# MLCB25 Assignment 1 Report

BMI Prediction Using Gut Microbiome Data

Defteraiou Maria
Student ID: 7115152400003

April 7 2025

## 1 Introduction

The human gut microbiome has emerged as a key player in health and disease, influencing various physiological processes that include immunity, metabolism, and energy balance. In particular, increasing evidence suggests a strong association between intestinal microbial composition and obesity-related traits, such as body mass index (BMI). This assignment explores whether metagenomic data, specifically the relative abundances of bacterial species in the human gut, can be used to accurately predict BMI using supervised machine learning techniques.

The main goal of this project is to develop and evaluate regression models that predict a subject's BMI based on their gut microbiome composition. This is accomplished through a structured machine learning pipeline that includes data pre-processing, model training, feature selection, hyperparameter tuning, and evaluation. The final objective is to identify the best performing and most interpretable model that generalizes well to unseen data.

Two data sets were provided for this assignment:

- **Development dataset:** Consists of 489 samples and 141 columns.
- **Evaluation dataset:** Consists of 211 samples and 141 columns.

Each dataset contains:

- **6 columns of metadata** (e.g., subject ID, age, gender, location, etc.)
- **1 column with BMI**, the regression target.
- **134 columns representing bacterial species** in the gut microbiome.

No preprocessing had been performed on the data prior to delivery. Therefore, one of the initial steps in this project involved careful data cleaning and structuring to ensure compatibility with machine learning workflows. Since high-dimensional biological data such as microbiome profiles often include noise and redundancy, preprocessing and feature selection play a critical role in improving model performance and interpretability.

The scope of this report includes a comprehensive breakdown of the data pipeline, the models developed, and the evaluation results. Through baseline modeling, dimensionality reduction, and evaluation on a hold-out dataset, we aim to uncover which regression technique performs best in predicting BMI from gut microbiota, while also reflecting on the biological plausibility and generalizability of the selected features.

# 2 Methods

## 2.1 Development Environment

The code was developed using Visual Studio Code connected to a Linux WSL environment. Python and Git were installed, and a GitHub repository was used for version control.

## 2.2 Data Exploration and Cleaning

To prepare the data sets for model training, we followed a structured data cleaning and exploration workflow, implemented in the `notebooks/data_exploration.ipynb` . This stage was critical to ensure that the features used in model development were relevant, clean, and interpretable.

### 2.2.1 Dataset Overview

The original datasets consisted of:

- A **development set** containing 489 samples and 141 columns.
- An evaluation set containing 211 samples and 141 columns.

Each data set included six metadata columns, 1 target variable column (BMI) and 134 columns corresponding to gut bacterial species.

### 2.2.2 Missing Values and Irrelevant Columns

Our first step was to inspect the data structure using exploratory commands such as `df.shape`, `df.head()`, and `df.describe()`. We verified that there were no missing values in either dataset, thereby avoiding the need for imputation.

We then examined the column relevance to the regression task. The following metadata columns were removed:

- `Unnamed: 0`
- `Project ID`
- `Experiment type`
- `Sex`
- `Host age`
- `Disease MESH ID`

These metadata fields were excluded because they were either identifiers or categorical variables with unclear encoding. Including them without careful pre-processing could introduce noise or bias into the model. Instead, we focused on using only the target variable (BMI) and the bacterial species profiles, resulting in a clean dataset with 135 columns.

### 2.2.3 Data Export

After cleaning, the processed datasets were saved for downstream tasks as:

- `development_final_data.csv`
- `evaluation_final_data.csv`

These files were stored in the `data` directory and used for all subsequent model training and evaluation tasks.

### 2.2.4 Exploratory Visualizations

To better understand the structure and relationships within the development dataset, we generated several visualizations:

- **BMI distribution:** Provided insight into the target variable's range and skewness.

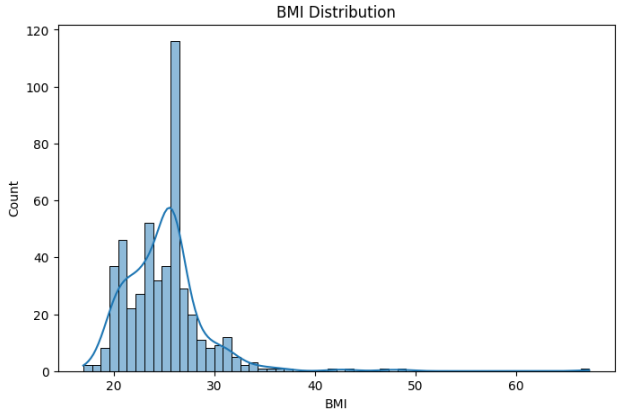


Figure 1: Distribution of BMI in the development dataset

Figure 1, shows a histogram of the BMI distribution across the development dataset. The distribution is right-skewed, with the majority of values concentrated between 20 and 30, indicating that most individuals fall into the normal or overweight categories. A peak is observed around BMI 27, with a long tail extending toward higher BMI values, suggesting a small number of outliers in the obese category. This skewness could influence regression model performance and motivates careful model selection and validation.

- **BMI and bacteria correlation heatmap:** Highlighted potential associations between specific species and BMI, informing feature selection strategies.

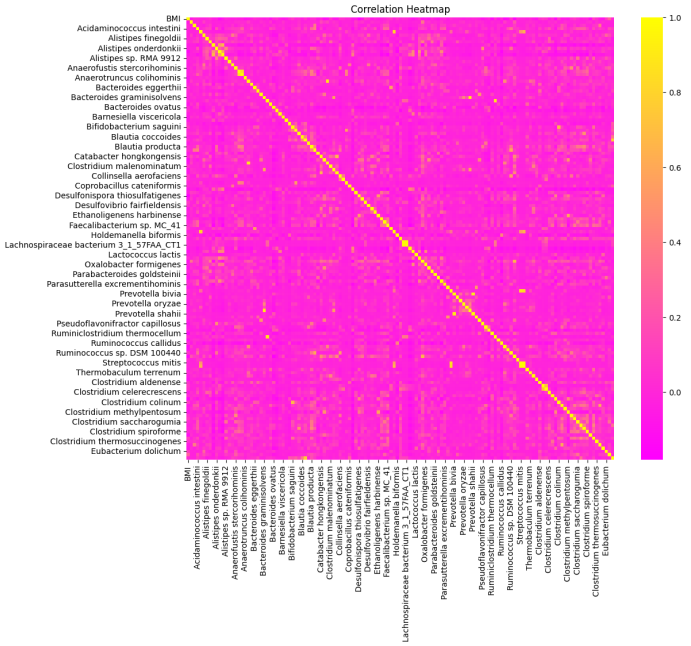


Figure 2: Heatmap showing correlations between BMI and bacterial species abundances.

In Figure 2, a heatmap of Pearson correlation coefficients is shown for all pairs of variables, including BMI and bacterial species. The diagonal shows perfect correlation (value = 1), and lighter colors indicate stronger correlations. While most species exhibit weak or no correlation with BMI, a few species display moderate positive or negative relationships. These insights are crucial for feature selection, allowing us to prioritize species that show potential predictive power for BMI.

- **Top 15 most abundant bacterial species:** Helped identify dominant taxa in the population, which may have biological relevance in BMI regulation.

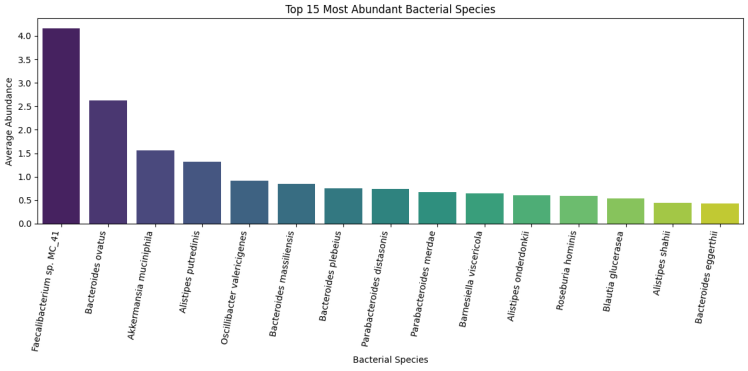


Figure 3: Top 15 most abundant bacterial species in the development dataset.

Figure 3 displays the top 15 most abundant bacterial species in the development dataset. *Faecalibacterium sp. MC_41* and *Bacteroides ovatus* are the most prevalent species, followed by *Akkermansia muciniphila*, which has been previously associated with metabolic health.

The abundance of these species may play an important role in BMI regulation and thus justify their inclusion as candidate predictors in the regression models.

These visuals not only supported biological interpretability but also informed later decisions on model complexity and feature engineering.

## 2.3 Model Development

### 2.3.1 Overview of Regression Algorithms

In this assignment, we developed and compared three different regression models for predicting BMI based on gut microbiome composition:

- **Elastic Net Regression:** Combines L1 and L2 regularization to perform both feature selection and coefficient shrinkage. It is particularly suitable for datasets with multicollinearity or when the number of predictors exceeds the number of observations.
- **Support Vector Regression (SVR):** A non-linear regression model that uses kernel functions to capture complex relationships between input features and the target variable. While it can achieve high performance, it lacks inherent interpretability.
- **Bayesian Ridge Regression:** A probabilistic linear model that places priors on the model coefficients and computes a posterior distribution. It offers regularization, robustness to overfitting, and provides uncertainty estimates for predictions.

### 2.3.2 Basic Code Structure

To ensure reproducibility and scalability, we structured our codebase into a modular workflow defined in `src/functions.py`. This script includes:

- `data_preprocessing` function: Loads the dataset, performs cleaning, and splits it into features ($X$) and target ($y$).
- `load_data` function: Loads the dataset and splits it into $X$ and $y$.
- `BaseRegressor` class:
  - constructor: accepts arguments such as $X$, $y$, evaluation $X$, evaluation $y$, model type, and save directory.
  - `evaluate` method: returns evaluation key metrics and boxplots.
  - `save` method: saves the trained model class instance.
- `BaselineRegressor` class: Inherits from `BaseRegressor` and implements a basic `train` method.
- `FeatureSelectionRegressor` class: Inherits from `BaseRegressor`, performs feature selection, and includes a `train` method.
- `TuningRegressor` class: Inherits from `BaseRegressor`, performs the same feature selection, and includes hyperparameter tuning with 5-fold cross-validation in the `train` method.

This modular approach allows consistent application of the same workflow across all models and supports future extensions, such as integrating new algorithms or tuning strategies.

### 2.3.3 Baseline Establishment

The first step in our modeling pipeline was to establish baseline models for each regression algorithm using:

- All bacterial species as input features,
- Default hyperparameters from `scikit-learn`,
- No feature selection or tuning.

These baseline models served as performance benchmarks. The development dataset was used entirely for training, while the evaluation dataset was reserved for assessing generalization ability.

### 2.3.4 Feature Selection

To improve model performance and reduce complexity, we performed feature selection using the `r_regression` function from `scikit-learn`. The goal was to identify a minimal subset of bacterial species that contributed most to BMI prediction. We applied a correlation threshold of 0.1, retaining only bacterial species with an absolute correlation greater than or equal to 0.1.

The models were retrained using the selected features and evaluated again to assess the impact of dimensionality reduction.

### 2.3.5 Hyperparameter Tuning

Following feature selection, we performed hyperparameter tuning for each model using `GridSearchCV` with 5-fold cross-validation. This approach ensured a robust estimation of performance during tuning.

Each model was tuned over the following parameter grids:

- **Elastic Net:** `alpha`, `l1_ratio`, `max_iter`, `fit_intercept`, `selection`
- **SVR:** `C`, `epsilon`, `kernel`, `degree`, `gamma`, `shrinking`
- **Bayesian Ridge:** `alpha_1`, `alpha_2`, `lambda_1`, `lambda_2`, `fit_intercept`, `compute_score`, `tol`

The best parameter combination was selected based on minimizing the Root Mean Squared Error (RMSE).

### 2.3.6 Evaluation Metrics

Model performance was assessed on the hold-out evaluation dataset using the following metrics:

- **Root Mean Squared Error ($RMSE$):** Measures the square root of the average squared prediction error.
- **Mean Absolute Error ($MAE$):** Provides a more interpretable measure of average absolute error.
- **R-squared ($R^2$):** Represents the proportion of variance in BMI explained by the model.

To account for randomness and ensure reproducibility, we fixed the random seed to 42 throughout the pipeline. The evaluation dataset was split using `ShuffleSplit` with 50 iterations and a 20% test size. Evaluation metrics were collected over all splits to generate confidence intervals for fair comparison between the baseline, feature-selected, and tuned models.

### 2.3.7 Model Analysis

In the `notebooks/model_analysis.ipynb` notebook, we import the core codebase from `functions.py` to perform training, evaluation, and selection of the best model. A nested loop iterates over each regression algorithm and each modeling stage (baseline, feature selection, tuning). For every iteration, the model is trained, evaluated across all metrics, and the corresponding class instance is saved to the `models` directory.

During this process, we also perform model selection and store the best-performing instances in the `final_models` directory. At each evaluation step, we generate boxplots for RMSE, MAE, and $R^2$ to rigorously compare performance.

### 2.3.8 Best Model Selection, Training, and Evaluation

Best model selection was performed in two phases. The selection criterion was the minimum RMSE, which correlates well with overall predictive performance.

- **Phase 1:** Select the best stage (baseline, feature selection, tuning) for each of the three algorithms.
- **Phase 2:** Among the three best-stage models, choose the one with the lowest RMSE as the overall winner.

The winning model is retrained on the combined development and validation datasets. Finally, we construct a pipeline to apply the model on unseen data. The `data_preprocessing` function is used to transform the raw test set, and the model named `"winner"` is loaded and evaluated using the `evaluate` method.

# 3 Results

## 3.1 Overview

This section presents the results from the three modeling stages—**baseline**, **feature selection**, and **hyperparameter tuning**—across all regression algorithms: Elastic Net, Support Vector Regression (SVR), and Bayesian Ridge Regression. We report the mean RMSE, MAE, and $R^2$ values across 50 random splits. Model performance variability is illustrated using boxplots, and the winning model is selected based on lowest RMSE while considering complexity and robustness.

## 3.2 Baseline Models

The table below shows the performance of the baseline models trained on all available features with default hyperparameters.

Table 1: Evaluation Metrics – Baseline Models

| Metric | Elastic Net | SVR | Bayesian Ridge |
|---|---|---|---|
| RMSE | 3.935 | 3.810 | **3.712** |
| MAE | 2.854 | **2.497** | 5.553 |
| $R^2$ | -0.007 | 0.062 | **0.099** |

Bayesian Ridge achieved the lowest RMSE and highest $R^2$, indicating strong overall performance, despite an unusually high MAE—likely due to sensitivity to outliers. SVR performed competitively, while Elastic Net underperformed in terms of $R^2$.

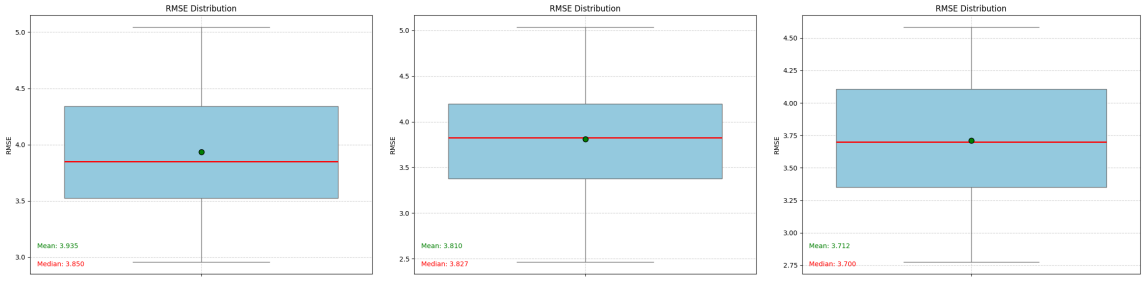


Figure 4: RMSE distribution boxplots for baseline models: Elastic Net (left), SVR (middle), Bayesian Ridge (right).

### 3.3 Feature Selection Models

Feature selection reduced model complexity by retaining only features with absolute correlation $\geq 0.1$ with BMI. The updated metrics are:

Table 2: Evaluation Metrics – Feature Selection Models

| Metric | Elastic Net | SVR | Bayesian Ridge |
|---|---|---|---|
| RMSE | 3.935 | **3.803** | 3.812 |
| MAE | 2.854 | **2.481** | 2.645 |
| $R^2$ | -0.007 | 0.061 | **0.040** |

SVR maintained the best performance. Bayesian Ridge lost some $R^2$ compared to the baseline but gained in MAE. Elastic Net's performance remained largely unchanged.

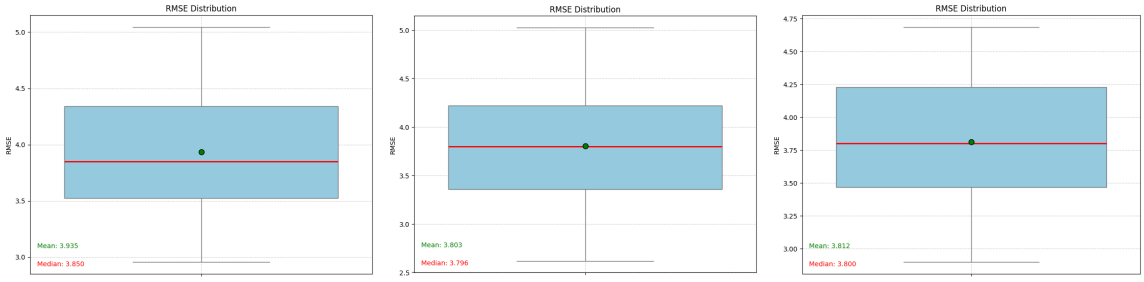


Figure 5: RMSE distribution boxplots after feature selection: Elastic Net (left), SVR (middle), Bayesian Ridge (right).

### 3.4 Tuned Models

The final modeling stage involved hyperparameter tuning using grid search with 5-fold cross-validation. The best-tuned results are:

SVR tuning led to the best $R^2$ and strong RMSE, slightly outperforming other models in generalization. Elastic Net tuning improved MAE but still showed moderate $R^2$. Bayesian Ridge

Table 3: Evaluation Metrics – Tuned Models

| Metric | Elastic Net | SVR | Bayesian Ridge |
|---|---|---|---|
| RMSE | **3.791** | 3.776 | 3.812 |
| MAE | **2.672** | 2.539 | 2.645 |
| $R^2$ | 0.061 | **0.069** | 0.040 |

remained consistent with its previous stages.

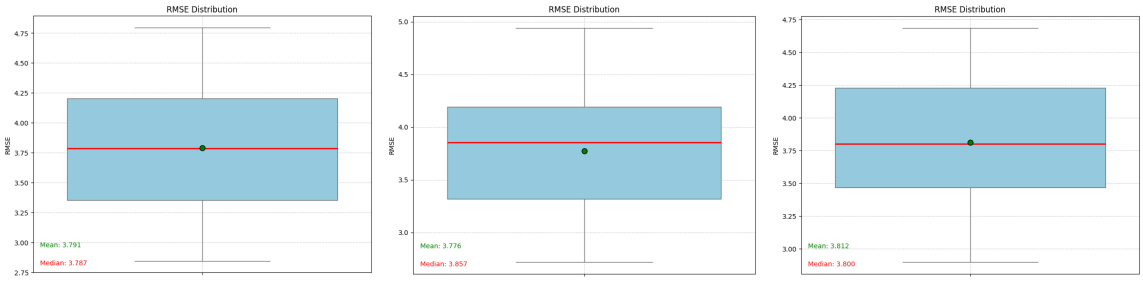


Figure 6: RMSE distribution boxplots after hyperparameter tuning: Elastic Net (left), SVR (middle), Bayesian Ridge (right).

### 3.5 Model Comparison and Selection

To identify the best configuration for each regression algorithm, we selected the top-performing model from each stage (baseline, feature selection, tuning) based on the lowest average RMSE across 50 ShuffleSplit evaluations. The best stage per algorithm was:

- **Elastic Net:** Hyperparameter tuning
- **SVR:** Hyperparameter tuning
- **Bayesian Ridge:** Baseline

Despite the expectations that feature selection or tuning would lead to measurable improvements, the Bayesian Ridge model achieved its best performance in the baseline stage. This suggests that the default regularization priors and inherent robustness of the Bayesian approach were already well-suited to the underlying data distribution. In contrast, both Elastic Net and SVR benefited from tuning, likely due to their sensitivity to hyperparameter values such as $\alpha$, $l1\_ratio$ (Elastic Net), and $C$, $\epsilon$, and kernel type (SVR).

The final model was selected from these three best-performing configurations, again using average RMSE as the principal selection criterion. Although SVR tuning showed slight gains in $R^2$ and competitive RMSE, the Bayesian Ridge baseline ultimately demonstrated the best trade-off between low error, model stability, and generalization capacity.

**Final Selected Model:** *Bayesian Ridge Regression (Baseline).*

This model had the lowest mean RMSE (3.712), highest $R^2$ (0.099), and exhibited minimal performance variability across the 50 resampling splits. These characteristics suggest that it generalized better and avoided overfitting, even without tuning or feature reduction.

### 3.6 Model Robustness and Interpretability

In addition to raw performance metrics, we also evaluated the models in terms of robustness, interpretability, and practical deployability.

**Bayesian Ridge Regression** is inherently robust due to its probabilistic formulation, which imposes Gaussian priors on the model coefficients. This regularization helps stabilize learning, especially in high-dimensional or collinear feature spaces, which are common in microbiome data. It also provides uncertainty estimates on predictions, which is valuable in biomedical applications where confidence intervals can aid decision-making.

**Support Vector Regression**, although powerful and non-linear, lacks interpretability and is computationally more expensive. Its reliance on kernel tricks makes it less transparent, especially when using RBF or polynomial kernels. Moreover, it showed greater sensitivity to the tuning process, which, while leading to performance gains, raises concerns about stability across datasets.

**Elastic Net** provides model simplicity and interpretability due to the sparsity induced by L1 regularization. It was the only model where feature selection and tuning had a modest but clear benefit. However, its predictive performance was consistently lower than that of Bayesian Ridge and SVR.

Taken together, Bayesian Ridge stood out not only in predictive accuracy but also in terms of consistency, ease of training, and interpretability, making it the most suitable model for this task.

## 4 Conclusion

In this study, we developed, evaluated, and compared three regression models—Elastic Net, Support Vector Regression (SVR), and Bayesian Ridge Regression—for predicting BMI based on gut microbiome composition. The modeling pipeline followed a structured approach involving baseline evaluation, feature selection, and hyperparameter tuning. Each model's performance was assessed using multiple evaluation metrics, including RMSE, MAE, and $R^2$, across 50 randomized splits to ensure robust and reproducible comparisons.

Our results showed that the baseline Bayesian Ridge model achieved the lowest RMSE and the highest $R^2$ among all configurations, indicating strong generalization ability without requiring additional complexity. Although SVR and Elastic Net showed improvements with hyperparameter tuning, they did not outperform Bayesian Ridge in predictive accuracy or robustness. Additionally, the interpretability and inherent regularization properties of Bayesian Ridge make it particularly well-suited for applications in biomedical data analysis, where stability and transparency are critical.

Feature selection offered moderate gains in model simplicity but did not significantly enhance performance. Hyperparameter tuning, while beneficial for certain models like SVR, introduced increased sensitivity and computational overhead.

Overall, this work demonstrates the value of combining rigorous evaluation strategies with principled model selection in microbiome-based prediction tasks.

## Appendix

- GitHub Repo: `https://github.com/maria-defteraiou/Assignment-1/tree/main`
- Scripts and notebooks: `functions.py`, `model_analysis.ipynb`, `data_exploration.ipynb`
- Final model path: `./models/winner.pkl`