Politécnico de Viseu

Escola Superior de Tecnologia e Gestão de Viseu

Enhancing Interpretability of Neural Networks in Food Recommendation Systems

João Edgar Lucas Rebelo

Trabalho de Projeto
Mestrado em Engenharia Informática - Sistemas de Informação

Trabalho efetuado sob a orientação de
Professor Doutor Carlos Augusto da Silva Cunha
Professor Doutor Rui Pedro Monteiro Amaro Duarte

Maio de 2024

Politécnico
de Viseu

Escola Superior
de Tecnologia
e Gestão de Viseu

Enhancing Interpretability of Neural Networks in Food Recommendation Systems

João Edgar Lucas Rebelo

Trabalho de Projeto
Mestrado em Engenharia Informática - Sistemas de Informação

Trabalho efetuado sob a orientação de

Professor Doutor Carlos Augusto da Silva Cunha
Professor Doutor Rui Pedro Monteiro Amaro Duarte

Maio de 2024

Agradecimentos

Gostaria de agradecer a todas as pessoas que contribuíram para a realização desta tese de mestrado. Em primeiro lugar, gostaria de agradecer aos meus orientadores, Professor Doutor Carlos Augusto da Silva Cunha e Professor Doutor Rui Pedro Monteiro Amaro Duarte, pela orientação, paciência e apoio inestimáveis durante todo o processo de pesquisa e escrita desta tese. Agradeço também aos meus amigos e familiares, pelo encorajamento e apoio incondicional em todos os momentos.

Abstract

Over the years the risk of developing diseases related to poor alimentation has been increasing. Many of these diseases are caused by obesity. Obesity is a silent disease related to being overweight, which due to its rapid growth has become a public health problem. Worldwide obesity has nearly tripled since 1975. Obesity can lead to health problems like type 2 diabetes, cardiovascular disease, and even cancer. The main factors that result in obesity are a sedentary lifestyle and a poor diet. Although obesity is uncured, it can be avoided/treated through a healthier lifestyle and diet. Amid so much information about diets and healthier recipes, it can be difficult to find a diet that meets the needs of each person. Recommendation systems can filter from a large dataset, the information that best suits the profile of each user. Due to the constant increase in information and computational power, recommendation systems have evolved from a traditional approach to a deep-learning one. Recommendation systems are a hot topic in deep learning. Research in the food recommendation systems area has seen little development when compared to recommendations systems in other areas, such as leisure and entertainment. A powerful tool to use in food recommendation systems is neural networks. Neural networks play an important role in our society, for their capacity to learn from complex and high-dimensional data. One side down of neural networks is the difficulties if not impossibility in understanding how the predictions are being made. The behind-the-scenes often remain opaque, leading neural networks to be characterized as "black boxes". With this research, we aim to give contribute to understanding how neural networks operate underneath and make them more transparent and so more trustworthy. With this goal in mind, we propose the use of a secondary model to predict the errors of a primary neural network. By analyzing the error predictions of the second model, we aim to gain insights into its decision-making process. With this approach, we hope not only to help to understand the functioning of neural networks but also to provide an idea of how to improve their performance. Improving neural networks' understanding can make them more simple and accessible. With the work developed through this research, we look to stride towards making neural networks more transparent and explainable, thereby enhancing trust in these powerful models.

Keywords: Calories, Deep Learning, Machine Learning, Error, MAE, MSE, RMSE, Neural Networks, Predictions, MLP, RF, SHAP.

Resumo

Ao longo dos anos, o risco de desenvolver doenças relacionadas a má alimentação tem aumentado. Muitas destas doenças são causadas pela obesidade. A obesidade é uma doença silenciosa relacionada com o excesso de peso, que devido ao seu rápido crescimento tornou-se um problema de saúde pública. A nível mundial a obesidade quase que triplicou desde 1975. A obesidade pode levar a problemas de saúde como diabetes do tipo 2, doenças cardiovasculares e até mesmo cancro. Os principais fatores que resultam na obesidade são um estilo de vida sedentário e uma dieta pobre. Embora a obesidade não tenha cura, pode ser evitada/tratada através da adoção de um estilo de vida e de uma dieta mais saudáveis. No meio de tanta informação sobre dietas e receitas saudáveis, pode ser difícil encontrar uma dieta que satisfaça as necessidades de cada pessoa. Os sistemas de recomendação podem filtrar a partir de um grande conjunto de dados a informação que melhor se adapta ao perfil de cada utilizador. Devido ao constante aumento de informação e de poder computacional, os sistemas de recomendação evoluíram desde uma abordagem tradicional para uma abordagem de deep learning. Os sistemas de recomendação são um tema quente na área de deep learning. A investigação na área dos sistemas de recomendação alimentar tem visto pouco desenvolvimento quando comparada com os sistemas de recomendação em outras áreas, como lazer e entretenimento. Uma ferramenta poderosa a utilizar nos sistemas de recomendação alimentar são as redes neurais. As redes neurais desempenham um papel importante na nossa sociedade, pela sua capacidade de aprender a partir de dados complexos e de alta dimensão. Um dos lados negativos das redes neurais é a dificuldade, se não a impossibilidade, de compreender como as previsões estão a ser feitas. Os processos de decisão permanecem frequentemente opacos, levando as redes neurais a serem caracterizadas como "caixas pretas". Com esta investigação, pretendemos contribuir para a compreensão de como as redes neurais operam debaixo dos panos e torná-las mais transparentes e, portanto, mais confiáveis. Com este objetivo em mente, propomos o uso de um segundo modelo para prever os erros de uma rede neural. Ao analisar as previsões de erro do segundo modelo, pretendemos obter noções sobre o processo de tomada de decisão da rede neural. Com esta abordagem, esperamos não só ajudar a entender o funcionamento das redes neurais, mas também fornecer uma ideia de como melhorar o seu desempenho. Melhorar a compreensão das redes neurais pode torná-las mais simples e acessíveis. Com o trabalho desenvolvido através desta investigação, procuramos avançar no sentido de tornar as redes neurais mais transparentes e explicáveis, aumentando assim a confiança nestes modelos poderosos.

Palavras-chave: Calorias, Deep Learning, Machine Learning, Erro, MAE, MSE, RMSE, Redes Neurais, Previsões, MLP, RF, SHAP.

Summary

List of Tables

List of Figures

Listings

Acronyms

ANN Artificial Neural Network

DT Decision Tree

MAE Mean Absolute Error

MLP Multilayer Perceptron

MSE Mean Squared Error

RF Random Forest

RMSE Root Mean Squared Error

SHAP SHapley Additive exPlanations

1 Introduction

This chapter presents the motivation behind the research, an overview of the study, problem definition, research goals, expected results, work plan, and the document's structure.

1.1 Motivation and Overview

Obesity is one of the leading risk factors for noncommunicable diseases worldwide (Allom et al., 2018). This disease is mainly caused by an energy imbalance between calories consumed and expended. This happens because of the increased intake of food high in fat and sugars and increased physical inactivity (Puska et al., 2003). With the increased production of processed foods and the changes in habits, people are consuming more food rich in energy, fats, free sugars, salt/sodium, and less fruit and vegetables (Organization, 2020). People suffering from obesity exhibit a raised BMI (Body Mass Index). They have an increased risk of high blood pressure (hypertension), high LDL cholesterol, low HDL cholesterol, or high levels of triglycerides (dyslipidemia), type 2 diabetes, coronary heart disease, stroke, gallbladder disease, osteoarthritis, sleep apnea, breathing problems, cancers like breast, prostate, liver, kidney, and colon (Puska et al., 2003). Moreover, they see their quality of life decrease, manifested in several ways: mental illness such as clinical depression, anxiety, and other mental disorders, body pain, and difficulty with physical functioning (for Disease Control & Prevention, 2022).

A healthier diet helps prevent obesity, a risk factor for many non-communicable diseases (Organization, 2020). General rules for a healthier diet plan include consuming fruits, vegetables, legumes, nuts, and whole grains. However, acceptance of a specific diet may depend on culture, eating habits, and personal tastes. Moreover, a personalized diet plan also depends on personal achievements, such as the number of calories ingested, the distribution of macronutrients and micronutrients, and the goal of weight loss or gain. Personalized diet plans aligned with people's goals and preferences can be created using food recommendation systems (Kumar et al., 2016). Food recommendation systems can benefit from the tools provided by recommendation systems supported by artificial intelligence approaches (Rostami et al., 2022). Recommendation systems help users identify or receive accurate recommendations for decisions and services by reducing the effort of information research and analysis required for making a decision (Rostami et al., 2022).

Machine Learning recommendation systems have gained ground over traditional approaches due to their ability to model complex patterns and their lack of assumptions about data (Mu, 2018). Machine learning techniques extract features from data by combining low-level features from denser high-level semantic abstractions (Mu, 2018). Recommendation systems powered by neural networks, also known as Artificial Neural Networks (ANNs), have become popular. Like

the neural network in our brains, an ANN consists of various nodes (neurons) connected in a layered structure. These connections, formed by artificial neurons, resemble synapses in the human brain, enabling the flow of information and facilitating the ANNs to adapt and learn. The layered structure is formed by one input layer, one or more hidden layers, and an output layer. ANNs bridge the fields of computer science and neuroscience. ANNs draw inspiration from the complex and fascinating human brain, aiming to replicate its biological neural network through interconnected artificial neurons. These artificial neurons mimic the elemental units of the biological nervous system (Jeon & Kim, 2023) and can perform mathematical computations.

ANNs enable computers to learn and model nonlinear and complex input and output data relationships. They play a crucial role in decision-making processes, such as aiding doctors in diagnosing diseases and creating personalized treatment plans for each patient. Despite their utility, neural networks are often called "Black Boxes" because their internal workings are not fully transparent. The term "Black Boxes" refers to a system where we can observe the inputs and outputs of the model, but understanding how predictions are generated remains challenging. Understanding the process behind predictions is crucial, particularly in fields like healthcare. Having insights into why a diagnosis or treatment recommendation is made can be vital.

This research aims to develop techniques for interpreting neural networks and determining prediction errors. By employing machine learning techniques, feature importance analysis, model interpretability techniques, and the Euclidean distance technique, we seek to shed light on how neural networks make predictions by predicting the error associated with the predictions. Understanding the rationale behind these predictions is essential, particularly in fields like healthcare, where insights into diagnosis and treatment recommendations are critical for decision-making. By unravelling the inner workings of neural networks, we can enhance interpretability, fairness, reliability, debugging, and overall model improvement.

1.2 Problem Definition

Despite widespread adoption and predictive capabilities, neural networks often operate as opaque "Black Box" models, obscuring the underlying mechanisms driving their predictions. This lack of transparency in recommendation systems poses significant challenges, particularly in understanding the rationale behind their decisions. To address this issue and contribute to advancing neural networks, this research explores the feasibility of predicting errors generated by ANNs.

The central focus of this study revolves around developing and evaluating a methodology for predicting errors inherent in ANNs. This methodology involves training a secondary model specifically designed to capture and analyze the er-

rors produced by the primary neural network model. Leveraging techniques such as feature importance analysis and Euclidean distance measurement, this work aims to elucidate the factors influencing neural network decisions and identify patterns contributing to prediction errors.

By systematically analyzing error predictions, the objective is to uncover valuable insights into the inner workings of neural networks, thus enhancing their interpretability, trustworthiness, reliability, and overall performance. Through this endeavour, we aspire to bridge the gap between the predictive power of neural networks and the need for transparency and explainability in decision-making processes, ultimately fostering greater confidence and utility in neural network models across diverse applications, including recommendation systems.

1.3 Research Goals and Objectives

The main goals of our research are: (1) to study the relationship between input features and model output in the nutrition domain using model interpretability methods; (2) to evaluate the performance of a model used to predict the error of the first model, assuming it is possible to determine the error associated with neural network predictions; and (3) to understand if it is possible to determine the error associated with predictions using the difference between the values used in the test set and the nearest records in the training set, weighted by the impact on the result using the SHAP values. This research also aims to gain in-depth knowledge in machine learning.

To achieve the goals, the following objectives must be pursued.

- O1. Conduct a thorough review of the literature on personalized diet planning, food recommendation systems, error prediction, and machine learning models.
- O2. Collect and preprocess food-related data to develop a personalized food recommendation system.
- O3. Explore and apply machine learning techniques to extract features from the data and create accurate food recommendations.
- O4. Contribute to the understanding of how neural networks make predictions
- O5. Propose the use of a secondary model to predict the errors of a primary neural network
- O6. Collect and preprocess data to develop an error prediction system.
- O7. Evaluate the proposed system's effectiveness in predicting neural network errors.

- O8. Gain insights into the decision-making process of neural networks by analyzing the error predictions
- O9. Draw conclusions based on the research findings and propose recommendations for future research in the field of error predictions diet planning and food recommendation systems.

1.4 Research Questions

In this research, research questions are crucial in connecting the goals with the objectives presented in the previous section. These questions provide clear focus points around which the study revolves. Overall, research questions serve as a bridge between the goals and the practical actions required to accomplish them. To this end, the following research questions are addressed:

- RQ1. How can a second model predict the error of a primary neural network, and what insights can be gained from its error predictions?
- RQ2. Which data structure is required to build an error prediction system?
- RQ3. How effective is an error prediction model including features describing the weight of specific inputs on the output obtained using model interpretability techniques?
- RQ4: What insights can be gained into the decision-making process of neural networks by analyzing the error predictions?
- RQ5: What conclusions can be drawn based on the research findings and what recommendations can be proposed for future research in the field of error predictions, diet planning, and food recommendation systems?

1.5 Expected Results

After conducting this research it is expected to enhance the interpretability of neural networks by predicting their errors using a second model. This approach aims to make ANNs more transparent, thereby facilitating an understanding of their decision-making process. Improving neural network interpretability, it is also expected to improve their reliability and trustworthiness, fostering a deeper comprehension of the rationale behind the predictions.

1.6 Work Plan

The work plan presented in Figure 1 outlines the timeline and tasks undertaken during the search and thesis writing.

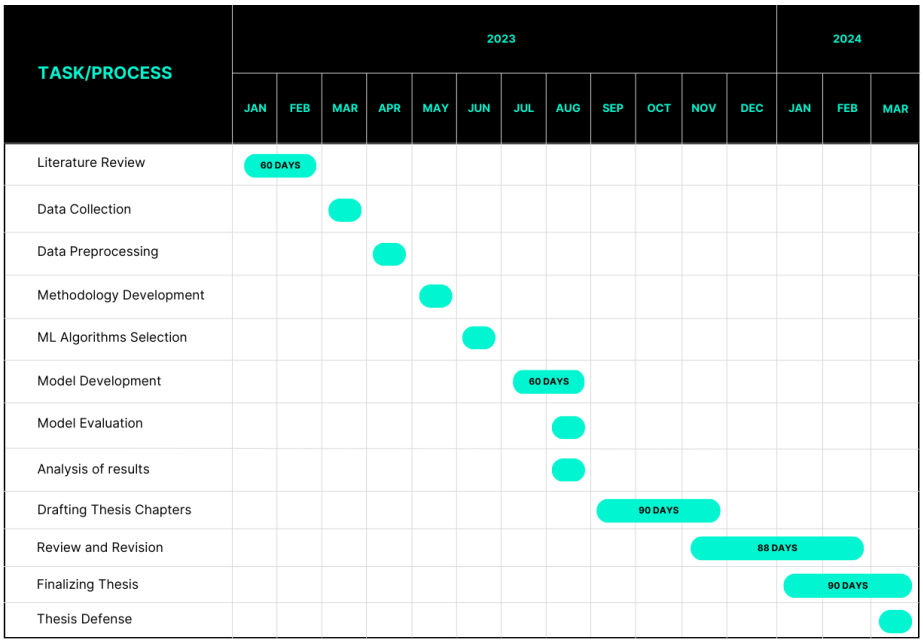

Figure 1: Work Plan

1.7 Document Structure

This document is structured into six chapters: (1) Introduction, (2) Literature Review, (3) Methodology, (4) Experimental Results, (5) Discussion, and (6) Conclusion. The Literature Review provides important background explanations, related work concepts, methodologies, and findings. The Methodology chapter explains the data used, collection methods, preprocessing techniques, training process, and evaluation metrics for assessing model performance. In the Experimental Results chapter, model outputs are presented using graphs and accompanied by explanatory and analytical text. The Discussion chapter addresses the research objectives and questions, highlights research limitations, and suggests areas for future work. Finally, the Conclusion chapter summarizes the overall findings and conclusions of the research.

2 Literature Review

This chapter provides background information on machine learning concepts and reviews related machine/deep learning work.

2.1 Background on Machine Learning

In this section, we discuss some fundamental concepts in machine learning, focusing on the neural network Multilayer Perceptron (MLP) and the Random Forest (RF) algorithm. Later in the section, we delve into concepts for data splitting, normalization/standardization, and evaluation metrics.

2.1.1 Multilayer Perceptron

Multilayer Perceptron (MLP) is a type of ANN. An ANN is a computational model miming how nerve cells work in the human brain (Yegnanarayana, 2009). Like in the human brain, where the pivotal structure in the biological nervous system is the neuron, in ANN, that pivotal element is called an artificial neuron. Before presenting the MLP architecture is essential to understand the essence of artificial neurons. They are the main processing units within the network. Artificial Neurons are the principal element, the building blocks in an artificial neural network (ANN). They are nothing more than a mathematical function developed to mimic the neurons present in the biological neural network. Figure 2 shows a comparison between a biological neuron and an artificial neuron.

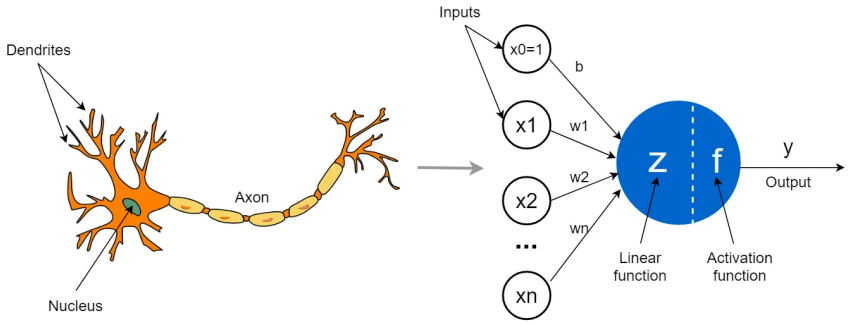

Figure 2: Neuron - Artificial Neuron, from (Pramoditha, 2021)

Artificial neurons are simple computational units with weighted input signals and produce an output signal using an activation function (Krogh, 2008). The artificial neuron was first proposed by McCulloch and Pitts in 1943. Receives inputs with values 1 or 0, where 1 represents an electrical impulse, and 0 represents no electrical impulse (Nwadiugwu, 2020), similar to what happens in our brains. A neuron is formed by inputs with associated weights and an activation

function.

An MLP is a fully connected feed-forward multi-layer neural network. A feed-forward neural network is one of the simplest types of artificial neural networks devised (Sazli, 2006). The information flows in one direction (forward) through each layer. Three or more layers of nodes form this type of ANN. An input layer, one or more hidden layers, and an output layer. If it gets more than one hidden layer goes from being a normal ANN to a deep ANN. As the name suggests, the input layer receives the features as inputs. Each feature will be a node (neuron), and each node will connect to a node in the next layer carrying a weight. The dimensions of the input data will determine the number of nodes (neurons) in the input layer. Hidden Layers can be considered as the computational engine of the neural network (Sazli, 2006). Each node in each hidden layer will take the value from every node in the previous layer, multiply them by the corresponding weights, sum everything, apply the activation function, and pass the information to the next layer. This process will be repeated in every hidden layer until it reaches the output layer. The last layer (output layer) will take the values from the last hidden layer and make the predictions. The predictions will be made based on the problem type. Figure 3 presents a scheme of the MLP architecture.

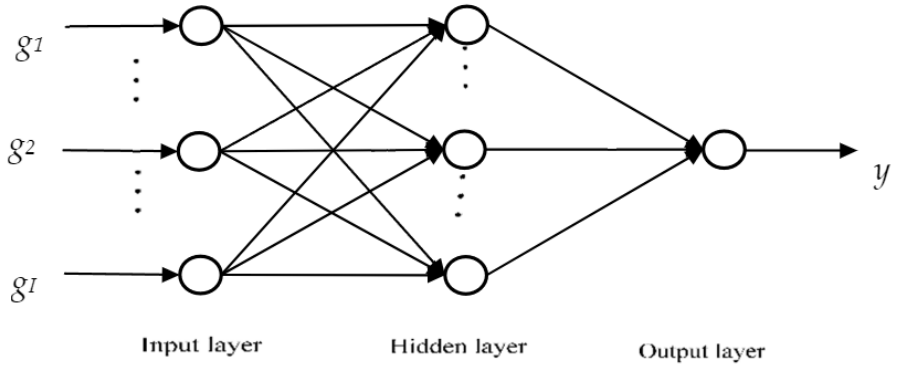

Figure 3: MLP Architecture from (Isabona et al., 2022)

After carrying out an overview of the architecture of MLP, it is important to cover some of the most used activation functions. These functions are important parts of the model since they determine how the hidden layers operate. According to Jason Brownlee, the three activation functions to consider using in hidden layers are Rectified Linear Activation (ReLU), Logistic (Sigmoid), and Hyperbolic Tangent (Tanh) (Sharma et al., 2017).

ReLU is probably the most commonly used activation function. A simple and effective function that returns 0 if it receives any negative input, but for any positive value x, it returns that value. The mathematical function is $f(x) = max(0, x)$. One key advantage of ReLU, compared to other activation

functions, is that it uses fewer resources because it involves simpler mathematical operations. Another advantage is that it avoids and rectifies the vanishing gradient problem (Hu et al., 2018). Usually, the same activation function is used in all hidden layers, and ReLU is not recommended in the output layer. The Sigmoid activation function takes any real value as input and outputs values from 0 to 1 (Brownlee, 2021). The mathematical function is $f(x) = \frac{1}{1+e^{-x}}$. Tanh, a hyperbolic tangent, is very similar to Sigmoid. Instead of output values in the range 0 to 1, it outputs values from -1 to 1. The bigger the input value, the closer the output is to 1; on the other side, the smaller the input value, the closer to -1. The mathematical function is $f(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$.

The choice of the activation function to use in the hidden layers will be based on the type of neural network architecture (Hayou et al., 2018). In the specific case of this research, where MLP was used, the most appropriate activation function is ReLU (Figure 4).

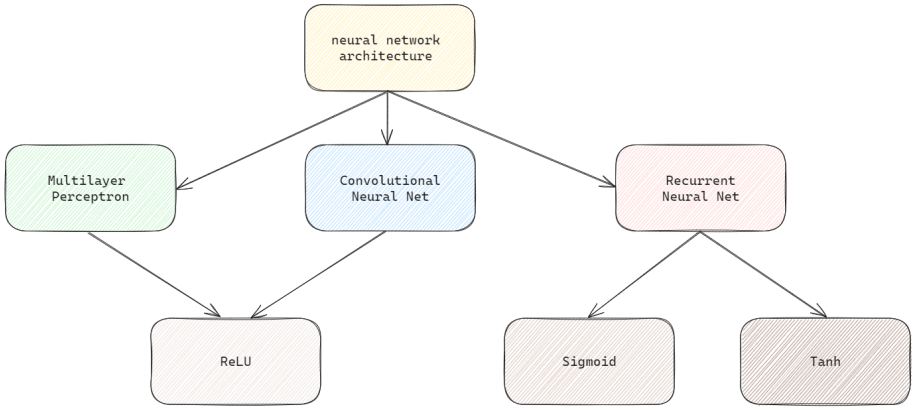

Figure 4: Activation Function to use based on the model

2.1.2 Random Forest Algorithm

Random Forest (RF) is one of the most popular machine-learning models for classification and regression problems, it is formed by multiple decision tree algorithms. It combines the predictions of all the decision tree models (Breiman, 2001). For this reason, before presenting the RF architecture is important to understand the DT (Decision Tree) model.

A Decision Tree (DT) is a non-parametric supervised learning algorithm (Kotsiantis, 2013). Its architecture is reminiscent of a tree, with a root, branches, and leaves, as shown in Figure 5. Using the tree representation will take a series of decisions and their possible outcome. It will split the data recursively into two parts upon a particular criterion until the condition is met. Decision trees can be used for regression and classification tasks based on the criterion used

in the decision process. The standard splitting criteria for classification and regression tasks are entropy and mean squared error. The starting point is the root node, where the training data is located. The data goes into the splitting process from the root node, dividing it into two or more sub-nodes. Creating sub-nodes increases the homogeneity of resultant sub-nodes. The splitting process will be repeated until a decision or result is reached. The leaf nodes represent the predictions/outcome of the model. Due to its more straightforward way of operating, it is easier to construct than the numerical weights in the neural network of connections between nodes. The algorithm used for splitting the data is selected based on the type of the target variables. Some of the algorithms that the decision tree can use are ID3 (Iterative Dichotomiser 3), C4.5, CART, CHAID, and MARS. Figure 5 illustrates the recursive split. It starts with the question "Is there a swell?" where the data is divided into the sub-nodes "YES" and "NO". The left branch is then divided using the criterion "High wind".

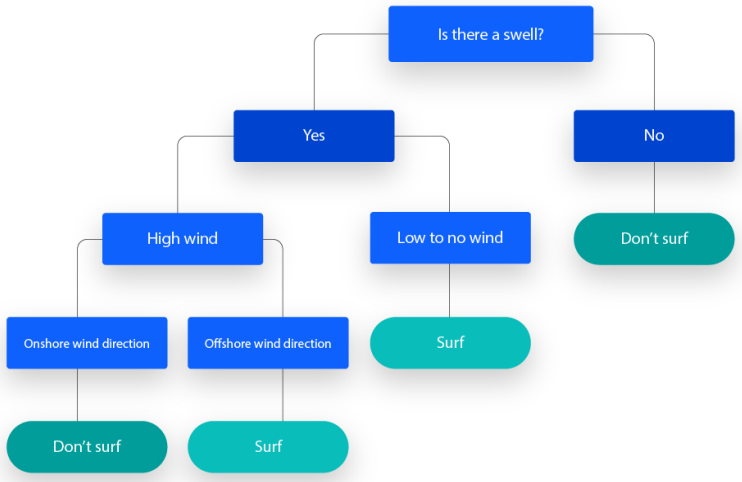

Figure 5: Decision Tree Architecture, from (IBM, n.d.-a)

The major drawback of the decision tree model is that it is susceptible to overfitting (Ying, 2019), which results in low prediction accuracy.

One way to overcome the DT problems is by using the RF model, which uses multiple decision trees and acts on the following (Ali et al., 2012): averages predictions over many individual trees, combines the output to reach a single result, and uses bootstrap aggregation, which consists of using bootstrap samples in the individual trees instead of the original data. Bootstrap Aggregation combines the predictions from multiple decision trees to make more accurate predictions than one DT model (T.-H. Lee et al., 2020). Bootstrap aggregating

helps reduce the overfitting caused by the DF model. However, bootstrap aggregating is still not enough. The different decision trees used can have similar structures, resulting in a high correlation in the predictions they will output. To resolve the possible problem of high correlation, RF changes the algorithm for how the sub-trees are learned so that the resulting predictions from all subtrees have less correlation. Figure 6 shows a diagram of the RF model with multiple decision trees.

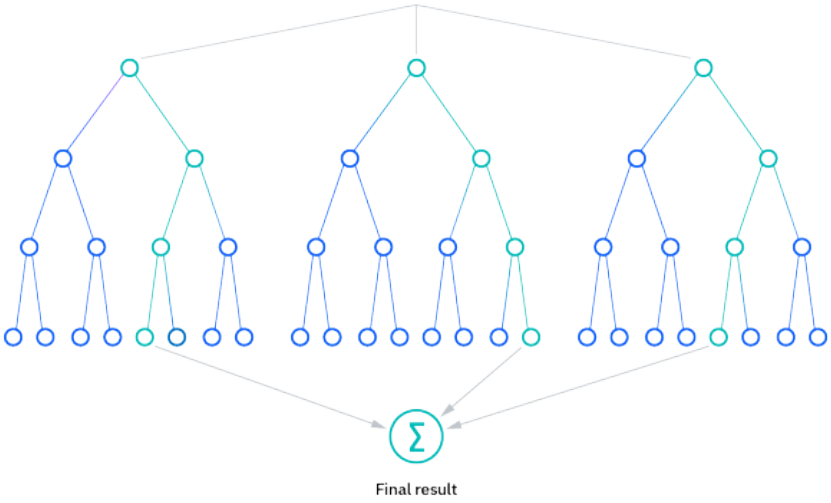

Figure 6: RF Architecture, from (IBM, n.d.-b)

2.1.3 Model Development and Evaluation

In this subsection, we delve deeper into core machine learning model development, offering an overview of fundamental principles related to data splitting, normalization/standardization, and evaluation metrics.

2.1.3.1 Data Splitting

The model analyzes the train set multiple times in the training process to learn its characteristics and adjust for better performance. The test set must consist of unseen data later used to evaluate the model. This is necessary because the model has already learned the characteristics of the training set, making it essential to assess its performance on new, unseen data. Testing the model against such data helps determine if it was trained effectively (Barkved, 2022).

A helpful function for splitting the data into training and test sets is the *train_test_split* function from the *sklearn* library. This function takes parameters such as the input and output data, the desired test size, and the random state and returns a list containing the train-test split of inputs (Bergstra et al.,

2013). The random state parameter controls the function's randomness to the data before splitting it.

2.1.3.2 Normalization/Standardization

An essential step in creating a machine learning model is normalizing/standardizing the data to ensure all features are on a common scale. This process helps streamline training and improve the model's performance and accuracy. The *MinMaxScaler* function provided by the *sklearn* library is helpful for this task. It works by subtracting the minimum value in the feature and then dividing it by the range. The range used by the scaler is the difference between the original maximum and minimum values. After normalization, the data's values fall between 0 and 1. The *MinMaxScaler* preserves the shape of the original distribution while slightly altering the information encapsulated in the original data (Bergstra et al., 2013).

2.1.3.3 Evaluation Metrics

For regression problems, the Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) evaluation metrics are commonly used to assess the model's performance (Chai & Draxler, 2014). The MAE metric calculates the absolute difference between the predicted and actual values for each prediction, followed by determining the mean of these values (Brownlee, n.d.). Similarly, the MSE metric calculates the square of the residual error for each prediction and then determines the average of these values (Brownlee, n.d.). RMSE is the square root of MSE. The formulas for MAE (Equation 1), MSE (Equation 2), and RMSE (Equation 3) are presented below.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - p_i| \tag{1}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2 \tag{2}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - p_i)^2} \tag{3}$$

2.2 Related Work

This subsection discusses some related work in machine learning regarding food recommendation systems, data challenges, and error prediction.

2.2.1 Multilayer Perceptron for nutritional prediction

In (Chen et al., 2020), the authors proposed a novel framework to model the interactions between ingredients and their proportions within recipes to offer healthy recommendations. The framework was divided into 3 parts. The first part predicts the relevant ingredients with user-defined initial ingredients. The second predicts the amounts of the relevant ingredients. The third part creates a healthy pseudo-recipe with a list of ingredients and their amounts according to the nutritional information and recommends the top similar recipes with the pseudo-recipe. In the second part, to learn the relationships between ingredients and their proportions within recipes, the authors used the MLP, which allowed them to predict the ingredients that would result in a healthier recipe. The authors used two recipe datasets, Allrecipes and Yummly, with 125.842 recipes for their research. The authors demonstrated the novel framework's ability to retrieve healthier recipes through experiments.

The authors of (Bikku, 2020) propose the use of a Multilayer perceptron algorithm for health risk prediction. The proposed method was compared against traditional methods (SVM, KNN, and K-means) and state of art models (LSTM and RNN), using the metrics accuracy, sensitivity, precision, and F-score. The authors emphasize the potential of using machine/deep learning for future health risk prediction with a certain probability. The experiments were conducted using 3 standard datasets from the University of California at Irvine (UCI) ML Repository. Each dataset corresponds to a disease. The first dataset is Wisconsin Breast Cancer (WBC), the second is SaHeart (SHt) and the third is Pima Indians Diabetes (PID). To evaluate the performance the authors used a confusion matrix, accuracy, precision, f-score, and recall. The proposed approach outperformed the other methods in accurately classifying and analyzing medical data. With their research, the authors aim to contribute to medical death prevention. The proposed approach offers a promising solution for finding hidden patterns in real historical medical data and improving risk prediction in the medical domain.

In (Ranjeeth & Latchoumi, 2020) the authors discuss the importance of predicting malnutrition in children under the age of five, highlighting the benefits of taking remedial actions based on such predictions. With their research, the authors aim to contribute to previous studies that have focused on analyzing and finding correlations related to malnutrition, creating a malnutrition predictive model. The authors have collected data from parents in Repalle town, Andhra Pradesh, India, who have children under the age of five. The dataset used consists of 2956 records with 12 parameters and 3 class labels. The purpose

approach consists of a Multilayer perceptron classifier with a stochastic gradient descent optimization technique for classifying the data effectively. Accuracy, recall, and precision were used to evaluate the model performance. After experiments, the authors conclude that the purpose approach performs well. Aiming to increase the model performance the authors used a filter-based method for feature selection. The proposed solution achieved good accuracy, recall, and precision in testing and validation results when compared against existing classifiers. Classifying 41 out of 44 new kids' data correctly, with an accuracy of 93.6%.

2.2.2 Deep learning food recommendation system

In (Rostami et al., 2022), the authors present a food recommendation system based on user preferences and food ingredients. For food content-based recommendations, the authors explore graph clustering, food deep embedding, food similarity calculation, food clustering, and food-based rating prediction. They used deep-learning techniques and user similarity calculation, generation of the trusted network, graph representation of users, user clustering, and user-based rating prediction for a user-based recommendation. In the pre-processing phase, the authors realized ingredient formalization and pre-processing of the input foods, which included tokenization, stemming, and stop-word removal. A default stop word list was consulted and reshaped to remove noisy data, using Porter's stemming method (M. C. Lee, 2011). The solution was evaluated using a dataset created by crawling the Allrecipes.com website. The results have outperformed other state-of-the-art approaches.

In (Gao et al., 2022), the authors propose a food recommendation model with a graph convolutional network (FGCN). The work focused on food-related relations: ingredient-ingredient, ingredient-recipe, and recipe-user. Previous works overlooked all of those relations. The authors developed three key components: the embedding layer, information propagation, and prediction layers. The embedding layer explores the collaborative signals between users, recipes, and ingredients. The information propagation is divided into ingredient-ingredient, ingredient-recipe, and recipe-user. Ingredient-Ingredient considers that users will consume ingredients with similar attributes. Ingredient-Recipe considers the list of ingredients of each recipe. Recipe-User considers the users-recipe collaborative signals. To test the developed solution, the authors used a real-world dataset collected from Allrecipes.com. The dataset used contains a user-recipe bipartite graph and a recipe-ingredient bipartite graph. The proposed solution was compared against four state-of-the-art works. After extensive tests, the authors conclude that the presented solution outperforms all other methods on both Recall and NDCG (Normalized Discounted Cumulative Gain) metrics.

2.2.3 Multi-criteria food recommendation system

The authors of (Toledo et al., 2019) propose a framework for daily meal plan recommendations. The proposed framework contains nutritional-aware and preference-aware information. It's pointed out that although multiple types of research focus on food intake recommendations, the majority do not manage user preferences and nutritional information simultaneously. The proposed solution maximizes the global user's preference over the generated menu. Multi-criteria decisions filter out the foods that do not match the user's preferences and aspects. The authors also include in their proposal an optimization-based stage that generates daily meal plans to meet each user's daily nutritional goals. The data preparation activity builds food profiles and defines the menu templates. To filter out foods that are not nutritionally indicated to recommend, it was used AHPSort (Ishizaka et al., 2012), a multicriteria decision analysis-based. Three phases form the recommendation model. The first phase is frequency-based menu generation. The second phase is the probabilistic-based menu refining. The third phase is restricted frequency-based menu generation. The developed solution was confronted with previous works that use traditional optimization-based menu approaches and with others that use typical recommendation approaches like content-based and collaborative filtering-based recommendations. Against traditional optimization-based menu approaches and considering the average preference value of the recommended menu, the proposed solution presented a higher preference in comparison with other works. It was stated that because typical recommendation approaches consider little information from the nutritional domain, the authors were not able to fairly compare those approaches against the proposed model. The developed framework was also compared against its most direct antecedents, but the authors were once again not able to make direct comparisons because they lack sufficient documentation or because they focused on other recommendation contexts.

2.2.4 Multi-criteria recommendation system

A deep learning multi-criteria recommendation system is presented in (Shambour, 2021). The main goal of the proposed solution is to overcome some of the problems inherited from single-criteria recommendation systems like cold start and weaker accuracy. The authors resorted to the multi-criteria Yahoo! Movies MC and TripAdvisor MC datasets to evaluate the solution. The first dataset contains aspects like story, acting, direction, and visuals for each movie. The second dataset contains aspects like value for money, quality of rooms, location of the hotel cleanliness, quality of check-in, overall quality of services, and particular business services to each hotel. Both datasets were prepared to train and test the autoencoder model. The rating scale used in both datasets was from 1 to 5. The authors used statistical accuracy metrics like Mean Absolute Error (MAE) and the Root Mean Square Error (RSME). The proposed solution was implemented in python using Keras 2.1.5 with Tensorflow backend. The proposed recommendation system was put against multi-criteria and single-criteria

collaborative filtering state-of-the-art methods and outperformed all of them.

The authors in (H. I. Lee et al., 2020) propose a multi-period product recommender system. It used an RNN-based recommendation model. To measure if the recommended items are being consumed by the users the authors needed to measure the accuracy. To measure the accuracy the authors decided to use the F1 metric. The F1 metric takes into consideration both precision and recall when measuring accuracy. The authors used the dataset Fresh Food Delivery Service Company in USA from Kaggle. Before the tests, the authors arranged the purchase items by customers based on purchase time. Everything related to the same order number was composed on the same shopping list. They used the shopping information of 7716 customers with 10 shopping carts. It also used 9073 items, they excluded the top and lower 10% of the sale volumes. Because the presented solution used measurements by multiple recommendation periods, data before point T is considered training data, and after that is considered test data. After experiments, the authors conclude that the RNN structure used (LSTM-based) had a better performance than CF based model in a multi-period perspective. The proposed solution showed a 21% higher performance at T point and around 10% higher at T + 4 time. Experiments also showed that having into consideration the purchase order increases the accuracy in multi periods.

2.2.5 Hybrid food recommendation system

In (Kim & Chung, 2020) the authors propose a food recommendation model, using a knowledge-base hybrid decision model and neural networks. The proposed solution tries to solve the problems associated with representative methods like cooperative and content-based filtering. Problems like cold-start, low accuracy, and precision. By combining multiple methods the authors resolve the problems caused by each filtering method. The neural network was used to resolve problems like cold-start and sparsity. To develop the knowledge-based hybrid decision model using a neural network for nutrition management the authors collected health data, configure the transactions for nutrition management, and developed the neural network. To make recommendations the authors collected healthcare and food preference data. The healthcare dataset contains aspects like data, age, gender, BMI, region, and presence of chronic disease. The food preference dataset contains aspects like calories, carbohydrates, protein, fat, and sugars for Koreans' favorite meals. The food preference was collected from a database of food nutritional contents from the Food Safety Information Portal. Once the datasets were collected the authors subdivided the user's preference for food. They also inferred the nutritional information that users will prefer. Through their tests, the authors conclude that the neural network-based prediction model is superior in comparison to both the correlation-based prediction model and the regression analysis-based prediction model.

2.2.6 Addressing Data Challenges

The authors of (Gurupur & Shelleh, 2021) introduce a concept called Machine Learning Analysis for Data Incompleteness (MADI), which uses statistical distributions like Kolmogorov-Smirnov goodness-of-fit, Mielke distribution, and beta distributions for a comprehensive analysis of data incompleteness. The authors compare methods such as stochastic gradient descent, generalized additive models, and support vector machines to provide a complete set of methods and algorithms to predict data incompleteness in medical settings. They provide practical suggestions for the application of these methods to predict data incompleteness in EHRs.

The authors of (Sambasivan et al., 2021) conducted interviews with 53 AI practitioners across India, the USA, and East and West African countries, defining and presenting empirical evidence on compounding events that cause negative downstream effects from data issues. The paper reveals that such events are pervasive, invisible, delayed, but often avoidable. It discusses opportunities in human-computer interaction (HCI) for designing and incentivizing data excellence, leading to safer and more robust AI systems. Key findings include that data cascades are common (92% prevalence), and often result from conventional AI practices applied in high-stakes domains such as health, conservation, and loan allocations. These cascades have significant negative impacts, including technical debt, project failures, and harm to communities, but are largely preventable with better data practices and incentives for data quality.

2.2.7 Error Prediction

In (Kyerematen-Boateng et al., 2023) the authors address the need for users to trust the predictions made by machine learning models, with that statement in mind, in their research they propose a method to compute the confidence score of the model's predictions based on the distribution statistics of the latent space features. The purpose method consists of mapping the latent space feature values to a hypothesized prior distribution and storing the parameters of the prior distributions for each node. The confidence associated with a prediction is determined by how well the model's penultimate layer activations match the stored distribution parameters for the predicted label. The paper demonstrates the effectiveness of utilizing different statistical properties of the latent space activations in the confidence scoring algorithm and compares the results against Softmax probabilities-based confidence scores on the CIFAR-10 and MNIST datasets. The results show that the confidence score computation based on latent space activation statistics outperforms the Softmax-based approach, indicating the potential of using latent space features for improving trust in machine learning predictions. Through experiments using the CIFAR-10 and MNIST datasets the authors successfully demonstrated the capacity of the proposed method to accurately compute the output confidence of a model. The authors concluded that by mapping the latent space feature values to a

hypothesized prior distribution and comparing the model's penultimate layer activations with the stored distribution parameters, the level of confidence associated with a prediction can be determined.

The authors of (Iwendi et al., 2020) propose a deep learning solution for a health-based medical dataset that automatically detects which food should be given to which patient based on disease and other features like age, gender, weight, calories, protein, fat, sodium, fiber, cholesterol. The proposed research framework implements machine/deep learning algorithms like logistic regression, naive Bayes, RNN, MLP, GRU, and LSTM. The authors used a medical dataset with 13 features of different diseases and 1000 products, with 8 features in the product section for 30 patients. The algorithms' performance was evaluated using the metrics forecasting accuracy, recall, precision, and F1 measures. LSTM outperformed the other algorithms, achieving 97.74% on accuracy, 98% on precision, 99% on recall, 89% on precision, 73% recall, and 80% F1-measure for the not-allowed class, and 99% for the allowed class.

In (Esser-Skala & Fortelny, 2023) the authors discuss the limited interpretability of deep neural networks and the existing potential to improve it. The authors stated that biology-inspired deep learning models offer better interpretability by encoding real-world concepts in hidden nodes. In their previous work, the authors demonstrated the lack of robustness and biases in node-level interpretations in models trained on single-cell transcriptomes. In this research, the authors continued the methodology of P-NET, a biology-inspired model trained on patient mutation data, and observed the variability of interpretations and susceptibility to knowledge biases. The research was conducted using data from single-cell transcriptomes, to train models for biology-inspired deep learning, and patient mutation data to train the P-NET model. The authors through their research have identified network properties that drive interpretation biases and presented an approach to control the robustness and biases of interpretations, leading to more specific interpretations. In the findings, the authors highlight the importance of methods to ensure robust and bias-aware interpretability in biology-inspired deep learning.

The authors of (Ahmad et al., 2023) propose a novel method grounded in causal analysis to interpret the inner function of neural networks and capture cause-effect mechanisms in pre-trained models. The proposed method relies on path interventions to infer causal mechanisms within hidden layers and isolate relevant information, resulting in task-specific causal explanatory graphs. The authors applied their approach to the LeNet vision models trained on classification tasks. Through quantitative experiments on image classification tasks, the authors conclude that the proposed method can provide more stable and faithful explanations compared to standard attribution-based methods. The authors have analyzed the effects that individual neurons have on model prediction by intervening in their connections. The experiments were conducted using MNIST data. The proposed approach has demonstrated the effectiveness of the method

in generating consistent and stable explanations. The explanations generated from the causal graphs show lower variance to perturbations compared to other attribution methods, indicating their consistent stability. With their research, the authors highlight the little research on DNN explainability by manipulating weights and presenting a hypothesis-testing formula to identify the most influential nodes in the model.

3 Methodology

This chapter presents the methodology applied to this research, aimed at describing the process of data gathering, preprocessing, algorithm selection, evaluation metrics, hyperparameter tuning, and training and testing.

Figure 7 presents a diagram with all the different phases, in our methodology approach. It is structured into two parts, each representing one model. The first part addresses the prediction model, where we collect and preprocess the data, select the machine learning algorithm, choose the evaluation metrics, tune the model, and train and test it. The second part addresses the error model, where we create a custom dataset for the error model, preprocess the data, select the machine learning algorithm, choose the evaluation metrics, tune the model, and train and test it.

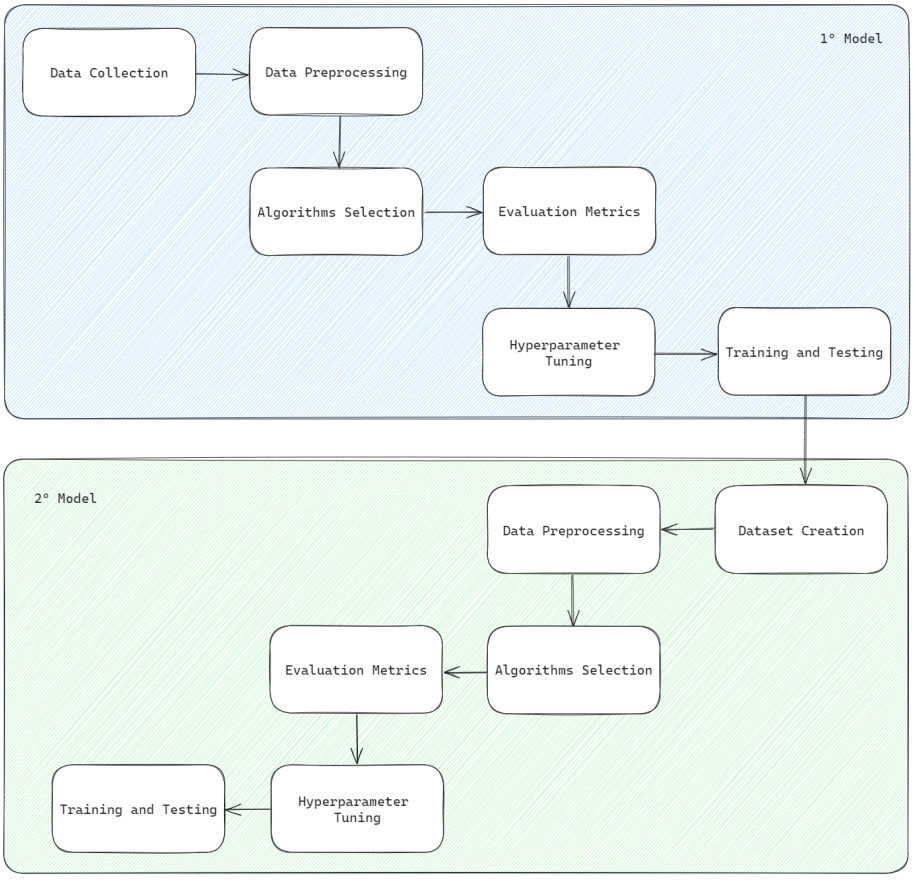

Figure 7: Methodology Approach

3.1 Data Collection

The data collection phase addresses the identification of data types for the problem we are trying to solve. Knowing the data type we search for relevant data sources. Our research focuses on looking into large, publicly available datasets in a macronutrient context.

3.2 Data Preprocessing

This phase is a crucial step where we transform raw data into clean and structured data, in a format suitable for analysis and modeling. It encompasses various sub-tasks, such as data cleaning, data splitting, and normalization/standardization. Once we have the dataset, we proceed to prepare the data. This involves transforming the raw data into a format that the model can effectively understand and use. The data preprocessing process is divided into smaller phases, including data cleaning, data splitting, and normalization/standardization.

3.2.1 Data Cleaning

The data cleaning activity involves manipulating metadata and removing rows containing empty or NULL values. Initially, we renamed the most important columns into a more recognizable and user-friendly format. To handle missing values, we first identify all columns containing NULL values. Then, for each column with NULL values, we remove the corresponding rows.

3.2.2 Data Splitting

In this phase, the dataset is split into two smaller sets: the training set and the test set. The training set, also known as the learning set, contains the data on which the model will learn. The test set contains the data against which the model's predictions will be evaluated. For the prediction model, we utilize a 60/40 split for training and testing, respectively. Conversely, for the error model, we employ a 70/30 split for training and testing. Listing 1 provides a code example for data splitting.

```
input = df.drop(['Calories'], axis=1)
output = df[['Calories']]

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(
input, output, test_size=0.4, random_state=42)
```

Listing 1: Code example for data splitting

3.2.3 Normalization/Standardization

To ensure that all features are on a common scale, streamline the training process, and improve the model's performance and accuracy, the input data is normalized. We utilize the MinMaxScaler **<empty citation>** for data normalization/standardization. Listing 2 provides a code example for performing data normalization/standardization.

```
X_scaler = MinMaxScaler()
X_train = X_scaler.fit_transform(X_train)
X_test = X_scaler.transform(X_test)
```

Listing 2: Code example 1 for Normalization/Standardization

3.3 Machine Learning Algorithms Selection

Choosing the most appropriate machine learning algorithms is a crucial step that can significantly impact the success or failure of the research. To make informed decisions about which algorithms to employ, we carefully consider the research goals and desired outcomes. We analyze the data and determine the problem type, whether it involves classification or regression tasks. Based on their suitability for the problem and the characteristics of the data, we then select the most appropriate algorithms.

Our research employs two models: one for predicting *Calories* in a macronutrient context and another for predicting the error associated with the predictions made by the first one. For the prediction of *Calories*, we utilize an ANN to forecast a continuous value based on numerical input features such as *Carbs, Fat, and Protein*. This scenario poses a regression problem. While it is common to assume linear relationships among features, potential hidden complexities prompt us to explore non-linear models. We examined various non-linear ANN models, including LSTM and MLP. MLP provides some interpretability through feature importance analysis, such as SHAP, making it suitable for large datasets with potentially complex data structures.

For error prediction, we opt for RF due to its simplicity, non-linearity, and robustness. RF excels in capturing complex relationships between features, offering a beneficial solution to our specific problem. Furthermore, RF exhibits a lower risk of overfitting and demonstrates versatility in handling diverse data characteristics. These factors solidify RF as a promising choice for error prediction in our research.

3.4 Evaluation Metrics

To evaluate model performance, we utilize evaluation metrics selected based on the model and problem type, whether it involves classification or regression.

The features in the *MyFitnessPal* dataset (*Calories, Carbs, Fat, Protein*) are continuous variables, also known as quantitative variables, indicating they can assume an infinite range of values. The objective of the prediction model, utilizing *Carbs*, *Fat*, and *Protein*, is to predict *Calories* based on the levels of the other features. Given that both the input and output variables are continuous, this constitutes a regression problem. Therefore, evaluation metrics commonly used in classification problems, such as precision, recall, and F1-score, are not applicable.

The performance of the models used in the research will be evaluated using the metrics MAE (Mean Absolute Error), MSE (Mean Squared Error), and RMSE (Root Mean Square Error).

3.5 Hyperparameter Tuning

In this phase of the process, we optimize the performance of machine learning models by adjusting their hyperparameters. Hyperparameters are pre-defined settings that influence the learning process and can vary depending on the model. For some models, these may include the number of neurons, the number of hidden layers, etc., while for others, it may involve the number of decision trees to use. To identify the best set of hyperparameters for each model, we experiment with various combinations and select the one that yields the best results based on evaluation metrics.

3.5.1 MLP Tuning

The hyperparameters used in MLP include the learning rate, number of hidden layers, number of neurons in hidden layers, batch size, and number of epochs. When no learning rate is defined, TensorFlow (the library used) sets it to the default value for the Adam optimizer, which is 0.001. The model will be trained with this default learning rate, demonstrating satisfactory convergence and stability. Due to the dataset's dimension and possible non-linear complex relationships, we opted for four hidden layers. This choice aims to provide the model with sufficient capacity to learn these complex relationships without sacrificing too much performance. Four hidden layers were found to be the optimal configuration. Regarding the number of neurons in each layer, we experimented with various configurations. The configuration that struck the best overall balance between the model complexity and performance consisted of 256 neurons in the input layer, and 128, 64, 32, and 16 neurons in the first, second, third, and fourth hidden layers, respectively. The batch size affects convergence speed and computational resources. After training the model with batch sizes of 32, 64, and 128, we found that a batch of 64 offered the best performance-to-resource ratio. Finally, we employed 100 epochs for training.

3.5.2 RF Tuning

The hyperparameters of the RF consist of the number of decision trees to use and the random state. After multiple runs using 180, 185, 190, 195, and 200 decision trees, the value that provided the best result was 190 decision trees (n_estimators). The random state was set to 42 to ensure consistent results.

3.6 Training and Testing

The performance of the models is evaluated through a dedicated training and testing process. Both models are tested using their best hyperparameter configurations achieved through the hyperparameter tuning process. The training and testing process involves varying dataset sizes for both models to assess their sensitivity to data volume. After training, each model is evaluated against its designated testing set, which remains separate during the training process.
The detailed procedure is presented in the Implementation section with all the results achieved presented in the Experimental Results section.

4 Experimental Results and Discussion

This chapter presents the experimental results and discusses the outcomes of models' hyperparameter tuning, along with sensitive analysis and the model's predictions. The results provided here offer insights into the effectiveness and performance of the models utilized in the developed solution. We aim to address the research questions and contribute to the study area through quantitative metrics, and visual representations.

4.1 Data Interpretation

To train and test the prediction model, we utilized the public dataset *MyFitness-Pal* from Kaggle (ZVIKINOZA, n.d.). This dataset meets our requirements, it is large and contains data in a macronutrient context. Table 1 provides detailed information on the dataset columns.

Features	Features Description
User ID	Anonymized user ID
Date	Diary date
Sodium_Total	Daily aggregate of Sodium intake
Sugar_Total	Daily aggregate of Sugar intake
Fiber_Total	Daily aggregate of Fiber intake
Potass._Total	Daily aggregate of Potassium intake
Iron_Total	Daily aggregate of Iron intake
Calcium_Total	Daily aggregate of Calcium intake
Sat Fat_Total	Daily aggregate of Sat Fat intake
Chol_Total	Daily aggregate of Cholesterol intake
Vit A_Total	Daily aggregate of Vitamin A intake
Trn Fat_Total	Daily aggregate of Trn intake
Mon Fat_Total	Daily aggregate of Mon intake
Ply Fat_Total	Daily aggregate of Ply intake
Calories_Goal	Daily aggregate of Calories goal
Carbs_Goal	Daily aggregate of Carbs goal
Fat_Goal	Daily aggregate of Fat goal
Protein_Goal	Daily aggregate of Protein goal
Sodium_Goal	Daily aggregate of Sodium goal
Sugar_Goal	Daily aggregate of Sugar goal
Fiber_Goal	Daily aggregate of Fiber goal
Potass._Goal	Daily aggregate of Potassium goal
Iron_Goal	Daily aggregate of Iron goal
Calcium_Goal	Daily aggregate of Calcium goal
Sat Fat_Goal	Daily aggregate of Sat Fat goal
Chol_Goal	Daily aggregate of Cholesterol goal
Vit A_Goal	Daily aggregate of Vitamin A goal
Vit C_Goal	Daily aggregate of Vitamin C goal
Trn Fat_Goal	Daily aggregate of Trn goal
Mon Fat_Goal	Daily aggregate of Mon goal
Ply Fat_Goal	Daily aggregate of Ply goal

Table 1: Dataset Features

4.2 Hyperparameter Tuning - MLP

The primary objective of hyperparameter tuning is to utilize evaluation metrics and adjust the hyperparameters to minimize them. A dataset size of 10K was employed before splitting. To tune the model, various combinations of hyperparameters were tested, with each combination evaluated three times. The average value of each performance metric was rounded to two decimals and presented alongside the corresponding standard deviation. Tables 2, 3, and 4 depict the results obtained with 128, 256, and 512 neurons, respectively, and batch sizes of 32, 64, and 128. Lower values of MAE, MSE, and RMSE indicate better model

performance. When using a batch size of 32, the optimal number of neurons was found to be 128 neurons. For a batch size of 64, the best performing configuration was 256 neurons. Conversely, with a batch size of 128, the most effective combination was with 512 neurons. The overall best-performing combination was identified as 256 neurons with a batch size of 64, exhibiting the lowest MAE and consistent RMSE values. In summary, the model achieved relatively low MSE across all configurations.

	MAE	MSE	RMSE
128 Neurons	132,58 ± 1,11	87757,08 ± 1340,15	296,23 ± 2,26
256 Neurons	137,34 ± 2,79	85116,67 ± 1752,59	291,74 ± 3,00
512 Neurons	145,89 ± 7,24	86974,98 ± 5630,00	294,81 ± 9,62

Table 2: Hyperparameter Tuning - Batch Size 32

	MAE	MSE	RMSE
128 Neurons	139,88 ± 1,56	88068,35 ± 454,91	296,76 ± 0,77
256 Neurons	129,27 ± 0,08	88312,40 ± 415,35	297,17 ± 0,70
512 Neurons	130,25 ± 0,64	90814,68 ± 2903,85	301,33 ± 4,80

Table 3: Hyperparameter Tuning - Batch Size 64

	MAE	MSE	RMSE
128 Neurons	157,12 ± 5,90	91694,62 ± 1736,31	302,80 ± 2,86
256 Neurons	150,01 ± 2,71	88109,66 ± 1013,50	296,83 ± 1,70
512 Neurons	133,63 ± 0,40	35119,15 ± 45378,17	297,71 ± 4,05

Table 4: Hyperparameter Tuning - Batch Size 128

4.3 Hyperparameter Tuning - RF

To optimize the RF model, adjustments were made to the number of decision trees utilized. The dataset employed by the RF model was derived from the test set utilized by the MLP model. Specifically, the MLP model utilized 10K records with a 60%-40% split, resulting in a dataset size of 4K for RF before splitting. The RF was evaluated under five conditions, varying the number of decision trees: 175, 180, 185, 190, and 195. Each condition was tested three times, and the average results rounded to two decimals, along with their standard deviation, are presented in Table 5. Lower values of MAE, MSE, and RMSE indicate better model performance with the given number of decision trees. The model demonstrated superior performance when employing 190 decision trees, exhibiting lower values across all metrics, indicating both strong predictive performance and consistency.

	MAE	MSE	RMSE
175 Trees	145,13 ± 3,38	76558,14 ± 8909,15	276,39 ± 15,90
180 Trees	143,00 ± 2,41	70033,19 ± 2609,85	264,61 ± 4,95
185 Trees	146,93 ± 3,41	80366,74 ± 4091,36	283,43 ± 7,27
190 Trees	141,25 ± 0,92	67518,53 ± 2010,27	259,82 ± 3,88
195 Trees	144,75 ± 2,58	73791,68 ± 2177,74	271,63 ± 4,02

Table 5: Hyperparameter Tuning - RF

4.4 Dataset Size Sensitivity Analysis

Tables 6 and 7 present the results of the data size-sensitive analysis, with each table displaying the average results rounded to two decimals alongside the standard deviation for each evaluation metric and condition. The models underwent testing under four different conditions, with each condition being run three times. Lower values MAE, MSE, and RMSE suggest better fitting of the models to the dataset.

The dataset utilized by the RF model is highly dependent on the original dataset employed by MLP, constituting 40% of the MLP dataset. Specifically, for original dataset sizes of 10k, 15k, and 20k records in MLP, RF utilized 4k, 6k, and 8k records, respectively. Notably, when the original dataset consisted of 10k records, MLP exhibited the best performance, albeit with higher MSE and RMSE values but lower standard deviation values, indicating greater consistency and stability. However, this resulted in a smaller dataset for RF (4k records), which might benefit from a larger dataset.

Increasing the original dataset from 10k to 15k records led to a slight drop in MLP performance, characterized by higher MAE but lower MSE and RMSE values, albeit with higher standard deviations across all metrics, suggesting decreased consistency. Nonetheless, this trade-off resulted in RF presenting lower standard deviations across all metrics, with lower MAE, but higher MSE and RMSE values, indicating more consistent performance from the RF model. Further increasing the dataset from 15k to 20k records provided the largest dataset size for RF to utilize (8k), yet significantly impacted the MLP performance. Consequently, having the largest dataset for RF did not yield the expected improvements, with the lowest MAE value and its respective standard deviation. However, MSE and RMSE decreased, accompanied by increased standard deviations, signifying a loss of consistency.

Considering all factors, the optimal dataset size appears to be 15k, striking the best trade-off from MLP to RF while maintaining some level of consistency and stability in the predictions.

	MAE	MSE	RMSE
10k	129,25 ± 0,59	88078,33 ± 1409,00	296,78 ± 2,38
15k	132,55 ± 1,55	81184,51 ± 2982,44	284,89 ± 5,24
20k	135,88 ± 6,12	147980,93 ± 9278,42	384,56 ± 12,09

Table 6: Sensitive Analysis - MLP

	MAE	MSE	RMSE
4k	145,48 ± 3,73	76778,42 ± 8147,82	268,56 ± 155,07
6k	139,12 ± 3,03	107594,58 ± 321,90	328,02 ± 0,49
8k	135,27 ± 0,76	91349,86 ± 1109,09	302,24 ± 1,84

Table 7: Sensitive Analysis - RF

4.5 MLP - Predictions

Through the experimental process, we determined that MLP achieved optimal performance with 256 neurons in the input layer, a batch size of 64, and a dataset size with 15K records. Figure 8 illustrates the model's performance with a scatter plot, where the diagonal line represents the reference for true values (green dots). Generally, the model's predictions (blue dots) closely align with the true values, although some outliers indicate challenges in predicting Calories, particularly for lower values.

Figure 9 provides another visual representation of the model's performance using matplotlib. The blue line represents the true values, while the orange line depicts the predicted values. Overall, the predicted and true values overlap, indicating generally accurate predictions. However, two significant spikes are observed: one around index 3000 for the true values, exceeding 20,000 calories, and another near index 5000 for the predicted values, albeit lower than the first spike. These spikes suggest instances where the model's predictions significantly deviate from the true values. Despite these deviations, the model performs reasonably well for the majority of the data, producing predictions within an acceptable margin of error.

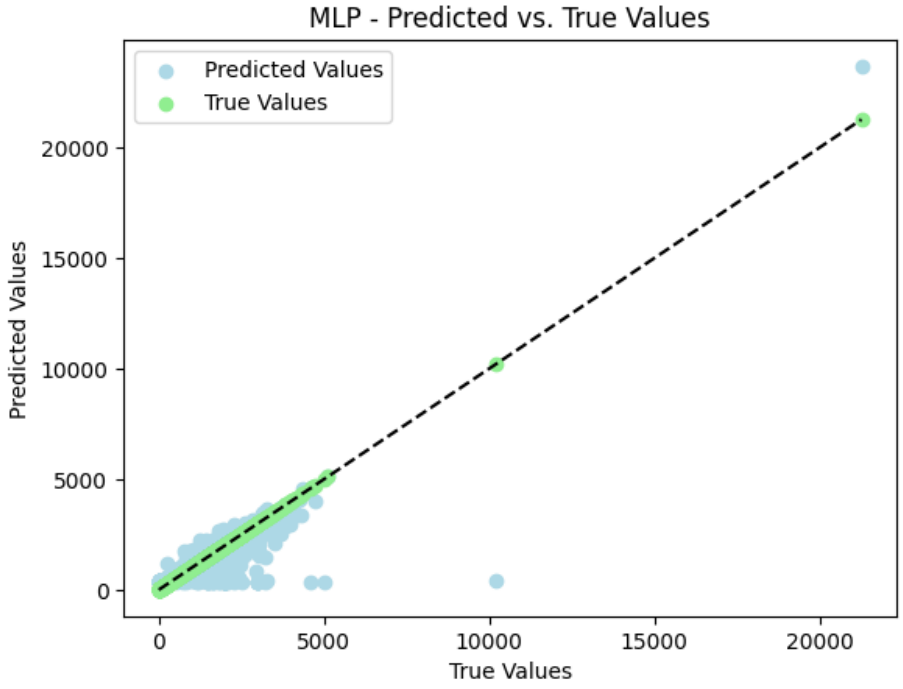

Figure 8: MLP Predictions 1

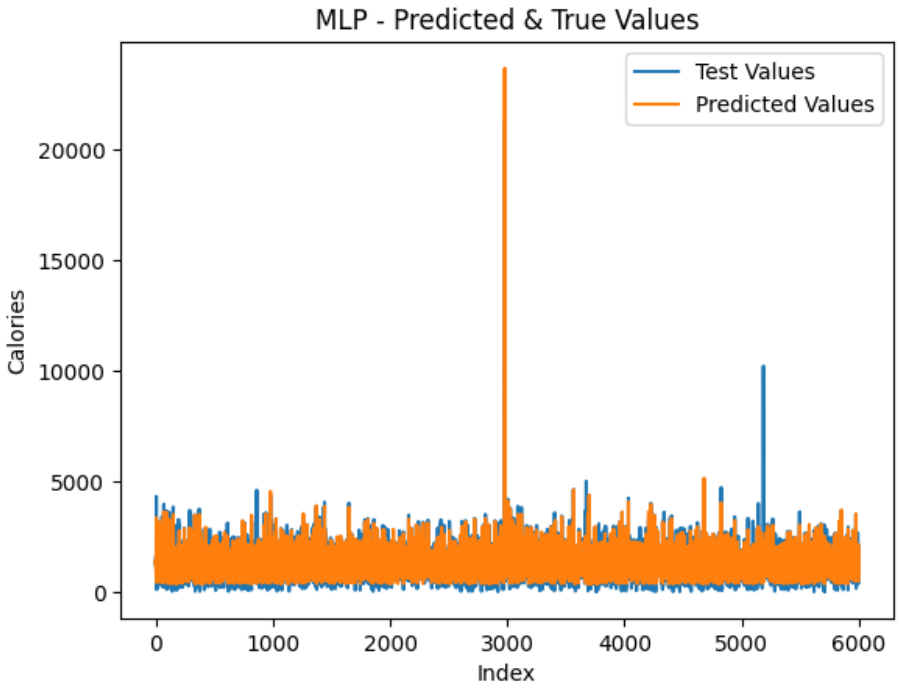

Figure 9: MLP Predictions 2

4.6 MLP - Feature Importance

To aid in interpreting the behavior of the MLP and the influence of each feature on its predictions, SHAP values were utilized. Figure 10 illustrates the contribution of each feature (*Carbs, Fat, Protein*) to the test set used for MLP evaluation.

The *Carbs* feature consistently exhibits a positive impact on the model's output, indicating that higher *Carbs* values generally lead to increase model output. This suggests a positive correlation between the *Carbs* feature and *Calories*. *Carbs* appears to be the most consistent feature, exerting slightly more influence than the others.

Fat, on the other hand, presents a more complex relationship with the model. It yields a mix of positive and negative SHAP values, suggesting that its impact on the model output can vary depending on its quantity or type. This complexity implies that the correlation between the *Fat* feature and *Calories* is intricate, likely influenced by interactions with other features or specific thresholds.

Similarly, *Protein* exhibits a mix of positive and negative SHAP values, albeit

with a stronger positive influence overall. This indicates that higher protein values tend to increase the model output, reflecting a positive correlation between the *Protein* feature and *Calories*.

Figure 11 displays the feature importance of the MLP train set. *Carbs* SHAP values span both positive and negative ranges, with a notable skew towards positive values. This suggests that while higher *Carbs* values generally result in increased predicted *Calories*, there are instances where higher *Carbs* values lead to decreased model output. This suggests a nuanced relationship between *Carbs* and *Calories*, possibly influenced by interactions with other features or specific thresholds.

Fat SHAP values are evenly distributed around zero, indicating that its impact on predicted *Calories* can be both positive and negative. This suggests a complex relationship between *Fat* and *Calories*, likely influenced by interactions with other features.

Protein SHAP values are predominantly negative, indicating that higher *Protein* values tend to decrease predicted *Calories* values. However, there are also instances where higher *Protein* values lead to increased predicted *Calories*, mirroring the results observed for the *Carbs* feature. Again, this complexity may stem from interactions with other features or specific thresholds.

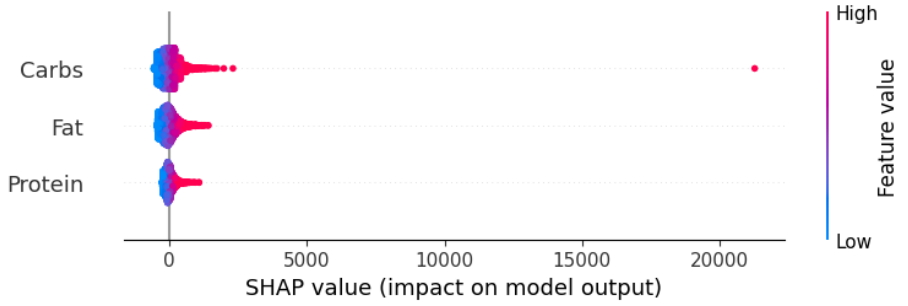

Figure 10: Shap Values Test Set

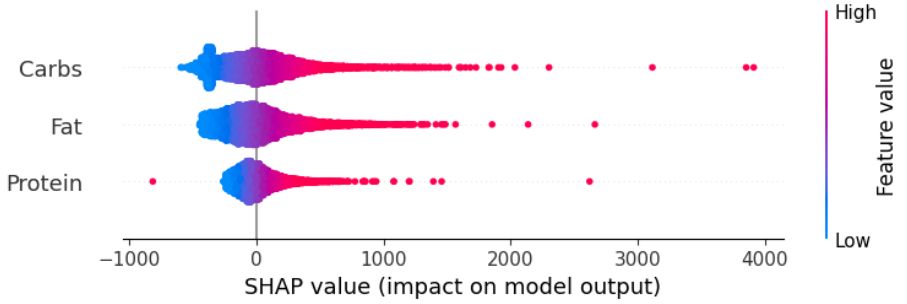

Figure 11: Shap Values Train Set

4.7 Dataset Creation - 2º Model

Given the primary objective of the research, which is to predict the errors in MLP's predictions, there arises a need to construct a second dataset. This dataset will encompass both the model's prediction errors and the associated feature importance. By adopting this approach, we aim to enable the RF algorithm to address the limitations of the MLP model by systematically analyzing prediction errors and feature importance.

This concept involves calculating the difference between the 'X_test' values and the values from the 'closest_row' for each input feature (*Carbs*, *Fat*, *Protein*) in the first dataset. Subsequently, we retrieve the corresponding feature importance from 'X_test' and 'X_train', alongside the MLP prediction errors. The resulting dataset includes the features outlined in Table 8.

Features	**Features Description**
Difference Carbs	Carbs Feature Difference
Shap_Values Test Carbs	Carbs Feature Shap_Values Test
Shap_Values Train Carbs	Carbs Feature Shap_Values Train
Difference Fat	Fat Feature Difference
Shap_Values Test Fat	Fat Feature Shap_Values Test
Shap_Values Train Fat	Fat Feature Shap_Values Train
Difference Protein	Protein Feature Difference
Shap_Values Test Protein	Protein Feature Shap_Values Test
Shap_Values Train Protein	Protein Feature Shap_Values Train
Error	MLP Prediction Error

Table 8: Error Dataset

To construct the columns *Difference Carbs*, *Difference Fats*, and *Difference Protein* we employ the Euclidean distance calculation based on the input values of the test set. Specifically, the 'weighted_euclidean' function computes the Euclidean distance between each vector in the test set and its corresponding vector

in the train set, utilizing a predefined set of weights. The implementation of this function is provided in Listing 3.

```
def weighted_euclidean(a, b, weights):
    # Ensure all inputs are numpy arrays
    a, b, weights = np.array(a), np.array(b), np.array(
weights)

    # Flatten the arrays
    a, b, weights = a.flatten(), b.flatten(), weights.
flatten()

    return np.sqrt(np.sum(weights * (a - b)**2))
```

Listing 3: Code for Euclidean distance

Listing 4 presents the code utilized to determine the closest rows and their corresponding indices in the 'X_train' set for each row in the 'X_test' set. The list 'closest_indices' stores, for each row in 'X_test', the index of its closest row in 'X_train'. Similarly, the list 'closest_rows' stores, for each row in 'X_test', the closest row itself from 'X_train'. The process involves iterating through each row in 'X_test' using a for loop. For each test row, another for loop iterates through each row in 'X_train'. The Euclidean distance between each pair of test and train rows is calculated using the 'weighted_euclidean' function. If the calculated distance is smaller than the current 'best_distance', the 'best_distance' is updated, and the 'best_rows' variable receives the current row from the train set. Once the closest train row to the current test row is found, the row is appended to the 'closest_rows' list, and its index is appended to the 'closest_indices' list. In summary, the 'closest_rows' and 'closest_indices' lists will contain all the closest rows and their respective indexes from 'X_train' for each row in 'X_test'.

```
closest_rows = []
closest_indices = []

for i in range(len(X_test)):
    test_row = X_test[i]
    best_distance = np.inf
    best_index = None
    best_row = None
    for j in range(len(X_train)):
        train_row = X_train[j]
        distance = weighted_euclidean(test_row,
train_row, shap_values2[j])
        if distance < best_distance:
            best_distance = distance
            best_index = j
            best_row = train_row
    closest_rows.append(best_row)
```

```
        closest_indices.append(best_index)

    closest_array = np.array(closest_rows)
    closest_indices = np.array(closest_indices))
```

Listing 4: Code for closest rows and indices

Once the 'closest_row' is identified, the subsequent step involves calculating the difference between the values of 'X_test' and the values from the 'closest_row'. The code for this process is presented in Listing 5.

```
    # Calculate the difference
    difference_array = X_test - closest_array

    difference_column_names = ["Carbs", "Fat", "Protein"]
    difference_df = pd.DataFrame(difference_array, columns=
    difference_column_names)

    difference_df
```

Listing 5: Code to calculate the difference

To construct the columns *Shap_Values Test Carbs*, *Shap_Values Test Fat*, and *Shap_Values Test Protein*, we utilize the *shap_values_feature* function to extract the SHAP values for a specific feature. The code for this process is presented in Listing 6.

```
    def shap_values_feature(column_index):
        # Select the corresponding column in shap_t
        test_shap_values = shap_t[:, column_index]

        # Print the SHAP values for the desired
    characteristic
        return test_shap_values
```

Listing 6: Code to get SHAP values

The features *Carbs*, *Fat*, and *Protein* correspond to column indices 0, 1, and 2 respectively. To obtain their respective SHAP values for each feature using the *shap_values_feature* function, we utilize the code provided in Listing 7.

```
    test_shap_values_1 = shap_values_feature(0)
    test_shap_values_2 = shap_values_feature(1)
    test_shap_values_3 = shap_values_feature(2)
```

Listing 7: Code to get test SHAP values

The prediction error of the first model is calculated by subtracting *y_pred* from *y_test*, where *y_test* represents the true values of the target variable (*Calories*) in the test set and *y_pred* represents the predictions made by the first model (MLP) for the target variable (*Calories*) in the test set.

```
prediction_error = y_test - y_pred
df_error = pd.DataFrame(prediction_error, columns=['
error'])
error = df_error.to_numpy().ravel()
```

Listing 8: Code to create the Error column

With all the necessary data available, the final step is to construct the dataset. The code for this process is presented in Listing 9.

```
df2 = pd.DataFrame({'Difference Carbs': first_array,
                    'Shap_Values Test Carbs':
test_shap_values_1,
                    'Shap_Values Train Carbs':
carbs_column_shap_values_train,
                    'Difference Fat': second_array,
                    'Shap_Values Test Fat':
test_shap_values_2,
                    'Shap_Values Train Fat':
fat_column_shap_values_train,
                    'Difference Protein': third_array,
                    'Shap_Values Test Protein':
test_shap_values_3,
                    'Shap_Values Train Protein':
protein_column_shap_values_train,
                    'Error': error})
```

Listing 9: Code to build the second dataset

4.8 RF - Predictions

In the hyperparameter tuning process, we determined that RF performed best with 190 decision trees. Figure 12 provides a visual representation of the model's performance, showing that the majority of predicted values closely align with the true values. This suggests that the model generally performs well. However, there are instances where it fails to accurately predict values, which could be attributed to outliers in the data or certain limitations within the model itself.

Figure 13 presents another visualization of the model's performance using matplotlib. In this representation, the predicted and true values do not overlap, indicating poor predictive accuracy. Overall, the predictions tend to be low,

with a notable spike in the Predicted Values Error around index 1000. This spike may indicate an outlier or a specific instance where the model's prediction significantly deviates from the true value.

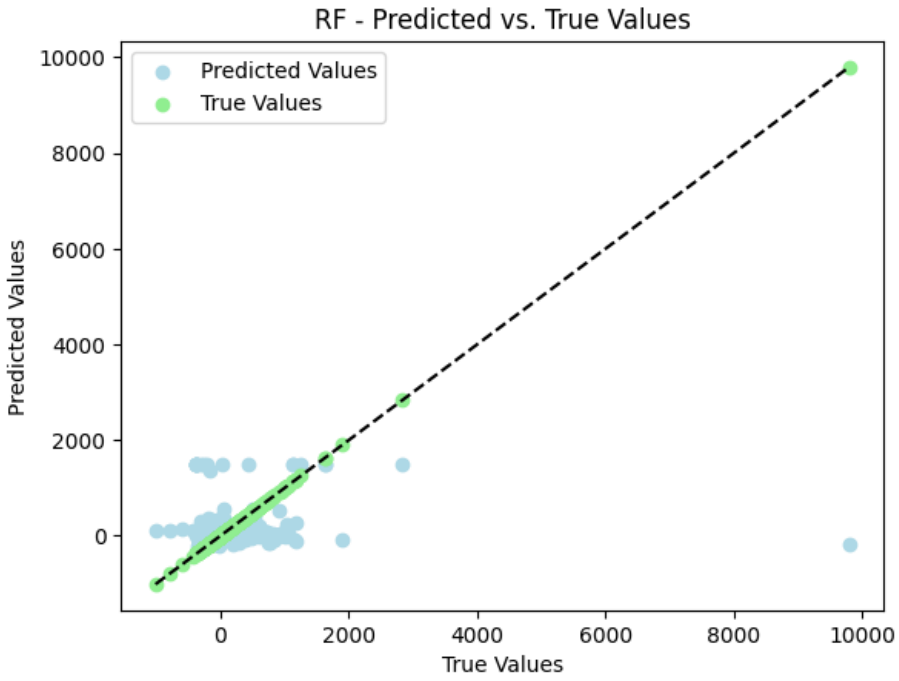

Figure 12: RF Predictions 1

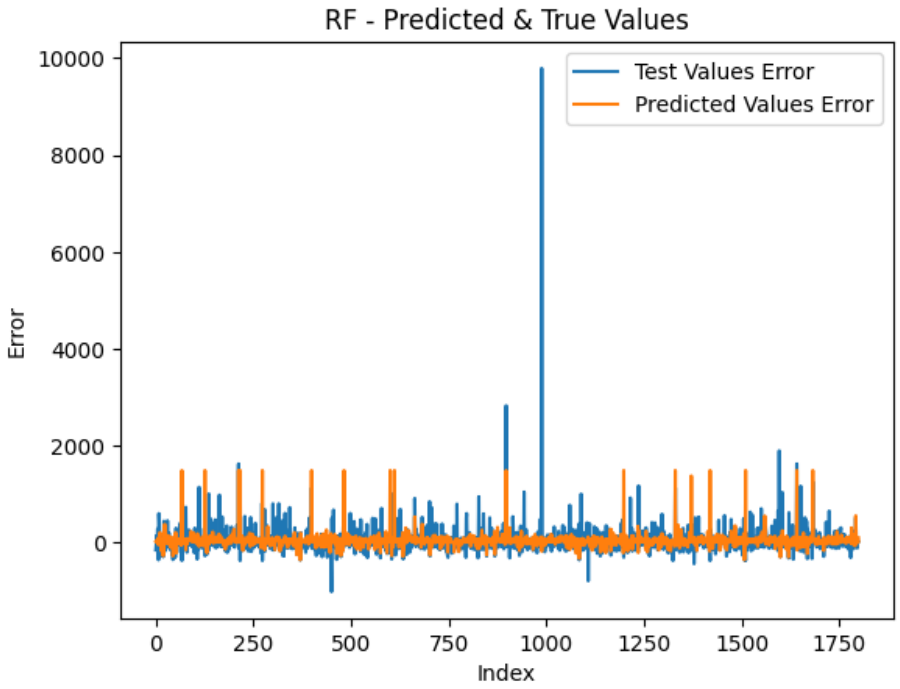

Figure 13: RF Predictions 2

Table 9 presents the evaluation metrics obtained for both models, which were tuned before execution. MLP predictions are generally accurate, with an MAE of 133.70, an MSE of 83670.07, and an RMSE of 289.26. However, as shown by the feature importance analysis, the model struggles to predict *Calories* accurately in some cases. RF predictions achieved an MAE of 139.44, an MSE of 110664.20, and an RMSE of 332.66. These values are notably high, indicating that RF does not make accurate predictions of the MLP predictions' error.

MLP			RF		
MAE	MSE	RMSE	MAE	MSE	RMSE
133,70	83670,07	289,26	139,44	110664,20	332,66

Table 9: Evaluation Metrics - MLP and RF

5 Discussion

This section presents a discussion of the results presented in the previous section. The main goals of this research are: (1) to study the relationship between input features and the model output in the nutrition domain using model interpretability methods; (2) to evaluate the performance of a model used to predict the error of the first model.

The research involved understanding neural network predictions, proposing an error prediction model, evaluating its effectiveness, gaining insights into neural network decision-making, and proposing future research recommendations.

The experimental results answer the following research questions.

RQ1. How can a second model predict the error of a primary neural network, and what insights can be gained from its error predictions?

A second model was trained using the prediction errors of the first model, which are weighted by the impact of the input features and the difference between the values used in the test set, and the closest records from the training dataset. By utilizing this training data, the second model gains the ability to predict errors associated with the neural network predictions. This insight reveals systematic patterns in the errors made by the neural network, allowing the second model to learn and predict them. With the second model outcomes, we gain a deeper understanding of the decision-making process of the neural network. The results obtained can support solutions that depend on model interpretability to proportionate user trust. The error model complements the ability of SHAP analysis to interpret models. By modeling the error, it is possible to estimate the confidence of predictions. With an MAE of 139,44, an MSE of 110664,20, and an RMSE of 332,66, the predictions of the second model as shown in Figures 12 and 13 have a significant discrepancy between the predicted and true values at around the index of 1000. This indicates an anomaly or outlier at that point in the data. It might be worth investigating the data around this index further to understand the cause of this spike in error.

RQ2. Which data structure is required to build an error prediction system?

The quality of the dataset in which the error prediction system is trained is crucial to ensure reliable and trustworthy performance in error predictions. Because we want the secondary model to predict the error in the predictions of a primary model. To ensure that crucial aspects of the primary model are covered, the dataset for the secondary model should include the error associated with predictions of the primary model, the difference between the values used in the test set and the nearest records in the training set of the primary model, and the impact of feature importance on the primary model's predictions. A dataset with those characteristics has shown potential, as it covers various scenarios where the primary model might make errors. These scenarios can include edge cases and instances where the model's performance is poor.

RQ3. How effective is an error prediction model including features describing the weight of specific inputs on the output obtained using model interpretability techniques?

The use of SHAP values to weigh the differences between test values and their closest training dataset records has shown to be an effective strategy in predicting neural network errors. We think that this approach resulted in a training dataset for the error model with crucial information about the primary model, to understand its behavior. With this fundamental data configuration, the secondary model was able to better predict the primary model error. By considering the contribution of each feature to the prediction using the SHAP values, we could better understand why certain errors were being made and how they can be corrected. This approach is robust to changes in the data distribution, as it considers the local behavior of the model around each prediction. However, it is important to state that while promising this approach still has room for improvement. In future work, we consider focusing on refining the weighting scheme or exploring other interpretability methods to further enhance error prediction.

While the proposed system shows promising results in predicting errors of a neural network, there is still ample room for improvement. This indicates that the system can learn from the discrepancies between the predicted and actual values of the primary model, but not effectively, committing itself errors.

Even with just a glimpse of what could become, the proposed system was able to provide valuable insights into the patterns of errors made by the neural network. We believe that in future work the system can be refined, increasing its prediction accuracy and making it more robust and useful.

RQ4: What insights can be gained into the decision-making process of neural networks by analyzing the error predictions?

When analyzing the differences between test and predicted values, we can gain insight into where the model may be making errors, paying special attention to considerable zones of high discrepancies between true and predicted values. Having this knowledge can be useful during the training process and model architecture, serving as another tool to help make the model more accurate and reliable.

The ability to predict the error associated with predictions helps to demystify the concept of the "black box" often associated with neural networks, bringing more transparency to the prediction process and thereby increasing confidence in their predictions.

The second model is trained using the prediction errors of the first model, which are weighted by the impact of the input features and the difference between the values used in the test set, and the closest records from the training dataset. By utilizing this training data, the second model gains the ability to predict errors associated with the neural network predictions. This insight reveals systematic patterns in the errors made by the neural network, allowing the second model to learn and predict them.

The predictions of the second model provide valuable insights into the performance of the neural network. This information allows us to identify the conditions under which the neural network is more likely to make errors.

Such insights are powerful tools for improving neural networks, significantly enhancing their interpretability.

The insights gained from using a second model to predict the errors of the primary model can be valuable for future research in the field of neural networks. This approach enables researchers to understand potential sources of errors and develop more effective and powerful neural networks.

RQ5: What conclusions can be drawn based on the research findings and what recommendations can be proposed for future research in the field of error predictions, diet planning, and food recommendation systems?

We can conclude that the proposed system is promising. Contributing to making neural networks more transparent and easier to understand, providing important insights into what conditions the primary neural network may fail, and also offering a roadway to improve the reliability and trustworthiness of neural networks in general.

In the specific context of diet planning and food recommendation systems, having the ability to predict errors can help improve the quality of recommenda-

tions, something specially important for healthcare systems. By understanding when and why inaccuracies occur, we can better refine the system to provide more personalized and accurate predictions.

While the secondary model has shown promising results, it is still far from ideal, with a considerable discrepancy between true and predicted values, and there is room for improvement. For future research in the area, aiming to improve the accuracy we could explore different model architectures, different training strategies, and different datasets for the error model. This research focused on a specific type of neural network, future research could try the same approach relying on different types of machine learning models. Adding to future work we could also explore a viable integration of the error prediction systems into diet planning and food recommendation systems. This approach could result in more accurate and personalized recommendations, improving health outcomes.

In conclusion, our research opens up exciting future possibilities to explore, promising to push the boundaries of what is possible in the realm of error prediction and its application in diet planning and food recommendation systems.

5.1 Limitations and Future Work

Our work depends on a simple model that relates macronutrients with calories. Thus, there are complex problems in the nutrition domain that can be explored using the same approach.

As in any machine learning research problem, the results depend on the data used for training models. Thus, other data can lead to different outcomes. However, the results obtained using our methodology are promising, which makes the methodology a candidate for other settings.

Based on the insights, and looking for future work, we may need to use more data, use more features alongside the ones already in use, and in the last instance try a different model. Both models have been tuned and a sensitivity analysis has been conducted. Despite these efforts, the RF prediction of the MLP prediction error has a lot of room for improvement. The significant spikes in the Predicted Values Error suggest certain instances where the RF model fails to predict the error accurately.

Further investigation would be needed to determine the exact cause of the discrepancies found in the predictions, especially in the RF model. This could pass into preprocessing the data differently, building the dataset used by the RF differently, or even trying different model combinations for both ANN and error predictions.

6 Conclusion

For this research, we set goals to develop a food recommendation system, understand neural network predictions, propose an error prediction model, evaluate its effectiveness, and enhance the interpretability of neural networks within a food recommendation system, gaining insights into neural network decision-making.

We aimed to demystify the "black box" idea of neural networks by predicting its prediction errors using a secondary model. With the purpose approach, we looked at making neural networks' inner works easier to understand and therefore more transparent. To feed the secondary model we collect the prediction errors of the primary model, the differences between each test set record and its closest record in the train set, and to enhance the interpretability we also collect the impact that each feature had on the primary model predictions. The second model was trained using the described data from the primary model to predict the errors associated with its predictions, offering insights into its behavior. Although the primary objective was to enhance understanding of neural networks, the research also provided valuable experience in machine learning, particularly in fine-tuning neural network performance.

Throughout this research, we meticulously reviewed related work in the area of machine learning food recommendation systems, recommendations systems, and error prediction, we also collected and preprocessed data, and applied machine learning techniques, focusing specifically on the use of the algorithms Multilayer Perceptron (MLP) and Random Forest (RF). The methodology applied involved data collection, preprocessing, algorithm selection, hyperparameter tuning, training and testing, and the creation of a second dataset.

Despite the promising results, the second model developed in the proposed approach presented a considerable difference between true and predicted values, indicating that it is a good first step but with room for improvement. With the primary model predictions close to true values we expected better error predictions from the second model, however, the predictions were not entirely reliable, although they provided valuable insights into the decision-making process of the primary model. We believe that the results can have significant implications for the field, as they show potential in developing tools and methods to create more reliable and understandable AI systems in dietary planning and beyond, as they identify problems like faulty model architecture or weak datasets for the model to learn.

Looking into future work and seeing this research as a first step in an approach where we see the potential to improve neural network development as a whole, we think the next steps would pass by refining the error prediction model to handle a wider variety of datasets and scenarios. We also think that exploring different neural network architectures and the integration of more sophisticated

interpretability techniques could yield even deeper insights and enhance predictive accuracy. Additionally, future work can expand the scope to include real-time data processing and personalized user feedback, this could significantly enhance the practical applicability of our food recommendation system, and the error prediction approach. The ultimate goal for future work remains to bridge the gap between advanced predictive capabilities and the need for transparency in AI-driven decision-making processes.

This research provided a solid foundation for future exploration in the area of machine learning and contributed to advancing our understanding of neural network operations.

Bibliography

Ahmad, O., Bereux, N., Baret, L., Hashemi, V., & Lecue, F. (2023). Causal Analysis for Robust Interpretability of Neural Networks. *arXiv*.

Ali, J., Khan, R., Ahmad, N., & Maqsood, I. (2012). Random forests and decision trees. *International Journal of Computer Science Issues (IJCSI)*, *9*(5), 272.

Allom, V., Mullan, B., Smith, E., Hay, P., & Raman, J. (2018). Breaking bad habits by improving executive function in individuals with obesity. *BMC Public Health*, *18*(1), 505. https://doi.org/10.1186/s12889-018-5392-y

Barkved, K. (2022). *The difference between training data vs. test data in machine learning*. Retrieved January 19, 2024, from https://www.obviously.ai/post/the-difference-between-training-data-vs-test-data-in-machine-learning

Bergstra, J., Yamins, D., Cox, D. D., et al. (2013). Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. *SciPy*, *13*, 20.

Bikku, T. (2020). Multi-layered deep learning perceptron approach for health risk prediction. *Journal of Big Data*, *7*(1), 50. https://doi.org/10.1186/s40537-020-00316-7

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5–32.

Brownlee, J. (n.d.). *Regression metrics for machine learning*. https://machinelearningmastery.com/regression-metrics-for-machine-learning/ (accessed: 17.03.2024).

Brownlee, J. (2021). *How to choose an activation function for deep learning*. Retrieved January 24, 2024, from https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

Chai, T., & Draxler, R. R. (2014). Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature. *Geoscientific model development*, *7*(3), 1247–1250.

Chen, M., Jia, X., Gorbonos, E., Hoang, C. T., Yu, X., & Liu, Y. (2020). Eating healthier: Exploring nutrition information for healthier recipe recommendation. *Information Processing & Management*, *57*(6), 102051. https://doi.org/10.1016/j.ipm.2019.05.012

Esser-Skala, W., & Fortelny, N. (2023). Reliable interpretability of biology-inspired deep neural networks. *npj Systems Biology and Applications*, *9*(1), 50. https://doi.org/10.1038/s41540-023-00310-8

for Disease Control, C., & Prevention. (2022). *Health effects of overweight obesity*. Retrieved March 16, 2024, from https://www.cdc.gov/healthyweight/effects/index.html

Gao, X., Feng, F., Huang, H., Mao, X.-L., Lan, T., & Chi, Z. (2022). Food recommendation with graph convolutional network. *Information Sciences*, *584*, 170–183. https://doi.org/10.1016/j.ins.2021.10.040

Gurupur, V. P., & Shelleh, M. (2021). Machine learning analysis for data incompleteness (madi): Analyzing the data completeness of patient records using a random variable approach to predict the incompleteness of electronic health records. *IEEE Access*, *9*, 95994–96001. https://doi.org/10.1109/ACCESS.2021.3095240

Hayou, S., Doucet, A., & Rousseau, J. (2018). On the selection of initialization and activation function for deep neural networks. *arXiv preprint arXiv:1805.08266*.

Hu, Y., Huber, A., Anumula, J., & Liu, S.-C. (2018). Overcoming the vanishing gradient problem in plain recurrent networks. *arXiv preprint arXiv:1801.06105*.

IBM. (n.d.-a). *What is a decision tree?* Retrieved November 30, 2023, from https://www.ibm.com/topics/decision-trees

IBM. (n.d.-b). *What is random forest?* Retrieved November 29, 2023, from https://www.ibm.com/topics/random-forest

Isabona, J., Imoize, A. L., Ojo, S., Karunwi, O., Kim, Y., Lee, C.-C., & Li, C.-T. (2022). Development of a Multilayer Perceptron Neural Network for Optimal Predictive Modeling in Urban Microcellular Radio Environments. *Applied Sciences*, *12*(11), 5713. https://doi.org/10.3390/app12115713

Ishizaka, A., Pearman, C., & Nemery, P. (2012). Ahpsort: An ahp-based method for sorting problems. *International Journal of Production Research*, *50*(17), 4767–4784.

Iwendi, C., Khan, S., Anajemba, J. H., Bashir, A. K., & Noor, F. (2020). Realizing an Efficient IoMT-Assisted Patient Diet Recommendation System Through Machine Learning Model. *IEEE Access*, *8*, 28462–28474. https://doi.org/10.1109/access.2020.2968537

Jeon, I., & Kim, T. (2023). Distinctive properties of biological neural networks and recent advances in bottom-up approaches toward a better biologically plausible neural network. *Frontiers in Computational Neuroscience*, *17*, 1092185. https://doi.org/10.3389/fncom.2023.1092185

Kim, J.-C., & Chung, K. (2020). Knowledge-based hybrid decision model using neural network for nutrition management. *Information Technology and Management*, *21*(1), 29–39. https://doi.org/10.1007/s10799-019-00300-5

Kotsiantis, S. B. (2013). Decision trees: A recent overview. *Artificial Intelligence Review*, *39*, 261–283.

Krogh, A. (2008). What are artificial neural networks? *Nature biotechnology*, *26*(2), 195–197.

Kumar, A., Tanwar, P., & Nigam, S. (2016). Survey and evaluation of food recommendation systems and techniques. *2016 3rd International Confer-*

ence on Computing for Sustainable Global Development (INDIACom), 3592–3596.

Kyerematen-Boateng, H., Josyula, D., & Conn, M. (2023). Computing Confidence Score for Neural Network Predictions from Latent Features. *2023 International Conference on Control, Communication and Computing (ICCC)*, *00*, 1–6. https://doi.org/10.1109/iccc57789.2023.10165294

Lee, H. I., Choi, I. Y., Moon, H. S., & Kim, J. K. (2020). A Multi-Period Product Recommender System in Online Food Market based on Recurrent Neural Networks. *Sustainability*, *12*(3), 969. https://doi.org/10.3390/su12030969

Lee, M. C. (2011). A novel sentence similarity measure for semantic-based expert systems. *Expert Systems with Applications*, *38*(5), 6392–6399.

Lee, T.-H., Ullah, A., & Wang, R. (2020). Bootstrap aggregating and random forest. *Macroeconomic forecasting in the era of big data: Theory and practice*, 389–429.

Mu, R. (2018). A Survey of Recommender Systems Based on Deep Learning. *IEEE Access*, *6*, 69009–69022. https://doi.org/10.1109/access.2018.2880197

Nwadiugwu, M. C. (2020). Neural networks, artificial intelligence and the computational brain.

Organization, W. H. (2020). *Healthy diet*. Retrieved March 16, 2024, from https://www.who.int/news-room/fact-sheets/detail/healthy-diet

Pramoditha, R. (2021). *The concept of artificial neurons (perceptrons) in neural networks*. Retrieved January 20, 2024, from https://towardsdatascience.com/the-concept-of-artificial-neurons-perceptrons-in-neural-networks-fab22249cbfc

Puska, P., Nishida, C., Porter, D., Organization, W. H., et al. (2003). Obesity and overweight. *World Health Organization*, 1–2.

Ranjeeth, S., & Latchoumi, T. P. (2020). Predicting Kids Malnutrition Using Multilayer Perceptron with Stochastic Gradient Descent. *Revue d'Intelligence Artificielle*, *34*(5), 631–636. https://doi.org/10.18280/ria.340514

Rostami, M., Oussalah, M., & Farrahi, V. (2022). A Novel Time-Aware Food Recommender-System Based on Deep Learning and Graph Clustering [Conference Name: IEEE Access]. *IEEE Access*, *10*, 52508–52524. https://doi.org/10.1109/access.2022.3175317

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021). "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. https://doi.org/10.1145/3411764.3445518

Sazli, M. H. (2006). A brief review of feed-forward neural networks. *Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering*, *50*(01).

Shambour, Q. (2021). A deep learning based algorithm for multi-criteria recommender systems. *Knowledge-Based Systems*, *211*, 106545. https://doi.org/10.1016/j.knosys.2020.106545

Sharma, S., Sharma, S., & Athaiya, A. (2017). Activation functions in neural networks. *Towards Data Sci*, *6*(12), 310–316.

Toledo, R. Y., Alzahrani, A. A., & Martínez, L. (2019). A Food Recommender System Considering Nutritional Information and User Preferences. *IEEE Access*, *7*, 96695–96711. https://doi.org/10.1109/access.2019.2929413

Yegnanarayana, B. (2009). *Artificial neural networks*. PHI Learning Pvt. Ltd.

Ying, X. (2019). An overview of overfitting and its solutions. *Journal of physics: Conference series*, *1168*, 022022.

ZVIKINOZA. (n.d.). *Myfitnesspal dataset*. Retrieved February 8, 2024, from https://www.kaggle.com/datasets/zvikinozadze/myfitnesspal-dataset