KAZAKH BRITISH TECHNICAL UNIVERSITY

Data Mining

Final Project Report

The Impact of Lifestyle, Nutrition, and Exercise on Physical Fitness and Health (Personalized health analytics using machine learning)

Students:

Tolkynbekova S., Darmesh A., Shabalina A.

Almaty 2025

1. Actuality and Relevance

In recent years, interest in healthy lifestyles, physical activity, and balanced nutrition has increased significantly. Despite the widespread availability of fitness applications, online workout programs, and diet planners, many users struggle to achieve long-term and sustainable results. One of the main reasons for this is the lack of personalization in existing solutions, which often rely on generic workout and nutrition plans.

The research by Ratola et al. (2025) reports high exercise dropout rates in fitness centres, frequently linked to misalignment between service quality, training supervision, and individual user characteristics such as fitness level, lifestyle constraints, and personal goals. This issue has become particularly salient in 2024–2025, as fitness services continue to expand while user engagement and long-term adherence remain low.

This project focuses on addressing these issues by providing a framework which produces a personalized workout and nutrition plans to individual user profiles using data-driven and machine learning approaches.

Source: Ratola, L., Caleiras, E., Veríssimo, F., Neto, H., Aveiro, D., Almeida, D., Aleixo, J., & Campos, F. (2025). *Possible reasons for exercise dropout in fitness centres: Insights from active users*. Journal of Physical Education and Sport, 25(9), 1799–1805. https://doi.org/10.7752/jpes.2025.09199

2. Related work

This section reviews existing solutions to workout and diet recommendation problems, focusing on representative methodological approaches used in the field. By examining their strengths and limitations, these studies provide context for current challenges and directly motivate the design choices adopted in our project.

Research work 1: Yadav, A., Gautam, P., Choudhary, M., Sankhe, K., & Sirsat, A. (2023). *Workout prediction using machine learning*. In Proceedings of the 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN 2023) (pp. 123–126). IEEE. https://doi.org/10.1109/ICPCSN58827.2023.00026

This study presents a machine-learning–based fitness application that predicts workout routines using user physical attributes collected via mobile and wearable devices.

The proposed system applies **K-Means** clustering to group users with similar fitness characteristics and calorie ranges, and workout suggestions are derived by mapping these clusters to **predefined workout types**. Although the study shows that clustering can effectively organize users with similar profiles, the recommendation stage is driven by **fixed, rule-based mappings** between clusters and workouts, not by a learned predictive model. Consequently, personalization remains static and coarse-grained, and the system does not provide an end-to-end framework that adaptively optimizes workout recommendations at the individual level.

Research work 2: Kumari, D. N. N., Satya, T. P., Manikanta, B., Chandana, A. P., & Aditya, Y. L. S. (2023). ***Personalized diet recommendation system using machine learning*****. SSRN. https://ssrn.com/abstract=4877349**

The study proposes a **personalized diet recommendation system** that generates daily meal plans (breakfast, lunch, snacks, dinner) based on a user's physical characteristics (age, gender, BMI), activity level, and dietary preferences, with the goal of supporting healthier eating and weight management. It uses a **content-based recommendation approach** with a **Nearest Neighbors model (k-NN)** and **cosine similarity** to match users with recipes that have similar nutritional profiles. The system preprocesses nutritional data (scaling, feature selection), computes similarity between user requirements and recipes, and delivers recommendations via a **FastAPI backend** and **Streamlit web interface**.

3. Novelty & Originality

Existing machine learning–based fitness studies provide personalization based on **predefined rules**. In the first related work, users are clustered using K-Means based on a fixed set of physical attributes, and workout recommendations are assigned according to cluster membership. The workouts themselves are selected from predefined categories and parameters, without deeper feature construction or adaptation beyond the cluster level.

The second study applies a k-Nearest Neighbors approach with cosine similarity to recommend diets based on user similarity. While this method captures similarity between users, personalization remains limited to matching existing profiles and does not involve joint modeling of nutrition and physical activity or deeper feature-level optimization.

Our project extends these approaches by shifting the focus from isolated models to a **fully integrated personalization framework**. Instead of stopping at clustering or prediction, we design an **end-to-end machine learning pipeline** that transforms raw user data into a complete and practical fitness solution.

Unlike existing studies, our system:

- combines **user clustering, advanced feature engineering, and supervised models** into a single workflow,

- performs **joint optimization of workout and meal models**, ensuring alignment between energy expenditure, calorie intake, and macro-nutrient balance,

- dynamically incorporates a **real user profile (new_user)** at **runtime**, allowing true individual-level personalization rather than static recommendations,

- produces a **final personalized plan in PDF format**, making the results directly usable outside an academic setting.

The novelty of our work lies in the **integration of multiple machine learning components into a practical, user-oriented system** that bridges the gap between predictive modeling and real-world application. By delivering a complete personalized plan rather than intermediate predictions or clusters, the proposed solution advances beyond existing approaches and addresses key limitations identified in prior research.

4. Dataset description and understanding

Our initial dataset - Life Style Data. This dataset contains information related to people's lifestyles, which includes their daily food and workout descriptions, and physical characteristics. To conduct a reliable machine learning model, we first need to understand our data.

The dataset includes 20.000 rows (observations) and 43 columns (variables). Each row represents a single participant/individual, and the columns reflect various attributes related to their lifestyle. For example, the variables include the participants' age, gender, weight, name of the exercise, and food parameters (proteins, carbs, fats), and so on.

During the EDA analysis, we find that the dataset has no missing, empty, or duplicate data, but there are some logical inconsistencies.

5. Dataset Preprocessing and Outlier Handling

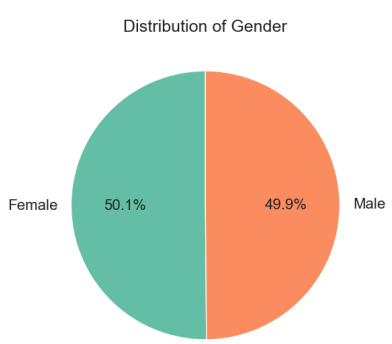
During the preprocessing:

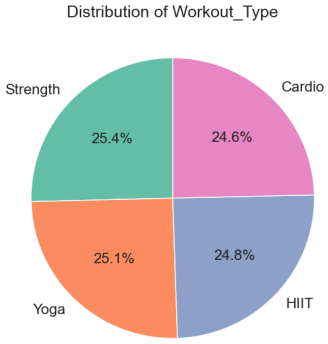
1. We removed columns that we don't need in the report and columns with unclear and ambiguous meanings (such as `'meal_name', 'Physical exercise', 'prep_time_min', 'cook_time_min'`)

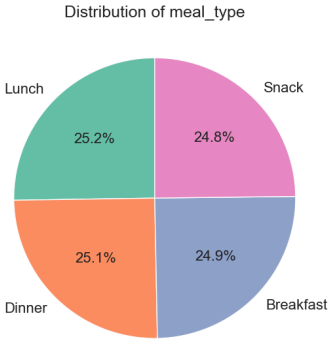
2. Deleted values that logically cannot be less than zero (such as `'sugar_g', 'cholesterol_mg'` - These columns represent nutritional quantities, which cannot be negative)

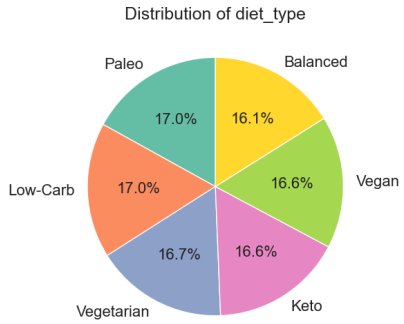
3. Rounded some columns (`'Age', 'Workout_Frequency (days)', 'Daily meals frequency', 'Experience_Level', 'Sets'`) – Since these columns cannot logically be decimals

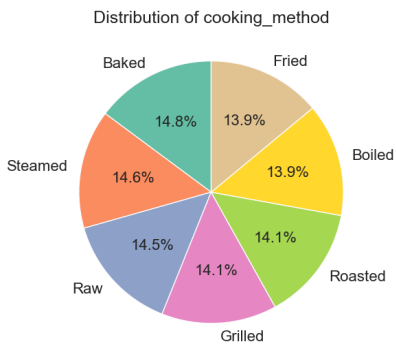
Main distributions of categorical variables:

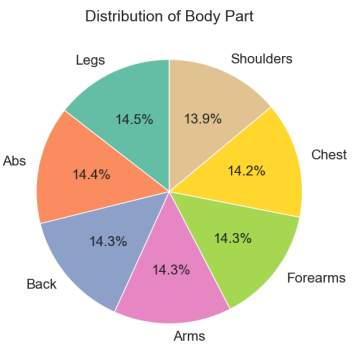

Distribution of Gender
Female
50.1%
49.9%
Male

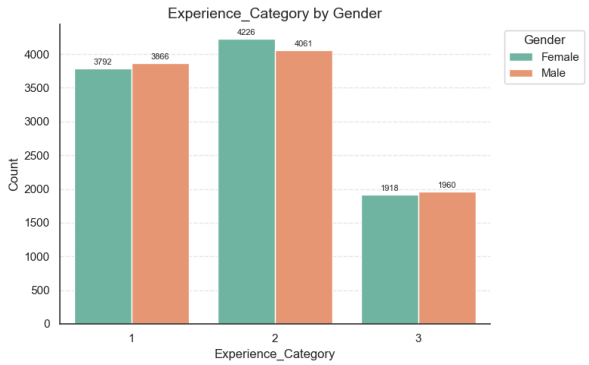

Distribution of Workout_Type
Strength
25.4%
Cardio
24.6%
25.1%
Yoga
24.8%
HIIT

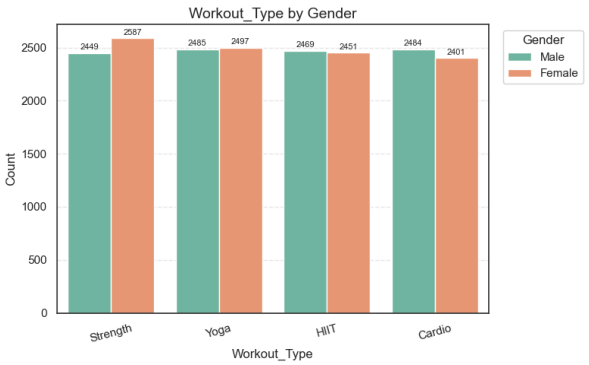

Distribution of meal_type
Lunch
25.2%
Snack
24.8%
25.1%
Dinner
24.9%
Breakfast

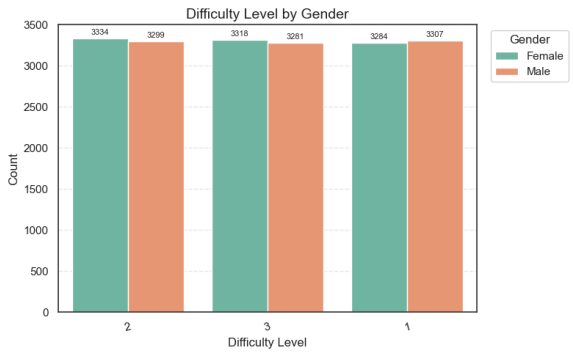

Distribution of diet_type
Paleo
17.0%
Balanced
16.1%
Low-Carb
17.0%
16.6%
Vegan
16.7%
16.6%
Vegetarian
Keto


Distribution of cooking_method
Baked
14.8%
Fried
13.9%
Steamed
14.6%
13.9%
Boiled
14.5%
14.1%
Roasted
Raw
14.1%
Grilled

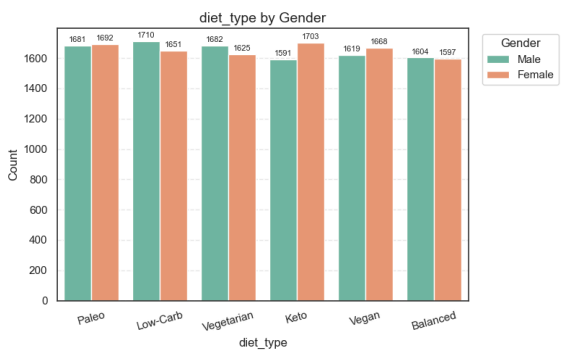

Distribution of Body Part
Legs
14.5%
Shoulders
13.9%
Abs
14.4%
14.2%
Chest
14.3%
14.3%
Back
14.3%
Forearms
Arms

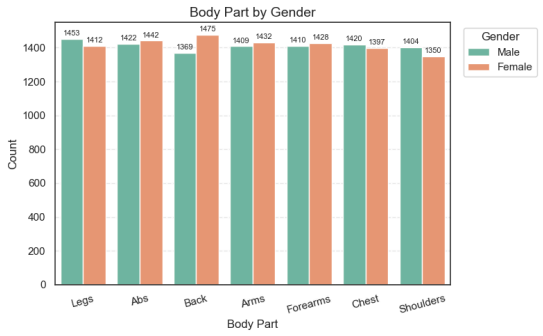

Experience_Category by Gender
Gender
Female
Male
Count
Experience_Category
3792
3866
4226
4061
1918
1960


Workout_Type by Gender
Gender
Male
Female
Count
Workout_Type
Strength
Yoga
HIIT
Cardio
2449
2587
2485
2497
2469
2451
2484
2401


Difficulty Level by Gender
Gender
Female
Male
Count
Difficulty Level
3334
3299
3318
3281
3284
3307


meal_type by Gender
Gender
Male
Female
Count
meal_type
Lunch
Dinner
Breakfast
Snack
2485
2519
2499
2469
2477
2456
2426
2492

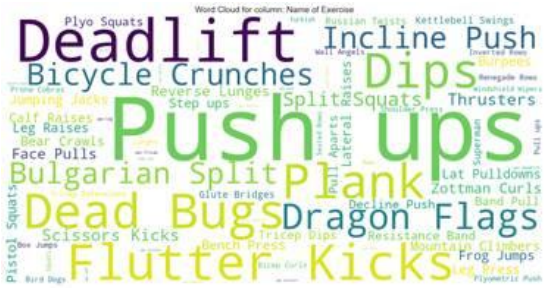


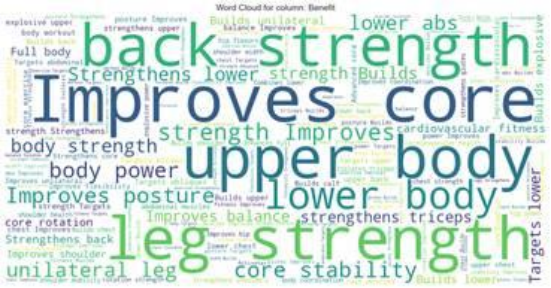


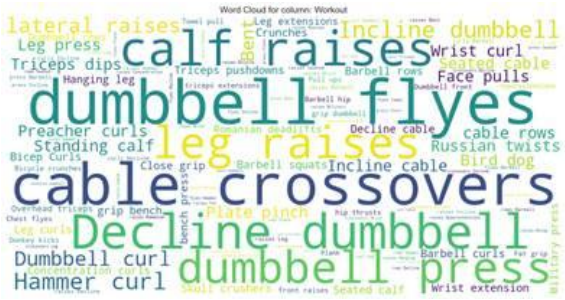
We can see a uniform distribution of categorical variables. This means that no one category dominates the dataset. For example, groups based on gender, activity level, lifestyle, or eating habits are represented in comparable proportions. In the context of this project, this is a significant advantage because: the recommendation system will not be biased in favor of one user group; training models receive a sufficient number of examples for each category; workout and nutrition recommendations can be more personalized and fair for different types of users.

Distribution of Numerical Features:

For categorical variables that may have many possible values, it will be inconvenient to show all categories at one pie chart or bar chart. That is why we decided to show them using the wordcloud so it will be more readable and easier to understand

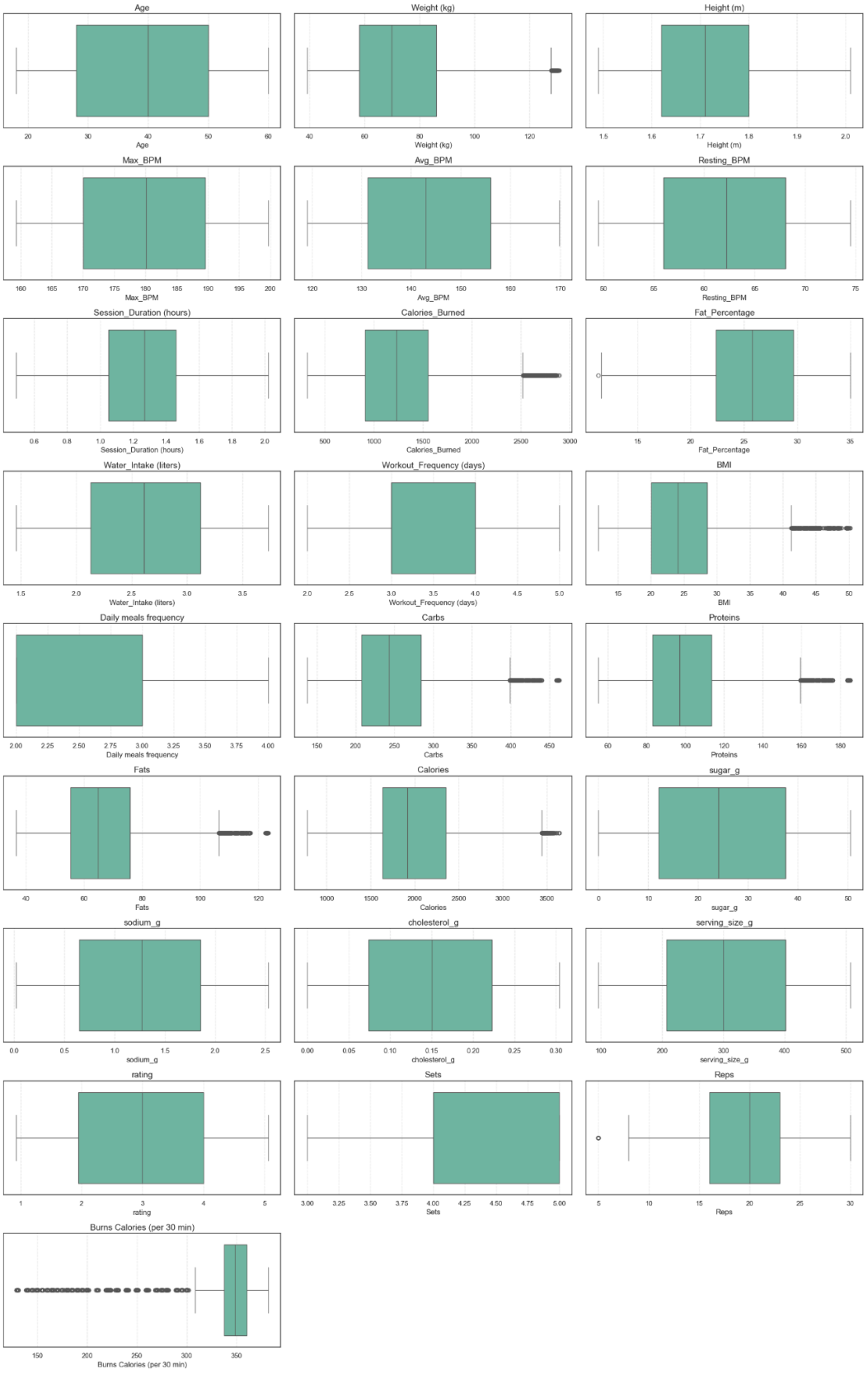






Outliers Analysis:

To find outliers, we created boxplots for all numerical features



From the boxplots it is clearly shown that most of the values do not have outliers. In the project for workout and nutrition recommendation system, **aggressive** outlier removal was not used, as such values may reflect real-world individual characteristics of users (e.g., high or low weight, unusual physical activity levels).

This is especially important for a recommendation system because: users with atypical characteristics should also receive relevant recommendations. For example, for columns:

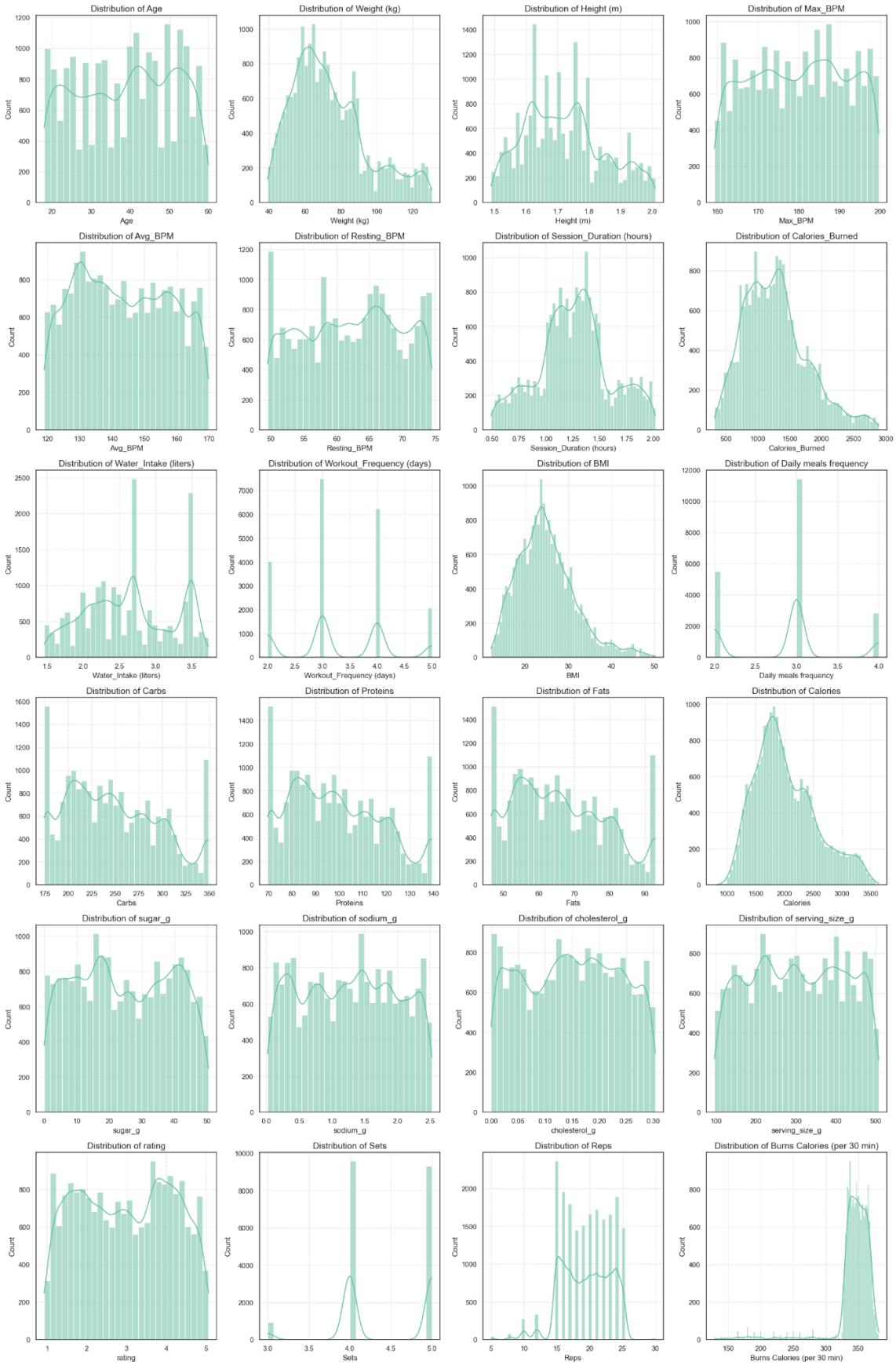
`Weight (kg)` - We should not remove outliers in weight because these extreme values represent real clients, and removing them could make the model ignore people with very high weight. Weight is critical for calculating calories, BMI, and personalized recommendations. The values do not go beyond the limits of what is abnormal for a person, so we left them.

`Calories_burned and Calories` - Big outliers can dominate model training if we don't handle them. Standardization (z-scaling) puts all values on the same scale. Outliers stay in the data, but their influence is balanced, so the model doesn't overfit to extreme cases. This way, the model can learn properly from all clients, including those with very high values. (So we are going to use scaling for model preprocessing)

`BMI` – BMI is calculated from Weight and Heights. We should keep BMI outliers because they represent real people with very low or very high BMI. Removing them would make the model ignore these clients. Standardization or scaling can handle extreme values without losing important information.

We chose Winsorization for `Carbs, Proteins, and Fats` because these attributes can have extreme values that are real but rare, like very large portion sizes. Removing them would lose important information, while leaving them as-is could make the model too sensitive to outliers. Winsorization reduces the influence of these extreme values by capping them at a chosen percentile, keeping most of the data intact.

Distributions after outliers handling:



After preprocessing and outliers handling, we save our cleaned dataset to `preprocessed_without_scaling.csv`

6. Feature Engineering

6.1. User Features creation

To estimate required number of calories to burn from workouts and calories to be received from food we create new user features:

$$BMRmen = 10 \cdot weight + 6.25 \cdot height(cm) - 5 \cdot age + 5$$
$$BMRwomen = 10 \cdot weight + 6.25 \cdot height(cm) - 5 \cdot age - 161$$

$$PAL = 1.2 + 0.175 \cdot Workout\_Frequency$$
$$TDEE = BMR \cdot PAL$$

$$CalorieChange = WeightChange \cdot 7700$$
$$CaloriesToBurnTraining = CalorieChange \cdot 0.5$$
$$CaloriesReducedFromFood = CalorieChange \cdot 0.5$$

$$CaloriesPerDay = TDEE - \frac{CaloriesReducedFromFood}{GoalDays}$$

$$TotalWorkouts = Workouts\_Frequency \cdot \frac{GoalDays}{7}$$

$$CaloriesPerWorkout = \frac{CaloriesBurnedTrainingTotal}{TotalWorkouts}$$

`BMR` - **Basal Metabolic Rate** is the rate at which the body expends energy to maintain essential life functions while at complete rest.

`PAL` - The **physical activity** level is a way to express a person's daily **physical activity** as a number and is used to estimate their total energy expenditure

`TDEE` - **Total Daily Energy Expenditure,** which is the total number of calories your body is burning each day

`CalorieChange` - converts the user's weight loss or gain goal from kilograms into calories

`CaloriesToBurnTraining`, `CaloriesReducedFromFood` - divides `CalorieChange` into change in Calories for workouts and for meals.

`CaloriesPerDay` - shows the required number of calories per day for the user based on his/her goal. `TDEE` is the daily calories needed to maintain weight. To lose weight, you need to eat less `TDEE`, so we subtract the daily deficit. To gain weight, you need to eat more `TDEE`, so we add the daily surplus.

`TotalWorkouts` - Total number of workouts during whole chosen period (`GoalDays`)

`CaloriesPerWorkout` - show number of calories to burn during one workout

6.2 Workout Features

We determined 5 main features that will show the significance and effectiveness of workout

`E - Energy Consumption`

E – Energy Consumption

$$E = 0.5 * E\_raw + 0.5 * E\_eff$$

$$E\_raw = \frac{Calories\_Burned - \min(Calories\_Burned)}{\max(Calories\_Burned) - \min(Calories\_Burned)}$$

$$E\_eff = \frac{Burn\_Cal\_per30\,\text{min} - \min(Burn\_Cal\_per30\,\text{min})}{\max(Burn\_Cal\_per30\,\text{min}) - \min(Burn\_Cal\_per30\,\text{min})}$$

Uses variables `Calories_Burned` - total calorie expenditure during the workout. and `Burn_Cal_per30min` - intensity, how many calories are burned per unit of time. It's important to consider both because: total expenditure reflects the total energy expended during a session. A high expenditure helps create a calorie deficit, promoting weight loss (PMC).

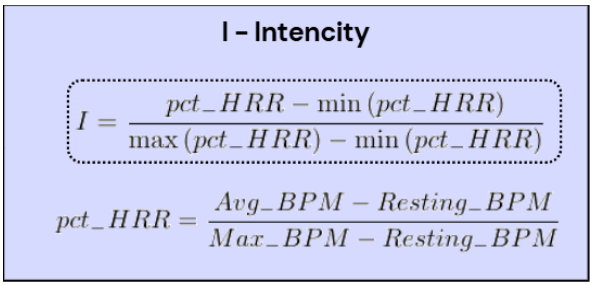
Intensity affects metabolic processes and post-exercise energy expenditure. When time is limited, a short but intense workout can be more effective than a long but low-intensity one (Wikipedia)

Sources:

Effect of a 12-Week High-Calorie-Expenditure Multimodal Exercise Program on Health Indices in Women With Overweight: Protocol for a Randomized Controlled Trial - PMC

Зависимость доза-эффект в спорте — Википедия

`I - Intensity`

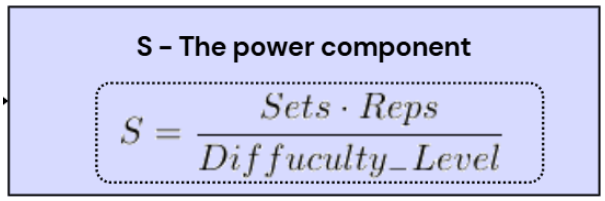


`HRR` measures the intensity of your workout relative to your personal heart rate range, taking into account your maximum and resting heart rate. The Karvonen formula is used: it takes into account not only Max HR but also Resting HR to estimate the actual load on the body for a specific individual. This makes the measurement more accurate than a simple percentage of Max HR.

Sources:

The effects of training on heart rate - a longitudinal study (Karvonen).pdf

`S - Power component`



`Workload = Sets * Reps` shows total exercise volume, while `Difficulty_mean` - average workload for a difficulty level (Beginner/Intermediate/Advanced).

`S` - shows how well the workload of a particular exercise compares to the norm for that level. Some exercises are more difficult than others for the same volume, so we divide by the average for the category.

Sources:

ACSM's guidelines for exercise testing and prescription - NLM Catalog - NCBI

`D - Duration`

D – Duration

$$D = \frac{Duration - \min(Duration)}{\max(Duration) - \min(Duration)}$$

D - is the duration of the workout in hours (hours instead of minutes). Values are normalized so that the maximum is set to 1. When assessing the effectiveness of a workout, it's not just how long a person spent in the gym that matters, but also how long that time actually produced benefits.

`R - Risk (1 - penalties)`

R – Risk

$$R = 1 - (0.4 * pen\_age + 0.3 * pen\_bmi + \\ + 0.2 * pen\_hrr + 0.1 * pen\_skill)$$

$$pen\_age = \frac{Age - \min(Age)}{\max(Age) - \min(Age)} \quad pen\_bmi = \frac{BMI - \min(BMI)}{\max(BMI) - \min(BMI)}$$

$$pen\_hrr = \frac{\max(0, pct\_HRR - pct\_HRR.quantile(0.9))}{\max(pct\_HRR) - pct\_HRR.quantile(0.9)}$$

$$pen\_skil = \max(0, Difficulty\_Level - Experience\_Level)$$

R - is an indicator that assesses the likelihood of overtraining, injury, or excessive fatigue during a user's workout. It doesn't directly measure effectiveness, but rather helps adjust recommendations to ensure a safe and personalized workout.

Reduces the risk of injury and overtraining, especially for people with a high BMI, those over 40-45 years old, or those with insufficient experience.

Balances effectiveness and safety: training should be productive but not dangerous.

Includes:

- `Age (pen_age)` - risk increases with age, adaptively adjusted for the sample.
- `BMI (pen_bmi)` - takes into account stress on joints and the cardiovascular system.
- `HRR (pen_hrr)` - intensity relative to Heart Rate Reserve; penalty for excessively high loads.
- `Experience vs. difficulty (pen_skill)` - penalty if the difficulty of the workout exceeds the user's experience.

R = 1 - minimum risk, R = 0 - maximum risk

Sources:

Exercise Training and Cardiovascular Risk Factors in Males with Overweight or Obesity: A Systematic Review of Randomized Controlled Trials - PubMed

Association of heart rate recovery after exercise with indices of obesity in healthy, non-obese adults - PubMed

6.3 Meal Features

Similarly as for workout we conducted features that will rate meal suitability and healthiness.

`C - Calorie Fit`

C – Calorie Fit

$$C = 1 - \frac{\left| \frac{Calories}{Meal\_Frequency} - MealTarget \right|}{MealTarget}$$

$$MealTarget = \frac{CaloriesPerDay}{Meal\_Frequency}$$

C - evaluates how closely a meal's calorie intake matches the user's daily goal, which is determined based on their energy expenditure and weight goals. First, the target calorie intake per meal is calculated: `meal_target = calories_per_day / Daily meals frequency`. Next, the deviation of the actual calorie intake from the target is measured, normalized by the target calorie intake, and a compliance score is calculated: `C = 1 - (deviation / meal_target).`

Sources:

Calorie counting - Better Health - NHS

`P - Protein Per Meal`

P – Proteins

$$P = \frac{Proteins}{Meal\_Frequency}$$

Protein is an essential nutrient for muscle growth and maintenance, post-workout recovery, and metabolism. It helps you feel fuller longer and maintain muscle mass during weight loss. Incorporating protein into your diet allows your body to properly distribute nutrients and maintain a balance of energy and recovery.

Sources:

[International Society of Sports Nutrition Position Stand: protein and exercise - PubMed]

`M - Macro Match`

M - Macro Match

$$MacroMatch = 1 - \frac{1}{3}\left(\frac{\left|\frac{Proteins \cdot 4}{Cal\_from\_macro} - target\_p\right|}{target\_p} + \frac{\left|\frac{Carbs \cdot 4}{Cal\_from\_macro} - target\_c\right|}{target\_c} + \frac{\left|\frac{Fats \cdot 9}{Cal\_from\_macro} - target\_f\right|}{target\_f}\right)$$

loss_target = (protein=0.30, carbs=0.35, fats=0.35)
maintain_target = (protein=0.20, carbs=0.50, fats=0.30)
gain_target = (protein=0.25, carbs=0.55, fats=0.20)

`The M (MacroFit)` variable shows how well the distribution of macronutrients - proteins, carbohydrates, and fats in the diet matches the user's goals. First, the calories from each macronutrient are calculated, then the proportion of each in the overall diet (`pct_p, pct_c, pct_f`) is compared with target proportions, which depend on the goal: more protein for weight loss, more carbohydrates for weight gain, and a balanced diet for weight maintenance. This metric helps the system create personalized menus and monitor the quality of nutrition, not just the overall calorie intake.

Sources:

[The Best Macronutrient Ratio for Weight Loss]

`ED - Energy Density`

ED - Energy Density

$$ED = \frac{Calories}{Serving\_Size\_g \cdot Meal\_Frequency}$$

The `ED (Energy Density)` variable measures the calories per gram of food, taking into account the number of meals per day. The ratio of calories to the total mass of daily servings is calculated. This metric is important because high-energy-density foods provide a lot of calories in a small volume, which impacts satiety and weight control, while low-calorie foods allow you to feel fuller longer with fewer calories.

F - Food Safety

Similar to Risk in workout features. Evaluates the nutritional "safety" or quality of food based on its sugar, sodium, and cholesterol content. First, the 90th percentile of the sample is taken for each component to identify high values as potentially risky. The actual sugar, sodium, and cholesterol content is then normalized to this threshold (maximum 1), and the standard deviation is converted into a score where 1 represents optimal safety, and values less than 1 represent high levels of potentially harmful components. It allows for food quality to be taken into account, reducing the risk of excess sugar, salt, and cholesterol consumption, and promoting a healthier diet.

Sources:

WHO calls on countries to reduce sugars intake among adults and children

7. Methods, Algorithms, Metrics

Models used in the project:

1. K-Means clustering
2. Linear Regression as a Baseline Model
3. XGBoost Regression Models (Main model): Workout Model, Meal Model

Metrics:

1. Elbow method, Silhouette Score
2. RMSE on XGBoost

Methods:

1. One Hot Encoding
2. Standard Scaling for clustering
3. Train/Test/Validation split in 0.7/0.15/0.15 proportions
4. Randomized Search CV for hyperparameter tuning

5. Cosine similarity, TF-IDF vectors, MMR reranking

Overall workflow of the system for new user

1. The user first enters personal information, which is stored in `new_user.csv`.
2. Based on this information, the user is assigned to a cluster, and the predicted cluster_id is added to `new_user.csv`.
3. The main **preprocessed workout dataset** is enriched by computing the workout components **E, I, D, S, R** using predefined domain formulas.
4. These components are combined into a single **target variable**, whose weights depend on the user's goal (Loss, Maintain, or Gain).
5. This target is added as a new column to the **preprocessed workout dataset**, forming the **training dataset** where each row represents a past workout with known quality.
6. To prevent data leakage, all intermediate calculation variables and meal-related features are removed, leaving only valid user–workout attributes for model training.
7. The meal dataset is split into **training (70%)**, **validation (15%)**, and **test (15%)** sets, ensuring that model tuning and final evaluation remain independent.
8. For **prediction**, the enriched `new_user.csv` is **not merged into the training data**. Instead, the user profile is **repeated for every exercise**, and each exercise is scored individually using the trained model `workout_model.pkl`.
9. As a result, the training dataset is used to **learn general user–exercise patterns**, while the prediction dataset is constructed dynamically to **rank all possible exercises for one specific user**.
10. After the `workout_model.pkl` model predicts a **workout_score for every exercise** for the given user, the system **does not train anything further**; it switches from modeling to **recommendation logic**.
11. The top **100 highest-scoring exercises** are selected as candidates to ensure quality and reduce noise.
12. For these candidates, a **content-based representation** is built by combining textual exercise attributes (muscle group, equipment, body part, workout type, exercise name, etc.).
13. These text profiles are transformed into **TF-IDF vectors**, allowing exercises to be compared in a semantic vector space.

14. Predefined **training-day prototypes** (Legs, Push, Pull, Core) are also expressed as text and embedded into the same vector space.
15. Using **cosine similarity**, each exercise is assigned to the training day it best matches, creating structured workout categories instead of a flat ranked list.
16. Within each training day, **MMR (Maximal Marginal Relevance)** is applied to select exercises that are:

- highly relevant (high workout score),
- not redundant (low similarity to already selected exercises),
- and together meet the **calorie target per workout**.

17. This process is repeated for the required number of workouts, rotating through training-day types and aligning them with the user's **goal duration and workout frequency**.
18. The final output is a **day-by-day workout plan**, saved to `workout_plan.csv`, where machine-learning predictions ensure relevance and cosine similarity ensures **balance, structure, and diversity**.
19. For the meal pipeline, the system first constructs **meal-level features** (C, P, M, ED, F) that quantify how well each meal fits the user's nutritional needs, based on calories, macros, energy density, and food safety indicators.
20. These components are combined into a **single meal target score**, with weights that depend on the user's goal (Loss, Maintain, Gain), and this target is added to the main meal dataset to form the **training dataset**.
21. To prevent data leakage, all intermediate calculation variables and workout-related features are removed, leaving only valid user–meal attributes for model training.
22. The meal dataset is split into **training (70%)**, **validation (15%)**, and **test (15%)** sets, ensuring that model tuning and final evaluation remain independent.
23. A preprocessing pipeline is fitted **only on the training data**, applying one-hot encoding to categorical variables (meal type, diet type, cooking method, cluster_id, meal_name) and passing numerical features unchanged.
24. An XGBoost regression model is trained to learn how user characteristics and meal properties relate to the constructed meal target score.
25. During prediction, the trained meal model is applied by **repeating the enriched user profile for every meal**, producing a personalized **meal_score** for each available meal.

26. From these predictions, the system selects the **top 100 meals**, filtered by the user's diet type, to form a high-quality candidate set.
27. Each candidate meal is converted into a **text-based representation** (meal name, diet type, cooking method) and embedded using **TF-IDF**, enabling semantic comparison between meals.
28. Predefined **meal-time prototypes** (Breakfast, Lunch, Dinner, Snack) are embedded into the same vector space, and **cosine similarity** is used to assign each meal to its most suitable meal time.
29. Within each meal-time group, **MMR-based selection** balances three factors: high meal score (relevance), low similarity to already selected meals (diversity), and alignment with calorie targets.
30. Meals are then distributed across all goal days and daily meal slots, ensuring consistent coverage even when candidate meals are limited.
31. Portion sizes are initialized conservatively and then **iteratively adjusted** using a controlled set of allowed portion values to meet the user's total calorie target.
32. The final output is a **day-by-day meal plan** with meal timing, portion size, final calories, and macronutrients, saved to `meal_plan.csv`.
33. After both plans are generated, the system loads the finalized outputs: `meal_plan.csv`. , `workout_plan.csv`, and the enriched user profile from `new_user.csv`.
34. Before exporting, it cleans the presentation values by rounding key numeric columns so the report is readable.
35. The PDF report is created in **landscape A4** format with small margins to fit wide tables for meals and workouts.
36. The report begins with a **User Info** block (age, gender, weight, height, BMI, goal, target weight change), directly taken from user_df.
37. It then computes **summary totals**: total planned meal calories (sum of Calories_Final) and total planned workout calories burned (sum of Calories_Burned).
38. Using the user's **TDEE and GoalDays**, it estimates total baseline energy spent over the whole period, then adds workout calories burned to get **total energy spent**.
39. It compares total food intake vs total energy spent to compute an **actual calorie balance**, and converts this into an **estimated weight change** using 7700 kcal = 1 kg.
40. The summary section prints three comparisons:

- target total meal calories vs actual from the plan,
- target total workout calories to burn vs actual from the plan,
- target weight change vs estimated weight change from the plan.

41. The PDF is organized day-by-day: it builds the full set of days that appear in either the meal plan or workout plan, and iterates through them in order.
42. For each day, it extracts that day's meals and workouts and renders them into separate tables (or prints "No meals" / "No workouts" if empty).
43. Column widths are automatically calculated based on the longest text in each column, then scaled down if the table exceeds the available page width so the table always fits.
44. Tables are styled for clarity: header row highlighted (blue for meals, green for workouts), grid lines added, bold headers, and small font size for compact layout.
45. Finally, the full structured report is written into a single PDF file `plan.pdf` containing user info, summary metrics, and daily meal/workout tables.
46. The `check()` function is a console verification step: it reloads the saved CSV files and recomputes the same totals and estimated weight change to confirm that the exported plan matches the intended calorie and weight-change logic.

PDF example output:

User Info:
Age: 22
Gender: Female
Weight (kg): 56.0
Height (m): 1.7
BMI: 19.38
Goal: Loss
Target weight change (kg): 5.0
Summary:
Target meal calories: 45689.58 | Actual: 45717.41
Target workout burned calories: 19250.0 | Actual: 21403.34
Target weight change (kg): 5.0 | Actual: -5.28

Day 1

meal_name	diet_type	Calories_Final	day_label	Day	Portion	Proteins	Carbs	Fats
Steamed Paleo Snack	Low-Carb	496.25	Lunch	1	1.5	82.65	209.57	55.85
Steamed Paleo Snack	Low-Carb	450.0	Breakfast	1	1.5	82.67	208.57	55.87

No workouts

Day 2

meal_name	diet_type	Calories_Final	day_label	Day	Portion	Proteins	Carbs	Fats
Roasted Low-Carb Dinner	Low-Carb	1511.5	Lunch	2	2.0	112.06	279.54	74.98
Boiled Paleo Snack	Low-Carb	993.67	Breakfast	2	2.0	94.27	237.09	63.22

No workouts

...

Day 4

meal_name	diet_type	Calories_Final	day_label	Day	Portion	Proteins	Carbs	Fats
Raw Vegan Breakfast	Low-Carb	456.0	Breakfast	4	1.5	83.02	208.58	55.06
Boiled Paleo Snack	Low-Carb	580.12	Lunch	4	1.5	132.0	330.16	87.64

Workout_Type	Workout	Name of Exercise	Body Part	Equipment Needed	Sets	Reps	Benefit	Calories_Burned	Day
HIIT	Leg raises	Tricep Extensions	Forearms	Cable Machine	5	18	Improves core rotation strength	1468.09	4
HIIT	Wrist curl	Calf Raises	Forearms	Bench, Barbell	4	19	Improves posture and back strength	859.1	4

Workout Model example results:

[Baseline] Validation RMSE (Linear Regression): 0.074544

[Manual HP] Best n_estimators: 500

[Manual HP] Validation RMSE: 0.049358

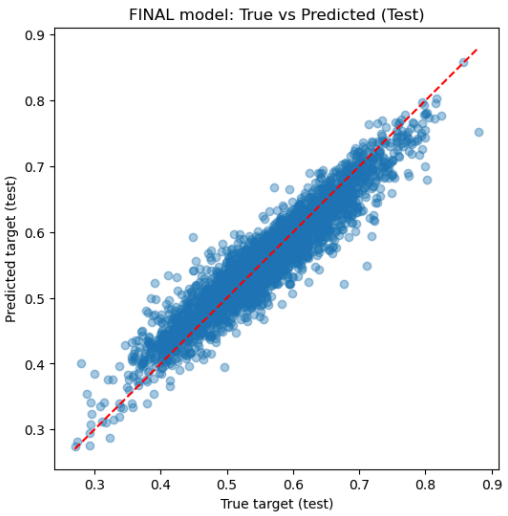
[Manual HP] Test RMSE: 0.050557

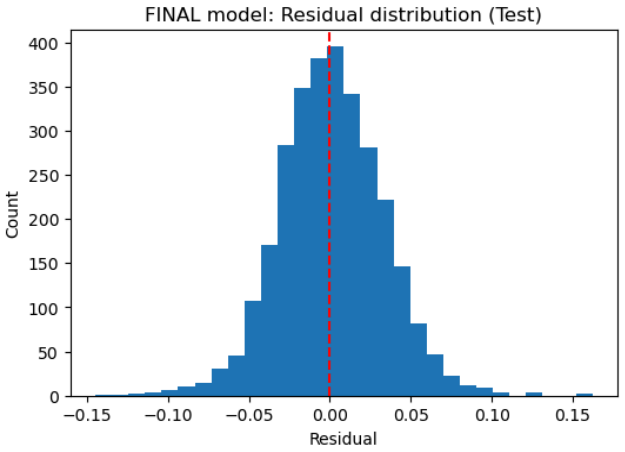
[Tuned HP] Best CV params: {'subsample': 1.0, 'reg_lambda': 5.0, 'reg_alpha': 0.0001, 'n_estimators': 500, 'min_child_weight': 10, 'max_depth': 8, 'learning_rate': 0.2, 'gamma': 0, 'colsample_bytree': 0.6}

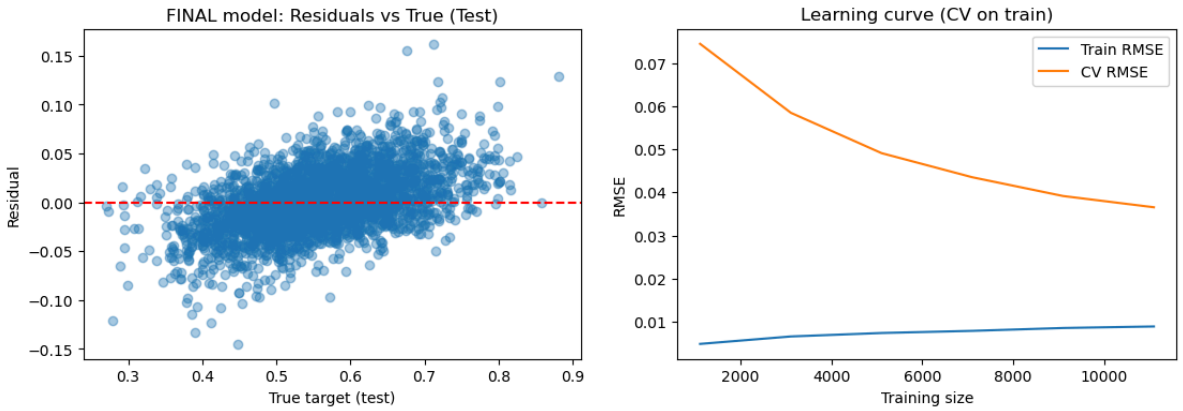
[Tuned HP] Best CV RMSE: 0.036428

[Tuned HP] Validation RMSE (holdout check): 0.033019

[FINAL] Test RMSE (trained on train+val): 0.032145







The workout model shows a clear and consistent improvement from the linear baseline to the tuned XGBoost model, with RMSE decreasing from 0.0745 to 0.0321 on the test set. The true vs. predicted plot indicates strong alignment along the diagonal, while the residual distribution is approximately centered around zero, suggesting low bias. Residuals do not show strong systematic patterns across the target range, and the learning curve demonstrates that validation error steadily decreases as more data is used, indicating good generalization rather than overfitting.

Meal Model example results:

Validation RMSE (Linear Regression): 0.0670849

Best n_estimators: 500

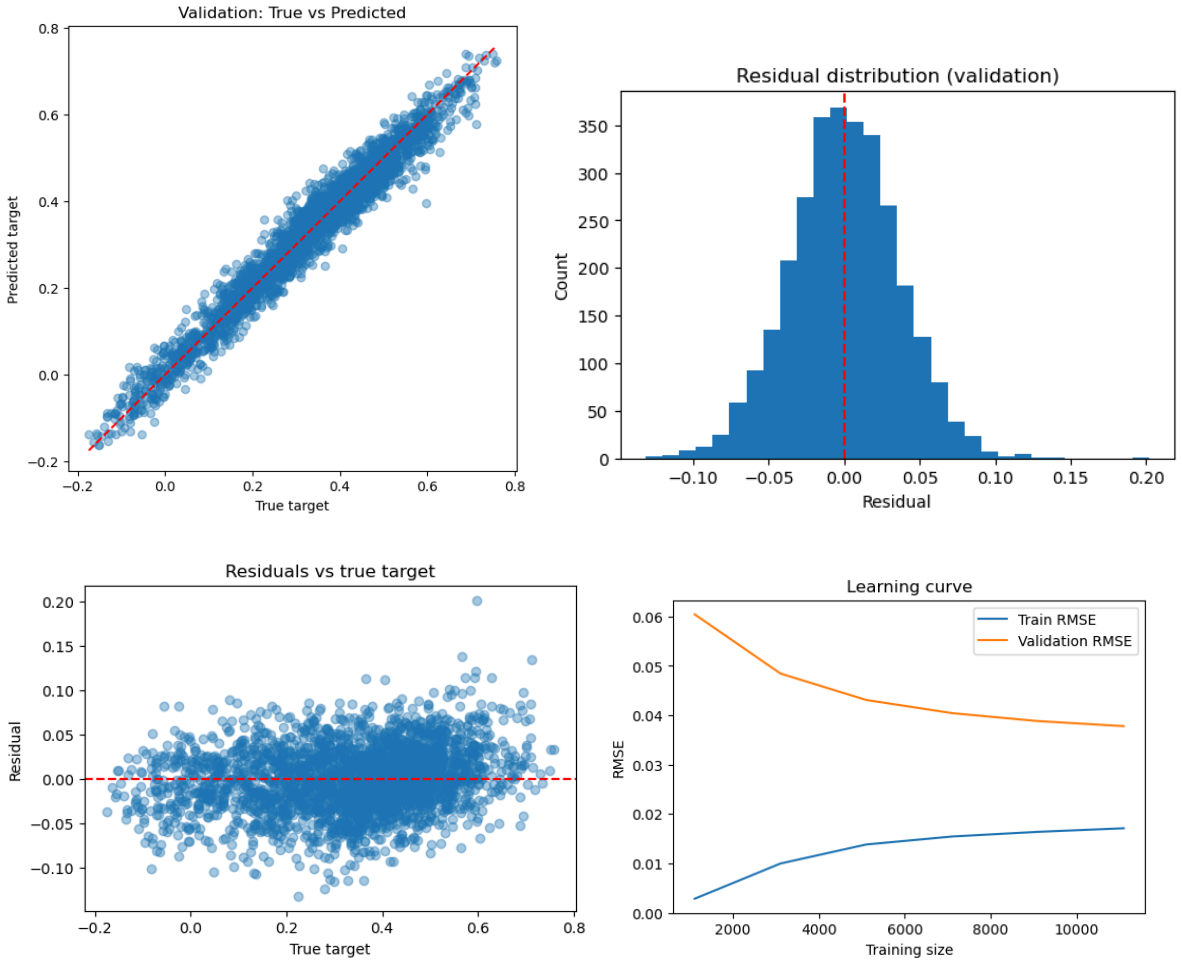
Validation RMSE: 0.0439

Test RMSE: 0.0449

Best CV params: {'subsample': 1.0, 'reg_lambda': 5.0, 'reg_alpha': 0.0001, 'n_estimators': 500, 'min_child_weight': 10, 'max_depth': 8, 'learning_rate': 0.2, 'gamma': 0, 'colsample_bytree': 0.6}

Best CV RMSE: 0.037644144527335735

Validation RMSE after tuning: 0.035947



The meal model follows a similar pattern, improving from a linear baseline RMSE of 0.0671 to a tuned validation RMSE of 0.0359. Predicted values closely track the true targets, and residuals are symmetrically distributed around zero, indicating stable predictions across different meal scores. The learning curve shows convergence between training and validation errors as dataset size increases, suggesting that the model effectively captures user–meal relationships while maintaining generalization performance.

More detailed look on each model.

KMeans Clustering (2_Clustering.ipynb)

Input file: `data/preprocessed_without_scaling.csv`

Input user features

	Age	Gender	Weight (kg)	Height (m)	BMI	Experience_Level	Workout_Frequency (days)	Daily meals frequency	diet_type
0	35	Male	65.27	1.62	24.87	2	4	3	Vegan
1	23	Female	56.41	1.55	23.48	2	4	3	Vegetarian
2	33	Female	58.98	1.67	21.15	1	3	2	Paleo

Output file: `data/dataset_with_clusters.csv`

Output: `cluster_id` column added to `preprocessed_without_scaling.csv`

Dataset preprocessing

- Standard Scaler for numerical columns
- OneHot Encoding for ‘Gender’, ‘diet_type’

Functions

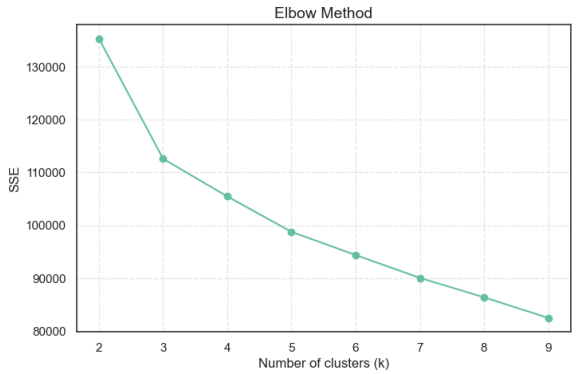
`find_optimal_k` - computes SSE (Elbow) and Silhouette scores for K-Means clustering and returns the best k based on silhouette score.

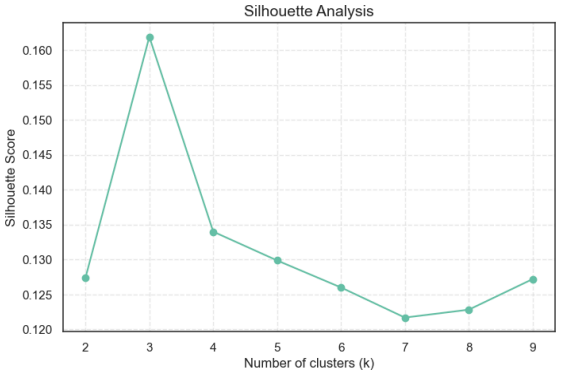
Two versions of clustering were created.

Version 1

K-Means clustering with diet_type and Gender

```
Best k based on silhouette score: 3
Silhouette score at best k: 0.162
```

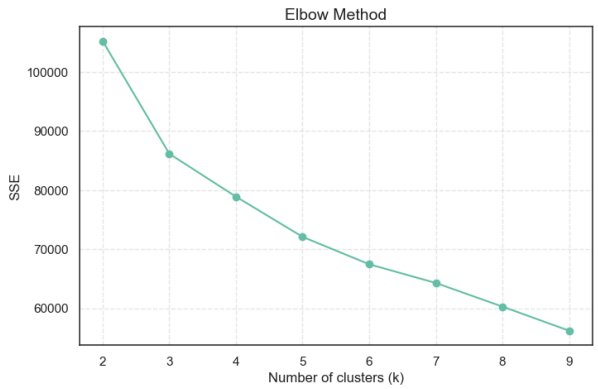


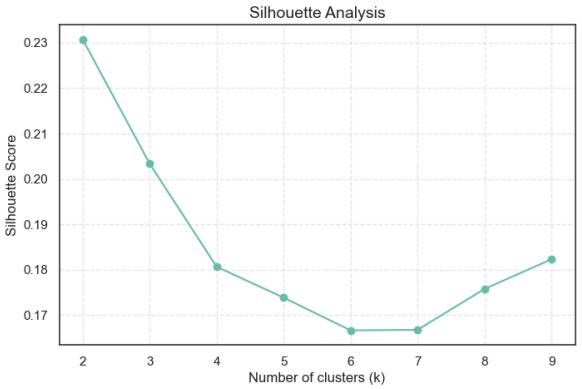


Version 2

K-Means clustering without diet_type and Gender

```
Best k based on silhouette score: 2
Silhouette score at best k: 0.231
```



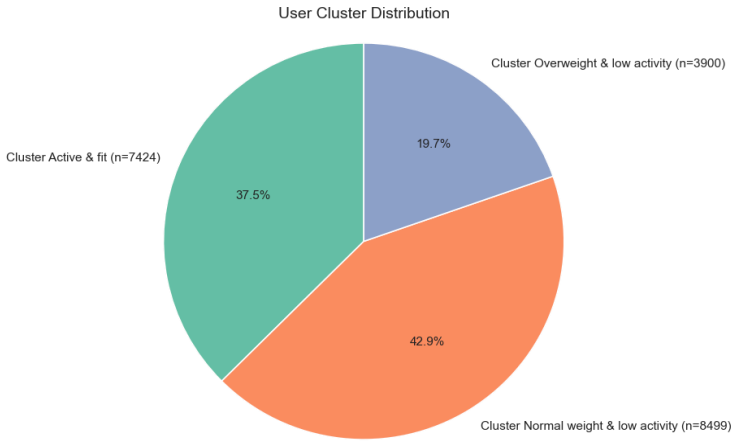


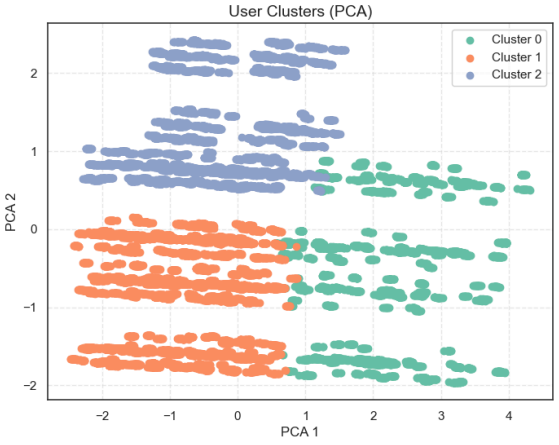
Final version

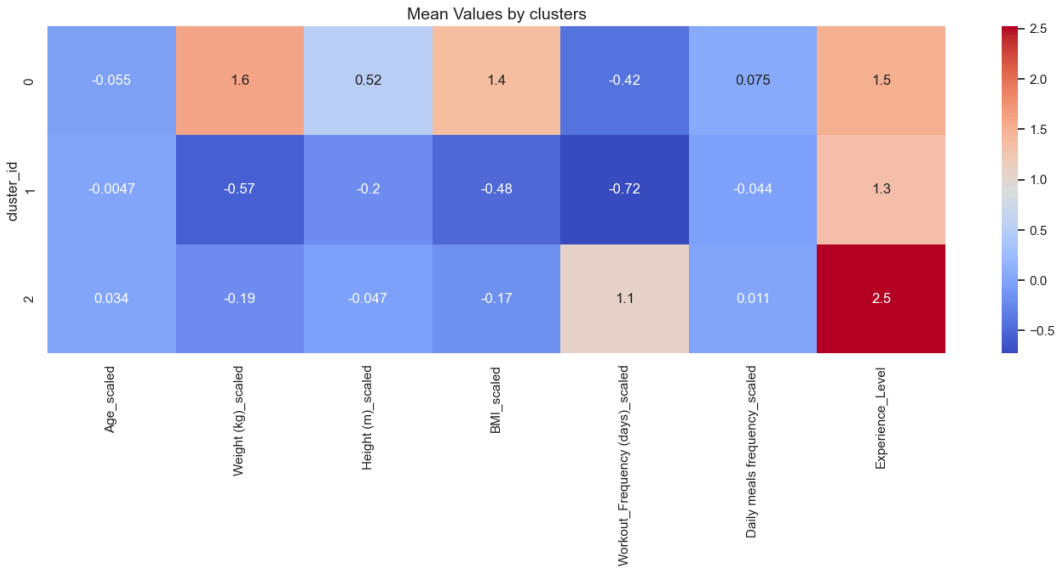
Although the silhouette score is maximized at k = 2 in the Version 2, the elbow method consistently indicates k = 3 in both Version 1 and Version 2. Also k = 3 provides more interpretable number of user segments from business logic, so we decided to take the model with Silhouette = 0.231 from Version 2, but manually state k = 3.

Result

The clustering results reveal three distinct user groups primarily differentiated by **BMI** and **workout frequency** rather than age. This suggests that lifestyle and physical activity play a stronger role in user segmentation than demographic factors.







Interpretation

This heatmap shows three distinct user groups based on standardized averages of body metrics, behavior, and experience. **Cluster 1 Normal weight & low activity (42.9%)** represents beginners: they have below-average weight, BMI, workout frequency, and the

lowest experience level, showing low physical activity and early fitness stages. **Cluster 0 Overweight & low activity (19.7%)** includes more experienced users with above-average weight, height, and BMI but still relatively low workout frequency, those are people with stronger or heavier body profiles who train inconsistently. **Cluster 2 Active & fit (37.5%)** captures advanced and highly active users: they train much more frequently and have by far the highest experience level and maintain slightly leaner body metrics. Overall, the clustering is driven mainly by **training behavior and experience**, **not age/gender**.

KMeans clustering Evaluation

```
Silhouette stability over random initializations:
  Seed  0: 0.2033800
  Seed  1: 0.2033800
  Seed  2: 0.2033692
  Seed  3: 0.2033601
  Seed  4: 0.2033725
  Seed  5: 0.2033861
  Seed  6: 0.2033800
  Seed  7: 0.2033732
  Seed  8: 0.2033671
  Seed  9: 0.2033750

Summary:
  Mean silhouette score: 0.20337433323747073
  Std deviation:         7.216768141512571e-06
```

The silhouette scores remain approximately the same across different random initializations. It means that the clustering separation is stable and not sensitive to the choice of seed. The mean silhouette value of approximately 0.20 shows that while the groups overlap to some extent, the structure captured by the clustering is consistent and reproducible.

Workout Model (Test_Workout_model.ipynb)

Type: XGBoost

In this project, XGBoost is used because the target outcome depends on **complex, non-linear interactions** between body metrics, workout behavior, experience level, and lifestyle features that simpler models (like linear regression) cannot capture well. Additionally, its regularization and tree-based structure help control overfitting while delivering stable performance on relatively small to medium-sized datasets, making it well suited for personalized fitness and recommendation-driven predictions in the system.

Creation of Training "Workout Dataset"

Function 1: `create_workout_features`

```
E - Energy Consumption
I - Intensity
S - Power component
D - Duration
R - Risk (1 - penalties)
```

1. Calculates Workout target variables (E, I, S, D, R) via pre-defined formulas.
2. Drops the features below from the main dataset to avoid data leakage.

Data Leakage Prevention:

We exclude the following features from the workout dataset:

- helper variables used in calculation of target variables: `"E_raw", "E_eff", "pct_HRR", "Session_Duration (hours)", "pen_age", "pen_bmi", "pen_hrr", "pen_skill"`
- user features not participating in workout model training: `"BMR", "PAL", "TDEE", "CalorieChange", "CaloriesToBurnTraining", "CaloriesReducedFromFood", "CaloriesPerDay", "TotalWorkouts", "CaloriesPerWorkout"`
- meal related features not helping the workout model in training: `"cooking_method","meal_type", "Calories", "serving_size_g", "sugar_g", "sodium_g", "cholesterol_g", "Carbs", "Proteins", "Fats"`

Function 2: `build_and_train_workout_model`

1. Calculates the target for each row by using different formulas depending on the goal of the user and adds the 'target' to the dataset.

Weight Loss

$0.45*E + 0.25*I + 0.10*D + 0.05*S + 0.15*R$

Maintain

$0.25*E + 0.20*I + 0.15*D + 0.20*S + 0.20*R$

Gain Weight

$0.05*E + 0.15*I + 0.10*D + 0.50*S + 0.20*R$

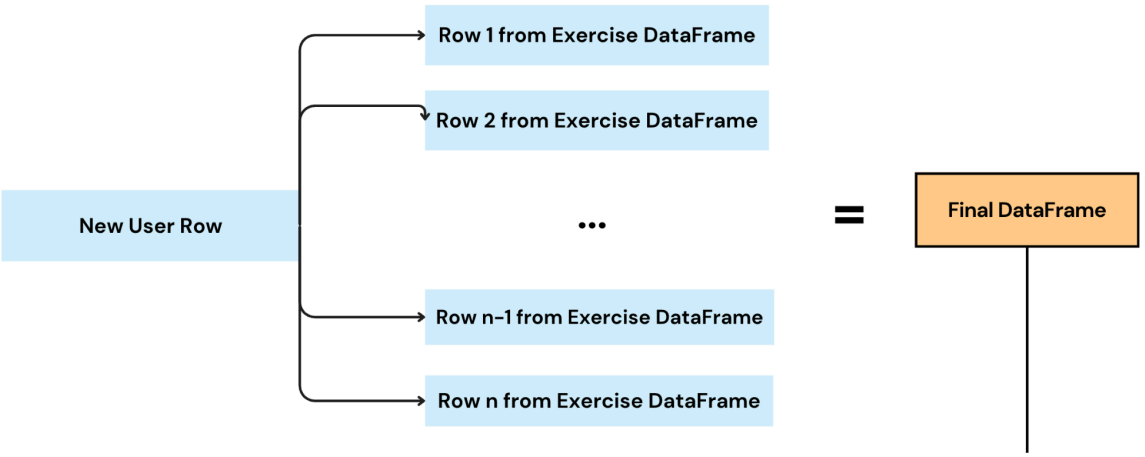
2. Trains Linear Regression model as a Baseline Model.
3. Trains XGBoost with manual hyperparameters.
4. Conducts Randomized Search CV `(cv=5, n_iter=40)`.
5. Trains XGBoost on tuned hyperparameters.
6. Saves the encoder and the trained model into `encoders/workout_encoder.pkl` and `models/workout_model.pkl`.

Creation of Testing "Workout Dataset"

Function 3: `predict_workout_score`

1. Creates exercise dataframe consisting from the features related to workout: `'Sets', 'Reps', 'rating', 'Workout_Type', 'Name of Exercise', 'Benefit','Target Muscle Group', 'Equipment Needed', 'Body Part', 'Type of Muscle', 'Workout', 'Calories_Burned'`

2. Creates a user–exercise pairing by repeating the user profile for every exercise, enabling personalized scoring of each option.



3. Runs `workout_model.pkl` on final dataframe.
4. Adds predicted workout scores to each row of final dataframe.

Cosine Similarity

Function 4: `run_cosine_similarity_workout`

1. Generates workout candidates by selecting the top 100 best-scoring exercises from the above dataset with predicted workout scores.
2. Builds a text profile for each exercise (TEXT_COLS) and converts it into TF-IDF vectors.

```
TEXT_COLS = ["Benefit", "Target Muscle Group", "Equipment Needed", "Workout", "Body Part", "Type of Muscle", "Workout_Type", "Name of Exercise"]
```

3. Uses MMR reranking to select exercises per day that are high-quality (high score), diverse (not redundant), and fit the calorie target.
4. Constructs a multi-day workout plan based on the user's total workouts and goal duration, then exports the final plan to `data/workout_plan.csv`.

Meal Model (Test_Meal_model.ipynb)

Type: XGBoost

Creation of Training "Workout Dataset"

Function 1: `create_meal_features`

`C - Calorie Fit`

P - Protein Per Meal

M - Macro Match

ED - Energy Density

F - Food Safety

1. Calculates Meal target variables (C, P, M, ED, F) via pre-defined formulas.
2. Creates ‘meal_name’ feature by combining ‘cooking_method’, ‘diet_type’ and ‘meal_type’.
3. Drops the features below from the main dataset to avoid data leakage.

Data Leakage Prevention:

We exclude the following features from the meal dataset:

- helper variables used in calculation of target variables: `"Calories_Burned", "Workout_Frequency (days)", "cholesterol_g", "sodium_g", "sugar_g", "Calories", "Daily meals frequency", "serving_size_g", "pct_p", "pct_c", "pct_f","cal_from_protein", "cal_from_carbs", "cal_from_fats"`
- user features not participating in workout model training: `"BMR", "PAL", "TDEE", "CalorieChange", "CaloriesToBurnTraining", "CaloriesReducedFromFood", "CaloriesPerDay", "TotalWorkouts", "CaloriesPerWorkout"`
- workout related features not helping the workout model in training: `'Sets', 'Reps', 'rating', 'Workout_Type', 'Name of Exercise', 'Benefit', 'Target Muscle Group', 'Equipment Needed', 'Body Part', 'Max_BPM', 'Avg_BPM', 'Resting_BPM', 'Session_Duration (hours)','Type of Muscle', 'Workout', 'Experience_Level', 'Difficulty Level', 'Burns Calories (per 30 min)'`

Function 2: `build_and_train_meal_model`

1. Calculates the target for each row by using different formulas depending on the goal of the user and adds the ‘target’ to the dataset.

Weight Loss

$0.35*C + 0.25*P + 0.15*M + 0.15 * ED + 0.10*F$

Maintain

$0.30*C + 0.25*M + 0.20*P + 0.10*ED + 0.15*F$

Gain Weight

$0.40*C + 0.30*P + 0.15*M + 0.10*ED + 0.05*F$

2. Trains Linear Regression model as a Baseline Model.
3. Trains XGBoost with manual hyperparameters.
4. Conducts Randomized Search CV `(cv=5, n_iter=40)`.
5. Trains XGBoost on tuned hyperparameters.
6. Saves the encoder and the trained model into `encoders/meal_encoder.pkl` and `models/meal_model.pkl`.

Creation of Testing "Meal Dataset"

Function 3: `predict_meal_score`

1. Extracts meal-related attributes (meal type, cooking method, meal name, water intake) from the dataset.
2. Creates a user–meal pairing by repeating the user profile for every meal, enabling personalized scoring of each option.
3. Applies the saved model and encoder to encode features consistently with training.
4. Uses the trained meal prediction model to compute a meal suitability score for each meal.
5. Attaches nutritional information (calories, proteins, carbs, fats) to the results for cosine similarity input.

Cosine Similarity

Function 4: `run_cosine_similarity_meal`

Function 5: `adjust_portions_per_meal`

1. Filters meals to match the user's diet_type, then selects the top 100 highest-scoring meals as candidates.
2. Builds text profile for each candidate meal (TEXT_COLS) and converts it into TF-IDF vectors.
   `TEXT_COLS = ["meal_name", "diet_type", "cooking_method"]`
3. Uses cosine similarity to assign each meal to the most suitable meal time label (Breakfast/Lunch/Dinner/Snack) based on text similarity to those prototypes.
4. Applies an MMR-style selection (relevance and diversity) to pick meals that are high-scoring, not too similar to each other, and that accumulate toward the calorie target.
5. Constructs a day-by-day plan for all 'GoalDays', allocating 'Daily meals frequency' meals per day.
6. Adjusts portions by scaling calories initially (sets portion to 0.5 and halves calories), then calls `adjust_portions_per_meal` to match the overall target calories.
7. Outputs a clean table (meal info, macros, day labels, portions) and saves it to `data/meal_plan.csv`.

8. Limitations and Potential Improvements

While the proposed system demonstrates a robust and well-structured end-to-end personalization pipeline, one important direction for future improvement lies in expanding the diversity and depth of user-level data available to the models. In the current project, the construction of workout and meal target variables relies on a carefully selected but finite set of interpretable lifestyle, physical, and nutritional features. This design choice ensures transparency, controllability, and strong generalization, but it also constrains the system's ability to capture more subtle, long-term behavioral patterns such as habit formation, adherence dynamics, recovery trends, or temporal changes in user performance. Incorporating longitudinal user data, real-time feedback signals, or richer physiological measurements (e.g., sleep quality, stress indicators, wearable time-series data) could allow future versions of the system to move beyond static personalization toward adaptive, continuously learning recommendations. Importantly, such extensions would not replace the current framework, but rather build upon its modular architecture, enabling more fine-grained

personalization while preserving the system’s interpretability, safety constraints, and alignment between nutrition and physical activity planning.