



**islington college**  
(इस्लिंग्टन कॉलेज)

**Module Code & Module Title**

**CU6051NI Artificial Intelligence**

**25% Individual Coursework**

**Submission: Milestone 1**

**Academic Semester: Autumn Semester 2025**

**Credit: 15 credit semester long module**

**Student Name: Ajoob Sagar Kansakar**

**London Met ID: 23056153**

**College ID: NP01CP4S240080**

**Assignment Due Date: 07/01/2026**

**Assignment Submission Date: 07/01/2026**

**Submitted To: Er. Roshan Shrestha**

**GitHub Link**

[https://github.com/AjoobKansakar/AI\\_Coursework.git](https://github.com/AjoobKansakar/AI_Coursework.git)

*I confirm that I understand my coursework needs to be submitted online via MST Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.*

## **Table of contents:**

1.	Introduction .....	1
1.1	Explanation of the Topic / AI Concepts Used .....	1
1.2	Introduction of the Chosen Problem Domain .....	2
2.	Background.....	3
2.1	Research Work Done in Coursework 1 .....	3
3.	Solution .....	7
3.1	Explanation of the Proposed Solution .....	7
3.2	Pseudocode of the solution .....	8
3.2.1	Pseudocode for Linear Regression .....	10
3.2.2	Pseudocode for Decision Tree Regressor .....	11
3.2.3	Pseudocode for Random Forest Regressor .....	12
3.3	Diagrammatical Representation .....	13
3.3.1	Combined Flowchart.....	13
3.3.2	Linear Regression Flowchart.....	14
3.3.3	Decision Tree Regressor Flowchart .....	15
3.3.4	Random Forest Regressor Flowchart.....	16
3.4	Explanation of Development Process .....	17
3.5	Achieved Results .....	18
4.	Conclusion .....	36
5.	References.....	37

## **Table of Figures:**

Figure 1: Article 1 .....	4
Figure 2: Article 2 .....	5
Figure 3: Article 3 .....	6
Figure 4: Combined Flowchart .....	13
Figure 5: Linear Regression Flowchart.....	14
Figure 6: Decision Tree Regressor Flowchart .....	15
Figure 7: Random Forest Regressor Flowchart .....	16
Figure 8: Loading Dataset in Jupyter notebook.....	18
Figure 9: Checking for Missing NaN values .....	19
Figure 10: Checking Dataset Shape and Datatypes .....	19
Figure 11: Statistical Summary of Numerical Columns .....	20
Figure 12: Dropping Invalid Columns .....	21
Figure 13: Calorie Distribution Chart .....	22
Figure 14: Calorie Distribution Heatmap .....	23
Figure 15: Feature Scaling for Linear Regression.....	23
Figure 16: Input Feature(X) for train_test_split .....	24
Figure 17: Target Variable(y) for train_test_split .....	24
Figure 18: Train_test_split.....	25
Figure 19: Imports for algorithm testing.....	26
Figure 20: Test sizes and Evaluation .....	26
Figure 21: Linear Regression .....	27
Figure 22: Decision Tree Regressor.....	28
Figure 23: Random Forest Regressor .....	29
Figure 24: Result Comparing and Best Model Selection .....	30
Figure 25: MAE Comparison .....	31
Figure 26: RSME Comparison .....	32
Figure 27: R2 Comparison .....	33
Figure 28: Combined Bar Chart Comparison of all Algorithms .....	34
Figure 29: Heatmap Comparison of all Algorithms .....	35

## 1. Introduction

### 1.1 Explanation of the Topic / AI Concepts Used

Artificial Intelligence (AI) is the capability of machines to simulate human intelligence, such as learning from data, and making predictions. One of the most widely used branches of Artificial Intelligence is Machine Learning (ML), it is a concept of AI where a system learns patterns from huge set of data and use them to make predictions on unseen data.

Machine Learning problems are mainly categorized into the following categories:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

In this project, Supervised learning concepts are used where the model is trained using pre-labelled data meaning the data contains input features and a known output value.

Machine learning concepts used in this project is Regression which is used when the target variable is a continuous numerical value. The main objective of this project is to calculate the target variable calorie content (Energy in kcal), which is a continuous numerical value. Therefore, Regression was chosen for this project.

In order to predict calories based on nutritional information, the project uses the following regression algorithms:

- Linear Regression
- Decision Tree Regressor
- Random Forest Regressor

These algorithms study the relationship between nutritional attributes such as proteins, carbohydrates, and fats, and the corresponding calorie values.

## 1.2 Introduction of the Chosen Problem Domain

Nutrition plays a crucial role in maintaining a healthy lifestyle, and calorie estimation is one of the most important in health management, diet planning, and fitness tracking. Daily calorie intake directly affects one's body weight, energy levels, and overall health. This project was carried out to understand nutrition labels to calculate calorie intake as not all the food items consist of a nutrition label or accurate calorie information.

With the increasing availability of nutritional datasets, this project focuses on the development of a machine learning software that provides an effective solution to automatically predict calorie content based on known nutritional properties. By analyzing the relationship between macronutrients such as carbohydrates, proteins, and fats, it is possible to estimate calorie values with upmost accuracy.

The aim of the project is to predict the calorie content of food items using machine learning processes such as regression models trained on a real-world nutrition dataset. The project solution aims to demonstrate the use of artificial intelligence to assist in automating calorie estimation and support healthy lifestyle.

For this project, the dataset was obtained from Hugging Face:

<https://huggingface.co/datasets/adarshzolekar/foods-nutrition-dataset>

## 2. Background

### 2.1 Research Work Done in Coursework 1

During Coursework 1, many of the foundational machine learning concepts were studied which included data pre-processing, supervised learning, regression techniques, and model evaluation. During initial data exploration, several issues were identified and special priority was placed on understanding how regression models learn relationships between input variable and continuous output values.

After reviewing the dataset, concepts such as train-test splitting, feature normalization, and evaluation metrics like Mean Absolute Error (MAE), Root Mean Square (RMSE), and R2 score were introduced which forms the theoretical foundation for this project and are applied directly for predicting calorie values from nutritional dataset.

Several studies have been conducted for food analysis and calorie estimation using machine learning techniques like regression-based models that can effectively predict calorie values when trained on nutritional attributes like macronutrients and portion sizes. Some of the articles related to the issue is provided below:

## 1) Machine learning In Nutrition Research

This study provides a review of how machine learning techniques are used in nutrition research and also discusses the use of supervised learning models, including regression-based approaches, to analyse nutritional datasets. The article highlights that traditional methods of estimating calorie content struggles with complex nutrition data, whereas using the machine learning models are more effective in estimating the calories of a food. This research supports the use of regression models for calorie prediction tasks (Antwi, J.B, Tuffour, Mensah, 2022).

**Machine Learning in Nutrition Research**

Daniel Kirk <sup>1,2\*</sup>, Esther Kok <sup>2</sup>, Michele Tufano <sup>3</sup>, Bedir Tekinerdogan <sup>4</sup>, Edith J M Feskens <sup>5</sup>, Guido Campos <sup>6,7</sup>

Author information Article notes Copyright and License information

PMCID: PMC9776646 PMID: 36166846

This article has been corrected. See [Adv Nutr. 2023 Mar 10;14\(3\):584.](#)

**ABSTRACT**

Data currently generated in the field of nutrition are becoming increasingly complex and high-dimensional, bringing with them new methods of data analysis. The characteristics of machine learning (ML) make it suitable for such analysis and thus lend itself as an alternative tool to deal with data of this nature. ML has already been applied in important problem areas in nutrition, such as obesity, metabolic health, and malnutrition. Despite this, experts in nutrition are often without an understanding of ML, which limits its application and therefore potential to solve currently open questions. The current article aims to bridge this knowledge gap by supplying nutrition researchers with a resource to facilitate the use of ML in their research. ML is first explained and distinguished from existing solutions, with key examples of applications in the nutrition literature provided. Two case studies of domains in which ML is particularly applicable, precision nutrition and metabolomics, are then presented. Finally, a framework is outlined to guide interested researchers in integrating ML into their work. By acting as a resource to which researchers can refer, we hope to support the integration of ML in the field of nutrition to facilitate modern research.

**Keywords:** machine learning, personalized nutrition, omics, obesity, diabetes, cardiovascular disease, models, random forest, XGBoost

**RESOURCES**

- Similar articles +
- Cited by other articles +
- Links to NCBI Databases +

**ON THIS PAGE**

- ABSTRACT
- Introduction
- Machine Learning Capabilities
- Machine Learning Overview
- Case Studies: Applications of Machine Learning in Nutrition Domains
- Framework for Applying Machine Learning in Nutrition Science
- Limitations of Machine Learning in Nutrition Research
- Conclusion
- Supplementary Material
- Acknowledgements
- Notes
- Contributor Information
- References
- Associated Data

Figure 1: Article 1

## 2) AI Applications to Measure Food and Nutrient Intakes

This research analyses multiple artificial intelligence approaches used for estimating food and nutrient intake by reviewing supervised learning techniques, including regression-based models, that are applied to predict calories in a food. The authors highlight that machine learning models improve accuracy compares to manual traditional estimation methods (Zheng J, Wang J, Shen J, An R , 2023).

The screenshot shows the JMIR Publications website interface. At the top, there is a navigation bar with links for 'Articles', 'Search articles', 'Career Center', 'Login', and 'Register'. Below the navigation bar, there are tabs for 'Journal of Medical Internet Research', 'Journal Information', 'Browse Journal', and 'Submit Article'. The main content area displays the article details:

- Published on 28.Nov.2024 in Vol 26 (2024)**
- Preprints (earlier versions) of this paper are available at <https://preprints.jmir.org/preprint/54557>, first published 14.Nov.2023.**
- Artificial Intelligence Applications to Measure Food and Nutrient Intakes: Scoping Review**
- Authors:** Jiakun Zheng<sup>1</sup>, Junjie Wang<sup>2</sup>, Jing Shen<sup>3</sup>, Ruopeng An<sup>4</sup>
- Metrics:** Cited by (25), Tweetations, Metrics
- Abstract:** (Link to full abstract)
- Background:** Accurate measurement of food and nutrient intake is crucial for nutrition research, dietary surveillance, and disease management, but traditional methods such as 24-hour dietary recalls, food diaries, and food frequency questionnaires are often prone to recall error and social desirability bias, limiting their reliability. With the advancement of artificial intelligence (AI), there is potential to overcome these limitations through automated, objective, and scalable dietary assessment techniques. However, the effectiveness and challenges of AI applications in this domain remain inadequately explored.
- Objective:** This study aimed to conduct a scoping review to synthesize existing literature on the efficacy, accuracy, and challenges of using AI tools in assessing food and nutrient intakes, offering insights into their current advantages and areas of improvement.
- Methods:** This review followed the PRISMA-ScR (Preferred Reporting Items for Systematic Reviews and Meta-Analyses extension for Scoping Reviews) guidelines. A comprehensive literature search was conducted in 4 databases—PubMed, Web of Science, Cochrane Library, and EBSCO—covering publications from the databases' inception to June 30, 2023. Studies were included if they used machine learning approaches to measure food and nutrient intake in human subjects.

On the right side of the page, there are sections for 'Citation', 'Please cite as:', and 'Copy Citation to Clipboard'. There is also a 'Export Metadata' section with links for 'END for: Endnote', 'BibTeX for: BibDesk, LaTeX', 'RIS for: RefMan, Procite, Endnote, RefWorks', and 'Add this article to your Mendeley library'. Below these are sections for 'This paper is in the following e-collection/theme issue:' (Digital Health Reviews (1749), Instruments and Questionnaires for Nutrition and Food Intake (216), Innovations and Technology for Healthy Eating Education (492), Artificial Intelligence (2170), Applications of AI (295)) and a 'Download' section with various file formats (PDF, Word, etc.).

Figure 2: Article 2

### 3) Predicting Nutrient Density in Foods Using Machine Learning Models

This research focuses on predicting the nutrition density in food items using various machine learning regression models and compares algorithms such as Linear Regression, Decision Tree Regression, and Random Forest Regression to predict nutritional datasets. This study also shows performance metrics such as Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R<sup>2</sup> score are used to gain model accuracy. This study shows that Random Forest outperform simpler models due to their ability to capture non-linear relationships in nutritional data which directly aligns with the models used in this project (Changhe Yang, 2024).

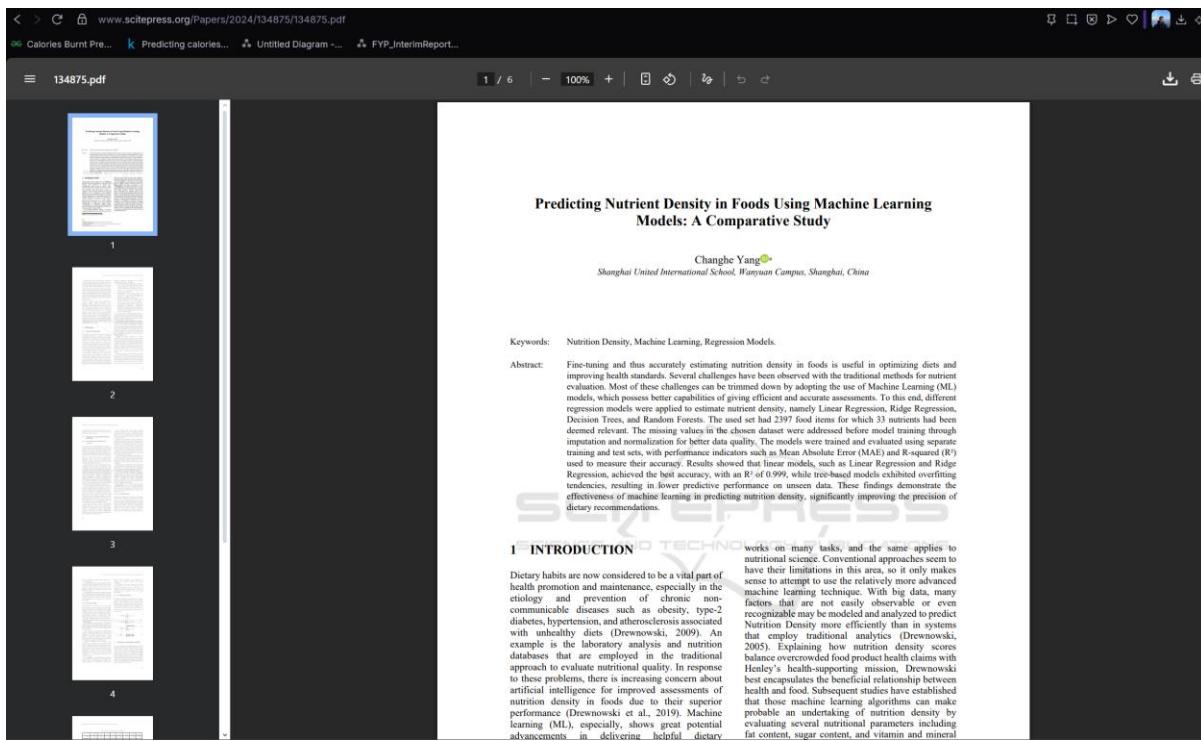


Figure 3: Article 3

### 3. Solution

#### 3.1 Explanation of the Proposed Solution

The proposed solution for this project uses machine learning regression techniques to predict and estimate calorie content of food items based on their nutritional values. After reviewing the dataset obtained from Hugging Face, it was observed that detailed nutritional information for various food items such as carbohydrates, protein, fats, fiber and other nutrients including energy (kcal).

During initial data exploration, only the relevant attributes were selected for further analysis to ensure data quality and consistency. The dataset was then divided into training and testing sets to evaluate model performance on unseen data.

The proposed solution for this project involves developing a machine learning based system that predicts nutritional values using the selected food dataset. The system uses Linear Regression, Decision Tree, and Random Forest models to compare accuracy and performance. Linear Regression was initially implemented as a baseline model due to its simplicity as it is easier to interpret. Decision Tree Regressor was implemented to capture non-linear relationship between nutritional features and calorie values. Random forest was later introduced to improve prediction accuracy and reduce overfitting.

The performance of each of the mentioned model is evaluated using standard regression metrics, and the results are compared to determine the most effective approach to estimate and calories of a food.

### 3.2 Pseudocode of the solution

Pseudocode refers to the step-by-step description of an algorithm without using any programming language using simple English language for human understanding which helps developers to plan the system logic and structure before writing the actual coding part.

**START**

**IMPORT** required libraries

**LOAD** Food nutrition dataset

**DISPLAY** dataset structure and summary statistics

**SELECT INPUT** features (protein, carbohydrates, fats, etc.)

**SELECT** target variable (Energy kcal)

**PREPROCESS** data:

**HANDLE** missing values

**APPLY** feature scaling using StandardScaler

For each feature:

    Compute mean and standard deviation

**SPLIT** dataset **INTO** training **AND** testing sets

**FOR EACH** algorithm **IN** [Linear Regression, Decision Tree, Random Forest]:

**INITIALIZE** model

**TRAIN** model **ON** training data

**PREDICT** calories **ON** test data

**COMPUTE** evaluation metrics:

Mean Absolute Error (MAE)

Root Mean Squared Error (RMSE)

R2 Score

**STORE** model performance results

**COMPARE** all models based on evaluation metrics

**IDENTIFY** best-performing model

**VISUALIZE** results using bar charts and heatmaps

**END**

### 3.2.1 Pseudocode for Linear Regression

**START**

**IMPORT** required libraries

**LOAD** nutrition dataset

**SELECT** input features (Protein, Carbohydrates, Fat, Fibre, Sugar, Cholesterol, Calcium)

**SELECT** target variables (Energy kcal)

**PREPROCESS** dataset:

**HANDLE** missing values

**APPLY** feature scaling using StandardScaler

**SPLIT** dataset into training and testing sets

**INITIALIZE** Linear Regression model

**TRAIN** model using training data

**PREDICT** calorie values for test data

**EVALUATE** model performance:

**COMPUTE** MAE = average of [actual - predicted]

**COMPUTE** RMSE = square root of average squared errors

**COMPUTE** R2 = proportion of variance explained by model

**STORE** evaluation results

**END**

### 3.2.2 Pseudocode for Decision Tree Regressor

**START**

**IMPORT** required libraries

**LOAD** food nutrition dataset

**SELECT** input features (Protein, Carbohydrates, Fat, Fibre, Sugar, Cholesterol, Calcium)

**SELECT** target variables (Energy kcal)

**PREPROCESS** dataset:

**HANDLE** missing values

**SPLIT** dataset into training and testing sets

**INITIALIZE** Decision Tree Regressor

**TRAIN** model using training data

**PREDICT** calorie values for test dataset

**EVALUATE** model performance:

**COMPUTE** MAE

**COMPUTE** RMSE

**COMPUTE** R2 Score

**STORE** evaluation results

**END**

### 3.2.3 Pseudocode for Random Forest Regressor

**START**

**IMPORT** required libraries

**LOAD** nutrition dataset

**SELECT** input features (Protein, Carbohydrates, Fat, Fibre, Sugar, Cholesterol, Calcium)

**SELECT** target variables (Energy kcal)

**PREPROCESS** dataset:

**HANDLE** missing values

**APPLY** feature scaling where required

**SPLIT** dataset into training and testing sets

**INITIALIZE** Random Forest Regressor with multiple decision trees

**TRAIN** model using training data

**PREDICT** calorie values:

**COLLECT** predictions from all trees

**COMPUTE** final prediction as the average of all tree predictions

**EVALUATE** model performance:

**COMPUTE** MAE

**COMPUTE** RMSE

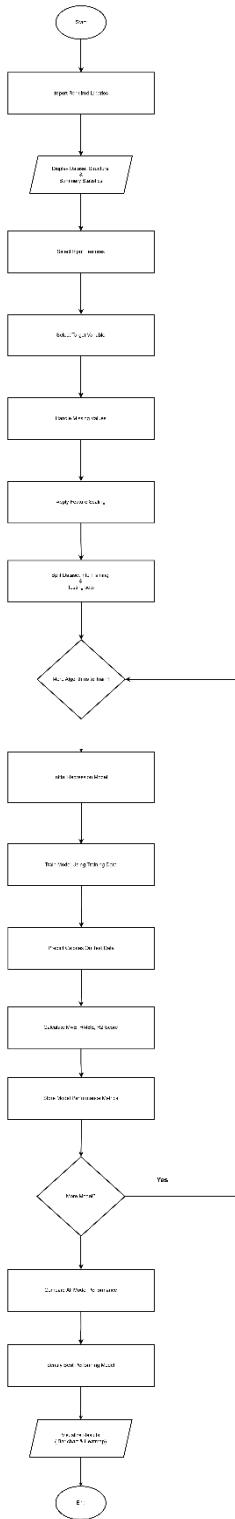
**COMPUTE** R2 Score

**STORE** evaluation results

**END**

### 3.3 Diagrammatical Representation

### 3.3.1 Combined Flowchart



*Figure 4: Combined Flowchart*

### 3.3.2 Linear Regression Flowchart

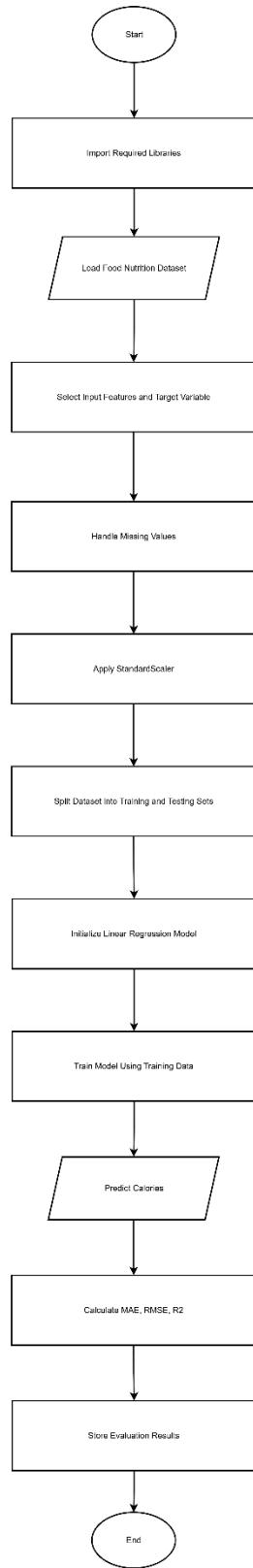


Figure 5: Linear Regression Flowchart

### 3.3.3 Decision Tree Regressor Flowchart

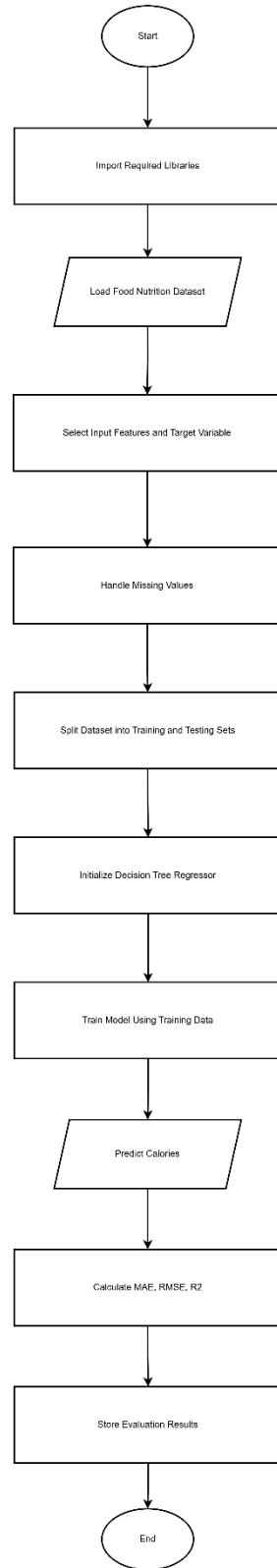


Figure 6: Decision Tree Regressor Flowchart

### 3.3.4 Random Forest Regressor Flowchart

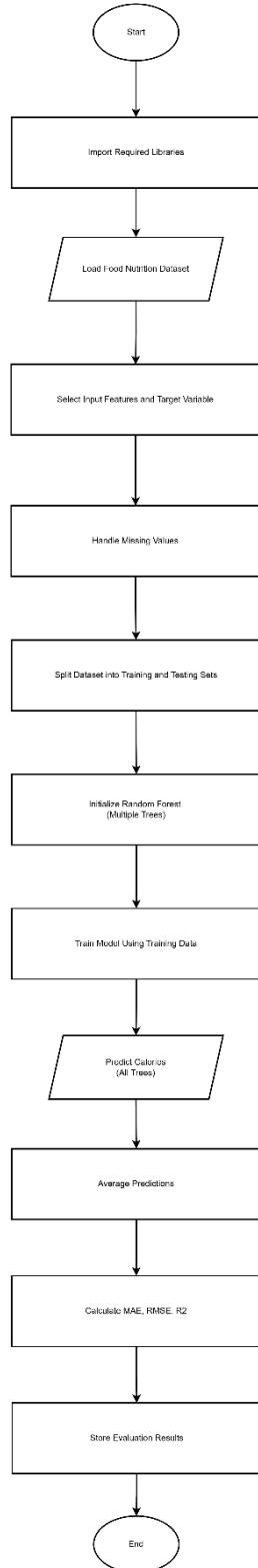


Figure 7: Random Forest Regressor Flowchart

### **3.4 Explanation of Development Process**

The development process for this project was done using Python programming language due to its extensive support for data analysis and machine learning and the implementation of the project was carried out in Jupyter Notebook, which allows interactive data exploration and visualization.

The development process began with data preparation, followed by model implementation and evaluation and the following key libraries were used during the development process:

1. Pandas: Data manipulation and pre-processing
2. NumPy: Numerical Computations
3. Scikit-learn: Implementing regression models and evaluation metrics
4. Matplotlib / Seaborn: Data visualization

### 3.5 Achieved Results

```
[1]: import pandas as pd
[2]: df = pd.read_csv('Nutrition_Dataset_AICoursework/foods.csv')
df
```

	Food Items	Energy kcal	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)	Cholesterol(mg)	Calcium(mg)
0	Butternaan	300.00	50.00	7.00	10.00	2.00	2.00	15.0	50.00
1	Cupcake	200.00	30.00	2.00	8.00	20.00	0.50	20.0	20.00
2	Donuts	250.00	30.00	3.00	12.00	10.00	1.00	20.0	20.00
3	French Fries	312.00	41.00	3.40	15.00	0.30	3.80	0.0	20.00
4	Garlic Bread	200.00	25.00	4.00	10.00	1.00	1.00	10.0	30.00
...	...	...	...	...	...	...	...	...	...
1023	Sweet and sour tomato pickle (Khatta meetha ta...	60.88	6.55	1.26	3.24	4.31	2.20	0.0	15.18
1024	Jhatpat achar with carrot (Jhatpat achaar gaja...	91.21	6.32	1.98	6.55	3.04	5.08	0.0	54.31
1025	Tomato chutney (Tamatar ki chutney)	176.07	31.85	0.97	6.01	30.02	1.49	0.0	25.34
1026	Tomato ketchup	33.07	6.48	0.91	0.30	4.68	1.90	0.0	15.33
1027	Bengal 5 Spice Blend (Panch Phoran)	289.79	20.00	18.26	22.16	1.40	18.40	0.0	523.00

1028 rows × 9 columns

Figure 8: Loading Dataset in Jupyter notebook

Importing the Pandas library for data manipulation and analysis in Python.

`pd.read_csv('Nutrition_Dataset_AICoursework/foods.csv')` to load the raw nutrition dataset

#### Output:

Dataset has been successfully loaded, revealing a total of 1028 rows and 9 columns



The screenshot shows a JupyterLab interface with a notebook titled 'Coursework\_C... - JupyterLab' and a file path 'e/Sem5-(AI)/Coursework\_CalorieEstimation.ipynb'. The code cell [3] contains:

```
[3]: df.isnull().values.any()
[3]: np.False_
[4]: print(df.isnull().sum())
Food Items      0
Energy kcal     0
Carbs          0
Protein(g)     0
Fat(g)          0
Freesugar(g)   0
Fibre(g)        0
Cholesterol(mg) 0
Calcium(mg)     0
dtype: int64
```

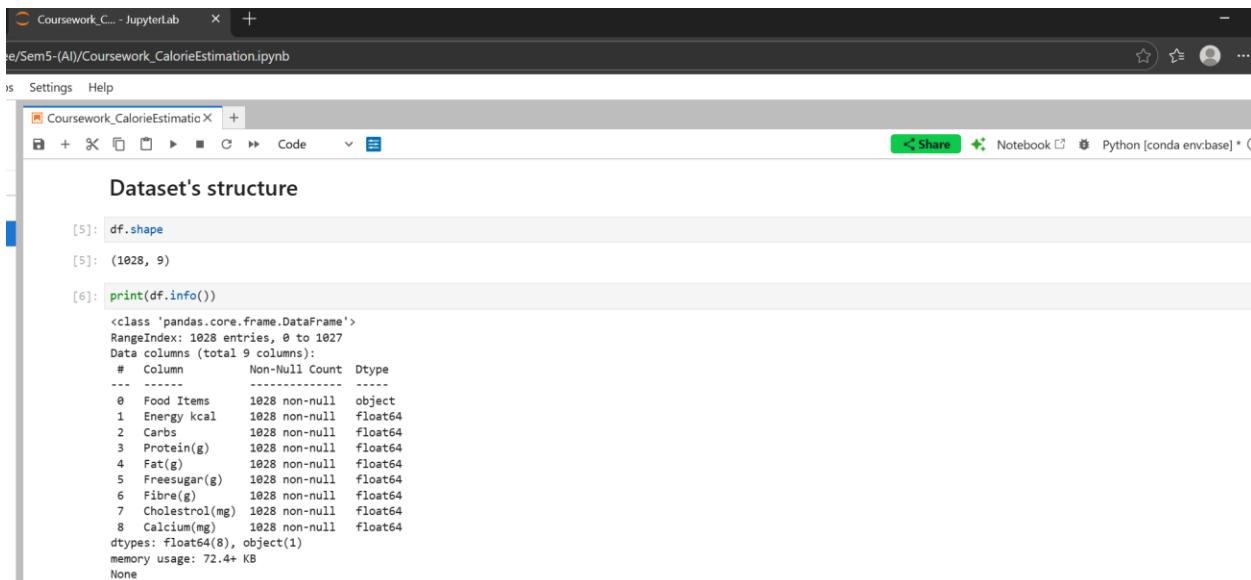
Figure 9: Checking for Missing NaN values

**df.isnull().values.any()** to check any missing values(NaN) in the dataset

**print(df.isnull().sum())** to display total number of null entries in each columns

### Output:

Dataset is complete with no missing values in any of the columns



The screenshot shows a JupyterLab interface with a notebook titled 'Coursework\_C... - JupyterLab' and a file path 'e/Sem5-(AI)/Coursework\_CalorieEstimation.ipynb'. The code cell [5] contains:

```
[5]: df.shape
[5]: (1028, 9)
[6]: print(df.info())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1028 entries, 0 to 1027
Data columns (total 9 columns):
 #   Column      Non-Null Count  Dtype  
--- 
 0   Food Items    1028 non-null   object  
 1   Energy kcal    1028 non-null   float64 
 2   Carbs         1028 non-null   float64 
 3   Protein(g)    1028 non-null   float64 
 4   Fat(g)         1028 non-null   float64 
 5   Freesugar(g)  1028 non-null   float64 
 6   Fibre(g)       1028 non-null   float64 
 7   Cholesterol(mg) 1028 non-null   float64 
 8   Calcium(mg)    1028 non-null   float64 
dtypes: float64(8), object(1)
memory usage: 72.4+ KB
None
```

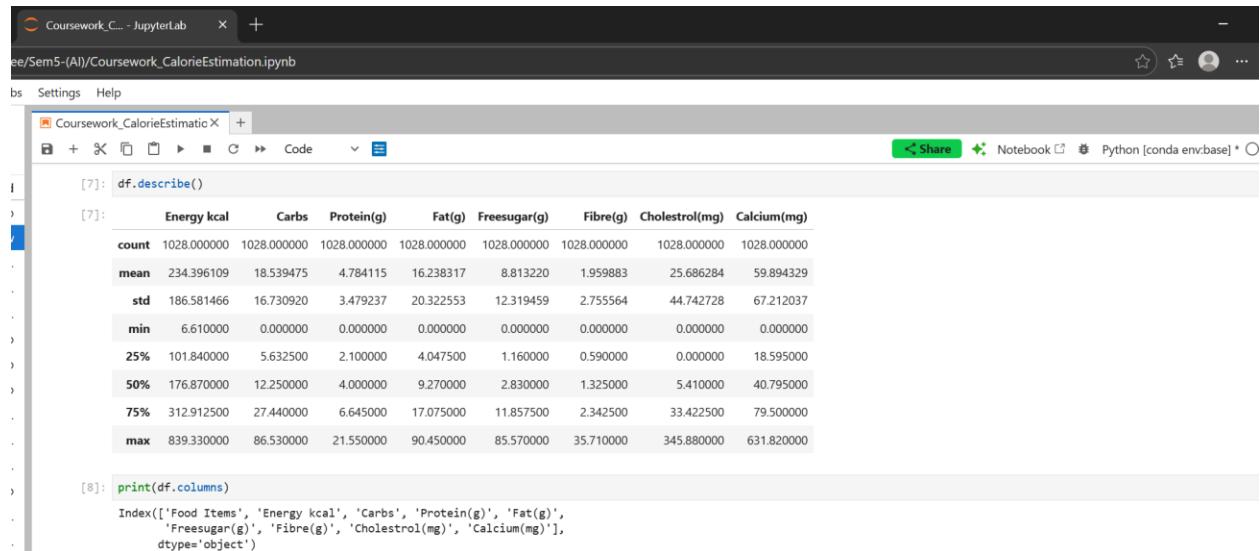
Figure 10: Checking Dataset Shape and Datatypes

**df.shape** to check the rows and columns of the dataset

**df.info()** to show summary of the dataframe, which includes number of null-null entries and the datatype of each columns

### output:

Displays that the dataset consists of 1028 rows and 9 columns and reveals that 8 of the columns are numerical(float64) and 1 column is an object(string)



```
[7]: df.describe()
[7]:   Energy kcal      Carbs  Protein(g)      Fat(g)  Freesugar(g)    Fibre(g) Cholesterol(mg)  Calcium(mg)
count 1028.000000 1028.000000 1028.000000 1028.000000 1028.000000 1028.000000 1028.000000 1028.000000
mean 234.396109 18.539475 4.784115 16.238317 8.813220 1.959883 25.686284 59.894329
std 186.581466 16.730920 3.479237 20.322553 12.319459 2.755564 44.742728 67.212037
min 6.610000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000
25% 101.840000 5.632500 2.100000 4.047500 1.160000 0.590000 0.000000 18.595000
50% 176.870000 12.250000 4.000000 9.270000 2.830000 1.325000 5.410000 40.795000
75% 312.912500 27.440000 6.645000 17.075000 11.857500 2.342500 33.422500 79.500000
max 839.330000 86.530000 21.550000 90.450000 85.570000 35.710000 345.880000 631.820000

[8]: print(df.columns)
Index(['Food Items', 'Energy kcal', 'Carbs', 'Protein(g)', 'Fat(g)', 'Freesugar(g)', 'Fibre(g)', 'Cholesterol(mg)', 'Calcium(mg)'],
      dtype='object')
```

Figure 11: Statistical Summary of Numerical Columns

**df.describe()** generates statistics that shows central tendency, dispersion, and shape of the dataset's distribution

**df.columns** lists all the columns of the dataset

### Output:

Mean, Standard deviation, minimum, maximum, and quartile values displayed for all nutritional attributes.

The screenshot shows a Jupyter Notebook interface with the title 'Coursework\_CalorieEstimation.ipynb'. In the code editor, the following code is written:

```
[9]: df = df.drop(columns=['Food_Items'])
df
```

This code creates a new DataFrame 'df' by dropping the 'Food\_Items' column from the original DataFrame. The resulting DataFrame has 1028 rows and 8 columns, with the first few rows of data being:

	Energy kcal	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)	Cholesterol(mg)	Calcium(mg)
0	300.00	50.00	7.00	10.00	2.00	2.00	15.0	50.00
1	200.00	30.00	2.00	8.00	20.00	0.50	20.0	20.00
2	250.00	30.00	3.00	12.00	10.00	1.00	20.0	20.00
3	312.00	41.00	3.40	15.00	0.30	3.80	0.0	20.00
4	200.00	25.00	4.00	10.00	1.00	1.00	10.0	30.00
...	...	...	...	...	...	...	...	...
1023	60.88	6.55	1.26	3.24	4.31	2.20	0.0	15.18
1024	91.21	6.32	1.98	6.55	3.04	5.08	0.0	54.31
1025	176.07	31.85	0.97	6.01	30.02	1.49	0.0	25.34
1026	33.07	6.48	0.91	0.30	4.68	1.90	0.0	15.33
1027	289.79	20.00	18.26	22.16	1.40	18.40	0.0	523.00

Below the table, it says '1028 rows × 8 columns'.

```
[10]: df.columns
```

```
[10]: Index(['Energy kcal', 'Carbs', 'Protein(g)', 'Fat(g)', 'Freesugar(g)', 'Fibre(g)', 'Cholesterol(mg)', 'Calcium(mg)'], dtype='object')
```

Figure 12: Dropping Invalid Columns

**df.drop()** removes the selected column from the dataset

### Output:

Invalid column (food names) was dropped as it is not needed to estimate the calorie content of the food item.

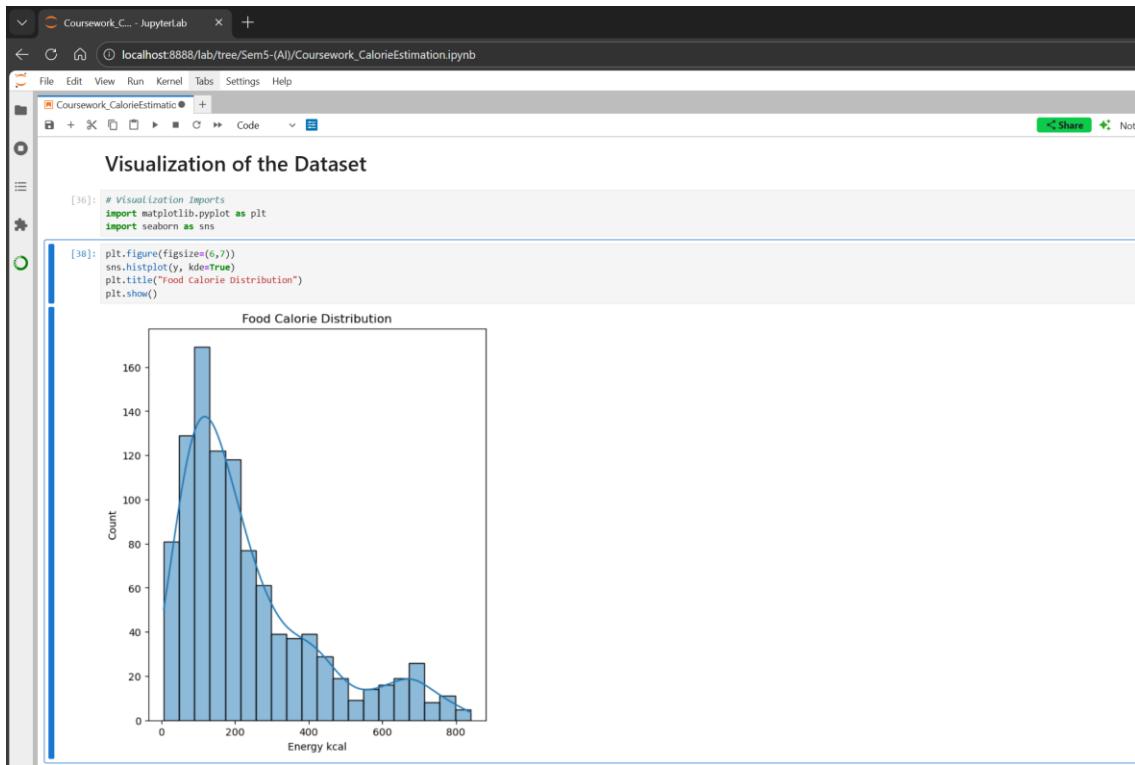


Figure 13: Calorie Distribution Chart

For Visual representation of data using **Seaborn** and **Matplotlib libraries**

**sns.histplot()** to visualize target variable and how calorie values are distributed across dataset

#### Output:

Displays distribution of right-skewed, therefore most of food items in the dataset contain lower calorie counts(0-300kcals), with fewer reaching the high end of the scale(>600kcals).

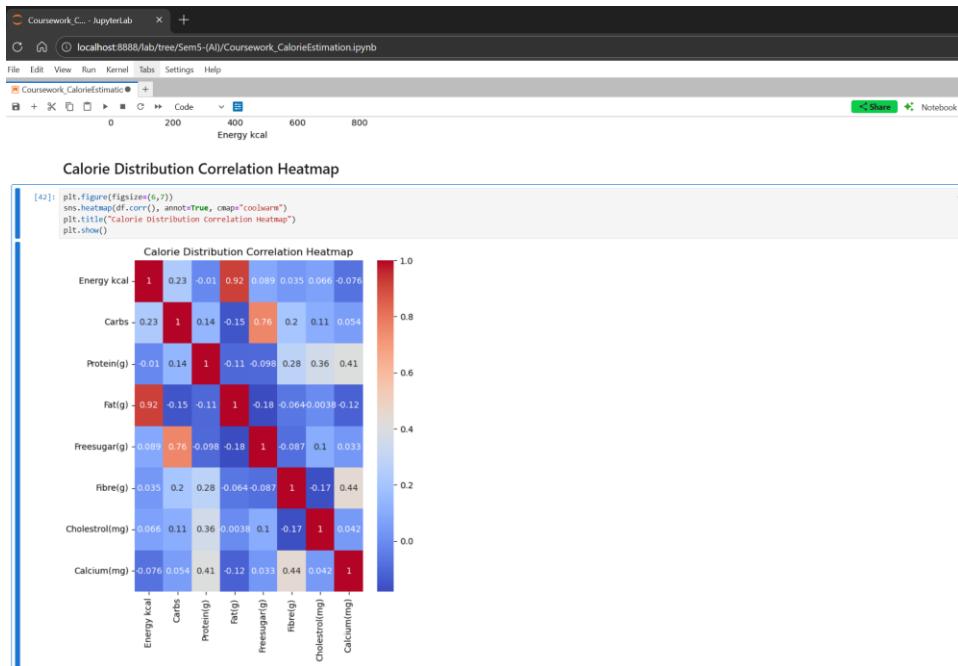


Figure 14: Calorie Distribution Heatmap

`df.corr()` to calculate the correlation matrix

`sns.heatmap` to visualize the correlation matrix

#### Output:

The heatmap shows a very strong positive correlation (0.92) between Fat(g) and Energy kcals, meaning the majority of the calories in food comes from fat contain.

It also shows that moderate correlation with Carbs (0.23), providing insight into which features the model should prioritize.

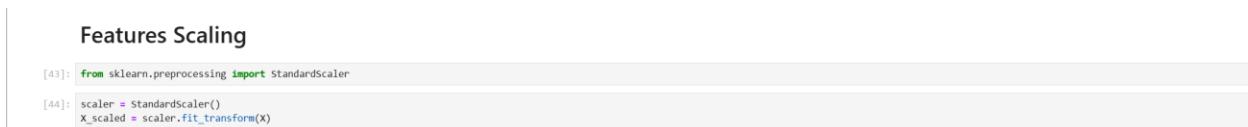


Figure 15: Feature Scaling for Linear Regression

**StandardScaler library** from Scikit-learn used to standardised different measuring scales to transform the data to mean of 0 and a standard deviation of 1.

#### Output:

Successfully normalized into `x_scaled` to scale the features with larger raw values.

```
[11]: X = df.drop(columns=['Energy kcal'])
X
```

	Carbs	Protein(g)	Fat(g)	FreeSugar(g)	Fibre(g)	Cholesterol(mg)	Calcium(mg)
0	50.00	7.00	10.00	2.00	2.00	15.0	50.00
1	30.00	2.00	8.00	20.00	0.50	20.0	20.00
2	30.00	3.00	12.00	10.00	1.00	20.0	20.00
3	41.00	3.40	15.00	0.30	3.80	0.0	20.00
4	25.00	4.00	10.00	1.00	1.00	10.0	30.00
...	...	...	...	...	...	...	...
1023	6.55	1.26	3.24	4.31	2.20	0.0	15.18
1024	6.32	1.98	6.55	3.04	5.08	0.0	54.31
1025	31.85	0.97	6.01	30.02	1.49	0.0	25.34
1026	6.48	0.91	0.30	4.68	1.90	0.0	15.33
1027	20.00	18.26	22.16	1.40	18.40	0.0	523.00

1028 rows × 7 columns

Figure 16: Input Feature(X) for train\_test\_split

### Output:

Separated the independent variables (Features) from the dependent variable and creating a new DataFrame X to contain only the nutritional components.

```
[12]: y = df['Energy kcal']
y
```

	Energy kcal
0	300.00
1	200.00
2	250.00
3	312.00
4	200.00
...	...
1023	66.88
1024	91.21
1025	176.87
1026	33.87
1027	289.79

Name: Energy kcal, Length: 1028, dtype: float64

Figure 17: Target Variable(y) for train\_test\_split

### Output:

Displays the target variable, which is the value to predict.

```
[12]: from sklearn.model_selection import train_test_split
[14]: X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.2, random_state=42)
[14]:
```

	Carbs	Protein(g)	Fat(g)	Fructose(g)	Fibre(g)	Cholesterol(mg)	Calcium(mg)
622	5.06	3.95	1.86	3.97	1.39	5.41	142.96
265	6.35	4.04	2.08	5.22	0.86	7.21	138.58
589	5.47	0.79	51.79	0.80	1.44	0.00	8.11
467	19.42	6.97	5.11	1.11	4.25	0.00	39.34
650	15.62	5.89	9.78	13.88	0.00	98.14	55.03
...	-	-	-	-	-	-	-
458	2.69	0.44	0.39	0.10	0.43	0.00	5.21
330	52.86	6.71	21.86	33.28	4.68	27.87	106.42
466	21.52	4.89	8.07	1.40	3.76	0.00	28.01
121	8.76	1.98	59.81	1.11	0.62	0.00	33.09
860	2.21	14.01	9.41	0.83	0.99	43.28	34.86

822 rows × 7 columns

```
[15]: y_train
```

622	52.75
265	59.13
589	491.76
467	154.89
650	169.87
...	
458	16.26
330	422.21
466	188.52
121	581.91
860	159.24

Name: Energy kcal, Length: 822, dtype: float64

```
[16]: X_test
```

	Carbs	Protein(g)	Fat(g)	Fructose(g)	Fibre(g)	Cholesterol(mg)	Calcium(mg)
428	3.70	2.46	90.45	0.17	1.47	0.00	12.16
533	3.37	1.13	0.88	0.83	1.60	0.00	16.56
388	4.81	2.21	72.57	0.63	0.18	37.78	30.01
107	18.36	9.41	13.91	0.99	3.44	25.66	19.35
423	58.00	5.33	24.92	21.73	1.37	63.58	15.97
...	-	-	-	-	-	-	-
593	2.34	2.49	2.48	1.08	2.83	0.00	26.99
522	10.04	9.35	9.51	1.41	1.08	13.91	73.66
371	69.17	2.43	0.70	61.86	3.74	0.00	72.05
984	51.96	0.30	0.08	51.31	0.41	0.00	6.39
277	15.88	3.43	4.20	12.88	0.51	0.00	105.99

206 rows × 7 columns

```
[17]: y_test
```

428	929.33
533	26.74
388	681.28
107	238.09
423	475.33
...	
593	46.76
522	97.43
371	298.13
984	108.33
277	113.21

Name: Energy kcal, Length: 206, dtype: float64

Figure 18: Train\_test\_split

**train\_test\_split** used to divide the data into 2 separated sets

**test\_size=0.2** 80% of data used for training and 20% data reserved to test performance on unseen data.

**random\_state=42** ensures that the random value generated always remains constant.

### Output:

Data split into 822 training samples and 206 testing samples.

### Algorithm Testing

```
[78]: # for Preprocessing Data
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split

[79]: # Models used for testing
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor

[80]: # to calculate Evaluation metrics
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

Figure 19: Imports for algorithm testing

### Output:

Imports for all the libraries used

#### Test Sizes

```
[81]: test_sizes = [0.2, 0.3, 0.4]
```

```
[ ]:
```

#### Evaluation Helper

```
[82]: def evaluate_model(y_test, y_pred):
    mae = mean_absolute_error(y_test, y_pred)
    rmse = np.sqrt(mean_squared_error(y_test, y_pred))
    r2 = r2_score(y_test, y_pred)
    return mae, rmse, r2
```

Figure 20: Test sizes and Evaluation

### Output:

3 different Test sizes for the algorithms where:

1. 80% Train, 20% Test
2. 70% Train, 30% Test
3. 60% Train, 40% Test

## Linear Regression

```
[83]: # to store the results to compare later
lr_results = []

[84]: # training and testing for different test sizes
for size in test_sizes:
    X_train, X_test, y_train, y_test = train_test_split (
        X_scaled, y, test_size=size, random_state=42
    )

[103]: lr = LinearRegression()
lr

[103]: <__main__.LinearRegression object at 0x0000000003E8A8D0>

[104]: lr.fit(X_train, y_train)

[104]: <__main__.LinearRegression object at 0x0000000003E8A8D0>

[86]: y_pred = lr.predict(X_test)

[87]: mae, rmse, r2 = evaluate_model(y_test, y_pred)

[88]: lr_results.append(["Linear Regression", size, mae, rmse, r2])
```

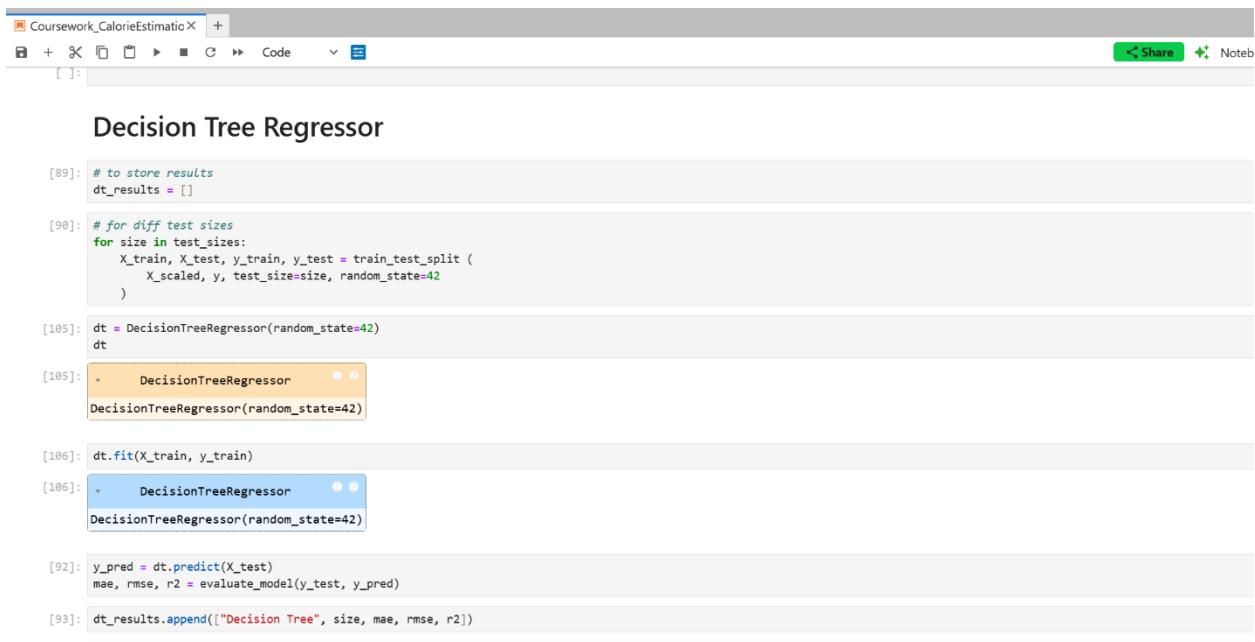
Figure 21: Linear Regression

### Output:

Provides the baseline for the project, uses **For** loop to iterate through different test sizes, spitted scaled features (X\_scaled)

**.fit()** training the model

**lr\_results** to store results in a dedicated list



```

Coursework_CalorieEstimation
+ Share Noteb
Code
[ ] + [ ] Decision Tree Regressor

[89]: # to store results
dt_results = []

[90]: # for diff test sizes
for size in test_sizes:
    X_train, X_test, y_train, y_test = train_test_split (
        X_scaled, y, test_size=size, random_state=42
    )

[105]: dt = DecisionTreeRegressor(random_state=42)
dt
DecisionTreeRegressor()
DecisionTreeRegressor(random_state=42)

[106]: dt.fit(X_train, y_train)
dt
DecisionTreeRegressor()
DecisionTreeRegressor(random_state=42)

[92]: y_pred = dt.predict(X_test)
mae, rmse, r2 = evaluate_model(y_test, y_pred)

[93]: dt_results.append(["Decision Tree", size, mae, rmse, r2])

```

Figure 22: Decision Tree Regressor

## Output:

Non-linear relationships shown

Displays complexity of a single tree compared to the linear baseline

```

File Edit View Run Kernel Tabs Settings Help
Coursework_CalorieEstimator X +
Share Notebook
Random Forest Regressor

[94]: # storing the results
rf_results = []

[95]: for size in test_sizes:
    X_train, X_test, y_train, y_test = train_test_split (
        X_scaled, y, test_size=size, random_state=42
    )

[97]: rf = RandomForestRegressor (
    n_estimators=100,
    random_state=42
)
rf

[107]: RandomForestRegressor(random_state=42)

[108]: rf.fit(X_train, y_train)

[108]: RandomForestRegressor(random_state=42)

[97]: y_pred = rf.predict(X_test)
mae, rmse, r2 = evaluate_model(y_test, y_pred)

[98]: rf_results.append(["Random Forest", size, mae, rmse, r2])

```

Figure 23: Random Forest Regressor

## Output:

100 individual decision trees ensembled to improve accuracy and reduce the risk of overfitting.

```

File Edit View Run Kernel Tabs Settings Help
Coursework_CalorieEstimation + Share
Results comparing
[99]: results = pd.DataFrame(
    lr_results + dt_results + rf_results,
    columns=["Model used for testing", "Test Size", "MAE", "RMSE", "R2 Score"]
)
results
[99]:
  Model used for testing Test Size MAE RMSE R2 Score
0   Linear Regression     0.4  8.301592 19.340058 0.990227
1   Decision Tree         0.4 17.982403 30.678935 0.975409
2   Random Forest         0.4 11.557625 18.951737 0.990616
[ ]:

```

To see the best model

```

[100]: results.sort_values(by="R2 Score", ascending=False)
[100]:
  Model used for testing Test Size MAE RMSE R2 Score
2   Random Forest         0.4 11.557625 18.951737 0.990616
0   Linear Regression     0.4  8.301592 19.340058 0.990227
1   Decision Tree         0.4 17.982403 30.678935 0.975409

```

Figure 24: Result Comparing and Best Model Selection

## Output:

Displays ranking of all the models based on all the test sizes

**Random forest model** shows the highest R2 score

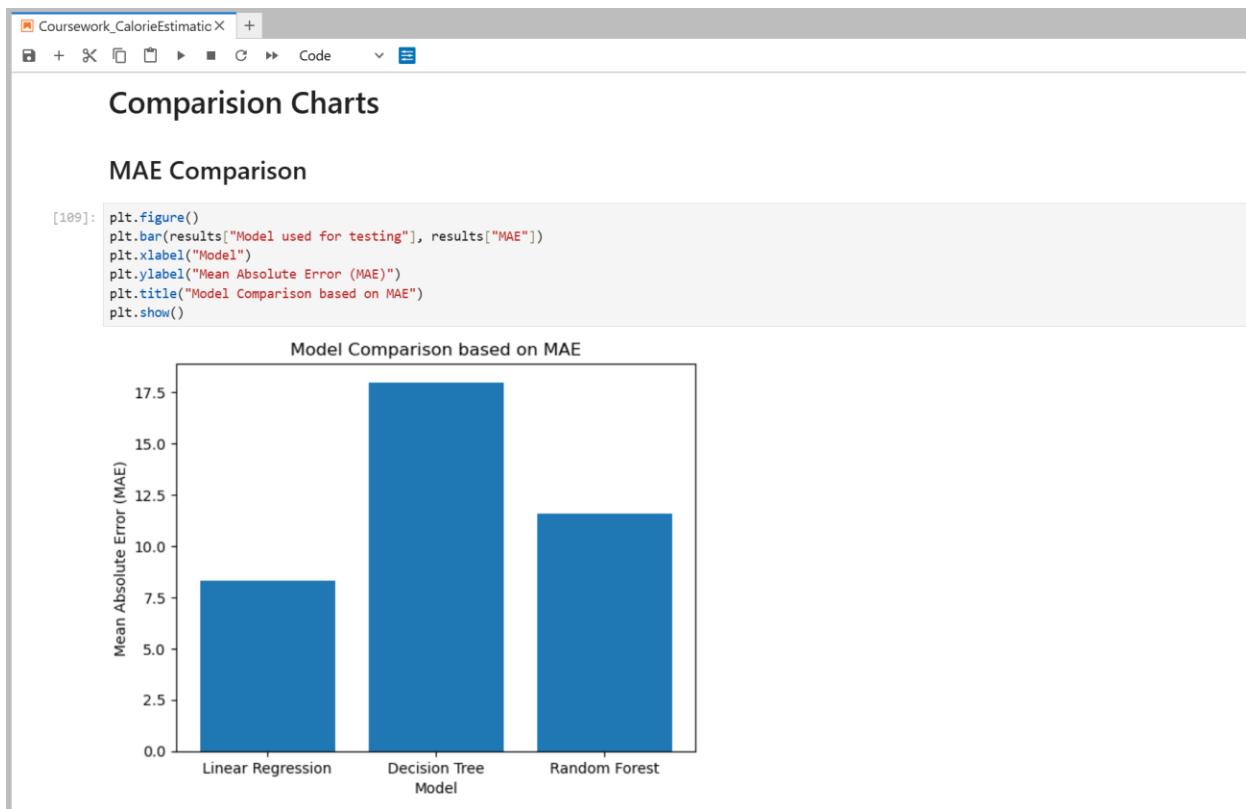


Figure 25: MAE Comparison

### Output:

**MAE value** for all the models compared with the help of a bar chart.

### RSME Comparison

```
[110]: plt.figure()
plt.bar(results["Model used for testing"], results["RMSE"])
plt.xlabel("Model")
plt.ylabel("Root Mean Squared Error (RMSE)")
plt.title("Model Comparison based on RMSE")
plt.show()
```

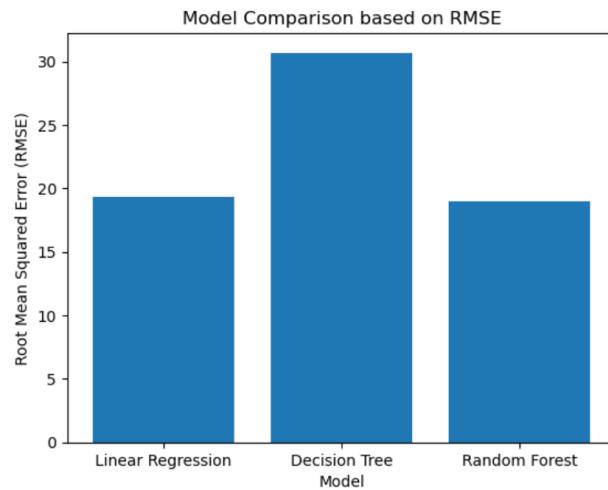


Figure 26: RSME Comparison

### Output:

**RSME value** for all the model compared with the help of a bar chart.

## R2 Comparison

```
[114]: plt.figure()
plt.bar(results["Model used for testing"], results["R2 Score"])
plt.xlabel("Model")
plt.ylabel("R2 Score")
plt.title("Model Comparison based on R2 Score")
plt.show()
```

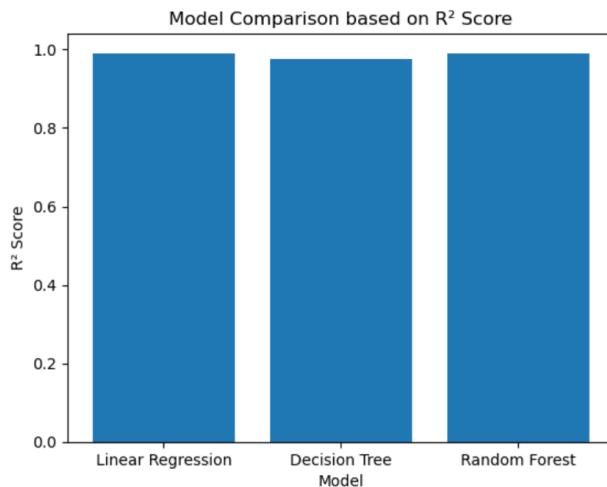


Figure 27: R2 Comparison

## Output:

**R2 score** for all the models compared using a bar chart visualization.

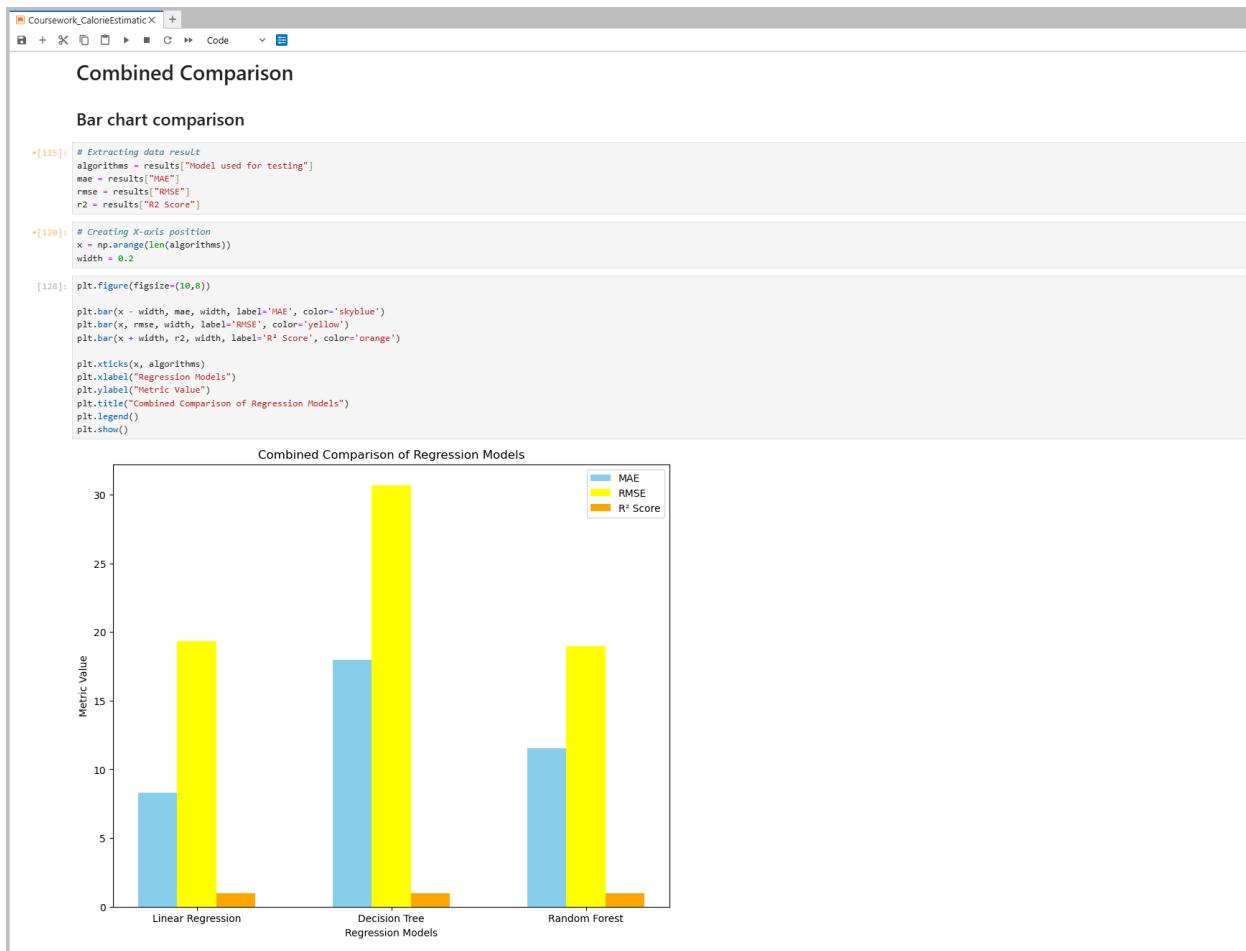


Figure 28: Combined Bar Chart Comparison of all Algorithms

## Output:

Shows multi-dimensional comparison between the models which shows that Decision Tree model is the least accurate model for this dataset.

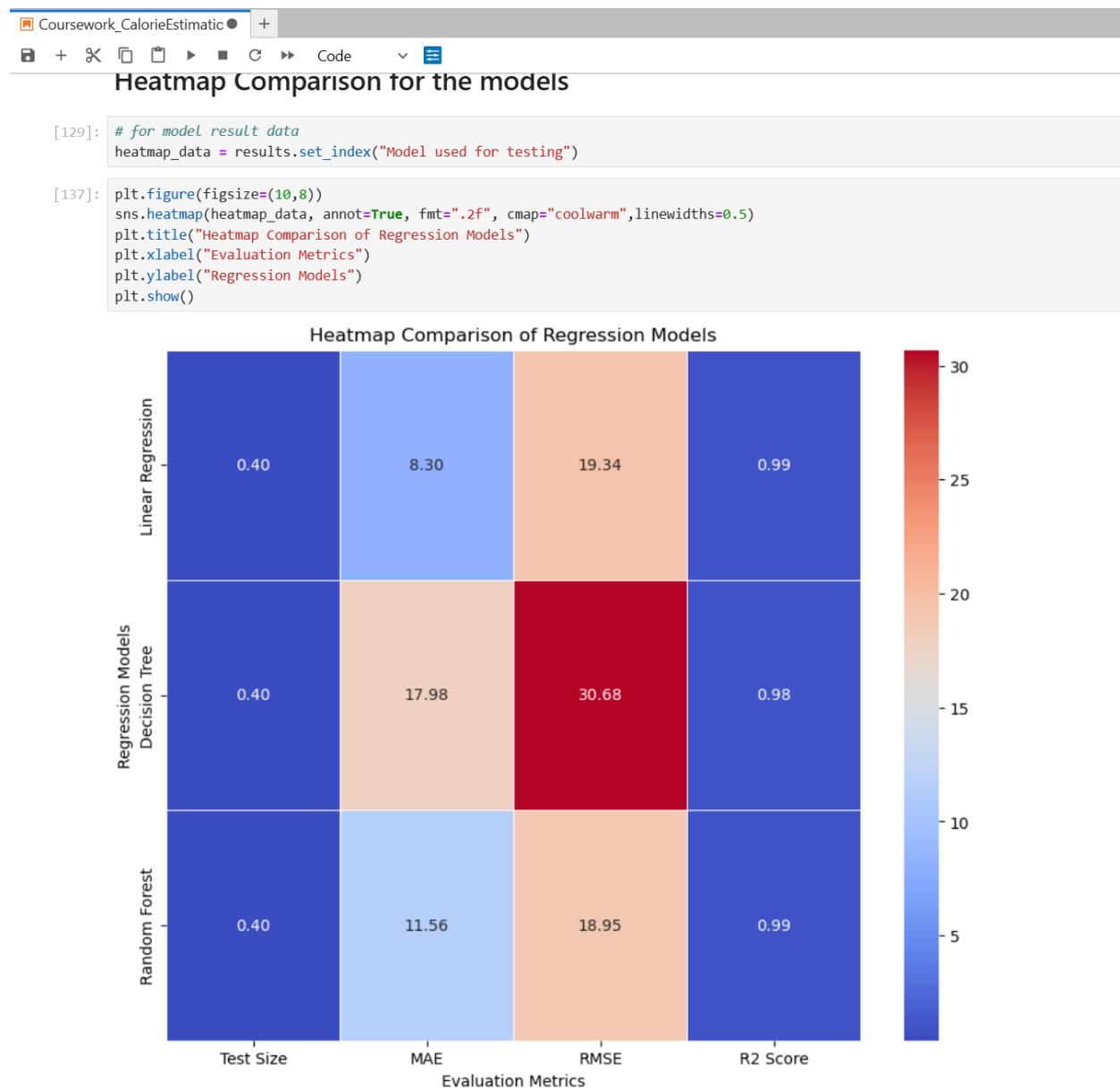


Figure 29: Heatmap Comparison of all Algorithms

### Output:

Displays a dense summary of results, which shows that Decision Tree contains the highest RMSE, while Random Forest and Linear Regression shows high predictive reliability across all metrics.

## 4. Conclusion

In conclusion, this project focused on predicting the calorie content of food items in the dataset using nutritional attributes of the food items. Machine learning regression techniques including Linear Regression, Decision Tree Regressor, and Random Forest Regressor were implemented and evaluated to estimate calorie values.

The dataset was pre-processed by handling missing values. Applying feature scaling, and splitting the data into training and testing sets. Further each model was trained and evaluated using regression performance metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R2 score. Overall, the comparative analysis showed that based models, particularly Random Forest Regressor, provided better prediction performance compared to simpler models due to their ability to capture non-linear relationships between nutritional features and calorie content.

This project provided valuable insight into how machine learning techniques can be effectively applied to nutritional data to estimate calorie content, which has real-world relevance in health monitoring, dietary planning, and food analysis systems. Future improvements could include incorporating larger datasets, additional nutritional attributes, hyperparameter tuning, and cross-validation techniques to further improve prediction accuracy of the system.

## 5. References

1. Antwi, J.B, Tuffour, Mensah, 2022. Machine Learning in Nutrition Research. *Nutrition Journal*, Issue [https://pmc.ncbi.nlm.nih.gov/articles/PMC9776646/](https://PMC9776646/).
2. Changhe Yang, 2024. Predicting Nutrient Density in Foods Using Machine Learning Models. *DAMI 2024*, p. 6.
3. Zheng J, Wang J, Shen J, An R , 2023. Artificial Intelligence Applications to Measure Food and Nutrient Intakes. *Nutrition Journal*, Issue <https://www.jmir.org/2024/1/e54557/>.