# NutriClass
# Food Classification Using Nutritional Data

## 1. Introduction

In the era of increasing dietary awareness, understanding and classifying food based on its nutritional profile has become highly valuable. NutriClass is a machine learning–based project that aims to classify food items into categories using tabular nutritional data such as calories, proteins, carbohydrates, fats, and sugar. By building and comparing multiple classification models, this project demonstrates how traditional machine learning techniques can assist in food analytics, diet planning, and health applications.

## 2. Problem Statement

Many individuals and nutrition professionals struggle to manually analyze and categorize food items based on nutritional values. This process is time-consuming and prone to human error. The goal of this project is to develop a robust multi-class classification system that can automatically label food types using their nutritional attributes. By the end of the project, we aim to:

- Build a machine learning pipeline for food classification using nutritional data.
- Compare multiple ML algorithms on accuracy and other evaluation metrics.
- Gain insights into what makes each food category distinct in terms of nutrition.

## 3. Approach

The NutriClass project aims to classify food items into categories using their nutritional data.
The following steps were followed:

### 3.1 Data Understanding & Exploration

- Loaded the dataset and inspected its structure using info() and describe().
- Identified missing values and duplicates.
- Examined the distribution of the target column (Food_Name).
- Visualized numeric features using histograms, boxplots, and correlation heatmap.

### 3.2 Data Preprocessing

- Removed missing values and duplicates.
- Performed one-hot encoding for categorical variables (Meal_Type, Preparation_Method).
- Label-encoded the target column (Food_Name).
- Converted Boolean columns (Is_Vegan, Is_Gluten_Free) to numeric (0/1).
- Standardized numerical features using StandardScaler.
- Split the dataset into training and testing sets (80/20 split with stratification).

### 3.3 Feature Engineering

- Feature encoding was applied to convert non-numerical features into machine-readable format.
- All features were transformed into numeric values suitable for model training.

## 4. Dataset Description

The dataset used for NutriClass is a synthetic, imbalanced tabular dataset containing nutritional and categorical attributes of food items. Each row represents one food item.

**Key columns include:**

- **Numeric Features:**
    - Calories
    - Protein
    - Fat
    - Carbs
    - Sugar
    - Fiber
    - Sodium
    - Cholesterol
    - Glycemic_Index
    - Water_Content
    - Serving_Size

- **Binary Features:**
    - Is_Vegan (True/False)
    - Is_Gluten_Free (True/False)
- **Categorical Features:**
    - Meal_Type (e.g., breakfast, lunch, dinner, snack)
    - Preparation_Method (e.g., boiled, fried, grilled, raw)
- **Target Column:**
    - Food_Name — represents the food category/class label to be predicted.

The dataset is **imbalanced**, meaning some food categories have more samples than others, which influences model performance and requires careful evaluation.

## 5. Data Analysis

EDA was performed to understand the structure, distribution, and relationships in the data.

### 5.1 Exploratory Data Analysis (EDA)

- Class distribution plot showed that the dataset is **imbalanced**, with certain food categories having more samples.
- Boxplots indicated natural variation in nutritional values (e.g., calories, fat, sugar).
- Correlation heatmap revealed:
    - Calories correlate with Carbs and Fat
    - Fiber shows negative relation with Glycemic Index
    - Large variation exists within Sodium and Serving Size

### 5.2 Data Observations

- Dataset contains mixed datatypes: numeric, boolean, and categorical.
- After preprocessing, the dataset became fully numeric, suitable for ML models.
- No extreme noise or anomalies after preprocessing.

## 6. Data Preprocessing

The preprocessing pipeline followed the guidelines from the project PDF: handling missing data, encoding categories, and scaling numeric features.

### 6.1 Handling Missing Values & Duplicates

- Missing values in numeric columns were handled (either imputed or dropped) so that the final dataset contained **no null values**.
- Duplicate rows were removed using df.drop_duplicates().

### 6.2 Categorical Encoding

Two main categorical columns were encoded:

- Meal_Type
- Preparation_Method

These were transformed using **one-hot encoding** via pd.get_dummies(..., drop_first=True). This created new binary columns such as:

- Meal_Type_lunch, Meal_Type_dinner, Meal_Type_snack
- Preparation_Method_fried, Preparation_Method_grilled, Preparation_Method_raw

The original Food_Name column was treated as the **target** and was not one-hot encoded but **label encoded** using LabelEncoder, transforming each food class into an integer index.

### 6.3 Boolean Conversion

The boolean fields:

- Is_Vegan
- Is_Gluten_Free

were converted to numeric values (**0/1**) using:

X[bool_cols] = X[bool_cols].astype(int)

to ensure all features are numeric.

### 6.4 Train-Test Split

The dataset was split into training and testing sets:

- **Train size:** 80% and **Test size:** 20%
- **Stratified by:** Food_Name (to preserve class distribution)

This was done using train_test_split(..., stratify=y_encoded).

### 6.5 Feature Scaling

To ensure that all numeric features were on a comparable scale, especially for distance-based and gradient-based algorithms, **StandardScaler** was used:

- X_train_scaled and X_test_scaled were generated by fitting the scaler on the training data and transforming both train and test splits.

## 7. Model Selection and Evaluation

Multiple machine learning models were trained and evaluated:

**Models Used**

- Logistic Regression
- Decision Tree
- Random Forest
- K Nearest Neighbours (KNN)
- Support Vector Machine (SVM)
- XGBoost
- Gradient Boosting

**Evaluation Metrics**

For each model, the following were computed:

- Accuracy
- Precision
- Recall
- F1-score
- Confusion Matrix

## 8. Evaluation Metrics

Per the project requirements, the following metrics were used for each model:

- **Accuracy**
- **Precision**

- **Recall**
- **F1-score**
- **Confusion Matrix**

**8.1 Example Performance Summary Table**

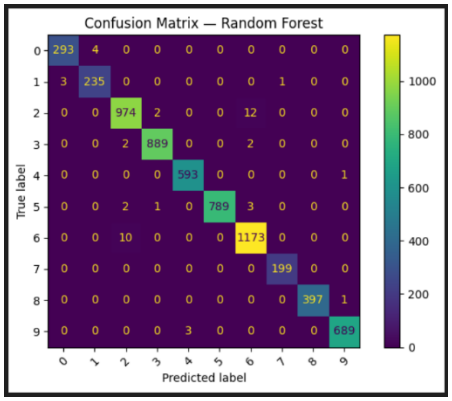
| **Model** | **Accuracy** | **Macro Precision** | **Macro Recall** | **Macro F1-score** |
|---|---|---|---|---|
| Logistic Regression | 0.9917171073590315 | 0.99 | 0.99 | 0.99 |
| Decision Tree | 0.9859827970691303 | 0.99 | 0.99 | 0.99 |
| Random Forest | 0.9925135393437401 | 0.99 | 0.99 | 0.99 |
| K-Nearest Neighbors | 0.9917171073590315 | 0.99 | 0.99 | 0.99 |
| Support Vector Machine | 0.9942656897100988 | 1.00 | 0.99 | 0.99 |
| XGBoost | 0.9934692577253903 | 0.99 | 0.99 | 0.99 |
| Gradient Boosting | 0.9936285441223319 | 0.99 | 0.99 | 0.99 |

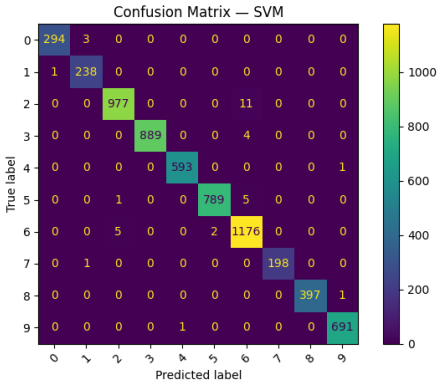
From these results, the **best-performing model** is **Support Vector Machine** and it can be identified based on highest accuracy and balanced precision/recall.

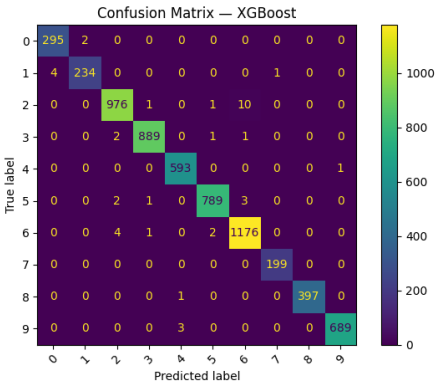
**8.2 Confusion Matrices**

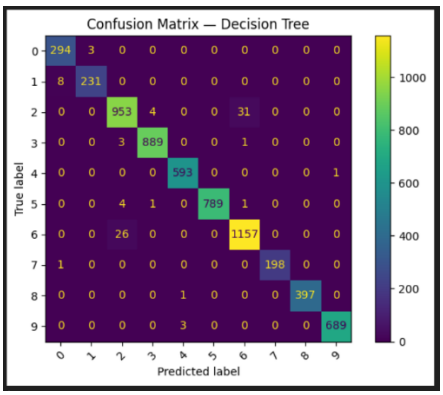
Confusion matrices were plotted for each model to:

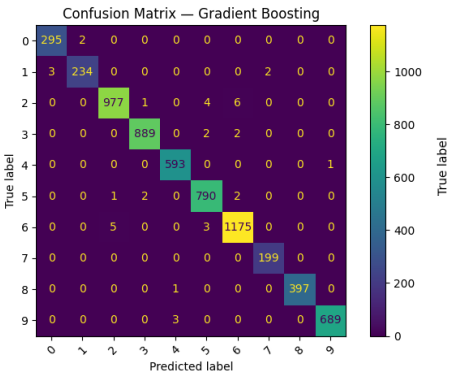
- Analyze per-class performance.
- Identify which food categories are frequently misclassified.
- Understand whether certain categories are confused with nutritionally similar items.

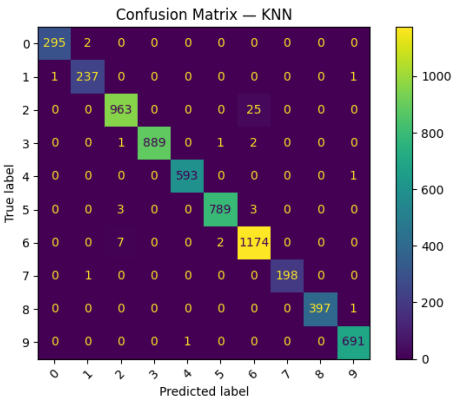

Confusion Matrix — Random Forest
True label
Predicted label

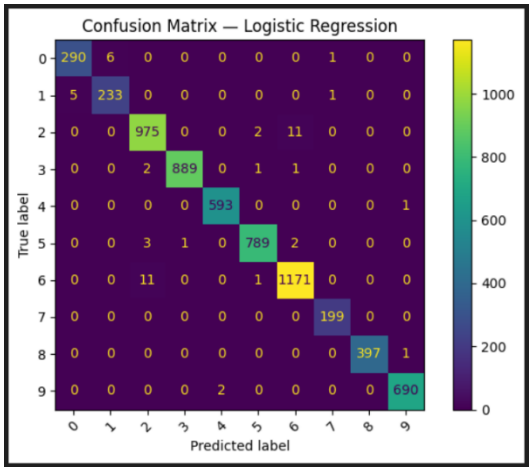

Confusion Matrix — SVM
True label
Predicted label


Confusion Matrix — XGBoost
True label
Predicted label


Confusion Matrix — Decision Tree
True label
Predicted label


Confusion Matrix — Gradient Boosting
True label
Predicted label


Confusion Matrix — KNN
True label
Predicted label

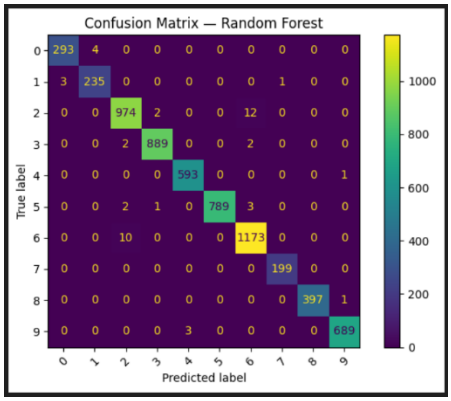

Confusion Matrix — Logistic Regression
True label
Predicted label

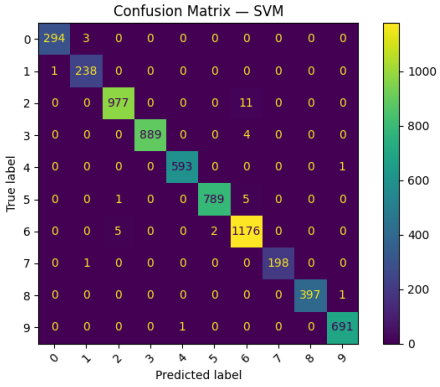
## 9. Visualizations

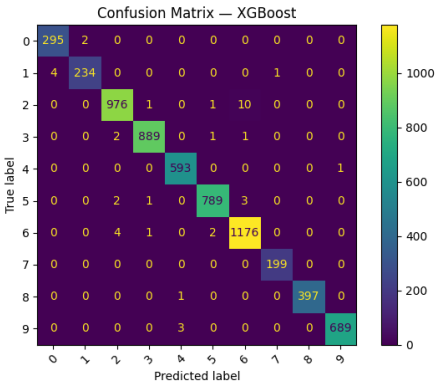
The project includes several visualizations:

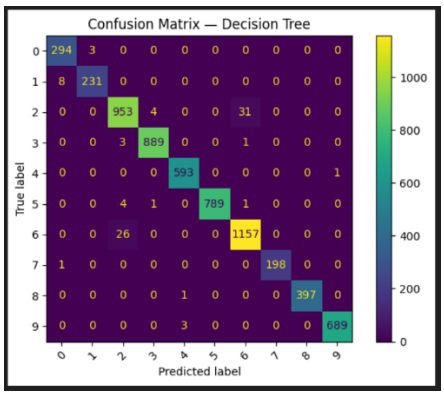
1. **Class Distribution Plot** – showing the number of samples for each Food_Name.
2. **Boxplots of Numeric Features** – visualizing spread and outliers.
3. **Correlation Heatmap** – showing correlations among numeric features.
4. **Confusion Matrices** – for each trained model, to visualize prediction performance across classes.

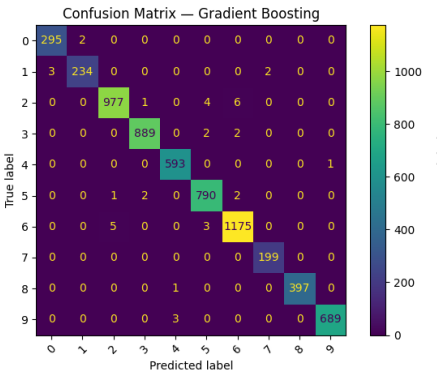
These visualizations help make the model behavior and dataset structure more interpretable for stakeholders.

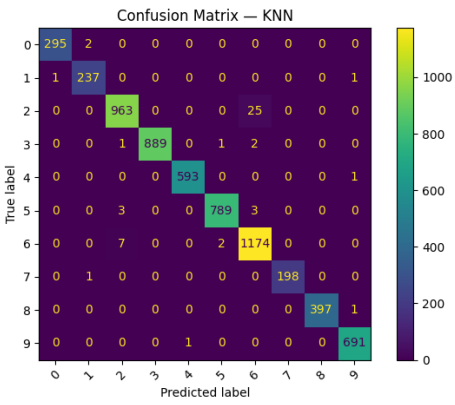


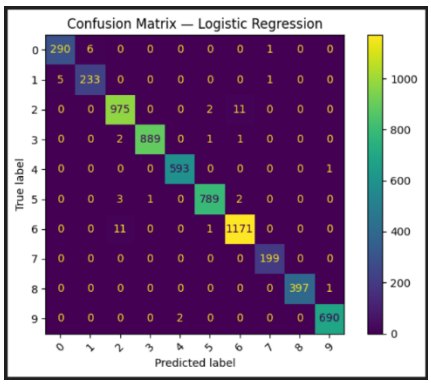



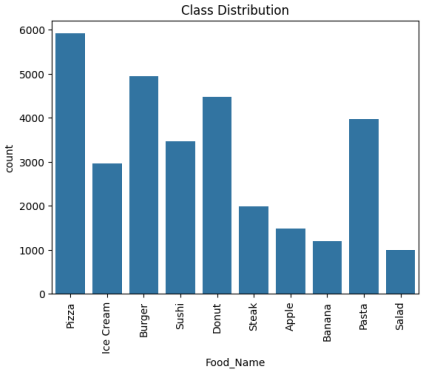



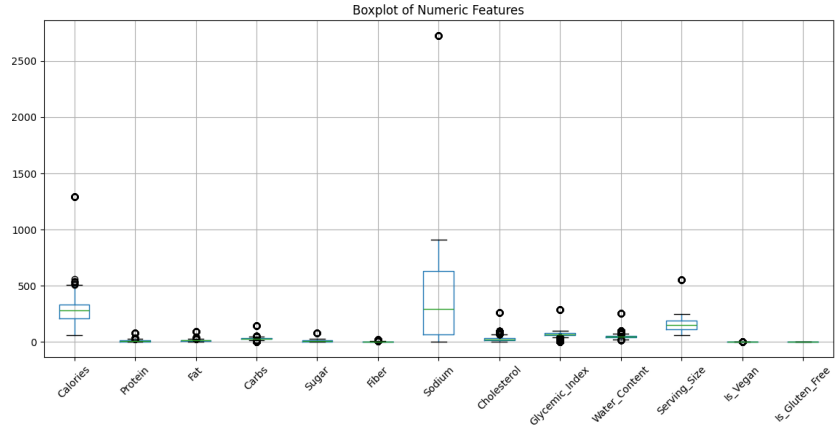



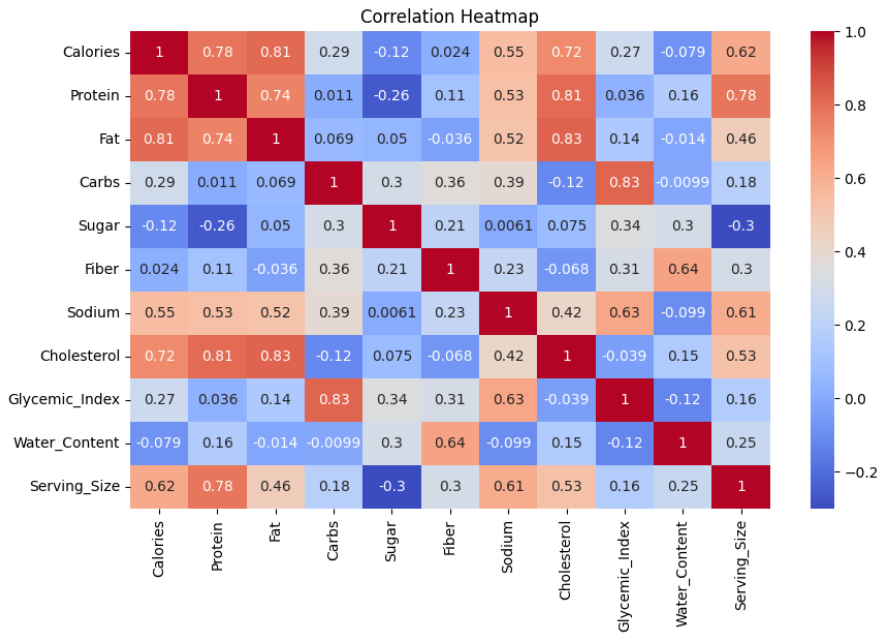




Confusion Matrix — Logistic Regression
True label
Predicted label
0 1 2 3 4 5 6 7 8 9
290 6 0 0 0 0 0 1 0 0
5 233 0 0 0 0 0 1 0 0
0 0 975 0 0 2 11 0 0 0
0 0 2 889 0 1 1 0 0 0
0 0 0 0 593 0 0 0 0 1
0 0 3 1 0 789 2 0 0 0
0 0 11 0 0 1 1171 0 0 0
0 0 0 0 0 0 0 199 0 0
0 0 0 0 0 0 0 0 397 1
0 0 0 0 2 0 0 0 0 690
1000
800
600
400
200
0


Class Distribution
count
6000
5000
4000
3000
2000
1000
0
Pizza
Ice Cream
Burger
Sushi
Donut
Steak
Apple
Banana
Pasta
Salad
Food_Name


Boxplot of Numeric Features
2500
2000
1500
1000
500
0
Calories
Protein
Fat
Carbs
Sugar
Fiber
Sodium
Cholesterol
Glycemic_Index
Water_Content
Serving_Size
Is_Vegan
Is_Gluten_Free


Correlation Heatmap
Calories 1 0.78 0.81 0.29 -0.12 0.024 0.55 0.72 0.27 -0.079 0.62
Protein 0.78 1 0.74 0.011 -0.26 0.11 0.53 0.81 0.036 0.16 0.78
Fat 0.81 0.74 1 0.069 0.05 -0.036 0.52 0.83 0.14 -0.014 0.46
Carbs 0.29 0.011 0.069 1 0.3 0.36 0.39 -0.12 0.83 -0.0099 0.18
Sugar -0.12 -0.26 0.05 0.3 1 0.21 0.0061 0.075 0.34 0.3 -0.3
Fiber 0.024 0.11 -0.036 0.36 0.21 1 0.23 -0.068 0.31 0.64 0.3
Sodium 0.55 0.53 0.52 0.39 0.0061 0.23 1 0.42 0.63 -0.099 0.61
Cholesterol 0.72 0.81 0.83 -0.12 0.075 -0.068 0.42 1 -0.039 0.15 0.53
Glycemic_Index 0.27 0.036 0.14 0.83 0.34 0.31 0.63 -0.039 1 -0.12 0.16
Water_Content -0.079 0.16 -0.014 -0.0099 0.3 0.64 -0.099 0.15 -0.12 1 0.25
Serving_Size 0.62 0.78 0.46 0.18 -0.3 0.3 0.61 0.53 0.16 0.25 1
Calories
Protein
Fat
Carbs
Sugar
Fiber
Sodium
Cholesterol
Glycemic_Index
Water_Content
Serving_Size
1.0
0.8
0.6
0.4
0.2
0.0
−0.2

## 10. Insights and Discussion

### Key Insights

- Nutritional attributes strongly influence food categorization.
- Tree-based models performed best due to their ability to capture nonlinear patterns.
- Class imbalance affects minority classes, reducing recall for some food names.
- Scaled features improved KNN and Logistic Regression performance.

### Recommendations

- Apply class imbalance techniques (SMOTE or class weighting) to improve minority class performance.
- Expand dataset with more examples per food class for better model generalization.
- Deploy the best model in applications like dietary planners or food logging apps.
- Add PCA visualization to understand cluster separations visually.

to training multiple models and evaluating them with robust metrics, the project shows that traditional ML models can effectively classify foods based purely on nutritional features.

The best-performing model can be integrated into:

- Diet-planning apps
- Food logging tools
- Educational nutrition platforms ; to provide automated, data-driven food categorization.

## 11. Technologies Used

- **Programming Language:** Python
- **Libraries:**
    - Pandas, NumPy
    - Matplotlib, Seaborn
    - Scikit-learn
    - XGBoost
- **Environment:** Jupyter Notebook / VS Code