Module Code & Module Title

CU6051NI Artificial Intelligence

75% Individual Coursework

Submission: Final Submission

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: 21/01/2026

Assignment Submission Date: 21/01/2026

Submitted To: Er. Roshan Shrestha

GitHub Link	*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 3
2. Background 4
 2.1 Research Work Done on Calories Prediction 4
 2.2 Existing Research Work 5
 2.3 Dataset Information and Background 8
3. Solution 9
 3.1 Proposed Solution 9
 3.2 Algorithms Used: 10
 3.3 Pseudocode of the solution 11
  3.3.1 Pseudocode for Linear Regression 13
  3.3.2 Pseudocode for Decision Tree Regressor 14
  3.3.3 Pseudocode for Random Forest Regressor 15
 3.4 Diagrammatical Representation 16
  3.4.1 Combined Flowchart 16
  3.4.2 Linear Regression Flowchart 17
  3.4.3 Decision Tree Regressor Flowchart 18
  3.4.4 Random Forest Regressor Flowchart 19
 3.5 Development Process 20
  3.5.1 Loading the Dataset 21
  3.5.2 Checking Missing Values (NaN) 22
  3.5.3 Data Structure 23
  3.5.4 Dataset Information 24
  3.5.5 Dropping Invalid Columns 25
  3.5.6 Data Visualization 26
  3.5.7 Features Scaling 28
  3.5.8 Input Features(X) 29
  3.5.9 Target Variable(y) 29
  3.5.10 Train_test_split with 80% training data and 20% testing data 30

3.6 Achieved Results 32
3.6.1 Algorithm testing 32
3.6.2 Test Sizes 32
3.6.3 Linear Regression Model Testing 33
3.6.4 Decision Tree Regressor Model Testing 34
3.6.5 Random Forest Regressor Model Testing 35
3.6.6 Result Comparing for 3 models 36
3.6.7 Comparison Chart for 3 models 37
3.6.8 Combined Comparison 40
3.6.9 Overfitting Test for Linear Regression 42
3.6.10 Overfitting Test for Decision Tree Regressor 42
3.6.11 Overfitting Test for Random Forest Regressor 44
3.6.12 Calorie Prediction Function Using Random Forest Algorithm 45
3.6.13 Calories Prediction with Random Forest Regressor: 46
3.6.14 Comparison of Calorie Estimation using all models 47
3.6.15 Prediction Comparison of all 3 models 48
4. Conclusion 49
5. References 50

Table of Figures:

Figure 1: Artificial Intelligence, Machine Learning, and Deep Learning 1
Figure 2: Article 1 5
Figure 3: Article 2 6
Figure 4: Article 3 7
Figure 5: Combined Flowchart 16
Figure 6: Linear Regression Flowchart 17
Figure 7: Decision Tree Regressor Flowchart 18
Figure 8: Random Forest Regressor Flowchart 19
Figure 9: Loading Dataset in Jupyter notebook 21
Figure 10: Checking for Missing NaN values 22
Figure 11: Checking Dataset Shape and Datatypes 23
Figure 12: Statistical Summary of Numerical Columns 24
Figure 13: Dropping Invalid Columns 25
Figure 14: Calorie Distribution Chart 26
Figure 15: Calorie Distribution Heatmap 27
Figure 16: Feature Scaling for Linear Regression 28
Figure 17: Input Feature(X) for train_test_split 29
Figure 18: Target Variable(y) for train_test_split 29
Figure 19: Train_test_split 30
Figure 20: Imports for algorithm testing 32
Figure 21: Test sizes and Evaluation 32
Figure 22: Linear Regression Test 33
Figure 23: Decision Tree Regressor Test 34
Figure 24: Random Forest Regressor Test 35
Figure 25: Result Comparing and Best Model Selection 36
Figure 26: MAE Comparison 37
Figure 27: RSME Comparison 38
Figure 28: R2 Comparison 39
Figure 29: Combined Bar Chart Comparison of all Algorithms 40
Figure 30: Heatmap Comparison of all Algorithms 41
Figure 31: Overfitting test for Linear Regression 42
Figure 32: Overfitting test for Decision Tree Regressor 42
Figure 33: Hyperparameter tuning for Decision Tree 43
Figure 34: Overfitting test for Random Forest Regressor 44
Figure 35: Calorie Estimation Function 45
Figure 36: Prediction 1 46
Figure 37: Prediction 2 46
Figure 38: Prediction 3 46
Figure 39: Calorie Prediction function for all 3 models combined 47
Figure 40: Prediction 4: Comparison of all models 48

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.

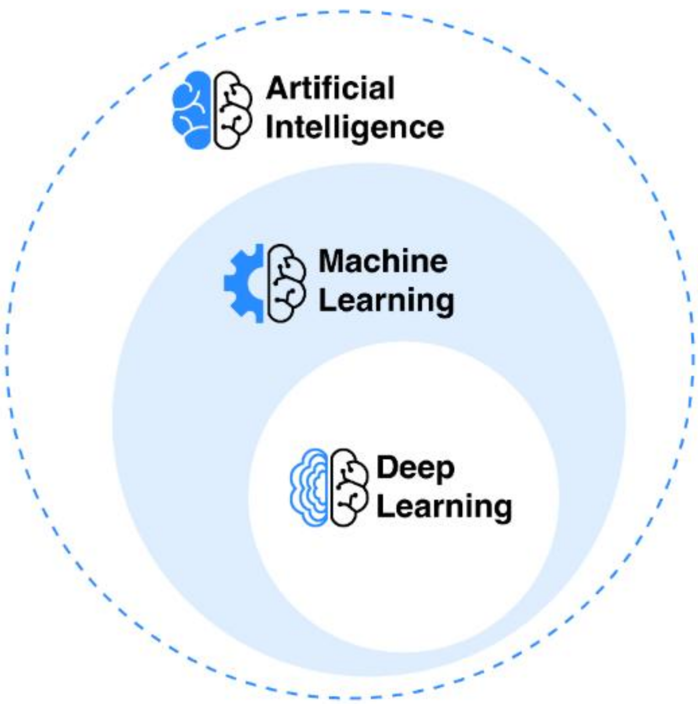


Figure 1: Artificial Intelligence, Machine Learning, and Deep Learning

Machine Learning problems are mainly categorized into the following categories:

- Supervised learning
- Machine learning technique that learns the relationship between input (X variables) and output (y variables). Learns patterns and relationship between input and output data with the use of labelled data (Ali, 2022).
- Unsupervised learning

  Machine learning technique that finds patterns and relationships within data on its own as labelled data is not present as it uses unlabelled data, meaning it gets no instructions (Grammerly, 2024).
- Reinforcement learning
- Machine learning method where decisions are made using interaction with an environment through trial and error, aiming to maximize rewards.

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:

1. Linear Regression

2. Decision Tree Regressor

3. Random Forest Regressor

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.

Calories are traditionally calculated with the help of a formula which is known as the 4-9-4 system that refers that protein and carbohydrate each contains 4 calories per gram and fats contains 9 calories per gram (Harvey, 2022).

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.

2. Background

2.1 Research Work Done on Calories Prediction

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 metrices 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:

2.2 Existing Research Work

1) Article 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 (Antwl, J.B, Tuffour, Mensah, 2022).

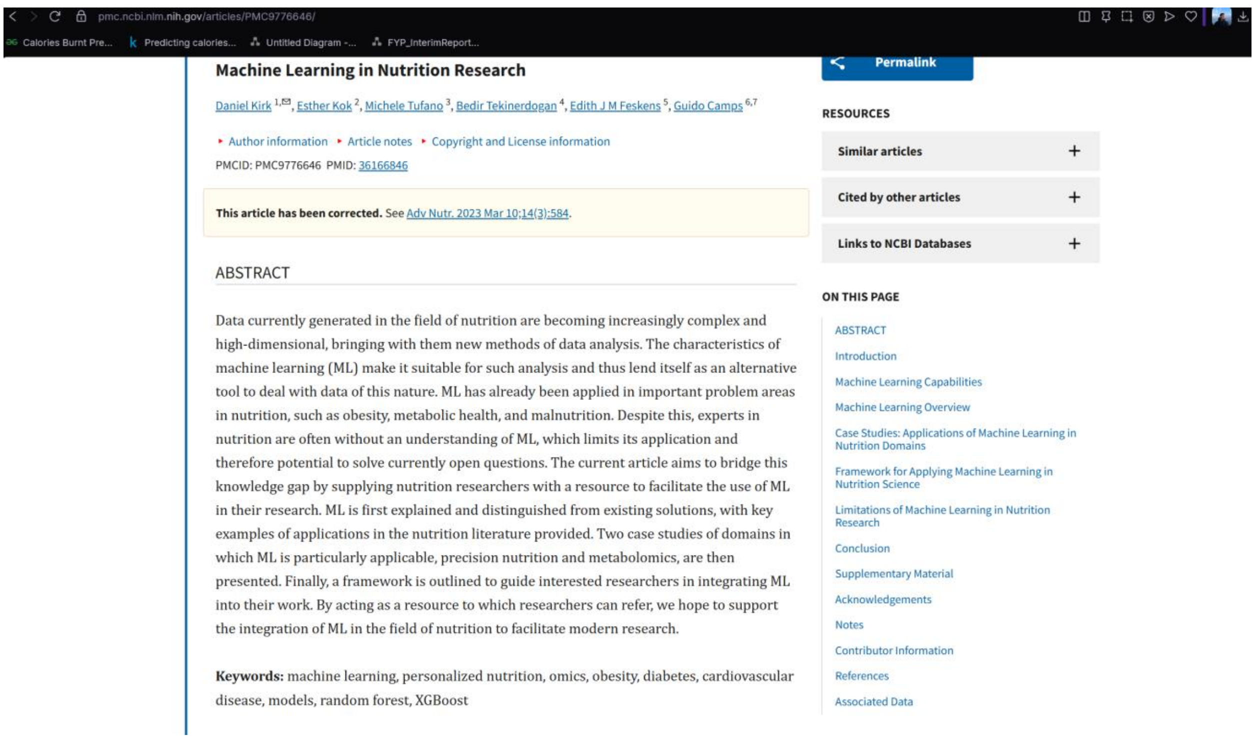


Figure 2: Article 1

2) Article 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).

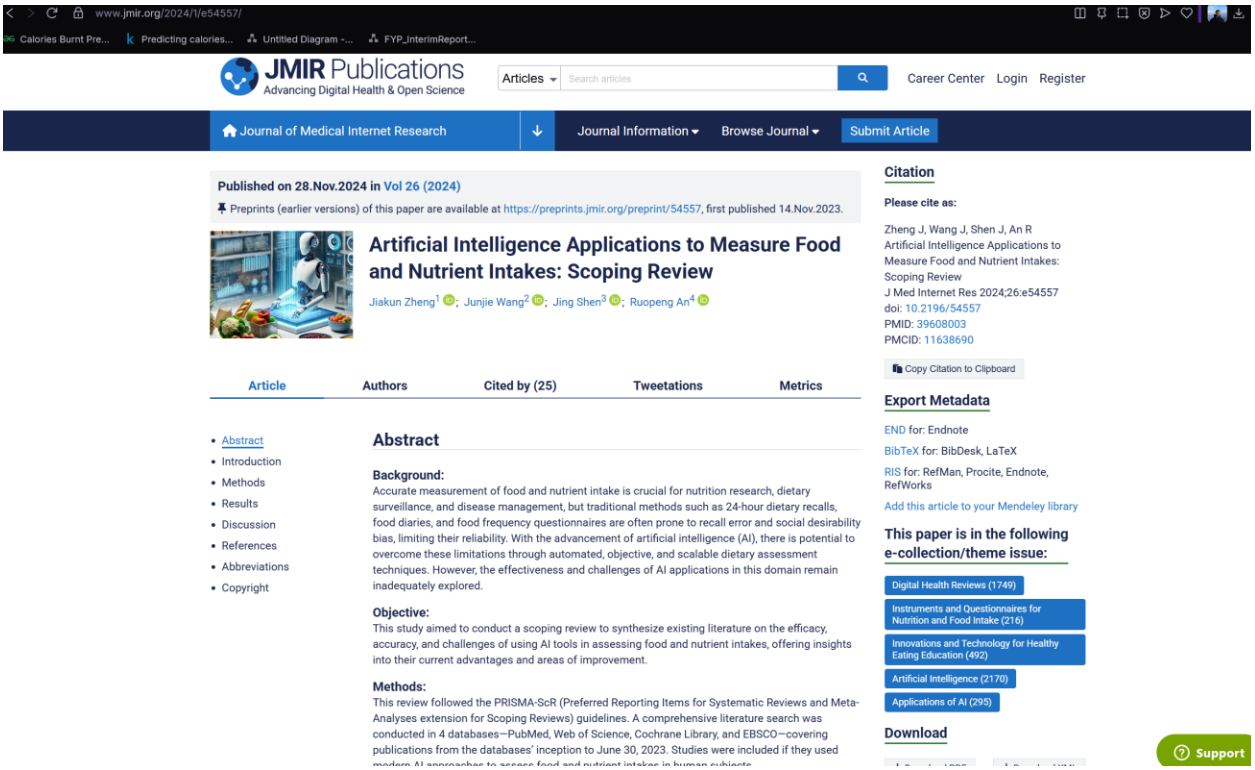
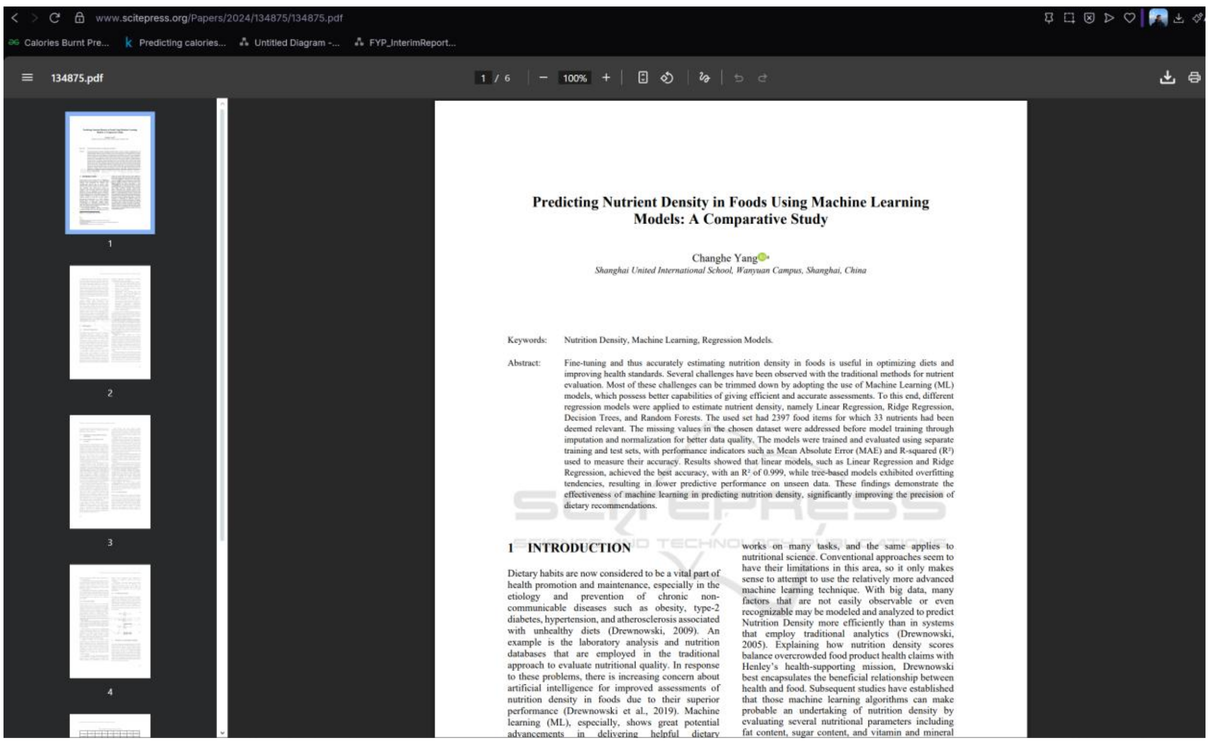


Figure 3: Article 2

3) Article 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 R2 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).



Predicting Nutrient Density in Foods Using Machine Learning Models: A Comparative Study

Changhe Yang

Shanghai United International School, Wanyuan Campus, Shanghai, China

Keywords: Nutrition Density, Machine Learning, Regression Models.

Abstract: Fine-tuning and thus accurately estimating nutrition density in foods is useful in optimizing diets and improving health standards. Several challenges have been observed with the traditional methods for nutrient evaluation. Most of these challenges can be trimmed down by adopting the use of Machine Learning (ML) models, which possess better capabilities of giving efficient and accurate assessments. To this end, different regression models were applied to estimate nutrient density, namely Linear Regression, Ridge Regression, Decision Trees, and Random Forests. The used set had 2397 food items for which 33 nutrients had been deemed relevant. The missing values in the chosen dataset were addressed before model training through imputation and normalization for better data quality. The models were trained and evaluated using separate training and test sets, with performance indicators such as Mean Absolute Error (MAE) and R-squared (R^2) used to measure their accuracy. Results showed that linear models, such as Linear Regression and Ridge Regression, achieved the best accuracy, with an R^2 of 0.999, while tree-based models exhibited overfitting tendencies, resulting in lower predictive performance on unseen data. These findings demonstrate the effectiveness of machine learning in predicting nutrition density, significantly improving the precision of dietary recommendations.

1 INTRODUCTION

Dietary habits are now considered to be a vital part of health promotion and maintenance, especially in the etiology and prevention of chronic non-communicable diseases such as obesity, type-2 diabetes, hypertension, and atherosclerosis associated with unhealthy diets (Drewnowski, 2009). An example is the laboratory analysis and nutrition databases that are employed in the traditional approach to evaluate nutritional quality. In response to these problems, there is increasing concern about artificial intelligence for improved assessments of nutrition density in foods due to their superior performance (Drewnowski et al., 2019). Machine learning (ML), especially, shows great potential advancements in delivering helpful dietary

works on many tasks, and the same applies to nutritional science. Conventional approaches seem to have their limitations in this area, so it only makes sense to attempt to use the relatively more advanced machine learning technique. With big data, many factors that are not easily observable or even recognizable may be modeled and analyzed to predict Nutrition Density more efficiently than in systems that employ traditional analytics (Drewnowski, 2005). Explaining how nutrition density scores balance overcrowded food product health claims with Henley's health-supporting mission, Drewnowski best encapsulates the beneficial relationship between health and food. Subsequent studies have established that those machine learning algorithms can make probable an undertaking of nutrition density by evaluating several nutritional parameters including fat content, sugar content, and vitamin and mineral

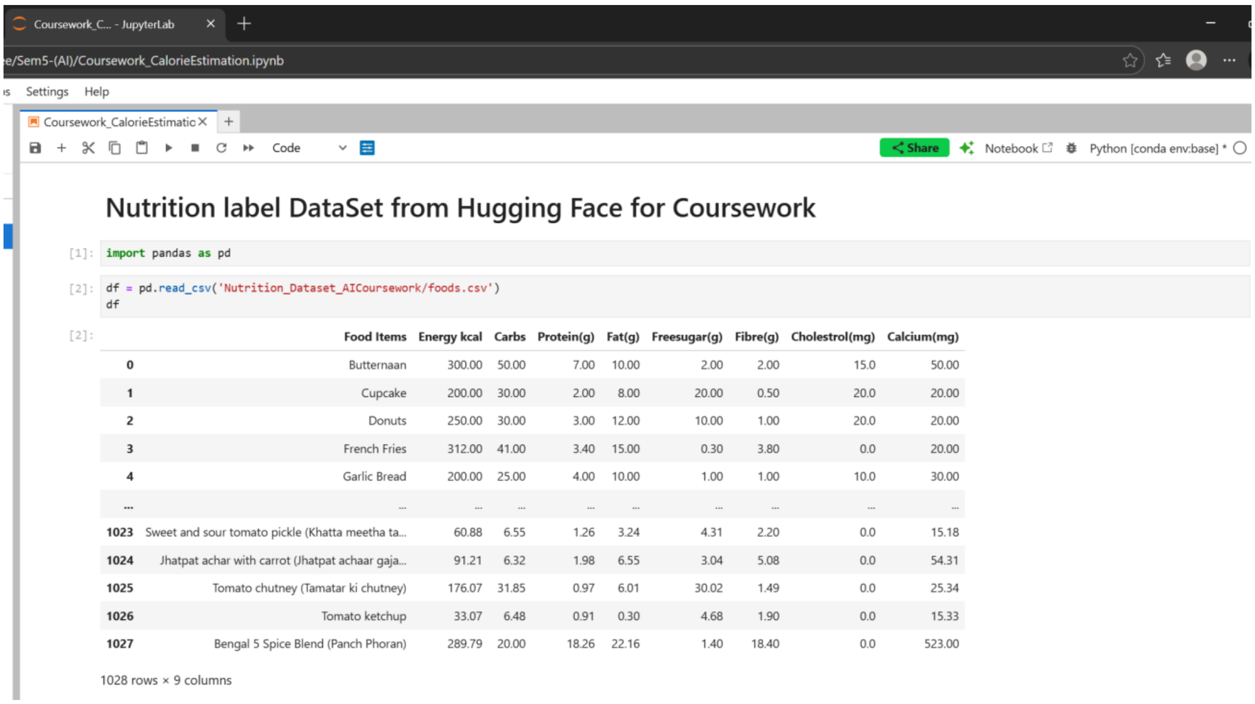
Figure 4: Article 3

2.3 Dataset Information and Background

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

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

The dataset contains 1028 rows and 9 columns with the following attributes:



Nutrition label DataSet from Hugging Face for Coursework

```
[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)	Cholestrol(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

1. Food items
2. Energy Kcal
3. Carbs
4. Protein(g)
5. Fat(g)
6. Freesugar(g)
7. Fibre(g)
8. Cholestrol(mg)
9. Calcium(mg)

The dataset contains both input values X and target values y.

3. Solution

3.1 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 Algorithms Used:

a) Linear Regression

- It is a model that estimates the relationship between a dependent target variable and one or more independent predictor variable by fitting a straight line that shows the relationship. It's a fundamental statistical method to define linear relationship between dependent and independent variables (Kavita, 2025).

b) Decision Tree Regressor

- Supervised machine learning algorithm that work by recursively splitting data into branches based on features values until reaching a prediction at leaf node. Model uses a tree-like structure here each internal node represents a feature, each branch represents a decision rule, and each leaf node represents the outcome making this algorithm easy to visualize and interpret, making it an excellent tool for explaining the logic behind predictions (Jain, 2024).

c) Random Forest Regressor

- An ensemble machine learning method that builds multiple decision trees during training and then combines their predictions for regression which reduces overfitting and improves prediction accuracy making this algorithm one of the most powerful and popular algorithm used to create machine learning models (Snowflake, Inc, n.d.).

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

3.3 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

3.3.1 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

INTIALIZE 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

3.3.2 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

INTIALIZE 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

3.3.3 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

INTIALIZE 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

3.4 Diagrammatical Representation

3.4.1 Combined Flowchart

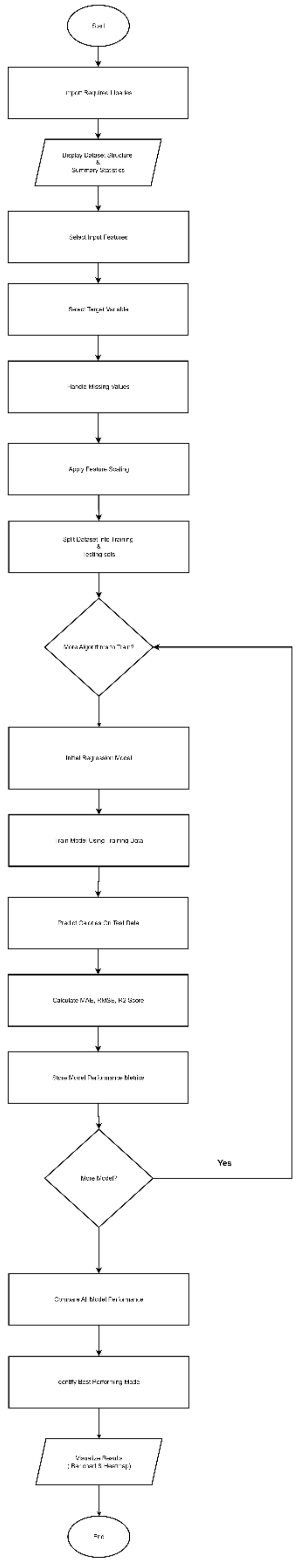


Figure 5: Combined Flowchart

3.4.2 Linear Regression Flowchart

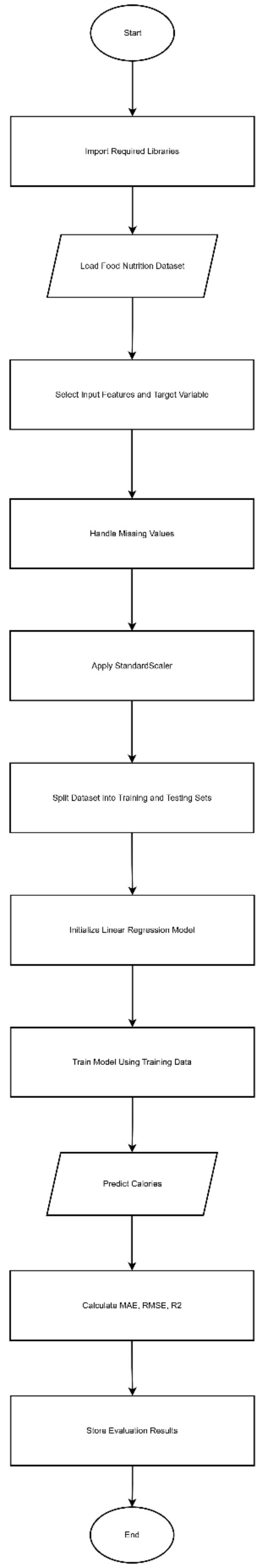


Figure 6: Linear Regression Flowchart

3.4.3 Decision Tree Regressor Flowchart

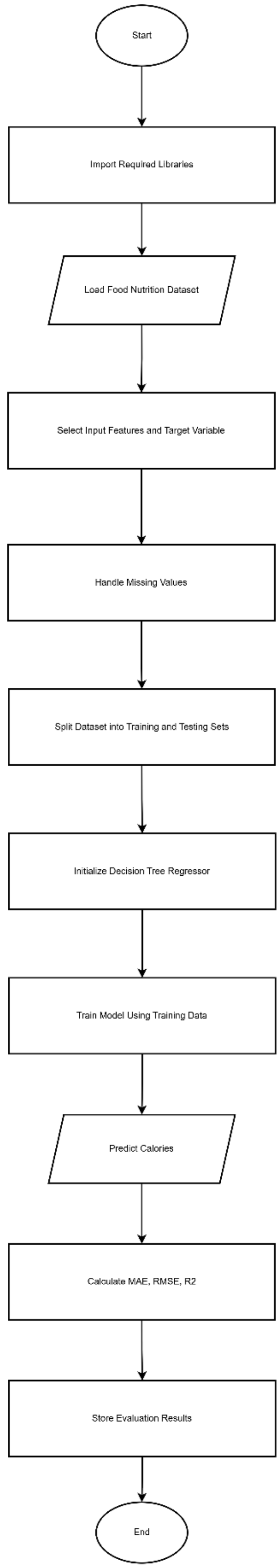


Figure 7: Decision Tree Regressor Flowchart

3.4.4 Random Forest Regressor Flowchart

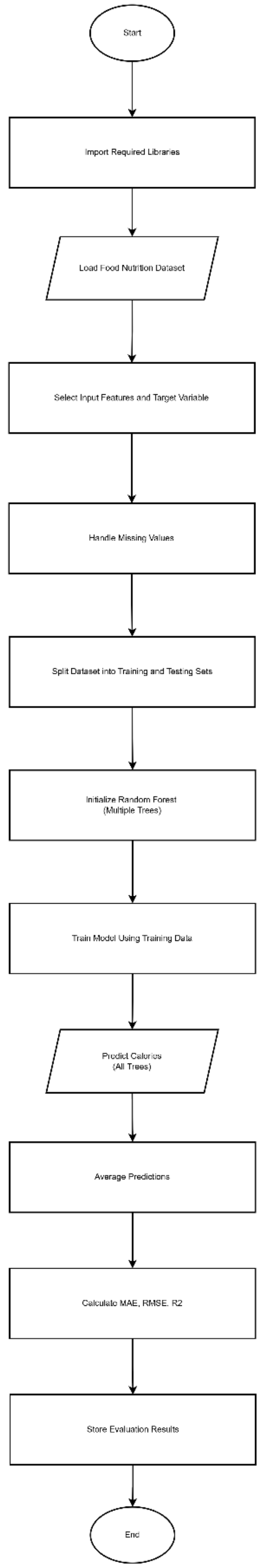


Figure 8: Random Forest Regressor Flowchart

3.5 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.

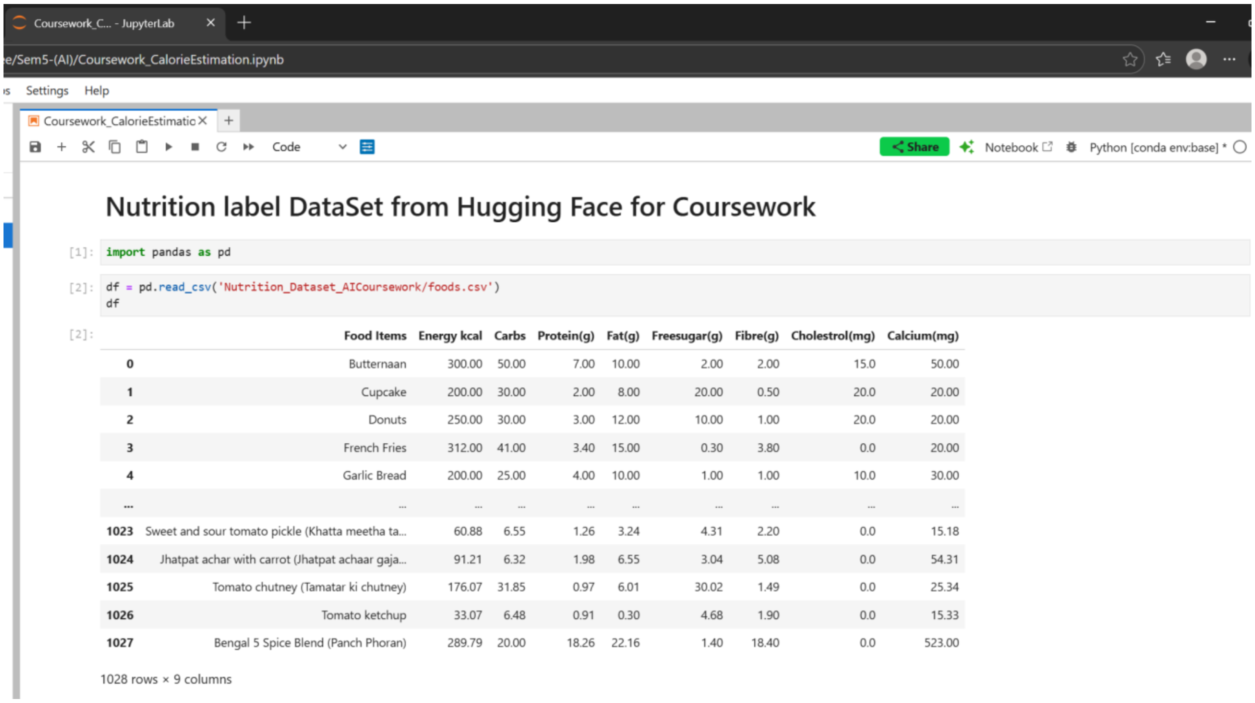
The development process began with data preparation, followed by model implementation and evaluation and the following key libraries were used during the development process:

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

Tools used:

1. Programming Language: Python
2. Computing Environment: Jupyter Notebook
3. Version Control: Github

3.5.1 Loading the Dataset



Nutrition label DataSet from Hugging Face for Coursework

```
[1]: import pandas as pd
```

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

[2]:

	Food Items	Energy kcal	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)	Cholestrol(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 9: 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

3.5.2 Checking Missing Values (NaN)

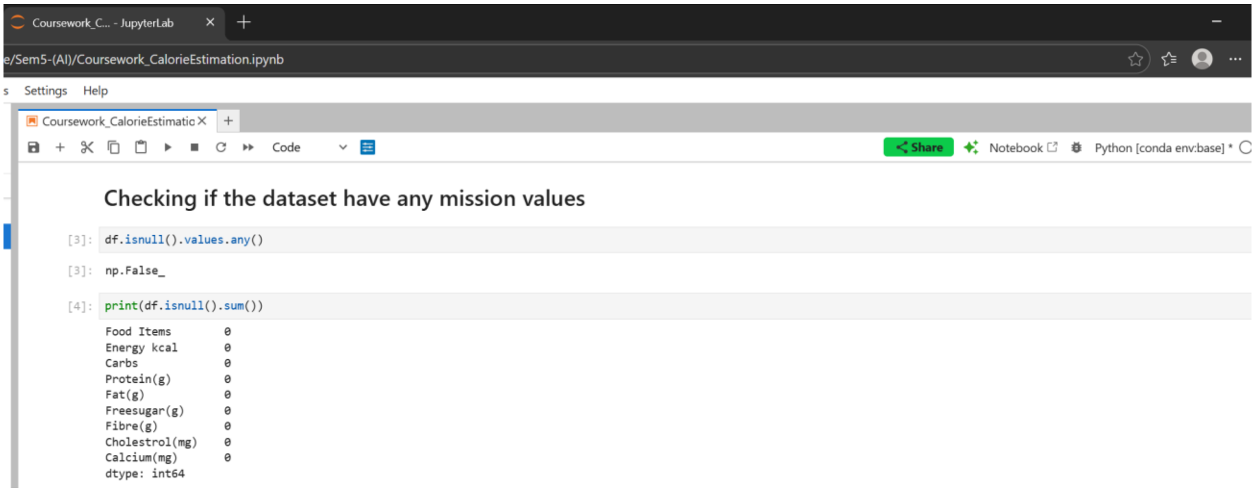


Figure 10: Checking for Missing NaN values

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

3.5.3 Data Structure

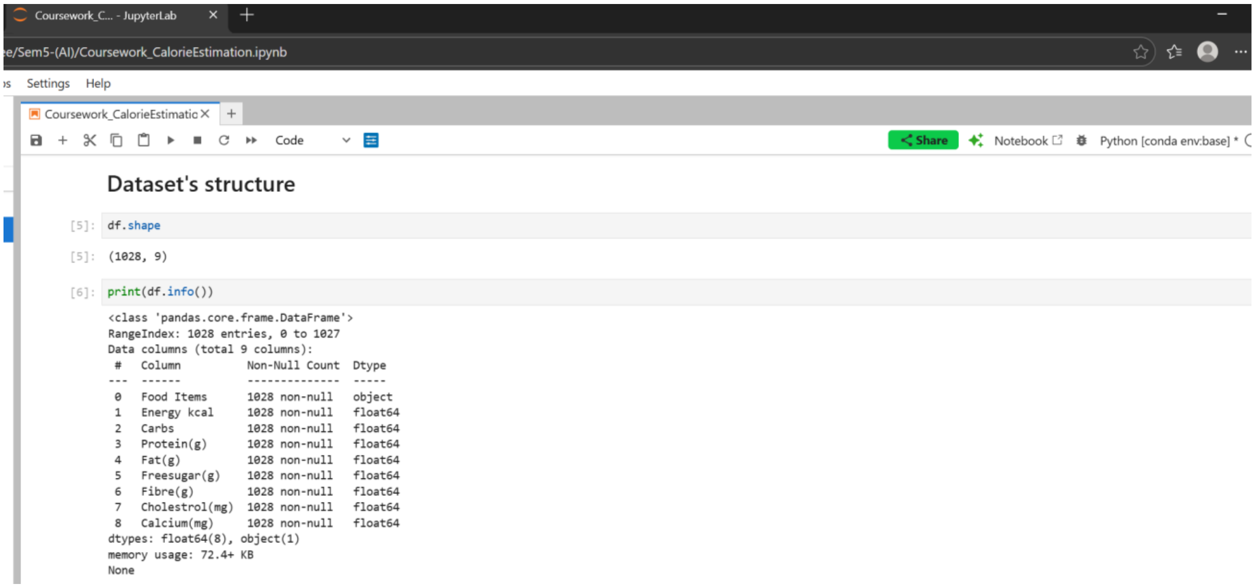


Figure 11: Checking Dataset Shape and Datatypes

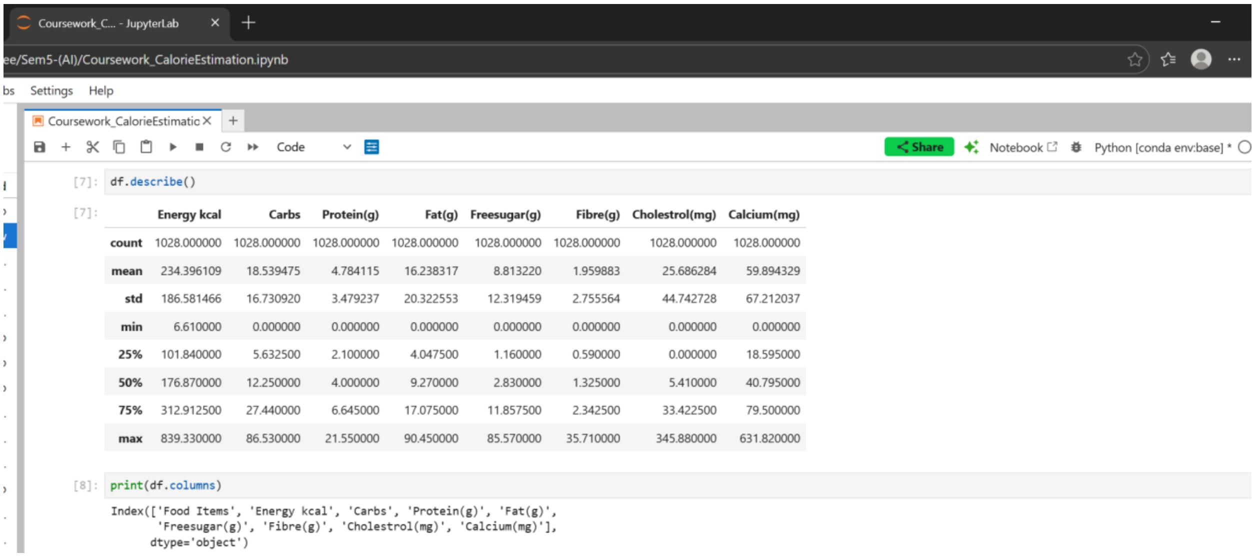
df.shape to check the rows and columns of the dataset

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)

3.5.4 Dataset Information




```
[7]: df.describe()
```

	Energy kcal	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)	Cholestrol(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)', 'Cholestrol(mg)', 'Calcium(mg)'],
      dtype='object')
```

Figure 12: Statistical Summary of Numerical Columns

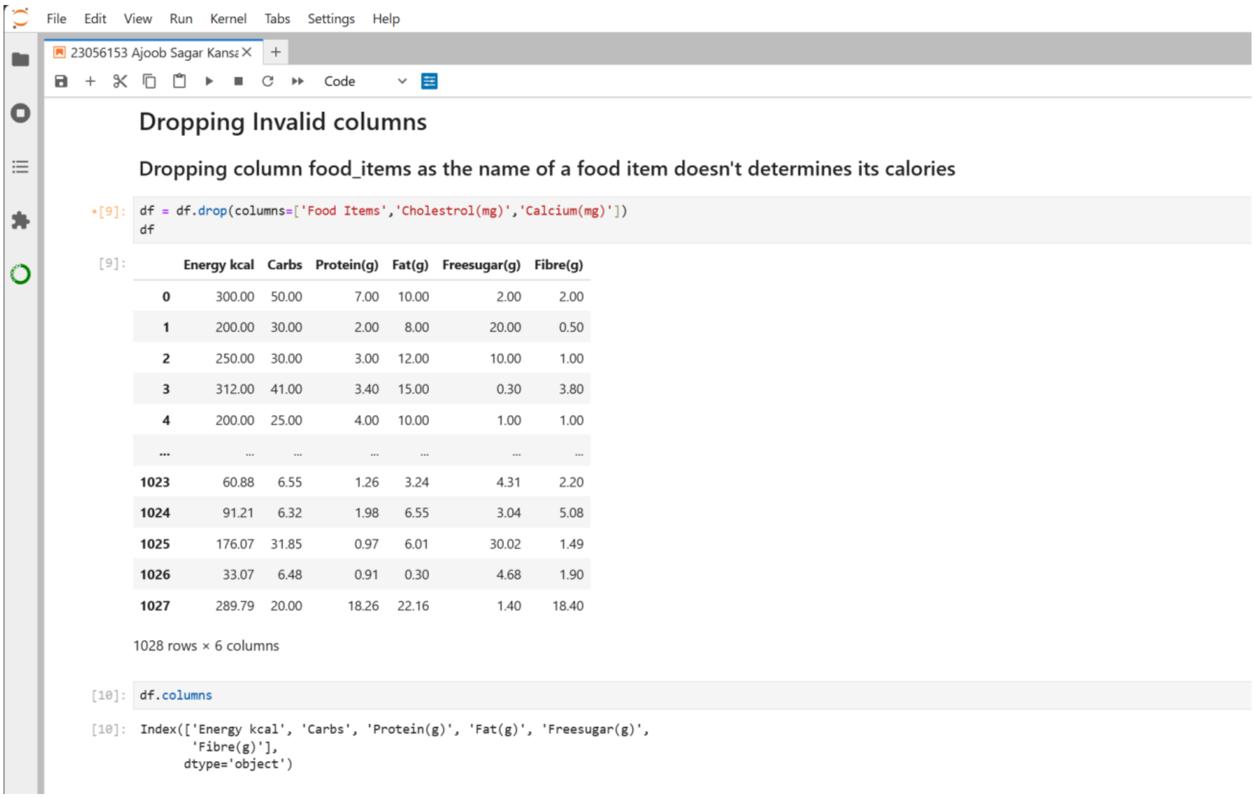
df.describe() generates statistics that shows central tendency, dispersion, and shape of the dataset’s distribution

df.columns lists all the columns of the dataset

Output:

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

3.5.5 Dropping Invalid Columns


File Edit View Run Kernel Tabs Settings Help

23056153 Ajoob Sagar Kans

Dropping Invalid columns

Dropping column food_items as the name of a food item doesn't determines its calories

```
[9]: df = df.drop(columns=['Food Items','Cholestrol(mg)','Calcium(mg)'])
df
```

[9]:

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

1028 rows × 6 columns

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

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

Figure 13: Dropping Invalid Columns

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

df.columns shows all the currently present columns in the dataset

Output:

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

(Cholesterol) and (Calcium) columns were dropped as it does not majorly effect calorie content of a food.

3.5.6 Data Visualization

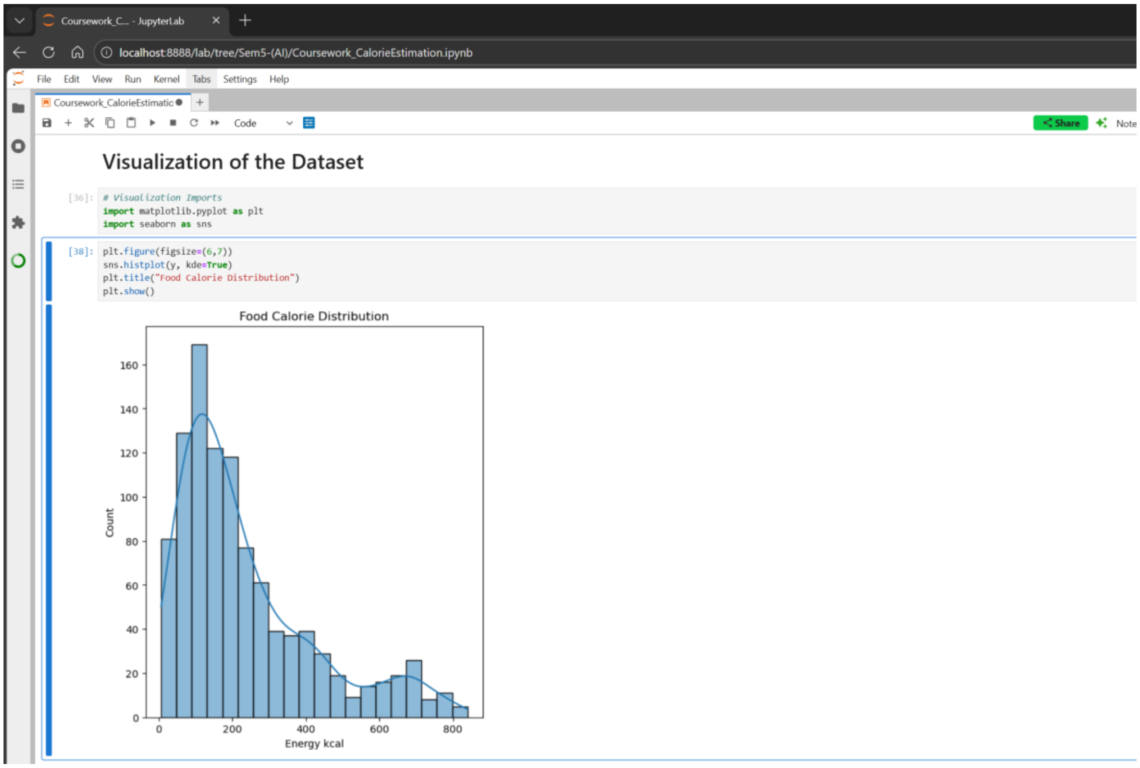


Figure 14: Calorie Distribution Chart

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).

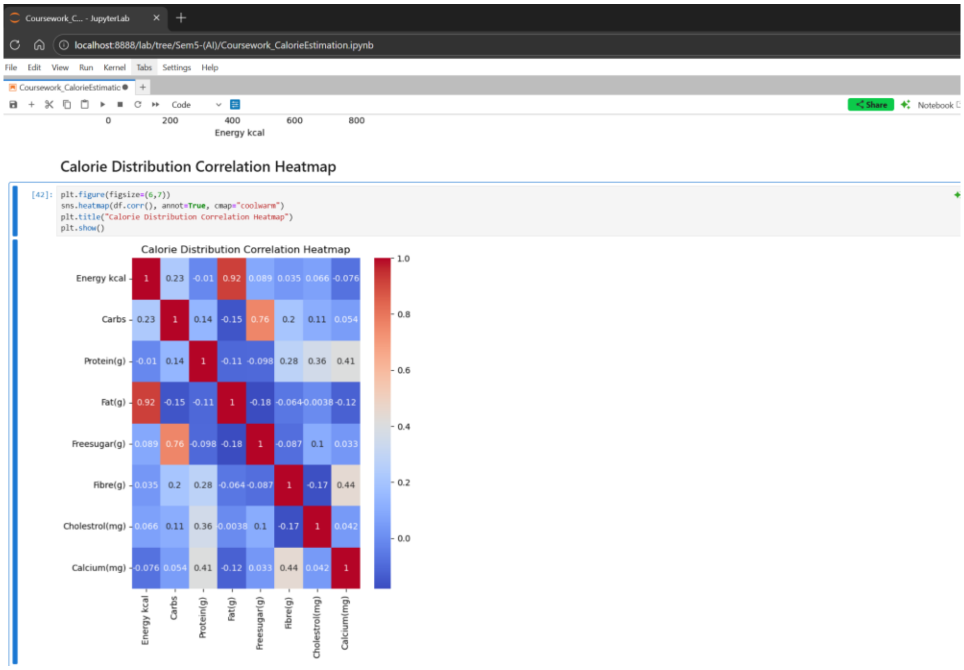


Figure 15: Calorie Distribution Heatmap

df.corr() to calculate the correlation matrix

sns.heatmap to visualize the correlation matrix

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.

3.5.7 Features Scaling

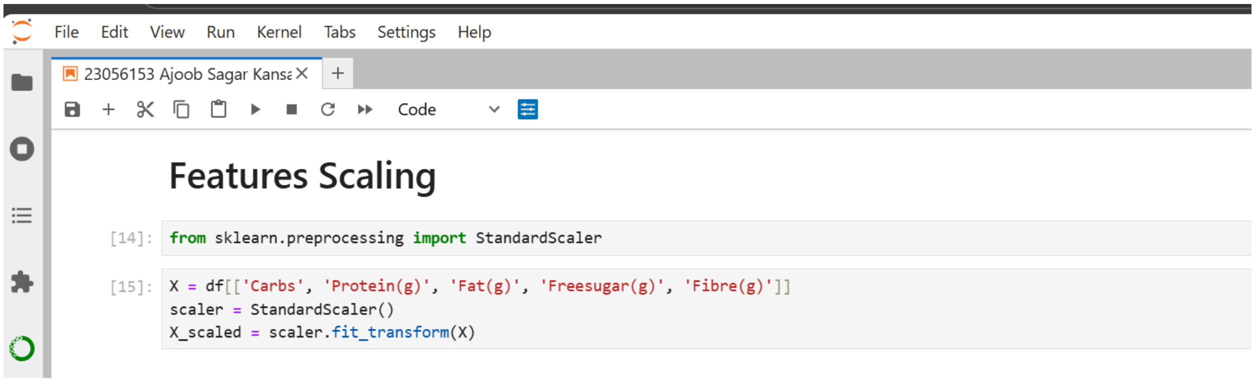


Figure 16: Feature Scaling for Linear Regression

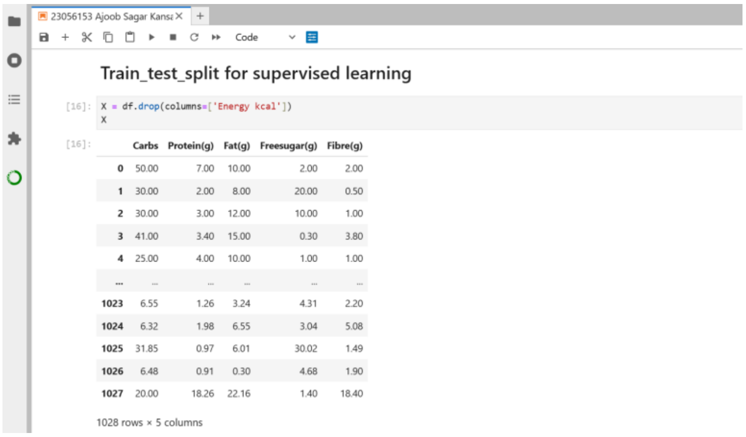
StandardScaler library from Scikit-learn used to standardised different measuring scales to transform the data to mean of 0 and a standard deviation of 1.

Output:

With the help of StandardScalar the input features are normalized so that all the features have a mean of 0 and standard deviation of 1.

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

3.5.8 Input Features(X)



23056153 Ajoob Sagar Kans

Train_test_split for supervised learning

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

[16]:

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

1028 rows × 5 columns

Figure 17: 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.

3.5.9 Target Variable(y)

23056153 Ajoob Sagar Kans

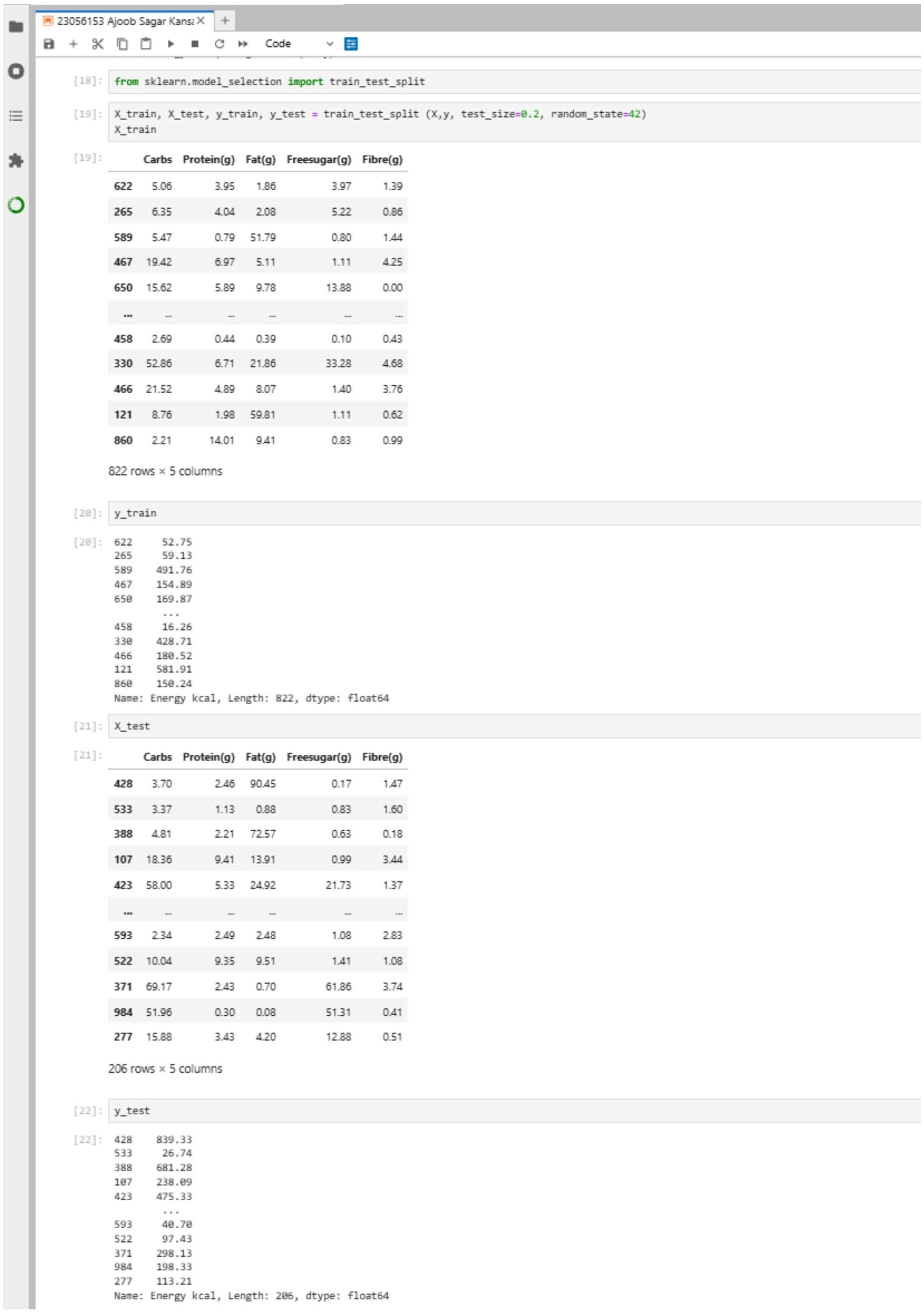
```
[17]: y = df['Energy kcal']
      y

[17]: 0       300.00
      1       200.00
      2       250.00
      3       312.00
      4       200.00
               ...
      1023     60.88
      1024     91.21
      1025    176.07
      1026     33.07
      1027    289.79
      Name: Energy kcal, Length: 1028, dtype: float64
```

Figure 18: Target Variable(y) for train_test_split

Output:

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

3.5.10 Train_test_split with 80% training data and 20% testing data


23056153 Ajoob Sagar Kans:

```
[18]: from sklearn.model_selection import train_test_split
```

```
[19]: X_train, X_test, y_train, y_test = train_test_split (X,y, test_size=0.2, random_state=42)
X_train
```

[19]:

	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)
622	5.06	3.95	1.86	3.97	1.39
265	6.35	4.04	2.08	5.22	0.86
589	5.47	0.79	51.79	0.80	1.44
467	19.42	6.97	5.11	1.11	4.25
650	15.62	5.89	9.78	13.88	0.00
...	...	...	...	...	...
458	2.69	0.44	0.39	0.10	0.43
330	52.86	6.71	21.86	33.28	4.68
466	21.52	4.89	8.07	1.40	3.76
121	8.76	1.98	59.81	1.11	0.62
860	2.21	14.01	9.41	0.83	0.99

822 rows × 5 columns

```
[20]: y_train
```

```
[20]: 622     52.75
265     59.13
589    491.76
467    154.89
650    169.87
        ...
458     16.26
330    428.71
466    180.52
121    581.91
860    150.24
Name: Energy kcal, Length: 822, dtype: float64
```

```
[21]: X_test
```

[21]:

	Carbs	Protein(g)	Fat(g)	Freesugar(g)	Fibre(g)
428	3.70	2.46	90.45	0.17	1.47
533	3.37	1.13	0.88	0.83	1.60
388	4.81	2.21	72.57	0.63	0.18
107	18.36	9.41	13.91	0.99	3.44
423	58.00	5.33	24.92	21.73	1.37
...	...	...	...	...	...
593	2.34	2.49	2.48	1.08	2.83
522	10.04	9.35	9.51	1.41	1.08
371	69.17	2.43	0.70	61.86	3.74
984	51.96	0.30	0.08	51.31	0.41
277	15.88	3.43	4.20	12.88	0.51

206 rows × 5 columns

```
[22]: y_test
```

```
[22]: 428    839.33
533     26.74
388    681.28
107    238.09
423    475.33
        ...
593     40.70
522     97.43
371    298.13
984    198.33
277    113.21
Name: Energy kcal, Length: 206, dtype: float64
```

Figure 19: 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.

3.6 Achieved Results

3.6.1 Algorithm testing

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 20: Imports for algorithm testing

Output:

Imports for all the libraries used

3.6.2 Test Sizes

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 21: 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

3.6.3 Linear Regression Model Testing

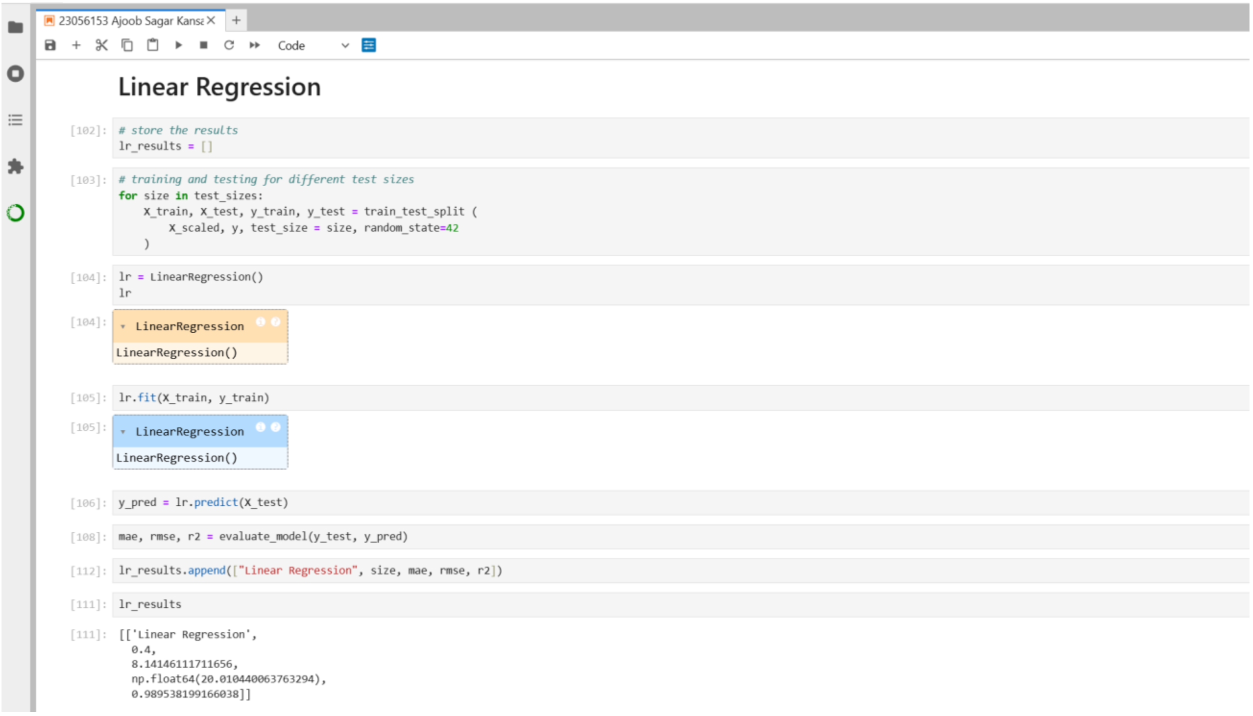


Figure 22: Linear Regression Test

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

3.6.4 Decision Tree Regressor Model Testing

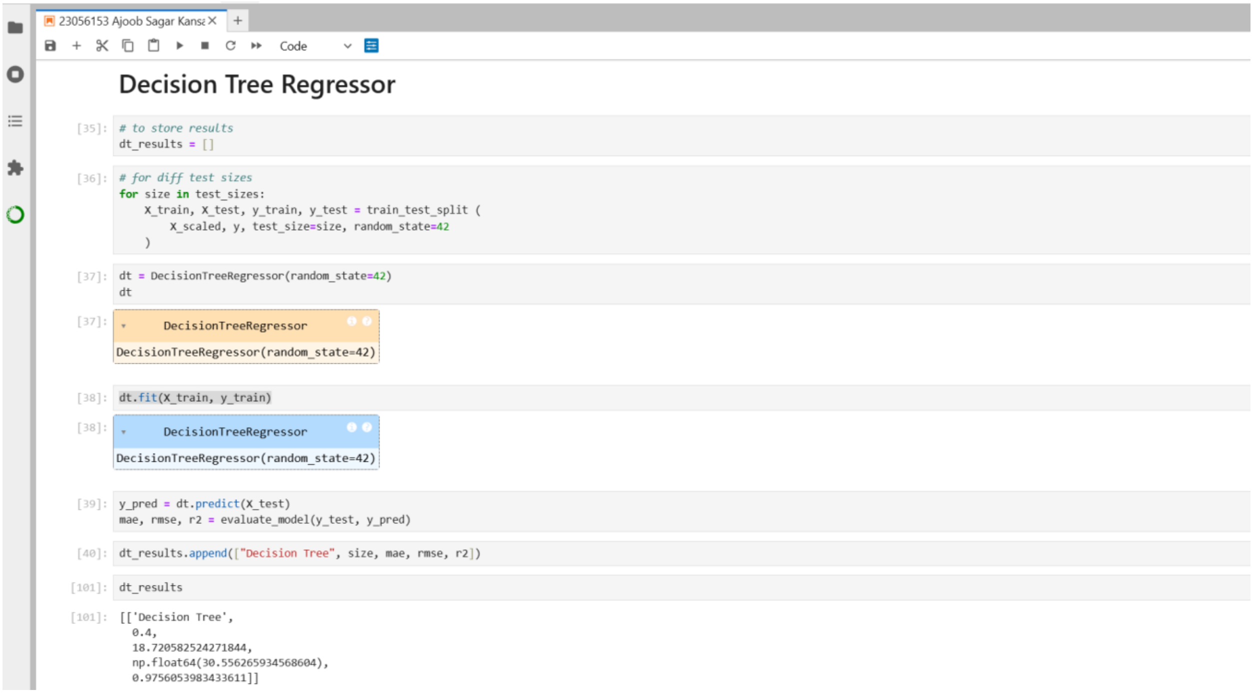


Figure 23: Decision Tree Regressor Test

Output:

Non-linear relationships are shown using Decision Tree Regressor

Displays complexity of a single tree compared to the linear baseline

3.6.5 Random Forest Regressor Model Testing

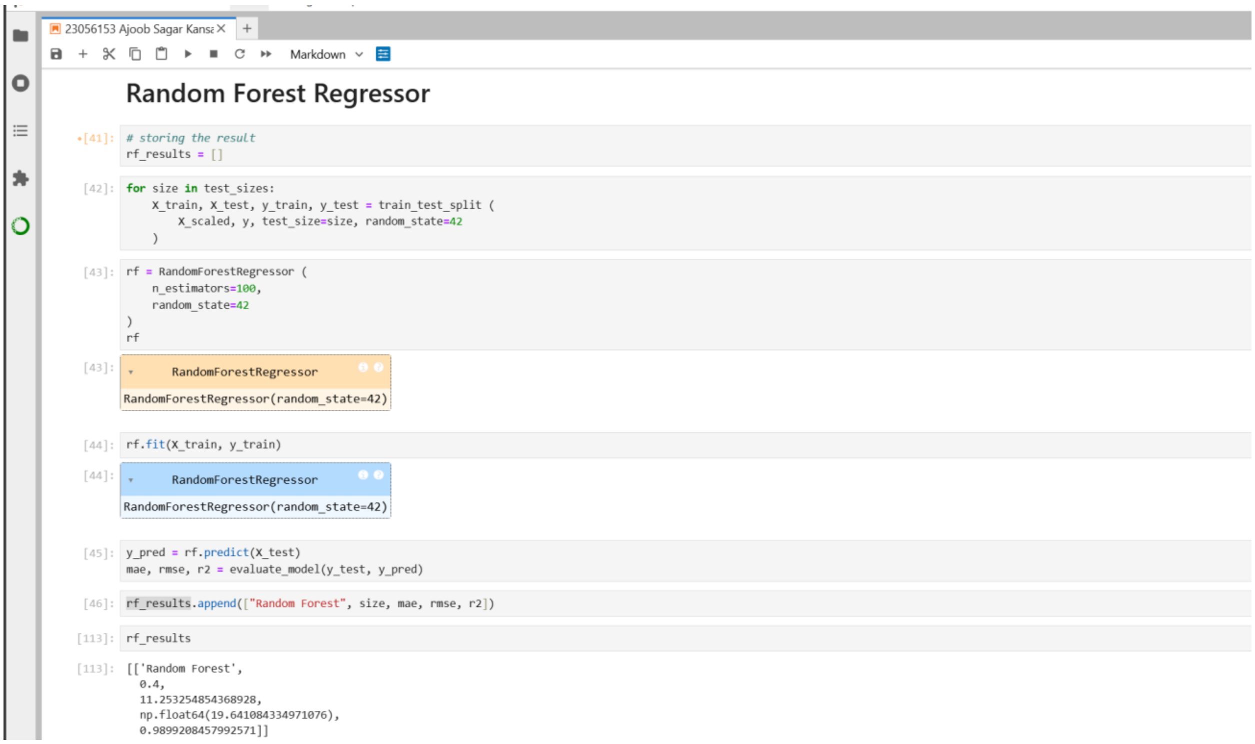
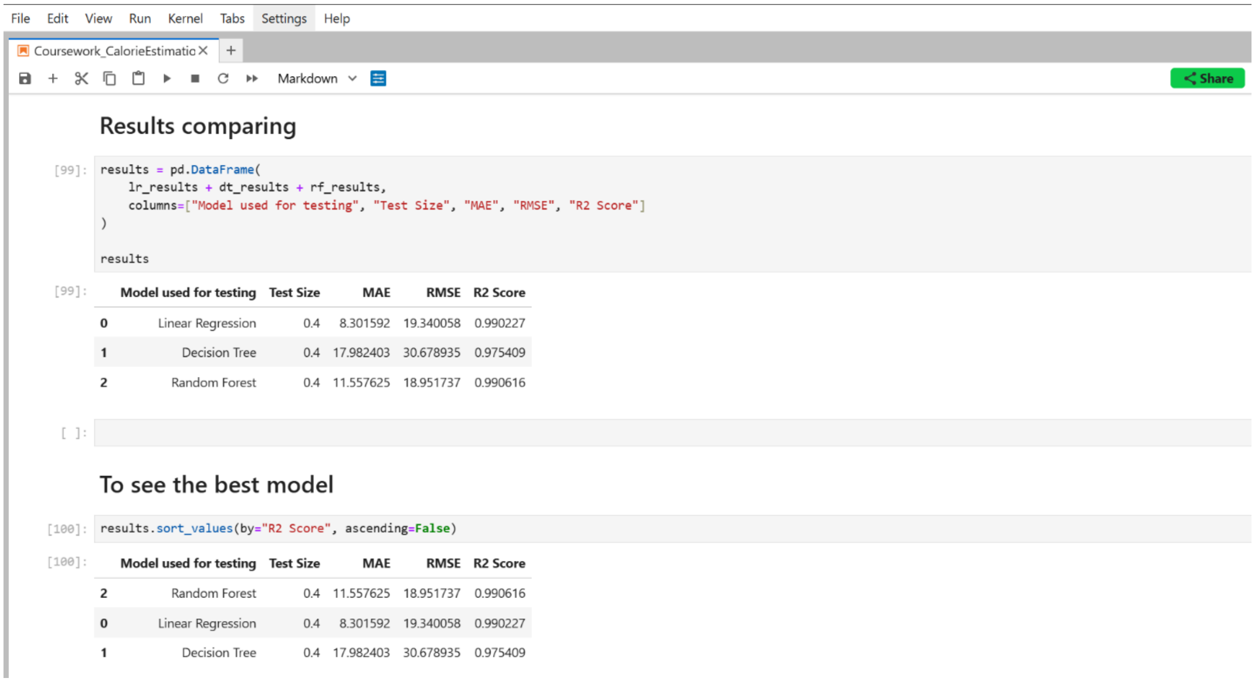


Figure 24: Random Forest Regressor Test

Output:

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

3.6.6 Result Comparing for 3 models



File Edit View Run Kernel Tabs Settings Help

Coursework_CalorieEstimatio

Markdown

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 25: 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 so it is the best model to use to estimate calories of a food.

R2 score is very high in my project as the input features present in my dataset is directly related to the target variable as calorie estimation is usually done with the help of proteins, carbohydrates and fats which is present in my dataset for all the food items.

R2 score may indicate overfitting, but with the use of MAE and RMSE alongside ensures robust evaluation and reliable output.

3.6.7 Comparison Chart for 3 models

During train_test_split multiple test_sizes were used to test the model (0.2, 0.3,0.4)

But for visualization and comparison, performance metrics obtained using test size of 0.4 (40% test and 60% train) was used.

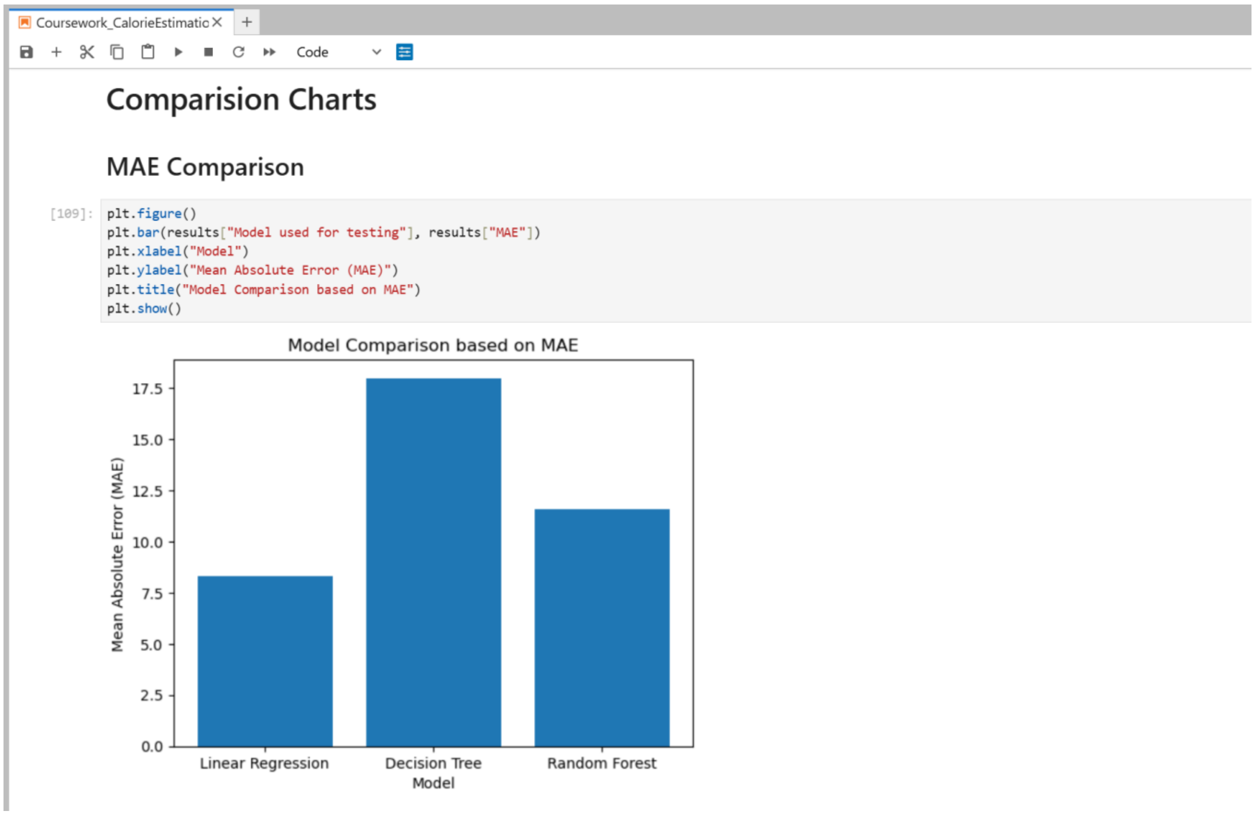


Figure 26: MAE Comparison

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()
```

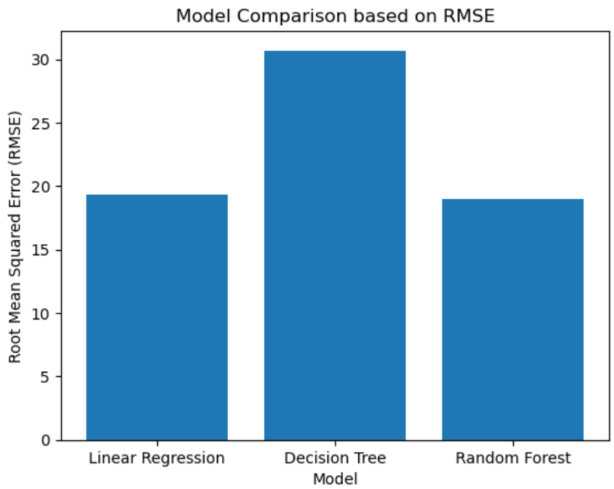


Figure 27: RSME Comparison

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()
```

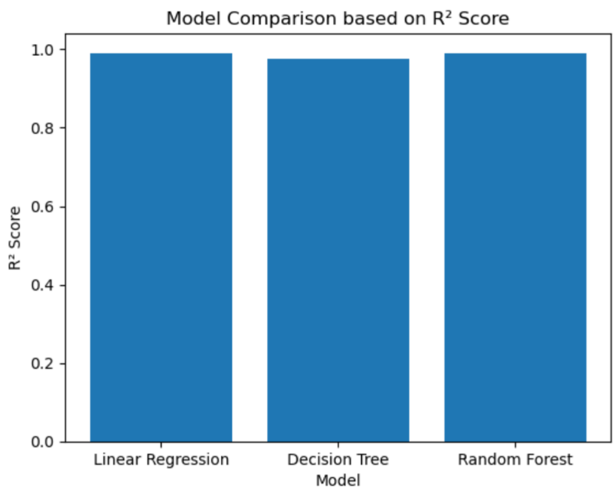


Figure 28: R2 Comparison

Output:

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

3.6.8 Combined Comparison

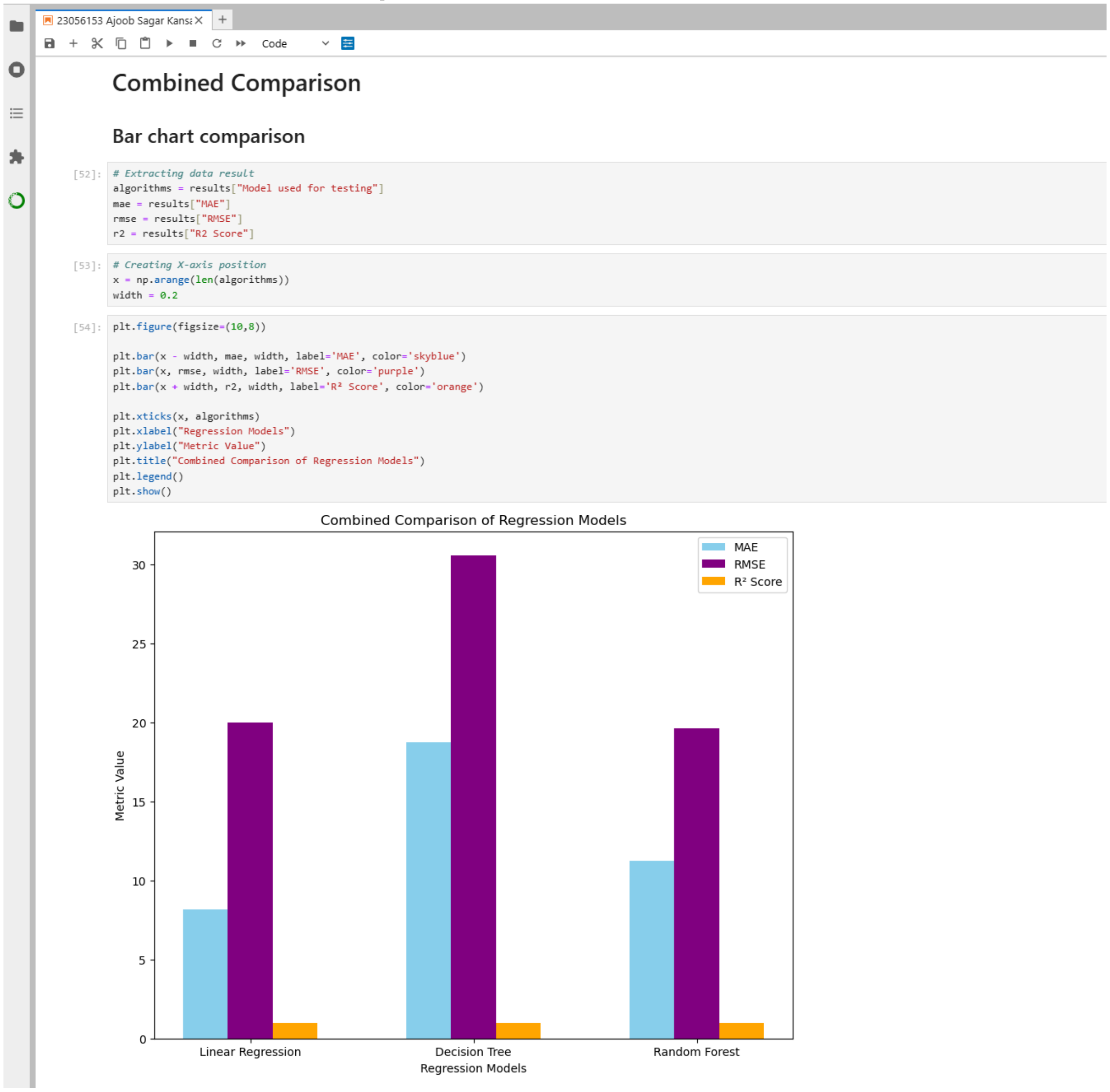


Figure 29: Combined Bar Chart Comparison of all Algorithms

Output:

The figure shows the combined comparison of Linear Regression, Decision Tree, and Random Forest models using MAE, RMSE, and R2 score. Lower MAE and RMSE values indicate better prediction accuracy, while higher R2 value indicates stronger explanatory power. Linear Regression demonstrates low error values which shows that strong linear relationship between nutritional features and calories. Decision Tree shows the highest error values, indicating overfitting despite a high R2 score. Random Forest shows balance performance with low error rates and high R2, making it the most reliable model for calorie prediction.

Coursework_CalorieEstimatic

Heatmap Comparison for the models

```
[129]: # for model result data
heatmap_data = results.set_index("Model used for testing")
```

```
[137]: plt.figure(figsize=(10,8))
sns.heatmap(heatmap_data, annot=True, fmt=".2f", cmap="coolwarm",linewidths=0.5)
plt.title("Heatmap Comparison of Regression Models")
plt.xlabel("Evaluation Metrics")
plt.ylabel("Regression Models")
plt.show()
```

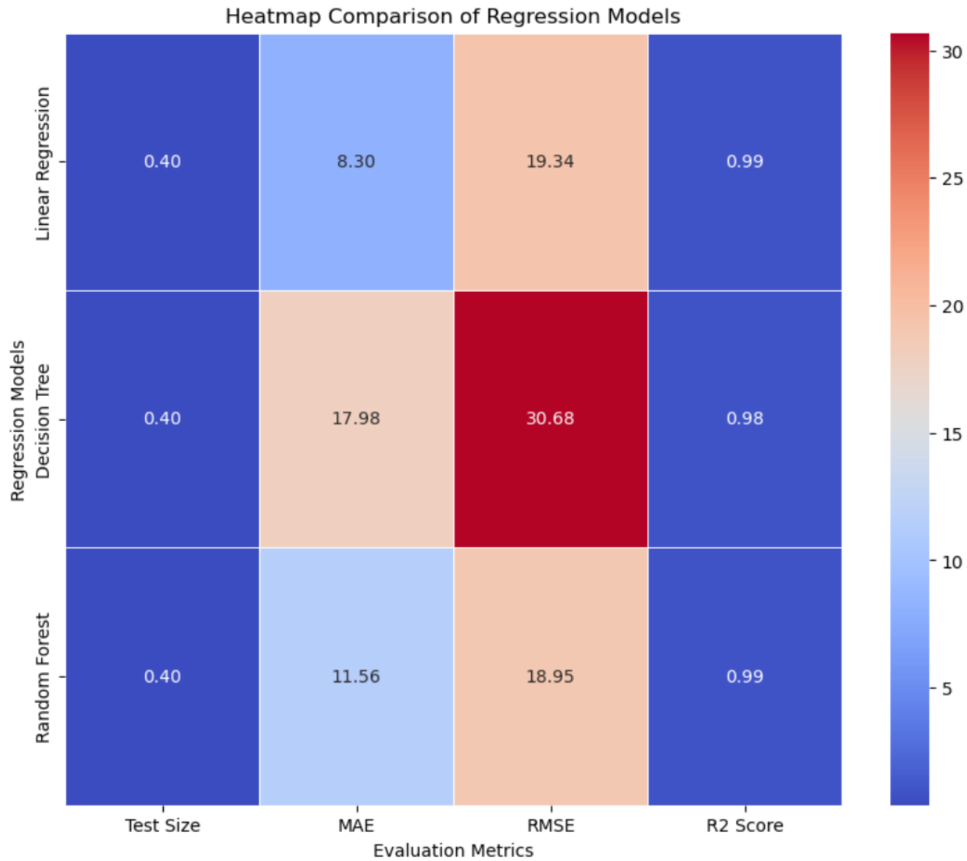


Figure 30: Heatmap Comparison of all Algorithms

Output:

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

From the heatmap, we observe that Decision Tree is the worst performer amongst the models as it has the highest errors MAE & RMSE.

Random Forest is the best performer amongst the models as it has the lowest RMSE.

3.6.9 Overfitting Test for Linear Regression

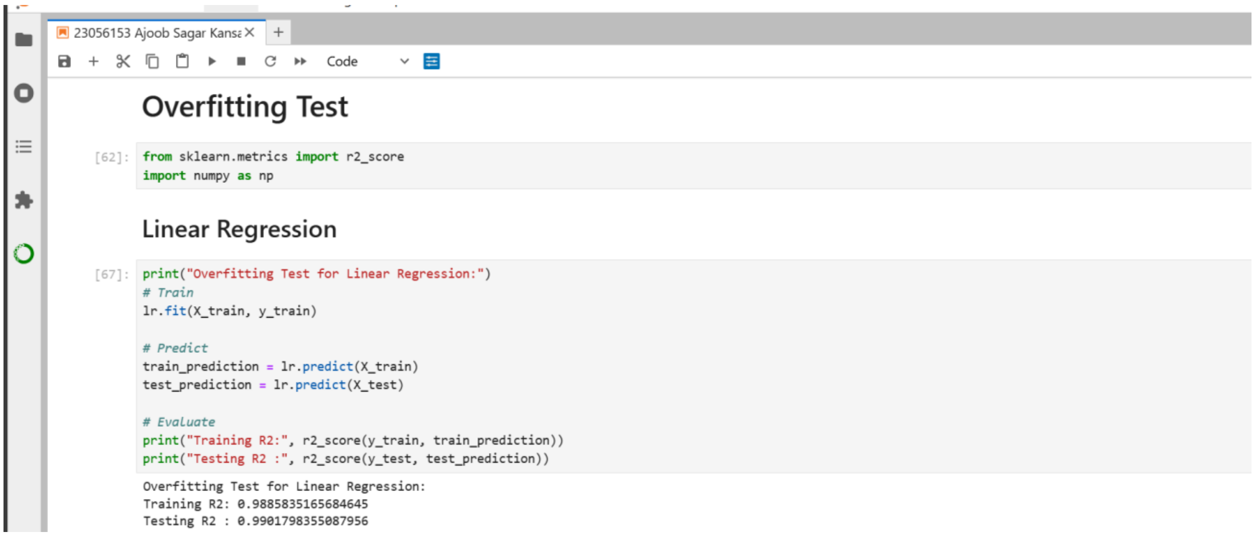


Figure 31: Overfitting test for Linear Regression

Overfitting was evaluated by comparing the R2 score of training and testing datasets.

The Training R2 score and Testing R2 score have minimal difference in performance, indicating that Linear Regression generalizes well and have no overfitting issues.

3.6.10 Overfitting Test for Decision Tree Regressor

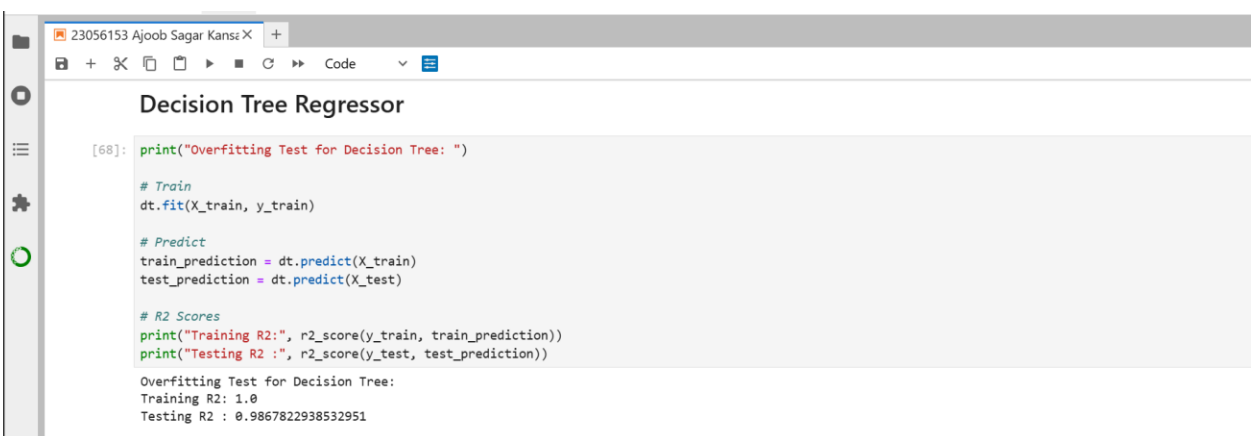


Figure 32: Overfitting test for Decision Tree Regressor

Decision tree shows a training R2 of 1.0, which indicates a perfect fit. And testing R2 of 0.98, meaning there is minimal overfitting and the model generalizes well.

Hyperparameter Tuning for Decision Tree

Figure 33: Hyperparameter tuning for Decision Tree

Although my project had minimal overfitting for decision tree, manual hyperparameter tuning was applied by limiting the maximum depth of tree.

max_depth parameter was set to 10 in order to control model complexity and improve generalization. After tuning, the training R2 score was 0.99 and testing R2 score was 0.98, indicating good balance.

Small difference between training and testing performance confirms that overfitting was effectively reduced and the model generalizes well.

3.6.11 Overfitting Test for Random Forest Regressor

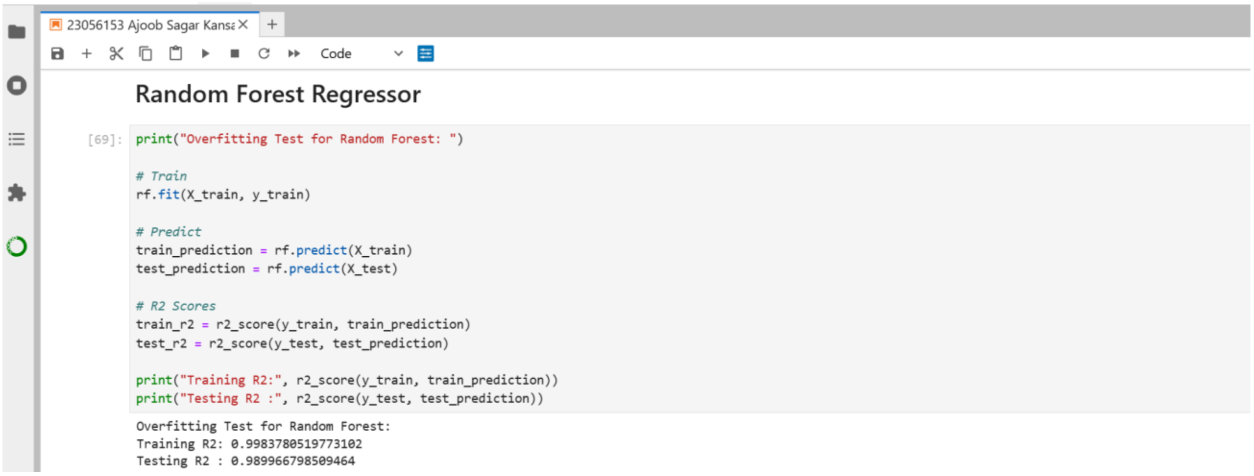
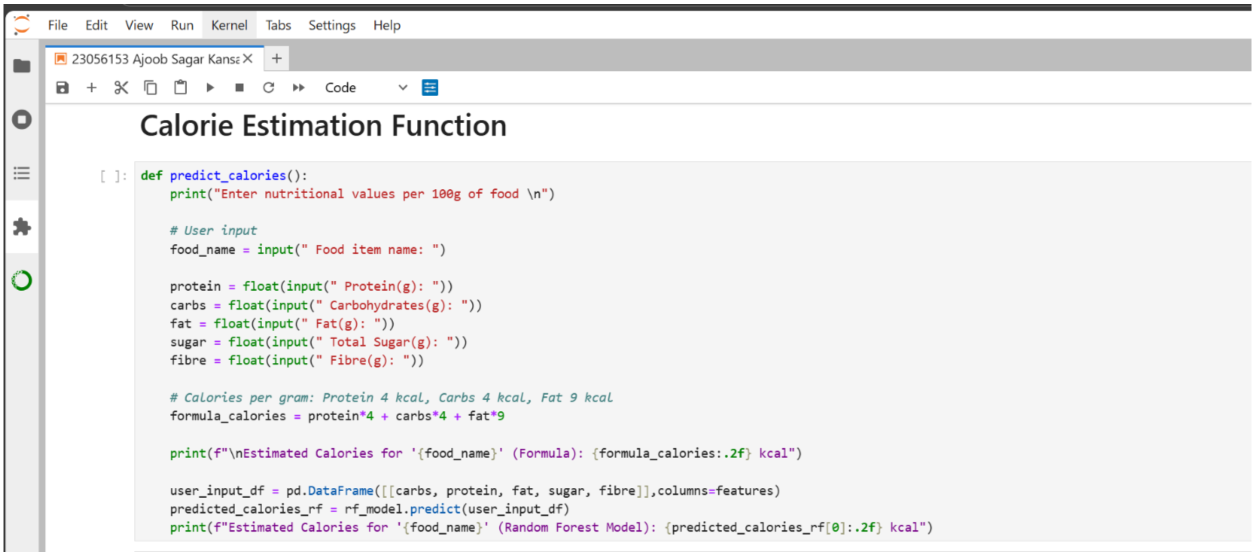


Figure 34: Overfitting test for Random Forest Regressor

Random Forest shows a training R2 of 0.99, and testing R2 of 0.98, meaning there is minimal overfitting and the model generalizes well. Random Forest reduces overfitting by averaging the predictions of multiple decision trees. So, difference between training and testing R2 is smaller compared to Decision Tree.

3.6.12 Calorie Prediction Function Using Random Forest Algorithm




```python
def predict_calories():
    print("Enter nutritional values per 100g of food \n")

    # User input
    food_name = input(" Food item name: ")

    protein = float(input(" Protein(g): "))
    carbs = float(input(" Carbohydrates(g): "))
    fat = float(input(" Fat(g): "))
    sugar = float(input(" Total Sugar(g): "))
    fibre = float(input(" Fibre(g): "))

    # Calories per gram: Protein 4 kcal, Carbs 4 kcal, Fat 9 kcal
    formula_calories = protein*4 + carbs*4 + fat*9

    print(f"\nEstimated Calories for '{food_name}' (Formula): {formula_calories:.2f} kcal")

    user_input_df = pd.DataFrame([[carbs, protein, fat, sugar, fibre]],columns=features)
    predicted_calories_rf = rf_model.predict(user_input_df)
    print(f"Estimated Calories for '{food_name}' (Random Forest Model): {predicted_calories_rf[0]:.2f} kcal")
```

Figure 35: Calorie Estimation Function

predict_calories() function collects all the nutritional information of a food item per 100g from the user and estimates its caloric value using two approaches.

First the standard nutritional formula, which is:

Total calories = Protein*4 + Carbs*4 + Fat*9

Second inputs are passed to a trained Random Forest regression model, which predicts calories based on learned patterns from the provided nutrition dataset.

Using both the approaches we could compare between traditional rule-based calculation and Machine learning-based prediction.

3.6.13 Calories Prediction with Random Forest Regressor:

Figure 36: Prediction 1

Figure 37: Prediction 2

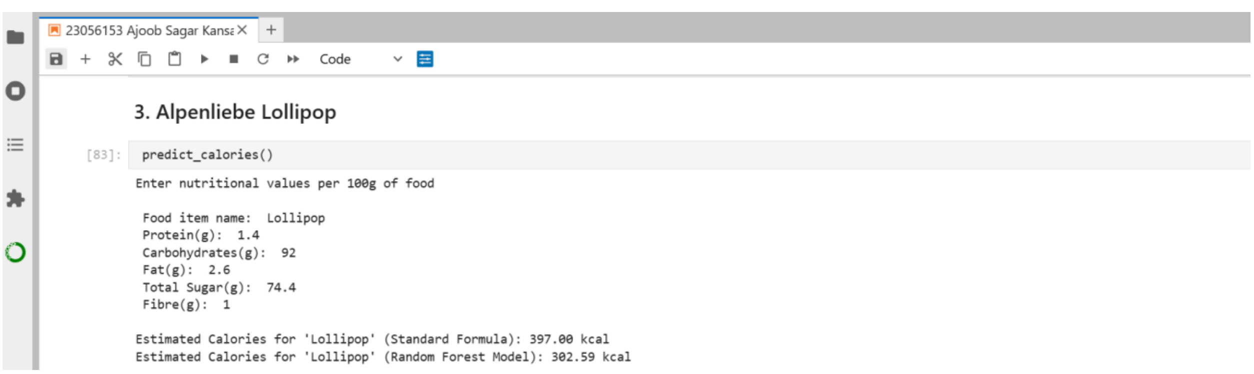


Figure 38: Prediction 3

3.6.14 Comparison of Calorie Estimation using all models

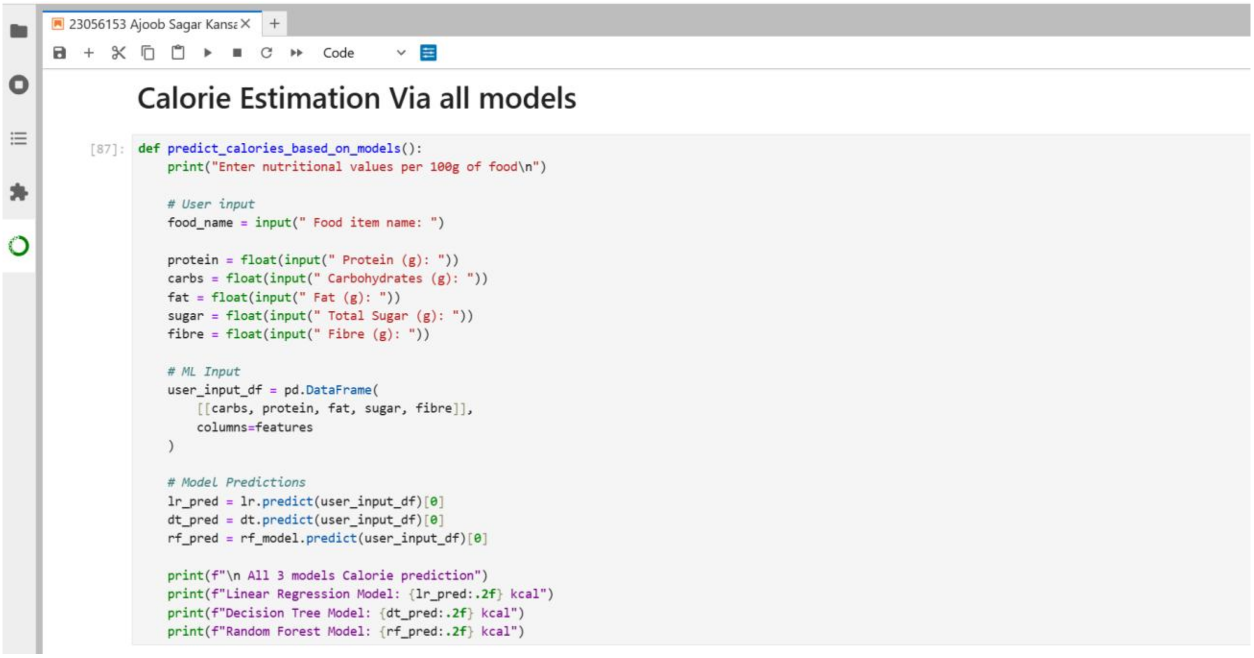


Figure 39: Calorie Prediction function for all 3 models combined

This function displays the calorie prediction of a food using all three models which are Linear Regression, Decision Tree Regressor, and Random Forest Regressor.

3.6.15 Prediction Comparison of all 3 models

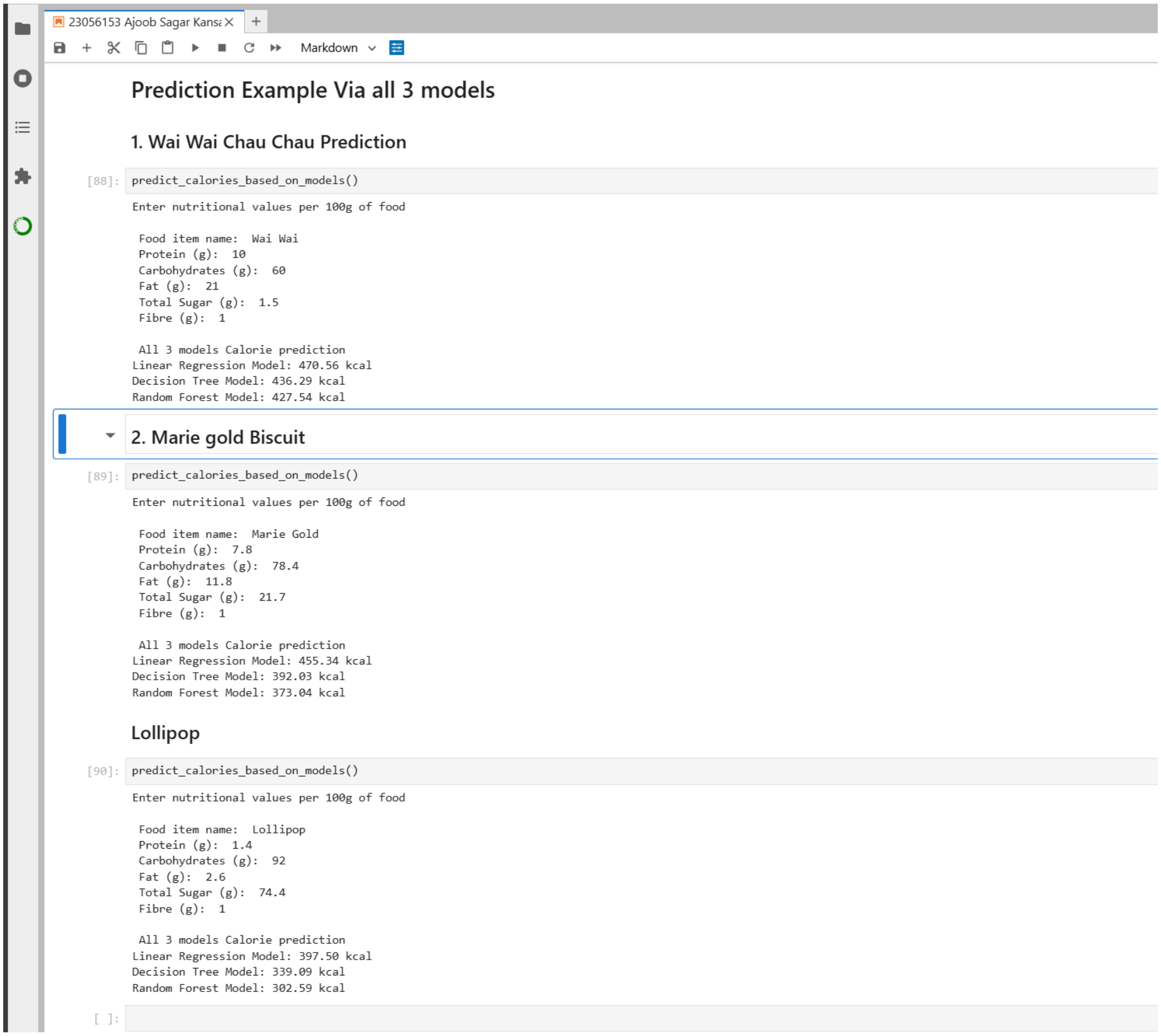


Figure 40: Prediction 4: Comparison of all models

4. Conclusion

In conclusion, this project successfully demonstrates the application of machine learning techniques to predict the calorie content of food items using their nutritional composition, by training a machine learning model using nutrition dataset and selecting the best model to estimate the calorie content of a food item. The calorie values were estimated based on macronutrient information such as carbohydrates, protein, fats, sugar, and fibre, rather than relying solely on manually calculated formulas. To achieve this, a structured machine learning pipeline was implemented, beginning with dataset exploration and pre-processing, followed by model training, evaluation, and comparison. Three regression algorithms Linear Regression, Decision Tree Regressor, and Random Forest Regressor were used due to their relevance in continues value prediction. The dataset was cleaned by handling missing values, invalid columns were dropped, and the data was split into training and testing sets using multiple test sizes to ensure robust evaluation.

Each model was trained and assessed using standard regression performance metrics which includes Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the R2 score. The analysis of the model revealed that while Linear Regression performed reasonably well due to strong mathematical relationship between macronutrients and calories, tree-based models were better at capturing non-linear patterns within the nutritional data. Random Forest Regressor achieved the highest R2 score and lowest error values across different test sizes, indicating superior generalization capability and predictive accuracy. Overfitting analysis was conducted by comparing training and testing R2 scores, results confirm that ensemble-based models are effective for nutritional calorie estimation tasks.

Overall, the 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) Ali, M., 2022. *Supervised Machine Learning.* [Online]
Available at: https://www.datacamp.com/blog/supervised-machine-learning
[Accessed 18 January 2026].

2) Antwl, J.B, Tuffour, Mensah, 2022. Machine Learning in Nutrition Research. *Nutrition Journal,* Issue https://pmc.ncbi.nlm.nih.gov/articles/PMC9776646/.

3) Changhe Yang, 2024. Predicting Nutrient Density in Foods Using Machine Learning Models. *DAMI 2024,* p. 6.

4) Grammerly, 2024. *Unsupervised Learning: What it is and How it works.* [Online]
Available at: https://www.grammarly.com/blog/ai/what-is-unsupervised-learning/
[Accessed 18 January 2026].

5) Harvey, A., 2022. *How calories are calculated: The science behind your food..* [Online]
Available at: https://www.livescience.com/62808-how-calories-are-calculated.html
[Accessed 18 January 2026].

6) Jain, A., 2024. *All about decison trees.* [Online]
Available at: https://medium.com/@abhishekjainindore24/all-about-decision-trees-80ea55e37fef
[Accessed 18 January 2026].

7) Kavita, 2025. *Linear Regression in Machine Learning.* [Online]
Available at: https://www.analyticsvidhya.com/blog/2021/10/everything-you-need-to-know-about-linear-regression/
[Accessed 18 January 2026].

8) Snowflake, Inc, n.d. *What is Random Forest in Machine Learning?.* [Online]
Available at: https://www.snowflake.com/en/fundamentals/random-forest/
[Accessed 18 January 2026].

9) 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/.