data) to enhance the performance of LSTM models.

Machine Learning Models	Based on the time series data, the system must train and use machine learning models for predicting dairy cow health alerts.
Save and Reload Models	The system must allow the user to save their trained models to allow for future model use.
Data safety	The data user uploads must be erased automatically once the user closes the dairy herd monitoring application.
Usability	The system should have a simple user-friendly interface so that a user's with minimal technical knowledge can use the dairy herd monitoring software effectively.
Predictive Analysis	The system should provide health predictions based on historical time series data.
Visualize Predictions	The system must provide visualized output of the LSTM models health alert predictions.

Table 3.1

3.2.2 Non-functional Requirements

Performance	To ensure smooth operation, the system should be capable to generate predictions within a reasonable amount of time
Scalability	The system must be scalable to accommodate increasing number of cows and their sensor data without degrading the model performance.
Maintainability	The system must be able to allow enhancements to the user interface and updates to the machine learning models without majorly disrupting the software.

Table 3.2

3.3 Evolution of User Interface Design

The user interface design initially focused on providing visualization of forecasting and predicting health alerts based on activity and temperature data. However, the design had difficulties in visualization and in improving end user experience. hence a redesign of the dairy herd monitoring system was necessary.

3.3.1 Initial Design and Its Limitations

In the initial design, the GUI interface was designed to handle generic architecture of the dairy herd. Allowing users to visualize both forecast data and health alerts. The software provided the following functionalities

1. **Data Upload:** End users can upload activity and temperature data of the dairy herd for analysis.
2. **Data Visualization:** The uploaded data will be displayed in a graph allowing the users to view raw or pre-processed data.
3. **Activity Data Analysis**: A button labelled “Daily Activity Levels” allows users to generate two graphs, one graph to show the forecast of the herds activity using a regression model and another graph to detect the health alerts using a classification model.
4. **Temperature Data Analysis:** Another button labelled “Temperature Analysis” allows users to generate two graphs, one utilizing a regression model to show the forecast the herds temperature and a classification model to detect the health alerts based on temperature data.
5. **Overall Analysis:** A third button, labelled “Overall Analysis” integrates both the activity and temperature datasets and visualizes both forecast and health alerts utilizing regression model for forecasting and classification model for predicting health alerts,
6. **Health Alert Pie Chart:** A pie chart was designed to visualize the possible health alert of the herd. A green alert indicated no immediate action was required, an orange alert indicated a possible health issue where the action was required but not immediate. A red alert signalled that a critical issue, requiring an immediate action.
7. **Download Report:** A “Download Report” button was designed, allowing the users to download a report containing snapshots of the graphs they had visualized for further analysis.

The initial design faced some limitations that impacted the software's effectiveness.

1. **Graph Size:** One of the main issues was the small size of the graphs, which can potentially make the end user find it difficult to visualize forecast graphs and health alert graphs. The limited display reduced the possibility to gain clear clarity of the information on the graph being presented, specifically when it came to analysing trends.
2. **Complexity of Final Results:** The most critical challenge was the complexity involved in rendering the pie chart to indicate the health alerts. To develop a functionality that shows the overall alerts percentage and final alert became complex to develop in a short period of time.

These issues lead to redesign the user interface that provides the user to have a much simpler and better experience.

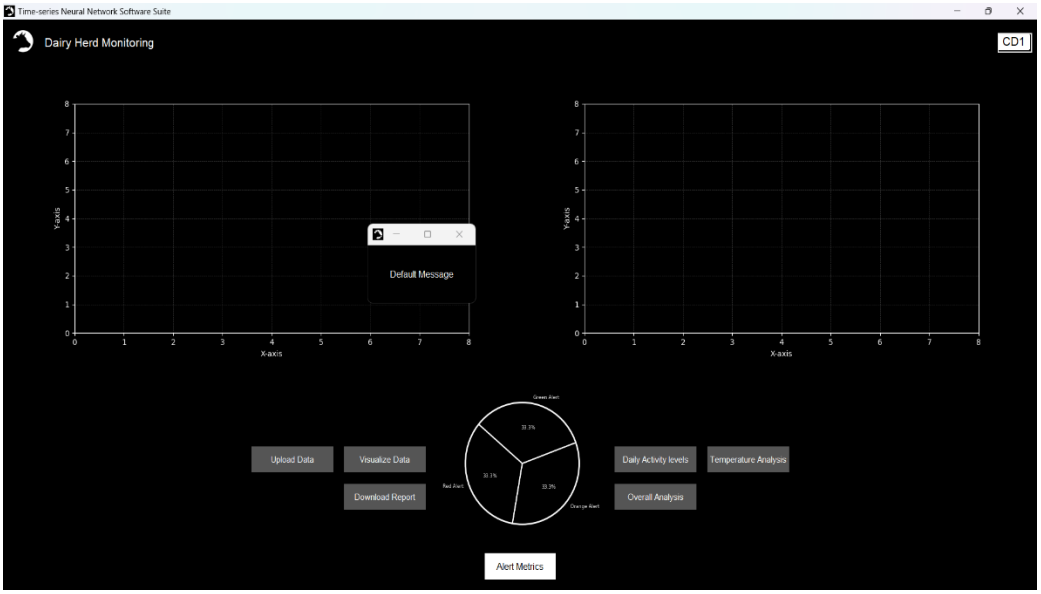


Figure 3.1: Initial interface layout with multiple graphs, buttons to upload and visualize uploaded data, model selections and a pie chart to show the possible health alert percentages.

3.3.2 Final Design

The final user interface designed to address the limitations faced in the initial design. The new design enhanced the user experience. Below is the list of key features involved in the final design of the dairy herd monitoring.

1. **Streamlined Button Layout:** The new interface was built for simple navigation, with buttons namely Upload Data, Visualize Data, Generic Architecture Buttons and Bespoke Architecture buttons.
2. **Graph Display:** Generally, for visualizing uploaded data, there will be only one graph generated, but when using models there will be separate graphs generated which allows the end user to visualize both forecast and health alerts much clearly.
3. **Adding Tooltips:** Adding tooltips to the graphs can give accurate information to the end user
4. **Model Switching:** Users can switch between Generic and Bespoke models seamlessly and visualize the graphs. The pdfs will be automatically downloaded in the new design.
5. **Removals:** Pie chart was removed from the user interface, instead the final alert will be shown in a GUI window after generating the prediction of health alert and forecast graphs.
6. **Reload and Save Models:** Users can train and save their Generic models, and customize the model names while saving, users can also reload the saved models adding this feature to the user interface will improve user experience.

This redesign improves the user experience because the graphs will be more clearly visible and the user can easily switch between options improving simplicity of the user interface.

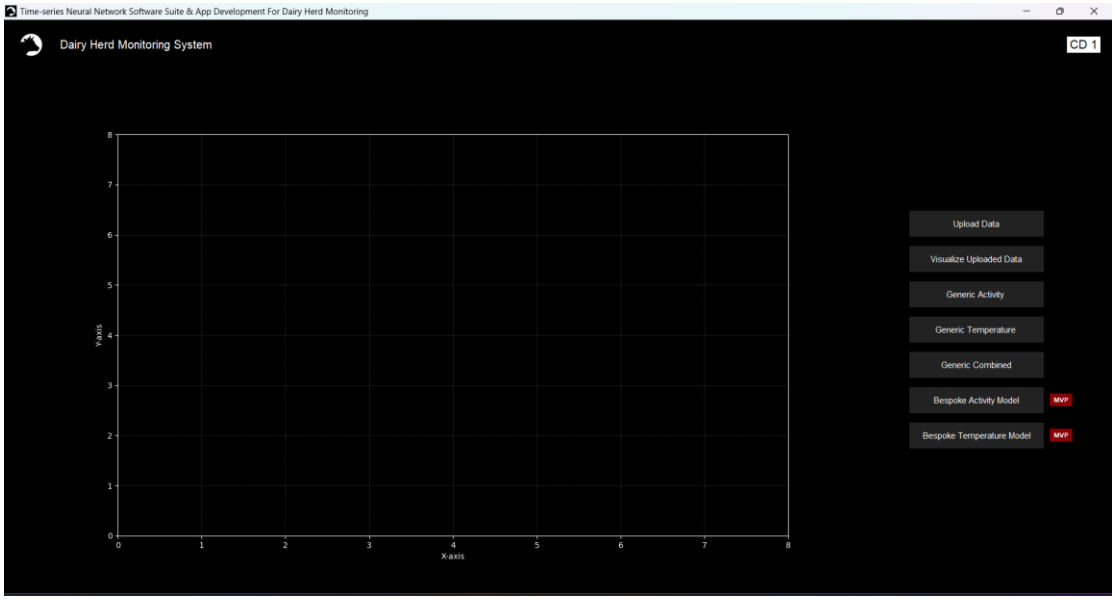


Figure 3.2: A single graph to visualize the raw or pre-processed data, another graph will be generated below the present one if the user uses Generic or Bespoke models.

3.4 Analysis of Backend Approach

The backend process also went under significant changes evolving from a simpler backend architecture to a more complex design to provide more options to the end user.

3.4.1 Initial Backend Approach

The initial backend process was mainly focused on generic architecture such as generic activity, generic temperature and generic combined where the end user can visualize forecasts and health alert predictions from regression and classification models. Vanilla LSTM models are suitable for simpler time-series tasks, particularly while managing simpler multivariable data (Brownlee, 2020). The initial approach was to use vanilla

LSTM models for each metric, generating both regression and classification model outputs for generic activity and temperature architecture.

Additionally, a multi-model approach using Stacked LSTM model was planned to use for overall analysis button. A stacked LSTM model is made up of multiple LSTM layers that are stacked on top of one another, requiring a three-dimensional input to produce a two-dimensional output (Brownlee, 2020). The plan was to use stacked LSTM model which takes the outputs of both generic activity and temperature and convert into a 3D input format and fed as input to the stacked LSTM model. This stacked LSTM model further provides the health alert prediction and forecast of overall analysis button.

Challenges:

1. **Complexity:** Stacked LSTM models might lead to longer training times and difficulties in managing outputs from multiple models and later integrating into a pie chart to show difference in alerts could be complicated to implement in a short period of time.
2. **Performance Issues:** The stacked LSTM models might lead to software performance while switching between generic architectures.

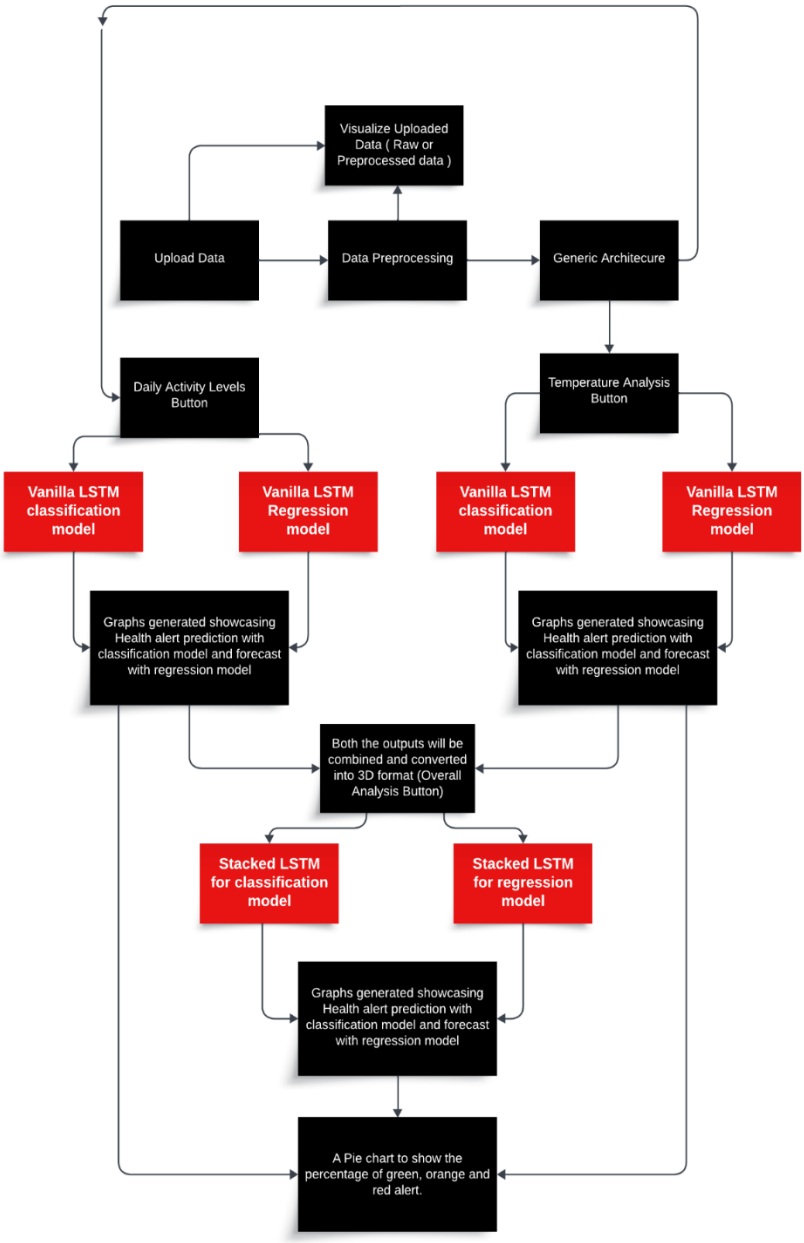


Figure 3.3: showcasing the initial approach to develop the backend of the dairy herd monitoring.

3.4.2 Finalized Backend Approach

In the final approach, using Vanilla LSTM models for all generic architectures and Bespoke models simplified the backend process. This approach eliminated the need for

a stacked LSTM model. In the new approach both activity and temperature data after preprocessing were given as input directly to the vanilla LSTM model for multi model health alert predictions and forecast. In the finalized approach, pie chart to show the health alerts was removed to reduce the complexity of the software.

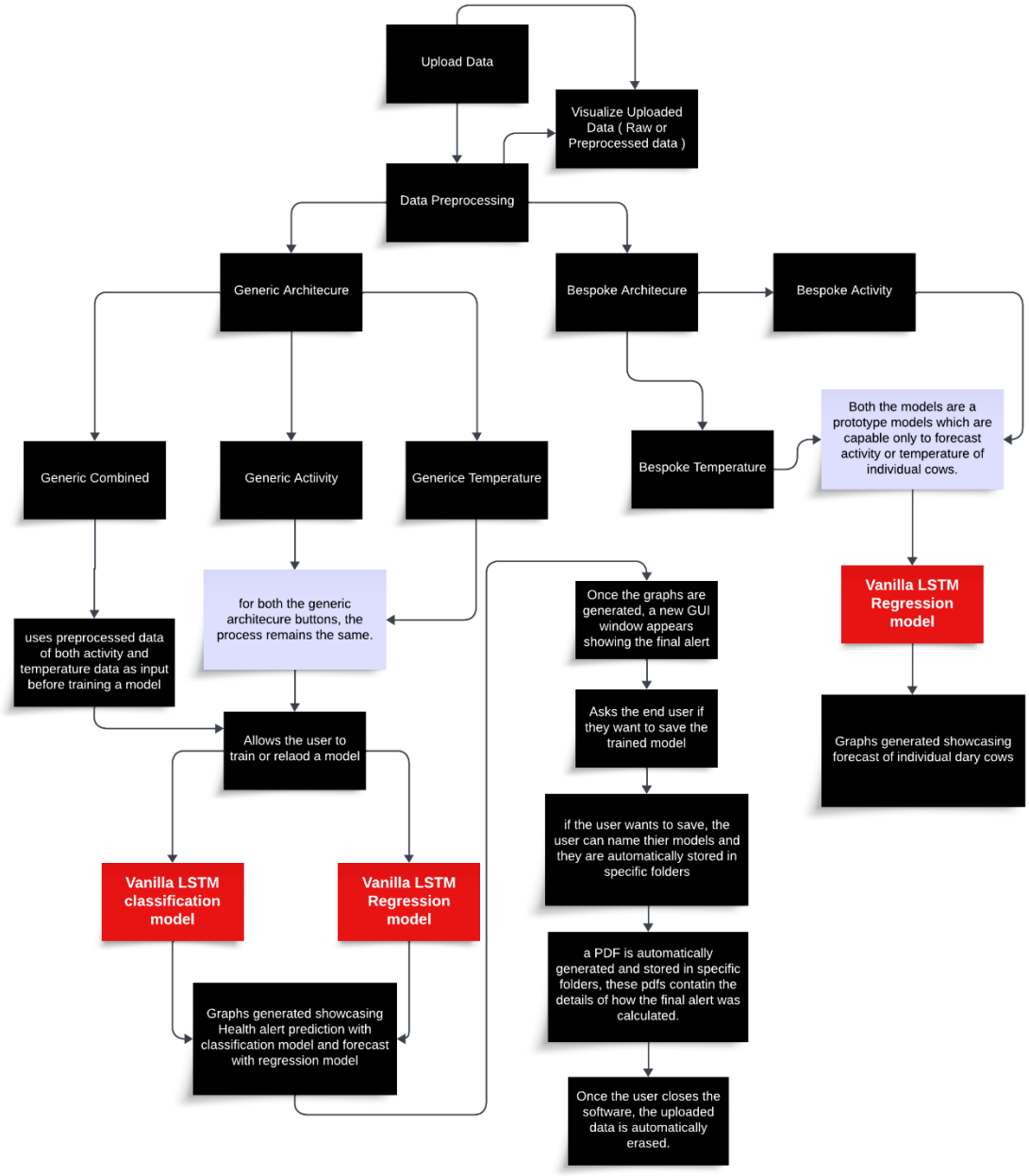


Figure 3.4: Showcases the finalized backend approach of dairy herd monitoring.

3.5 Conclusion

This chapter reviews how the system design focuses on the functional and non-functional requirements of the dairy herd monitoring system. It discusses the evolution of user interface and backend approach showcasing the improvements made to enhance user interface and better performance. In conclusion, the design and analysis were made to meet the needs of user requirements.

Chapter 4 – Frontend Development

4.1 Introduction

This chapter explores the frontend development of the Time-Series Neural Network Software for Dairy Herd Monitoring System. One of the primary objectives was to create a user-friendly interface that provides the users to upload data and visualize results effectively. The frontend of the dairy herd monitoring system was developed using Python's Tkinter library. Tkinter Graphical User Interface (GUI) is chosen for its simplicity and its ability to seamlessly integrate machine learning libraries such as TensorFlow and Scikit-learn which are used in the AI/ML implementation developed in python.

4.2 Overview of Technology Choices.

In the short period of time, selecting the right technology to develop the frontend of the software was crucial, after analysing the options, Tkinter library was finalized because of its straightforward functionality and its platform compatibility. This makes it an ideal choice to build simple user interfaces in minimal development time.

4.2.1 Why Tkinter?

Tkinter, as Fatima (2024) explains, it is a beginner friendly python library which is used to develop GUI applications and Tkinter's cross-platform supports to enable the function on different operating systems such as windows, macOS and Linux. The main reason to choose Tkinter library to develop the frontend of the dairy herd monitoring was to save time and focus more on developing and implementing machine learning models without spending more time on frontend complexities. The library also provides built in widgets, providing the developer to construct a functional GUI with labels, buttons and entry fields. Shukla (2024) highlights that Tkinter is capable to easily integrate with other python libraries. This Tkinter library would be an advantage as the machine learning models for the dairy herd monitoring are developed using python and its libraries such as TensorFlow and Scikit-learn. Hence making Tkinter library suitable for this project, where the GUI interacts with machine learning models. Hence the user can upload, analyse and visualize health alerts directly from the GUI interface.

4.2.2 Challenges with Web and Mobile Applications

Initially, to develop a web or a mobile application was considered for the frontend. But implementing multiple frameworks and hosting after developing the machine learning models, seemed to be impractical for this project in the short period of time. Specifically in terms of user experience and data safety because web or mobile application would be hosted online and it won't be practical do develop encrypted storage of the uploaded data and a secure user authentication feature for users in the given short period of time.

4.2.2.1 Web Applications

Building a web application would require a complex backend infrastructure to manage the data storage, model processing, model management and security. According to Verma et al. (2021), for data storage and processing, web applications depend on backend servers, raising, concerns about data privacy. Handling sensitive data unveils security risks, specifically in cloud-based systems. Hence web applications might have privacy risks and as this project involves in using sensitive data, developing a web application would be less suitable for this project in the given time frame.

4.2.2.2 Mobile Applications

Developing a mobile application might have computational limitations while developing a cross-platform application. As Dai et al. (2020) explains, mobile devices may have computational limitations when running the machine learning models on a local environment. This might lead to negatively impact the performance when models are trained on mobile application environment and could lead to negatively impacting the user experience. Visualizing multiple graphs clearly could be difficult to the end user. Additionally, mobile applications do pose the same data security issues that web applications have. Developing a mobile application for dairy herd monitoring in the given short period of time may not be an ideal approach to develop the software.

4.3 Main Components of the GUI

The Dairy Herd Monitoring System GUI was developed to ease user interaction with machine learning models and data management easier. It is main dashboard where the user can perform their tasks. Below are the main components of the GUI.

4.3.1 Main Dashboard

The main dashboard is the core of the application, where all the options related to data management and interaction with machine learning models are visible. The layout was developed based on the final design to prioritize simplicity and user interaction. The main dashboard provides the user to perform tasks such as uploaded and visualizing data, model training, saving and reloading models and visualizing results. Figure 3.2 visualizes how the main dashboard looks like with all the core functionality buttons for the user to interact with the software.

4.3.2 Data Upload and Visualization

One of the most crucial features of the dairy herd monitoring system software is to provide user with options to upload their data and also to visualize them in both their csv format and in a graph format before the data is automatically pre-processed and given as an input to the machine learning models. This step also provides the user to visualize their uploaded data in pre-processed format. This ensures the end user that the data is complete and accurate, reducing the risks of model training.

4.3.2.1 Upload Data

The Upload Data labelled button is the first of the features user interacts on the main dashboard. This button when clicked opens up a GUI window where the user can upload their activity and temperature data in csv format. This interface is developed to be simple and easy to use, ensuring end users with minimal software knowledge can navigate through the system easily.

- **File Validation:** Initially after the user uploads their datasets, the system provides the user to visualize their uploaded data in csv format. Until then these buttons are disabled. The system also performs an automatic check to validate the uploaded files. Before proceeding, the system ensures the files are in csv format and both data fields are uploaded before proceeding. If there is any issue, the system notifies the user. Once the user clicks on continue, the systems inform the user that the uploaded data will be automatically erased when the user closes the main dashboard.

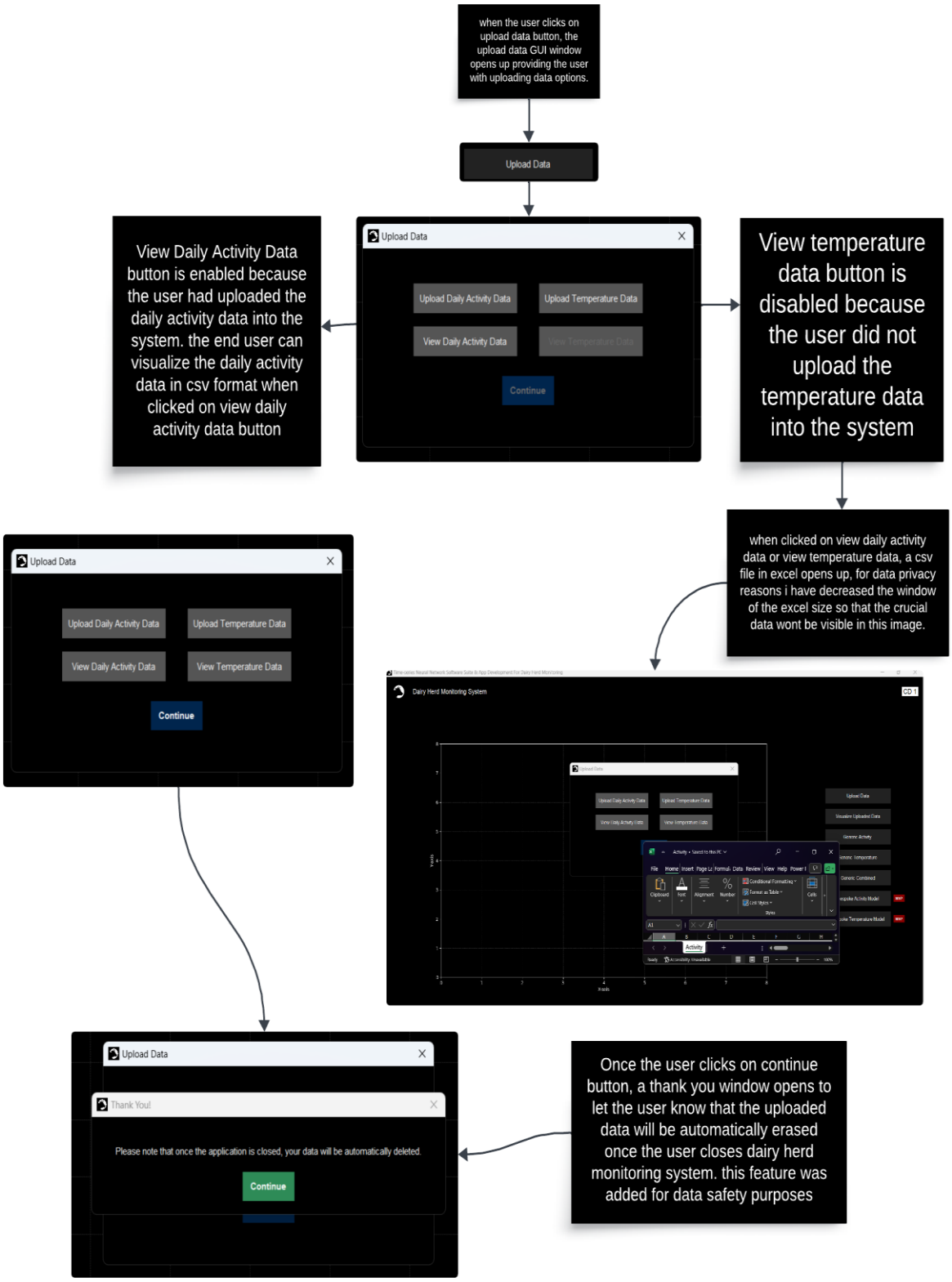


Figure 4.1: Upload Data button user interface

4.3.2.2 Data Visualization

Once the user uploads the datasets, they can also visualize them in graphical form to see the trends for both raw data and pre-processed data, providing the end user to ensure data quality. The user can choose to visualize the entire herds activity or temperature data or they can choose to visualize individual cows activity or temperature data. Visualizing the pre-processed data gives them the clear picture of the input given to the machine learning models.

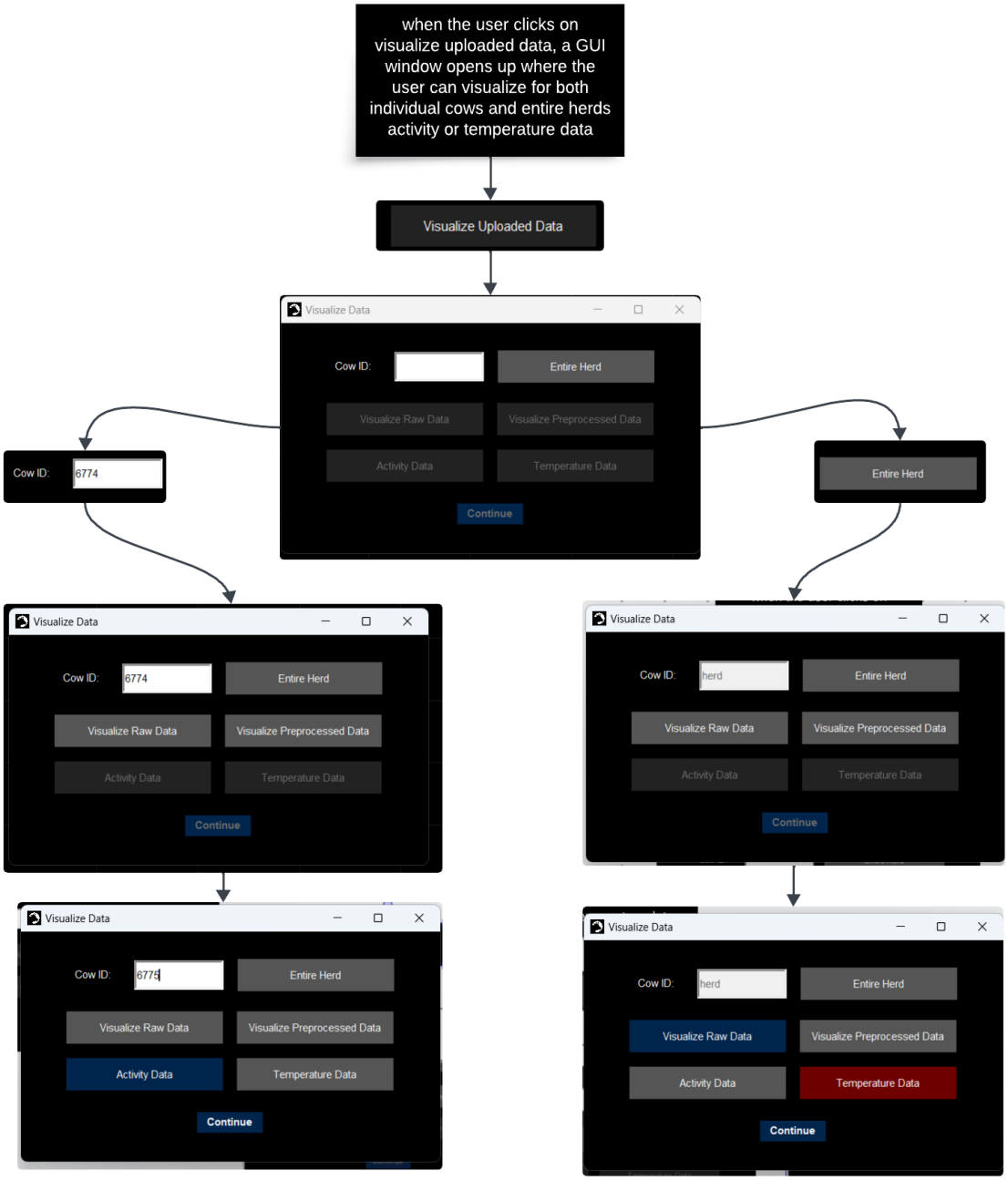


Figure 4.2:Visualize uploaded data user interface

- Visualizing Raw Data: Upon selecting either the individual cows by providing the cow id or by selecting the entire herd option, the user can select the raw data button and then select either for activity data or temperature data to visualize

the trends. the graphs have tooltip functionality that renders when the user hovers on the datapoints.

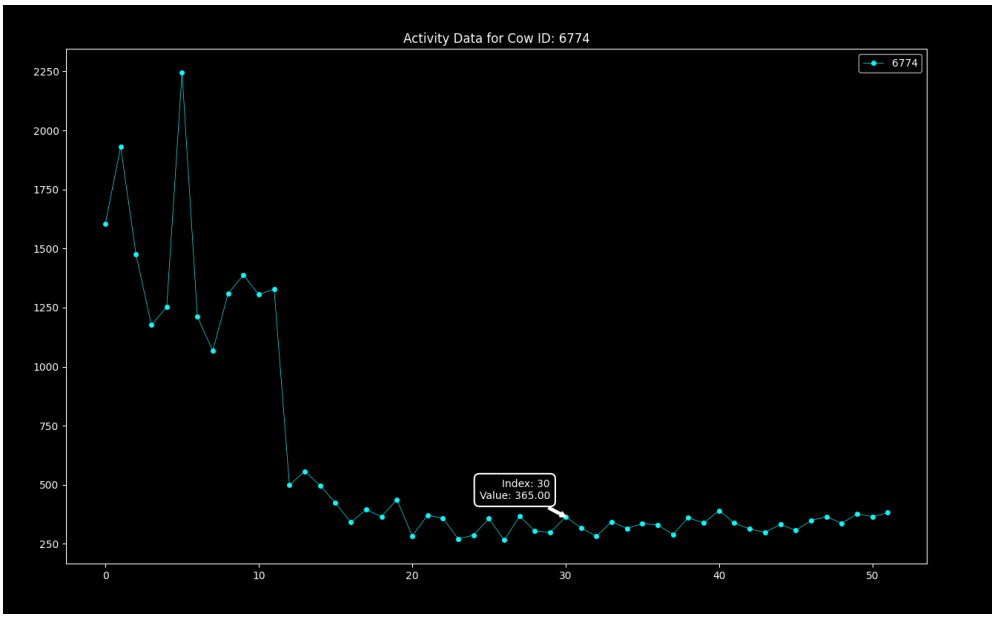


Figure 4.3: shows how the graph is generated when the user selects cow id input field to visualize individual cow's data for activity dataset.

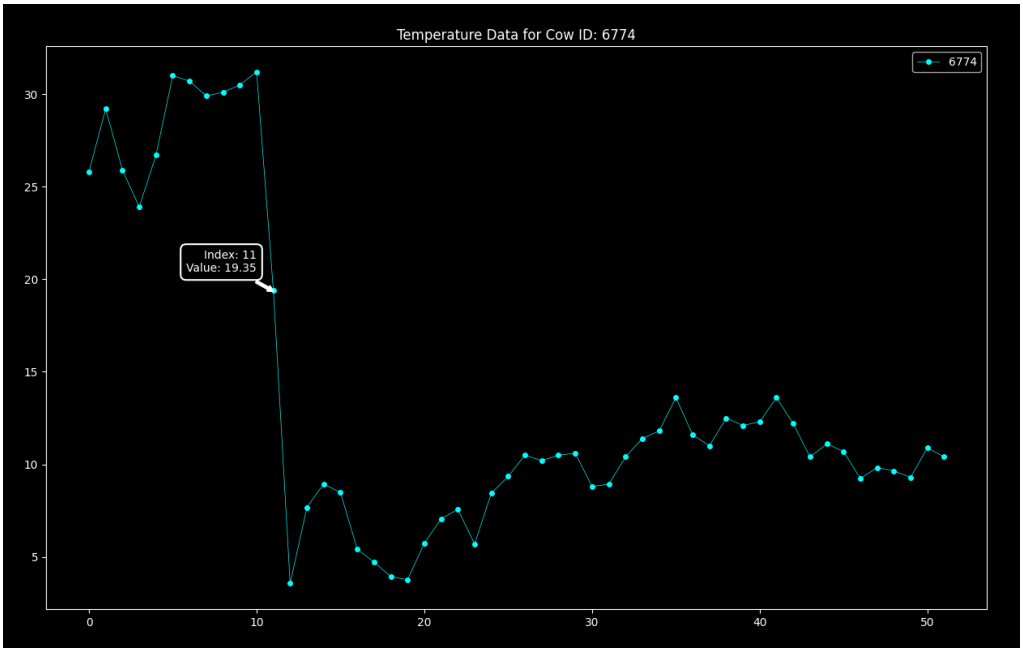


Figure 4.4: shows how the graph is generated when the user selects cow id input field to visualize individual cow's data for temperature dataset

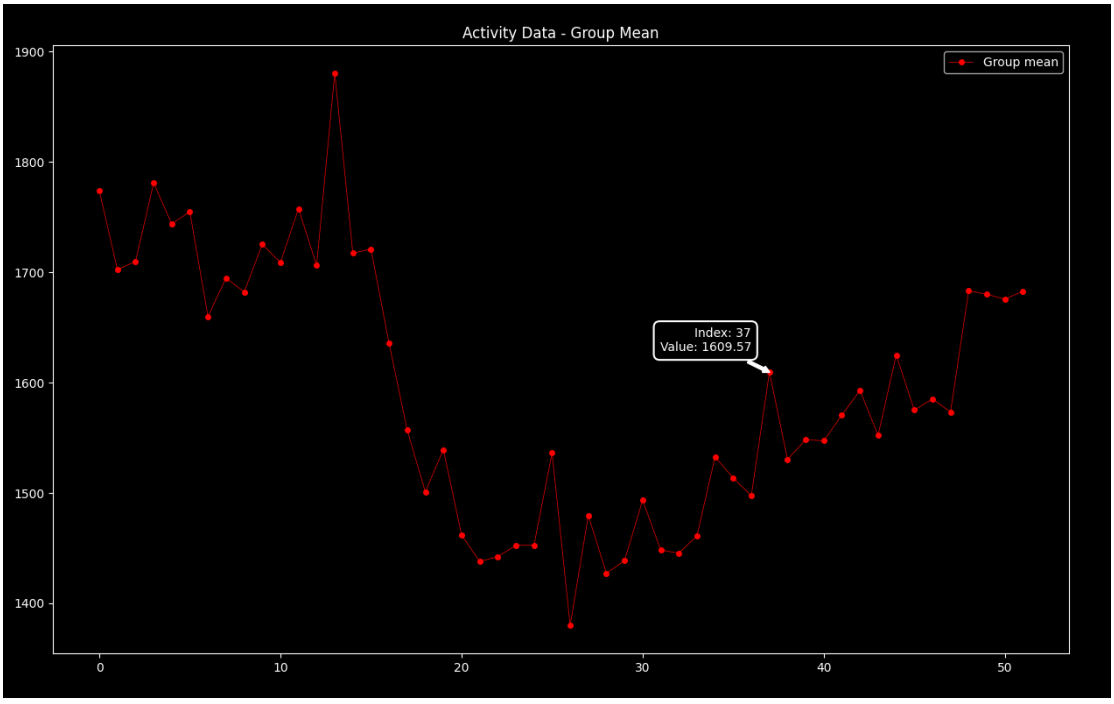


Figure 4.5: shows how the graph is generated when the user selects entre herd button to visualize entire dairy herds activity data

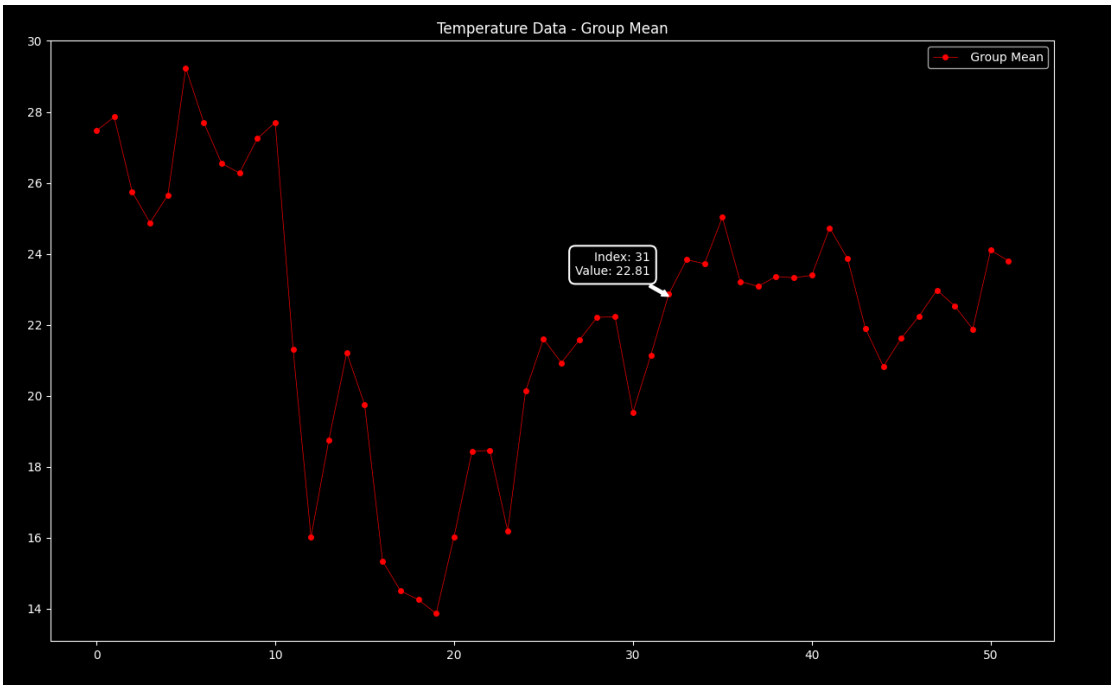


Figure 4.6: shows how the graph is generated when the user selects entre herd button to visualize entire dairy herds temperature data

- **Visualizing Pre-processed Data**: Upon selecting either the individual cows by providing the cow id or by selecting the entire herd option, the user can choose the pre-processed data button and select either for activity or temperature data to visually see the pre processed data that will be given as an input to the machine learning models. The graphs have tooltips that will appear automatically when the user hovers on data points.

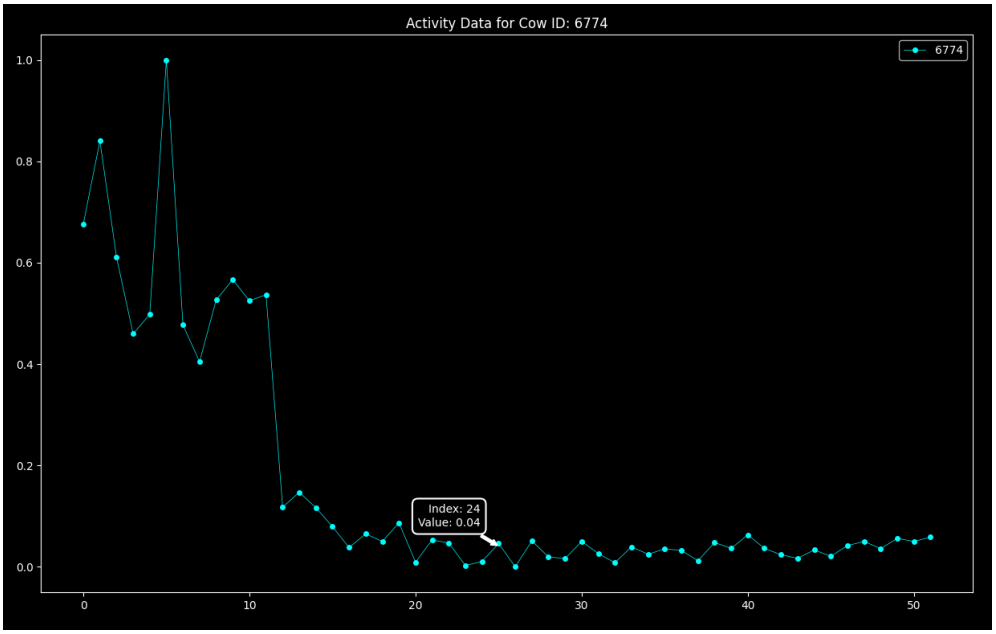


Figure 4.7: shows how the graph is generated when the user selects cow id input field to visualize individual cow's data for pre-processed activity dataset.

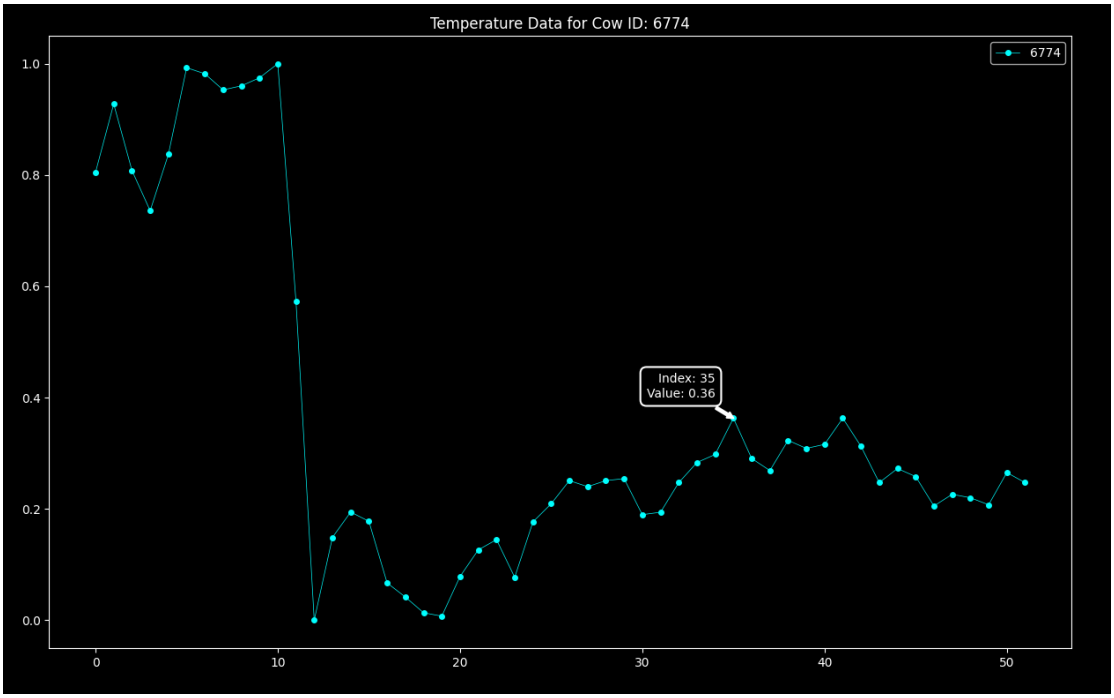


Figure 4.8: shows how the graph is generated when the user selects cow id input field to visualize individual cow's data for pre-processed temperature dataset

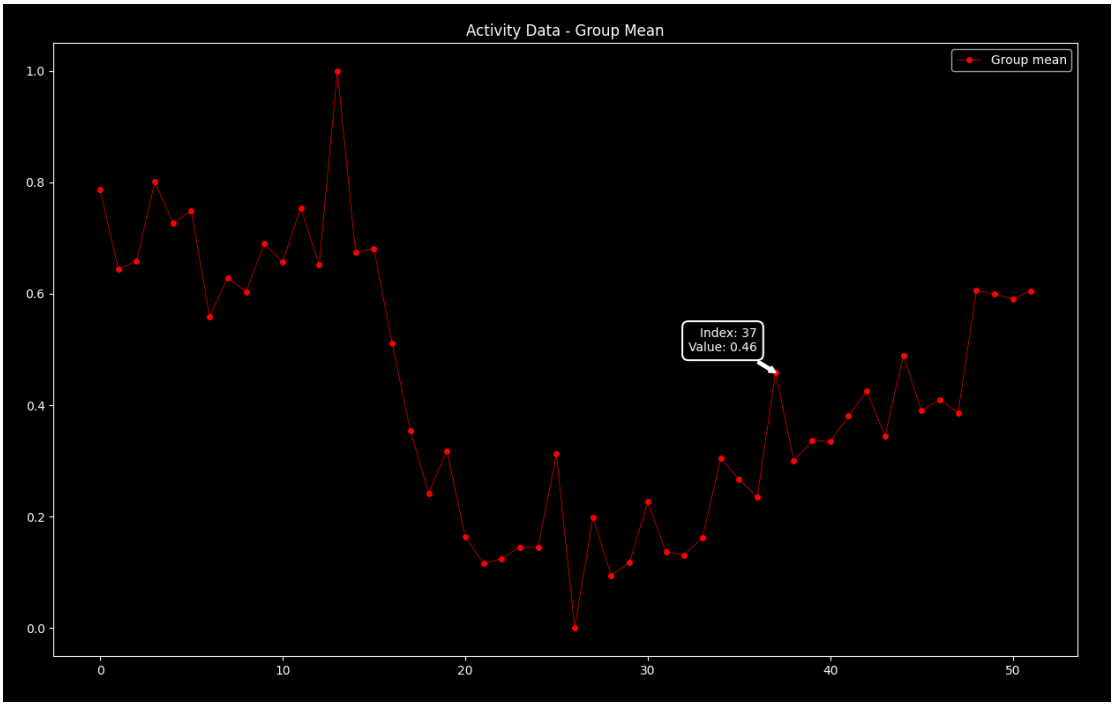


Figure 4.9: shows how the graph is generated when the user selects entre herd button to visualize entire dairy herds pre-processed activity data

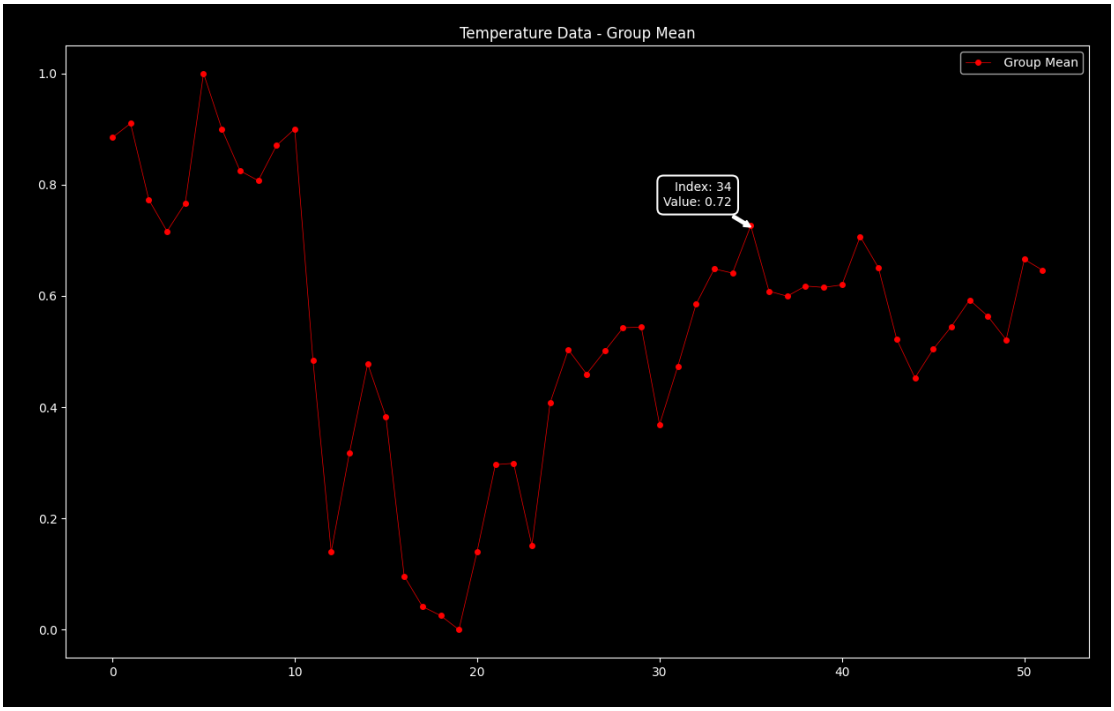


Figure 4.10: shows how the graph is generated when the user selects entre herd button to visualize entire dairy herds pre-processed temperature data

4.3.3 Interaction with Machine Learning Models

The most important feature of the software is the user interaction with machine learning models. Users can train new models, reload the existing ones, or they can save their trained models and can customize the folder names of their trained models. The user interaction with these models is made simple with a step-by-step approach. These features are applicable for generic models as they were the main aim and objectives of the project.

4.3.3.1 Training and Reloading Models

The software allows the user to either train new models or reload saved models. This flexibility in the system ensures that it could possibly adapt to the evolving needs of the dairy herd.

- **Training Models:** when user chooses any of the generic architecture buttons, such as generic activity, generic temperature and generic combined, a window appears with option if the user wants to train or reload a model. Upon selecting the train a model, all the required steps are processed on the backend and two graphs will be generated, one graph will show the forecast and another graph will show the health alert.

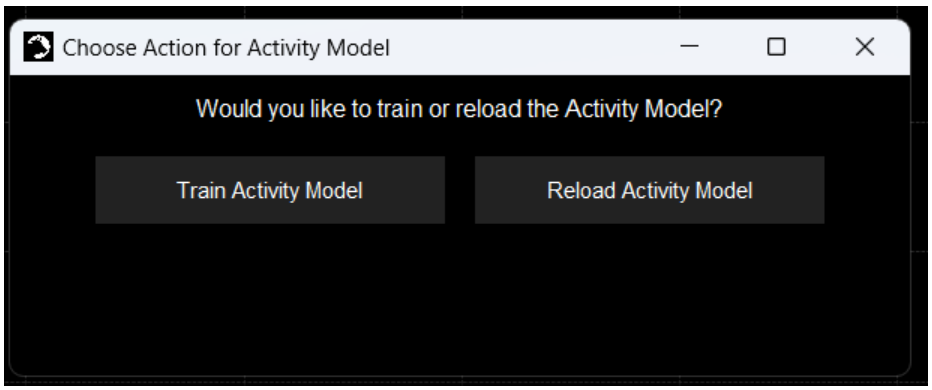


Figure 4.11: shows how the system allows the user to train an activity generic architecture when they click on generic activity button.

- **Reloading Models**: if the models are already trained and saved, users can click on reload activity model to reload models of their choice stored in specific folders. When the user clicks on Reload button, a new GUI window will appear that shows all the saved models, upon selecting one, the models are reloaded and visualized on the main dashboard.

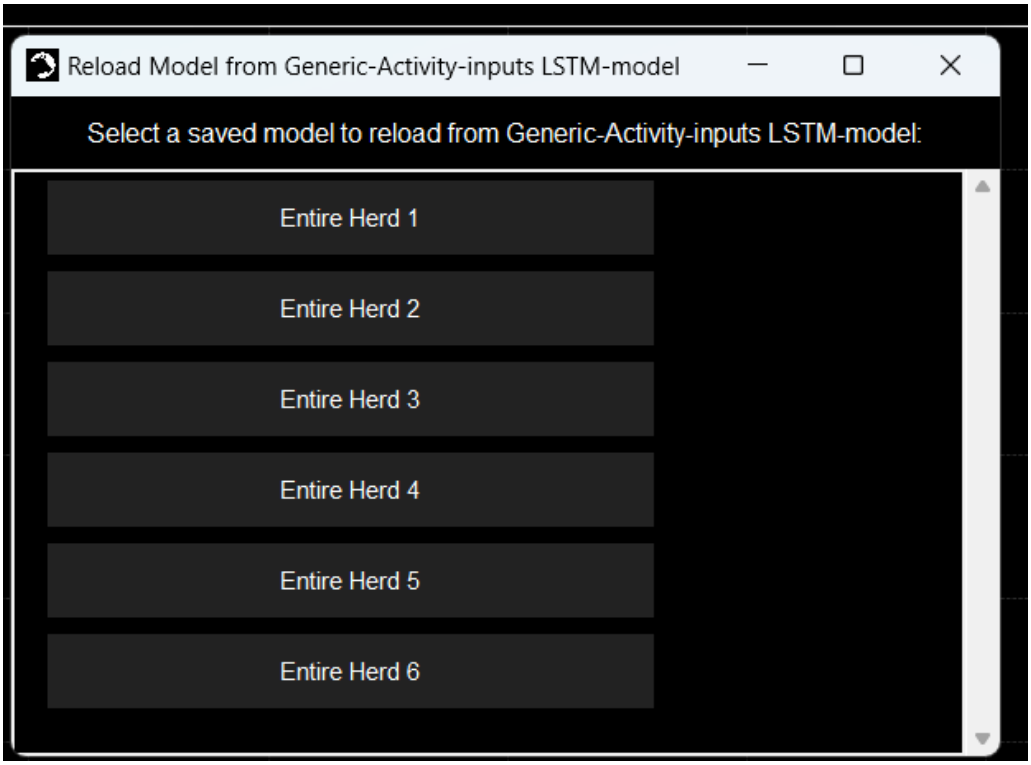


Figure 4.12: shows how the system allows the user to select a trained model from the specific folder where the models are saved after training.

4.3.3.2 Saving Models

After training machine learning models, the users can save it for future use. The model saving interface allows the users to name and organize their saved models in ease. This feature is only implemented for generic models.

- **Custom Folder Names:** when saving the models, user can assign a custom folder name, and this folder with a custom name will be saved inside a specific folder. For example, if the user saves a generic Activity models, and customize their name as test-one, then this will be saved inside “Generic-Activity-Inputs LSTM-model”.

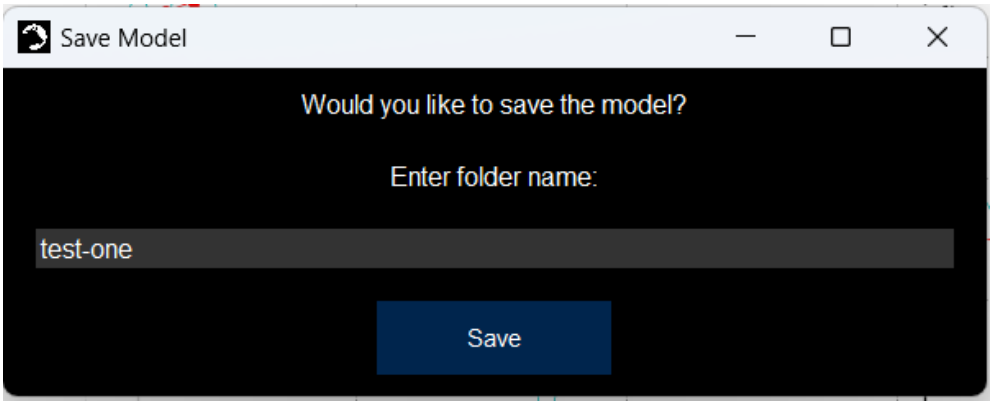


Figure 4.13: shows how the system allows the user to save the trained models where the user can assign a custom folder name when saving models.

4.3.4 Bespoke Models and Customization

The system supports bespoke models, but they are a prototype, a minimum viable product (MVP). These models only forecast the activity or temperature of individual cows. This feature was built additionally to target individual cows that might require careful monitoring if they tend to have serious health concerns.

4.3.4.1 Cow ID Input

Users can input a specific cow id to analyze and visualize forecast for their activity or temperature for individual cows. This allows the system to provide visualization for individual cows with their specific data from the dataset.

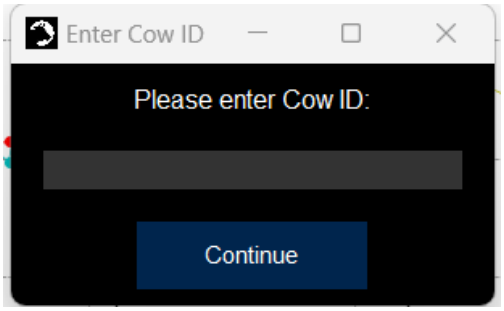


Figure 4.14: shows how the system allows the user to input a cow id to visualize trends and forecasts of individual cows for both their activity and temperature.

4.3.4.2 Bespoke Activity and Temperature Models

With bespoke model's users can visualize forecasts and monitor activity and temperature of individual cows, this feature was added to gain insights into the health of the individual cows, adding in better dairy herd management.

4.3.5 Data Handling and Error Management

Handling datasets is a crucial part of the system because they are involved in every functionality of the main dashboard. To enhance smooth user experience from providing datasets to visualizing results, implementing error management in each of the functionality of the dashboard is an important step. Users must be promptly alerted of their usage in every functionality of the dashboard.

- **Error Notifications:** the system performs checks on each step taken by the user, for instance, if the user doesn't provide the required data fields while uploading the data, the system alerts the user to correct the issue before continuing to the next step.

4.4 Conclusion

The GUI of dairy herd monitoring system was designed and developed with simplicity and proper functionality with error management in mind. Tkinter library, with its ease to use of cross-platform capabilities, was an ideal choice to develop the frontend of the dairy herd monitoring. Tkinter seamlessly interacts with the machine learning models. The development is simple yet focuses on accessibility and usability enhancing end user experience.

Chapter 5 – AI/ML Implementation

5.1 Introduction

The Dairy Herd Monitoring System is a time-series neural network software suite developed for the early detection of potential health issues in dairy herd. To predict the health status and provide alerts, the dairy herd monitoring system processes the provided activity and temperature data by using both regression and classification models. The software includes generic models for overall dairy herd monitoring and bespoke models as a prototype for individual cow health predictions. These models goal is to provide an early detection system for health issues, allowing the user to take any required action based on the herd. This chapter also discuss evaluation metrics such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Confusion Matrix, and classification reports. Additionally, the chapter also covers how the software

provides features like model saving, reloading and report generation backend process works.

5.2 Data Preprocessing

data preprocessing is a crucial step required to ensure the performance and accuracy of machine learning models. In dairy herd monitoring system software, both the activity data and temperature data uploaded by the end-user and pre-processed before they are given as an input to train the LSTM models. As highlighted by Brownlee (2020), scaling data is important for neural networks prevent large input values slowing down learning or preventing the network from effectively learning the problem. This is an important step because it ensures that the data is cleaned and normalized which is critical for time-series models implemented in the dairy herd monitoring system.

5.2.1 Data Cleaning

The raw sensor data collected from the dairy herd may have missing values when there is an issue with data transmission or sensor readings. During the preprocessing stage, data collected from a CSV file may contain extra spaces in column names, missing values or outliers. Therefore, the program first strips any extra spaces in the dataset columns using the strip() method. Next the code drops irrelevant columns if they exist in the dataset using drop() method. If there are any missing values in the dataset, the code using forward fill technique ffill() to fill the missing values in the dataset in the data cleaning stage.

5.2.2 Normalization

Normalization is a crucial step in data preprocessing, specially for machine learning models like LSTMs to improve model accuracy and performance. As Brownlee (2020) explains, normalization rescales the data from its original range to a standard range between 0 and 1. This is made possible using the MinMaxScalar from the scikit-learn library. To estimate the minimum and maximum values, the MinMaxScalar first fits the scalar to an available training data., then the MinMaxScalar transforms the dataset using the transform() function to scale the values between 0 and 1. Finally, the same scale of data can be applied for forecast while making the predictions. The datasets that are normalized allows the LSTM models to efficiently learn patterns in the data.

5.3 Regression Models

A regression model is a statistical method used to predict a continuous output variable based on individual or multiple independent parameters, where the aim is to form a relationship between these variables and the output, allowing the model to predict future values (Yildiz et al., 2017). The equation for the general form of regression model is

$$y = a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + e$$

Here, a_1 to a_n are the coefficients corresponding to each influence parameter $x_1$to x_n, where e is the associated error term and y is the actual output.

5.4 Classification Models

A classification model is a machine learning model used to predict a class label for input data, where it involves assigning a class label based on learned examples from a dataset (Brownlee, 2020). The classification model is trained on time series datasets and uses

patterns from the features to make predictions on new unseen data. General algorithms for classification include decision trees, logistic regression and neural networks, each suitable for different types of classification tasks.

5.5 Neural Network Models for Combined Classification and Regression

Neural network models can be developed to make both classification and regression predictions from the same input data, solving requirements that need both numeric values and class labels simultaneously (Brownlee, 2021). Instead of developing separate models for each task, a more effective approach is to develop a unified model capable of handling both tasks.

5.6 Softmax Classifier

A softmax classifier is a supervised learning model commonly used in deep learning for multi-classification tasks, because the softmax function converts the outputs of a model into probabilities, ensuring that the overall probability across all classes sums to one (Khan, 2023). It is specifically suited for problems where an input needs to be classified into one of the several categories.

5.7 Root Mean Squared Error (RMSE):

RMSE calculates the square root of average squared differences between actual and predicted values. Due to its squared component, RMSE is sensitive to the outliers which makes it a good indicator. Brownlee (2021), highlights that RMSE is the square root of the Mean Squared Error (MSE), making it more accessible because it shares the same units as the target variable.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

y_i, is the actual value and $\hat{y}_i$is the predicted value. If the RMSE value is lower, that indicates that the model is performing better.

5.8 Mean Absolute Error (MAE):

MAE calculates the average absolute difference between the actual and predicted values. This shows how well the model is performing. According to Schneider and Xhafa (2022), the main advantage of MAE over evaluation metrics such as RMSE and MSE is that it does not correct large errors more significantly but the errors in MAE increases linearly with the magnitude of the error, making it more instinctive for interpreting errors.

It is calculated by the formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|^2$$

Mean Absolute Percentage Error (MAPE) in Regression Models

MAPE is an evaluation metric used for evaluating the accuracy of regression models. According to Myttenaere et al. (2016), MAPE is calculated as the average of absolute percentage differences between actual and predicted values. This ability makes MAPE

ideal for applications were relative changes matter more than absolute ones, such as forecasting.

Formula for MAPE is:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

Where, y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the number of observations.

5.9 Confusion Matrix for classification models

For Classification models, a confusion matrix is a crucial evaluation metric because it provides a detailed analysis of how the models predictions compare with the actual class labels. There are four key components of the matrix,

- True Positives (TP)
- False Positives (FP)
- True Negatives (TN)
- False Negatives (FN)

These components of the matrix will provide a detailed view of the model is performing across each class. Patro and Patra (2015) highlights, that the confusion matrix provides for a systematic evaluation of classification accuracy where the matrix is constructed by comparing the predicted vs actual values by placing the correct predictions along the diagonal.

For multi-class classification, the confusion matrix is displayed in a 3 x 3 matrix format.

	Predicted Green	Predicted Orange	Predicted Red
Actual Green	TP(Green)	FP(Orange)	FP(Red)
Actual Orange	FN(Green)	TP(Orange)	FP(Red)
Actual Red	FN(Green)	FN(Orange)	TP(Red)

Table 5.1

Patro and Patra (2015) also explain that confusion matrices are particularly useful when dealing with multi-class classification because they allow classification accuracy across multiple classes. For example, in a three-class scenario, the confusion matrix calculates errors for each class separately and provides an overall understanding of how the model performs.

Code Snippet to Generate Confusion Matrix

```
from sklearn.metrics import confusion_matrix,
classification_report


# true_labels and pred_labels contain the actual and predicted
health states respectively
cm = confusion_matrix(true_labels, pred_labels, labels=[0, 1,
2])  # 0: Green, 1: Orange, 2: Red


# Generating the classification report
target_names = ["Green Alert", "Orange Alert", "Red Alert"]
```

```
report = classification_report(true_labels, pred_labels,
target_names=target_names, zero_division=0)


print("Confusion Matrix:\n", cm)
print("\nClassification Report:\n", report)
```

5.10 Classification Reports

The classification report is a summary of the performance made by the classification models.it includes evaluation metrics such as Precision, Recall, F1-Score, and Support for each class. In multi-class classification tasks, these metrics are particularly useful. When the dairy herd is categorized into 3 different health states.

Classification Report Metrics:

- Precision: precision metric evaluates how many cows predicted to be in a certain health state were actually in that state.
- Recall: Recall metric measures how correctly did the model identify the number of cows that are truly in a certain state.
- F1-Score: A mean score of both precision and recall to provide a balanced measure of model performance.
- Support: evaluates the actual instances of each class in the dataset.

Formula for F1-Score:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

For accurate model evaluation, metrics such as precision, recall, and F1-Score are crucial and these metrics can be calculated using scikit-learn metrics for deep learning models (Brownlee, 2022). The classification report implemented in the dairy herd monitoring system, provides the user with model performance result on each state. For example, if the model has a higher precision for a red alert state, it means it is likely correct when the system predicts a cow in a critical state. A higher F1-score across all classes indicates a balanced model that is capable of correctly identifying herds each health state with higher accuracy.

Code snippet to generate classification report

```
from sklearn.metrics import classification_report


# true_labels and pred_labels contain the actual and predicted
health states
report = classification_report(true_labels, pred_labels,
target_names=["Green Alert", "Orange Alert", "Red Alert"])
print(report)
```

5.11 Generic Models: Regression and Classification

The generic system has 3 types of features in the dairy herd monitoring system

- **Generic Activity:** developed to visualize results based on activity data.
- **Generic Temperature:** developed to visualize results based on temperature data.
- **Generic Combined:** developed to visualize results based on both activity and temperature data given as inputs to the machine learning models.

The generic system incorporates two types of generic models in generic activity, generic temperature and generic combined button features.

- **Generic Regression Model:** This model is designed to predict and forecast continues health score which could be for generic activity, generic temperature or generic combined.
- **Generic Classification Model:** this model is designed to classify cows' health alerts for generic activity, generic temperature and generic combined.

Both the models use vanilla LSTM networks, which are particularly suitable for handling time-series data. These models process activity and temperature data to predict the health outcomes of the entire herd.

5.11.1 Generic Regression Model

The generic regression model implemented in the dairy herd monitoring system is developed using LSTM networks, which are specifically effective while capturing long term dependencies in time series data. As noted by Rather et al. (2021), to predict the continues values based on temporal data, the regression model has been

implemented on LSTM deep neural networks, which highlights the suitability of this architecture. In dairy herd monitoring system, the vanilla LSTM models process activity and temperature data to provide continues health score. The same principals can be applied in generic model architecture. Because the cow's health conditions can fluctuate over time, the system requires an architecture that can predict this changes by tracking them. Informed decisions can be taken now based on the predictions generated by the LSTM model allowing the end user to take better decisions related to cow health and dairy management.

To ensure that the temporarily dependencies are properly captured, dairy herds data will be normalized before training the model. To measure the accuracy of the predictions, evaluation metrics such as RMSE and MAE are used. These provides the end-user to visualize the accuracy aiding in their decision making based on the model's predictions, supporting end users in early detection of health risks in dairy herd.

5.11.1.1 Activity Regression Model Architecture

The Generic Activity model predicts the future activity of the dairy herd based on the past data by processing the normalized data on vanilla LSTM architecture.

- **Input Layer**

  In LSTMs, Ryan T. J.J., (2020) highlights, the input layer is responsible for receiving the sequential data The input given to the model is a time-series dataset consisting of multiple time steps of activity data and it is organized into a 3D array with the following dimensions below

Batch Size: batch size is the number of sequences that are processed together

Sequence Length: sequence length is number of past time stamps the model looks at. In this architecture it takes first 5 days as the batch size,

Features: features is the number of features that will be analysed based on the dataset. In this case it considers all the cows present in the dataset.

Code snippet: input_shape=(X_train.shape[1], X_train.shape[2])

- **LSTM Layer**

  The LSTM layer is most crucial part of the activity model architecture. LSTM models will learn the temporaril dependencies in the activity data (Ryan T.J.J.,2020). Following below are the parts of LSTM layer that explains how it is designed to predict time series data.

  Units: units are the memory capacity which allows the model to capture complex temporal patterns (Ryan T.J.J.,2020). For dairy herd monitoring system, 100 units were set to capture complex temporal patterns.

  Activation function: a function named Rectified Linear Unit (ReLU) is used to help the model learn by passing only the positive values for model training.

  Return Sequence: in order to make the LSTM output only final hidden state for each sequence, return sequence is set to False, hence summarizing the entire time series (Ryan T.J.J.,2020).

  Code snippet: LSTM(100, activation='relu', return_sequences=False)

- **Dropout Layer**

  To prevent the model from overfitting, a dropout layer is used. A dropout layer randomly disables neuros during training, reduicing overfitting by preventing

reliance on specific features and improving generalization (Mudigonda, 2021).Overfitting can occur when the model learns to much from the training data. This could possibly lead to perform poorly on the test data. In order to perform better on the unseen data, the model needs to generalize better. Adding a dropout layer can prevent the model from overfitting leading the model to generalize better.

The dropout layer implemented in the generic activity model architecture will randomly ignore 20% of the neurons during each training step, this forces the model to generalize better on the training data.

Code snippet: Dropout(0.2)

- **Dense Layer**

  Dense layer is a fully connected layer which is used to take the output from the LSTM and reduce it to match the number of features that is meant to be predicted. Moawad (2019), highlights that a dense layer in a neural network applies a linear transformation to the input and passes the result through a non-linear activation function to introduce complexity in the model (Moawad, 2019). The software has 28 dense layers.

  Code Snippet: Dense(28)

- **Model Compilation**

  The model is further designed to minimize the squared difference between actual and predicted values. The model is compiled to use Adam Optimizer and Mean Squared Error (MSE) loss function to minimize the difference between actual and predicted values. Adam optimizer dense is an optimization techniquie that is used to update network weights iteratively based on training

data, in place of traditional gradient descent process (Brownlee, 2021). Adam Optimizer is implemented because it is efficient in handling sparse gradients and MSE is implemented because it is a common loss function in regression tasks. This compilation will minimize the squared difference between actual and predicted values.

Code snippet: model.compile(optimizer='adam', loss='mean_squared_error')

5.11.1.2 Temperature Regression Model Architecture

The temperature model architecture follows the same model implementation as the activity model architecture, but instead of predicting activity values, the temperature model architecture is designed to predict temperature values.

- **Input Layer**

  The input to this model is structured similarly to the activity model architecture, using batch size, sequence length and features similar to the activity model architecture. however, historic temperature data for the cows are used as input.

- **LSTM layer**

  The LSTM layer process the temperature data similar to how the activity model architecture process the activity data. The LSTM captures the temporal dependencies in the cow's temperature values over time.

- **Dropout Layer**

  Dropout layer is used in temperature model architecture to prevent overfitting by randomly ignoring 20% of the neurons during each training step, similar to how the activity model architecture uses the drop out layer.

- **Dense Layer**

  The dense layer decreases the output of the LSTM to 28 values, each represents the predicted temperature of each data point of the original temperature data.

- **Model Compilation**

  Just like the activity model architecture, the temperature model architecture compiles Adam Optimizer and MSE loss function to minimize the difference between actual and predicted values.

5.11.1.3 Combined Model Architecture

the combined model architecture is designed to integrate both activity data and temperature data to visualize an overall prediction of dairy herd.

- **Input Layer**

  The combined model architecure takes in two inputs

  Activity Input: A sequence of historical activity data

  Temperature Input: A sequence of historical temperature data

  Each input will be processed separately by its respective LSTM layer.

  Code Snippet:

  temp_input = Input(shape=(sequence_length, len(temperature_columns)))

  activity_input = Input(shape=(sequence_length, len(activity_columns)))

- **LSTM Layers**

  The combined model uses two separate LSTM layers, one for the activity data and another for the temperature data. Both layers learn the temporal dependencies separately.

Activity LSTM Layer: this layer process the activity data

Temperature LSTM Layer: this layer process the temperature data

Code snippet:

```
temp_lstm = LSTM(100, activation='relu',
return_sequences=False)(temp_input)
activity_lstm = LSTM(100, activation='relu',
return_sequences=False)(activity_input)
```

- **Concatenation Layer**

  After the LSTM layers process the activity and temperature data separately, the outputs of the two LSTM layers are concatenated. This allows the combined model architecture to learn relationships between activity and temperature data by combining both data streams.

  Code snippet: `combined = concatenate([temp_lstm, activity_lstm])`

- **Dense Layer**

  For the model to learn more complex relationships between activity and temperature data, the combined output is passed through a dense layer with 50 neurons. This will help in extracting high level features that will be useful for prediction.

  Code snippet: `dense_layer = Dense(50, activation='relu')(combined)`

- **Dropout Layer**

Similar to the activity and temperature model architecture uses a dropout layer to prevent overfitting, 20% of the neurons are randomly ignored during training to prevent overfitting in the combined model architecture.

Code snippet: Dropout(0.2)

- **Output Layer**

  Based on both activity and temperature data, a single continuous value is generated representing the overall health score of the herd is generated as the final output of the combined model.

  Code snippet: Dense(1)(dropout_layer)

- **Model Compilation**

  The combined model is further compiled using an Adam optimizer and MSE loss function, to ensure the model learns to minimize the difference between the actual and predicted health scores similar to the previous model architectures.

5.11.2 Generic Classification Model

LSTM classification models are widely used for their ability to capture long-term dependencies in time-series data and perform sequential data tasks. They learn patterns in sequences, allowing them to classify movements without manual feature engineering (Brownlee, 2020). The LSTM classification models architecture consists of an LSTM layer, dropout to prevent overfitting, and a dense layer to predict probabilities for different activities (Brownlee, 2020). Similarly, the generic classification model used in dairy herd monitoring system to predict health alerts also uses time-series data to classify health alerts into three alerts. Green Alert indicating

healthy State, Orange Alert indicating Risky state, and Red Alert indicating Critical States.

5.11.2.1 Activity Classification Model Architecture

The generic activity classification model is developed to process the historical activity data of the dairy her to predict health states.

- **Input Layer:** the input given to the activity classification model consists of time-series activity data, structured similarly to the input layer of activity regression model.

- **LSTM Layer:** similar to the activity regression models LSTM layer, the activity classification models LSTM layer captures temporal dependencies in activity data over time, with 100 units and ReLU activation.
- **Dropout Layer:** to prevent overfitting, 20% of the neurons are randomly ignored during model training similar to the activity regression models dropout layer.
- **Dense Layer**: The dense layer reduces the output to 3 values after the LSTM layer processes the time-series data. Each output values representing one of the three states, green, orange or red. The next step is to pass the output of the dense layer to the softmax layer.

  Code snippet: Dense(3)
- **Softmax Output Layer:** softmax activation function is used in the final layer of the activity classification model. This function converts the dense layer output into probabilities for each of the three states of health alerts. To

effectively make the probability distribution, the softmax layer ensures that the sum of all outputs is equal to 1.

Code snippet: Dense(3, activation='softmax')(dropout_layer)

- **Model Compilation:** Adam optimizer and categorical cross-entropy loss function is used to compile the model. For multi-class classification problems, categorical cross entropy is highly effective, where the aim is to compare the predicted class probabilities with the actual class labels.

  Code snippet: model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

5.11.2.2 Temperature Classification Model Architecture

Temperature classification model architecture is designed similarly to the activity classification model architecture, but instead of processing activity data, the model processes historical temperature data.

- **Input Layer**: the input given to the temperature classification model, processes historical temperature data, and the structure is similar to the activity classification models input layer.
- **LSTM Layer**: The LSTM layer captures the temporal dependencies in the herd temperature patterns by processing the historical temperature data.
- **Dropout Layer:** To prevent overfitting, the dropout layer ignores 20% of the neurons similar to the dropout layer of activity classification model.
- **Dense Layer:** LSTM outputs are reduced to 3 values representing the health states by the dense layer.

- **Softmax Output Layer:** similar to the activity classification model, the softmax layer outputs probabilities of three of the health states which are green alert, orange alert and red alert.
- **Model Compilation:** similar to the activity classification model, the model is compiled using Adam optimizer and categorical cross-entropy function to compare the predicted class probabilities and actual class labels.

5.11.2.3 Combined Classification Model Architecture

To classify the health states of the entire herd, the combined classification model integrates both the activity and temperature data. Similar to the combined generic model architecture, the combined classification model architecture processes two separate inputs data before combining them to generate a prediction.

- **Input Layer:** the combined model is designed to handle two input layers, one for activity data and another for temperature data. The design of the input layer is similar to the combined regression model, where both inputs are time series datasets.
- **LSTM layers:** the combined model uses two separate input layers to process both the time series datasets to learn the temporal dependencies similar to combined regression model.
- **Concatenation Layer:** after the LSTM layers have processed their inputs individually, both the output layers are concatenated. Similarly to the combined regression model architecture, the concatenation layer allows the model to learn relationship between the activity and temperature data by combining the outputs.

Code snippet: combined = concatenate([temp_lstm, activity_lstm])

- **Dense Layer**: dense layer with 50 neurons will receive the combined data in this step. To help the model learn the complex relationships between the activity and temperature data, the dense layer is implemented into the combined classification model.

  Code snippet = dense_layer = Dense(50, activation='relu')(combined)

- **Softmax Output Layer:** the final dense layer is further linked to the softmax output layer. This layer is programmed to generate the probability distribution for the three health states.

  Code snippet: Dense(3, activation='softmax')(dropout_layer)

- **Model Compilation:** similar to the individual classification models model compilation feature, the combined model also compiles using adam optimizer and categorical cross-entropy loss function.

5.12 Bespoke Regression Model

This section explains how bespoke regression models are developed for the dairy herd monitoring system. There are two bespoke regression models that were developed for the dairy herd. They are Bespoke Activity Model and Bespoke Temperature Model.
To further develop the dairy herd monitoring system, these bespoke regression models were developed as a prototype. Although they were not initially the main objectives of the project, they serve as a minimum viable product (MVPs). These models are

specifically designed to monitor individual cows, showing the advancements of the dairy herd monitoring.

5.12.1 Bespoke Activity Model

The bespoke activity model predicts the individual cow's activity based on the historic activity of the individual cows. This model is focused on specific cow's activity allowing to create more accurate predictions.

Model Architecture and Training

The model is developed by LSTM architecture, which is specifically useful for time series data like activity data of the dairy herd. This architecture includes LSTM layers with dropout to prevent overfitting while model training. Dense layers are further implemented to make the final predictions.

```
model = Sequential()
model.add(LSTM(150, return_sequences=True, input_shape=(1,
X_train.shape[2])))
model.add(Dropout(0.3))
model.add(LSTM(150, return_sequences=False))
model.add(Dropout(0.3))
model.add(Dense(50))
model.add(Dense(1))
```

Adam optimizer and MSE as the loss function is used to train the model.

Predictions and Evaluation

Once the bespoke activity model is trained, the model predicts the activity levels on both training and test sets, these predictions are transformed back to the original scale and model performance is evaluated using Mean Absolute Percentage Error (MAPE). The model also generates a 15-day forecast predicting the future activity levels of the individual cows.

```
train_predictions =
scaler_y.inverse_transform(model.predict(X_train))
test_predictions =
scaler_y.inverse_transform(model.predict(X_test))
train_accuracy = 100 - np.mean(np.abs((y_train_actual -
train_predictions) / y_train_actual)) * 100
```

5.12.2 Bespoke Temperature Model

The Bespoke temperature model is similar to the bespoke activity model, but it focuses on predicting temperature patterns based on the historical temperature data. This bespoke temperature model learns each cow's fluctuations and predicts future temperatures of the cow.

Model Architecture and Training

The model is developed using an LSTM architecture, which is helpful for time-series data, such as dairy herd temperature data. This model architecture has the similar model architecture as the bespoke activity model. This allows the architecture to learn temperature sequences and make future predictions.

```
model = Sequential()
model.add(LSTM(150, return_sequences=True, input_shape=(1,
X_train.shape[2])))
model.add(Dropout(0.3))
model.add(LSTM(150, return_sequences=False))
model.add(Dropout(0.3))
model.add(Dense(50))
model.add(Dense(1))
```

over 200 epochs of training is conducted using the Adam optimizer and MSE loss function

Prediction and Evaluation

Similar to the bespoke activity model, the bespoke temperature model generates predictions on both activity and temperature datasets and also generates a forecast of 15 days, providing the end user to identify abnormal trends in the temperature of the individual cow earlier.

5.12.3 Final Alert System

To generate the final herd alert, the final alert system will utilize the results of the confusion matrix and classification report. The classifications of the entire herd is aggregated and the most common alert is selected to be the final alert for the entire dairy herd.

5.13 Conclusion

AI/ML implementation chapter explains on the use of LSTM models for both regression and classification tasks. the dairy herd monitoring system preprocesses the data and trains models based on the user selection to predict health alerts and shows a forecast of the herd. The trained models are evaluated with RMSE, MAE and classification reports showcasing how well the models are performing. In conclusion, both bespoke and generic models developed for this project provide high accuracy predictions that helps the dairy herd management.

Chapter – 6: System Integration

6.1 Introduction

This chapters explains how the machine learning models developed for the dairy herd are integrated into the frontend of the dairy monitoring system. To create a smooth interaction between the user interface and the AI/ML models was one of the crucial objectives of this project. Allowing the end user to upload their datasets, interact with the machine learning models and view the final results.

6.2 Frontend and Backend Interaction

The frontend of the dairy herd monitoring system was designed to be a user-friendly interface where the users can easily upload and visualize their activity and temperature datasets, initiate model training or reloading, visualize predictions and results. The frontend was developed using Tkinter library for its simplicity and its capability to

handle event driven programming. It allows the users to upload data, model interaction and to visualize results.

6.2.1 Buttons as Triggers

The dairy herd monitoring system consist of multiple buttons on the frontend where each of them linked to a backend functionality. When user clicks on a specific button on the main dashboard, the button acts a trigger that sends a request to the backend, these requests initiates types of functionalities such as data uploading, data preprocessing, model training or visualization of the results.

6.3 Integration of All Buttons

6.3.1 Upload Data Button

When the user selects the upload data button, the functionality opens a new window where the user can upload csv files containing dairy herds activity and temperature data. This triggers the open_upload_data() function. To manage the upload process, this open_upload_data() function uses the UploadDataApp class. Once the user uploads their datasets, the program automatically stores the datasets in a predefined directory on the system which is ready for preprocessing, visualization and model training tasks.

Backend Functionality:

- Upload_data.py handles the file management to ensure that the user uploaded dataset is saved properly.

- SAVE_DIR is used to store the uploaded data for further use like visualization or model training.

6.3.2 Visualize Uploaded Data Button

When user selects the visualize uploaded data button, the open_visualize_uploaded_data functionality is triggered. The function retrieves the saved uploaded data from the predefined directory on the system and uses Matplotlib to generate graphs for visualizations like that of activity data or temperature data for the entire herd or individual cows. The functionality allows the user to choose raw data or pre-processed data and upon selection, the visualizations are dynamically loaded onto the GUI's frame.

Backend Functionality:

- Visualize_uploaded_data.py handles logic for visualization and uses matplotlib for plotting graphs.
- FigureCanvasTkAgg is used to display the graphs within the Tkinter window.

6.3.3 Generic Activity Button

When the user clicks on the generic activity button, the system opens a new GUI window and prompts the user to either train a new model or reload an existing model. when the user chooses to train a new model, the backend preprocesses the uploaded activity data to build and train an LSTM model using TensorFlow.

Backend Functionality:

- Generic_Activity_Model.py is the backend code which is responsible for preprocessing the data, building and training the LSTM model on the pre-processed activity data.
- Once the model is trained, the trained model predicts future activity of the dairy herd and displays a graph on the main dashboard for visualization.
- If the user chooses to reload an existing model instead, a new GUI window is displayed with the list of all the pre-trained model folders so the user can select to reload and visualize.

6.3.4 Generic Temperature Button

Generic temperature button is developed similarly like the generic activity button. The generic temperature button opens a GUI window allowing user to train a new model or reload a pretrained model when clicked on the generic activity button. The backend then processes the temperature data, train a LSTM model and visualize results of the entire herd if the user selects train a new model or reloads and visualizes results of the entire herd if the user selects to reload a model.

Backend Functionality:

- Generic_Temperature_model.py is the backend code programmed to preprocess the temperature data, build and train the LSTM models.
- The results are displayed as a graph on the main dashboard, showing predictions and alerts for the entire herds temperature data.

6.3.5 Generic Combined Button

When the user clicks on the generic combined button, the user is provided options to choose if they want to train a new model or reload an existing model. if the user selects to train a new model, the generic combined architecture pre-processes both the datasets, trains the LSTM model on both the datasets independently, and then visualizes the results in graphs. If the user wants to reload an existing model, upon selection the saved models are reloaded and visualized on the main dashboard. These generic combined models are developed to provide a more detailed understanding of the herds overall health by implementing both activity and temperature data into the predictions.

Backend Functionality:

- Generic_Multi_model.py is the backend code programmed to take both activity data and temperature data as input to train the multi-layer LSTM models.
- Once the model is trained, the predictions and classifications are visualized by a unified graph using Matplotlib.

6.4 Bespoke Model Integration

The bespoke models are another approach in the dairy herd monitoring system that were specifically build to monitor individual cows. There are two types of models

6.4.1 Bespoke Activity Model Button

When the user selects bespoke activity model button, a new window opens up asking the user to input a specific cow id. When the user gives in the cow it and clicks continue, the Bespoke_Activity_Model.py code renders and calls the

start_bespoke_activity_graph() function is called. The backend processes the activity data by preprocessing, regression model training and visualizing the results in graph.

Backend Functionality:

- Bespoke_Activity_Model.py is the backend code developed to preprocess the activity data for the selected cow by selecting the individual column of the cow id in the dataset, training a LSTM regression model for that individual column of the activity dataset.
- The results are generated and visualized in a graph on the main dashboard showing the predictions of the individual cow's trends in activity.

6.4.2 Bespoke Temperature Model Button

The bespoke temperature model button's backend functionality is developed similarly as the bespoke activity model button where the end user clicks on the bespoke temperature model button, a window is prompted with an input field to enter the cow id. Once the user enters the cow id, genereate_bespoke_temp_model_graph() is called in the start_bespoke_temp_graph() and the results are visualized.

Backend Functionality:

- Bespoke_Temp_Model.py is the backend code programmed to preprocess the temperature data for the selected cow, where the generate_bespoke_temp_model_graph() function uses the temperature data by selecting an individual column of that specific cow id, it continues to train an LSTM regression model, and generate predictions.

- The graph is generated on the main dashboard, showing the temperature predictions of the individual cow id that the user wants to visualize.

6.5 Save Model Functionality for Generic Models

After user generates a model either from generic activity, generic temperature or generic combined. The software provides an option to the user where they can save the trained model. if the user decides to save, the save button triggers a prompt_save_model() function, this function saves the custom named folder where the user can customize the name of the model before clicking on saving. Once the prompt_save_model() is called, it saves the custom named folder in a predefined directory allowing the user to reload and visualize their custom saved models and also beneficial for future uses.

Backend Functionality:

- the trained generic models are saved using the TensorFlow model.save() method. This provides the user to reload saved models in the future.

6.6 GIF Integration for Process Indication

After user selects to generate a generic or a bespoke model, while the timeframe where the models are reloaded or retrained, in that time gap, a loading GIF animation is played on the main dashboard indicating that the system is training the model on the backend. This functionality is implemented using the display_gif_frame() function.

6.7 Closing the Application

When the user decides to close the dairy herd monitoring system main dashboard, and closes it, the datasets uploaded by the user which are saved in the predefined directory

in JSON format are automatically erased. This makes sure that the system won't store any datasets provided by the user while interacting with the software.

6.8 Site Map of the Dairy Herd Monitoring System

This figure below shows the site map of the dairy herd monitoring system on how all the python files are linked to one another.

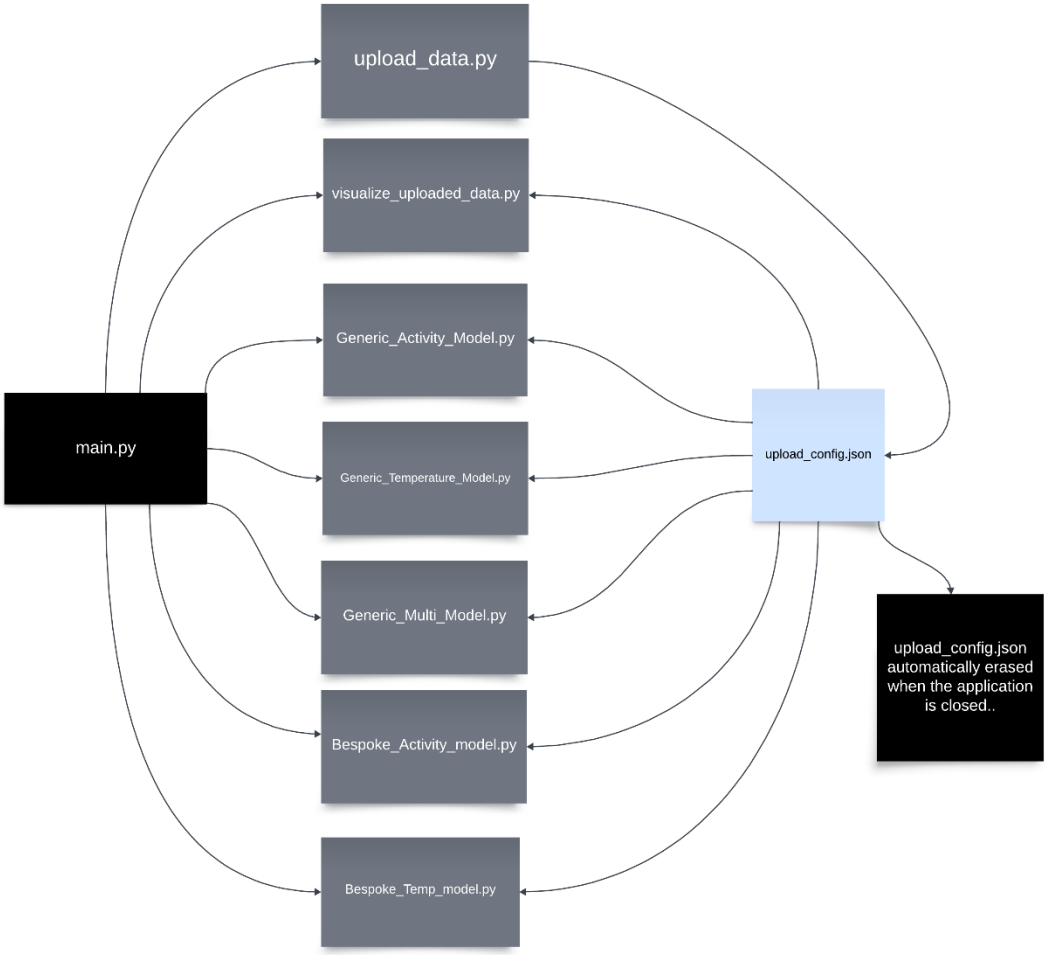


Figure 6.1

Figure 6.1: site map diagram of how the dairy herd monitoring system's python files are developed and linked.

6.9 Conclusion

This chapter reviews how the frontend and the backend of the dairy herd monitoring system were integrated to ensure even interaction between the user and the machine learning models. Buttons on the GUI interface act as triggers to call functions for model training or reloading and visualize results.

Chapter 7 – Tests and Results

7.1 Introduction to Testing and AI/ML Evaluation

Developing and implementing machine learning models for the dairy herd monitoring system is a crucial process that requires continues testing of the results achieved. The models are evaluated over time by train and error, several iterations and continuous refinement to get the highest accuracy possible. To improve accuracy of the models, initially the models were developed by experimenting with hyperparameters on all the models and their architectures. The aim was to forecast and identify potential health alerts across the dairy herd. During the initial trails, to improve the model accuracy, multiple epochs were tried and the results were tested. Sometimes as many as 500 epochs. But the results were inconsistent because the classification results were unreliable and the predictions had high variance compared to the actual data.

Both the generic and bespoke models were experimented with different configurations of LSTM layers, dropout layers and even the optimizers were tested. Some of them failed to capture the temporal dependencies which are crucial for time series data while some models were overfitted on the training data. Over several trials, I adjusted the LSTM architecture by balancing the epochs, improved generalization and experimented with smaller learning rates to avoid overfitting on the training data.

After several trails and fine tuning the models, the finalized models with high accuracy were discussed in chapter 5. These models that are finalized had the best possible balance between accuracy and efficiency. These finalized generic and bespoke models integrated into the frontend are now the ones that are the core components of the dairy herd monitoring system providing better predictions and health alerts of the dairy herd.

7.2 Introduction to Results

This part of the chapter explains the results of the finalized generic and bespoke models graph results after the evaluation of the models. The final results are visualized in the form of graphs, metrics and pdf reports when specific buttons are clicked. Below sections will also cover how confusion matrices, classification and regression results and downloadable reports are generated.

7.3 Visualization of Dairy Herd Data

The dairy herd monitoring system visualizes the dairy herds activity data and temperature data or combined through generic and bespoke models. The buttons for each of them allow the user to generate and visualize the graphs on the main dashboard.

7.3.1 Upload Data and Visualize Uploaded Data

The upload data button allows the user to upload their activity and temperature data which is crucial for data preprocessing, model training and visualization. Refer figure 4.1 to see the upload data buttons user interaction. Visualize uploaded data button allows the user to visualize graphs based on raw data for both individual cow's data and the entire herd. Figure 4.2 shows how the system provides different options to visualize the data. Additionally, the visualize uploaded data button allows the user to visualize pre-processed data of both individual cows and entire herd. figures 4.3, 4.4, 4.5, 4.6,

4.7, 4.8, 4.9 and 4.10 shows how the graphs are displayed on the system for both raw data and pre-processed data of the dairy herd uploaded by the user.

7.3.2 Generic Activity

Selecting generic activity button provides options to either train a new model or reload a model, that predicts cow's future activity with LSTM regression model and uses LSTM classification model to display the final alert.

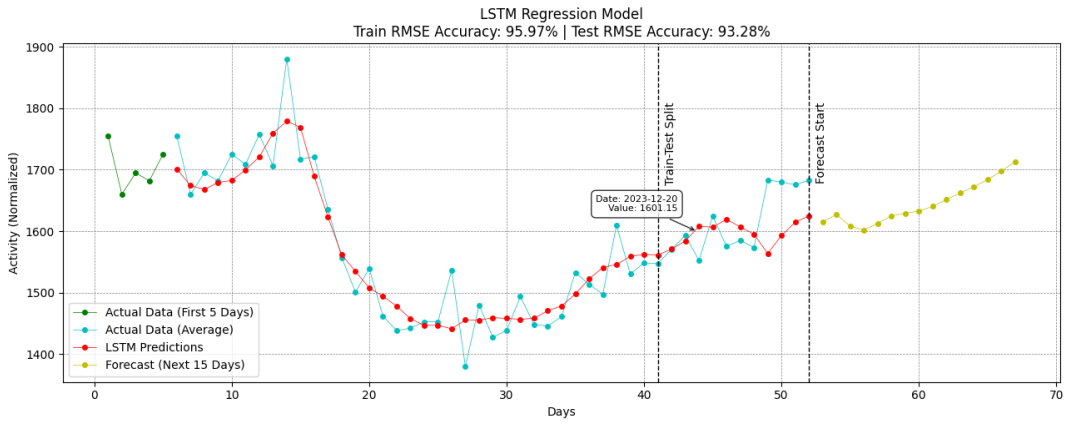


Figure 7.1: Predicted vs Actual activity data for the entire herd and 15-day forecast generated by generic activity regression model.

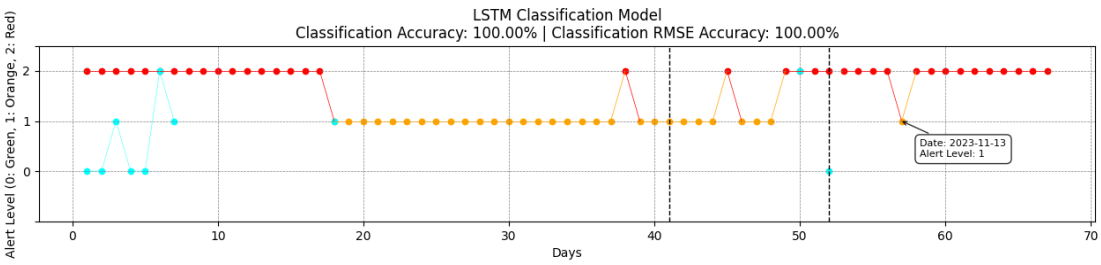


Figure 7.2: Predicted health alert graph generated by generic activity classification model. red colour indicating red alert, orange colour indicating orange alert and green colour indicating green alert. The cyan coloured data points are the actual cow alerts of the entire herd provided by Keele harper Adams Veterinary Science School.

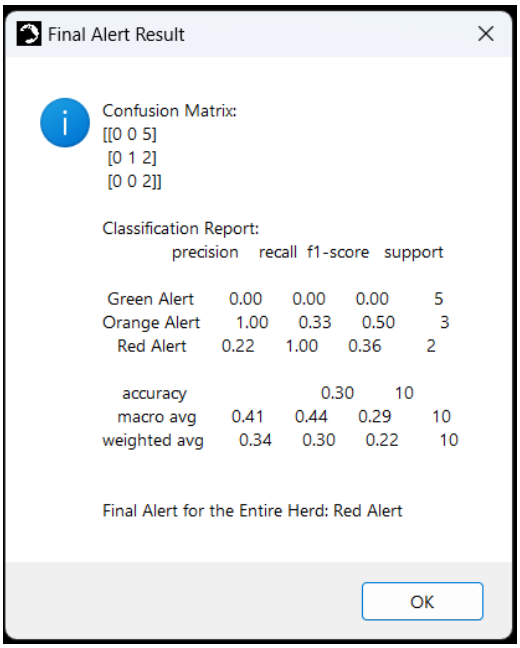
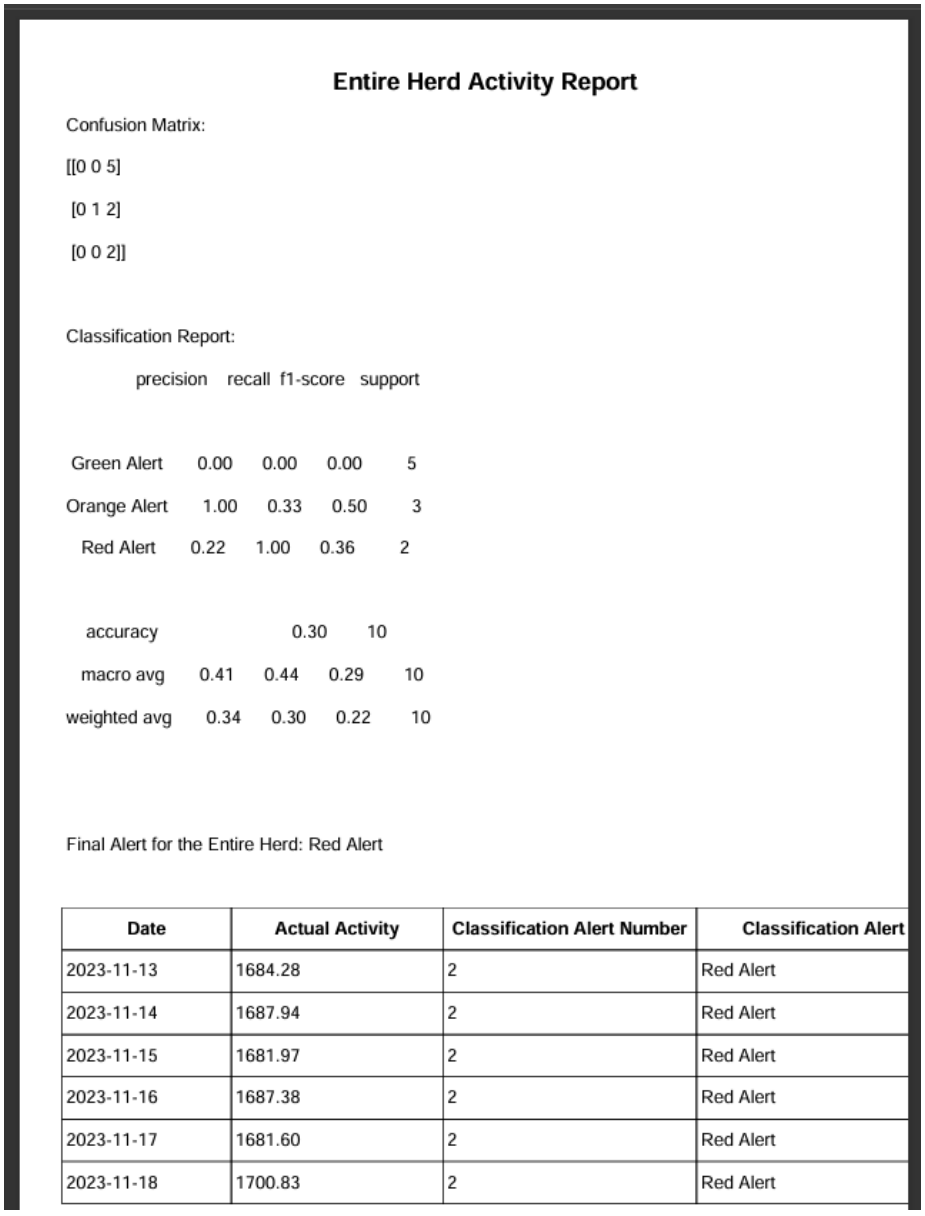


Figure 7.3: Confusion matrix results describing the predicted final alert of the entire herd for generic activity



Entire Herd Activity Report

Confusion Matrix:

[[0 0 5]

[0 1 2]

[0 0 2]]

Classification Report:

	precision	recall	f1-score	support
Green Alert	0.00	0.00	0.00	5
Orange Alert	1.00	0.33	0.50	3
Red Alert	0.22	1.00	0.36	2
accuracy			0.30	10
macro avg	0.41	0.44	0.29	10
weighted avg	0.34	0.30	0.22	10

Final Alert for the Entire Herd: Red Alert

Date	Actual Activity	Classification Alert Number	Classification Alert
2023-11-13	1684.28	2	Red Alert
2023-11-14	1687.94	2	Red Alert
2023-11-15	1681.97	2	Red Alert
2023-11-16	1687.38	2	Red Alert
2023-11-17	1681.60	2	Red Alert
2023-11-18	1700.83	2	Red Alert

Figure 7.4: image of report which is automatically downloaded after the graph is generated showing insights of the generic activity classification model performance.

7.3.3 Generic Temperature

Selecting the generic temperature button provides options similarly as the generic activity button, allowing the user to train or reload a model, providing users to visualize

predictions for entire herd with LSTM regression model and health alerts with LSTM classification model.

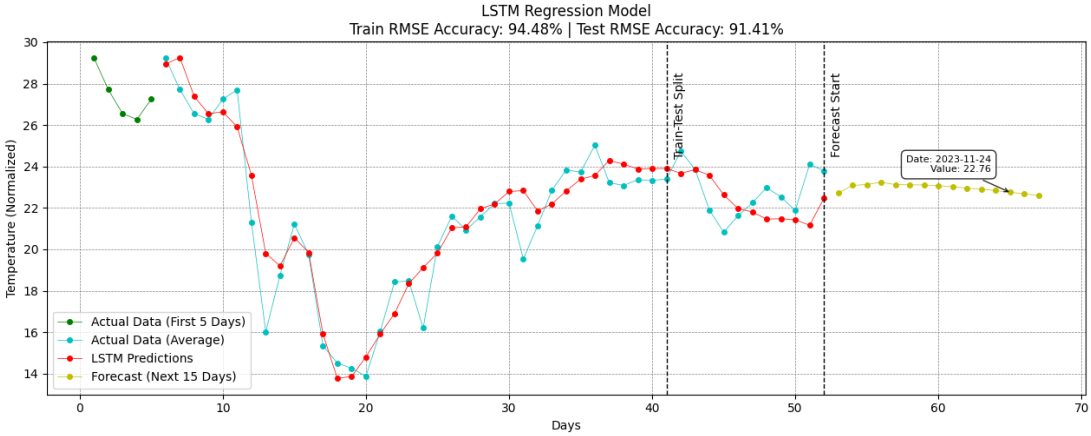


Figure 7.5: Predicted vs Actual temperature data for the entire herd generated by generic temperature regression model.

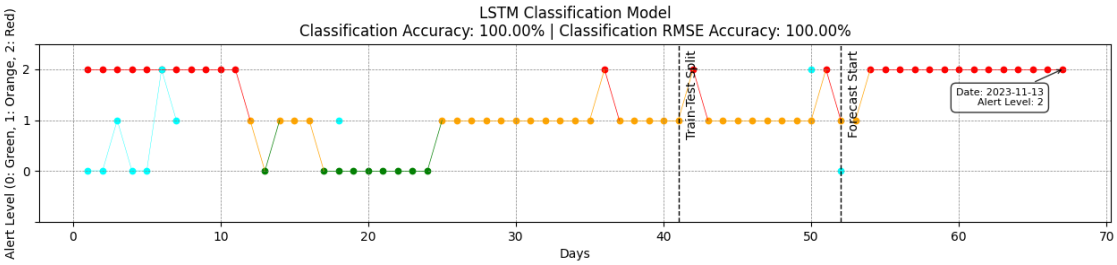


Figure 7.6: Predicted health alert graph generated by generic temperature classification model. red colour indicating red alert, orange colour indicating orange alert and green colour indicating green alert. The cyan coloured data points are the actual cow alerts of the entire herd provided by Keele harper Adams Veterinary Science School.

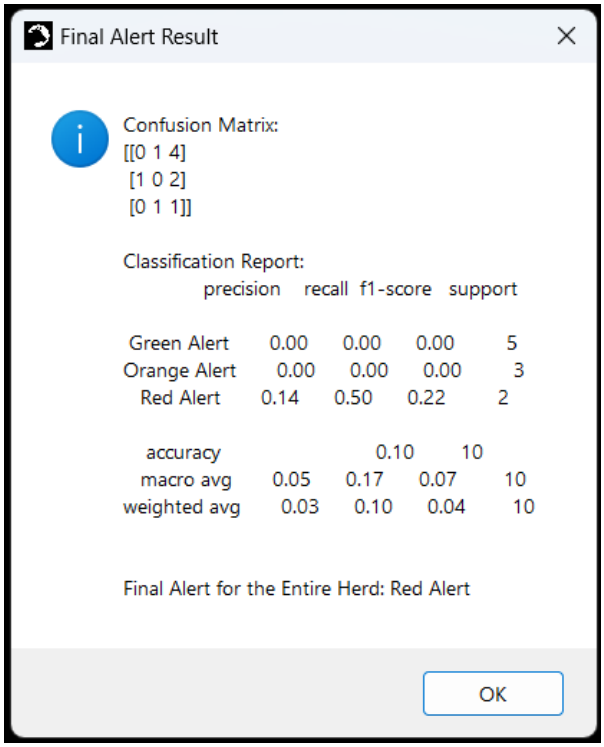
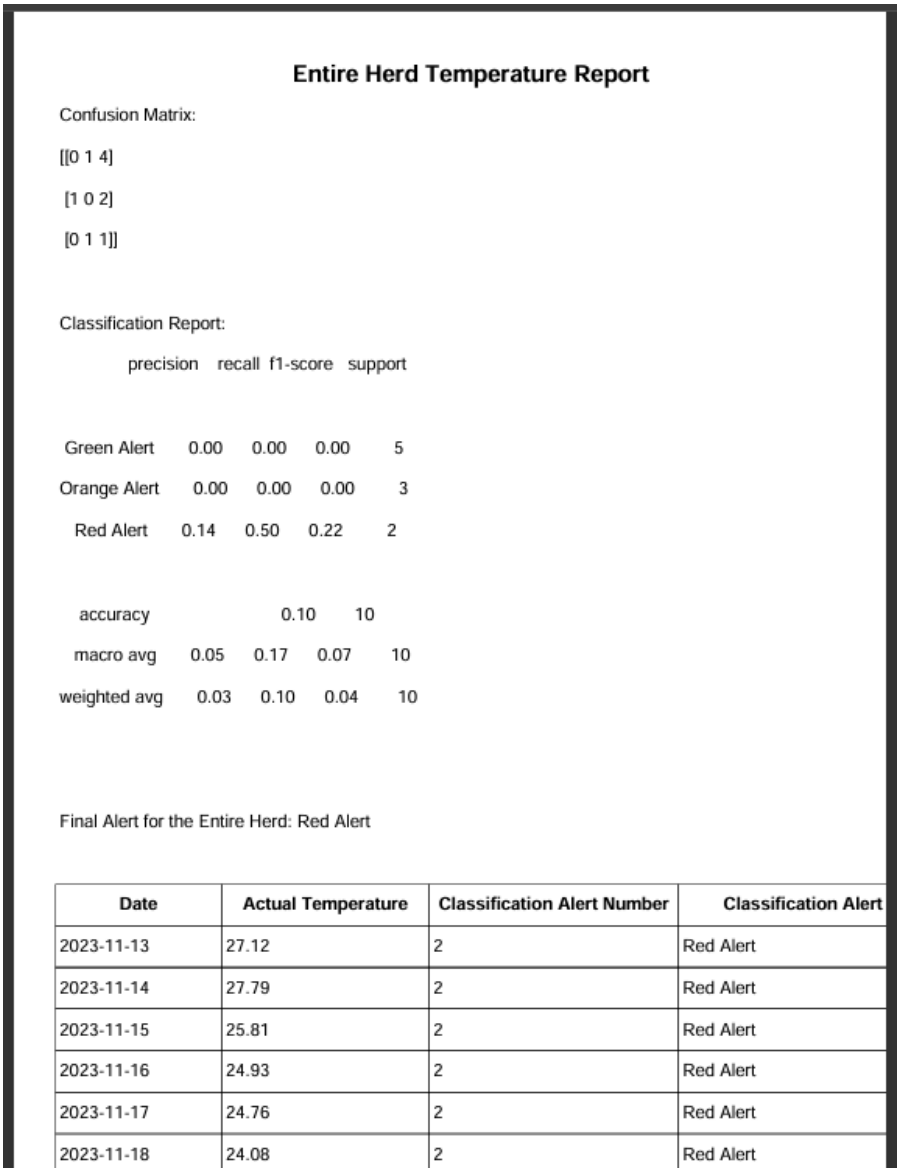


Figure 7.7: Confusion matrix results describing the predicted final alert of the entire herd for generic temperature



Entire Herd Temperature Report

Confusion Matrix:

[[0 1 4]
[1 0 2]
[0 1 1]]

Classification Report:

	precision	recall	f1-score	support
Green Alert	0.00	0.00	0.00	5
Orange Alert	0.00	0.00	0.00	3
Red Alert	0.14	0.50	0.22	2
accuracy			0.10	10
macro avg	0.05	0.17	0.07	10
weighted avg	0.03	0.10	0.04	10

Final Alert for the Entire Herd: Red Alert

Date	Actual Temperature	Classification Alert Number	Classification Alert
2023-11-13	27.12	2	Red Alert
2023-11-14	27.79	2	Red Alert
2023-11-15	25.81	2	Red Alert
2023-11-16	24.93	2	Red Alert
2023-11-17	24.76	2	Red Alert
2023-11-18	24.08	2	Red Alert

Figure 7.8: image of report which is automatically downloaded after the graph is downloaded showing insights of the generic temperature classification model's performance.

7.3.4 Generic Combined

The generic combined button provides options to train or reload a model, this generic combined architecture uses activity and temperature as separate inputs to train both

LSTM regression and LSTM classification model and visualize the predictions and health alerts of the entire herds overall health alert.

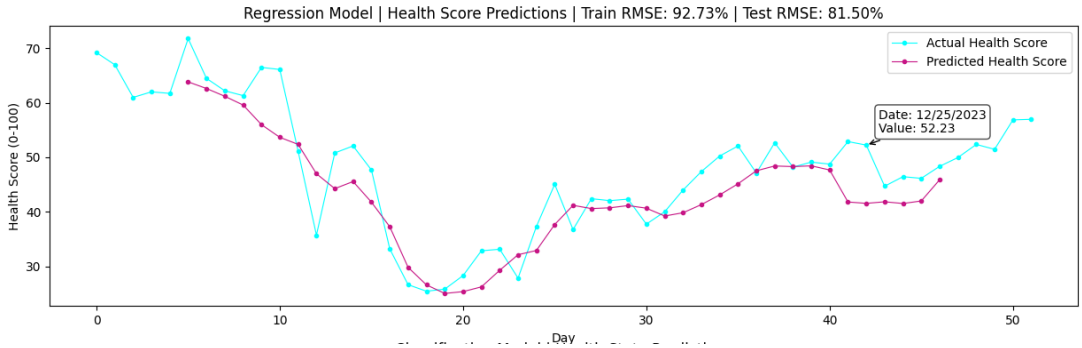


Figure 7.9: Predicted vs Actual combined data for the entire herd generated by generic regression model.

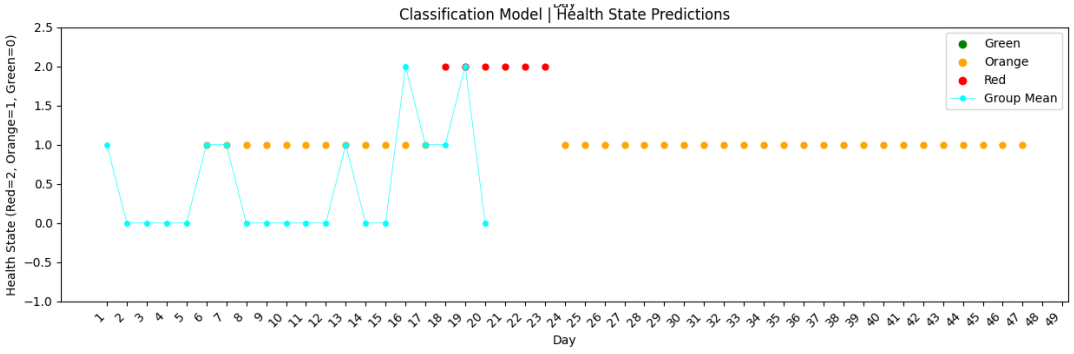


Figure 7.10: Predicted health alert graph generated by generic classification model. red colour indicating red alert, orange colour indicating orange alert and green colour indicating green alert. The cyan coloured data points are the actual cow alerts of the entire herd provided by Keele harper Adams Veterinary Science School.

Entire Herd Multi-Model Report

Confusion Matrix:

[[0 0 8]

[0 1 1]

[0 0 0]]

Classification Report:

	precision	recall	f1-score	support
Green Alert	0.00	0.00	0.00	8
Orange Alert	1.00	0.50	0.67	2
Red Alert	0.00	0.00	0.00	0
accuracy			0.10	10
macro avg	0.33	0.17	0.22	10
weighted avg	0.20	0.10	0.13	10

Final Alert for the Entire Herd: Red Alert

Date	Actual Activity	Classification Alert Number	Classification Alert
2023-11-13	1737.38	2	Red Alert
2023-11-14	1718.11	2	Red Alert
2023-11-15	1698.24	2	Red Alert
2023-11-16	1704.24	2	Red Alert
2023-11-17	1685.87	2	Red Alert
2023-11-18	1717.11	2	Red Alert

Figure 7.11: image of report which is automatically downloaded after the graph is downloaded showing insights of the model's performance including confusion matrix

7.3.5 Bespoke Activity Model

The bespoke activity model provides user to input a specific cow id so that the user can train an LSTM regression model and show the forecast of daily activity of the selected individual cow from the entire herd.

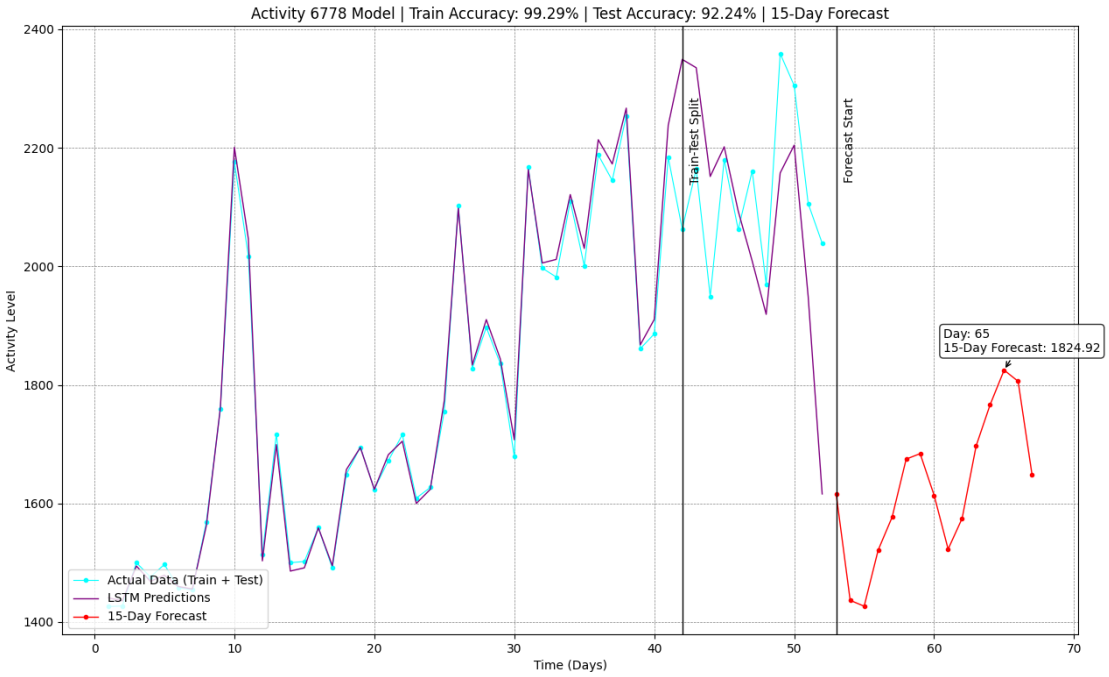


Figure: 7.12: shows the forecast of 15 days of the selected individual cows daily activity

7.3.6 Bespoke Temperature Model

The bespoke temperature model provides the end user to input a cow id. Once provided the backend trains an LSTM regression model a shows the forecast of future temperature data of the selected individual cow from the entire herd.

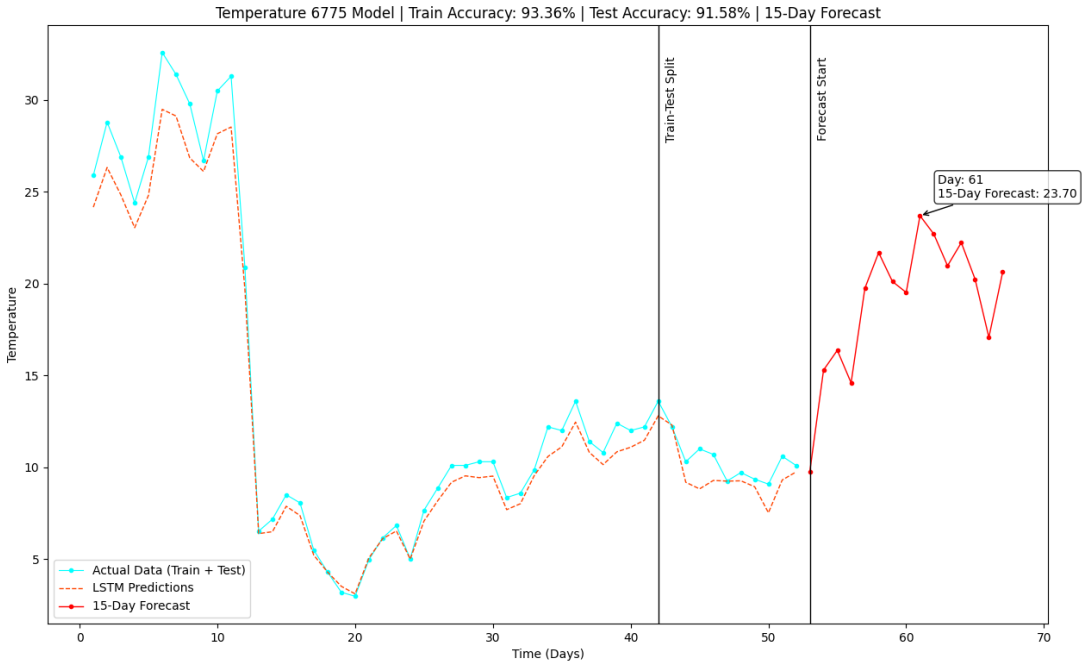


Figure: 7.13: shows the forecast of 15 days of the selected individual cows temperature data

7.4 Conclusion

This chapter reviews the overview of the testing process and evaluation of models developed for dairy herd monitoring system. Highlights all the generic and bespoke model's final results for both LSTM regression models and LSTM classification models. It also reviews the results of confusion matrix and classification reports generated while visualizing the generic models.

\

Chapter 8 – Conclusion and Recommendations.

8.1 Introduction

This chapter provides an overview of findings, conclusions and recommendations derived from participants comments and development of the Time-Series Neural Network Software for Dairy Herd Monitoring. The project aimed to develop a prediction and classification tools for early detection of health issues in dairy herds utilizing AI/ML techniques. The recommendations section includes valuable insights from participants P1, P2 and P3, who reviewed the software.\

8.2 Executive Summary

The development of dairy herd monitoring system met the primary aims and objectives of the project. The software allows users to upload their datasets, processes the uploaded time series data and predicts the health alerts and provides future predictions in cows daily activity and temperature data. Achievements of this project include:

- **Effectiveness of machine learning models**: the implementation of LSTM regression and LSTM classification models effectively captured time-series based patterns in the data and provided forecasts and health alerts in high accuracy.
- **Importance of Data Preprocessing:** data preprocessing steps included handling missing values and normalization which improved model performance and resulted in reliable outputs.
- **User Interface Design:** the GUI built with python was a simple interface to interact with machine learning models, met the projects objectives of providing ease of use to non-technical users.

- **Flexibility:** the system allows users to save and reload models providing flexibility for long term usage.

8.3 Recommendations

8.3.1 Prediction Accuracy

- **Participant feedback:** P1 mentioned that the tooltips were helpful and the software was easy to use. They were happy with the model accuracy of the predictions.

8.3.2 Improving User Experience

- Participant Feedback: P2 appreciated how simple the dairy herd monitoring system was to use, specially with the tooltips to view the datapoints on the graphs. They recommended to add a screen that gives easy to understand overview of the herds health, so users can quickly check at a glance.
- Recommendation: add a simple summary screen that shows the herds health status clearly, helping users see the most important information without viewing the reports.

8.3.3 Improving Model Flexibility

- **Participant Feedback:** P3 recommended giving users the ability to change model training parameters, including the model's settings or epoch count.
- **Recommendation:** To improve performance in various circumstances, future iterations should give users the ability to alter the model training settings according to their unique datasets and requirements.

8.4 Future Work

Although the system performed well of activity and temperature data, there are several areas where the dairy herd monitoring system can improve in its functionality.

8.4.1 Incorporating Additional Data Sources

In future enhancements of the dairy herd monitoring system, additional features for different types of data such as feed consumption or weather conditions could improve the model's accuracy. With more types of data more advanced LSTM models could be implemented to detect potential health issues in the dairy herd.

8.4.2 Exploring Advanced Models

While the current system uses vanilla LSTM models, future versions of the dairy herd monitoring system could be implemented with advanced architectures such as Bidirectional LSTMs or GRUs. These could enhance the systems ability to capture more complex data patterns which leads to more accurate health alerts.

8.4.3 Mobile and Web Application Development

Developing a mobile or web application would increase accessibility for farmers and veterans where they can remotely monitor the dairy herd. Developing a cloud-based solutions can provide real-time monitoring, allowing users to receive instant alerts of the dairy herd.

8.4.4 Data Security

Enhancing data security is a crucial step in dairy herd monitoring, because the system handles sensitive dairy herd data. Implementing features such as user authentication and encrypted database storage would ensure privacy and data safety.

8.5 Conclusion

The project successfully developed the machine learning based dairy herd monitoring system, capable of predicting health alerts in the dairy herd. Participants P1, P2 and P3 provided positive feedback and useful insights. While the dairy herd monitoring system meets its core objectives, enhancements in data security, adding additional data inputs and implementing more complex models makes the software more reliable in health monitoring. Developing a cloud-based web or mobile application software allows the users to use the software remotely making the system even more practical and valuable for farmers or veterans.

References

Brownlee, J. (2020) *4 Types of Classification Tasks in Machine Learning.* Machine Learning Mastery. Available at: https://machinelearningmastery.com/types-of-classification-in-machine-learning/ (Accessed: 20 August, 2024)

Brownlee, J. (2020) *How to Develop RNN models for Human Activity Recognition Time Series Classification.* Machine Learning Mastery. Available at: https://machinelearningmastery.com/how-to-develop-rnn-models-for-human-activity-recognition-time-series-classification/ (Accessed: 20 August, 2024).

Brownlee, J. (2020) *LSTMs for Human Activity Recognition Time Series Classification*. Machine Learning Mastery. Available at: https://machinelearningmastery.com/lstms-for-human-activity-recognition/ (Accessed: 22 August, 2024).

Brownlee, J. (2021) *Regression Metrics for Machine Learning.* Machine Learning Mastery. Available at: https://machinelearningmastery.com/regression-metrics-for-machine-learning/ (Accessed: 23 August, 2024).

Brownlee, J. (2021) Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. Machine Learning Mastery. Available at: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/ (Accessed: 27 August 2024)

Brownlee, J. (2021) *Neural Network Models for Combined Classification and Regression*. Machine Learning Mastery. Available at: https://machinelearningmastery.com/neural-network-models-for-combined-classification-and-regression/ (Accessed: 28 August 2024).

Brownlee, J. (2022). *How to Calculate Precision, Recall, F1, and More for Deep Learning Models*. Machine Learning Mastery. Available at: https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/ (Accessed: 02 September 2024).

de Myttenaere, A., Golden, B., Le Grand, B. & Rossi, F. (2016). *Mean Absolute Percentage Error for regression models*. Neurocomputing, 192, pp.38-48. Available at: https://doi.org/10.1016/j.neucom.2015.12.114 (Accessed:03 September 2024).

Fatima, S., (2024). *An Introduction to Tkinter: Python's GUI Toolkit*. Medium. Available at: https://medium.com/@saba_fatima/an-introduction-to-tkinter-pythons-gui-toolkit-59e9819d209f (Accessed: 03 September 2024).

Keceli, A.S., Catal, C., Kaya, A. & Tekinerdogan, B. (2020). *Development of a recurrent neural networks-based calving prediction model using activity and behavioral data*. Computers and Electronics in Agriculture, 170, p.105285. Available at: https://doi.org/10.1016/j.compag.2020.105285 (Accessed 06 September 2024).

Khan, M. A. I. (2023) *Building a Softmax Classifier for Images in PyTorch*. Available at: https://machinelearningmastery.com/building-a-softmax-classifier-for-images-in-pytorch/ (Accessed: 08 September 2024).

Mudigonda, V.S., (2021). *Understanding Dropout in Deep Neural Networks*. Medium. Available at: https://medium.com/datathings/dense-layers-explained-in-a-simple-way-62fe1db0ed75 (Accessed: 12 September 2024).

Moawad, A., (2019). Dense layers explained in a simple way. Medium. Available at: https://medium.com/datathings/dense-layers-explained-in-a-simple-way-62fe1db0ed75 (Accessed 10 September 2024).

Nalage, S.S., Santosh, K.A., Rajaram, S.A. & Sanjay, J.R. (2024). *Cattle Health Monitoring System with Smart Shelter*. International Journal of Solid State Innovations & Research, 2(1), pp.15-20. Available at: https://journals.stmjournals.com/ijssir/article=2024/view=170349/ (Accessed: 10 September 2024).

Patro, V.M. and Patra, M.R., 2015. A novel approach to compute confusion matrix for classification of n-class attributes with feature selection. *Transactions on Machine Learning and Artificial Intelligence*, *3*(2), p.52.

Rather, A.M. (2021). *LSTM-based Deep Learning Model for Stock Prediction and Predictive Optimization Model*. EURO Journal on Decision Processes, 9, p.100001. Available at: https://doi.org/10.1016/j.ejdp.2021.100001 (Accessed 10 September 2024).

Ryan T. J. J., (2020). *LSTMs Explained: A Complete, Technically Accurate, Conceptual Guide with Keras.* Medium. Available at: https://www.analyticsvidhya.com/blog/2020/09/lstms-explained-a-complete-technically-accurate-conceptual-guide-with-keras/ (Accessed 13 Sep. 2024).

Schneider, P., & Xhafa, F. (2022). Anomaly detection: Concepts and methods. In *Anomaly Detection and Complex Event Processing over IoT Data Streams.* Academic Press, pp. 49-66). Available at: https://doi.org/10.1016/B978-0-12-823818-9.00013-4 (Accessed: 10 September 2024).

Sharma, A., Jain, A., Gupta, P. and Chowdary, V., 2020. Machine learning applications for precision agriculture: A comprehensive review. *IEEE Access*, *9*, pp.4843-4873.

Sherstinksy, A. (2020). 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network', *Physica D: Nonlinear Phenomena*, 404, p. 132306. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0167278919305974?via%3Dihub (Accessed: 21 August 2024).

Shukla, S. (2024). *Building Graphical User Interfaces with Tkinter: A Dive into Python's GUI Toolkit*. Medium. Available at: https://medium.com/@shuklashubh818/building-graphical-user-interfaces-with-tkinter-a-dive-into-pythons-gui-toolkit-9f17a06aee75 (Accessed: 12 September 2024).

Verma, A., Kapoor, C., Sharma, A. and Mishra, B., 2021, April. Web application implementation with machine learning. In *2021 2nd International Conference on Intelligent Engineering and Management (ICIEM)* (pp. 423-428). IEEE.

Yildiz, B., Bilbao, J.I. and Sproul, A.B. (2017). *A review and analysis of regression and machine learning models on commercial building electricity load forecasting.*

Renewable and Sustainable Energy Reviews, 73, pp.1104-1122. Available at: https://doi.org/10.1016/j.rser.2017.02.023 (Accessed 08 September 2024).

Zhao, R., Wang, J., Yan, R. & Mao, K. (2016). Machine health monitoring with LSTM networks. In: *2016 10th International Conference on Sensing Technology (ICST*), Nanjing, China, pp.1-6. Available at: https://doi.org/10.1109/ICSensT.2016. (Accessed 03 September 2024).

Appendices

Appendix A: Msc Project Plan

MSc Project Plan CSC-40040 Project Plan

Student Name: Jyothesh Karnam

Student Username: y0k96

Student Number: 23032448

Module: CSC-40040 MSc Project

Keele Supervisor Name: Charles Day

Project Title: Time-series neural network software suite & app development for dairy herd monitoring

Project Overview and Description

Please provide a brief Project Description:

Colleagues in the Keele/Harper Adams veterinary science School have access to some small sets of data that have recorded the daily activity of a closely monitored herd of daily cows. Typically, these data have recorded the daily activity and temperature data of the herd over the period of a few months.

This project involves in the development of a time-series neural network software suite and an application for monitoring dairy herds. The main goal is to create an application that visualizes the data and applies AI/ML techniques which will detect the health issues in dairy cows before they occur.

Early detection will prevent the intensification of health issues, which can lead to affecting animal welfare and potential economic loss. By utilizing AI/ML techniques the project improves the accuracy and efficiency of health monitoring.

What are the aims and objectives of the Project?

Aims:

The aim of the project is to predict when taking action is required based on time-series dairy herd activity-level and temperature sensor data. The software suite will help in early detection of potential health issues in dairy cows. This improves the animal welfare and reduce economic losses for dairy farmers.

Objectives:

1. Data Selection and Visualization:
    - Develop a user-friendly interface that allows the selection of datasets of the dairy herd data.
    - Visualization tools will be implemented to provide insights of the daily activity levels and temperature variations of the dairy herd.
2. Data Pre-processing:
    - Apply the required pre-processing techniques to the dairy herd data to improve the effectiveness of machine learning models.
    - Visualize the pre-processed data and make sure it is ready for model training.
3. Model Training and Prediction:
    - To predict the alert levels based on pre-processed data, Train the suitable machine learning models, such as Long Short-Term Memory (LSTM) networks.
    - Visualize the results of these trained models. This will make it easy for the end users to better understand the predictions and take the required actions.
4. Model Management:
    - Implement features in the software suite to save and reload trained machine learning models, this ensures that the models can be reused and improved over time.

Please provide a brief overview of some background literature related to the Project:

Introduction to Time-Series Analysis and Neural Networks

Time-series analysis is an effective way to analyse and examine data points. Time series analysis involves statistical techniques. These techniques are used to examine data points which are collected at specifical intervals over time. For a highly effective time-series analysis, neural networks, specifically Long Short-Term Memory (LSTM) networks are used due to their ability to remember long term dependence in data.

Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network

LSTM networks and widely documented in scientific literature due to their efficiency in various applications. By transforming standard RNNs into “Vanilla LSTM” networks, they overcome the drawbacks of RNNs and improve training results by making them more capable of modelling long-term dependencies (Sherstinksy, 2020). This information allows LSTMs to retain important information over longer periods, which is a limitation in standard RNNs.

Develop an LSTM Network Model

This section involves a simple understanding of developing an LSTM network model for a dataset similar to what might be used in dairy herd health monitoring but applied in a general context.

LSTM networks and built to understand and retain information over long sequences of input data, typically covering from 200 to 400 times steps which makes them ideal for tasks involving continuous data streams (Brownlee, 2020). This capability of LSTMs makes them suitable for forecasting and analysing patterns in data that extend over longer durations.

The model can handle multiple parallel sequences of input data, identifying features and mapping them leading to different outcomes (Brownlee, 2020). Multiple streams of data can be processed by LSTMs simultaneously which improves their forecasting capabilities and accuracy.

Since LSTMs can learn directly from raw time series data, they do not require manual feature engineering, which is a significant advantage when applying them for sequence classification tasks (Brownlee, 2020). The ability of LSTMs to work directly with raw data simplifies the modelling process and reduces the requirement for intensive data preprocessing.

Project Process and Method

Please provide a brief overview of the Methodology you expect to be used in the Project:

I will be using Scrum framework for iterative progress and structured planning, there will be weekly sprints to set goals and track tasks. I will conduct reviews and retrospectives to analyse the work and make necessary improvements. Making sure there are continuous improvements and adjustments throughout the project.

Will any special Data Collection Methods will be employed (e.g card sorts, questionnaires, simulations, …)?

No special Data Collection Methods will be employed. The project will use existing datasets provided by Keele/Harper Adams veterinary science School which will be used to train the neural network models. For third-party software evaluation, a human participant will evaluate the software's functionality and performance.

Table of Risks (*if non Standard Hardware and/or Software to be used please include backup options/ contingency plans here)*:

Risk Description	Probability of occurrence	Prevention measures	Backup option (Remedy)

Loss of Project work	Low	Regular backups using cloud storage and version control with Git	Restore from the latest backup and ensure to commit frequently.
Shortage of time to complete project	Mid/ High	Detailed project planning with regular progress tracking.	Prioritize main parts of the software and complete them first.
Inability to proceed with certain library	Low	Initially test the libraries while building the software and continue with the one that works best for the software.	Identify alternative libraries to work with and maintain a clean and editable code for easy changes.

Skill Development?

1. Software Development:
    - GUI Development: Creating a GUI using python libraries like Tkinter or PyQT and developing the GUI in a user-friendly manner.
    - Backend Integration: Integration of machine learning models with the GUI
2. AI/ML:
    - Model design and Training: To train time-series neural network models such as LSTM, Python with TensorFlow and Keras will be used.
3. Predictive Analysis:
    - Data Analysis: Predictive analytics will be applied to make predictions based on historical data of the dairy herd.

- Model Validation: Ensure the reliability of time-series models by using the historical data.

4. Project Management:
    - Agile Methodologies: Utilizing Scrum for iterative development, including sprint planning and daily progress tracking.
    - Version Control: Using Git for version control to manage code changes.
5. Documentation:
    - Documentation process: keeping backup records of each section of software development process, including designs and testing outcomes.

Time and Resource Planning

Will Standard Departmental Hardware be used? YES

If NO please outline the Hardware/Materials to be used:

Will Software which is already available in department be used?

YES

If NO please outline the Software to be used including how any necessary licences will be obtained:

Will the project require any Programming? YES

If YES please list the (potential) Programming Languages to be used (including any IDEs and Libraries you may make use of):

1. Programming Languages: Python
2. IDEs: Visual Studio Code: Jupyter notebook
3. Libraries:
    - GUI Development: Tkinter or PyQT
    - AI/ML: TenserFlow, Keras, NumPy, Pandas
    - Visualization: Matplotlib, Seaborn
    - Version Control: Git, GitHub

Gantt Chart (must include milestones and deliverables):

Red Border: Submission dates.

	Weeks											
Week	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Tasks												
Project Plan & ethics												
Analysis & Design			Sprint1	Sprint2								
Software Development Phase				Sprint2	Sprint3	Sprint4	**Sprint5**	Sprint6	Sprint7			
AI/ML Implementation					Sprint3	Sprint4	Sprint5	Sprint6	Sprint7			
Integration					Sprint3	Sprint4	Sprint5	Sprint6	Sprint7			
Poster												
AI/ML Evaluation												
Software Evaluation												
Report (Writing & demo)												
Documentation												

References

Please include a list of References used in this Plan:

- Brownlee, J. (2020) *How to Develop RNN models for Human Activity Recognition Time Series Classification*. Available at: https://machinelearningmastery.com/how-to-develop-rnn-models-for-human-activity-recognition-time-series-classification/ (Accessed 29 June 2024).
- Sherstinksy, A. (2020) 'Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network', *Physica D: Nonlinear Phenomena*, 404, p. 132306. Available at:

https://www.sciencedirect.com/science/article/abs/pii/S0167278919305974?via%3Dihub (Accessed 29 June 2024).

Appendix B: GDPR and Ethics Check List

MSc Projects and Industrial Placements

GDPR and Ethics Checklist 2024

STUDENT NAME	Jyothesh Karnam
STUDENT NUMBER	23032448
E-MAIL	y0k96@students.keele.ac.uk
MODULE CODE	CSC-40040 MSc Project
PROJECT OR WORK PLACEMENT TITLE	Time-series neural network software suite & app development for dairy herd monitoring
KEELE SUPERVISOR NAME	Charles Day
DATE	21/06/2024

Complete all the questions below and electronically sign and date at the end. Ask your supervisor to check the responses and to also electronically sign and date below. You should submit this completed checklist to the KLE drop-box provided.

Please retain your own copy of the completed checklist as all end of project reports/dissertations must provide a copy of this completed checklist in the Appendices. If a Project Ethical Review Application Form has also had to be

completed a copy of the final ethical approval certificate must also be included with your Project Plan document in your final report/dissertation Appendices.

GDPR Check

Does your project involve the use or collection of “personal data” for which permission will not have been explicitly granted? The project will not involve the use or collection of personal data. The datasets which will be used in this project will be recorded data of daily activity levels and temperature data of dairy herd provided by Keele/Harper Adams veterinary science School.	No

Ethics Check

Will your project involve the use of human participants or capturing human data? This project will involve human participation for software evaluation and receive feedback on the functionality, usability and overall user experience of the software application.	Yes

Significant Ethical Concern Checklist

Could the project expose the participants or the project student to images and/or information that they might find distressing (e.g. pictures or descriptions of injuries, symptomatic health conditions, or atrocities, or pictures or descriptions of tumours or cancerous cells, or creatures in distress)?	No
Does the project involve deception of the participants?	No
Could the project uncover information about identifiable individuals that could cause embarrassment or distress to one or more of those individuals (e.g. evidence of illegal or unethical behaviour, such as fraud or illegal drug use or a personal revelation)?	No

Could the project cause pain, discomfort or risk to the participants and/or the project student?	No
Will the project involve participants who are vulnerable in any way? (e.g. participants who are under 18, or who are mentally or physically impaired, or participants who may feel under pressure to participate.)	No
Does the project involve recall or discussion of personal or sensitive memories?	No
Does the project involve a significant risk of participants later regretting taking part?	No
Does the project involve procedures which are likely to provoke interpersonal or inter-group conflict?	No

If the answer to any of the Significant Ethical Concern Checklist questions is "Yes" (or you think "Maybe") you must discuss your project and aims with your supervisor and possibly with the Module Co-ordinator to assess whether an appropriate level of ethical scrutiny might be required via the completion of the *Faculty of Natural Sciences (non-Psychology) Research Ethics Application Form*.

GDPR Check Guidance:

Personal data includes any and all of: names, addresses, emails, phone numbers, bank details, employment details, IP addresses, date of birth, medical or health data, images, video or audio recordings.

If you answered "Yes" you must not proceed with your project.

It is illegal under European GDPR legislation to make use of personal data without explicit permission. Discuss your project with your supervisor and revise your plans to ensure you do not risk illegal use of personal data.

Note. Even if personal data is publicly available on the Internet, it must not be used without permission. (Also note that you cannot contact individuals to request permission to use their on-line data without prior ethical approval to do so).

It is strongly recommended that you either:

1. use non-personal data for your project, or
2. use existing, well-established, publicly available databases or data repositories, for example: https://archive.ics.uci.edu/ml/index.php, https://physionet.org/ or https://www.kaggle.com/ etc. (A list of acceptable data repositories is maintained on the CSC-30014 KLE pages.) You might also see, for example, https://blog.scrapinghub.com/web-scraping-gdpr-compliance-guide) for further information.

Continued on the following page …

I confirm that the responses are correct and that the project, as proposed, is GDPR compliant and that ethical approval will be sought if required and that any work

requiring ethical approval will not take place unless ethical approval has been granted. If, during the course of the project work, any of the information supplied on this checklist changes substantially a new checklist will need to be completed and then brought to the attention of the School's Ethics Advisory team.

Signed (Student) Jyothesh Karnam	Date:22/06/2024

I confirm that I have read the form and that the project, as proposed, is GDPR compliant and that, to the best of my knowledge, the ethical information is correct.

Signed (Supervisor) Dr Charles Day	Date: 3/7/2024

Electronically typed signatures and dates *are* acceptable.

Appendix C: Ethical Evaluation Form

MSc Projects and Industrial Placements

Ethics Application Form 2024

Applicant details

Project title	**Time-series neural network software suite & app development for dairy herd monitoring**
Name of final-year project student	**Jyothesh Karnam**
Keele e-mail address of student	**y0k96@students.keele.ac.uk**
Name of project supervisor	**Charles Day**

Project Summary

Provide a short summary of your project **(max 250 words).**

The aim of the project is to develop a software that will predict when taking action is required based on time-series dairy herd activity-level and temperature sensor data. The software suite will help in early detection of potential health issues in dairy cows.

The work will involve:

1. Asking Student or Staff (as an end user) their user experience and feedback for the dairy herd monitoring app,
2. Testing the usability of the Dairy herd monitoring software application.

In simple terms, provide a *short* description of your planned experimental method that involves participants (e.g., focus groups, questionnaires, interviews, experimental observations).

- Test the final software product of dairy herd monitoring application (approx 20 mins).
- Provide feedback on their experience and overall functionality

Describe the characteristics of your participants, and any inclusion or exclusion criteria. Estimate the approximate number of participants.

All participants will be students or staff from School of Computer and Mathematics and/or from Veterinary Science School

Approximately, 2-3 participants will be recruited as an end user to provide feedback on their user experience and overall functionality.

Recruitment and Consent

Indicate how potential participants will be identified, approached and recruited and outline any relationship between the researcher and potential participants.

Students and staff will be invited in person or via email with the following invitation:

You are invited to participate in a study for a final-year project entitled: Time-series Neural Network software suite and App Development of Dairy Herd Monitoring

Participation will involve approximately 20 mins of your time to test and provide feedback of the Dairy Herd Monitoring Application. These activities will be scheduled according to your availability. If you are interested in participating, please read the Information Sheet.

Describe the process that will be used to seek and obtain informed consent.

If individuals express an interest in participating they will be provided with an Information Sheet (printed or electronic). If they are happy to proceed they will be asked to sign the Consent Form before participating.

Will consent be sought to use the data for other research? Yes ☐
No ☒

Will consent be sought to contact the individual to participate in future research? Yes ☐
No ☒

Can participants withdraw from the research? Yes ☒
No ☐

If yes, state up to what point participants are able to withdraw from the research

Participants can withdraw their data up to one week after participating.

If yes, outline how participants will be informed of their right to withdraw, how they can do this and what will happen to their data if they withdraw.

Contac Information for withdrawal is provided on the Participant Information Sheet.

If no, explain why they cannot withdraw (e.g., anonymous survey).

Confidentiality and anonymity

Outline the procedures that will be used to protect, as far as possible, the anonymity of participants and/or confidentiality of data during the conduct of the research and in the release of its findings.

All participant names will be replaced by codes, for example, P01, P02, P03, etc. No names (or other information that might reveal participant identity) will be included in the project dissertation or in any other presentation.

Storage, access to, management of, and disposal of data

Describe how participant data will be stored; where it will be stored and for how long; and how/when it will be disposed of.

All participant data will be securely stored until the end of the project. All electronic records will be stored on university drives. Only the applicant (student investigator) and supervisor will have access to participant data. At the end of the project the participant data will be deleted.

Other ethical issues raised by the research

Are there any other ethical issues that may be raised by the research? Yes ☐ No ☒

If yes, please give details.

Declarations

Declaration by student I confirm that: ● The form is accurate to the best of my knowledge. ● I will inform my supervisor and resubmit an ethical application if there are any changes to the project. ● I am aware of my responsibility to comply with the requirements of the law and any relevant professional guidelines.
Student name Jyothesh Karnam
Student signature Jyothesh
Date 30/07/2024

Declaration by supervisor I confirm that: ● I have read the application and am happy for it to proceed for ethical review. ● The application is accurate to the best of my knowledge. ● I am aware of my responsibility to ensure that the applicant is familiar with and complies with the requirements of the law and any relevant professional guidelines.
Supervisor name Dr Charles Day

Supervisor signature	Charles Day
Date	30/7/24

Participant Information Sheet

Study Title: Time-series Neural Network software suite and App Development for Dairy Herd Monitoring

Aims of the Study

The aim of the study is to create a software that will help in predicting potential health issues in dairy cows.

Invitation

You are being invited to consider taking part in the study: **Time-series Neural Network software suite and App Development for Dairy Herd Monitoring.** This project is being undertaken by **Jyothesh Karnam**

Before you decide whether or not you wish to take part, it is important for you to understand why this study is being done and what it will involve. Please take time to read this information carefully and discuss it with friends and relatives if you wish. Ask us if there is anything that is unclear or if you would like more information.

Why have I been invited?

As a student or member of staff of the School of Computing and Mathematics and/or Veterinary Science School you are eligible to participate.

Do I have to take part?

You are free to decide whether you wish to take part or not. If you do decide to participate you will be free to stop participating at any time and you can remove your data up to one week after participating without giving reasons.

What will happen if I take part?

You will be asked to test the software application of Dairy herd monitoring. You will be provided datasets to upload into the application. You will use the application and provide feedback on your experience and your opinions on the overall functionality of the software. This will take approximately 20 minutes. This will be scheduled according to your availability and the availability of other participants.

What are the benefits (if any) of taking part?

You will have the opportunity to test and experience how AI/ML techniques integrated into a software helps in predicting potential health issues in dairy cows. Additionally, student participants may find the experience of participating helps inform their ideas for their own current or future project.

What are the risks (if any) of taking part?

No risks have been identified for this participation.

How will information about me be used?

Completed consent forms will be retained securely until the end of the project when they will be destroyed. (Completed consent forms will not be included in the project dissertation).

Who will have access to information about me?

Your data will be stored securely until the end of the project. All electronic records will be stored on university drives. Only myself (student) and supervisor will have access to your data.

At the end of the project your data will be deleted.

What if there is a problem?

If you have a concern about any aspect of this study, you may wish to speak to me (the student investigator) or my supervisor. We will do our best to answer your questions. Our contact details are:

Student: Jyothesh Karnam, **Mail ID:** y0k96@students.keele.ac.uk

Supervisor: Charles Day, **Mail ID:** c.r.day@keele.ac.uk

Contact for further information and withdrawal

If you would like withdraw your data from the study, please contact:

Student: Jyothesh Karnam, **Mail ID**: y0k96@students.keele.ac.uk

Keele UNIVERSITY

CONSENT FORM

Title of Project: Time-series Neural Network Software Suite and App development for Dairy Herd Monitoring

Name and contact details of student investigator and supervisor:

Jyothesh Karnam, y0k96@students.keele.ac.uk , Charles Day, c.r.day@keele.ac.uk

Please tick box if you

agree with the statement

1. I confirm that I have read and understood the Participant Information Sheet and have had the opportunity to ask questions. ☐

2. I understand that my participation is voluntary and that I can stop participating at any time and can withdraw my data up to one week after participation without giving a reason. ☐

3. I agree to take part in this study. ☐

4. I understand that data collected about me during this study will be anonymised before it is reported in any reported results. ☐

5. I agree to audio recording of the focus group and testing. ☐

Name of participant	Date	Signature
Student investigator	Date	Signature

(for use of quotes)

Title of Project: Time-series Neural Network Software Suite and App development for Dairy Herd Monitoring

Name and contact details of student investigator and supervisor:

Jyothesh Karnam, y0k96@students.keele.ac.uk & Charles Day, c.r.day@keele.ac.uk

Please initial box if you

agree with the statement

1. I agree for my quotes to be used ☐

2. I do not agree for my quotes to be used ☐

SYSTEM USABILITY SCALE QUESTIONNAIRE

Participant ID:

Date:

System Usability Scale

Instructions: For each of the following statements, highlight the option that best describes your reactions for Dairy Herd Monitoring Software.

Please check and respond to the following usability questions 1 to 6:

1. How confident do you feel using the system to analyse and predict herd data.

- Very Difficult
- Difficult
- Neutral
- Easy
- Very Easy

2. How easy or difficult was it to navigate the system interface?

- Very Difficult
- Difficult
- Neutral
- Easy
- Very Easy

3. Do you feel the system provides sufficient feedback or error messages when something goes wrong?

- Strongly Disagree
- Disagree
- Neutral
- Agree
- Strongly Agree

4. How would you rate the clarity of the graphs and data visualizations presented by the system?

- Very Unclear
- Unclear
- Neutral
- Clear
- Very Clear

5. how easy do you find the system when performing tasks like loading data, training models and visualizing results?

- Very Difficult
- Difficult

- Neutral
- Easy
- Very Easy

6. How likely do you think the system would help a farmer or veteran for dairy health monitoring.

- Very Unlikely
- Unlikely
- Neutral
- Likely
- Very Likely

Comments and Recommendations: Please provide any additional feedback or suggestions for improving the system.

Your Answer: