Ben-Gurion University of the Negev

Faculty of Engineering Sciences

Department of Industrial Engineering and Management

A computer vision system and predictive models for individual feed intake measurement in dairy cows

Thesis submitted in partial fulfillment of the requirements for the M.Sc. degree

By: Benyamin Katz

Under the supervision of: Prof. Yael Edan and Prof. Ilan Halachmi

August 2024

Ben-Gurion University of the Negev

Faculty of Engineering Sciences

Department of Industrial Engineering and Management

A computer vision system and predictive models for individual feed intake measurement in dairy cows

Thesis submitted in partial fulfillment of the requirements for the M.Sc. degree

By: Benyamin Katz

Under the supervision of: Prof. Yael Edan and Prof. Ilan Halachmi

Author signature:	בניימין כץ	Date:	29/08/2024
Supervisor signature:	Yael Edan	Date:	29/08/2024
Supervisor signature:	Halachmi	Date:	29/08/2024
Chairman of Graduate Studies Committee:		Date:	

August 2024

Abstract

This thesis presents the design, development, and evaluation of a system for measuring individual feed intake on a dairy farm. The goal is to develop a non-mechanical and reliable system for outdoor conditions to improve feed efficiency, thereby enhancing economic viability and environmental sustainability. The system utilizes 3D cameras to detect cows in feeding zones and calculate the feed volume. It incorporates environmental factors that affect feed density. Predictive models have been developed to estimate feed intake of individual cows.

Data were collected using a 3D camera system, weight scales, and weather sensors in a commercial research cowshed of the Agricultural Research Organization in Israel. The data collection system was specially designed as part of this thesis and included design of a system to capture images, weight measurements, and environmental conditions at regular intervals in an outdoor cowshed. Image processing algorithms for cow detection were developed using YOLOv8. The feed-volume calculation algorithm was developed by manipulating a point cloud. Various regression models have been developed to predict the feed volume and weight decrease based on environmental factors, including linear regression, lasso, ridge, elastic net, decision tree, random forest, and gradient boost. Additionally, models were developed to predict meal weight by analyzing the reduction in feed volume over the course of individual meals and changes in environmental variables.

Data collection took place from June 27, 2023, to August 1, 2023, at the Volcani Agricultural Research Organization's dairy farm in Israel. During this period, the system collected data from 56 Holstein cows. Measurements were recorded every 20 seconds for feed volume and weight, and every 2 minutes for weather conditions.

The system demonstrated accuracy in predicting the feed volume and weight decrease. Random Forest and Gradient Boosting machine learning models, showed improved accuracy compared to linear models, achieving R^2 values of up to 0.973 for volume decrease prediction and 0.974 for weight decrease prediction. The system identified the key environmental factors that influence feed intake. The prediction of meal weight achieved an R^2 value of 0.880 and Mean Absolute Error (MAE) of 1.547. KG.

Key words: individual cow feed intake, feed volume measurement, depth camera, RGBD camera, precision livestock farming (PLF), computer vision in agriculture.

Acknowledgements

I gratefully thank all members of the Precision Livestock Farming (PLF) Laboratory and the staff of the Volcani research dairy-farm for their assistance in the research. Special thanks to Joseph Lepar and Assaf Godo for their advice and contribution in system installation and integration in the dairy-farm. I want to thank my friends from PLF lab who advised me when I needed; Tom Lev-Ron, Noam Bergman, Ron Biton and Ran Shmulevich. Thanks to ARO's dairy-farm team who hosted me in Volcani dairy-farm.

I want to extend my heartfelt thanks to my academic supervisors, Prof. Yael Edan and Prof. Ilan Halachmi, for their unwavering support throughout my academic journey. Even in the most challenging and difficult times, they have been invaluable, and I cannot express my gratitude sufficiently. I am truly grateful for the wealth of knowledge they shared with me.

I also want to extend my deepest thanks to my family. Your support, encouragement, belief in me throughout this journey are invaluable. Most importantly my wife for walking this path with me. My success is also yours.

I appreciate you all,

Benny Katz

List of Publications

Katz Benyamin, Edan Yael, Halachmi Ilan. A 3D computer vision system and predictive models for individual feed intake measurement in dairy cows. BioSystems Engineering, Special Issue. (in preparation)

Katz Benyamin, Edan Yael, Halachmi Ilan – A system for individual feed intake measurement in dairy cows: computer vision system and predictive models– National Cattle Science Conference 2024 – Jerusalem, Israel.

Presentations

Katz Benyamin, Edan Yael, Halachmi Ilan - Automatic Machine Learning Pipeline of a Machine Vision System for Measuring Individual Cow Feed Intake - Student Conference of the Institute of Agricultural Engineering 2024

Table of Contents

List of Appendices

List of Tables

List of Figures

1. Introduction

1.1. Problem Description

In dairy farming, achieving feed efficiency is important for the economic viability and environmental sustainability of farms. The industry faces the challenge of managing feed costs, which can make up to 60% of the farm's budget, while mitigating the environmental impacts associated with inefficient feed consumption and greenhouse gas emissions. A significant aspect of this challenge is the inconsistency in feed intake and the physical properties of the feed, which can be influenced by changing weather patterns. This inconsistency can affect the nutritional content of the feed and subsequently, the quantity of feed required by cows to maintain optimal productivity.

The identification of cows with superior feed efficiency—consuming less and producing more milk—is crucial. However, the lack of precise, cow-specific data on feed intake, combined by the influence of environmental factors on feed density, has hampered the development of effective feed-management strategies. Traditional methods for monitoring these variables often rely on mechanical systems, which, while useful, introduce complexities due to the need for frequent maintenance and calibration. The mechanical properties of these systems render them susceptible to wear and tear, operational failures, and inaccuracies over time, leading to increased operational costs and potential disruptions in data collection.

Given these challenges, a solution that can measure and analyze feed consumption and its influencing factors is required. The system should be non-mechanical to minimize maintenance. Implementing such a solution can benefit dairy farms by providing a sustainable, cost-effective, and reliable approach to improve feed efficiency, thereby reducing their workload. This study proposes a non-mechanical system that utilizes advanced imaging and data analysis technologies to address these issues. By offering detailed insights into individual cow feed intake and the environmental impacts on feed density without the drawbacks of mechanical maintenance, this system can improve feed management strategies and establish new standards of efficiency and sustainability in dairy farming.

1.2. Objective

The aim of this study was to design a computer vision system that employs three dimensional (3D) cameras to monitor and predict the feed consumption of cows in cowsheds. The system evaluates the environmental factors that influence feed density and develops predictive models to estimate the individual feed consumption of cows based on cow detection, feed volume reduction, and weather conditions.

1.3. Research contributions

This system contributes to the development of a method that can identify cows, calculate their consumed meal weight without using mechanical weight scales, and determine the feed intake of individual cows. This system may assist farmers in identifying cows with higher feed efficiency, potentially leading to increased milk or meat production with lower feed input. This approach can contribute to reducing greenhouse gas emissions and improving farm profitability. This thesis focused on individual feed intake measurements and predictions. The specific contributions include development of models that are able to cope with variable illumination conditions and environmental factors. The system was developed for a camera placed at 4m enabling to view multiple feeding stations. This research does not address individual biometric cow identification, which has been explored in other studies (e.g., Bergman et al., 2024).

2. Literature Review

2.1. Precision Livestock Farming in cows

Precision livestock farming (PLF) uses advanced technologies to optimize the management and welfare of livestock. In the dairy industry, PLF has the potential to reduce feed costs and environmental impacts, improve breeding, and provide valuable insights to farmers. This literature review examines key aspects of PLF in cows.

2.1.1. Feed Costs

Feed is a significant part of the cost of milk production that can account up to 84% of a dairy farm's total cost (Hemme, Uddin, & Ndambi, 2014). PLF technologies can help improve feed efficiency. For example, Halachmi et al. (1998) developed a real-time control system for individual cow feed intake. Van De Gucht et al. (2017) used PLF to detect lameness and enable early treatment, preventing drops in feed intake. André, Engel, Berentsen, Vellinga, & Oude Lansink (2011) demonstrated that using a machine learning model to predict feed intake could improve dependency of temperature.

2.1.2. Environmental Impact

The dairy sector is a major contributor to greenhouse gas emissions (GHG),(Gerber et al., 2013). However, PLF can help reduce this impact. De Haas et al. (2011) found that methane production is heritable in cows, suggesting that breeding could reduce emissions. Bell, Wall, Russell, Simm, & Stott (2011) showed that improving cow productivity decreases the carbon footprint. Intelligent climate control systems in barns can also reduce energy use and improve cow comfort (Laberge & Rousseau, 2017).

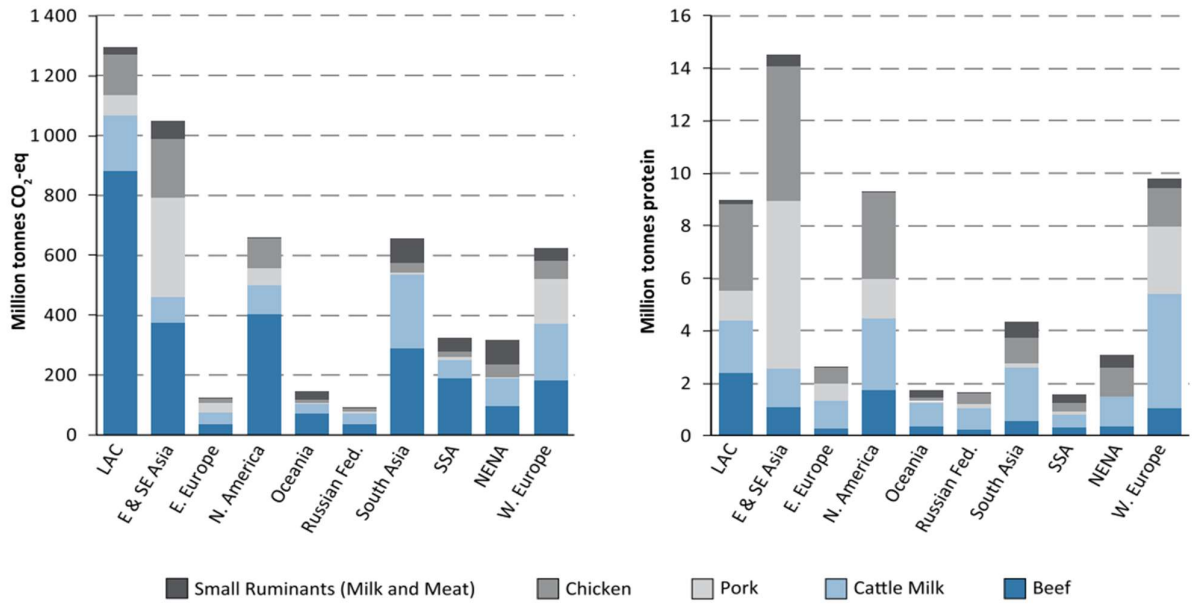

Figure 1: Global livestock production and GHG emissions from livestock, by commodity and regions (Gerber et al., 2013)

2.1.3. Breeding Applications

The diversity in the ecology and metabolism of cows' rumens has an impact on their feeding habits (Khiaosa-ard & Zebeli, 2014). Therefore, breeding for desired traits is important for sustainability. Brito et al. (2021) reviewed how genetic selection in high-yielding cows can promote sustainable farming in a changing climate. For instance, Pickering et al. (2015) used genomic selection to breed for lower residual feed intake (RFI). PLF enables collection of detailed genetic data to support these efforts (Cole & VanRaden, 2018). Stear et al. (2012) discussed how PLF can improve disease resistance.

2.1.4. Benefits for Farmers

PLF empowers farmers to make data-driven decisions that boost productivity and animal welfare. Wearable sensors can detect health issues (Arcidiacono, Porto, Mancino, & Cascone, 2017). Machine learning can find patterns in the data to guide management choices (Mirani et al., 2021). Rojo-Gimeno et al. (2019) discussed how besides profitability, information collection can improve animal welfare, environment food safety and food security. In addition to the economic gains, PLF reduces the labor cost for farmers. Steeneveld & Hogeveen (2015) noted that PLF could reduce issues related to labor availability.

2.2. Individual Feed Intake Measurement in Cows

Accurate individual feed intake measurement is important for improving feed efficiency, animal health, and farm's productivity (Davison et al., 2021). Different systems have been developed to monitor and manage feed intake in cows, each with its strengths and limitations. This section reviews the evolution of these systems, starting with traditional methods, followed by computer vision (CV) techniques, and concluding with advanced deep learning (DL)-based approaches.

2.2.1. Non-Computer Vision-Based Systems

Halachmi et al. (1998) developed a real-time system that could monitor individual cow feed intake, that detected a cow by a photocell sensor and an early version of radio frequency identification (RFID). Chizzotti et al. (2015) utilized an ear tag RFID technology in conjunction with electronic feeding stations and achieved a mean absolute error (MAE) of 0.9 kg/day. Ding et al. (2022) used triaxial accelerometers predict feed intake based on jaw movements in cattle using machine learning, achieving an R^2 of 0.97 and an RMSE of 0.36 g/min.

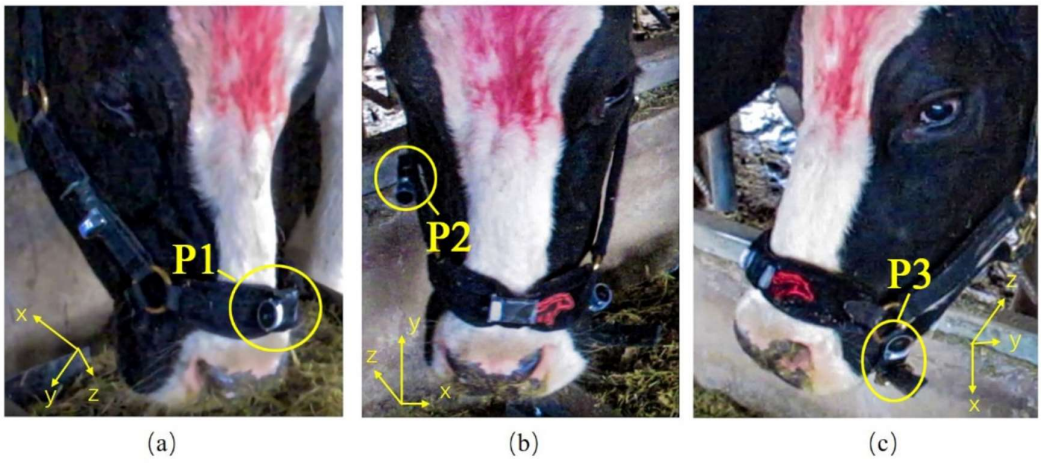

Figure 2: The orientations of the X, Y, and Z axes when the triaxial accelerometers were at the positions of (a) P1, (b) P2, and (c) P3. (Ding et al., 2022)

2.2.2. Computer Vision-Based Systems Without Deep Learning

Bloch, Levit, & Halachmi (2019) developed a photogrammetry-based system creating 3D models of feed heaps before and after feeding. The system utilized multiple cameras, with the highest positioned at 180 cm (centimeter) above the feed heaps. Tested in both lab and cowshed environments, the system achieved an error of 1.24 kg for feed heaps up to 40 kg and an R^2 of 0.997. Giagnoni et al. (2024) validated a system with 3D camera positioned 450 cm above feeding platforms, in indoor conditions. The system estimates feed volume and converts it to weight, achieving a relative error below 8% for daily intake. Lassen et al. (2018) positioned a 3D 420 cm above the feeding. The system calculated feed volume by dividing the feed into virtual boxes. Cow detection was based on identifying head outlines, with the system successfully assigning feed intake to individual cows.

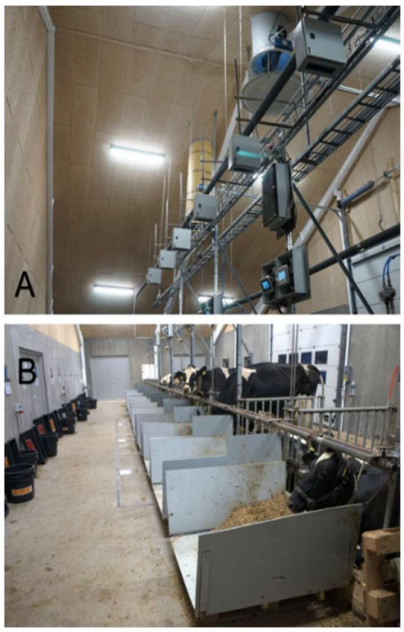

Figure 3: Pictures from the tie-stall housing, with the 3D cameras on top of the feeding platforms

2.2.3. Computer Vision-Based Systems with Deep Learning

Bezen, Edan, & Halachmi (2020) developed a computer vision system utilizing an RGB-D camera and deep learning algorithms for measuring individual feed intake, in an outdoor cowshed environment. The camera was positioned 140 cm above the feeding area. A convolutional neural network (CNN) was trained to identify a cow for the number its neck and achieved a 93.65% accuracy. The system tracked feed consumption with an MAE of 0.127 kg per meal. Saar et al. (2022) developed a system using an RGB-D camera and deep learning models, specifically CNNs with transfer learning, to predict individual cow feed intake in an outdoor cowshed. Positioned 130 cm above the feed lane, the system achieved an MAE of 0.12-0.14 kg per meal across different feed types. Wang et al. (2023) employed a Siamese network model to measure feed intake using depth images, with cameras positioned at a height of 97 cm in a semi-open cowshed environment. The system achieved a MAE of 0.1 KG. Cow identification was performed using RFID tags.

Figure 4: Illustration of the automatic system installed in the cowshed consisting of two feeding stations, cameras, computer, weighing palette, LED lights (Saar et al., 2022).

3. Materials and Methods

3.1. Overview

The system was designed to collect data to build models for predicting:

- Weight of meals consumed.
- Volume and weight decrease in the feed.

The data were collected simultaneously using a single system specially designed for this research. The system included two weight scales, one to collect data on consumed feed for meal weight prediction and the second to collect data on unconsumed feed to predict the effect of time and weather on the decrease in the weight and volume of unconsumed feed.

Data collection and preparation consisted of two phases for each element:

- cow detection model training.
- volume and weight decrease prediction models, and the meal weight prediction model.

3.2. Experimental Setup

The experimental data were collected within the research cowshed of the Agricultural Research Organization, Volcani Center, located in central Israel. The cowshed housed 56 Holstein cows throughout the experimental period. The ingredients of the feed provided to the cows are detailed in Appendix B.

3.2.1. System design

The experimental setup was designed as a part of this thesis (Figure 6). Observational data were recorded using a Zed 2i 3D camera system (Stereolabs, France) mounted 400 cm above the cowshed floor, positioned parallel to the ground and facing downward in a z-down orientation system. The Zed 2i camera was chosen because of its reliability and ease of integration with the Jetson Xavier NX device. The 400 cm height was selected to allow the tractors and other mechanical equipment to pass underneath, allowing for potential commercial applications. Also, the decision to raise the camera from 140 cm (Bezen et al., 2020) or 130 cm (Saar et al., 2022) to 400 cm allowed for the measurement of up to 6 positions simultaneously, significantly increasing the coverage area, enabling to measure from multiple feeding stations in future works. The camera was leveled using a water leveler to maintain the accuracy and stability of the experimental setup.

To monitor the actual feed intake, two weight scales were positioned beneath the camera (Figure 5). Scale #1 is placed out of the cows' reach and is used to measure unconsumed feed. This scale helps predict the volume and weight decrease of the feed due to environmental factors, without the influence of cows' eating behavior. Scale #2 is positioned within the cows' reach and is used to measure the

consumed feed. Data from this scale is used to predict consumed feed weight in a single meal. The scales are used to provide ground truth measures. Each scale used a 1000 kg loadcell with 0.023% precision (SQB, Keli CEE, Poland). The first scale, located outside the feed lane and out of the cows' reach, was 100 cm in both width and length The second scale was 140 by 105 cm, which was placed within the feed lane, was designed for access to individual cows. Both were surrounded by 15 cm high barriers to prevent feed spillage. Moreover, to avoid any potential displacement or misalignment of the weight scales, each was housed within a robust metal frame firmly secured to the ground. Rubber bases were positioned under the scale legs to ensure level positioning. For data acquisition from the weight scales, each scale was connected to an Arduino Uno (Arduino, Italy) paired with a BMP280 amplifier (Adafruit, USA), and calibration was performed using a 20 kg dumbbell.

Additionally, an Arduino Uno coupled with a BME280 sensor (Bosch Sensortec, Germany) was used

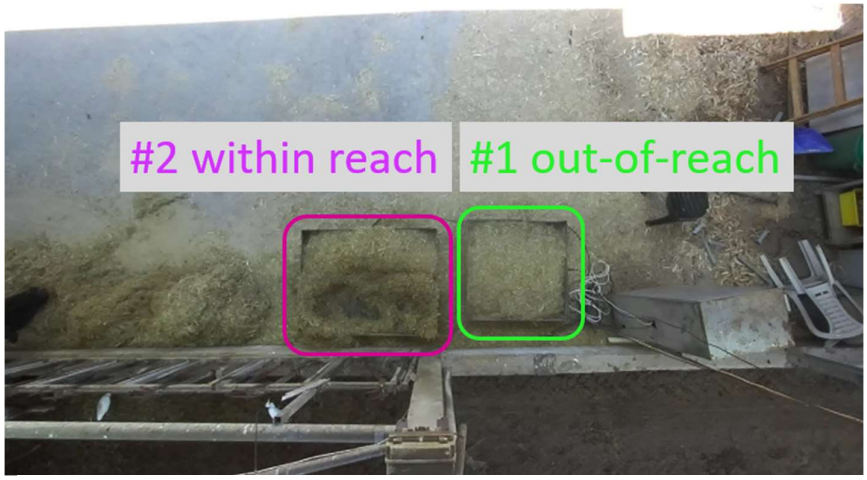

Figure 5: Weight scales as seen from camera point of view

to gather weather-related data and capture temperature, humidity, and pressure readings.

These devices were interconnected with a Jetson Xavier NX device (NVIDIA, USA), which managed them through a Python script, ensuring the integration and synchronization of data collection processes.

Weather parameters were recorded at two-minute intervals, while weight measurements, along with depth and RGB images, were captured every 20 seconds. This data collection approach aimed to ensure an analysis of the factors influencing the studied variables.

Two floodlights (Nisko, 200W, 1800 lumens, 6000K, beam angle 130°) were used to provide continuous illumination for the experimental area. One floodlight was positioned 300 cm to the right of the center point at a height of 330 cm, while the second was placed 250 cm to the left at a height of 300 cm. The floodlights remained on continuously throughout the experiment.

All data were collected using two TB external hard drives (Western Digital, USA). Four such hard drives were used, which were changed periodically to ensure continuous data collection and storage.

Data processing and analysis were performed using Python software. The specific libraries and versions used are detailed in Appendix C.

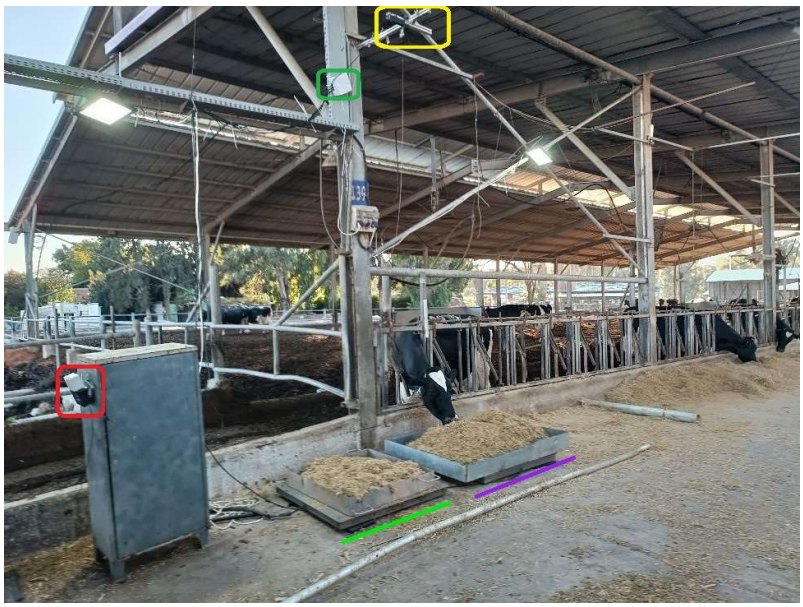

Figure 6: Specially designed experimental setup for monitoring feed intake and weather data in the cowshed, featuring a Zed 2i 3D camera system (yellow), weight scales with barriers for spillage prevention (light green and purple), and a Jetson Xavier NX device (dark green) managing the data collection. Weather data were gathered using an Arduino Uno with a BME280 sensor (red), while feed intake was measured using scales connected to an Arduino Uno and BMP280 amplifier.

3.3. Cow Detection

3.3.1. Data Collection

From January 20, 2023, to January 23, 2023, images were acquired for cow recognition purposes. Images were acquired every 5 seconds. The data aimed to cover a diverse range of hours throughout the day, thereby capturing variations in the illumination conditions. A total of 249 images were selected for inclusion in the dataset to ensure representative sampling of the lighting scenarios encountered at different times during the day. The data collection process and cow detection model training are detailed in Appendix A, and the libraries used for the model training are detailed in Appendix D.

3.3.2. Data Labeling

The images used in this study were annotated using the Roboflow platform (Dwyer, Nelson, & Solawetz, 2022). The bounding boxes were designed to encircle the entire head and ears of the cattle within the frame (Figure 7), while excluding the body. This approach allows for the determination of whether a cow is engaged in eating behavior or simply traverses the area when the model detects the location of the head. From this dataset, 70%, 20%, and 10% of the data were randomly allocated to the training, validation, and test sets, respectively (174, 49, and 26 images were designated as the training, validation, and test sets, respectively).

Figure 7: An example of an annotated image of cows' heads using the Roboflow platform, with bounding boxes encircling the head and ears while excluding the body.

3.3.3. Preprocessing and Augmentation

Using the Roboflow platform, the following preprocessing and augmentation steps were performed to prepare the images for training with YOLOv8(Hussain, 2023).

For preprocessing, an auto-orientation procedure was applied to ensure uniform alignment across the dataset. The images were resized to dimensions of 640x640 pixels to conform to the input requirements of the YOLOv8 model architecture.

Augmentation techniques were employed to improve the robustness and generalizability of the model (Figure 8). Image augmentation involved rotating the images by up to ± 45°, adjusting the brightness within a range of ± 40%, and applying blurring effects of up to ±0.75 pixels. For the training set only, each image was augmented using each of the techniques, resulting in three variations per image (original and two augmented images). This increased the train-set from 174 to 522 images. These

augmentation strategies augment the dataset with diverse variations, providing the model the ability to learn and generalize patterns under varying conditions.

Figure 8: Test-set images after augmentations

3.4. Models

Various models were developed to predict the Weight Decrease, Volume Decrease, and Meal Weight tasks. The selection of models prioritized interpretability, thus linear models and tree-based algorithms were chosen to be trained, over deep learning models.

Linear regression served as a baseline model, while stepwise regression techniques (both backward and forward) were employed for their feature selection capabilities. Principal Component Analysis (PCA) in conjunction with linear regression was utilized as an additional approach to feature selection and engineering. Regularized linear models, including Ridge, Lasso, and Elastic Net regression, were implemented to address high-dimensional feature spaces.

Tree-based models were selected for their ability to capture non-linear relationships and handle diverse data distributions without strong assumptions about the underlying structure. These models also offer robustness to outliers and the capacity to model complex interactions between variables.

Feature selection techniques were tailored to the model type and prediction task. For linear models, stepwise regression methods (both forward and backward) were employed to identify the most significant predictors. In the case of meal weight prediction, regression trees were utilized for feature selection, leveraging their ability to capture non-linear relationships.

Detailed explanation about models, and their evaluation are described in chapter 4.

3.5. Analysis

Model performance was evaluated using multiple metrics: R^2 for goodness of fit and Mean Absolute Error (MAE) for average magnitude of prediction errors and in relative absolute error for meals weight prediction. Relative absolute error defined as the absolute error of each meal divided by the meal weight.

Robustness was tested with the best model compared to the second best and the linear regression models, as a baseline model. Robustness analysis included three parts:

- 10-fold cross-validation to ensure consistent performance across different data subsets.
- Maximal test-set size analysis to determine model stability with varying amounts of test data.
- Random noise tests to evaluate model resilience to data perturbations.

Feature importance was quantified using permutation importance analysis, providing insights into the contribution of each feature to the model's performance.

4. Models

4.1. Cow Detection

Three model variations were trained: YOLOv8n, YOLOv8s, and YOLOv8m, which are referred to as Nano, Small, and Medium, respectively. The training process involved monitoring the precision, recall, mAP50, and mAP50-95. The small variant, YOLOv8s was selected for real-time implementation owing to its superior performance and ease of operation on an edge device. In our research, NVIDIA Jetson Xavier NX (NVIDIA, USA) was used for this purpose.

4.2. Volume and weight decrease prediction and meal weight prediction models

For the volume decrease, weight decrease, and meal-weight predictions, it was necessary to collect volume, weight, and weather data. The data collection phase took place between June 27, 2023, and August 1, 2023. Each day, data collection began at 11 am or the feeding time, whichever occurred later, and continued until 6 am the following day. This period was defined as the feeding day. Throughout every feeding day, every 20 seconds a cycle of volume and weight data collection was performed. Weather data from the cowshed were collected every two minutes.

In the Volume Data section, the volume calculation is described. This volume calculation is relevant for both volume decrease prediction and meal-weight prediction models.

4.2.1. Data Collection

All data collection and preparation procedures were implemented using Python software. The code for these processes is available in Appendix A.

4.2.1.1. Volume Data

4.2.1.1.1. Depth Measurement

To determine the depth for each frame, point cloud data were recorded at 20-second intervals (Figure 10). The process of calculating depth from these point clouds is illustrated in Figure 9. As shown in the figure, frames were then cropped to include only the area within the scales, leaving a 5 cm margin from each edge. This margin was implemented to mitigate the influence of noisy points near the edges of weight scales. From these cropped point clouds, the z-values were extracted to measure the depth of specific points in space. Averaging these z-values provided the average depth of the area of interest. The average depth was then compared to the baseline value to determine the depth of food on the weight scale. To calculate the volume of food, the previously determined depth of food was multiplied

by the width and length of the respective weight scale. To smoothen the volume over time, locally weighted scatterplot smoothing (LOWESS) regression was employed with a fraction value of 0.05.

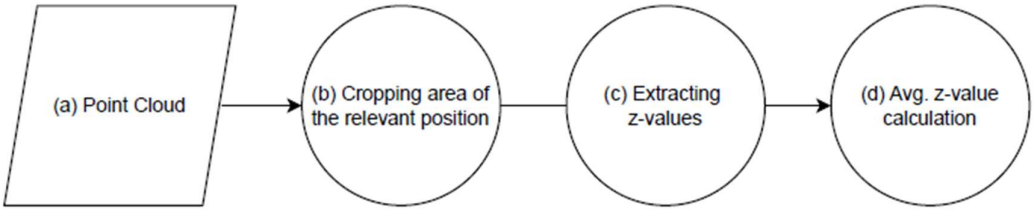

Figure 9: Flowchart of calculating depth of position (a) initial point cloud with x,y,z coordinates, (b) cropped point cloud (c) extracted z values, and (d) averaged z values.

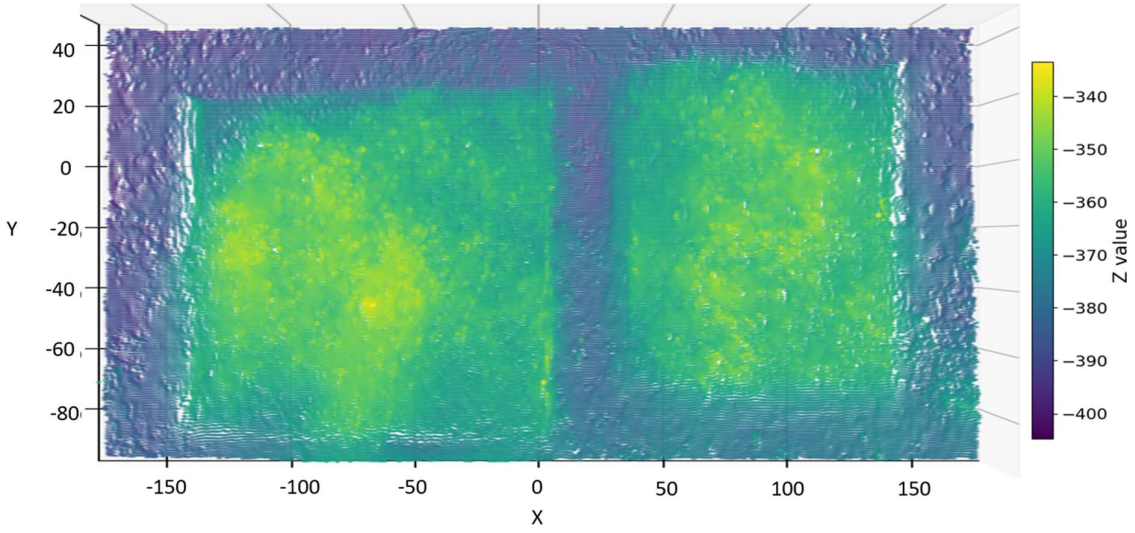

Figure 10: An example of a depth map of the weight-scales, after transforming from a captured point-cloud, as seen from the camera.

4.2.1.1.2. Baseline Depth Measurements

To establish the baseline depth for the empty surfaces, measurements were taken from an empty scale between 04/07/23 and 06/07/23. The average depth was recorded every 20 seconds over a 43-hour period. A regression model was then developed to predict the average depth at any time of day using these measurements and their corresponding timestamps (Figure 11). We employed a locally weighted scatterplot smoothing (LOWESS) function (Cleveland, 1979) with a fraction value of 0.05 to smooth the data. LOWESS was chosen for its non-parametric nature, allowing us to capture potential non-linear patterns in the baseline depth variations without assuming a specific underlying distribution. Following smoothing, the Ridge regression was trained using a grid search. The optimal found values

were alpha = 0.7, polynomial transformation degree = 16, for positions outside the cows' reach. and alpha = 2.3, polynomial transformation degree = 16 for the position within the cows' reach.

Figure 4 shows two scatter plots with regression lines representing the baseline depth throughout the day for two positions: out-of-reach (1) and within-reach (2) of cows. Each point represents an average empty position measurement.

This regression model was used to calculate food volume by providing a baseline depth for the empty weight scale at each time point of the day. This approach minimized noise from factors such as varying sunlight exposure on the scales. The Python code implementing this process is available in Appendix A.

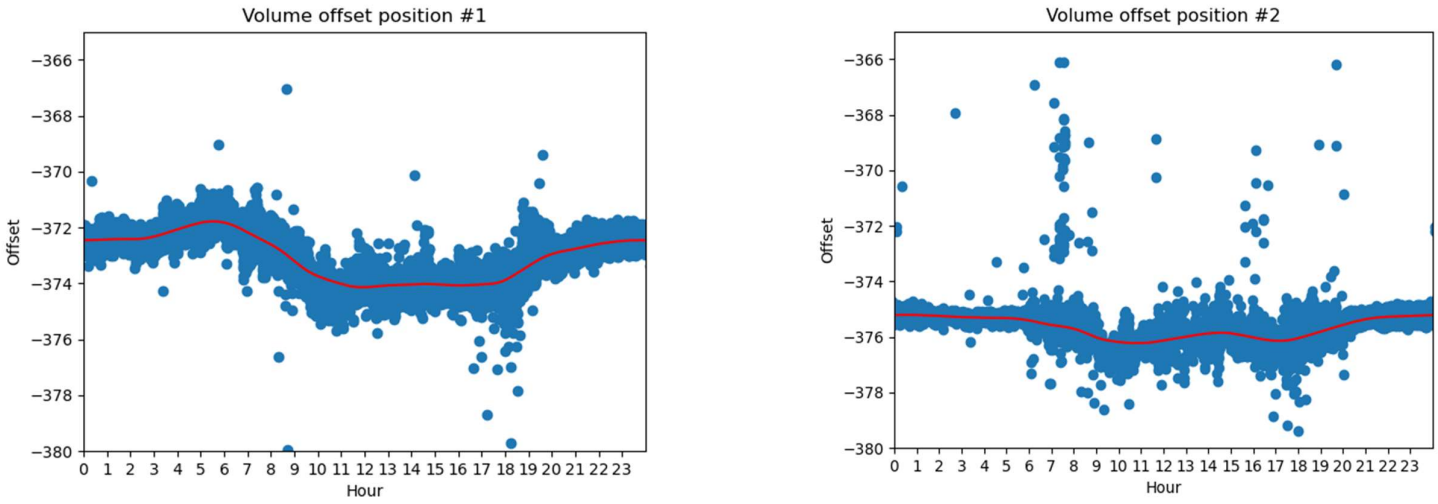

Figure 11: Regression of the baseline depth and measurements throughout the day of position out-of-reach (1) and within-reach (2) of cows

4.2.1.1.3. Volume calculation

Building upon the depth calculation method described in the Depth Measurement section and the baseline depth determination outlined in the Baseline Depth Measurements section, we developed an algorithm to calculate feed volume (Figure 12). For each measurement, the algorithm utilizes the calculated depth from the point cloud and the corresponding capture time. It then retrieves the appropriate baseline depth for an empty position from the previously established time-dependent regression model, specific to each measurement position. The actual feed depth is computed as the difference between the measured depth and this baseline depth. To convert depth to volume, the algorithm multiplies the feed depth by the position-specific dimensions: 100 cm × 100 cm for the out-

of-reach position and 140 cm × 105 cm for the within-reach position. The resulting volume, initially in cubic centimeters, is then converted to liters by dividing by 1000. This approach integrates our depth calculation and baseline determination methods to provide accurate feed volume estimates while accounting for temporal variations and position-specific characteristics. The full code is available in Appendix A.

```
Function calculate_volume(depth_measurement, datetime_of_capture, position_number):
    # Step 1: Calculate seconds since midnight from the given datetime
    seconds_since_midnight = calculate_seconds_since_midnight(datetime_of_capture)

    # Step 2: Get the baseline depth based on position number
    If position_number == 1:
        baseline_depth = get_baseline_depth_position_1(seconds_since_midnight)
        width = 100  # width in cm
        length = 100  # length in cm
    Else If position_number == 2:
        baseline_depth = get_baseline_depth_position_2(seconds_since_midnight)
        width = 140  # width in cm
        length = 105  # length in cm
    Else:
        Return "Invalid position number"

    # Step 3: Calculate the height of the feed
    feed_height = baseline_depth - depth_measurement

    # Step 4: Calculate the volume in cubic centimeters (cm³)
    volume_cm3 = feed_height * width * length

    # Step 5: Convert the volume to liters (since 1 liter = 1000 cm³)
    volume_liters = volume_cm3 / 1000

    # Step 6: Return the calculated volume in liters
    Return volume_liters
```

Figure 12: Pseudocode for the volume calculation algorithm

4.2.1.2. Weight data

The approach to weight data manipulation varied depending on the prediction target. For meal weight prediction, no manipulation was applied to the data; the raw values from the weight scale were used directly. This approach preserved the actual measured weights, allowing for precise meal consumption analysis. In contrast, for weight decrease prediction, as with volume decrease, a LOWESS algorithm was implemented. This smoothing technique was applied to reduce noise and highlight underlying trends in the weight data over time. The LOWESS smoothing used a fraction value of 0.05, which was the same fraction used for volume data smoothing. This value was determined to be the minimal value

that provided effective smoothing without over-simplifying the data trends for both weight and volume. The consistent use of this parameter across both weight and volume data ensured a uniform approach to trend identification. This smoothing was particularly useful for weight and volume decrease predictions, where the focus was on identifying general patterns rather than capturing instantaneous fluctuations. The differentiated approach - using raw data for meal weight and smoothed data for weight decrease - allowed for tailored analysis methods that best suited each prediction task's specific requirements.

4.2.1.3. Weather Data

Weather data were collected from two sources: a weather station positioned in the cowshed and the Meteorological Center in Bet Dagan, located 2,100 meters from the cowshed. This dual-source approach was employed to ensure comprehensive coverage of both local and regional weather conditions.

The cowshed weather station recorded pressure, temperature, and humidity every two minutes, providing high-resolution data on the immediate environment of the cows. This local measurement was crucial for capturing farm-specific conditions, which might differ from regional weather due to factors such as proximity to humid areas like the cows' ground or the influence of the cowshed roof.

Data from the Meteorological Center in Bet Dagan, an official government meteorological station, were collected every 10 minutes. These data included:

- Barometric pressure (hPa)
- Relative humidity
- Grass temperature
- Wind direction
- Wind gust direction
- Wind speed (m/sec)
- Maximum 1-minute wind speed
- 10-minute average wind speed (Ws10mm)
- Maximum wind speed
- Wind direction standard deviation
- Mean, maximum, and minimum temperatures

The Bet Dagan station, being comparatively close to the farm, provided reliable regional weather data to complement our on-site measurements.

The combination of these data sources allowed us to account for both micro-climate variations within the cowshed and broader weather patterns that might influence cow behavior or feed properties. This approach enhances the accuracy of our weather-related analyses by providing a comprehensive view of the environmental conditions.

The full code for data collection, processing, and analysis, along with the complete weather dataset used in this research, is detailed in Appendix A.

4.2.2. Data Preparation

During the data preparation phase, the following approach was adopted to refine the raw data collected from the Bet Dagan meteorological station and an Arduino-based weather station developed specifically for this study. The data encompassed weight and volume measurements related to feed consumption, along with environmental parameters, such as temperature, humidity, and atmospheric pressure.

4.2.2.1. Feed Volume and Weight Decrease Predictions

4.2.2.1.1. Time Frame Segmentation

The dataset was segmented into two-hour time frames to facilitate a detailed analysis of the variations (Figure 13). Each subsequent frame commenced ten minutes after the preceding frame, enabling a continuous data stream. This segmentation allowed the capture of nuanced changes in environmental conditions and feed metrics throughout the day.

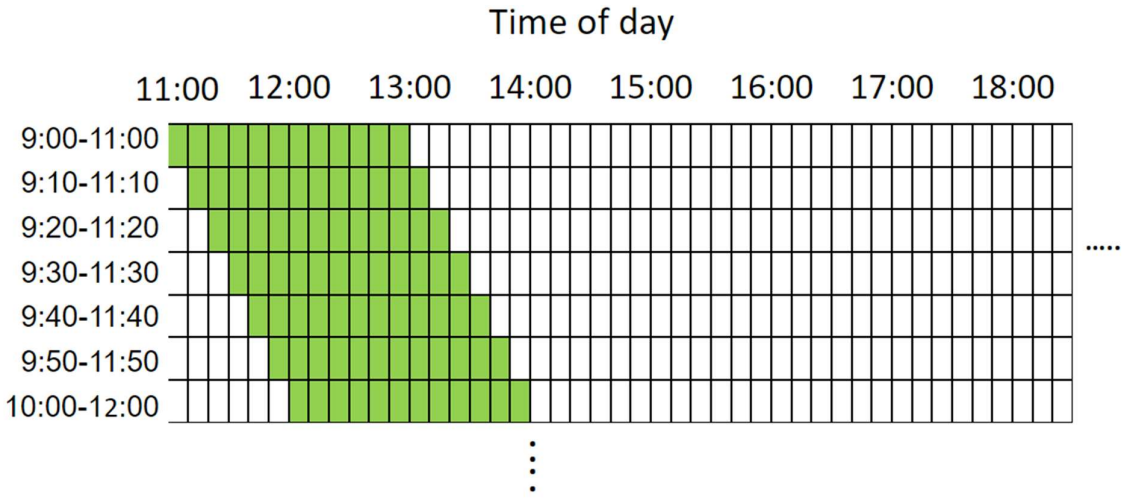

Figure 13: Illustration of the overlapping time frames used for data segmentation. Each green strip represents a two-hour time frame with a ten-minute overlap between consecutive frames.

4.2.2.1.2. Mean Calculation for Weather and Feed Metrics

For every time frame, the average of the measurements taken throughout the entire duration of the time frame was calculated for the weather parameters, providing a consolidated view of the environmental conditions (Figure 14). Similarly, for feed-related measurements, the mean of the first and last 15 recordings, equivalent to the initial and final five minutes of each frame, was calculated. This strategy aims to reduce noise in the data by focusing on stable periods, thus minimizing the influence of transient fluctuations in the feed weight and volume.

```python
def calculate_timeframe_variables(weather_data, volume_data, weight_data):

    volume_decrese_prediction_data = []
    weight_decrese_prediction_data = []

    for feeding_day in feeding_days:
        feeding_day_time_start = feeding_day.set_time(11,00,00)
        feeding_day_time_end = feeding_day.add_days(1).set_time(06,00,00)
        time_frame_start = feeding_day_time_start
        time_frame_end = time_frame_start.add_hours(2)
        while time_frame_end <= feeding_day_time_end:
            relevant_weather_data = weather_data[time_frame_start < measure_datetime < time_frame_end]
            relevant_volume_data = volume_data[time_frame_start < measure_datetime < time_frame_end]
            relevant_weight_data = weight_data[time_frame_start < measure_datetime < time_frame_end]

            #measure mean of weather data throughout the timeframe
            time_frame_weather_data = relevant_weather_data.mean()

            #measure time since 11 o'clock
            minutes_since_eleven = (feeding_day_time_start - time_frame_start).minutes()
            #measure time since start of feeding day
            minutes_since_start = (relevant_weather_data['datetime].min() - time_frame_start).minutes()

            #calculate weight decrease in percents
            initial_weight = relevant_weight_data.first(15).mean()
            final_weight = relevant_weight_data.last(15).mean()
            percent_decrese_weight = (initial_weight - final_weight) / initial_weight

            #calculate volume decrease in percents
            initial_volume = relevant_volume_data.first(15).mean()
            final_volume = relevant_volume_data.last(15).mean()
            percent_decrese_volume = (initial_volume - final_volume) / initial_volume

            volume_time_frame_prepared_data = {time_frame_weather_data, minutes_since_eleven, minutes_since_start, percent_decrese_volume}
            weight_time_frame_prepared_data = {time_frame_weather_data, minutes_since_eleven, minutes_since_start, percent_decrese_weight}

            #add time frame calculated data to previous calculated time frames
            volume_decrese_prediction_data.append(volume_time_frame_prepared_data)
            weight_decrese_prediction_data.append(weight_time_frame_prepared_data)

    return volume_decrese_prediction_data, weight_decrese_prediction_data
```

Figure 14: Python function 'calculate_timeframe_variables' processes weather, volume, and weight data within 2-hour time frames. It calculates mean weather data, minutes since 11:00 AM, minutes since the start of feeding, and percent decreases in volume and weigh

4.2.2.1.3. Variables for Volume and Weight decrease prediction models

Table 1: Variables for volume decrease and weight decrease prediction models

Variable	Explanation
hour	Hour of the day, recorded at the start of the timeframe.
minutes_since_start	Minutes elapsed since the start of observation, recorded at the start of the timeframe.
minutes_since_eleven	Minutes elapsed since 11 o'clock, recorded at the start of the timeframe.
baro_press_hPa	Barometric pressure in hectopascals, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
rel_humidity	Relative humidity, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
temp_grass	Grass temperature, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_dir	Wind direction, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_dir_gust	Wind gust direction, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_sp_m_sec	Wind speed in meters per second, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_sp_m_1m_max	Maximum wind speed in one minute, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
Ws10mm	Wind speed averaged over 10 minutes, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_sp_m_sec_max	Maximum wind speed, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
wind_dir_std	Standard deviation of wind direction, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
temp_mean	Mean temperature, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
temp_max	Maximum temperature, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
temp_min	Minimum temperature, averaged over a 2-hour timeframe. (Meteorological center, Beit Dagan)
pressure	General air pressure, averaged over a 2-hour timeframe. (Weather station in cowshed)
temperature	Ambient temperature, averaged over a 2-hour timeframe. (Weather station in cowshed)
humidity	Humidity levels, averaged over a 2-hour timeframe. (Weather station in cowshed)

Table 2: Target variables for volume decrease and weight decrease prediction models

Target Variable	Explanation
Volume Decrease	The reduction in the volume of unconsumed feed, measured over a 2 hours timeframe.
Weight Decrease	The reduction in the weight of unconsumed feed, measured over a 2 hours timeframe.

4.2.2.2. Meal Weight Prediction

4.2.2.2.1. Cow Presence and Meals Duration

To determine whether a cow was present and to identify the start and end of a meal in our study, we employed the YOLOv8 model, which was trained to recognize cow heads. This model works by identifying the position of a cow's head in an image and marking its location with bounding box coordinates after each prediction. We decided that a cow was considered present at the feeding station if its entire head fell within the boundaries of the weight-scale area in the captured image. To ensure that we captured every significant moment when the cow was present, an RGB image was acquired every 20 seconds.

To pinpoint the start of a meal, we looked for two consecutive images showing the cow's presence at the feeding station, as indicated by our head-detection criterion. The meal's start time was then recorded as the moment of the first image where the cow was detected. This approach helped us to accurately track when a cow began eating. Similarly, we determined the end of a meal by identifying two consecutive images in which the head of the cow was not detected in the feeding area. A meal was considered to have ended at the time of the first image where the cow was no longer present.

4.2.2.2.2. Variables Preparation

Weather, meal volume, and time data were used to calculate the variables necessary for meal-weight prediction. Weather data were summed and averaged from the start of the feeding day until the start of the meal. Meal duration and the amount of time that had passed since the beginning of feeding were also factored in, along with meal volume.

To determine meal weight, the last three measurements before the meal and the first three measurements after the meal were considered (Figure 15). The initial weight was calculated by averaging the last three measurements before the meal, whereas the final weight was determined by averaging the first three measurements after the meal. In the same way the volume was calculated.

To select meals for the analysis, rows with missing data (NaN values) were removed. Meals with a volume greater than 0.1 liters and a weight between 1 kg and 30 kg were then chosen. The z-score was computed from the remaining dataset, and only meals with z-scores within ±3 were retained. In total, 365 meals were selected for further analysis. The lightest meal weighed 1.01 kg, whereas the heaviest meal weighed 29.67 kg.

```python
def calculate_meals_volume_and_weight(volumes, weights, meal_start_time, meal_end_time):
    # keep only 3 measurement before and after the meal
    volumes_before = volumes['datetime' > meal_start_time][-3:]
    volumes_after = volumes[meal_end_time < 'datetime'][:3]
    weights_before = weights['datetime' > meal_start_time][-3:]
    weights_after = weights[meal_end_time > 'datetime'][:3]

    init_volume = weights _before.mean()
    final_volume = volumes_after.mean()
    init_weight = weights_before.mean()
    final_weight = weights_after.mean()

    meal_volume = init_volume - final_volume
    meal_weight = init_weight - final_weight

    return meal_volume, meal_weight
```

Figure 15: Python function 'calculate_meals_volume_and_weight' calculates the meal volume and weight. It analyzes the last three pre-meal and first three post-meal volume and weight measurements to determine the initial and final values, and then returns the difference as meal volume and weight.

4.2.2.2.3. Variables for Meal Weight prediction models

Table 3: Variables for meal weight prediction models

Variable	Explanation
meal_volume	The volume of the meal consumed by the cow (in liters).
meal_duration	The duration of the meal (in seconds).
seconds_since_start	The time elapsed since the start of the feeding day (in seconds).
baro_press_hPa_mean	The average barometric pressure from the start of the feeding day until the meal start (in hectopascals) from Beit Dagan meteorological station.
rel_humidity_mean	The average relative humidity from the start of the feeding day until the meal start (percentage) from Beit Dagan meteorological station.
temp_mean_mean	The average of the mean temperature readings from the start of the feeding day until the meal start (in degrees Celsius) from Beit Dagan meteorological station.
temp_max_mean	The average of the maximum temperature readings from the start of the feeding day until the meal start (in degrees Celsius) from Beit Dagan meteorological station.
temp_min_mean	The average of the minimum temperature readings from the start of the feeding day until the meal start (in degrees Celsius) from Beit Dagan meteorological station.
temp_grass_mean	The average temperature at grass level from the start of the feeding day until the meal start (in degrees Celsius) from Beit Dagan meteorological station.
wind_dir_mean	The average wind direction from the start of the feeding day until the meal start (in degrees) from Beit Dagan meteorological station.
wind_dir_gust_mean	The average wind direction during gusts from the start of the feeding day until the meal start (in degrees) from Beit Dagan meteorological station.
wind_sp_m_sec_mean	The average wind speed from the start of the feeding day until the meal start (in meters per second) from Beit Dagan meteorological station.
wind_sp_m_1m_max_mean	The average maximum wind speed at 1 meter height from the start of the feeding day until the meal start (in meters per second) from Beit Dagan meteorological station.
Ws10mm_mean	The average wind speed at 10 meters height from the start of the feeding day until the meal start (in meters per second) from Beit Dagan meteorological station.
wind_sp_m_sec_max_mean	The average of the maximum wind speed readings from the start of the feeding day until the meal start (in meters per second) from Beit Dagan meteorological station.
wind_dir_std_mean	The standard deviation of wind direction from the start of the feeding day until the meal start (in degrees) from Beit Dagan meteorological station.
rain_mm_mean	The average rainfall from the start of the feeding day until the meal start (in millimeters) from Beit Dagan meteorological station.
baro_press_hPa_sum	The sum of barometric pressure readings from the start of the feeding day until the meal start (in hPa) from Beit Dagan meteorological station.
rel_humidity_sum	The sum of relative humidity readings from the start of the feeding day until the meal start (percentage) from Beit Dagan meteorological station.
temp_mean_sum	The sum of mean temperature readings from the start of the feeding day until the meal start (in °C) from Beit Dagan meteorological station.
temp_max_sum	The sum of maximum temperature readings from the start of the feeding day until the meal start (in °C) from Beit Dagan meteorological station.
temp_min_sum	The sum of minimum temperature readings from the start of the feeding day until the meal start (in °C) from Beit Dagan meteorological station.
temp_grass_sum	The sum of temperature readings at grass level from the start of the feeding day until the meal start (°C) from Beit Dagan meteorological station.
wind_dir_sum	The sum of wind direction readings from the start of the feeding day until the meal start (in degrees) from Beit Dagan meteorological station.
wind_dir_gust_sum	The sum of wind direction during gusts from the start of the feeding day until the meal start (in degrees) from Beit Dagan meteorological station.

wind_sp_m_sec_sum	The sum of wind speed readings from the start of the feeding day until the meal start (in meters per second) from Beit Dagan meteorological station.
wind_sp_m_1m_max_sum	The sum of max wind speed at 1 meter height from the start of the feeding day until the meal start (m/s) from Beit Dagan meteorological station.
Ws10mm_sum	The sum of wind speed readings at 10 meters height from the start of the feeding day until the meal start from Beit Dagan meteorological station.
wind_sp_m_sec_max_sum	The sum of maximum wind speed readings from the start of the feeding day until the meal start (in m/s) from Beit Dagan meteorological station.
wind_dir_std_sum	The sum of standard deviation of wind direction from the start of the feeding day until the meal start from Beit Dagan meteorological station.
rain_mm_sum	The sum of rainfall readings from the start of the feeding day until the meal start (in millimeters) from Beit Dagan meteorological station.
temperature_mean	The average temperature during the meal (in degrees Celsius) from the weather station in the cowshed.
humidity_mean	The average humidity during the meal (percentage) from the weather station in the cowshed.
pressure_mean	The average atmospheric pressure during the meal (in hectopascals) from the weather station in the cowshed.
temperature_sum	The sum of temperature readings from the start of the feeding day until the meal start (in degrees Celsius) from the weather station in the cowshed.
humidity_sum	The sum of humidity readings from the start of the feeding day until the meal start (percentage) from the weather station in the cowshed.
pressure_sum	The sum of atmospheric pressure readings from the start of the feeding day until the meal start (in hPa) from the weather station in the cowshed.

4.2.3. Models

The models selected for evaluation were chosen based on their characteristics and suitability for regression tasks, as detailed below.

Linear Regression Models: Served as baseline models owing to their simplicity and interpretability. These models were evaluated with and without feature-selection techniques to assess the impact of variable reduction on performance. The linear regression models employed were regular linear regression as baseline model, and backward and forward stepwise regression for feature selection.

PCA Linear Regression: Investigated its ability to reduce dimensionality while retaining critical information. The aim was to reduce multicollinearity and improve model performance by focusing on the most informative aspects of the data. A linear regression model is then built using these principal components as predictors.

Regularization Models (Ridge, Lasso, and Elastic Net): These models were selected because of their regularization features, which help prevent overfitting, a common concern in predictive modeling. Ridge regression penalizes the sum of squared coefficients, effectively shrinking them to zero to reduce model complexity. Lasso regression penalizes the absolute value of the coefficients, promoting sparsity by potentially setting some coefficients to zero. Elastic Net combines both ridge and lasso penalties, providing a balance that can handle correlated predictors and maintain sparsity.

Tree-Based Models: Decision Tree, Random Forest, Gradient Boosting Regressor: These models were used based on their known characteristics to capture non-linear relationships without requiring transformations of the input data. Their structure allows for an intuitive interpretation of how different features affect the predicted outcome, thus providing valuable insights beyond mere prediction.

4.2.3.1. Parameter Optimization Techniques

Parameter tuning was carried out with the aim of optimizing each model's performance. Each model was fine-tuned using grid-search in a 10-fold cross-validation process. This process was critical in identifying the configurations that provided the best trade-off between bias and variance, ensuring that the models were neither underfitted nor overfitted.

Average results are reported in Appendix H, Appendix N and Appendix T.

Table 4: Models and fine-tuned parameters

Model Name	Parameter	Explanation [tuned values]
Ridge Regression	alpha	Regularization strength; higher values mean more shrinkage. [0.01, 0.1, 1, 10, 100]
Lasso Regression	alpha	Regularization strength; promotes sparsity by setting some coefficients to zero. [0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
Elastic Net	alpha	Regularization strength; balances between ridge and lasso penalties. [0.0001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100]
	l1_ratio	Mix ratio between lasso (L1) and ridge (L2) regularization. [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]
Decision Tree	criterion	Function to measure the quality of a split. ['squared_error', 'friedman_mse', 'absolute_error']
	max_depth	Maximum depth of the tree; limits the number of splits. [None, 10, 20, 30, 40, 50]
	max_features	Number of features to consider when looking for the best split. [None, 'sqrt', 'log2']
	min_samples_leaf	Minimum number of samples required to be at a leaf node. [1, 2, 4]
	min_samples_split	Minimum number of samples required to split an internal node. [2, 5, 10])
	splitter	Strategy used to choose the split at each node. ['best', 'random']
Random Forest	bootstrap	Whether bootstrap samples are used when building trees. [True, False]
	max_depth	Maximum depth of the tree; limits the number of splits. [None, 10, 20, 30]
	max_features	Number of features to consider when looking for the best split. ['auto', 'sqrt', 'log2']

	max_leaf_nodes	Maximum number of leaf nodes in the tree. [None, 10, 20, 50]
	min_impurity_decrease	A node will be split if this split induces a decrease of the impurity greater than or equal to this value. [0.0, 0.01, 0.1]
	min_samples_leaf	Minimum number of samples required to be at a leaf node. [1, 2, 4]
	min_samples_split	Minimum number of samples required to split an internal node. [2, 5, 10]
	n_estimators	Number of trees in the forest. [100, 200, 300]
Gradient Boosting	learning_rate	Shrinks the contribution of each tree by learning rate; affects the speed of convergence. [0.001, 0.01, 0.1, 0.2]
	max_depth	Maximum depth of the individual regression estimators. [3, 4, 5, 7, 9]
	max_features	Number of features to consider when looking for the best split. ['sqrt', 'log2', None]
	min_samples_leaf	Minimum number of samples required to be at a leaf node. [1, 2, 3, 5, 7, 9]
	min_samples_split	Minimum number of samples required to split an internal node. [2, 4]
	n_estimators	Number of boosting stages to be run. [100, 200, 300, 400]

4.2.3.2. Models Evaluation

For each of the task, *Volume Decrease Prediction, Weight Decrease Prediction* and *Meal Weight Prediction*, the development process included feature selection, model selection, robustness tests, and feature importance analysis.

4.2.3.2.1. Feature Selection

For linear models, feature selection was performed using stepwise regression.

- For forward selection, an empty model was the initial state and, in each step, the most significant feature was added until no more significant features were found.
- Similarly, for backward selection, a full set of features was the initial state, and the least non-significant feature was removed until all remaining features were significant.
- For the meal-weight prediction, a regression tree was used for feature selection.

4.2.3.2.2. Model Selection

For each task, the best model was selected based on R^2 and MAE (Mean Absolute Error). Errors were tested on the test data set.

4.2.3.2.3. Robustness Tests

For all models, consistency was assessed through 10-fold cross-validation. A maximal test-set size analysis was conducted to determine the largest test size before the best-performing model's performance significantly decreased. The analysis began with a test set size of 0.9 and progressively reduced it in steps of 0.1, down to 0.1, evaluating model performance at each step. Additionally, a random noise test was performed, where random noises of 1%, 2%, 5%, and 7% were added and subtracted from the target feature to test the robustness of the model to noise.

4.2.3.3. Feature Importance Analysis

To evaluate the importance of each feature, a permutation importance analysis was conducted in which each feature was randomly shuffled in turn, and its effect on the model's performance was assessed. The importance of each feature was determined to understand its contribution to the predictions of the model. For models in which features were not selected, the importance was recorded as 0.

5. Results

5.1. Weight and Volume and Density Analysis

5.1.1. Overview

The differences between consumed and unconsumed feed, specifically how these differences affect volume, weight, and density over time is presented. By measuring both consumed and unconsumed feed, we aim to understand how cows' eating behavior and feed selection influence feed density throughout the day.

5.1.2. Feed Volume and Weight Patterns

For the position within reach of cows (consumed feed), both volume and weight showed a consistent decrease until approximately 19:00, after which they stabilized at near-zero levels (Figure 16). This pattern suggests that cows typically consume their allocated feed by early evening.

In contrast, the position out of reach (unconsumed feed) exhibited different patterns for volume and weight (Figure 17). Weight began to decrease gradually, with a steeper decline observed from around 16:00. Volume, however, showed a moderate decline throughout the day, plateauing around 20:00. These non-linear patterns indicate that changes in unconsumed feed characteristics are time-dependent and influenced by factors beyond mere consumption.

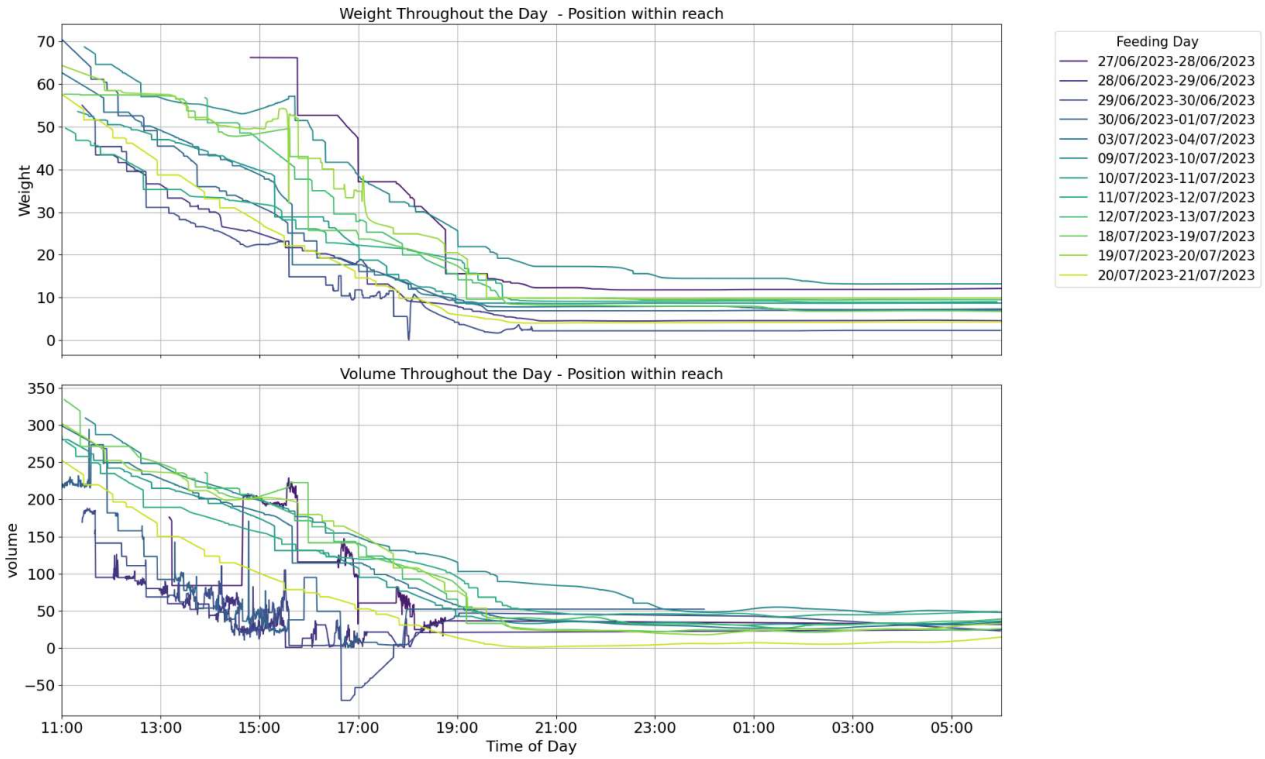

Figure 16: Food volume and weight throughout the day of position within reach

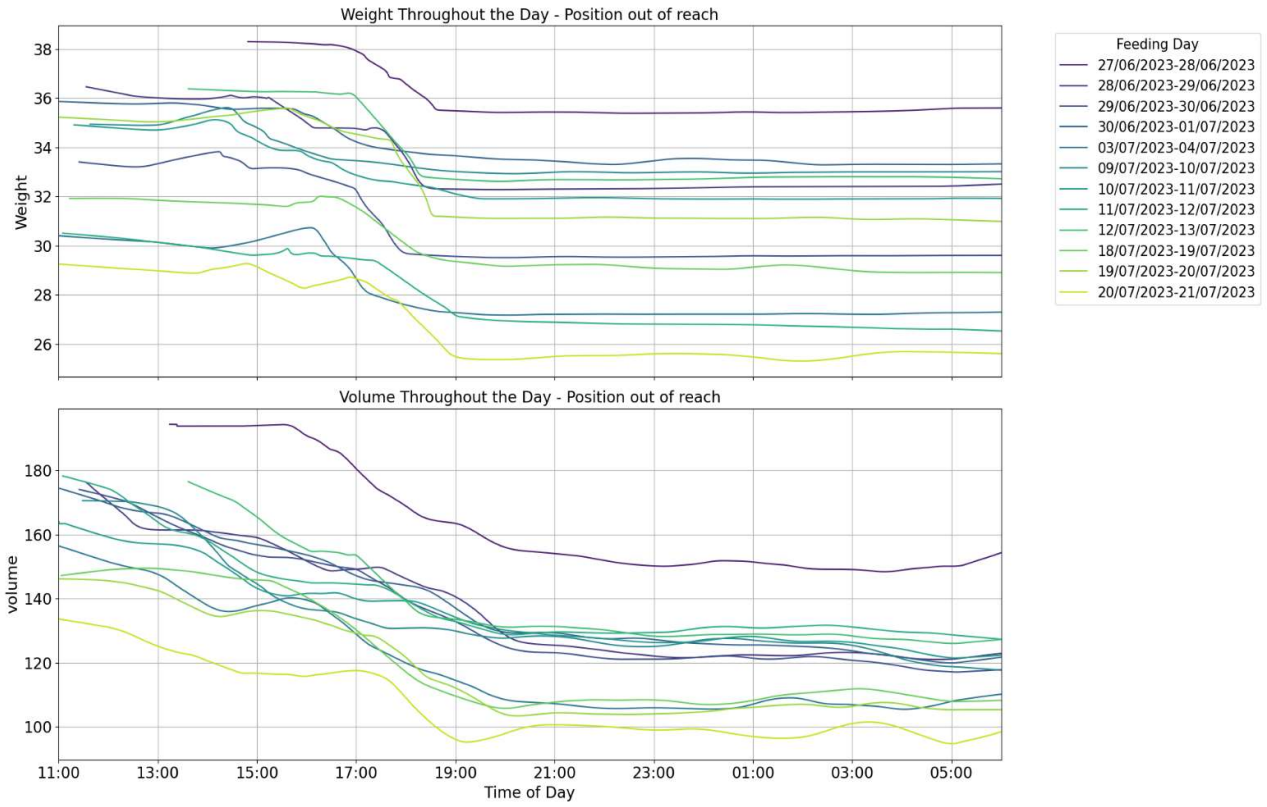

Figure 17: Volume and weight throughout the day of position outside of reach

5.1.3. Feed Density Analysis

Feed density was analyzed by converting it to kg/m³, a standard measure in cattle feed management. For unconsumed feed (Figure 19), density increased until approximately 20:00 before stabilizing. This trend suggests that volume decreases faster than weight in unconsumed feed, possibly due to settling or moisture loss.

Conversely, consumed feed (Figure 18) showed a decline in density over time, indicating that volume decreased more rapidly than weight. This pattern, when compared to unconsumed feed (Figure 20), suggests that cows preferentially consume denser feed components (e.g., grains, corn, wheat) earlier in the feeding period. As consumed feed approached depletion, density measurements became more variable, likely due to the instability of calculations involving near-zero values.

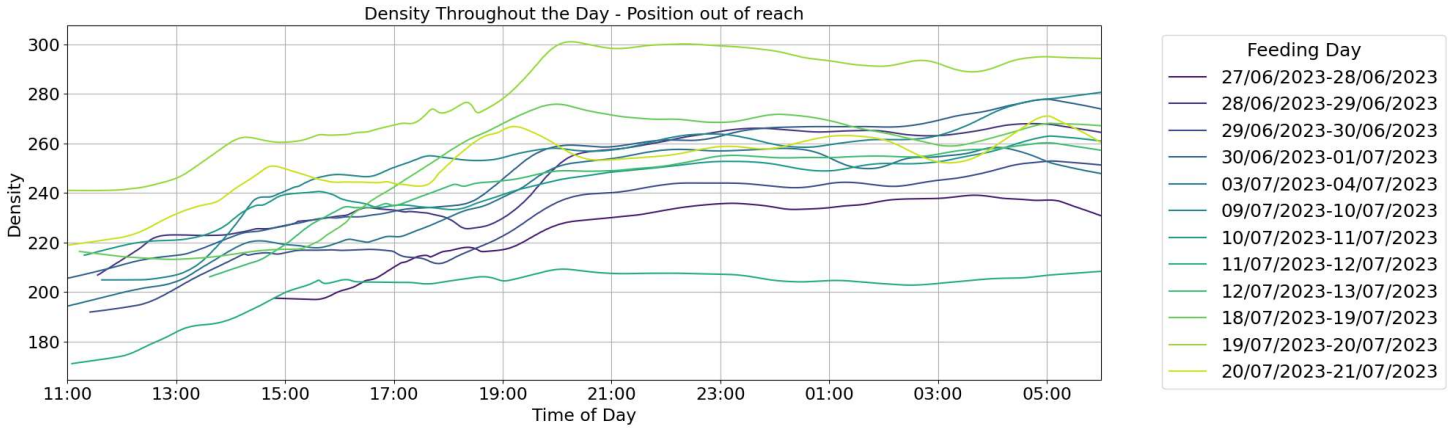

Figure 19: Density throughout the day of position within cow reach

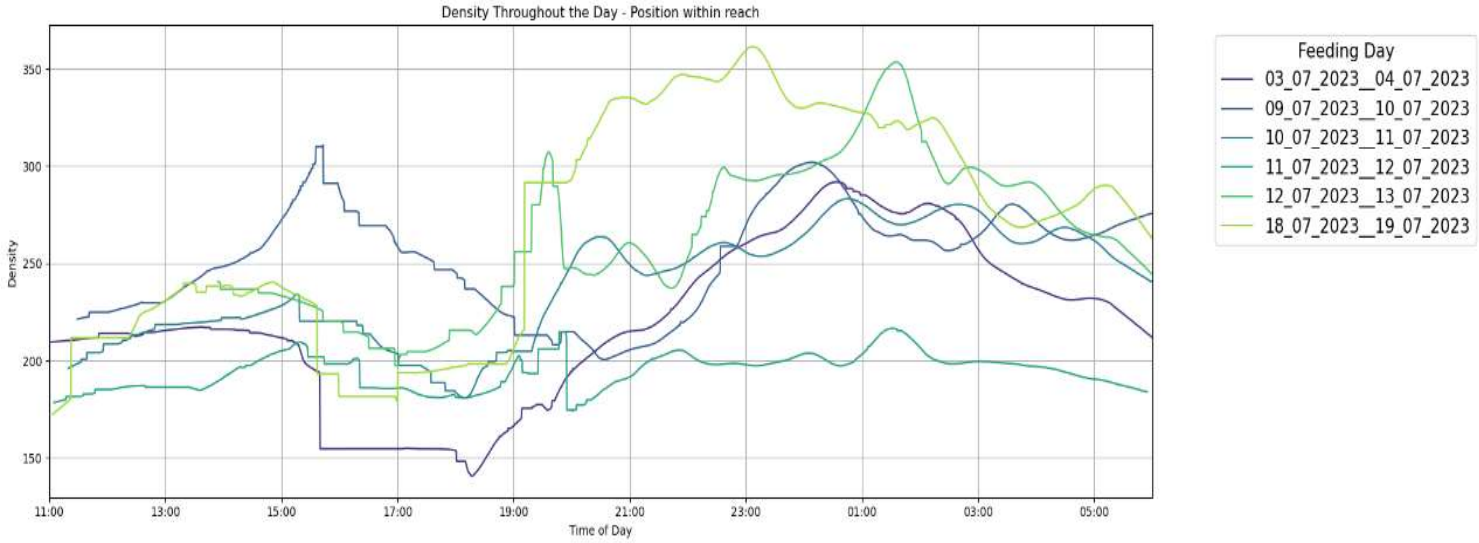

Figure 18: Density throughout the day of position within reach

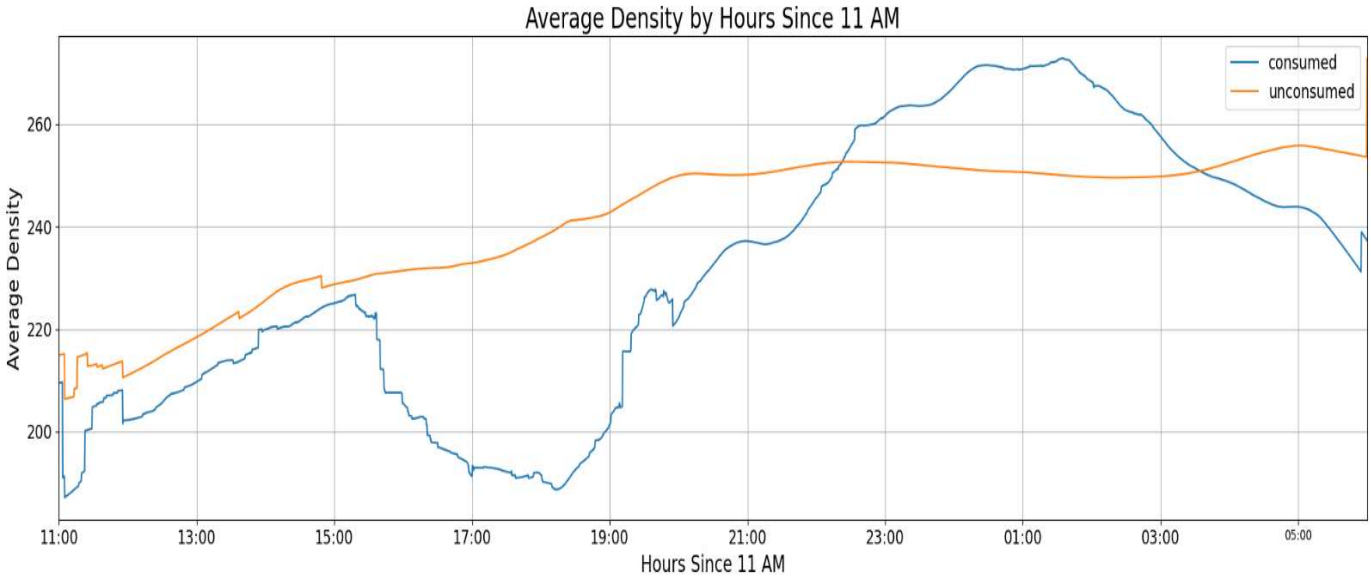

Figure 20: Average density comparison during the feeding day of consumed and unconsumed feed

5.1.4. Noise Reduction through Variance Analysis

The following method was employed to identify extra noisy feeding days to exclude these days from the training data. By plotting the variance of the volume and weight decreases for each day, days with high fluctuations were readily identified and omitted from further analyses (Figure 21). In this analysis feeding days 27/07/2023-28/07/2023 and 31/07/2023-01/08/2023 were removed since they were noisy in weight decrease and in volume decrease.

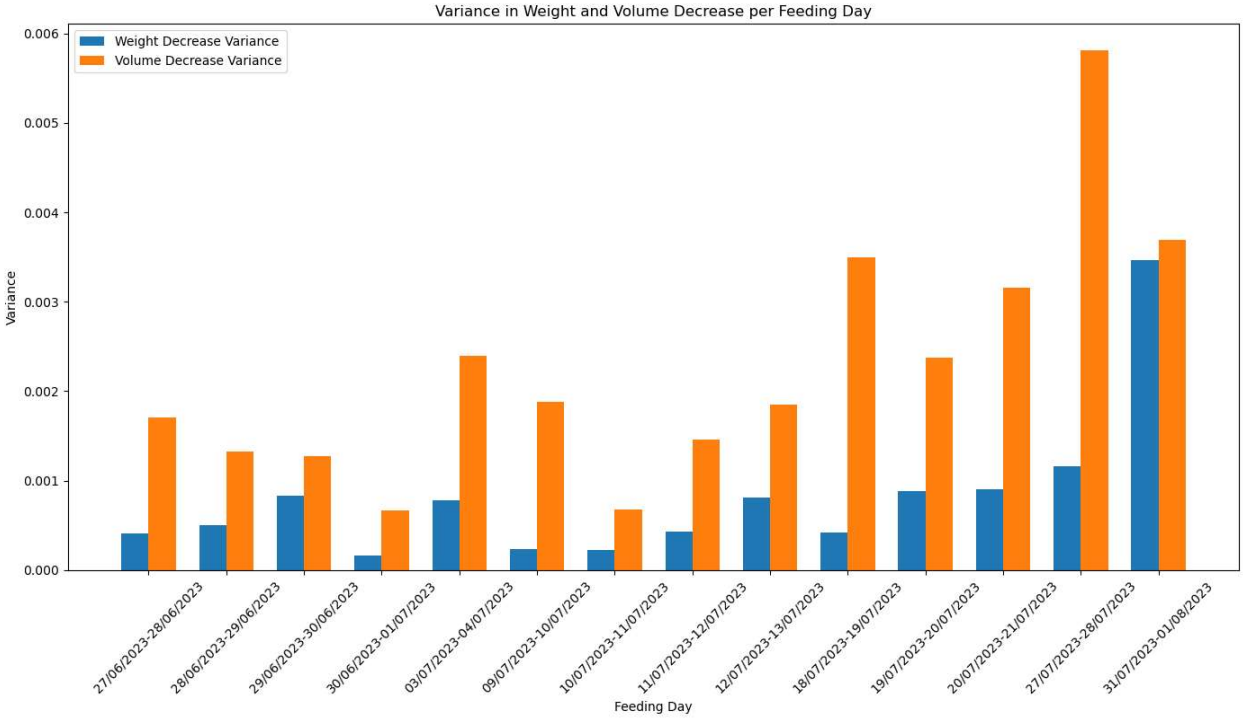

Figure 21: Variance in weight and volume per feeding day

5.1.5. Density Change Analysis

Further refinement was achieved by analyzing the density changes in the feed throughout the day. Frames exhibiting an increase in density of greater than 2.5% or a decrease exceeding 40% were removed. This threshold-based exclusion criterion was designed to eliminate data points likely to be affected by external disruptions or measurement anomalies, thereby preserving the integrity of the analysis related to feed density and consumption.

5.2. Unconsumed Feed - Volume Decrease Prediction

5.2.1. Overview

In this section, we analyze the reduction in feed volume over time, focusing on the impact of environmental factors. By estimating how unconsumed feed volume decreases, we can gain a clearer understanding of how weather conditions, such as humidity and temperature, influence the feed. This helps us to exclude the effects of cow behavior, allowing us to better isolate environmental factors. Volume decrease is measured by measuring volume of the unconsumed feed before and after a two hours time-frame and comparing the decrease in percent during this time-frame. Machine learning models are used to predict volume decrease, and the results are compared to ground truth to identify the most effective model for this task.

5.2.2. Feature Selection

Table 5 presents the features selected through forward and backward stepwise selection for volume decrease prediction.

Table 5. Features selected in forward and backward stepwise selection for volume decrease prediction

Feature	Forward Selection	Backward Selection
hour		V
minutes_since_start		
minutes_since_eleven	V	
baro_press_hPa		
rel_humidity		V
temp_mean		V
temp_max		V
temp_min		
temp_grass	V	V
wind_dir	V	
wind_dir_gust		V
wind_sp_m_sec		
wind_sp_m_1m_max		
Ws10mm	V	
wind_sp_m_sec_max		V
wind_dir_std	V	V
temperature		
humidity	V	V
pressure	V	V

5.2.3. Model Selection

We evaluated machine learning models to predict volume decrease (Figure 22). The Random Forest model showed the best results followed by the Gradient Boosting model with a R^2 score of 0.97285 and 0.970267, and a MAE of 0.004389 and 0.004945.

All models showed ME values close to zero, indicating unbiased predictions on average. The Random Forest model had an ME of -0.000095, Gradient Boosting -0.000003, and Linear Regression -0.000015.

Linear models, including various forms of linear regression, ridge, lasso, elastic net, and PCA-based linear regression, had lower R^2 scores and higher error metrics compared to the tree-based models. Model selection results for Volume Decrease models are presented in Appendix E.

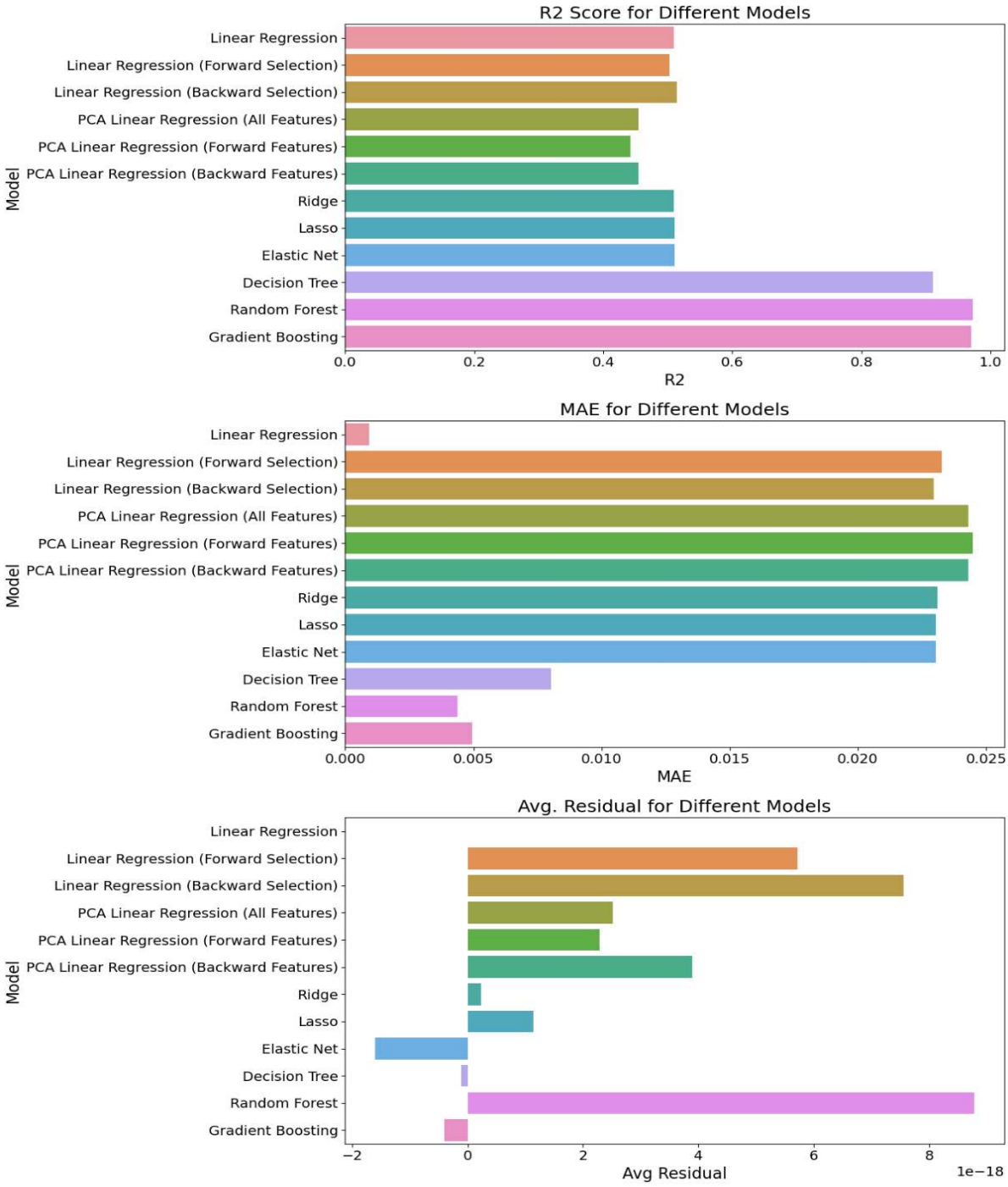

Figure 22: Performance comparison of regression models for feed volume decrease prediction

5.2.4. Robustness Tests

To ensure the reliability of the Random Forest model, we conducted several sensitivity tests, comparing it to the second-best performing model (Gradient Boosting) and the baseline model (Linear Regression):

5.2.4.1. 10-Fold Cross-Validation

The Random Forest model showed consistent performance across folds, with a mean R^2 of 0.973 and a low standard deviation of 0.009405 (Figure 23). This was better from the second-best model, Gradient Boosting, by 0.26% (mean R^2 of 0.970, standard deviation 0.008986). The baseline Linear Regression model achieved a mean R^2 of 0.510, which is 47.6% lower than the Random Forest model.

All three models showed ME values centered around 0, indicating unbiased predictions. The Random Forest model had a mean ME of -0.000095 (std dev: 0.000761), Gradient Boosting had -0.000003 (std dev: 0.000935), and Linear Regression had -0.000015 (std dev: 0.003679). This suggests that while the more complex models (Random Forest and Gradient Boosting) offer higher accuracy and precision, all models, including the baseline Linear Regression, provide unbiased estimates on average. Full cross-validation results are available in Appendix H and Appendix I.

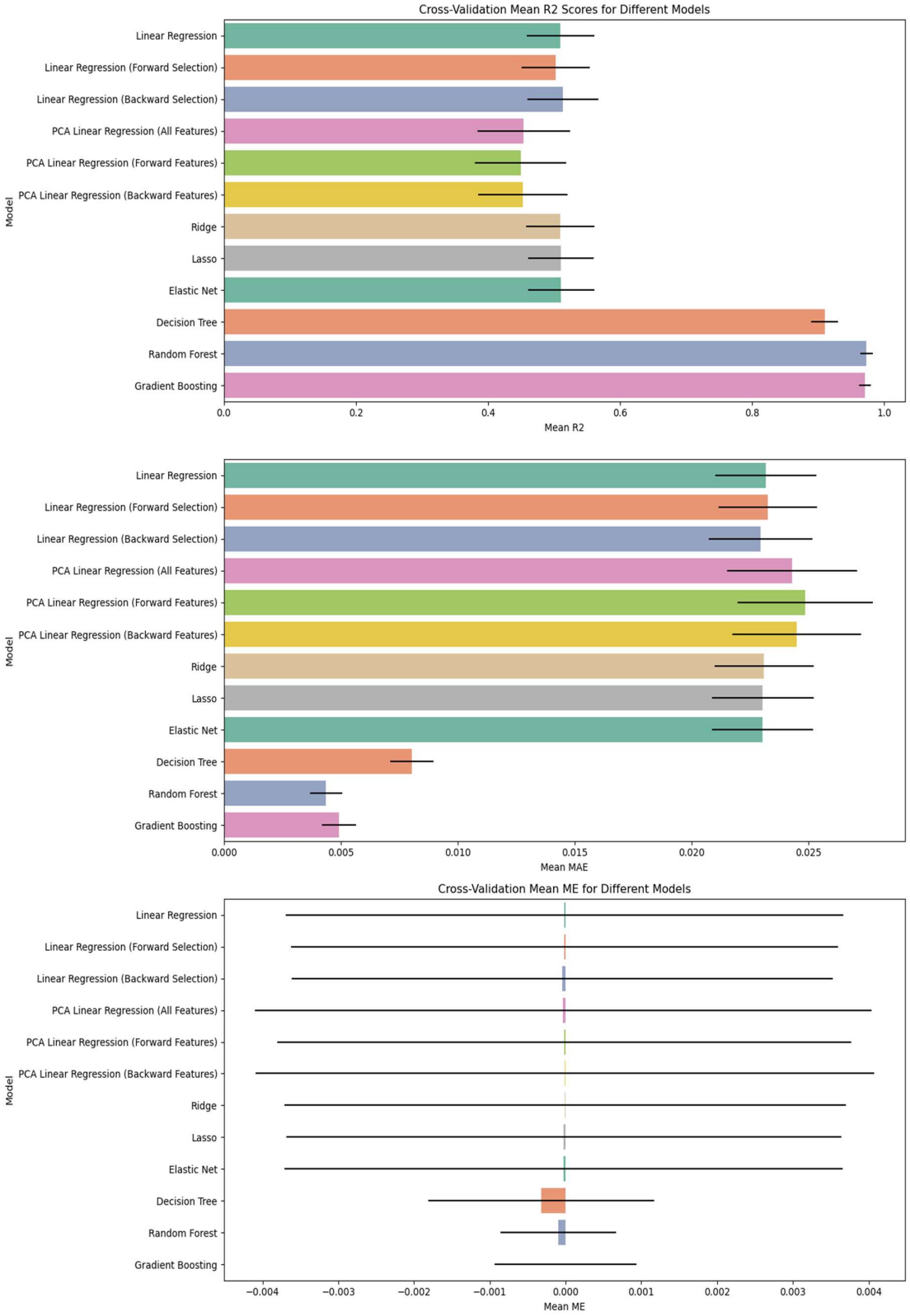

Figure 23: Volume Decrease - test-set size robustness test result

5.2.4.2. Test-Set Size Analysis

In test-set size sensitivity analysis test, the Random Forest and Gradient Boosting models showed similarly robust performance across various test set sizes, consistently surpassing the Linear Regression model results (). Both tree-based models-maintained R^2 values above 0.95 for test set sizes up to 0.3, and above 0.90 for sizes up to 0.6. At a test set size of 0.7, Random Forest (R^2 of 0.846) performed slightly better than Gradient Boosting (R^2 of 0.838) by 0.95%. In contrast, Linear Regression showed significantly lower performance, with R^2 values ranging from 0.397 to 0.493 across all test set sizes. Detailed results are available in Appendix G.

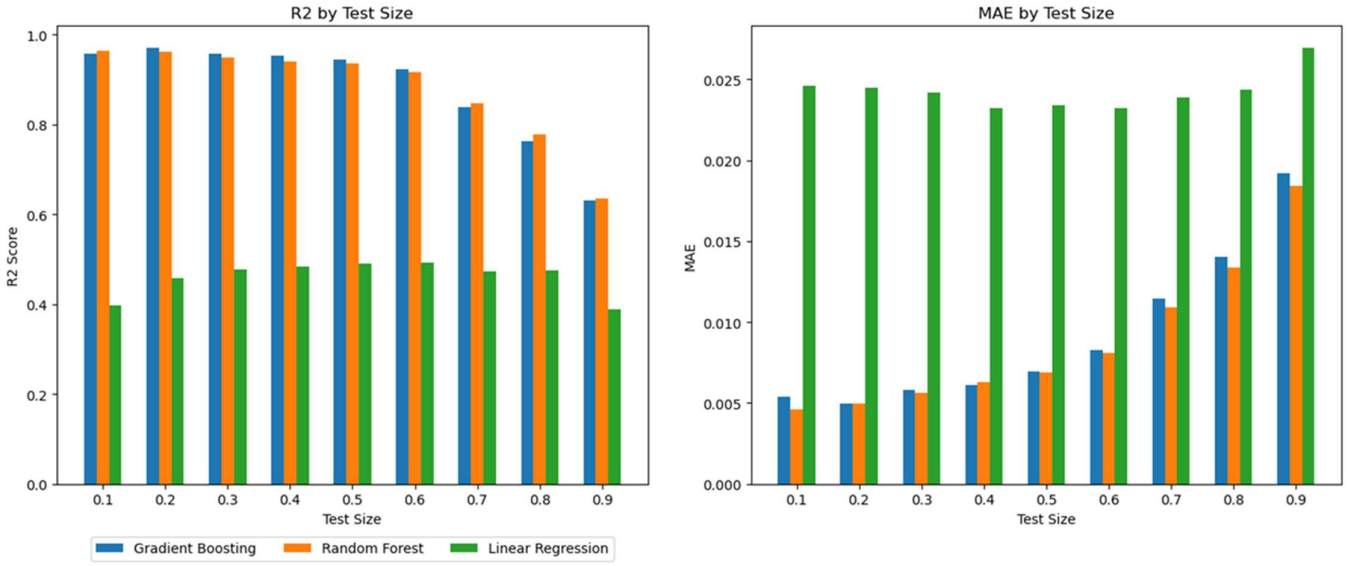

Figure 24: Volume decrease models - test-set size robustness test

5.2.4.3. Random Noise Robustness

The Random Forest and Gradient Boosting models showed similar robustness, with minor increases in MAE as noise levels increased (Table 6). At 10% noise, Random Forest had an MAE of 0.005901, while Gradient Boosting showed an MAE of 0.005782.

The Linear Regression model demonstrated less resilience to noise, with consistently higher MAE values across all noise levels. At 10% noise, it had an MAE of 0.024527, approximately 10 times higher than the tree-based models.

These results indicate that the tree-based models (Random Forest and Gradient Boosting) maintain their predictive accuracy even under noisy conditions, while the Linear Regression model is more susceptible to noise-induced errors.

Table 6: Volume decrease random noise robustness test

Model	Noise (+/-)	MAE
Random Forest	1.0%	0.004944
	2.0%	0.005038
	5.0%	0.005074
	7.0%	0.005377
	10.0%	0.005901
Gradient Boosting	1.0%	0.005128
	2.0%	0.005479
	5.0%	0.005489
	7.0%	0.005720
	10.0%	0.005782
Linear Regression	1.0%	0.024519
	2.0%	0.024482
	5.0%	0.024741
	7.0%	0.024450
	10.0%	0.024527

5.2.5. Feature Importance

The influential factors varied across different model types (Figure 25), In the linear models, temperature-related features were more influential, with mean temperature showing the highest importance in standard linear regression and backward selection. Humidity was the most important feature of PCA-enhanced models. In regularized models such as Ridge, Lasso, and Elastic Net, wind speed (specifically, wind_sp_m_sec_max) and grass temperature were identified as crucial predictors. For tree-based models, the feature "minutes_since_eleven" was identified as an important predictor in

both the Random Forest and Gradient Boosting models, indicating its importance in volume decrease prediction. The importance of wind speed is also notable in these ensemble methods.

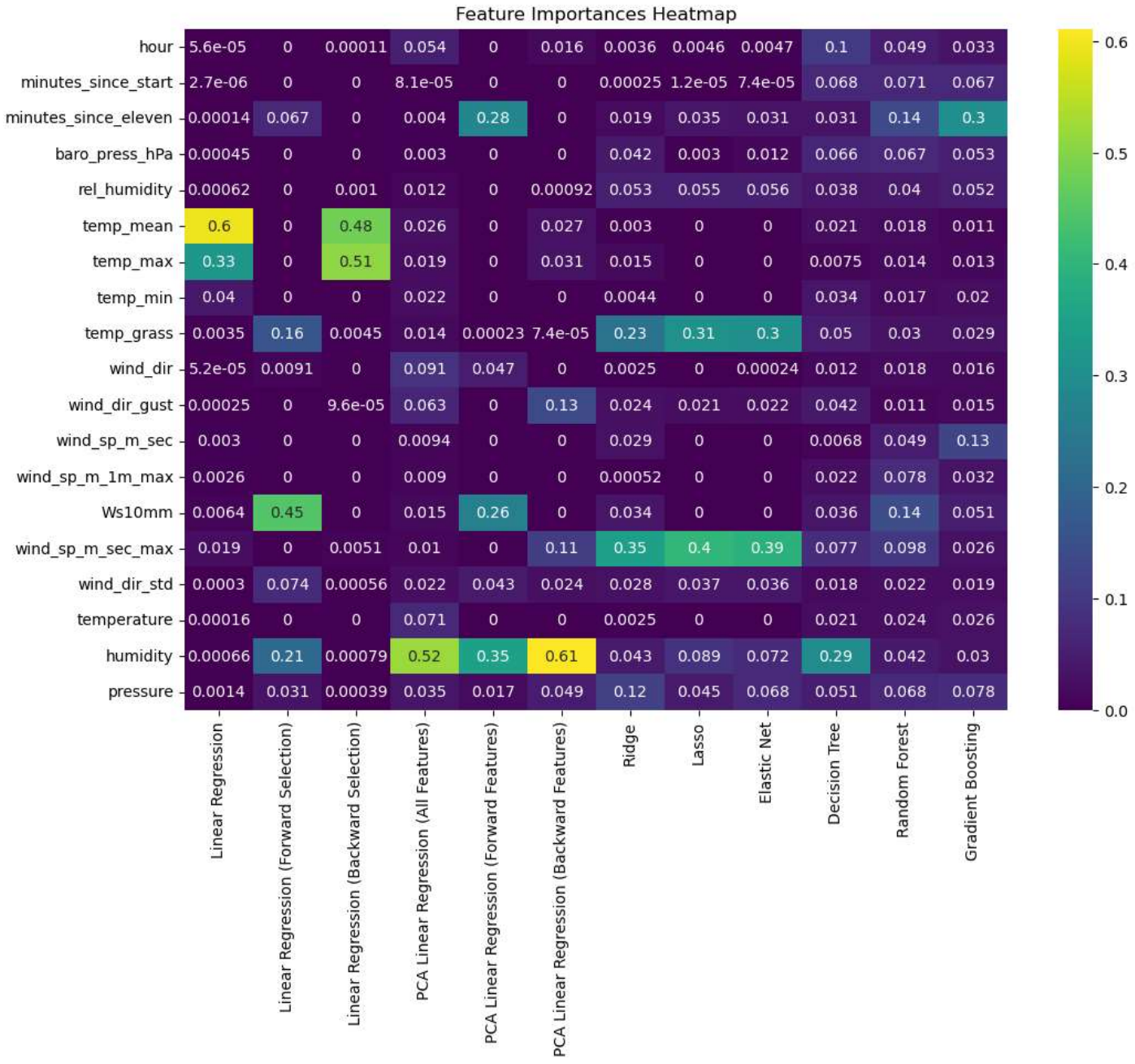

Figure 25: Volume decrease models- feature importance

5.3. Unconsumed Feed - Weight Decrease Prediction

5.3.1. Overview

This section explores the prediction of unconsumed feed weight reduction, emphasizing the role of environmental variables. Estimating weight decrease helps us differentiate between changes caused by external conditions and those resulting from the cows' feeding habits. The focus is on predicting how weather conditions affect the weight of unconsumed feed. Machine learning models are applied to predict weight reduction, and the results are evaluated to determine which model performs best in isolating environmental impacts on feed weight. Feed weight decrease is measured as the decrease, in percentage, over a period of two hours.

5.3.2. Feature Selection

Table 7 presents the features selected through forward and backward stepwise selection for weight decrease prediction.

Table 7: Selected features for weight decrease linear models

Feature	Forward Selection	Backward Selection
hour	V	V
minutes_since_start	V	V
minutes_since_eleven	V	V
baro_press_hPa	V	V
rel_humidity	V	
temp_mean		
temp_max		V
temp_min		V
temp_grass	V	V
wind_dir	V	V
wind_dir_gust		
wind_sp_m_sec		
wind_sp_m_1m_max	V	
Ws10mm		V
wind_sp_m_sec_max		
wind_dir_std	V	V
temperature		V
humidity	V	V
pressure		

5.3.3. Model Selection

Our comparative analysis of machine learning models for weight decrease prediction revealed performance differences (Figure 26). The Random Forest algorithm showed the highest performance, achieving an R^2 score of 0.9739, narrowly edging out Gradient Boosting by 0.04% (R^2 of 0.9741). Both tree-based approaches outperformed the Linear Regression baseline, with Random Forest demonstrating a 130.9% improvement over Linear Regression's R^2 of 0.4218. Regularization techniques like Ridge, Lasso, and Elastic Net offered minimal enhancements over standard Linear Regression, all showed R^2 values around 0.42. All models exhibited ME values close to zero, indicating minimal bias in predictions. Given its robust performance and slight computational advantage, we

selected Random Forest as our primary model for weight decrease prediction. Full Weight Decrease model selection results are available in Appendix K.

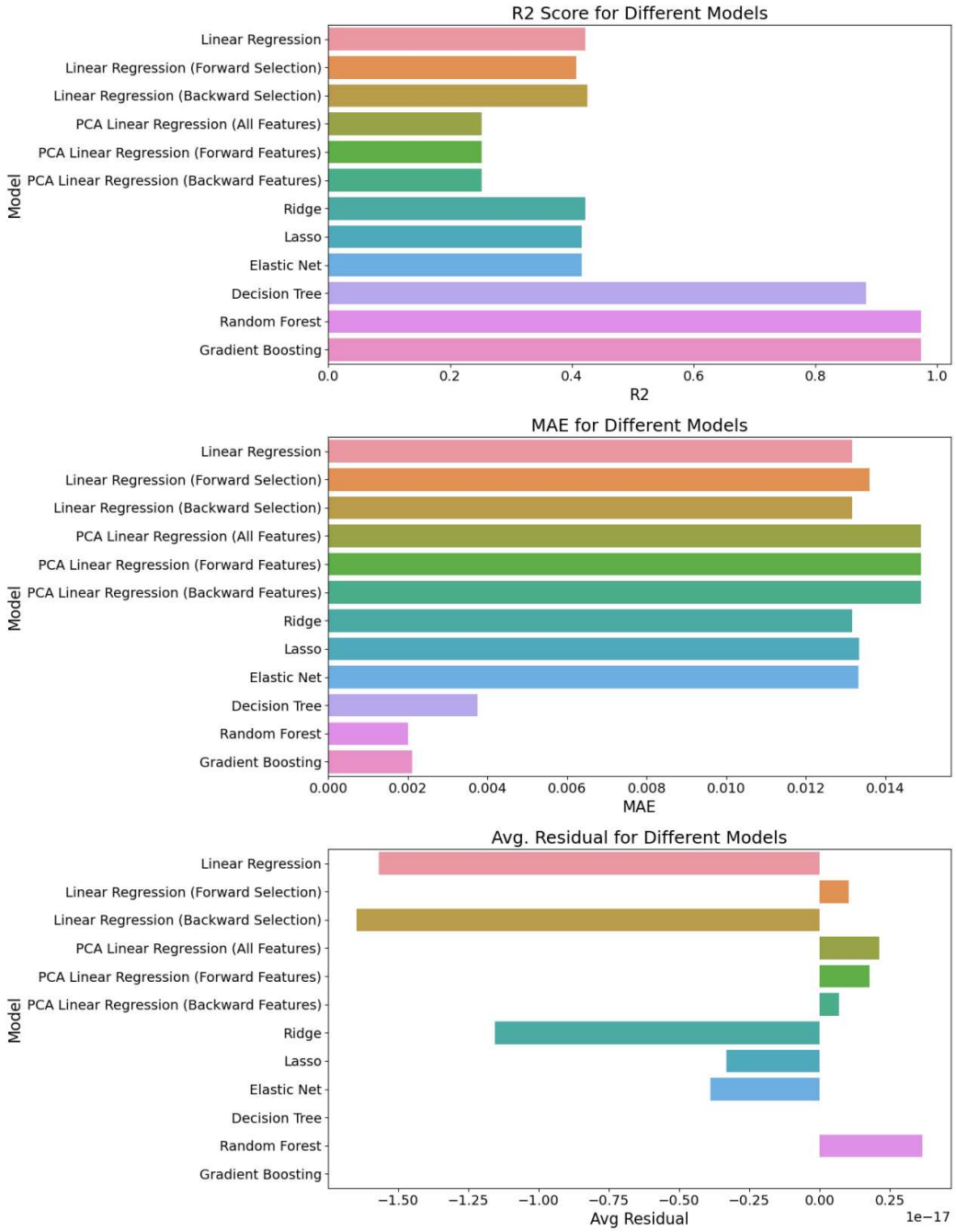

Figure 26: Performance comparison of weight decrease prediction models

5.3.4. Robustness Tests

5.3.4.1. 10-Fold Cross-Validation

The cross-validation process showed consistent performance across models (Figure 27). Random Forest exhibited consistent performance, maintaining a mean R^2 of 0.9737 with minimal fluctuation (standard deviation of 0.0112). Gradient Boosting performed comparably, achieving a mean R^2 of 0.9746, just 0.09% higher, with slightly lower variability (standard deviation of 0.0086). In contrast, Linear Regression significantly underperformed, with a mean R^2 of 0.4218, 56.7% lower than Random Forest, and showed higher variability (standard deviation of 0.0751). This significant performance difference highlights the superior predictive power and consistency of tree-based models in this context. All models maintained ME values clustered near zero: Random Forest at 0.000057, Gradient Boosting at 0.000175 (207% higher), and Linear Regression at -0.000052 (191% lower in absolute terms). These near-zero ME values suggest that despite significant differences in overall accuracy, each model provides relatively unbiased predictions. Full results are available in Appendix N and Appendix O.

.

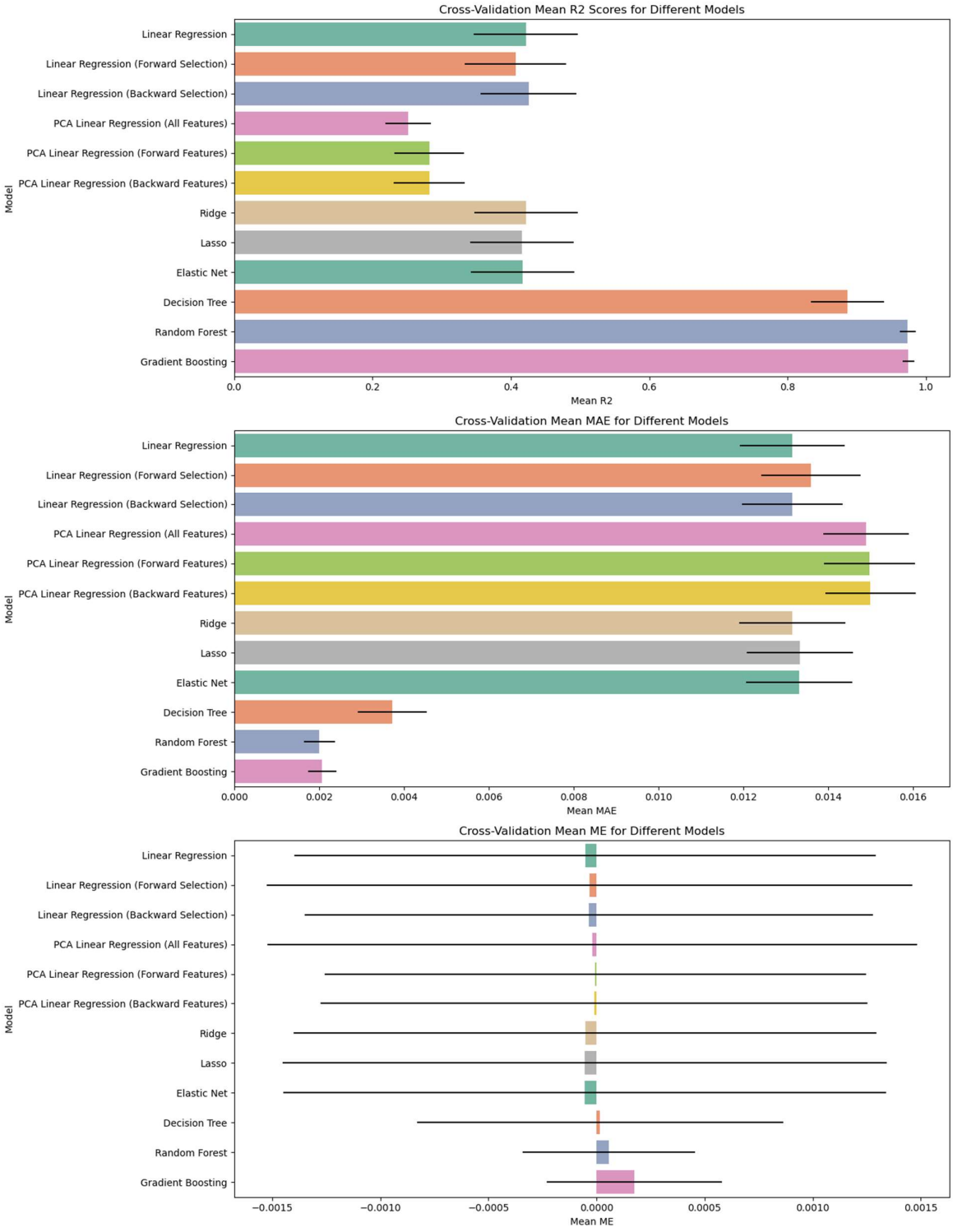

Figure 27: Weight decrease models- 10-fold cross validation results

5.3.4.2. Test Set Size Analysis

Our examination across varying test set proportions showed the generalization capabilities of tree-based models (Figure 28**Error! Reference source not found.**). For a 70% test set, Random Forest maintained a strong R^2 of 0.8461, which was better by 0.95% of Gradient Boosting (R^2 of 0.8381) and surpassing Linear Regression by a substantial 78.7% (R^2 of 0.4729). As test set size increased to 80%, Random Forest maintained higher performance with an R^2 of 0.7784, which was higher than Gradient Boosting by 2.1% (R^2 of 0.7620) and Linear Regression by 63.7% (R^2 of 0.4757). These results demonstrate Random Forest's better adaptability across diverse data split scenarios, consistently surpassing both its nearest competitor and the linear baseline. Detailed findings are presented in Appendix M.

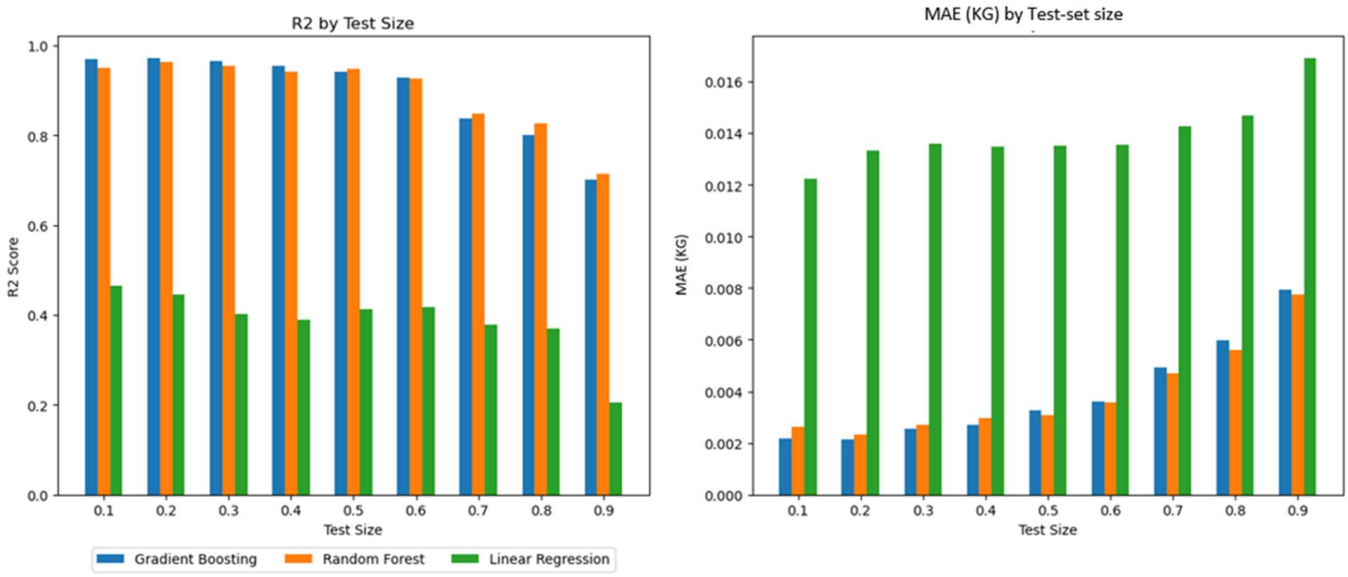

Figure 28: Weight decrease - test-set size robustness test result

5.3.4.3. Random Noise Robustness

In the random noise test (Table 8), at 10% noise, Random Forest recorded an MAE of 0.002638, performing similarly to Gradient Boosting (MAE: 0.002539) and significantly outperforming Linear Regression (MAE: 0.013470). The Random Forest model outperformed Linear Regression by 80.4% in MAE at the highest noise level. Interestingly, Gradient Boosting showed slightly better performance at lower noise levels but converged with Random Forest at higher levels. Both tree-based models demonstrated consistent performance across noise levels, with MAE values remaining below 0.0027 even at 10% noise. In contrast, Linear Regression showed consistently higher error rates, with minimal variation across noise levels. These findings highlight the superior noise tolerance of tree-based models in maintaining predictive accuracy under increasingly challenging conditions.

Table 8: Weight decrease random noise robustness test

Model	Noise (+/-)	MAE
Random Forest	1.0%	0.002403
	2.0%	0.002345
	5.0%	0.002391
	7.0%	0.002620
	10.0%	0.002638
Gradient Boosting	1.0%	0.002227
	2.0%	0.002162
	5.0%	0.002471
	7.0%	0.002415
	10.0%	0.002539
Linear Regression	1.0%	0.013308
	2.0%	0.013298
	5.0%	0.013325
	7.0%	0.013194
	10.0%	0.013470

5.3.5. Feature Importance

Feature importance analysis showed into the factors influencing feed weight decrease across different models (Figure 29). The Gradient Boosting model identified "minutes_since_eleven" as the most important feature, indicating its strong correlation with weight decrease. Wind speed (specifically, "wind_sp_m_sec_max") and temperature-related features were identified as important predictors. In comparison, different models highlight various features to varying extents.

In standard linear regression, "temp_mean" was the most important feature, while models with feature selection (both forward and backward) emphasized "minutes_since_eleven" and "temp_max." PCA linear models showed a more distributed importance among features such as "humidity" and "temperature." Regularized models like Ridge identified "temp_mean" and "temp_grass" as significant, while Lasso highlighted "minutes_since_eleven" and "temp_grass." Elastic Net exhibits a combination of these features, suggesting a balanced approach to feature importance.

Tree-based models provide additional insights. The Decision Tree model highlighted "minutes_since_start" and "minutes_since_eleven" as key features, while the Random Forest model emphasized "minutes_since_eleven" and "wind_sp_m_sec_max" as crucial predictors.

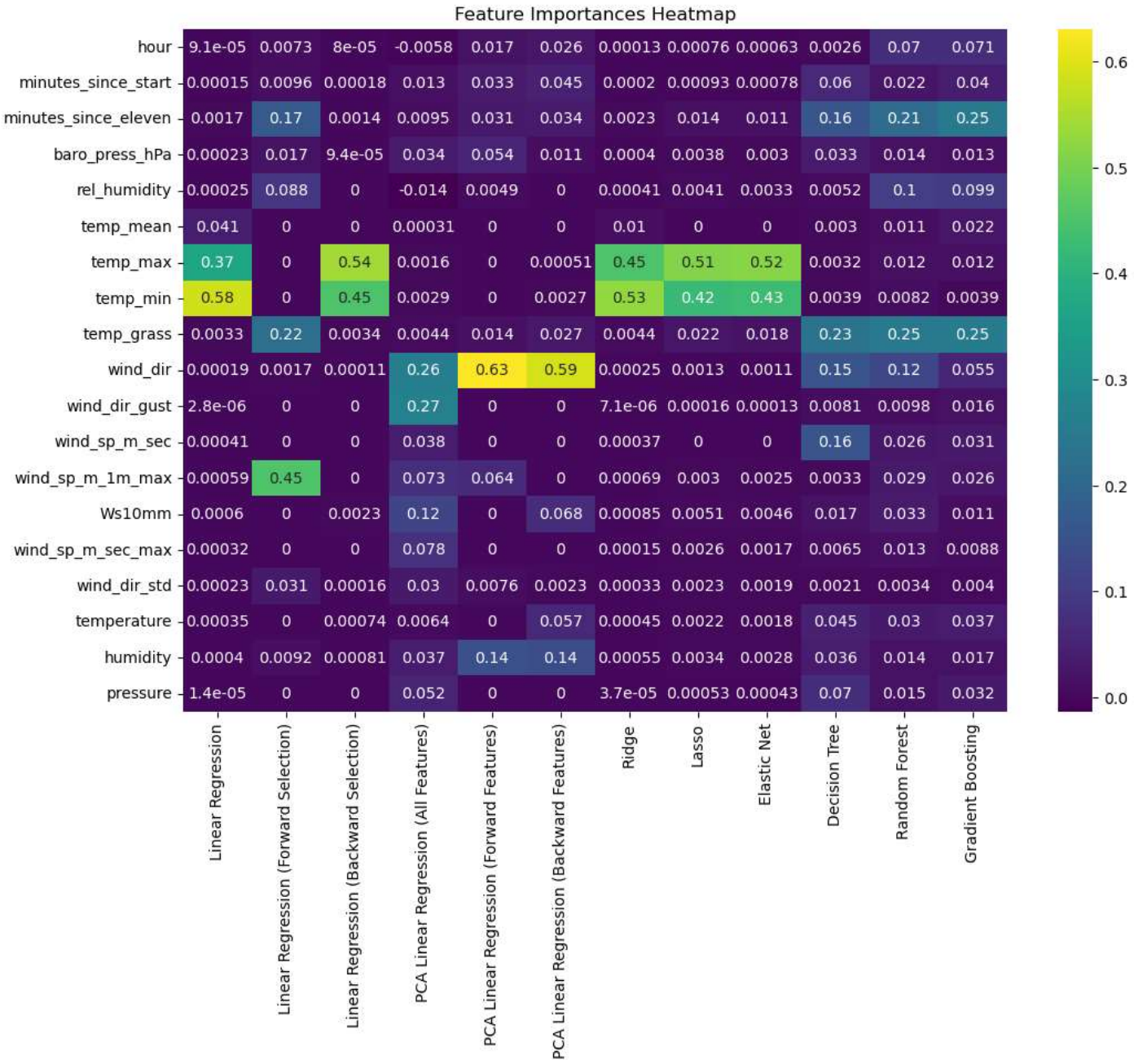

Figure 29: Weight decrease models- feature importance

5.4. Consumed Feed - Meal Weight Prediction

5.4.1. Overview

This section focuses on the prediction of meal weight, which refers to the total feed consumed by individual cows during a meal. Accurate meal weight prediction is important for optimizing feed efficiency and improve breeding of efficient cows. Using data on consumed feed volume and environmental factors, several machine learning models are trained to predict consumed feed weight during a meal. Meal weight is determined as the difference between the feed weight before and after a cow's presence on weight-scale #2.

5.4.2. Feature Selection

Table 9: Selected features for meal weight prediction models

Variable	Forward Selection	Backward Selection	Tree-based Selection
meal_volume	V	V	V
meal_duration			V
seconds_since_start		V	
baro_press_hPa_mean	V	V	
rel_humidity_mean	V	V	
temp_mean_mean		V	
temp_max_mean		V	
temp_min_mean			
temp_grass_mean	V	V	
wind_dir_mean	V	V	
wind_dir_gust_mean		V	
wind_sp_m_sec_mean	V	V	
wind_sp_m_1m_max_mean	V	V	V
Ws10mm_mean	V	V	
wind_sp_m_sec_max_mean	V	V	
wind_dir_std_mean	V		
rain_mm_mean		V	
baro_press_hPa_sum			
rel_humidity_sum			
temp_mean_sum			
temp_max_sum			
temp_min_sum			
temp_grass_sum			
wind_dir_sum			V
wind_dir_gust_sum		V	
wind_sp_m_sec_sum		V	
wind_sp_m_1m_max_sum		V	
Ws10mm_sum		V	
wind_sp_m_sec_max_sum		V	
wind_dir_std_sum			
rain_mm_sum			
temperature_mean	V	V	V
humidity_mean		V	
pressure_mean		V	V
temperature_sum	V		V
humidity_sum	V		
pressure_sum	V		

5.4.3. Model Selection

In the model selection phase (Figure 30), we evaluated various machine learning models to predict meal weight. The Gradient Boosting model showed the highest performance, achieving the highest R^2 of 0.880121 and lowest MAE of 1.546640 KG. Random Forest also showed high performance. Regularization techniques (Ridge, Lasso, Elastic Net) delivered comparable performance. Linear Regression models improved with feature selection, particularly backward selection. PCA Linear Regression models showed lower performance with feature selection. For tree-based models, feature selection produced mixed results. Decision Trees improved with feature selection, while Random Forest and Gradient Boosting benefited from it. The Random Forest model with selected features achieved an R^2 of 0.880683 and MAE of 1.512398 KG. Based on its higher performance across metrics, the Gradient Boosting model without feature selection was chosen as the final model for this analysis.

Figure 31 and Figure 32 highlight important trends in model performance. Figure 31 demonstrates that lower meal weights are associated with higher relative absolute errors. This occurs because, for small values, even minor deviations, such as 100 grams, result in a significant error percentage. However, from 6 kg onward, the relative error stabilizes, indicating that predicting smaller meal weights was considerably more challenging, while larger meal weights are easier to predict with more consistent accuracy.

In Figure 32, which shows the mean relative absolute error for each model, tree-based models performed significantly better than others, maintaining lower average errors across the test set. This pattern aligns with the results of other tests, further confirming the better accuracy of tree-based models in this context.

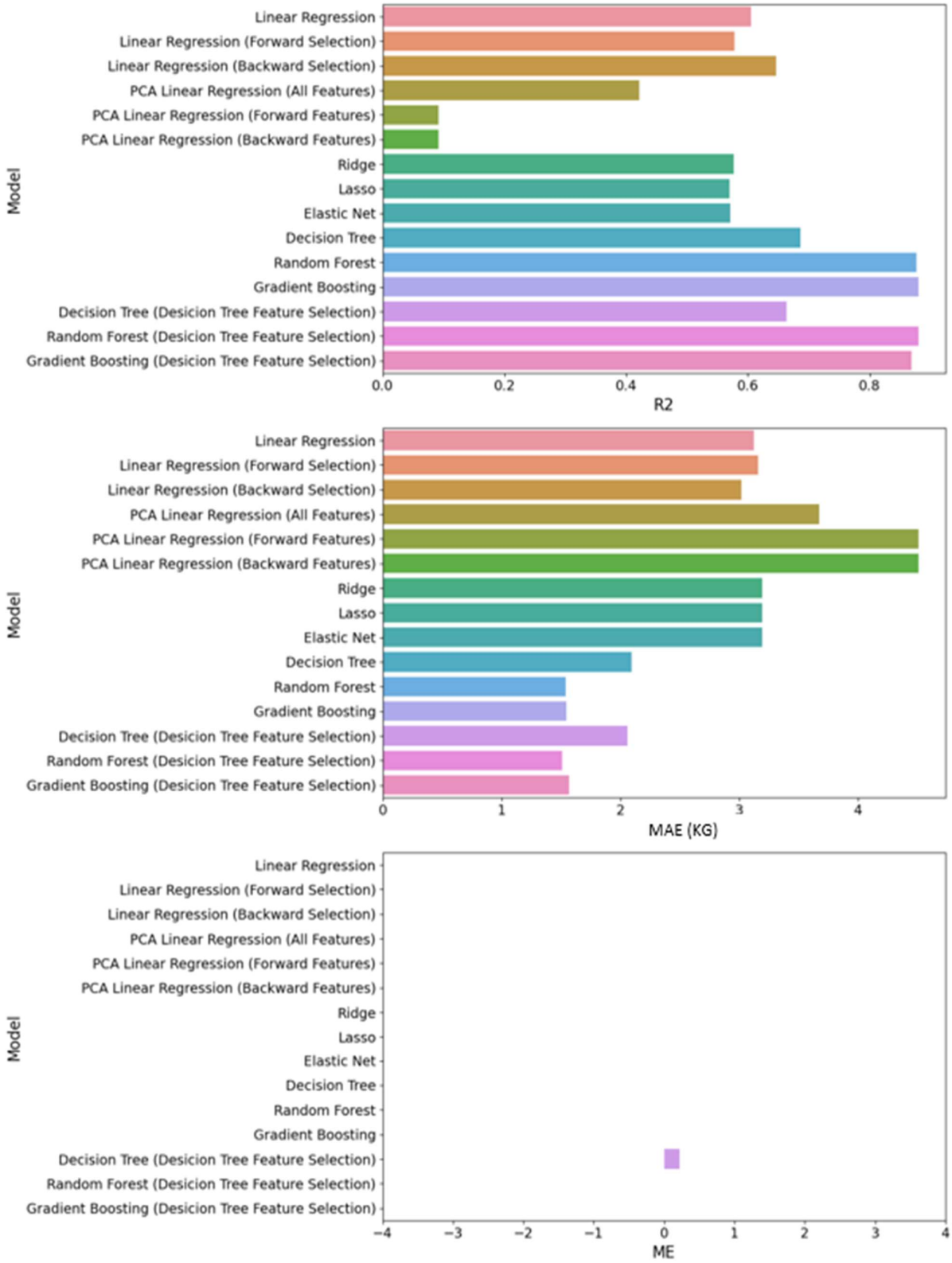

Figure 30: Performance comparison of regression models for meals weight prediction

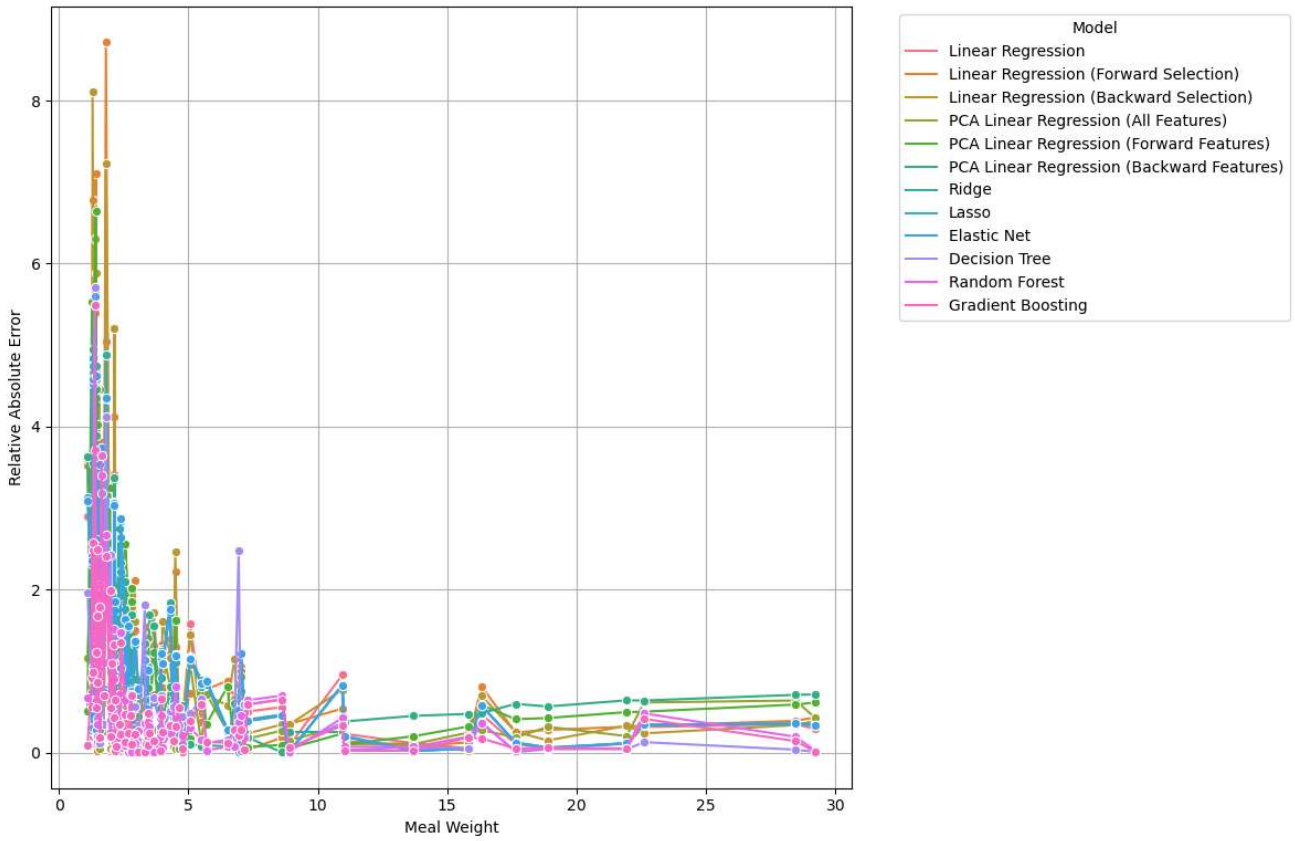

Figure 31: Relative absolute error per sample for all models

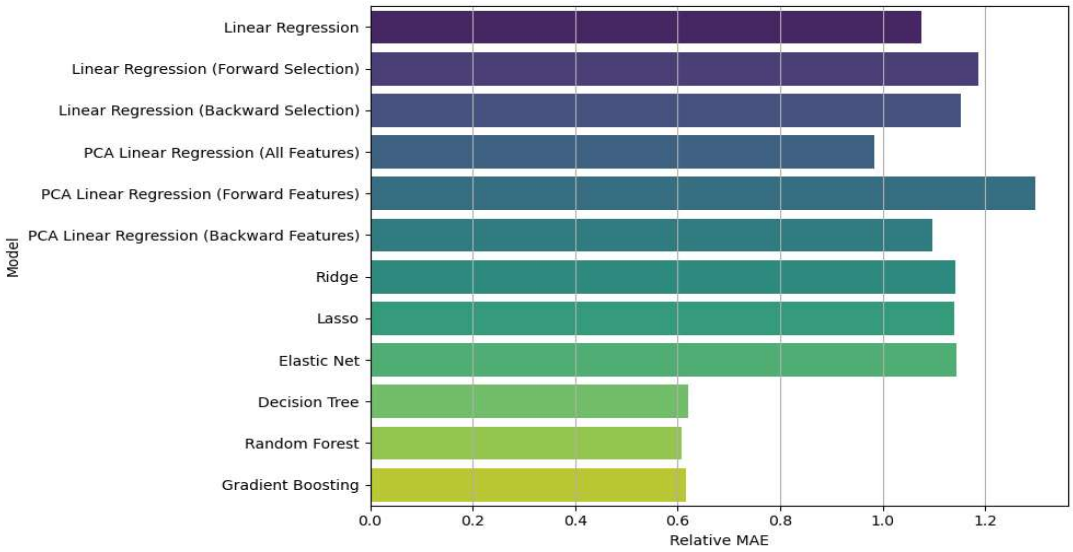

Figure 32: Relative MAE for each model

5.4.4. Robustness Tests

5.4.4.1. 10-Fold Cross Validation

The 10-fold cross-validation results (Figure 33) showed the higher performance of tree-based models. The Gradient Boosting model achieved the highest mean R^2 of 0.880000, outperforming the standard Linear Regression by 45.4%. The Random Forest model followed closely with a mean R^2 of 0.876710, just 0.4% lower than Gradient Boosting. The Random Forest model with Decision Tree Feature Selection slightly surpassed both, with a mean R^2 of 0.880683.

Linear models showed moderate performance, with Linear Regression (Backward Selection) achieving the best result among them (R^2 of 0.646116), still 26.6% lower than Gradient Boosting. PCA-based models underperformed, with the best (Forward Features) reaching an R^2 of 0.469760, 46.6% lower than Gradient Boosting.

Tree-based models also showed lower variance, with Gradient Boosting's R^2 standard deviation (0.033704) being 77.1% lower than Linear Regression's (0.147339), indicating more consistent performance across folds.

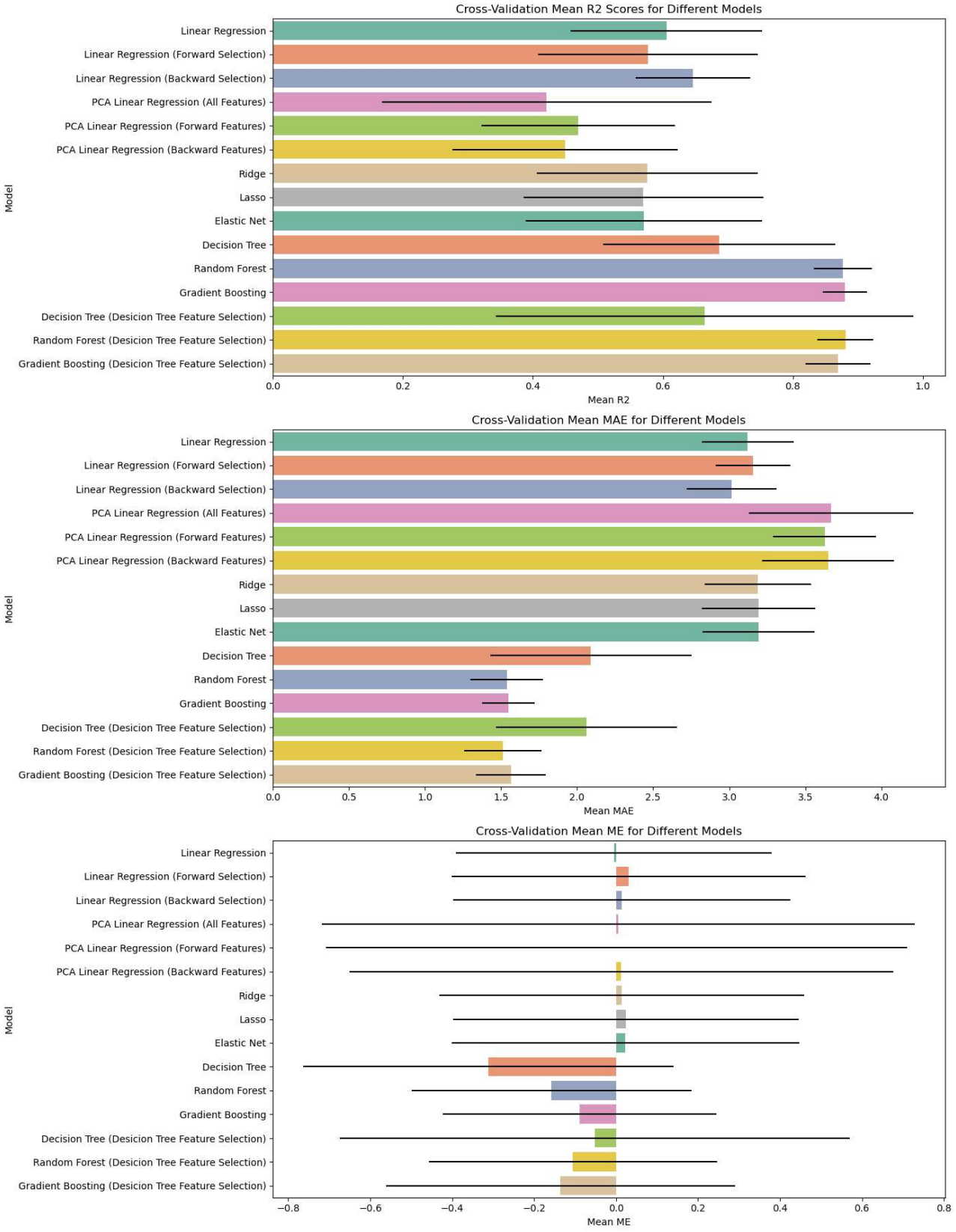

Figure 33: Meals weight models- 10-fold cross validation results

5.4.4.2. Test Set Size Analysis

The test-size analysis (Figure 34) revealed that for smaller test set sizes (0.1 to 0.3), both Gradient Boosting and Random Forest models showed high R^2 values above 0.82, with Random Forest slightly outperforming at 0.1. As test size increased to 0.5, both models improved, with Gradient Boosting reaching an R^2 of 0.866085.

At test sizes 0.6 and 0.7, both tree-based models maintained R^2 values around 0.82 to 0.85. At 0.8 test size, Gradient Boosting (R^2 = 0.730510) outperformed Random Forest (R^2 = 0.653105).

The Linear Regression model consistently underperformed, with R^2 values significantly lower across all test sizes. Its performance significantly deteriorated at larger test sizes, yielding negative R^2 values for 0.8 and 0.9 test sizes.

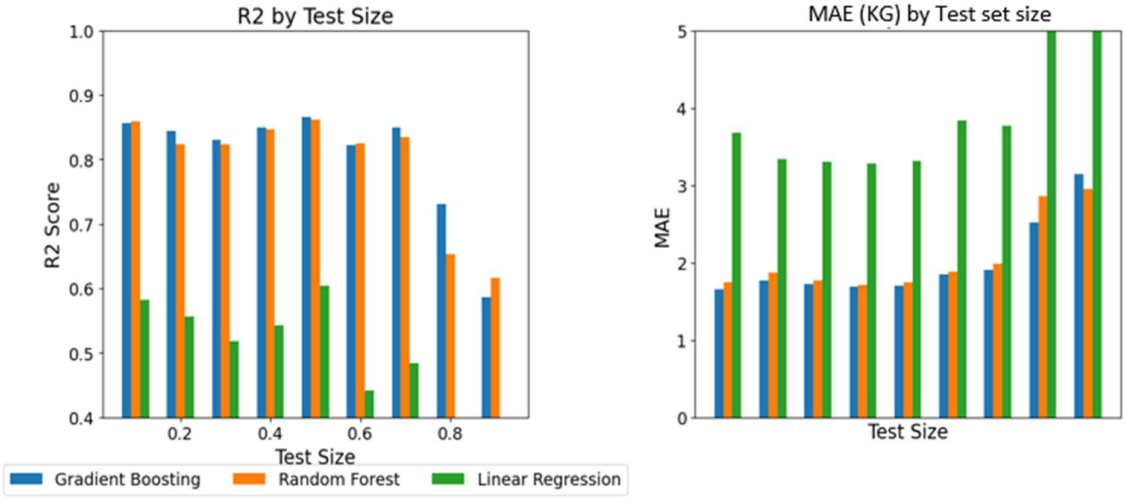

Figure 34: Meals weight - test-set size robustness test result

5.4.4.3. Random Noise Robustness

The Gradient-Boosting model "showed consistent performance in the random noise test (Table 10). Despite increasing noise levels from 1% to 10%, it maintained low Mean Absolute Error (MAE) values. The Random Forest model showed similar resilience, with slightly higher error rates. In contrast, the Linear Regression model consistently showed higher MAE values across all noise levels, indicating lower robustness to noise compared to the tree-based models.

Table 10: Meal weight random noise robustness test

Model	Noise (+/-)	MAE (KG)
Random Forest	1.0%	1.919919
	2.0%	1.895949
	5.0%	1.844348
	7.0%	1.842619
	10.0%	1.884450
Gradient Boosting	1.0%	1.850132
	2.0%	1.734328
	5.0%	1.822525
	7.0%	1.921764
	10.0%	1.820474
Linear Regression	1.0%	3.324279
	2.0%	3.365160
	5.0%	3.351906
	7.0%	3.262464
	10.0%	3.366937

5.4.5. Feature Importance

The feature importance analysis (Figure 35) revealed varying significant predictors across different models. Meal volume consistently was identified as the most important feature across all models, particularly in PCA-based approaches where it had importance values of 0.685599 to 0.839039.

For linear models, temperature-related features were particularly significant, with mean temperature showing the highest importance in standard linear regression (0.548141) and backward selection (0.494392). Barometric pressure sum was a key feature in regularized models like Ridge (0.013023), Lasso (0.024842), and Elastic Net (0.026750).

In tree-based models, while meal volume remained important, other features gained prominence. The Random Forest model highlighted temperature mean (0.154055) and pressure mean (0.147875) as important predictors. The Gradient Boosting model emphasized temperature mean (0.152033) and meal volume (0.389024).

This analysis indicates the importance of meal volume, temperature, and barometric pressure in predicting meal weight, while also highlighting the varied feature importance across different modelling approaches.

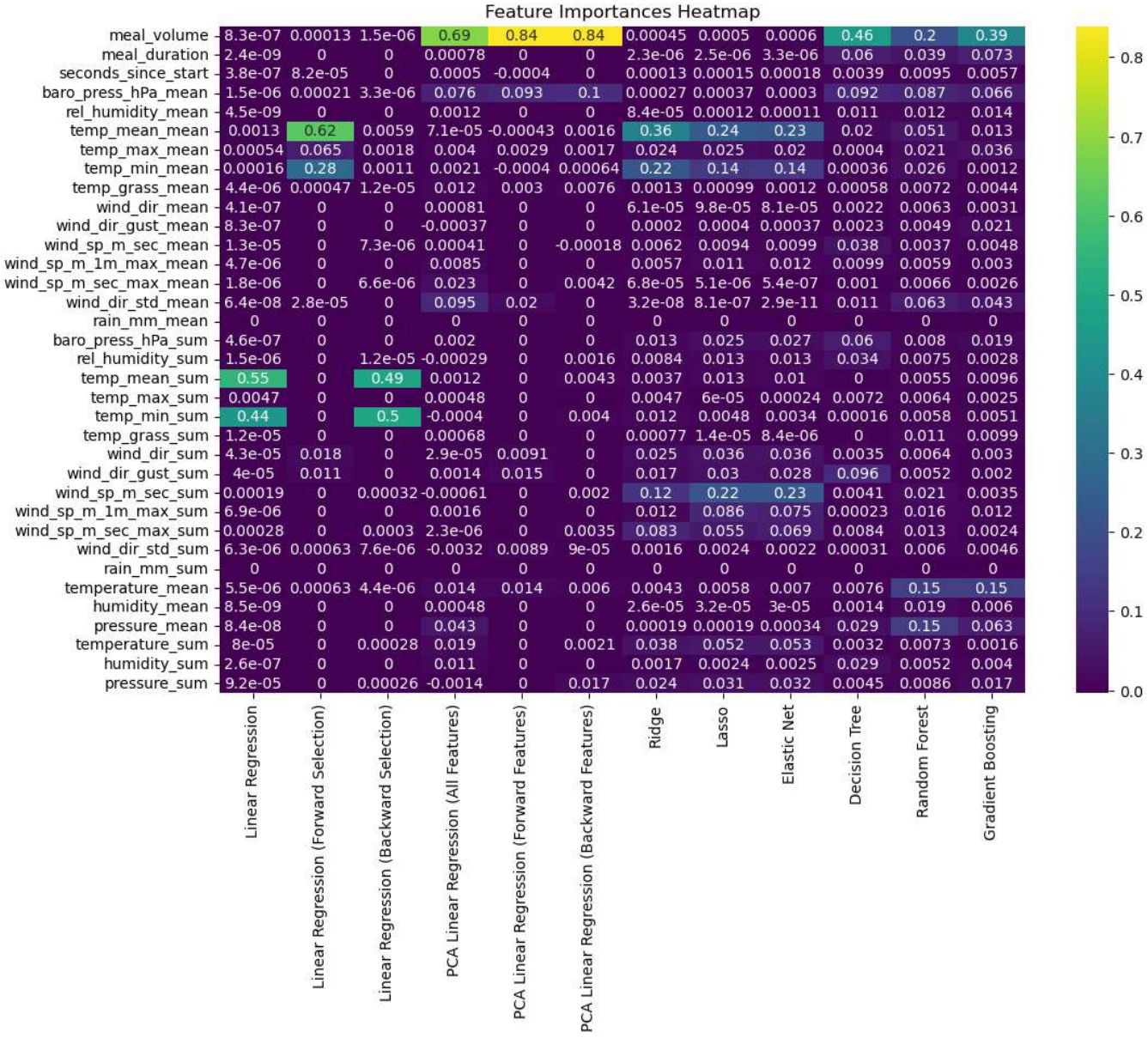

Figure 35: Meals weight models- feature importance

5.5. Cow Detection Model Performance

Although all three model sizes were trained with similar precision, the YOLOv8m model yielded lower recall. The YOLOv8s model outperformed the YOLOv8n and YOLOv8m models in terms of the mAP50 and mAP50-95. A common explanation is that YOLOv8n is too small and suffers from underfitting, whereas YOLOv8m is overfitted.

To improve the YOLOv8m model and attempt to train a larger model, it was necessary to extend the dataset. In Figure 36, Figure 37 and Figure 38 it can be seen that all models achieved convergence before epoch 200; therefore, further training was not required. In Table 10, the best results on the test set can be observed, highlighting the superior performance of the YOLOv8s model. In addition, because the aim was to implement the model on a real-time system with real-time detection, a YOLOv8s model was selected to ease running on an edge device.

Table 10: Best Test-set results for cow detection models

Model Size	Precision (B)	Recall (B)	mAP50 (B)	mAP50-95 (B)	Fitness	Preprocess Time (s)	Inference Time (ms)	Loss	Postprocess Time (s)
YOLOv8n	0.9654	0.9375	0.9321	0.6501	0.6783	2.1923	12.08	0.0	0.6923
YOLOv8s	0.9656	0.9375	0.9584	0.7340	0.7565	1.9996	16.85	0.0	0.6923
YOLOv8m	0.9666	0.9040	0.9338	0.6874	0.7120	2.0385	20.52	0.0	0.6538

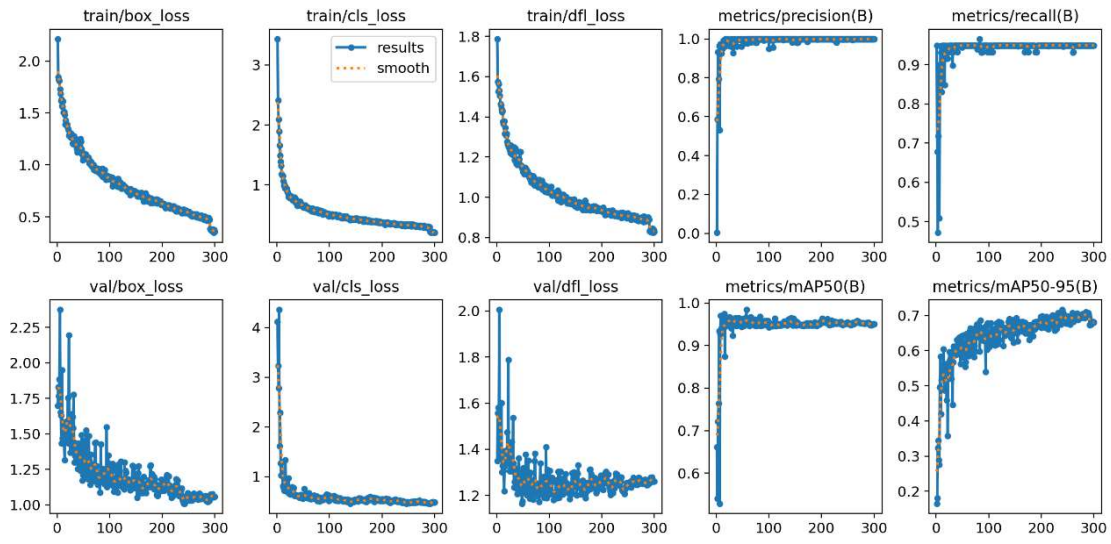

Figure 36: Training results of YOLOv8n detection model

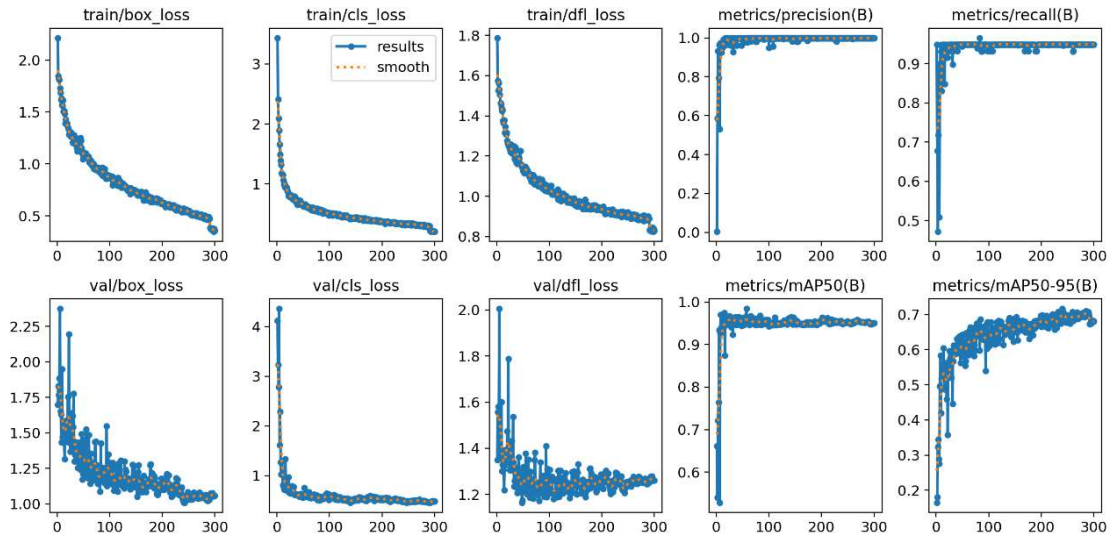

Figure 37：Training results of YOLOv8s detection model

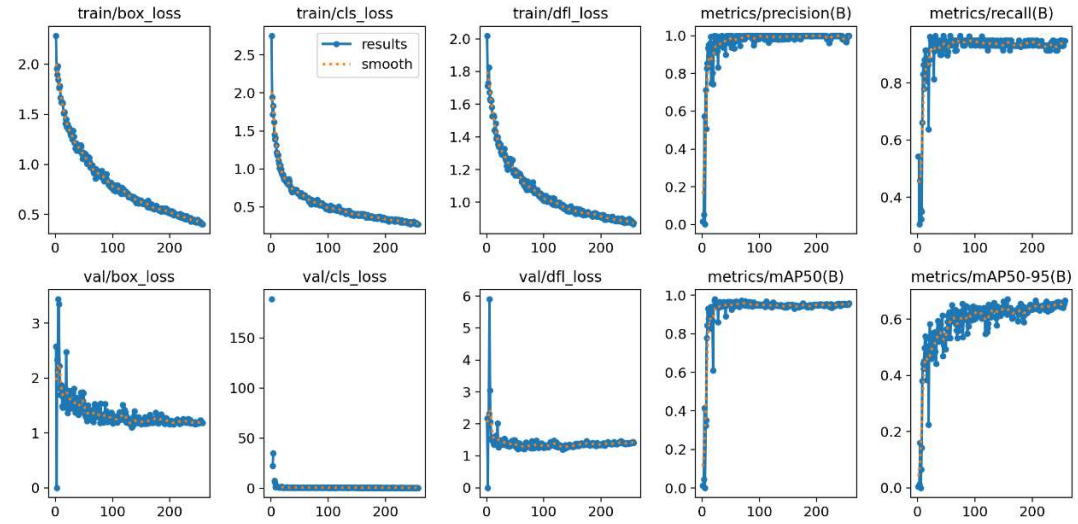

Figure 38：Training results of YOLOv8m detection model

6. Conclusions, Limitations and Future Recommendations

6.1. Conclusions

This study developed a computer vision to measure individual food intake and a model to predict meal weight with an R^2 of 0.880 and MAE of 1.547 KG, using a low-cost 3D camera from a height of 400 cm, while detecting the presence of a cow. The results suggest that measuring feed intake in a commercial cowshed, when a camera is positioned at a height that allows mechanical equipment to pass below it while viewing multiple feed positions, and when sunlight directly lights on the feed lane, is possible. To the best of our knowledge, this is the first study to identify the factors affecting the decrease in the volume and weight of unconsumed feed. This system allows for further development including integration of a cow identification system to connect feed consumption data to specific cows, or implementing the system in multiple stations, thereby enabling the monitoring of individual feed consumption for each cow in the cowshed. Future development should include also calibration of changes in feed density in different seasons.

The analysis of the density of unconsumed feed increased throughout the day, indicated that volume decreased faster than the feed's liquids evaporated, causing the feed to become more compact. This suggests that environmental heat accelerates the settling of feed volume more than it does the evaporation of moisture. In contrast, while the density of consumed feed initially grew, it eventually dropped significantly. This pattern highlights the cows' preference for denser, more nutrient-rich components, like grains, which they tend to consume earlier in the day.

The evaluation of various machine learning models for predicting volume decrease, weight decrease, and meal weight consistently showed that tree-based models outperformed linear models. Specifically, the Random Forest model showed higher performance for volume decrease and weight decrease, while the Gradient Boosting model showed higher accuracy for meal weight.

6.1.1. Unconsumed Feed - Volume Decrease Prediction

Tree-based models showed higher performance for predicting volume decrease. The Random Forest algorithm showed a higher R^2 of 0.972856, with minimal error rates (MAE: 0.004389). The Gradient Boosting model followed closely, with an R^2 of 0.970267 and comparable error metrics. The Random Forest's slight edge in performance led to its selection as the best suited model for this task in our tests.

Linear models, including various forms of linear regression, ridge, lasso, and elastic net, showed lower R^2 scores and higher error metrics. Feature selection techniques marginally improved linear regression performance, with backward selection yielding the results (R^2: 0.513533, MAE: 0.022961). However, these enhancements did not extend to PCA-based linear regression models.

6.1.2. Unconsumed Feed - Weight Decrease Prediction

The trend of weight loss prediction followed a similar pattern, with tree-based models consistently surpassing linear methods in performance. The Random Forest algorithm again showed better results with a R^2 of 0.973903 and low error rates (MAE: 0.002008). Gradient Boosting performed marginally better in terms of R^2 (0.974076) but was edged out by Random Forest when considering overall metrics.

The Random Forest model showed a 130.9% improvement over standard Linear Regression (R^2: 0.421820). The Ridge regularization technique slightly improved the basic linear models but still underperformed the tree-based models' performance.

6.1.3. Consumed Feed - Meal Weight Prediction

The Gradient Boosting model showed higher performance for meal weight prediction, achieving an R^2 of 0.880, the lowest MAE of 1.547 KG, and a relative MAE of 0.616. The Random Forest model also performed well, with an R^2 of 0.877, an MAE of 1.538 KG, and the lowest relative MAE of 0.607 among all models.

Linear models showed moderate performance, with Ridge Regression achieving an R^2 of 0.576 and a relative MAE of 1.142 KG. Feature selection techniques proved beneficial for linear regression, particularly backward selection, which improved the R^2 to 0.646 and resulted in a relative MAE of 1.152.

Feature selection using Decision Trees produced mixed results for tree-based models. The Random Forest model with selected features attained an R^2 of 0.881, slightly outperforming the standard Random Forest model, with a relative MAE of 0.607 KG. However, the Gradient Boosting model with feature selection, which had an R^2 of 0.870, performed slightly worse than the standard Gradient Boosting model and had a relative MAE of 0.616.

6.2. Limitations and Future Recommendations

The data used in this study were gathered during the summer months in Israel, specifically in June and July. This limited timeframe may have limited the diversity of the data collected. Additionally, noisy data from the sensors (weight scales and camera) further restricted the number of usable data collection days. To address this issue, data collection should be extended to additional months to capture the different light and weather conditions (specifically humidity). The limited timeframe also necessitated random train-test splits for model evaluation, rather than splits based on distinct days or time periods. This approach may not fully capture temporal variations in feeding behavior and environmental conditions.

The Intel RealSense cameras, D455 and D435, presented some difficulties when recording for extended periods of time, as they sometimes stopped responding after 5-6 hours of use. However, the Zed 2-i

camera was easier to communicate with and proved to be quite reliable. A future system should be developed and based on the new ZED-2i cameras and/or cameras fit for outdoor conditions.

Additionally, the feed placed on a metal plate made the recordings appear slightly shiny and distorted. Therefore, it is necessary to create a model for the empty feed position as a reference. If a solution can be found to address the issue of noisy 3-D recordings when the sun is on the feed, the accuracy of the results may be improved.
Note that future calibration must account for seasonal sun variations. While the current sunlight calibration was done to fit conditions close to the time of data collection, it is necessary to develop a year-round solution that addresses sunlight variations throughout different seasons.

This system should be tested using a larger number of feeding positions. Zed 2i's wide field of view at 120° enables it to capture a feeding area of 1385 cm when positioned at a height of 400 cm. This allowed six feeding positions. To minimize reflections from the metal, it is recommended to cover the weight scales. Although the camera is described as IP66 waterproof, it is still recommended to cover it as it suffers from moisture after a rainstorm, and finding a replacement is time-consuming.

7. References

André, G., Engel, B., Berentsen, P. B. M., Vellinga, T. V., & Oude Lansink, A. G. J. M. (2011). Quantifying the effect of heat stress on daily milk yield and monitoring dynamic changes using an adaptive dynamic model. *Journal of Dairy Science*, *94*(9), 4502–4513. https://doi.org/10.3168/jds.2010-4139

Arcidiacono, C., Porto, S. M. C., Mancino, M., & Cascone, G. (2017). Development of a threshold-based classifier for real-time recognition of cow feeding and standing behavioural activities from accelerometer data. *Computers and Electronics in Agriculture*, *134*, 124–134. https://doi.org/10.1016/j.compag.2017.01.021

Ayaz Mirani, A., Muhammad Suleman Memon, E., Qabulio, M., Suleman Memon, M., Chohan, R., & Ali Wagan, A. (2021). Machine Learning In Agriculture: A Review. *International Journal of Scientific & Technology Research*, *10*(05). Retrieved from https://www.ijstr.org/final-print/may2021/Machine-Learning-In-Agriculture-A-Review.pdf

Bell, M. J., Wall, E., Russell, G., Simm, G., & Stott, A. W. (2011). The effect of improving cow productivity, fertility, and longevity on the global warming potential of dairy systems. *Journal of Dairy Science*, *94*(7), 3662–3678. https://doi.org/10.3168/jds.2010-4023

Bergman, N., Yitzhaky, Y., & Halachmi, I. (2024). Biometric identification of dairy cows via real-time facial recognition. *Animal*, *18*(3). https://doi.org/10.1016/j.animal.2024.101079

Bezen, R., Edan, Y., & Halachmi, I. (2020). Computer vision system for measuring individual cow feed intake using RGB-D camera and deep learning algorithms. *Computers and Electronics in Agriculture*, *172*, 105345. https://doi.org/https://doi.org/10.1016/j.compag.2020.105345

Bloch, V., Levit, H., & Halachmi, I. (2019). Assessing the potential of photogrammetry to monitor feed intake of dairy cows. *Journal of Dairy Research*, *86*(1), 34–39. https://doi.org/10.1017/S0022029918000882

Brito, L. F., Bedere, N., Douhard, F., Oliveira, H. R., Arnal, M., Peñagaricano, F., … Miglior, F. (2021). Review: Genetic selection of high-yielding dairy cattle toward sustainable farming systems in a rapidly changing world. *Animal*, *15*. https://doi.org/10.1016/j.animal.2021.100292

Chizzotti, M. L., Machado, F. S., Valente, E. E. L., Pereira, L. G. R., Campos, M. M., Tomich, T. R., … Ribas, M. N. (2015). Technical note: Validation of a system for monitoring individual

feeding behavior and individual feed intake in dairy cattle. *Journal of Dairy Science*, *98*(5), 3438–3442. https://doi.org/10.3168/jds.2014-8925

Cleveland, W. S. (1979). Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, *74*(368), 829–836. https://doi.org/10.2307/2286407

Cole, J. B., & VanRaden, P. M. (2018). Symposium review: Possibilities in an age of genomics: The future of selection indices. *Journal of Dairy Science*, *101*(4), 3686–3701. https://doi.org/10.3168/jds.2017-13335

Davison, C., Bowen, J. M., Michie, C., Rooke, J. A., Jonsson, N., Andonovic, I., … Duthie, C.-A. (2021). Predicting feed intake using modelling based on feeding behaviour in finishing beef steers. *Animal: An International Journal of Animal Bioscience*, *10*(5), 100231. https://doi.org/10.1016/j.animal.2021.100231

De Haas, Y., Windig, J. J., Calus, M. P. L., Dijkstra, J., de Haan, M., Bannink, A., & Veerkamp, R. F. (2011). Genetic parameters for predicted methane production and potential for reducing enteric emissions through genomic selection. *Journal of Dairy Science*, *94*(12), 6122–6134. https://doi.org/10.3168/jds.2011-4439

Ding, L., Lv, Y., Jiang, R., Zhao, W., Li, Q., Yang, B., … Yu, Q. (2022). Predicting the Feed Intake of Cattle Based on Jaw Movement Using a Triaxial Accelerometer. *Agriculture (Switzerland)*, *12*(7). https://doi.org/10.3390/agriculture12070899

Dwyer, B., Nelson, J., & Solawetz, J. (2022). *Roboflow*. Retrieved from https://roboflow.com/

Gerber, PJ, Steinfeld, Henderson, Mottet, Opio, … Tempio. (2013). *Tackling climate change through livestock – A global assessment of emissions and mitigation opportunities*. Retrieved from https://www.fao.org/4/i3437e/i3437e.pdf

Giagnoni, G., Lassen, J., Lund, P., Foldager, L., Johansen, M., & Weisbjerg, M. R. (2024). Feed intake in housed dairy cows: validation of a three-dimensional camera−based feed intake measurement system. *Animal*, *18*(6). https://doi.org/10.1016/j.animal.2024.101178

Halachmi, I., Edan, Y., Maltz, E., Peiper, U. M., Moallem, U., & Brukental, I. (1998). A real-time control system for individual dairy cow food intake. *Computers and Electronics in Agriculture*, *20*(2), 131–144. https://doi.org/10.1016/S0168-1699(98)00013-1

Hemme, T., Uddin, M. M., & Ndambi, O. A. (2014). Benchmarking Cost of Milk Production in 46 Countries. *Journal of Reviews on Global Economics*, *3*, 254–270. https://doi.org/10.6000/1929-7092.2014.03.20

Hussain, M. (2023). YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. *Machines*, *11*(7), 677. https://doi.org/10.3390/machines11070677

Laberge, B., & Rousseau, A. N. (2017). Rethinking environment control strategy of confined animal housing systems through precision livestock farming. *Biosystems Engineering*, *155*, 96–123. https://doi.org/https://doi.org/10.1016/j.biosystemseng.2016.12.005

Lassen, J., Rind Thomasen, J., Hansen, R. H., Gunnar Brink Nielsen, G., Olsen, E., René, P., … Borchersen, S. (2018). Individual measure of feed intake on in-house commercial dairy cattle using 3D camera system. *WCGALP 2018 Proceedings*.

Pickering, N. K., Chagunda, M. G. G., Banos, G., Mrode, R., McEwan, J. C., & Wall, E. (2015). Genetic parameters for predicted methane production and laser methane detector measurements1. *Journal of Animal Science*, *93*(1), 11–20. https://doi.org/10.2527/jas.2014-8302

Rojo-Gimeno, C., van der Voort, M., Niemi, J. K., Lauwers, L., Kristensen, A. R., & Wauters, E. (2019). Assessment of the value of information of precision livestock farming: A conceptual framework. *NJAS: Wageningen Journal of Life Sciences*, *90–91*(1), 1–9. https://doi.org/10.1016/j.njas.2019.100311

Saar, M., Edan, Y., Godo, A., Lepar, J., Parmet, Y., & Halachmi, I. (2022). A machine vision system to predict individual cow feed intake of different feeds in a cowshed. *Animal*, *16*(1). https://doi.org/10.1016/J.ANIMAL.2021.100432

Stear, M. J., Nikbakht, G., Matthews, L., & Jonsson, N. N. (2012). Breeding for disease resistance in livestock and fish. *CABI Reviews*, 1–10. https://doi.org/10.1079/PAVSNNR20127007

Steeneveld, W., & Hogeveen, H. (2015). Characterization of Dutch dairy farms using sensor systems for cow management. *Journal of Dairy Science*, *98*(1), 709–717. https://doi.org/10.3168/jds.2014-8595

Van De Gucht, T., Saeys, W., Van Weyenberg, S., Lauwers, L., Mertens, K., Vandaele, L., … Van Nuffel, A. (2017). Automatic cow lameness detection with a pressure mat: Effects of mat length and sensor resolution. *Computers and Electronics in Agriculture*, *134*, 172–180. https://doi.org/10.1016/j.compag.2017.01.011

Wang, X., Dai, B., Wei, X., Shen, W., Zhang, Y., & Xiong, B. (2023). Vision-based measuring method for individual cow feed intake using depth images and a Siamese network. *International Journal of Agricultural and Biological Engineering*, *16*(3), 233–239. https://doi.org/10.25165/j.ijabe.20231603.7985

Statement on the use of AI:

During the preparation of this work the author used ChatGPT in order to improve language. (Wrote by myself in his language and then iterated with AI to improve wording). After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the content of the work.

8. Appendices

Appendix A: Raw data and code

GitHub: https://github.com/benjika/research_code

Appendix B: Ingredients, chemical and structural composition, and mineral concentrations of TMR

Ingredients (% of TMR DM)	TMR
Wheat silage	20.0
Corn grain, ground	18.5
Wheat hay	15.8
Corn gluten feed	12.4
Dried distillers grains, golden	6.6
Soy meal	5.0
Rapeseed meal	4.5
Wheat bran	3.3
Barley grain, ground	3.1
Whey concentrate	2.3
Wheat hay	2.2
Cotton seeds	1.9
Calcium and sodium chloride	1.4
Palmitic acid	1.0
Ca-LCFA[1]	1.0
Sodium bicarbonate	0.7
Limestone	0.2
Urea	0.1
Vitamins and trace minerals[2]	0.045
Composition (% of TMR DM)	
DM (% of wet TMR)	70.9
OM	91.6
CP	18.1
Ether extract	6.24
αNDFom	34.0
ADF	18.2
Lignin	2.4
Starch	18.6
Ca	0.97
P	0.72

[1] Calcium salts of long-chain fatty acids

[2] Mix containing (g/kg of mix DM): Zn, 24; Fe, 24; Cu, 12.8; Mn, 24; I, 1.44; Co, 0.32; Se, 0.32; 16 MIU of vitamin A; 3.2 MIU of vitamin D3; and 48 KIU of vitamin E.SS

Appendix C: Data analysis libraries

platform : win-64	aom=3.8.2=h63175ca_0	asttokens=2.0.5=pyhd3eb1b0_0
blas=2.121=mkl	blas-devel=3.9.0=21_win64_mkl	brotli=1.1.0=hcfcfb64_1
brotli-bin=1.1.0=hcfcfb64_1	brotli-python=1.1.0=py312h53d5487_1	bzip2=1.0.8=hcfcfb64_5
ca-certificates=2024.3.11=haa95532_0	cairo=1.18.0=h1fef639_0	certifi=2024.2.2=py312haa95532_0
charset-normalizer=3.3.2=pyhd8ed1ab_0	colorama=0.4.6=pyhd8ed1ab_0	contourpy=1.2.0=py312h0d7def4_0
cuda-cccl=12.4.99=0	cuda-cudart=11.8.89=0	cuda-cudart-dev=11.8.89=0

cuda-cupti=11.8.87=0	cuda-libraries=11.8.0=0	cuda-libraries-dev=11.8.0=0
cuda-nvrtc=11.8.89=0	cuda-nvrtc-dev=11.8.89=0	cuda-nvtx=11.8.86=0
cuda-profiler-api=12.4.99=0	cuda-runtime=11.8.0=0	cycler=0.12.1=pyhd8ed1ab_0
dav1d=1.2.1=hcfcfb64_0	decorator=5.1.1=pyhd3eb1b0_0	double-conversion=3.3.0=h63175ca_0
executing=0.8.3=pyhd3eb1b0_0	expat=2.6.2=h63175ca_0	ffmpeg=6.1.1=gpl_hb766fab_106
filelock=3.13.1=pyhd8ed1ab_0	font-ttf-dejavu-sans-mono=2.37=hab24e00_0	font-ttf-inconsolata=3.000=h77eed37_0
font-ttf-source-code-pro=2.038=h77eed37_0	font-ttf-ubuntu=0.83=h77eed37_1	fontconfig=2.14.2=hbde0cde_0
fonts-conda-ecosystem=1=0	fonts-conda-forge=1=0	fonttools=4.49.0=py312he70551f_0
freeglut=3.2.2=h63175ca_2	freetype=2.12.1=hdaf720e_2	gettext=0.21.1=h5728263_0
graphite2=1.3.13=1000	harfbuzz=8.3.0=h7ab893a_0	hdf5=1.14.3=nompi_h73e8ff5_100
icu=73.2=h63175ca_0	idna=3.6=pyhd8ed1ab_0	imath=3.1.11=h12be248_0
intel-openmp=2024.0.0=h57928b3_49841	ipython=8.20.0=py312haa95532_0	jasper=4.2.2=h28f2b1a_0
jedi=0.18.1=py312haa95532_1	jinja2=3.1.3=pyhd8ed1ab_0	khronos-opencl-icd-loader=2023.04.17=h64bf75a_0
kiwisolver=1.4.5=py312h0d7def4_1	krb5=1.21.2=heb0366b_0	lcms2=2.16=h67d730c_0
lerc=4.0.0=h63175ca_0	libabseil=20240116.1=cxx17_h63175ca_2	libaec=1.1.2=h63175ca_1
libblas=3.9.0=21_win64_mkl	libbrotlicommon=1.1.0=hcfcfb64_1	libbrotlidec=1.1.0=hcfcfb64_1
libbrotlienc=1.1.0=hcfcfb64_1	libcblas=3.9.0=21_win64_mkl	libclang=17.0.6=default_hde6756a_3
libclang13=17.0.6=default_h85b4d89_3	libcublas=11.11.3.6=0	libcublas-dev=11.11.3.6=0
libcufft=10.9.0.58=0	libcufft-dev=10.9.0.58=0	libcurand=10.3.5.119=0
libcurand-dev=10.3.5.119=0	libcurl=8.6.0=hd5e4a3a_0	libcusolver=11.4.1.48=0
libcusolver-dev=11.4.1.48=0	libcusparse=11.7.5.86=0	libcusparse-dev=11.7.5.86=0
libdeflate=1.19=hcfcfb64_0	libexpat=2.6.2=h63175ca_0	libffi=3.4.2=h8ffe710_5
libglib=2.80.0=h39d0aa6_0	libhwloc=2.9.3=default_haede6df_1009	libiconv=1.17=hcfcfb64_2
libjpeg-turbo=3.0.0=hcfcfb64_1	liblapack=3.9.0=21_win64_mkl	liblapacke=3.9.0=21_win64_mkl
libnpp=11.8.0.86=0	libnpp-dev=11.8.0.86=0	libnvjpeg=11.9.0.86=0
libnvjpeg-dev=11.9.0.86=0	libopencv=4.9.0=qt6_py312hfa05d50_610	libopenvino=2023.3.0=hc2557fa_3
libopenvino-auto-batch-plugin=2023.3.0=h002f227_3	libopenvino-auto-plugin=2023.3.0=h002f227_3	libopenvino-hetero-plugin=2023.3.0=h7e3b17c_3
libopenvino-intel-cpu-plugin=2023.3.0=hc2557fa_3	libopenvino-intel-gpu-plugin=2023.3.0=hc2557fa_3	libopenvino-ir-frontend=2023.3.0=h7e3b17c_3
libopenvino-onnx-frontend=2023.3.0=h1ef3bed_3	libopenvino-paddle-frontend=2023.3.0=h1ef3bed_3	libopenvino-pytorch-frontend=2023.3.0=h63175ca_3
libopenvino-tensorflow-frontend=2023.3.0=he87eab5_3	libopenvino-tensorflow-lite-frontend=2023.3.0=h63175ca_3	libopus=1.3.1=h8ffe710_1
libpng=1.6.43=h19919ed_0	libprotobuf=4.25.2=h503648d_1	libsqlite=3.45.2=hcfcfb64_0
libssh2=1.11.0=h7dfc565_0	libtiff=4.6.0=h6e2ebb7_2	libuv=1.48.0=hcfcfb64_0
libwebp-base=1.3.2=hcfcfb64_0	libxcb=1.15=hcd874cb_0	libxml2=2.12.6=hc3477c8_0
libzlib=1.2.13=hcfcfb64_5	m2w64-gcc-libgfortran=5.3.0=6	m2w64-gcc-libs=5.3.0=7
m2w64-gcc-libs-core=5.3.0=7	m2w64-gmp=6.1.0=2	m2w64-libwinpthread-git=5.0.0.4634.697f757=2
markupsafe=2.1.5=py312he70551f_0	matplotlib-base=3.8.3=py312h26ecaf7_0	matplotlib-inline=0.1.6=py312haa95532_0
mkl=2024.0.0=h66d3029_49657	mkl-devel=2024.0.0=h57928b3_49657	mkl-include=2024.0.0=h66d3029_49657
mpmath=1.3.0=pyhd8ed1ab_0	msys2-conda-epoch=20160418=1	munkres=1.1.4=pyh9f0ad1d_0
networkx=3.2.1=pyhd8ed1ab_0	numpy=1.26.4=py312h8753938_0	opencv=4.9.0=qt6_py312ha9d5604_610
openexr=3.2.2=h72640d8_1	openh264=2.4.1=h63175ca_0	openjpeg=2.5.2=h3d672ee_0
openssl=3.2.1=hcfcfb64_1	packaging=24.0=pyhd8ed1ab_0	pandas=2.2.1=py312h2ab9e98_0
parso=0.8.3=pyhd3eb1b0_0	patsy=0.5.6=pyhd8ed1ab_0	pcre2=10.43=h17e33f8_0
pillow=10.2.0=py312he768995_0	pip=24.0=pyhd8ed1ab_0	pixman=0.43.4=h63175ca_0
prompt-toolkit=3.0.43=py312haa95532_0	prompt_toolkit=3.0.43=hd3eb1b0_0	psutil=5.9.8=py312he70551f_0
pthread-stubs=0.4=hcd874cb_1001	pthreads-win32=2.9.1=hfa6e2cd_3	pugixml=1.14=h63175ca_0
pure_eval=0.2.2=pyhd3eb1b0_0	py-cpuinfo=9.0.0=pyhd8ed1ab_0	py-opencv=4.9.0=qt6_py312h34b1e23_610
pygments=2.15.1=py312haa95532_1	pyparsing=3.1.2=pyhd8ed1ab_0	pysocks=1.7.1=pyh0701188_6
python=3.12.2=h2628c8c_0_cpython	python-dateutil=2.9.0=pyhd8ed1ab_0	python-tzdata=2024.1=pyhd8ed1ab_0
python_abi=3.12=4_cp312	pytorch=2.2.1=py3.12_cuda11.8_cudnn8_0	pytorch-cuda=11.8=h24eeafa_5
pytorch-mutex=1.0=cuda	pytz=2024.1=pyhd8ed1ab_0	pyyaml=6.0.1=py312he70551f_1
qt6-main=6.6.2=h5dee92f_4	requests=2.31.0=pyhd8ed1ab_0	scipy=1.12.0=py312h8753938_2
seaborn=0.13.2=hd8ed1ab_0	seaborn-base=0.13.2=pyhd8ed1ab_0	setuptools=69.2.0=pyhd8ed1ab_0
six=1.16.0=pyh6c4a22f_0	snappy=1.1.10=hfb803bf_0	stack_data=0.2.0=pyhd3eb1b0_0
statsmodels=0.14.1=py312ha90f08f_0	svt-av1=1.8.0=h63175ca_0	sympy=1.12=pyh04b8f61_3
tbb=2021.11.0=h91493d7_1	tk=8.6.13=h5226925_1	torchvision=0.17.1=pypi_0
tqdm=4.66.2=pyhd8ed1ab_0	traitlets=5.7.1=py312haa95532_0	typing_extensions=4.10.0=pyha770c72_0
tzdata=2024a=h0c530f3_0	ucrt=10.0.22621.0=h57928b3_0	ultralytics=8.1.16=pyh2965483_0
urllib3=2.2.1=pyhd8ed1ab_0	vc=14.3=hcf57466_18	vc14_runtime=14.38.33130=h82b7239_18
vs2015_runtime=14.38.33130=hcb4865c_18	wcwidth=0.2.5=pyhd3eb1b0_0	wheel=0.42.0=pyhd8ed1ab_0
win_inet_pton=1.1.0=pyhd8ed1ab_6	x264=1!164.3095=h8ffe710_2	x265=3.5=h2d74725_3

xorg-libxau=1.0.11=hcd874cb_0	xorg-libxdmcp=1.1.3=hcd874cb_0	xz=5.2.6=h8d14728_0
yaml=0.2.5=h8ffe710_2	zlib=1.2.13=hcfcfb64_5	zstd=1.5.5=h12be248_0

Appendix D: Yolo Development libraries

platform : win-64	py-xgboost-mutex 2.0 gpu_0 conda-forge	asttokens 2.2.1 pypi_0 pypi
attrs 23.1.0 pypi_0 pypi	backcall 0.2.0 pypi_0 pypi	blas 1.0 mkl
bottleneck 1.3.5 py310h9128911_0	brotli 1.0.9 h2bbff1b_7	brotli-bin 1.0.9 h2bbff1b_7
bzip2 1.0.8 he774522_0	ca-certificates 2024.3.11 haa95532_0	cairo 1.16.0 he04af86_2 anaconda
certifi 2023.5.7 pypi_0 pypi	charset-normalizer 3.1.0 pypi_0 pypi	click 8.1.3 pypi_0 pypi
colorama 0.4.6 pypi_0 pypi	comm 0.1.3 pypi_0 pypi	configargparse 1.5.3 pypi_0 pypi
contourpy 1.0.7 pypi_0 pypi	cuda-version 11.8 h70ddcb2_2 conda-forge	cycler 0.11.0 pyhd3eb1b0_0
dash 2.10.0 pypi_0 pypi	dash-core-components 2.0.0 pypi_0 pypi	dash-html-components 2.0.0 pypi_0 pypi
dash-table 5.0.0 pypi_0 pypi	debugpy 1.6.7 pypi_0 pypi	decorator 5.1.1 pypi_0 pypi
executing 1.2.0 pypi_0 pypi	expat 2.4.9 h6c2663c_0 anaconda	fastjsonschema 2.17.1 pypi_0 pypi
filelock 3.12.0 pypi_0 pypi	flask 2.2.3 pypi_0 pypi	fonttools 4.39.4 pypi_0 pypi
freetype 2.12.1 ha860e81_0	fribidi 1.0.10 h62dcd97_0 anaconda	fsspec 2023.4.0 pypi_0 pypi
getopt-win32 0.1 h2bbff1b_0 anaconda	giflib 5.2.1 h8cc25b3_3	glib 2.69.1 h5dc1a3c_2
graphite2 1.3.14 hd77b12b_1 anaconda	graphviz 2.50.0 hdb8b0d4_0 anaconda	gst-plugins-base 1.18.5 h9e645db_0
gstreamer 1.18.5 hd78058f_0	gts 0.7.6 h63ab5a1_3 anaconda	harfbuzz 4.3.0 hda2c7e1_0 anaconda
icc_rt 2022.1.0 h6049295_2 anaconda	icu 58.2 ha925a31_3	idna 3.4 pypi_0 pypi
intel-openmp 2023.1.0 h59b6b97_46319	ipykernel 6.23.1 pypi_0 pypi	ipython 8.13.2 pypi_0 pypi
ipywidgets 8.0.6 pypi_0 pypi	itsdangerous 2.1.2 pypi_0 pypi	jedi 0.18.2 pypi_0 pypi
jinja2 3.1.2 pypi_0 pypi	joblib 1.1.1 py310haa95532_0 anaconda	jpeg 9e h2bbff1b_1
jsonschema 4.17.3 pypi_0 pypi	jupyter-client 8.2.0 pypi_0 pypi	jupyter-core 5.3.0 pypi_0 pypi
jupyterlab-widgets 3.0.7 pypi_0 pypi	kiwisolver 1.4.4 py310hd77b12b_0	krb5 1.19.4 h5b6d351_0
lerc 3.0 hd77b12b_0	libbrotlicommon 1.0.9 h2bbff1b_7	libbrotlidec 1.0.9 h2bbff1b_7
libbrotlienc 1.0.9 h2bbff1b_7	libclang 12.0.0 default_h627e005_2	libclang13 14.0.6 default_h8e68704_1
libdeflate 1.17 h2bbff1b_0	libffi 3.4.4 hd77b12b_0	libgd 2.3.3 ha43c60c_1 anaconda
libiconv 1.16 h2bbff1b_2	libogg 1.3.5 h2bbff1b_1	libpng 1.6.39 h8cc25b3_0
libtiff 4.5.0 h6c2663c_2	libvorbis 1.3.7 he774522_0	libwebp 1.2.4 hbc33d0d_1
libwebp-base 1.2.4 h2bbff1b_1	libxgboost 1.7.6 cuda112hab6cd6e_2 conda-forge	libxml2 2.10.3 h0ad7f3c_0
libxslt 1.1.37 h2bbff1b_0	lz4-c 1.9.4 h2bbff1b_0	markupsafe 2.1.2 pypi_0 pypi
matplotlib 3.7.1 py310haa95532_1	matplotlib-base 3.7.1 py310h4ed8f06_1	matplotlib-inline 0.1.6 pypi_0 pypi
mkl 2023.1.0 h8bd8f75_46356	mkl-service 2.4.0 py310h2bbff1b_1	mkl_fft 1.3.6 py310h4ed8f06_1
mkl_random 1.2.2 py310h4ed8f06_1	mlxtend 0.23.1 pyhd8ed1ab_0 conda-forge	mpmath 1.3.0 pypi_0 pypi
munkres 1.1.4 py_0	nbformat 5.7.0 pypi_0 pypi	nest-asyncio 1.5.6 pypi_0 pypi
networkx 3.1 pypi_0 pypi	numexpr 2.8.4 py310h2cd9be0_1	numpy 1.24.3 py310h055cbcc_1
numpy-base 1.24.3 py310h65a83cf_1	open3d 0.17.0 pypi_0 pypi	opencv-python 4.7.0.72 pypi_0 pypi
openssl 3.0.13 h2bbff1b_1	packaging 23.1 pypi_0 pypi	pandas 2.0.2 pypi_0 pypi

pango 1.50.7 h78c2152_0 anaconda	parso 0.8.3 pypi_0 pypi	patsy 0.5.3 pyhd8ed1ab_0 conda-forge
pcre 8.45 hd77b12b_0	pickleshare 0.7.5 pypi_0 pypi	pillow 9.5.0 pypi_0 pypi
pip 23.0.1 py310haa95532_0	pixman 0.40.0 h2bbff1b_1 anaconda	platformdirs 3.5.1 pypi_0 pypi
plotly 5.14.1 pypi_0 pypi	ply 3.11 py310haa95532_0	prompt-toolkit 3.0.38 pypi_0 pypi
psutil 5.9.5 pypi_0 pypi	pure-eval 0.2.2 pypi_0 pypi	py-xgboost 1.7.6 cuda112py310h9c3f586_2 conda-forge
py-xgboost-gpu 1.7.6 py310h95c4cff_2 conda-forge	pygments 2.15.1 pypi_0 pypi	pyparsing 3.0.9 py310haa95532_0
pyqt 5.15.7 py310hd77b12b_0	pyqt5-sip 12.11.0 py310hd77b12b_0	pyrsistent 0.19.3 pypi_0 pypi
python 3.10.14 he1021f5_1	python-dateutil 2.8.2 pyhd3eb1b0_0	python_abi 3.10 2_cp310 conda-forge
pytz 2023.3 pypi_0 pypi	pywin32 306 pypi_0 pypi	pyyaml 6.0 pypi_0 pypi
pyzmq 25.1.0 pypi_0 pypi	qt-main 5.15.2 he8e5bd7_7	qt-webengine 5.15.9 h5bd16bc_7
qtwebkit 5.212 h2bbfb41_5	requests 2.31.0 pypi_0 pypi	scikit-learn 1.2.0 py310hd77b12b_1 anaconda
scipy 1.10.1 pypi_0 pypi	seaborn 0.12.2 pypi_0 pypi	setuptools 67.8.0 py310haa95532_0
sip 6.6.2 py310hd77b12b_0	six 1.16.0 pyhd3eb1b0_1	sqlite 3.45.3 h2bbff1b_0
stack-data 0.6.2 pypi_0 pypi	statsmodels 0.14.0 py310h9128911_0	sympy 1.12 pypi_0 pypi
tbb 2021.8.0 h59b6b97_0	tenacity 8.2.2 pypi_0 pypi	threadpoolctl 2.2.0 pyh0d69192_0 anaconda
tk 8.6.14 h0416ee5_0	toml 0.10.2 pyhd3eb1b0_0	torch 2.1.0.dev20230530+cu121 pypi_0 pypi
Torchvision 0.15.2 pypi_0 pypi	Tornado 6.3.2 pypi_0 pypi	tqdm 4.65.0 pypi_0 pypi
traitlets 5.9.0 pypi_0 pypi	typing-extensions 4.6.2 pypi_0 pypi	tzdata 2023.3 pypi_0 pypi
ucrt 10.0.22621.0 h57928b3_0 conda-forge	ultralytics 8.0.111 pypi_0 pypi	urllib3 2.0.2 pypi_0 pypi
vc 14.2 h21ff451_1	vc14_runtime 14.36.32532 hdcecf7f_17 conda-forge	vs2015_runtime 14.36.32532 h05e6639_17 conda-forge
wcwidth 0.2.6 pypi_0 pypi	werkzeug 2.2.3 pypi_0 pypi	wheel 0.38.4 py310haa95532_0
widgetsnbextension 4.0.7 pypi_0 pypi	xz 5.4.6 h8cc25b3_1	zlib 1.2.13 h8cc25b3_0
zstd 1.5.5 hd43e919_0		

Appendix E: Volume Decrease model selection results

Model	R^2	Adj R^2	MAE	MSE	RMSE
Linear Regression	0.509653	0.522066	0.023182	0.000926	0.030437
Linear Regression (Forward Selection)	0.502656	0.512201	0.023267	0.000939	0.030639
Linear Regression (Backward Selection)	0.513533	0.521389	0.022961	0.000921	0.030350
PCA Linear Regression (All Features)	0.454244	0.452097	0.024308	0.001036	0.032186
PCA Linear Regression (Forward Features)	0.442648	0.439932	0.024488	0.001057	0.032519
PCA Linear Regression (Backward Features)	0.454244	0.452097	0.024308	0.001036	0.032186
Ridge	0.509354	0.519025	0.023109	0.000927	0.030448
Lasso	0.510784	0.516457	0.023054	0.000925	0.030410
Elastic Net	0.510900	0.517342	0.023047	0.000924	0.030404
Decision Tree	0.910367	0.984115	0.008021	0.000171	0.013077
Random Forest	0.972856	1.000000	0.004389	0.000052	0.007221
Gradient Boosting	0.970267	0.999921	0.004945	0.000056	0.007514

Appendix F: Volume Decrease models - Selected Parameters

Model	Feature	Selected Value
PCA Linear Regression (All Features)	n_components	5
PCA Linear Regression (Forward Features)	n_components	4
PCA Linear Regression (Backward Features)	n_components	5
Ridge	alpha	1
Lasso	alpha	0.0001
	max_iter	10000
Elastic Net	alpha	0.0001
	l1_ratio	0.7
	max_iter	10000
Decision Tree	criterion	squared_error
	max_depth	None
	max_features	None
	min_samples_leaf	2
	min_samples_split	5
	Splitter	random
Random Forest	bootstrap	False
	max_depth	20
	max_features	sqrt
	max_leaf_nodes	None
	min_impurity_decrease	0.0
	min_samples_leaf	1
	min_samples_split	2
	n_estimators	300
Gradient Boosting	learning_rate	0.1
	max_depth	9
	max_features	sqrt
	min_samples_leaf	9
	min_samples_split	2
	n_estimators	400

Appendix G: Volume Decrease test-set size robustness results

test_size	model	r2	mae
0.1	Gradient Boosting	0.957680	0.005390
	Random Forest	0.963613	0.004620
	Linear Regression	0.397386	0.024572
0.2	Gradient Boosting	0.970699	0.004949
	Random Forest	0.962212	0.004947
	Linear Regression	0.458705	0.024501
0.3	Gradient Boosting	0.957854	0.005810
	Random Forest	0.949319	0.005642
	Linear Regression	0.477823	0.024199
0.4	Gradient Boosting	0.953115	0.006131
	Random Forest	0.939026	0.006272
	Linear Regression	0.483460	0.023220
0.5	Gradient Boosting	0.943713	0.006926
	Random Forest	0.936204	0.006879
	Linear Regression	0.490038	0.023377
0.6	Gradient Boosting	0.922225	0.008286
	Random Forest	0.916203	0.008082
	Linear Regression	0.492564	0.023226
0.7	Gradient Boosting	0.838087	0.011438
	Random Forest	0.846065	0.010923
	Linear Regression	0.472885	0.023871
0.8	Gradient Boosting	0.761978	0.014060
	Random Forest	0.778390	0.013354
	Linear Regression	0.475688	0.024373
0.9	Gradient Boosting	0.631931	0.019210
	Random Forest	0.635392	0.018392
	Linear Regression	0.389109	0.026942

Appendix H: Volume Decrease 10-fold cross validation results

Model	Mean R2	Std R2	Mean MSE	Std MSE	Mean MAE	Std MAE	Mean ME	Std ME
Linear Regression	0.509653	0.051394	0.000926	0.000165	0.023182	0.002158	-0.000015	0.003679
Linear Regression (Forward Selection)	0.502656	0.051719	0.000939	0.000163	0.023267	0.002119	-0.000014	0.003610
Linear Regression (Backward Selection)	0.513533	0.053764	0.000921	0.000178	0.022961	0.002224	-0.000045	0.003574
PCA Linear Regression (All Features)	0.454244	0.070191	0.001036	0.000213	0.024308	0.002789	-0.000033	0.004069
PCA Linear Regression (Forward Features)	0.449373	0.069442	0.001046	0.000221	0.024856	0.002891	-0.000017	0.003789
PCA Linear Regression (Backward Features)	0.452820	0.067883	0.001038	0.000211	0.024499	0.002747	-0.000010	0.004082
Ridge	0.509359	0.051879	0.000927	0.000167	0.023109	0.002129	-0.000006	0.003707
Lasso	0.510789	0.049863	0.000925	0.000166	0.023054	0.002172	-0.000023	0.003660
Elastic Net	0.510903	0.050665	0.000924	0.000166	0.023047	0.002158	-0.000027	0.003682
Decision Tree	0.910028	0.020249	0.000172	0.000052	0.008036	0.000929	-0.000319	0.001492
Random Forest	0.972949	0.009405	0.000052	0.000022	0.004376	0.000697	-0.000095	0.000761
Gradient Boosting	0.970424	0.008986	0.000056	0.000020	0.004923	0.000731	-0.000003	0.000935

Appendix I: Volume Decrease 10-fold cross validation full results

Model	R2	MSE	MAE	ME
Linear Regression	[0.39738645 0.47417116 0.52508905 0.51399482 0.48464326 0.56666132 0.52478821 0.47428098 0.57738688 0.55812556]	[0.00095746 0.00113293 0.00089265 0.00071009 0.00102713 0.00079527 0.00098815 0.001234 0.00080086 0.00072578]	[0.02457179 0.02441144 0.02340318 0.01990613 0.0246931 0.02114679 0.02385302 0.02731866 0.02194988 0.02056759]	[-0.00682161 0.00580871 - 0.00296195 - 0.00455452 0.00459522 0.00168186 0.0002604 0.00042844 0.00026185 0.00114775]
Linear Regression (Forward Selection)	[0.38335421 0.4642021 0.51822887 0.51202971 0.5113425 0.57634142 0.51261645 0.46719418 0.56254596 0.51870297]	[0.00097976 0.0011544 0.00090554 0.00071296 0.00097391 0.0007775 0.00101346 0.00125064 0.00082898 0.00079053]	[0.02481472 0.02497239 0.02398918 0.01986804 0.02365841 0.02047704 0.02401729 0.02713239 0.02215042 0.02158894]	[-0.00676685 0.00525632 - 0.00250748 - 0.00456653 0.00481483 0.00179934 - 0.00027632 0.00140716 - 0.0003864 0.00108187]
Linear Regression (Backward Selection)	[0.40186141 0.46321161 0.51726344 0.55208796 0.50114206 0.57594971 0.52133251 0.46869356 0.58403014 0.54975883]	[0.00095035 0.00115654 0.00090736 0.00065444 0.00099424 0.00077822 0.00099534 0.00124712 0.00078827 0.00073952]	[0.02408222 0.0244431 0.02362675 0.01895669 0.02422933 0.0209682 0.02397277 0.02699042 0.0215422 0.02079475]	[-6.76272382e-03 5.42159336e-03 - 3.24016709e-03 - 4.12385190e-03 4.61898793e-03 1.42870519e-03 - 2.39016458e-05 5.98690643e-04 6.77765336e-04 9.57970443e-04]
PCA Linear Regression (All Features)	[0.30633916 0.39729423 0.46851187 0.56223271 0.4577143 0.50026943 0.39949418 0.43353402 0.48809856 0.52894854]	[0.00110212 0.00129856 0.00099899 0.00063961 0.0010808 0.00091711 0.00124869 0.00132965 0.00097006 0.0007737]	[0.0263402 0.02518747 0.02456963 0.01779087 0.02477718 0.02312985 0.02683058 0.02828486 0.02441121 0.02175375]	[-0.00710201 0.0084925 - 0.00311411 - 0.0028673 0.00330336 0.00226421 0.00059263 0.00158141 - 0.00203498 - 0.00144245]
PCA Linear Regression (Forward Features)	[0.31004324 0.39393488 0.43732544 0.55157982 0.46860155 0.50778451 0.43226644 0.39200262 0.46380974 0.53637902]	[0.00109624 0.0013058 0.00105761 0.00065518 0.0010591 0.00090332 0.00118054 0.00142713 0.00101609 0.0007615]	[0.0264481 0.02667061 0.02557943 0.01890411 0.02489124 0.02336784 0.02652651 0.02984464 0.02492819 0.02140155]	[-0.0054827 0.00751027 - 0.00333281 - 0.00388977 0.00387427 0.00330986 - 0.00106385 0.00101165 - 0.00157142 - 0.00053107]
PCA Linear Regression (Backward Features)	[0.31082189 0.40422113 0.47520881 0.54935244 0.44645377 0.50059082 0.39478922 0.43250105 0.4776945 0.5365711]	[0.001095 0.00128364 0.0009864 0.00065843 0.00110324 0.00091652 0.00125847 0.00133207 0.00098978 0.00076118]	[0.02623442 0.02504844 0.024552 0.01854463 0.02536193 0.02310747 0.02710848 0.02865515 0.0249956 0.02138616]	[-0.00691287 0.00855757 - 0.00301897 - 0.00337445 0.00331562 0.002671 0.00055145 0.00126078 - 0.00177554 - 0.00137787]

Ridge	[0.39571806 0.4727448 0.52599554 0.51750743 0.49090814 0.57564368 0.51318583 0.4744764 0.57860562 0.5488039]	[0.00096011 0.001136 0.00089094 0.00070496 0.00101464 0.00077878 0.00101228 0.00123355 0.00079855 0.00074109]	[0.02433933 0.02430544 0.02355236 0.01971699 0.02459333 0.02079851 0.02424763 0.02691393 0.02198321 0.02064319]	[-6.84878976e-03 6.03869260e-03 - 2.92776249e-03 - 4.47416822e-03 4.50873990e-03 1.98152837e-03 2.22103675e-04 4.48385885e-04 1.25224514e-05 9.74798209e-04]
Lasso	[0.39930434 0.47614183 0.52410486 0.52770053 0.50220157 0.57359429 0.51073405 0.47299508 0.57416499 0.54695145]	[0.00095442 0.00112868 0.0008945 0.00069007 0.00099213 0.00078254 0.00101738 0.00123702 0.00080697 0.00074413]	[0.02420316 0.02418961 0.02361447 0.01940594 0.02418579 0.02086084 0.02429679 0.02710286 0.02205708 0.0206212]	[-0.00689554 0.00588349 - 0.0028938 - 0.00433232 0.00448081 0.00188086 0.00010956 0.00064996 0.00011938 0.00076335]
Elastic Net	[0.39808506 0.47566732 0.52515854 0.5247304 0.49935645 0.57508798 0.51222958 0.47412362 0.57670603 0.54788719]	[0.00095635 0.0011297 0.00089252 0.00069441 0.0009978 0.0007798 0.00101427 0.00123437 0.00080215 0.00074259]	[0.02421459 0.02421538 0.02356037 0.01952064 0.0243148 0.0207747 0.02429006 0.02700404 0.02199276 0.020583]	[-6.92188771e-03 5.92984784e-03 - 2.91811980e-03 - 4.37625762e-03 4.48999133e-03 1.89538526e-03 1.15285700e-04 5.83067716e-04 8.30639788e-05 8.48222324e-04]
Decision Tree	[0.92708484 0.94489999 0.8879427 0.9252471 0.89364018 0.91861299 0.91428049 0.88744942 0.88025886 0.92085923]	[0.00011585 0.00011872 0.00021062 0.00010922 0.00021198 0.00014936 0.00017824 0.00026419 0.00022691 0.00012999]	[0.00726216 0.00706117 0.00813405 0.0072428 0.00809126 0.00803212 0.00810864 0.01012912 0.00915441 0.00714088]	[-0.00179245 - 0.00079712 - 0.00199299 0.00128102 0.00101698 0.00203681 - 0.00203683 0.00132071 - 0.00075833 - 0.00147184]
Random Forest	[0.96335465 0.97876468 0.97947548 0.97900984 0.97714243 0.98261943 0.95643973 0.96378144 0.96427319 0.98463016]	[5.82240629e-05 4.57525963e-05 3.85780508e-05 3.06683396e-05 4.55560491e-05 3.18969901e-05 9.05790735e-05 8.50147246e-05 6.77030403e-05 2.52449374e-05]	[0.00464469 0.00386662 0.00398468 0.00381984 0.00435239 0.00377912 0.00523961 0.00584144 0.00469967 0.0035353]	[-1.51968583e-03 5.76255187e-04 - 1.08054723e-03 - 2.39995933e-04 1.30531879e-03 4.90051355e-04 - 4.72389011e-05 - 1.72822393e-04 - 1.73422901e-04 - 8.46808256e-05]
Gradient Boosting	[0.95791073 0.98000458 0.9771226 0.97346286 0.96938613 0.98301005 0.95621895 0.96366629 0.96484541 0.97861042]	[6.68736450e-05 4.30811712e-05 4.30005599e-05 3.87729362e-05 6.10146589e-05 3.11801222e-05 9.10381738e-05 8.52850324e-05 6.66186623e-05 3.51323406e-05]	[0.00532848 0.00410555 0.00479126 0.00444293 0.00552615 0.00394229 0.00575526 0.00608657 0.00521862 0.00402822]	[-0.00179098 0.00050392 - 0.00108734 - 0.00011785 0.00153917 0.000948 - 0.00058989 0.00057976 0.00024443 - 0.00025467]

Appendix J: Volume Decrease feature importance

	Linear Regression	Linear Regression (Forward Selection)	Linear Regression (Backward Selection)	PCA Linear Regression (All Features)	PCA Linear Regression (Backward Features)	PCA Linear Regression (Backward Features)	Ridge	Lasso	Elastic Net	Decision Tree	Random Forest	Gradient Boosting
hour	0.000056	0.000000	0.000115	0.054487	0.000000	0.015869	0.003619	0.004569	0.004666	0.104714	0.049305	0.033316
minutes_since_start	0.000003	0.000000	0.000000	0.000081	0.000000	0.000000	0.000255	0.000012	0.000074	0.067942	0.071150	0.067100
minutes_since_eleven	0.000137	0.066817	0.000000	0.004039	0.281467	0.000000	0.018656	0.035424	0.031244	0.031061	0.137993	0.301300
baro_press_hPa	0.000449	0.000000	0.000000	0.003028	0.000000	0.000000	0.042457	0.003019	0.012131	0.065738	0.066947	0.052726
rel_humidity	0.000624	0.000000	0.000997	0.011616	0.000000	0.000923	0.053234	0.055303	0.056291	0.037547	0.040350	0.051825
temp_mean	0.595900	0.000000	0.481369	0.025748	0.000000	0.026708	0.002982	0.000000	0.000000	0.021023	0.017555	0.010611
temp_max	0.325100	0.000000	0.506118	0.019270	0.000000	0.030713	0.014974	0.000000	0.000000	0.007510	0.014470	0.013024
temp_min	0.039833	0.000000	0.000000	0.021922	0.000000	0.000000	0.004358	0.000000	0.000000	0.033830	0.016846	0.020390
temp_grass	0.003516	0.163050	0.004518	0.014438	0.000227	0.000074	0.231575	0.307020	0.304817	0.049908	0.030451	0.028833
wind_dir	0.000052	0.009079	0.000000	0.091103	0.046752	0.000000	0.002472	0.000000	0.000240	0.012258	0.017715	0.016435
wind_dir_gust	0.000251	0.000000	0.000096	0.063317	0.000000	0.134061	0.024404	0.021434	0.021946	0.042185	0.011444	0.014935
wind_sp_m_sec	0.003013	0.000000	0.000000	0.009427	0.000000	0.000000	0.028522	0.000000	0.000000	0.006767	0.049017	0.126072
wind_sp_m_1m_max	0.002641	0.000000	0.000000	0.008984	0.000000	0.000000	0.000523	0.000000	0.000000	0.021797	0.078050	0.031882
Ws10mm	0.006442	0.446671	0.000000	0.014606	0.264644	0.000000	0.034024	0.000000	0.000000	0.035757	0.144597	0.050642
wind_sp_m_sec_max	0.019446	0.000000	0.005054	0.010323	0.000000	0.107836	0.345309	0.401906	0.391962	0.077421	0.098306	0.026277
wind_dir_std	0.000299	0.073636	0.000559	0.021931	0.043473	0.024179	0.028463	0.037199	0.035948	0.018359	0.021700	0.019365
temperature	0.000163	0.000000	0.000000	0.071303	0.000000	0.000000	0.002464	0.000000	0.000000	0.021332	0.023762	0.026474
humidity	0.000662	0.209578	0.000788	0.519306	0.346617	0.610798	0.043196	0.089437	0.072261	0.293625	0.042322	0.030484
pressure	0.001414	0.031170	0.000387	0.035071	0.016819	0.048839	0.118513	0.044675	0.068421	0.051224	0.068020	0.078309

Appendix K: Weight Decrease model selection results

Model	R^2	Adj R^2	MAE	MSE	RMSE	Avg Residual
Linear Regression	0.421820	0.431389	0.013157	0.000355	0.018851	-1.568047e-17
Linear Regression (Forward Selection)	0.406938	0.413695	0.013598	0.000365	0.019105	1.030104e-18
Linear Regression (Backward Selection)	0.425546	0.432292	0.013156	0.000353	0.018785	-1.648166e-17
PCA Linear Regression (All Features)	0.251695	0.244839	0.014899	0.000461	0.021462	2.117436e-18
PCA Linear Regression (Forward Features)	0.251695	0.244839	0.014899	0.000461	0.021462	1.774068e-18
PCA Linear Regression (Backward Features)	0.251695	0.244839	0.014899	0.000461	0.021462	6.867359e-19
Ridge	0.422156	0.430924	0.013157	0.000355	0.018849	-1.156005e-17
Lasso	0.415950	0.422561	0.013333	0.000359	0.018960	-3.319223e-18
Elastic Net	0.416796	0.423452	0.013314	0.000359	0.018946	-3.891503e-18
Decision Tree	0.883456	0.999997	0.003754	0.000069	0.008327	2.682562e-21
Random Forest	0.973903	1.000000	0.002008	0.000016	0.003948	3.665184e-18
Gradient Boosting	0.974076	0.999997	0.002108	0.000016	0.003944	-2.770304e-19

Appendix L: Weight Decrease models - Selected Parameters

Model	Hyperparameter	Best value
PCA Linear Regression (All Features)	n_components	5
PCA Linear Regression (Forward Features)	n_components	5
PCA Linear Regression (Backward Features)	n_components	5
Ridge	alpha	0.01
Lasso	alpha	0.00001
	max_iter	10000
Elastic Net	alpha	0.00001
	l1_ratio	0.9
	max_iter	10000
Decision Tree	criterion	friedman_mse
	max_depth	20
	max_features	sqrt
	min_samples_leaf:	1
	min_samples_split	2
	splitter	best
Random Forest	bootstrap	False
	max_depth	30
	max_features	sqrt
	min_samples_leaf	1
	min_samples_split	2
	n_estimators	300
Gradient Boosting	learning_rate	0.1
	max_depth	9
	max_features	sqrt
	min_samples_leaf	5
	min_samples_split	2
	n_estimators	400

Appendix M: Weight Decrease test-set size robustness results

test_size	model	r2	mae
0.1	Gradient Boosting	0.968874	0.002172
	Random Forest	0.950726	0.002644
	Linear Regression	0.465542	0.012219
0.2	Gradient Boosting	0.971859	0.002153
	Random Forest	0.963575	0.002328
	Linear Regression	0.445478	0.013304
0.3	Gradient Boosting	0.964329	0.002539
	Random Forest	0.953619	0.002690
	Linear Regression	0.403196	0.013579
0.4	Gradient Boosting	0.954253	0.002704
	Random Forest	0.941499	0.002977
	Linear Regression	0.388373	0.013470
0.5	Gradient Boosting	0.942128	0.003282
	Random Forest	0.947244	0.003075
	Linear Regression	0.412763	0.013522
0.6	Gradient Boosting	0.929164	0.003626
	Random Forest	0.926233	0.003556
	Linear Regression	0.417712	0.013554
0.7	Gradient Boosting	0.836322	0.004943
	Random Forest	0.847587	0.004716
	Linear Regression	0.378315	0.014269
0.8	Gradient Boosting	0.799671	0.005986
	Random Forest	0.826615	0.005602
	Linear Regression	0.369281	0.014674
0.9	Gradient Boosting	0.700689	0.007956
	Random Forest	0.714747	0.007762
	Linear Regression	0.204681	0.016906

Appendix N: Weight Decrease 10-fold cross validation results

Model	Mean R2	Std R2	Mean MSE	Std MSE	Mean MAE	Std MAE	Mean ME	Std ME
Linear Regression	0.421820	0.075140	0.000355	0.000070	0.013157	0.001239	-0.000052	0.001345
Linear Regression (Forward Selection)	0.406938	0.073340	0.000365	0.000074	0.013598	0.001170	-0.000032	0.001494
Linear Regression (Backward Selection)	0.425546	0.069474	0.000353	0.000067	0.013156	0.001183	-0.000035	0.001314
PCA Linear Regression (All Features)	0.251695	0.032792	0.000461	0.000071	0.014899	0.001010	-0.000020	0.001502
PCA Linear Regression (Forward Features)	0.282410	0.050380	0.000441	0.000072	0.014983	0.001073	-0.000006	0.001252
PCA Linear Regression (Backward Features)	0.282004	0.051591	0.000441	0.000072	0.014996	0.001063	-0.000011	0.001264
Ridge	0.422156	0.074766	0.000355	0.000071	0.013157	0.001256	-0.000053	0.001349
Lasso	0.415950	0.074663	0.000359	0.000074	0.013333	0.001246	-0.000054	0.001398
Elastic Net	0.416796	0.074745	0.000359	0.000074	0.013314	0.001248	-0.000054	0.001394
Decision Tree	0.886309	0.052781	0.000068	0.000028	0.003727	0.000811	0.000016	0.000847
Random Forest	0.973701	0.011172	0.000016	0.000006	0.002007	0.000364	0.000057	0.000397
Gradient Boosting	0.974580	0.008637	0.000015	0.000005	0.002072	0.000336	0.000175	0.000405

Appendix O: Weight Decrease 10-fold cross validation full results

Model	R2	MSE	MAE	ME
Linear Regression	[0.46554221 0.43495178 0.30695705 0.34464005 0.48385545 0.49221553 0.50542305 0.30517883 0.49064552 0.38878992]	[0.00031718 0.00039697 0.00035113 0.000321 0.00034372 0.00028763 0.00030071 0.00053261 0.00029676 0.00040579]	[0.01221928 0.01400711 0.01375082 0.01241554 0.01363851 0.01170488 0.01260647 0.01615113 0.01205704 0.01301705]	[-0.00055453 0.00042165 - 0.002236 - 0.00163441 0.0006788 - 0.00033343 0.00102634 - 0.00077371 0.00010186 0.00278106]
Linear Regression (Forward Selection)	[0.46262656 0.39639461 0.30394939 0.33923327 0.45908146 0.45670915 0.49883164 0.28375992 0.48471086 0.38408505]	[0.00031891 0.00042406 0.00035265 0.00032365 0.00036022 0.00030775 0.00030472 0.00054903 0.00030022 0.00040891]	[0.01288184 0.01424176 0.01387598 0.01289787 0.01411423 0.01259458 0.01291742 0.01663299 0.01252902 0.0132915]	[-0.00029073 0.00051867 - 0.00220517 - 0.00193906 0.00048127 - 0.00011785 0.00069363 - 0.00119292 0.0004085 0.0033213]
Linear Regression	[0.45632648 0.44475632 0.32621335	[0.00032265 0.00039008 0.00034137	[0.01235948 0.0139617 0.01351992	[-0.00054331 0.00050365 - 0.00200777 -

(Backward Selection)	0.34876799 0.4990439 0.48693678 0.50479605 0.3163152 0.47845776 0.39384412]	0.00031898 0.00033361 0.00029062 0.00030109 0.00052408 0.00030386 0.00040244]	0.01240267 0.01342902 0.01173924 0.01270583 0.01614135 0.01225406 0.01304628]	0.0016777 0.0007887 - 0.00028588 0.00095237 - 0.0007346 - 0.0001274 0.00277838]
PCA Linear Regression (All Features)	[0.27129421 0.26802003 0.19224393 0.2790808 0.25391239 0.27711223 0.28980892 0.19152746 0.25197668 0.24197605]	[0.00043246 0.00051425 0.00040924 0.00035311 0.00049685 0.00040948 0.00043181 0.00061973 0.00043582 0.00050326]	[0.01451108 0.0145826 0.01469161 0.01341237 0.01579058 0.01387224 0.0150935 0.01730066 0.01474598 0.01499362]	[-3.40214987e-04 2.94597383e-03 - 2.57624322e-03 - 1.68585813e-03 4.83499035e-04 8.81569528e-05 1.29648609e-03 - 2.26533638e-04 - 1.16347944e-03 9.82949076e-04]
PCA Linear Regression (Forward Features)	[0.31692603 0.31490094 0.20011486 0.26647176 0.32464033 0.29925996 0.34310367 0.18960962 0.31400014 0.25507211]	[0.00040538 0.00048131 0.00040526 0.00035928 0.00044975 0.00039693 0.0003994 0.0006212 0.00039968 0.00049457]	[0.01429332 0.01515816 0.01457066 0.01384898 0.01518032 0.01450118 0.01520794 0.01792867 0.0142492 0.01489442]	[-0.00055805 0.00214638 - 0.00181014 - 0.00166103 0.00100614 - 0.00021659 0.00093199 - 0.00076141 - 0.0005245 0.00139129]
PCA Linear Regression (Backward Features)	[0.31576063 0.3200193 0.20120074 0.2632932 0.33323925 0.29851816 0.34475291 0.18605396 0.3019196 0.25528637]	[0.00040607 0.00047772 0.00040471 0.00036084 0.00044402 0.00039735 0.0003984 0.00062393 0.00040672 0.00049443]	[0.01433637 0.01515994 0.01448158 0.01383341 0.0149799 0.01454035 0.01530172 0.01794332 0.014551 0.01482917]	[-0.00073153 0.00205018 - 0.00167028 - 0.00173365 0.00115151 - 0.00023136 0.00083359 - 0.0007285 - 0.00061901 0.00156415]
Ridge	[0.46671349 0.43072657 0.31245214 0.34599882 0.48363847 0.48784057 0.50876126 0.30280788 0.49342814 0.38919128]	[0.00031649 0.00039994 0.00034834 0.00032033 0.00034387 0.00029011 0.00029868 0.00053443 0.00029514 0.00040552]	[0.01223334 0.01405783 0.01371908 0.01241015 0.01367297 0.0117325 0.01259448 0.01619235 0.0119814 0.01297782]	[-0.00046966 0.00035976 - 0.00221743 - 0.00165808 0.00062671 - 0.00030534 0.00093078 - 0.00081859 0.00016638 0.0028569]
Lasso	[0.46574229 0.40863476 0.31657275 0.34542338 0.47422405 0.46747835 0.51324576 0.29049737 0.49388443 0.38380177]	[0.00031706 0.00041546 0.00034625 0.00032061 0.00035013 0.00030165 0.00029595 0.00054387 0.00029488 0.0004091]	[0.01252009 0.01430434 0.01369424 0.01252166 0.01384999 0.01215526 0.0126936 0.01640519 0.01207845 0.01310899]	[-0.00027064 0.00017766 - 0.0021703 - 0.00176329 0.00049582 - 0.0002332 0.00072107 - 0.00096609 0.00034044 0.00312683]
Elastic Net	[0.46620169 0.41047526 0.31671011 0.34575336 0.47536897 0.46925004	[0.00031679 0.00041417 0.00034618 0.00032045 0.00034937 0.00030064	[0.01249605 0.01428557 0.01369535 0.01251051 0.01383004 0.01211356	[-0.00028433 0.00018935 - 0.0021739 - 0.00175545 0.00050499 - 0.00023812

	0.51386824 0.29160902 0.49425743 0.38446914]	0.00029557 0.00054301 0.00029466 0.00040866]	0.01267776 0.0163864 0.01205766 0.01308946]	0.00073473 - 0.00095721 0.00032983 0.00310708]
Decision Tree	[0.92540407 0.96704939 0.82125922 0.88394143 0.88452432 0.86867357 0.87269132 0.90337651 0.78406828 0.95210683]	[4.42701145e-05 2.31492705e-05 9.05577169e-05 5.68458817e-05 7.68997296e-05 7.43899114e-05 7.74053052e-05 7.40662425e-05 1.25807759e-04 3.17969394e-05]	[0.00305044 0.00252937 0.00394289 0.00293845 0.00453588 0.00383559 0.00465646 0.00374702 0.00509538 0.00294007]	[-6.63413124e-05 -5.14934127e-04 -3.81915857e-04 -1.40637838e-03 -4.79789253e-04 1.59722958e-03 1.53278338e-04 1.33510996e-03 2.79231712e-04 - 3.51836980e-04]
Random Forest	[0.94995923 0.98757003 0.96073409 0.97561332 0.97108812 0.97885888 0.96679967 0.98406387 0.97796223 0.98436081]	[2.96974701e-05 8.73260437e-06 1.98937879e-05 1.19446810e-05 1.92535416e-05 1.19753951e-05 2.01862236e-05 1.22157619e-05 1.28398125e-05 1.03830784e-05]	[0.00265262 0.00149703 0.00248361 0.00165789 0.00225982 0.00179472 0.0021898 0.00173538 0.00207093 0.00172766]	[-3.19979751e-04 -4.06255324e-05 -5.94721594e-04 2.57455107e-05 4.39476385e-04 4.57421562e-04 - 5.48462535e-04 3.81818734e-04 3.13247085e-04 4.54099689e-04]
Gradient Boosting	[0.96676807 0.98677799 0.96579356 0.97948475 0.97105331 0.97731777 0.9606888 0.98593297 0.96931708 0.98266582]	[1.97220055e-05 9.28905035e-06 1.73304440e-05 1.00484398e-05 1.92767244e-05 1.28483567e-05 2.39017118e-05 1.07830064e-05 1.78767157e-05 1.15084045e-05]	[0.00216349 0.00167503 0.00241313 0.00169593 0.00247576 0.00195616 0.00254589 0.00173576 0.00233671 0.00172672]	[0.00018355 0.00011369 - 0.00062491 0.00023678 0.0003203 0.00057483 - 0.00045423 0.0001478 0.00054815 0.00069914]

Appendix P: Weight Decrease feature importance

	Linear Regression	Linear Regression (Forward	Linear Regression (Backward	PCA Linear Regression	PCA Linear Regression	PCA Linear Regression	Ridge	Lasso	Elastic Net	Decision Tree	Random Forest	Gradient Boosting
hour	0.000 091	0.007 344	0.000 080	-0.005 779	0.017 419	0.026 358	0.000 127	0.000 760	0.000 627	0.002 631	0.070 117	0.071 107
minutes_since_start	0.000 149	0.009 583	0.000 177	0.012 661	0.032 734	0.045 320	0.000 196	0.000 932	0.000 782	0.060 384	0.022 165	0.040 387
minutes_since_eleven	0.001 693	0.165 173	0.001 387	0.009 497	0.031 024	0.034 009	0.002 350	0.013 922	0.011 499	0.157 637	0.212 356	0.249 550
baro_press_hPa	0.000 231	0.017 038	0.000 094	0.033 816	0.053 520	0.011 121	0.000 395	0.003 774	0.003 040	0.032 627	0.014 260	0.013 064
rel_humidity	0.000 254	0.087 656	0.000 000	-0.013 678	0.004 879	0.000 000	0.000 414	0.004 090	0.003 265	0.005 180	0.103 939	0.098 944
temp_mean	0.040 765	0.000 000	0.000 000	0.000 310	0.000 000	0.000 000	0.010 075	0.000 000	0.000 000	0.002 998	0.011 320	0.022 121
temp_max	0.367 020	0.000 000	0.540 270	0.001 632	0.000 000	0.000 511	0.451 575	0.511 648	0.515 604	0.003 245	0.011 547	0.012 239
temp_min	0.583 395	0.000 000	0.450 467	0.002 883	0.000 000	0.002 725	0.526 782	0.422 614	0.430 259	0.003 856	0.008 200	0.003 934
temp_grass	0.003 301	0.218 095	0.003 415	0.004 388	0.013 568	0.027 081	0.004 400	0.021 659	0.018 018	0.232 414	0.248 707	0.251 636
wind_dir	0.000 188	0.001 663	0.000 110	0.258 827	0.630 569	0.588 617	0.000 249	0.001 259	0.001 056	0.153 510	0.122 862	0.055 335
wind_dir_gust	0.000 003	0.000 000	0.000 000	0.265 790	0.000 000	0.000 000	0.000 007	0.000 164	0.000 127	0.008 081	0.009 833	0.015 758
wind_sp_m_sec	0.000 406	0.000 000	0.000 000	0.038 384	0.000 000	0.000 000	0.000 374	0.000 000	0.000 000	0.157 029	0.025 591	0.031 002
wind_sp_m_1m_max	0.000 587	0.453 457	0.000 000	0.073 132	0.064 128	0.000 000	0.000 690	0.002 991	0.002 481	0.003 280	0.029 460	0.025 769
Ws10mm	0.000 603	0.000 000	0.002 293	0.115 426	0.000 000	0.067 876	0.000 849	0.005 085	0.004 567	0.017 311	0.033 263	0.011 175
wind_sp_m_sec_max	0.000 324	0.000 000	0.000 000	0.078 152	0.000 000	0.000 000	0.000 148	0.002 588	0.001 722	0.006 471	0.013 418	0.008 806
wind_dir_std	0.000 229	0.030 791	0.000 157	0.029 636	0.007 596	0.002 261	0.000 331	0.002 300	0.001 865	0.002 114	0.003 445	0.004 011
temperature	0.000 351	0.000 000	0.000 743	0.006 442	0.000 000	0.057 144	0.000 448	0.002 234	0.001 841	0.044 861	0.030 475	0.036 722
humidity	0.000 396	0.009 200	0.000 806	0.036 864	0.144 563	0.136 975	0.000 552	0.003 449	0.002 820	0.036 275	0.014 257	0.016 889
pressure	0.000 014	0.000 000	0.000 000	0.051 617	0.000 000	0.000 000	0.000 037	0.000 532	0.000 427	0.070 096	0.014 786	0.031 550

Appendix Q: Meal Weight model selection results

Model	R2	Adj R2	MAE	MSE	RMSE	Avg Residual
Linear Regression	0.605328	0.558088	3.121468	15.746277	3.968158	2.063494e-15
Linear Regression (Forward Selection)	0.577623	0.561553	3.157399	16.806522	4.099576	9.694528e-15
Linear Regression (Backward Selection)	0.646116	0.625530	3.015634	14.544907	3.813779	1.537887e-14
PCA Linear Regression (All Features)	0.421252	0.351985	3.668883	23.665574	4.864728	-9.149454e-16
PCA Linear Regression (Forward Features)	0.091960	-0.016696	4.506888	38.737748	6.223966	-1.012280e-15
PCA Linear Regression (Backward Features)	0.091960	-0.016696	4.506888	38.737748	6.223966	-1.012280e-15
Ridge	0.576303	0.525585	3.189207	16.873846	4.107779	3.698716e-15
Lasso	0.569912	0.518428	3.192954	17.075605	4.132264	3.075774e-15
Elastic Net	0.570583	0.519179	3.193400	17.067329	4.131262	2.530700e-15
Decision Tree	0.686439	0.648909	2.090734	13.124602	3.622789	-1.533020e-16
Random Forest	0.876683	0.861926	1.537939	5.079333	2.253738	9.879464e-16
Gradient Boosting	0.880121	0.865776	1.546640	5.017110	2.239891	-4.866731e-18
Decision Tree (Desicion Tree Feature Selection)	0.664159	0.656827	2.061109	13.190201	3.631832	2.180774e-01
Random Forest (Desicion Tree Feature Selection)	0.880683	0.878077	1.512398	4.915933	2.217190	2.224096e-15
Gradient Boosting (Desicion Tree Feature Selection)	0.869567	0.866718	1.565547	5.267672	2.295141	-1.539104e-16

Appendix R: Meals Weight models - Selected Parameters

Model	Hyperparameter	Best value
PCA Linear Regression (All Features)	n_components	8
PCA Linear Regression (Forward Features)	n_components	5
PCA Linear Regression (Backward Features)	n_components	5
Ridge	alpha	0.01
Lasso	alpha	0.00001
	max_iter	10000
Elastic Net	alpha	0.0001
	l1_ratio	0.9
	max_iter	10000
Decision Tree	criterion	friedman_mse
	max_depth	10
	max_features	None
	min_samples_leaf:	1
	min_samples_split	5
	splitter	random
Random Forest	bootstrap	False
	max_depth	None
	max_features	sqrt
	max_leaf_nodes	50
	min_samples_leaf	1
	min_samples_split	2
	n_estimators	200
	min_impurity_decrease	0.01
Gradient Boosting	learning_rate	0.1
	max_depth	40
	max_features	log2
	min_samples_leaf	2
	min_samples_split	4
	n_estimators	300
Decision Tree (Selected Parameters)	criterion	absolute_error
	max_depth	40
	max_features	log2
	min_samples_leaf	2
	min_samples_split	10
	splitter	best
Random Forest (Selected Parameters)	bootstrap	False
	max_depth	10
	max_features	sqrt
	max_leaf_nodes	50
	min_impurity_decrease	0.01
	min_samples_leaf	1
	min_samples_split	2
	n_estimators	200
Gradient Boosting (Selected Parameters)	learning_rate	0.1
	max_depth	5
	max_features	log2
	min_samples_leaf	2
	min_samples_split	2
	n_estimators	100

Appendix S: Meal Weight test-set size robustness results

Model	Test Size	R2	MAE
Gradient Boosting	0.1	0.856471	1.650582
Random Forest	0.1	0.858453	1.745295
Linear Regression	0.1	0.582337	3.675447
Gradient Boosting	0.2	0.843939	1.772495
Random Forest	0.2	0.823038	1.875613
Linear Regression	0.2	0.556089	3.337820
Gradient Boosting	0.3	0.830384	1.725027
Random Forest	0.3	0.823461	1.773778
Linear Regression	0.3	0.518821	3.300395
Gradient Boosting	0.4	0.848798	1.692075
Random Forest	0.4	0.846820	1.711581
Linear Regression	0.4	0.542919	3.274087
Gradient Boosting	0.5	0.866085	1.696754
Random Forest	0.5	0.862321	1.740300
Linear Regression	0.5	0.603425	3.308050
Gradient Boosting	0.6	0.822647	1.849324
Random Forest	0.6	0.825176	1.883345
Linear Regression	0.6	0.441306	3.836810
Gradient Boosting	0.7	0.849251	1.910078
Random Forest	0.7	0.834407	1.980172
Linear Regression	0.7	0.484199	3.770528
Gradient Boosting	0.8	0.730510	2.514780
Random Forest	0.8	0.653105	2.855348
Linear Regression	0.8	-0.681326	5.848218
Gradient Boosting	0.9	0.585752	3.144358
Random Forest	0.9	0.616788	2.947516
Linear Regression	0.9	-92.574313	31.878308

Appendix T: Meal Weight 10-fold cross validation results

Model	Mean R2	Std R2	Mean MSE	Std MSE	Mean MAE	Std MAE	Mean ME	Std ME
Linear Regression	0.605328	0.147339	15.746277	3.011766	3.121468	0.301115	-0.005242	0.385127
Linear Regression (Forward Selection)	0.577623	0.168999	16.806522	2.320348	3.157399	0.245548	0.031052	0.431516
Linear Regression (Backward Selection)	0.646116	0.088002	14.544907	2.337737	3.015634	0.294533	0.013962	0.410940
PCA Linear Regression (All Features)	0.421252	0.253653	23.665574	6.607652	3.668883	0.539306	0.005415	0.722953
PCA Linear Regressio	0.469760	0.148647	22.120561	5.028581	3.627008	0.337855	0.000925	0.709443

n (Forward Features)								
PCA Linear Regression (Backward Features)	0.449651	0.172895	22.920807	5.454962	3.650935	0.434131	0.012878	0.663527
Ridge	0.576303	0.170333	16.873846	2.973158	3.189207	0.348935	0.013778	0.445581
Lasso	0.569912	0.184585	17.075605	3.294894	3.192954	0.373808	0.023861	0.422230
Elastic Net	0.570583	0.181772	17.067329	3.253247	3.193400	0.369458	0.023026	0.423730
Decision Tree	0.686439	0.178301	13.124602	8.176353	2.090734	0.660745	-0.311193	0.451967
Random Forest	0.876710	0.044398	5.077751	1.662152	1.537657	0.237926	-0.157506	0.342122
Gradient Boosting	0.880000	0.033704	5.021619	1.503423	1.548358	0.172772	-0.088505	0.333922
Decision Tree (Desicion Tree Feature Selection)	0.664159	0.320654	13.190201	12.026685	2.061109	0.595818	-0.052376	0.622639
Random Forest (Desicion Tree Feature Selection)	0.880683	0.043482	4.915933	1.646620	1.512398	0.254301	-0.105420	0.351914
Gradient Boosting (Desicion Tree Feature Selection)	0.869567	0.049785	5.267672	1.567875	1.565547	0.229515	-0.135496	0.425179

Appendix U: Meal Weight 10-fold cross validation full results

Model	R2	MSE	MAE	ME
Linear Regression	[0.58233662 0.40582568 0.31702027 0.65202842 0.74269905 0.69091528 0.76554717 0.47581791 0.69215668 0.72892998]"	[21.07158724 17.14375142 15.54240939 15.62145594 15.42189268 11.3366938 14.2920786 20.66816088 13.98916227 12.375576]	[3.67544654 3.22313103 3.21031171 3.06215471 3.25046598 2.68070878 2.94148513 3.46876443 3.03543935 2.66677668]	[0.33023074 - 0.2484253 - 0.19307372 - 0.08632257 0.04300751 0.28000037 0.39920149 - 0.88684679 0.46628094 - 0.15646836]
Linear Regression (Forward Selection)	[0.60778711 0.4351771 0.12141335 0.66272446 0.66999401 0.64136103 0.7213745 0.60428069 0.61890638 0.69321341]	[19.78758112 16.29687283 19.99379001 15.14127961 19.77962762 13.15425825 16.9848135 15.6029566 17.31783725 14.00619945]	[3.33661856 3.11429365 3.11345334 3.2310086 3.65768234 2.87681483 3.13194066 3.18232264 3.24532016 2.68453359]"	[0.34993878 - 0.4383752 - 0.12809053 - 0.41375345 0.30813031 - 0.33137309 0.63774868 - 0.57701019 0.54406684 0.3592386]
Linear Regression (Backward Selection)	[0.60652707 0.53318478 0.46166591 0.69831239 0.71607606 0.66343866 0.75382306 0.62520673 0.66261871 0.74030329]	[19.85115171 13.46905069 12.25074253 13.54363397 17.01759946 12.34448867 15.00677236 14.77785622 15.3314405 11.85633277]	[3.50027812 2.8835318 2.67228715 3.009888 3.48507751 2.89419746 2.92533701 3.05664257 3.18597637 2.54312119]	[0.42420377 - 0.20307851 - 0.47942691 - 0.10475935 0.08332816 0.45694505 0.41067075 - 0.75015618 0.4944253 - 0.19253019]
PCA Linear Regression (All Features)	[0.41713453 0.59131602 - 0.27866345 0.50613584 0.50954073 0.6181314 0.57433493 0.39197093 0.29011272 0.59250493]	[29.40621802 11.79178629 29.09824397 22.1709983 29.39674438 14.00628097 25.94824202 23.97419292 32.25903439 18.60399838]	[3.58294978 2.84760106 4.03789212 3.59469125 4.48768706 2.85198059 3.62423249 4.03197718 4.3490897 3.28072736]	[0.66246406 - 0.78612593 - 0.53227158 - 0.43980954 0.56754709 0.15773222 0.66291094 - 1.46617341 0.57292348 0.6549504]
PCA Linear Regression (Forward Features)	[0.45058944 0.57117625 0.07263808 0.53624975 0.54374204 0.55130377 0.58465839 0.44958742 0.36288466 0.57476646]	[27.71838022 12.37288028 21.10375744 20.81909719 27.34681457 16.45740292 25.31893107 21.70241196 28.95209745 19.41384023]	[3.68050927 3.07672348 3.55509118 3.59021503 4.2569468 3.17973372 3.61727702 3.83705078 4.02411022 3.45242271]	[0.90857708 - 0.75606629 - 1.00881627 - 0.26304881 0.35525579 0.3767805 0.58307799 - 1.19075181 0.40527787 0.59896647]
PCA Linear Regression (Backward Features)	[0.4614662 0.58525835 - 0.01896447 0.49515546	[27.16963584 11.96656865 23.18833525 22.66393946	[3.58964359 2.93307491 3.67226978 3.81149791	[0.64886327 - 0.59616386 - 0.77904146 - 0.34520113

	0.5245098 0.59376893 0.56082378 0.4093074 0.34703107 0.53815828]	28.49954026 14.89985421 26.77187223 23.29062719 29.67252389 21.08516947]	4.38763699 3.02945149 3.67898751 3.78567836 4.22317644 3.39793683]	0.48999953 0.21348438 0.63052415 - 1.25886876 0.54221392 0.58297266]
Ridge	[0.58410888 0.38816502 0.15216032 0.64328406 0.64791612 0.70024913 0.75341219 0.5556847 0.64881662 0.68922948]	[20.98217445 17.65331557 19.29408862 16.01401592 21.10291408 10.99434433 15.03181874 17.51906497 15.95864157 14.18808375]	[3.62433333 3.3529229 3.17470383 3.15616234 3.74629098 2.5915347 2.77011509 3.412012 3.20207032 2.86192172]	[0.24583132 - 0.53637293 0.073755 - 0.65589603 0.05659573 0.1226698 0.73892226 - 0.59077937 0.51033568 0.17272135]
Lasso	[0.57696935 0.40244748 0.08959757 0.64758382 0.6360869 0.70740588 0.75241601 0.55699173 0.64115371 0.68846592]	[21.34237207 17.24122278 20.71781436 15.82098715 21.81192394 10.73184708 15.09254518 17.46752978 16.30686323 14.22294376]	[21.34237207 17.24122278 20.71781436 15.82098715 21.81192394 10.73184708 15.09254518 17.46752978 16.30686323 14.22294376]	[3.66387311 3.33292883 3.26218721 3.13754491 3.79460998 2.51894467 2.74277207 3.36961087 3.22540441 2.88166804]" "[0.24126878 - 0.50112216 0.13987391 - 0.65964624 0.00551701 0.17420509 0.70511648 - 0.4878226 0.4921955 0.12902117]
Elastic Net	[0.57677871 0.40290537 0.09895627 0.64322365 0.63617998 0.70716754 0.7538668 0.56178022 0.63980326 0.68516332]	[21.35199003 17.22801134 20.50484053 16.01672773 21.80634521 10.74058898 15.00410619 17.2787226 16.36823136 14.37372246]	[3.65786048 3.33003744 3.25031265 3.14216524 3.79661459 2.53721637 2.73349714 3.36135932 3.23022907 2.89471164]	0.23960684 - 0.51822384 0.1314134 - 0.67114811 0.03636513 0.15118633 0.69535517 - 0.48028437 0.49630795 0.14968397]
Decision Tree	[0.68212395 0.62815628 0.64918558 0.46253485 0.77596836 0.73062609 0.91390417 0.88804049 0.30996265 0.82388659]	[16.03720395 10.7288317 7.98340145 24.12837316 13.42782429 9.88016985 5.24834079 4.41449103 31.35700452 8.04037587]	[2.27010392 1.70814188 1.91005066 2.03372735 2.40843701 2.24429932 1.4487512 1.31350474 3.79664269 1.77368461]	[-0.21358439 0.01668527 - 0.4014978 - 1.10637458 - 0.28607894 - 0.28565231 0.32465361 0.3305949 - 0.95841356 - 0.53226084]

Random Forest	[0.85659779 0.82848173 0.8469188 0.94523219 0.89094939 0.85720043 0.91987121 0.79940941 0.89397189 0.92846706]	[7.23480272 4.94882826 3.48363297 2.45868629 6.53618569 5.2376418 4.88459452 7.90915722 4.81817972 3.26580298]	[1.75137651 1.63524664 1.37354595 1.22241176 1.57822439 1.50087531 1.46358544 1.98431835 1.71177342 1.15520819]	[-0.01741605 - 0.56704039 - 0.5832352 - 0.28564414 0.6393604 0.03208584 -0.46653786 - 0.02167288 - 0.13628708 - 0.16867569]
Gradient Boosting	[0.86140632 0.88750836 0.83814373 0.92937213 0.8746886 0.84594232 0.93077346 0.8351431 0.88617126 0.91085229]	[6.99220708 3.24572892 3.68332506 3.17069055 7.51081144 5.65056974 4.22000067 6.50020113 5.17265953 4.06999753]	[1.63839191 1.48016541 1.45829331 1.38137864 1.67978864 1.54590385 1.38986203 1.93822449 1.63993335 1.33163524]	[-0.03910832 - 0.52448611 - 0.26379905 - 0.21027241 0.69282676 0.09351131 -0.44957572 0.1423984 - 0.02455045 - 0.30199716]
Decision Tree (Desicion Tree Feature Selection)	[0.89553053 0.82443091 0.46002956 0.792108 0.84441876 0.68560835 0.87868043 - 0.22943503 0.73206271 0.75815572]	[5.27060218 5.06570663 12.28798051 9.33287624 9.32510045 11.53134272 7.39555519 48.47582819 12.1757334 11.04128852]	[1.82273774 1.60388889 2.01670921 1.66757007 1.94313564 2.04240741 1.89347479 3.80405736 1.89282922 1.92427984]	[0.4298999 - 0.07672172 - 0.35641391 0.09135385 0.7898023 0.62081276 -0.46314043 - 1.44300026 0.30612654 - 0.42247428]
Random Forest (Desicion Tree Feature Selection)	[0.8770911 0.84067105 0.87539636 0.93504628 0.88927184 0.83198903 0.9447682 0.80428133 0.88392126 0.92439249]	[6.20089217 4.59712876 2.8355757 2.91596156 6.63673321 6.16235216 3.36689162 7.71706068 5.27490527 3.45182577]	[1.58313037 1.62503259 1.29230841 1.30943495 1.70221497 1.5263062 1.16919009 1.9851557 1.74620135 1.185002]	[0.24420066 - 0.54045259 - 0.28866628 - 0.30565114 0.69289471 - 0.0403517 -0.41585277 0.13417437 - 0.21440189 - 0.32009707]
Gradient Boosting (Desicion Tree Feature Selection)	[0.87537253 0.86662227 0.79295937 0.89685529 0.87909077 0.80430882 0.94501724 0.80710338 0.90398709 0.92435379]	[6.28759609 3.84835646 4.71157491 4.63046579 7.24695781 7.17761476 3.35171008 7.60578883 4.36306413 3.4535925]	[1.73682847 1.53765776 1.63036115 1.49644541 1.74582752 1.63846861 1.15453017 1.96915228 1.52135057 1.22484752]	[0.13434305 - 0.69726479 - 0.26110346 - 0.42436908 0.85892584 - 0.04230026 -0.40839042 0.22815188 - 0.32404649 - 0.4189111]

Appendix V: Meal Weight models - feature importance

	Linear Regression	Linear Regression (Forward Selection)	Linear Regression (Backward Selection)	PCA Linear Regression (All Features)	PCA Linear Regression (Forward Features)	PCA Linear Regression (Backward Features)	Ridge	Lasso	Elastic Net	Decision Tree	Random Forest	Gradient Boosting
meal_volume	8.271212e-07	0.000134	0.000001	0.685599	0.835578	0.839039	4.488227e-04	5.022015e-04	6.031411e-04	0.458683	0.203427	0.389024
meal_duration	2.443910e-09	0.000000	0.000000	0.000780	0.000000	0.000000	2.262628e-06	2.536847e-06	3.250186e-06	0.059820	0.039459	0.072693
seconds_since_start	3.816868e-07	0.000082	0.000000	0.000505	-0.000405	0.000000	1.263688e-04	1.451516e-04	1.838794e-04	0.003902	0.009511	0.005714
baro_press_hPa_mean	1.510578e-06	0.000215	0.000003	0.076281	0.092709	0.104784	2.681204e-04	3.736590e-04	2.980722e-04	0.091981	0.087317	0.066117
rel_humidity_mean	4.485526e-09	0.000000	0.000000	0.001188	0.000000	0.000000	8.398021e-05	1.214793e-04	1.132587e-04	0.011354	0.011809	0.014175
temp_mean_mean	1.281041e-03	0.622080	0.005940	0.000071	-0.000428	0.001616	3.641686e-01	2.386647e-01	2.260827e-01	0.019775	0.050789	0.013096
temp_max_mean	5.376809e-04	0.064755	0.001809	0.003952	0.002910	0.001744	2.426699e-02	2.543775e-02	2.002355e-02	0.000396	0.021079	0.035675
temp_min_mean	1.573569e-04	0.282001	0.001081	0.002096	-0.000399	0.000636	2.246631e-01	1.370005e-01	1.436618e-01	0.000364	0.025780	0.000129
temp_grass_mean	4.432794e-06	0.000471	0.000012	0.012171	0.002961	0.007573	1.338492e-03	9.909042e-04	1.173978e-03	0.000582	0.000726	0.004353
wind_dir_mean	4.117580e-07	0.000000	0.000000	0.000811	0.000000	0.000000	6.144795e-05	9.824673e-05	8.082835e-05	0.002215	0.006343	0.003116

wind_dir_gust_mean	8.287055e-07	0.000000	0.000000	-0.000367	0.000000	0.000000	2.047815e-04	4.008738e-04	3.734150e-04	0.002281	0.004924	0.022121
wind_sp_m_sec_mean	1.281846e-05	0.000000	0.000007	0.000406	0.000000	-0.000180	6.185010e-03	9.360262e-03	9.918618e-03	0.038375	0.003665	0.004847
wind_sp_m_1m_max_mean	4.698982e-06	0.000000	0.000000	0.000849	0.000000	0.000000	5.677053e-03	1.107174e-02	1.153147e-02	0.009911	0.005861	0.002950
wind_sp_m_sec_max_mean	1.820159e-06	0.000000	0.000007	0.023269	0.000000	0.004216	6.798179e-05	5.069656e-06	5.374379e-07	0.001006	0.006576	0.002600
wind_dir_std_mean	6.411883e-08	0.000028	0.000000	0.094723	0.020023	0.000000	3.246248e-08	8.079942e-07	2.864492e-11	0.010548	0.063451	0.042527
rain_mm_mean	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000
baro_press_hPa_sum	4.567787e-07	0.000000	0.000000	0.002038	0.000000	0.000000	1.302254e-02	2.484210e-02	2.674963e-02	0.060110	0.007984	0.018726
rel_humidity_sum	1.458671e-06	0.000000	0.000012	-0.000286	0.000000	0.001574	8.363805e-03	1.301426e-02	1.304573e-02	0.033711	0.007541	0.002792
temp_mean_sum	5.481406e-01	0.000000	0.494392	0.001234	0.000000	0.004323	3.725703e-03	1.269385e-02	9.996843e-03	0.000000	0.005479	0.009629
temp_max_sum	4.724712e-03	0.000000	0.000000	0.000481	0.000000	0.000000	4.708064e-03	6.041769e-05	2.384170e-04	0.007209	0.006414	0.002531
temp_min_sum	4.443688e-01	0.000000	0.495554	-0.000402	0.000000	0.004005	1.165824e-02	4.821987e-03	3.387021e-03	0.000162	0.005771	0.005053
temp_grass_sum	1.246223e-05	0.000000	0.000000	0.000677	0.000000	0.000000	7.698123e-04	1.354082e-05	8.396278e-06	0.000000	0.010621	0.009895
wind_dir_sum	4.275261e-05	0.018068	0.000000	0.000029	0.009140	0.000000	2.530233e-02	3.639840e-02	3.631339e-02	0.003504	0.006373	0.002987
wind_dir_gust_sum	4.043033e-05	0.010903	0.000000	0.001395	0.015116	0.000000	1.699148e-02	3.018478e-02	2.838204e-02	0.096231	0.005233	0.001982
wind_sp_m_sec_sum	1.920648e-04	0.000000	0.000317	-0.000607	0.000000	0.002034	1.241981e-01	2.192628e-01	2.256967e-01	0.004097	0.020713	0.003521
wind_sp_m_1m_max_sum	6.86E-06	0	0	0.001597	0	0	1.17E-02	8.56E-02	7.54E-02	0.000235	0.015551	0.012098

wind_sp_m_sec_max_sum	2.81E-04	0	0.000304	0.0000002	0	0.0003506	8.28E-02	5.50E-02	6.90E-02	0.0008356	0.0129778	0.0024122
wind_dir_std_sum	6.35E-06	0.0006331	0.0000008	-0.000319	0.0008939	0.000009	1.55E-03	2.36E-03	2.18E-03	0.0003009	0.000602	0.0045745
rain_mm_sum	0.00E+00	0	0	0	0	0	0.00E+00	0.00E+00	0.00E+00	0	0	0
temperature_mean	5.47E-06	0.0006333	0.0000004	0.0144441	0.0138536	0.0006006	4.33E-03	5.79E-03	7.03E-03	0.000763	0.1540545	0.1520333
humidity_mean	8.50E-09	0	0	0.000048	0	0	2.63E-05	3.23E-05	2.95E-05	0.0013996	0.0192006	0.0060009
pressure_mean	8.37E-08	0	0	0.0432999	0	0	1.86E-04	1.92E-04	3.36E-04	0.0289556	0.1478775	0.0633
temperature_sum	8.04E-05	0	0.0002844	0.0189145	0	0.0021344	3.76E-02	5.17E-02	5.32E-02	0.0032022	0.0072578	0.0016345
humidity_sum	2.56E-07	0	0	0.0112845	0	0	1.67E-03	2.36E-03	2.49E-03	0.0292335	0.0051544	0.0040017
pressure_sum	9.16E-05	0	0.0002655	-0.000136	0	0.0169	2.38E-02	3.14E-02	3.24E-02	0.0044468	0.0085532	0.0174887

תקציר

מטרת המחקר - לפתח מערכת מבוססת מצלמות תלת-מימד למדידה וחיזוי צריכת המזון של פרות ברפת. המערכת מזהה פרות באזורי האכילה, מחשבת את נפח המזון, ומתחשבת בגורמים סביבתיים המשפיעים על צפיפות המזון. פותחו מודלים לחיזוי צריכת המזון הפרטנית של הפרות.

הנתונים נאספו באמצעות מערכת מצלמות תלת-מימד, משקלים ומדי מזג אוויר ברפת המחקר המסחרית של מכון וולקני. פותחו אלגוריתמים לזיהוי פרות באמצעות מודל YOLOv8 ואלגוריתמים לחישוב נפח המזון באמצעות עיבוד צילומי תלת מימד. בנוסף ללמידת מכונה, נבנו מודלי רגרסיה שונים לחיזוי ירידת נפח ומשקל המזון וכן משקל הארוחה, על בסיס גורמים סביבתיים.

בחיזוי ירידת נפח ומשקל המזון, מודלי למידת מכונה מתקדמים כמו Random Forest ו-Gradient Boost השיגו ביצועים טובים יותר ממודלים ליניאריים, עם ערכי R^2 של 0.973 בחיזוי ירידת הנפח ו-0.974 בחיזוי ירידת המשקל. המערכת זיהתה בהצלחה גורמים סביבתיים מרכזיים המשפיעים על צריכת המזון. חיזוי משקל הארוחה השיג R^2 של 0.88 ו-MAE של 1.537 ק"ג.

המחקר מציג גישה כמעט ללא מיכון או עבודת ידים למדידת צריכת מזון פרטנית ברפת מסחרית, המשלבת ראיית מחשב ומודלים מתקדמים של למידת מכונהMachine Learning- ולמידה עמוקה – Deep Learning. הממצאים מדגישים את חשיבות שילוב גורמים סביבתיים בחיזוי צריכת המזון, ויספקו בעתיד, כלי יעיל לניהול תזונה מדויק יותר ברפתות.

אוניברסיטת בן גוריון בנגב

הפקולטה למדעי ההנדסה

המחלקה להנדסת תעשיה וניהול

מערכת ראייה ממוחשבת ומודלים חיזויים למדידת צריכת מזון פרטנית בפרות חלב

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: בנימין כץ

מנחים: פרופ' יעל אידן ופרופ' אילן הלחמי

חתימת המחבר	בנימין כץ	תאריך:	29/08/2024
אישור המנחה/ים	Yael Edan	תאריך:	29/08/2024
אישור המנחה/ים	Halachmi	תאריך:	29/08/2024
אישור יו"ר ועדת תואר שני מחלקתית		תאריך:	

אב ה'תשפ"ד

אוגוסט 2024

אוניברסיטת בן גוריון בנגב
הפקולטה למדעי ההנדסה
המחלקה להנדסת תעשיה וניהול

מערכת ראייה ממוחשבת ומודלים חיזויים למדידת צריכת מזון פרטנית בפרות חלב

חיבור זה מהווה חלק מהדרישות לקבלת תואר מגיסטר בהנדסה

מאת: בנימין כץ

מנחים: פרופ' יעל אידן ופרופ' אילן הלחמי

אב ה'תשפ"ד

אוגוסט 2024