MACHINE LEARNING REPORT 2025

Made By

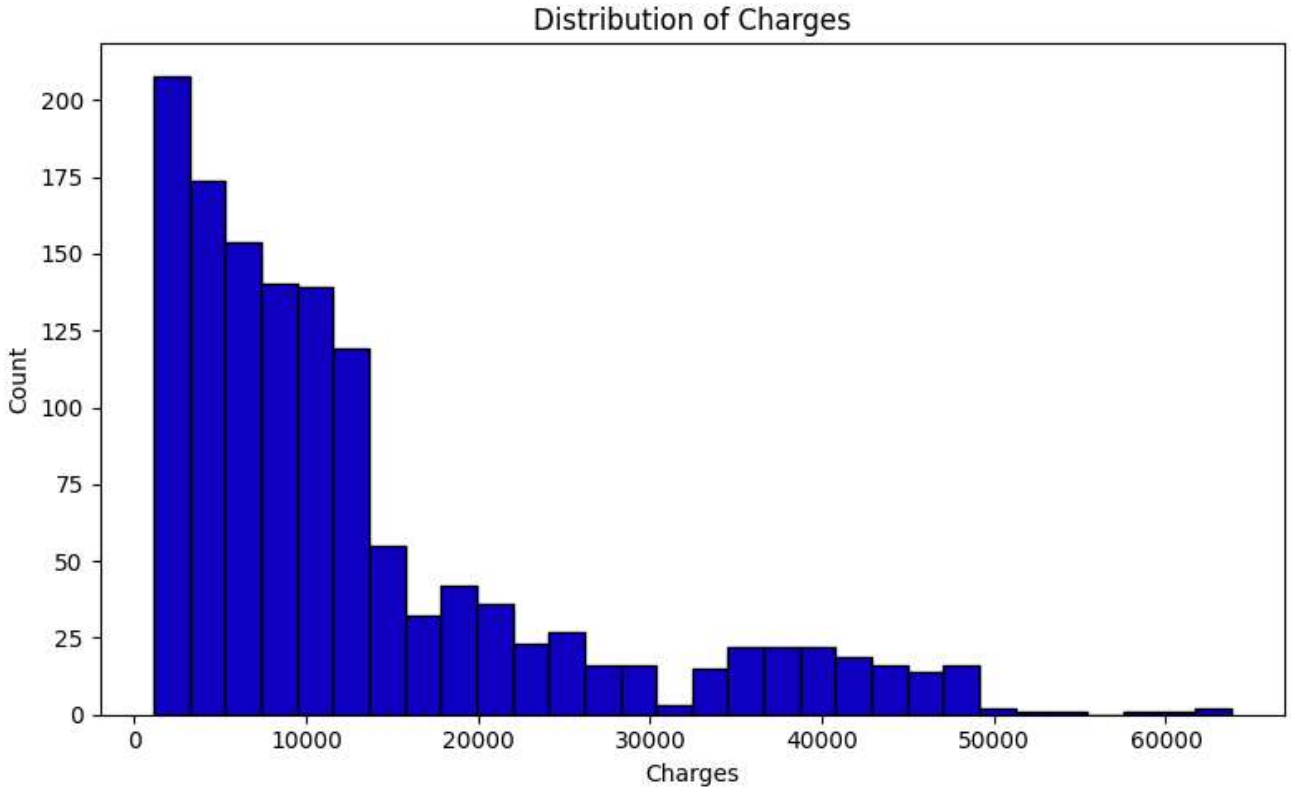
Hager Ahmed	20230634
Rana Alaa	20230207
Mohamed Ehab	20230456
Mazen Marie	20230440
Omar Ayman	20230367
Youssef Hatem	20230675

Table of Contents

1. Health Care Insurance Regression Project

1.1 Dataset Analysis

The dataset used in this project is the **Healthcare Insurance Expenses** Dataset, which contains demographic and health-related information used to predict individual healthcare insurance expenses. The dataset reflects real-world insurance data and is commonly used for regression and healthcare cost analysis tasks. Each record represents a single individual and includes attributes such as age, sex, body mass index (BMI), number of children, smoking status, residential region, and the corresponding medical insurance charges. These features capture both lifestyle-related and demographic factors that strongly influence healthcare costs. The dataset contains **1,338 records and 7 features**, with the target variable being **charges**, which represents the total medical insurance cost billed to the individual. Initial inspection showed that the dataset is complete, with **no missing values** across all columns. This allowed the analysis to proceed without the need for imputation or missing-value handling. The distribution of the target variable revealed a right-skewed pattern, indicating that while most individuals incur moderate medical expenses, a smaller portion of the population experiences very high healthcare costs. This skewness is typical in medical cost data and highlights the presence of outliers caused by severe illnesses or high-risk individuals.



1.2 Feature Selection and Preprocessing

To simplify the model and focus on the most impactful variables, the **region feature was removed** during preprocessing. This decision was made to reduce dimensionality and minimize noise introduced by geographical categorization, which showed limited contribution to predicting charges in this setup.

The remaining features were categorized into numerical and categorical variables:

- Numerical: age, BMI, children
- Categorical: sex, smoker

Categorical features were encoded using One-Hot Encoding, while **numerical features were standardized using StandardScaler**. Standardization ensures that all numerical features have zero mean and unit variance, which is especially important for distance-based models and linear regression optimization. A **ColumnTransformer** was used to apply the appropriate transformations to each feature group in a clean and reproducible manner. The dataset was then **split into training (80%) and testing (20%)** sets using a fixed random seed to ensure reproducibility.

1.3 Target

The target **Charges**, represent continuous numerical values and therefore doesn't require encoding. Due to the wide range and skewed distribution, evaluating model performance using error-based metrics like **RMSE** (Root Mean Squared Error) and **R2 score**

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(Predicted_i - Actual_i)^2}{n}}$$

$$R2 = 1 - \frac{Residual\ Sum\ of\ Squares}{Total\ Sum\ of\ Squares}$$

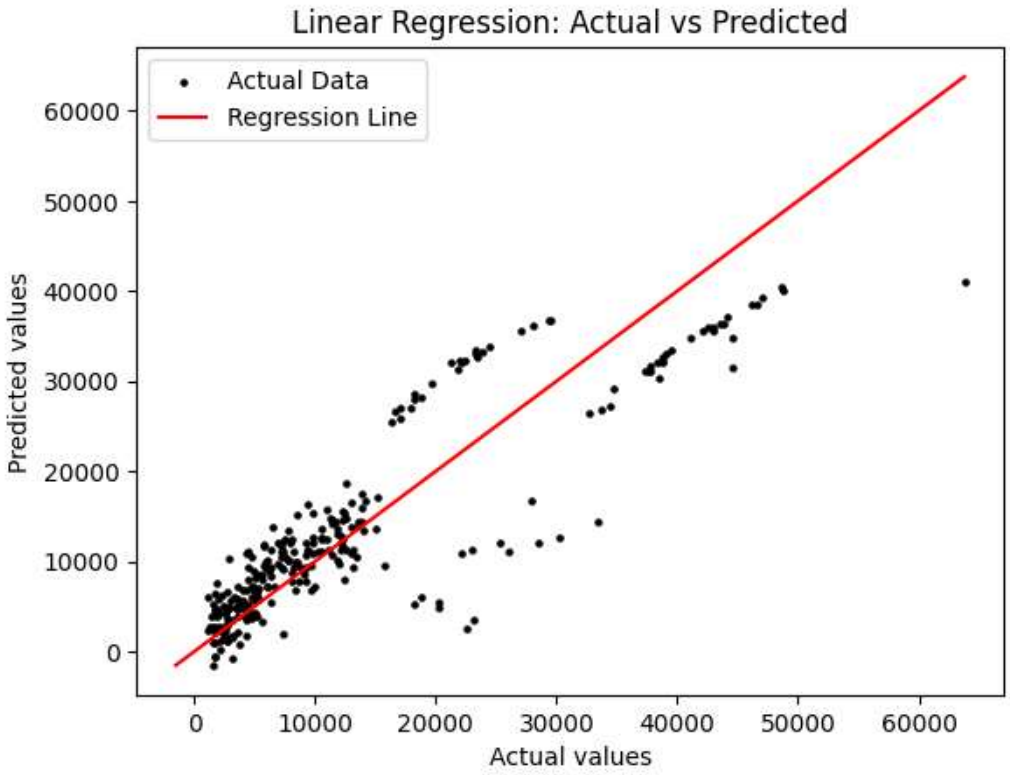
1.4 Linear Regression Model

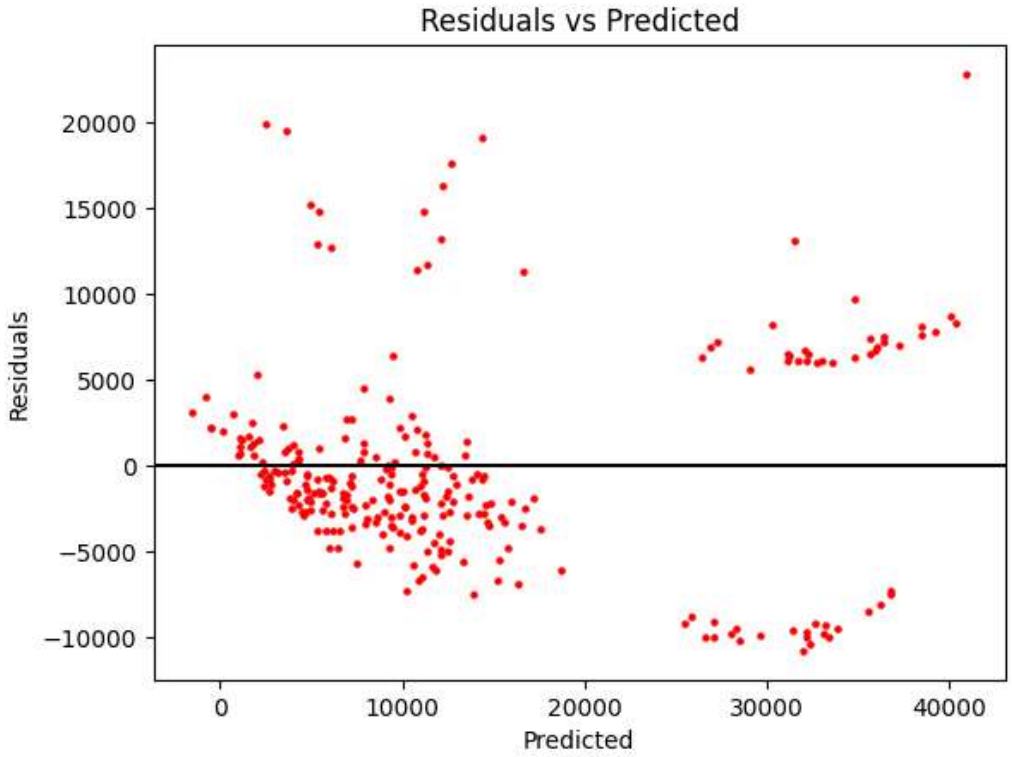
Linear Regression was used as a baseline supervised learning model to predict healthcare insurance expenses. This model assumes a linear relationship between the input features and the target variable. After training on the preprocessed feature set, the Linear Regression model achieved:

R2 Score: 0.7811302113434097

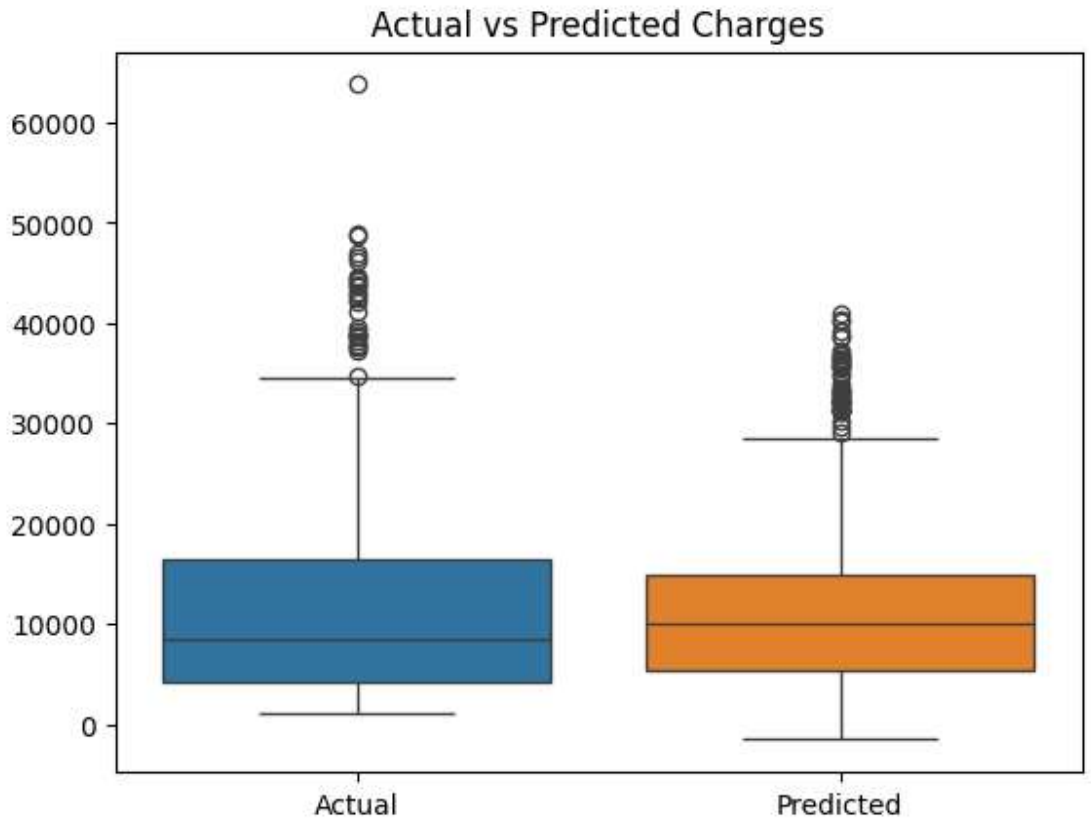
RMSE: 5829.172930254189

Scatter plot:





This indicate that the model explains around 78% of the variance in insurance charges. Scatter plots of actual versus predicted values showed a reasonable alignment along the diagonal reference line, suggesting acceptable predictive performance. It also revealed that while errors are generally centered around zero, larger residuals appear for higher charge values. This suggests that linear regression struggles to fully capture the non-linear behavior present in extreme medical cost cases, such as those involving smokers or individuals with very high BMI.



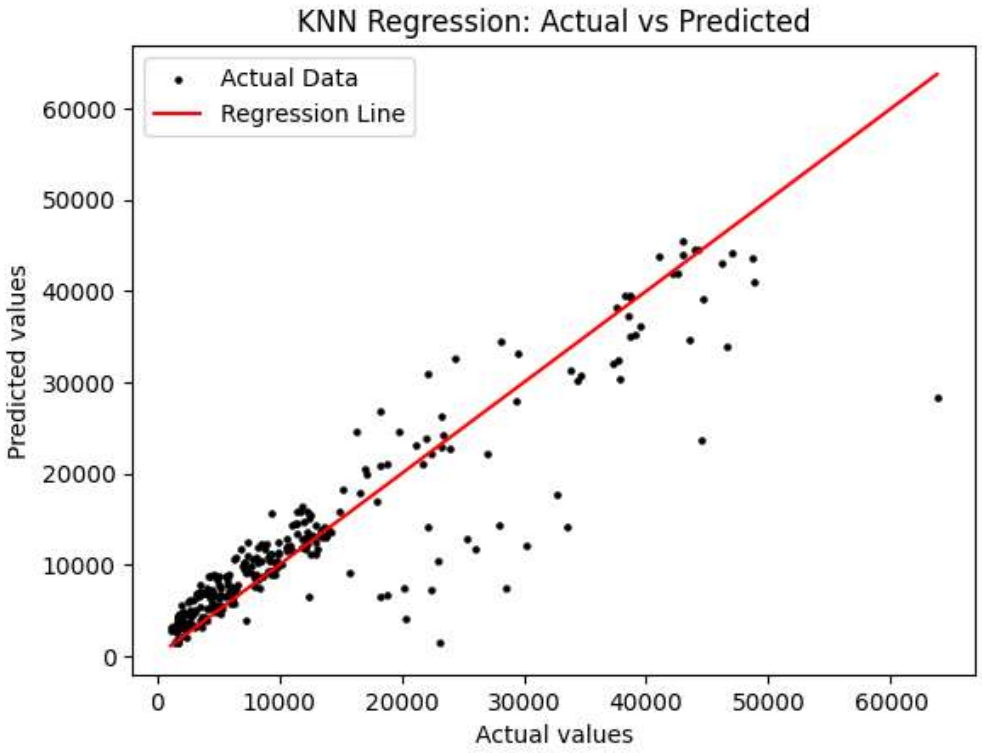
1.5 KNN Regression

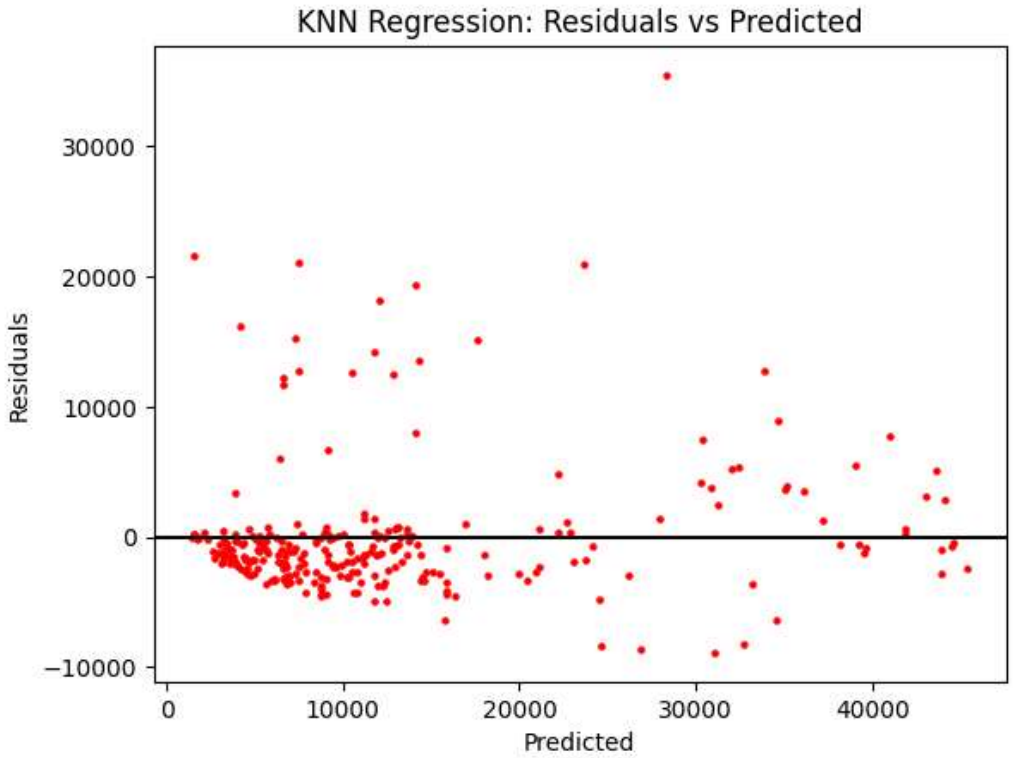
To better capture non-linear relationships in the data, a K-Nearest Neighbors Regressor was trained using the same preprocessed feature set. KNN predicts insurance charges based on the average values of the nearest neighbors in feature space. The model was trained **using k = 11 neighbors**, selected to balance bias and variance. Performance results for KNN regression were:

R2 Score: 0.8249128499031824

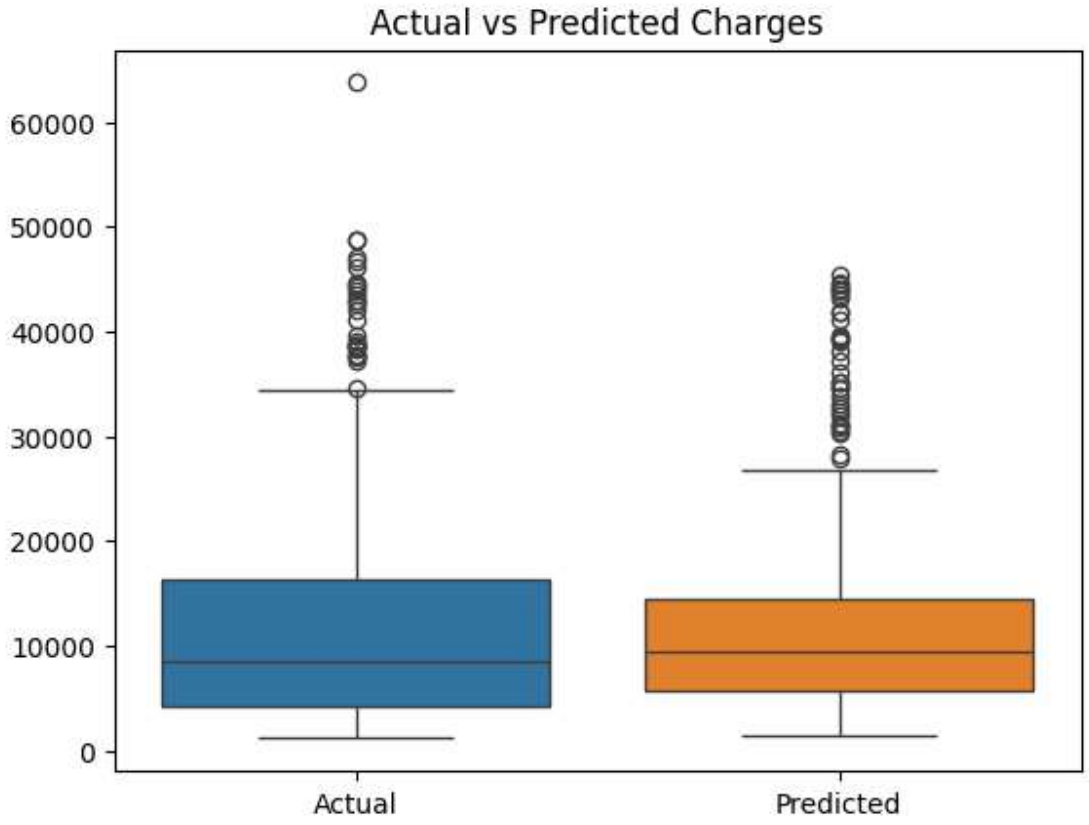
RMSE: 5213.64152517611

Scatter plots:





Compared to Linear Regression, KNN demonstrated **improved predictive performance**, explaining a larger portion of the variance and producing lower prediction errors. This improvement highlights the presence of non-linear relationships between individual attributes and insurance costs. Residual plots showed a more uniform error distribution, and boxplots comparing actual and predicted values indicated that KNN better approximates the overall charge distribution.



1.6 Model Comparison and Error Analysis

Both models were evaluated using scatter plots, residual plots, and boxplots to visually assess prediction accuracy and error behavior.

- **Linear Regression** provides **interpretability and strong baseline** performance but is **limited in modeling complex interactions**.
- **KNN Regression captures local patterns more effectively** and **performs better on higher-cost individuals, at the expense of increased computational complexity**.

This clearly demonstrates **that non-linear models are better suited for healthcare cost prediction** tasks where interactions between features such as age, smoking status, and BMI are significant.

1.7 Conclusion

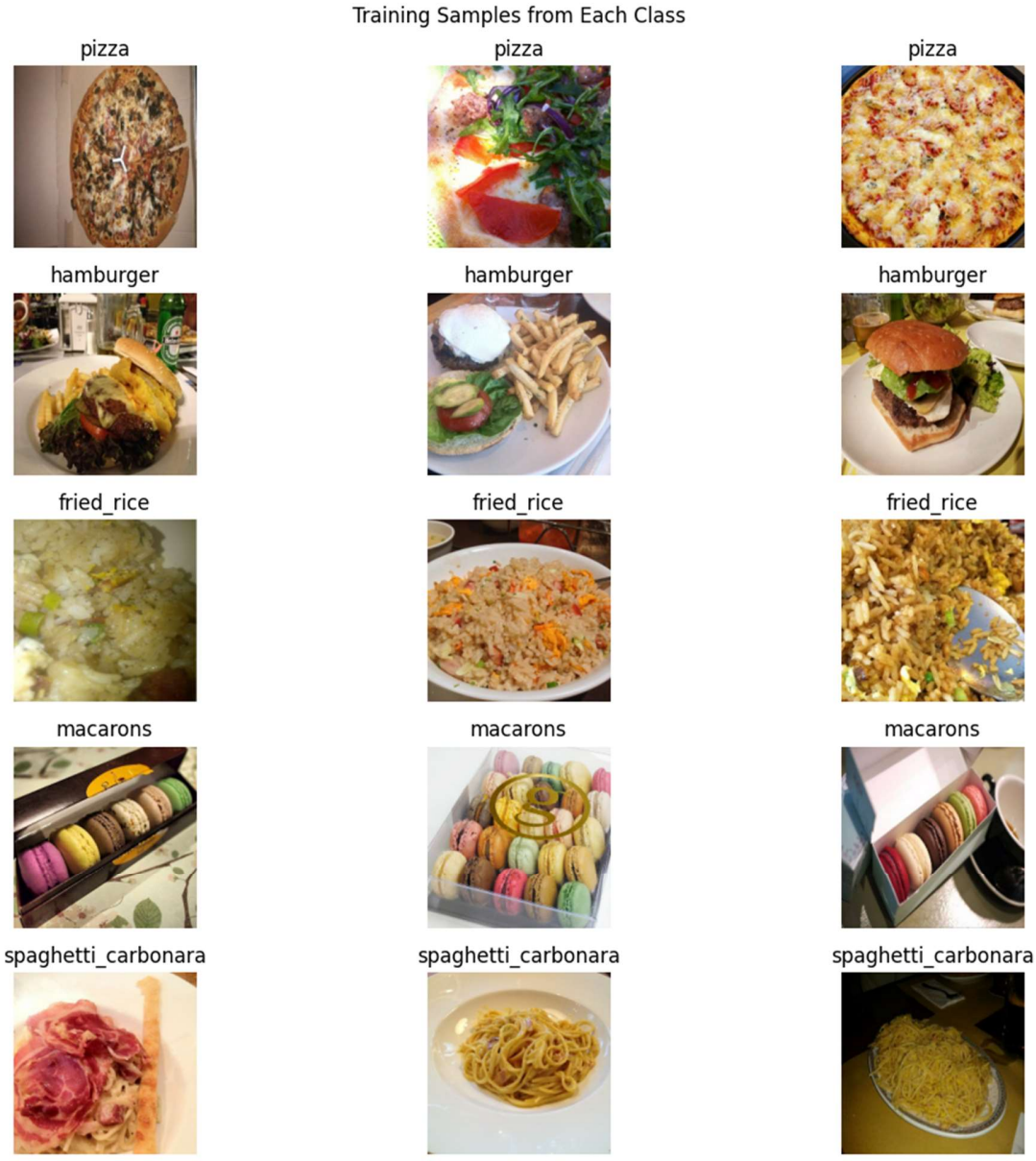
This project demonstrates the application of regression techniques to predict healthcare insurance expenses using demographic and lifestyle features. Exploratory data analysis revealed strong skewness in medical costs, emphasizing the importance of robust evaluation metrics. While Linear Regression provided a solid baseline with interpretable results, KNN Regression achieved superior performance by modeling non-linear relationships more effectively. The results highlight that healthcare cost prediction benefits from models capable of capturing complex patterns rather than relying solely on linear assumptions.

2. Food-101 Image Classification Project

2.1 Dataset Analysis

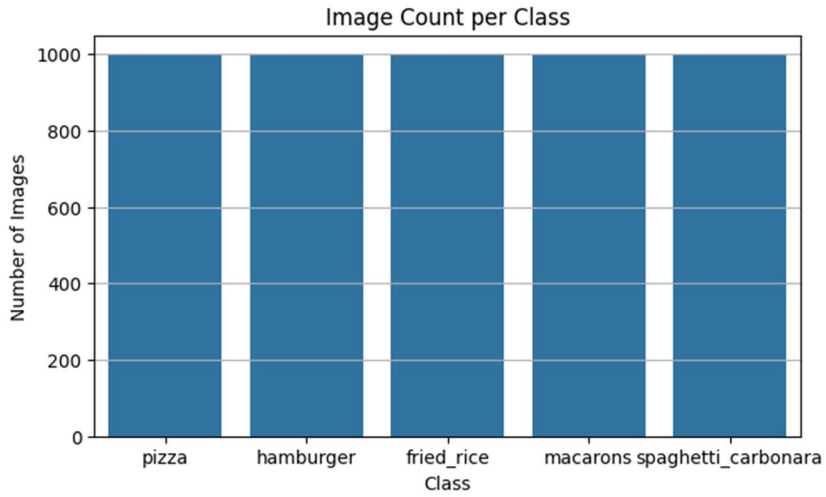
The dataset used in this project is the **Food-101 dataset**, which is a well-known benchmark dataset for food image classification and visual recognition tasks. The dataset was collected from real-world web sources, making it highly diverse and challenging. Images vary significantly in terms of lighting conditions, backgrounds, camera quality, viewing angles, and food presentation styles. This diversity closely reflects real-world scenarios and increases the complexity of the classification problem.

Although the original Food-101 dataset contains 101 food categories, this project focuses on a **subset of five representative classes** to reduce computational cost while maintaining meaningful classification difficulty. The selected classes are: *pizza, hamburger, fried rice, macarons,* and *spaghetti carbonara*. These classes were chosen because they exhibit both inter-class similarity and intra-class variation, which makes them suitable for evaluating feature extraction and classification performance.



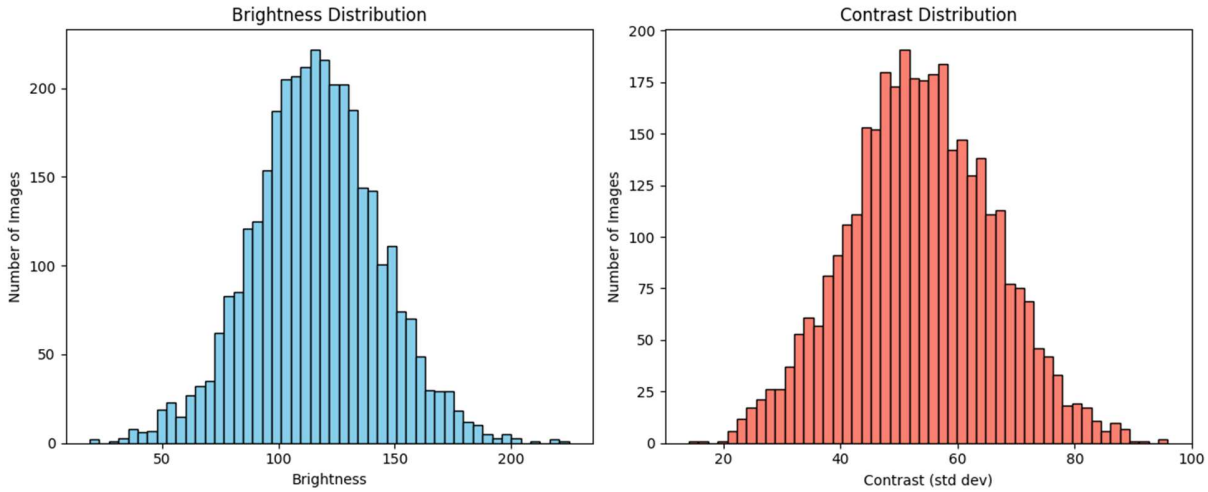
Each class contains a large and approximately equal number of images, ensuring that the dataset is balanced. This balance is important because it allows accuracy to be used as a reliable evaluation metric without being biased toward any particular class.

Number of images per class is: 1000 image



All images are RGB images with three color channels, and they come in varying resolutions, which necessitates resizing during preprocessing.

Before training any models, an exploratory data analysis (EDA) phase was conducted. During this phase, the number of images per class was computed and visualized to confirm that the dataset is balanced. Random samples from each class were displayed to visually inspect the data and understand similarities and differences between food categories. In addition, brightness and contrast distributions were analyzed by converting images to grayscale and computing the mean intensity and standard deviation for each image. The results showed noticeable variability in illumination and contrast across images, which justifies the need for normalization and robust feature extraction techniques.



Plots included in this section illustrate the class distribution, example images from each class, and the brightness and contrast histograms.

2.2 Data Preprocessing

All images were resized to a fixed resolution of **224 × 224 pixels**. This resolution was selected because it matches the input size required by MobileNetV2, which is used as the feature extractor in this project. Resizing ensures consistency across all inputs and enables efficient batch processing.

After resizing, images were preprocessed using the *mobilenet_v2.preprocess_input* function. This preprocessing step scales pixel values and applies normalization that aligns the input distribution with the one used during MobileNetV2 training on the ImageNet dataset. Applying this preprocessing is essential because pre-trained convolutional neural networks are highly sensitive to input distributions. Using the same preprocessing ensures that the extracted features are meaningful and compatible with the learned weights.

2.3 Dataset Splitting Strategy

The dataset was split into three disjoint subsets: training, validation, and test sets. The splitting process was performed after loading the images using TensorFlow's dataset utilities, with shuffling enabled and a fixed random seed to ensure reproducibility.

Approximately **70% of the data** was allocated to the training set. This subset is used to train the models and learn the underlying patterns in the data. Around **15% of the data** was assigned to the validation set, which is used during development to monitor performance, tune hyperparameters, and detect overfitting. The remaining **15% of the data** was reserved as the test set and kept completely unseen during training. This test set provides an unbiased evaluation of the final model performance.

This splitting strategy ensures a fair evaluation pipeline and prevents information leakage between training and evaluation stages.

2.4 Feature Extraction Using MobileNetV2

Instead of training a deep convolutional neural network from scratch, this project adopts a **transfer learning** approach. A pre-trained MobileNetV2 model is used as a fixed feature extractor. This approach significantly reduces training time and computational cost while leveraging powerful visual representations learned from large-scale datasets.

MobileNetV2 is a lightweight convolutional neural network designed for efficiency. Its architecture is based on depthwise separable convolutions and inverted residual blocks with linear bottlenecks. These architectural choices drastically reduce the number of parameters and computations while maintaining strong representational capacity.

In this project, the classification head of MobileNetV2 was removed by setting *include_top=False*. Global average pooling was applied to the output of the final convolutional block. As a result, each input image is transformed into a **1280-dimensional feature vector**. These features capture high-level semantic information such as shapes, textures, edges, and object-level patterns present in food images.

The MobileNetV2 weights were frozen during feature extraction, meaning that no fine-tuning was performed. This prevents overfitting and ensures that the extracted features remain stable across all datasets.

2.5 Feature Standardization

After feature extraction, the resulting 1280-dimensional feature vectors were standardized using **StandardScaler**. Standardization transforms each feature to have zero mean and unit variance.

This step is particularly important for the models used in this project. Logistic Regression relies on gradient-based optimization, which converges faster and more reliably when features are on similar scales. K-Means clustering uses Euclidean distance to assign points to clusters, and without standardization, features with larger numeric ranges would dominate the distance computation. Therefore, standardization is a crucial step to ensure fair feature contribution and stable model behavior.

Euclidean Distance:

$$d(x, y) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

2.6 Logistic Regression Model

Logistic Regression is a supervised linear classification algorithm. In this project, **multinomial Logistic Regression** was used to handle the multi-class nature of the problem. The model applies the softmax function to produce a probability distribution over the five food classes.

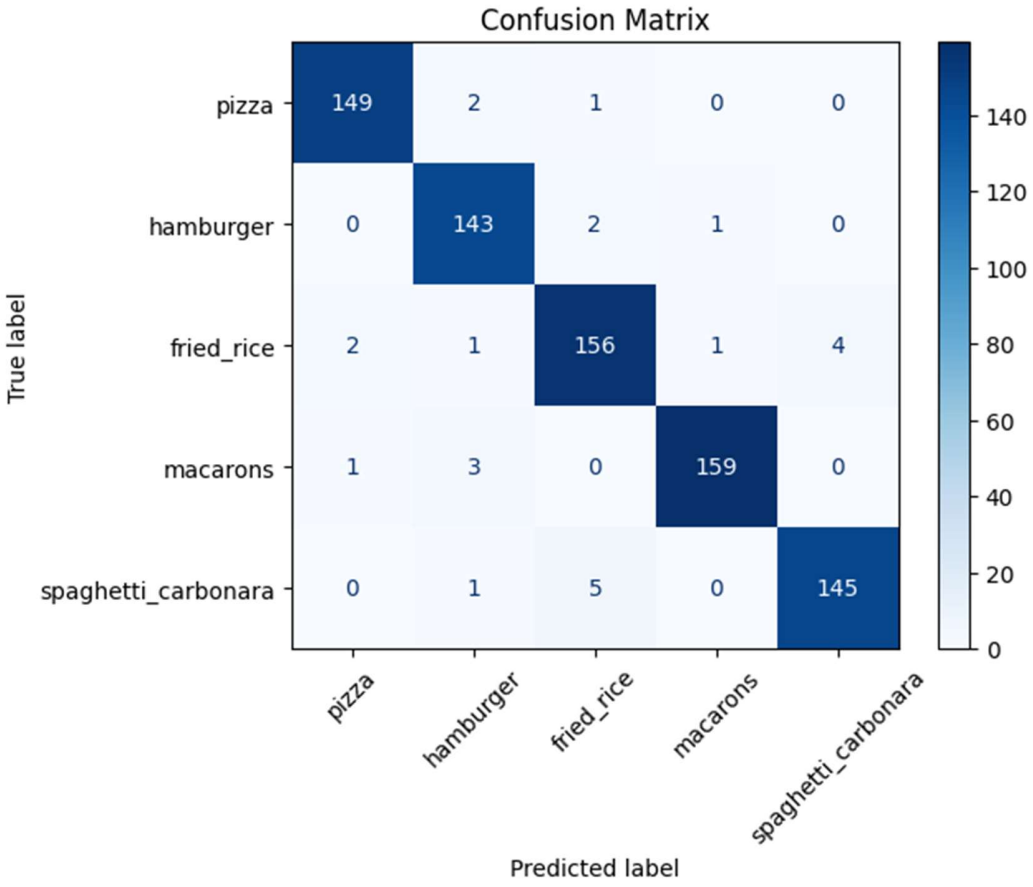
Softmax function:

$$z_i = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}}$$

The model was trained on the standardized MobileNetV2 feature vectors using the SAGA solver, which is well-suited for large-scale and multinomial optimization problems. A high maximum number of iterations was specified to ensure proper convergence.

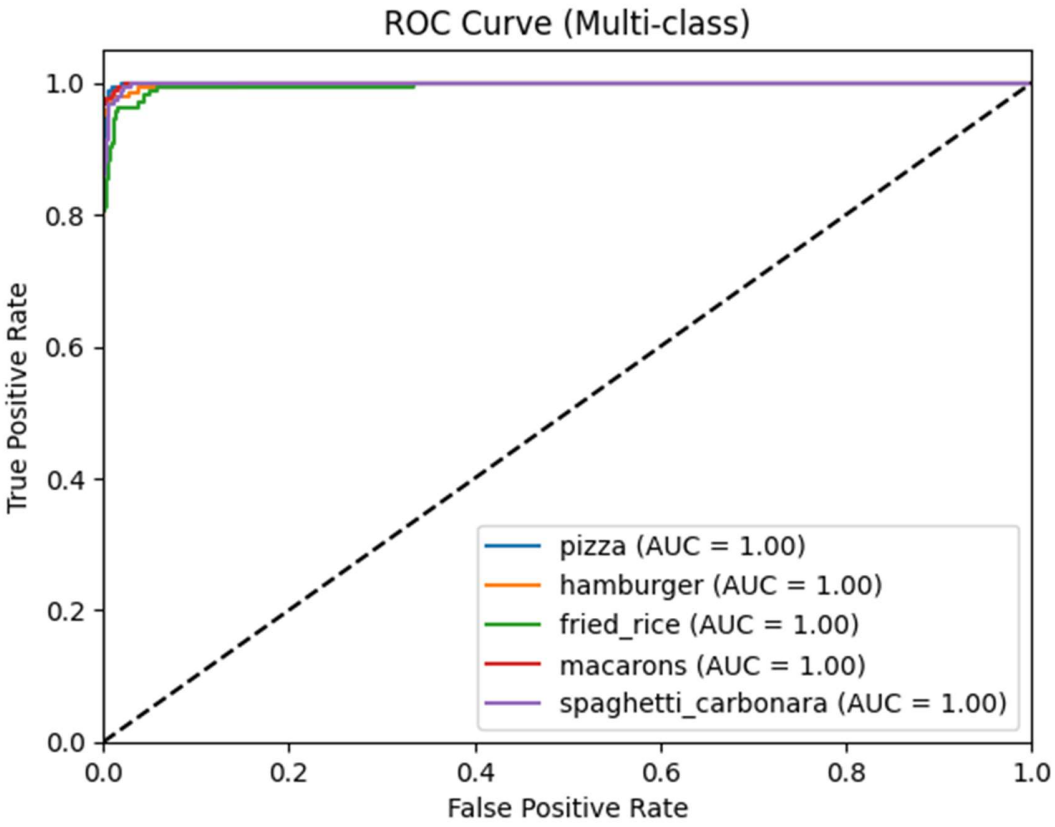
Based on the notebook results, Logistic Regression achieved a **training accuracy of 100.00%**, indicating that the deep features extracted by MobileNetV2 are highly linearly separable. More importantly, the model demonstrated excellent generalization performance with a **test accuracy of 96.91%**.

Additional evaluation metrics were used to further analyze performance. The confusion matrix showed strong diagonal dominance, indicating correct classification across most samples.



The log loss values for training and test sets were close, suggesting minimal overfitting.

Multi-class ROC curves showed high AUC values across all classes, confirming the robustness of the classifier.



Classification report:

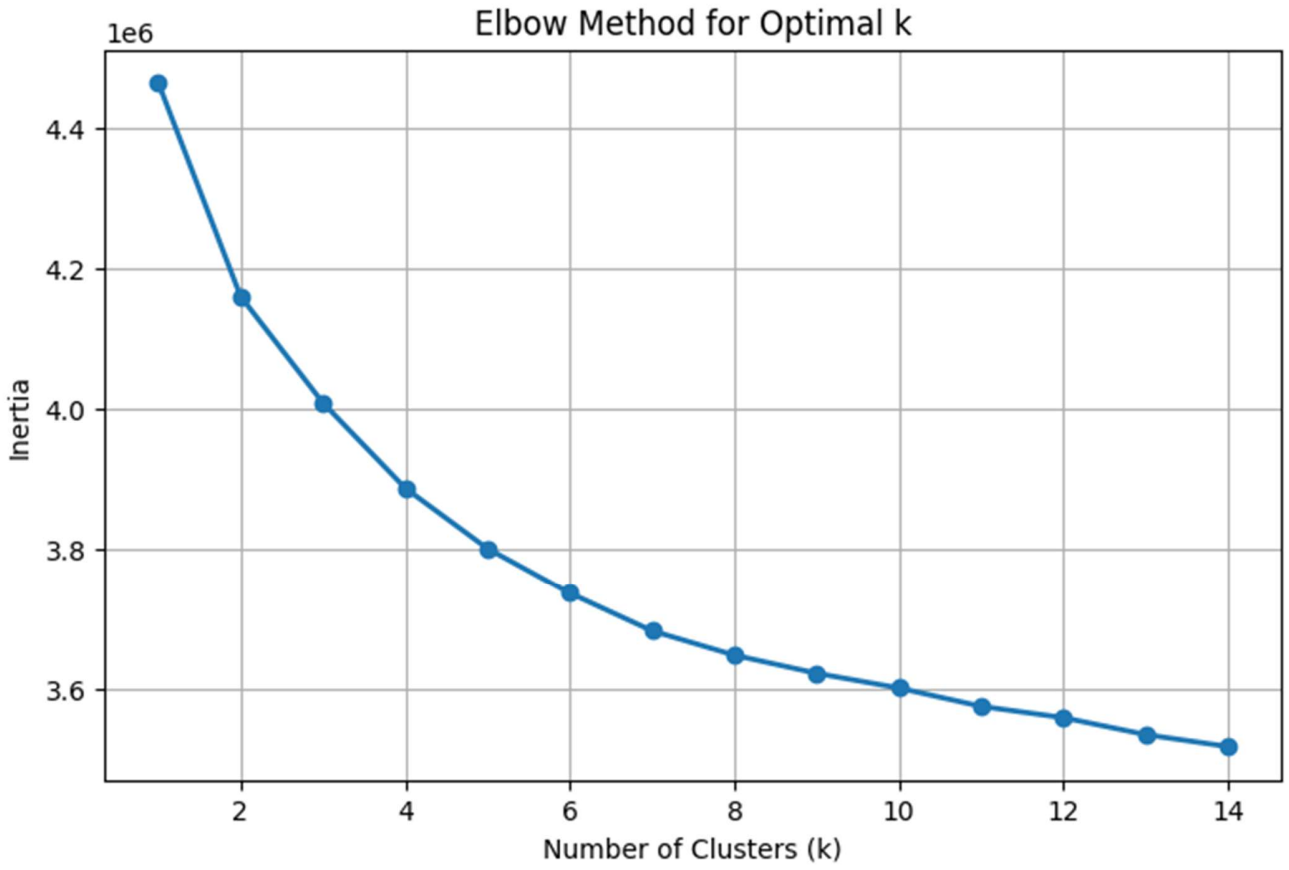
Classification Report (TEST):

	precision	recall	f1-score	support
pizza	0.98	0.98	0.98	152
hamburger	0.95	0.98	0.97	146
fried_rice	0.95	0.95	0.95	164
macarons	0.99	0.98	0.98	163
spaghetti_carbonara	0.97	0.96	0.97	151
accuracy			0.97	776
macro avg	0.97	0.97	0.97	776
weighted avg	0.97	0.97	0.97	776

2.7 K-Means Clustering

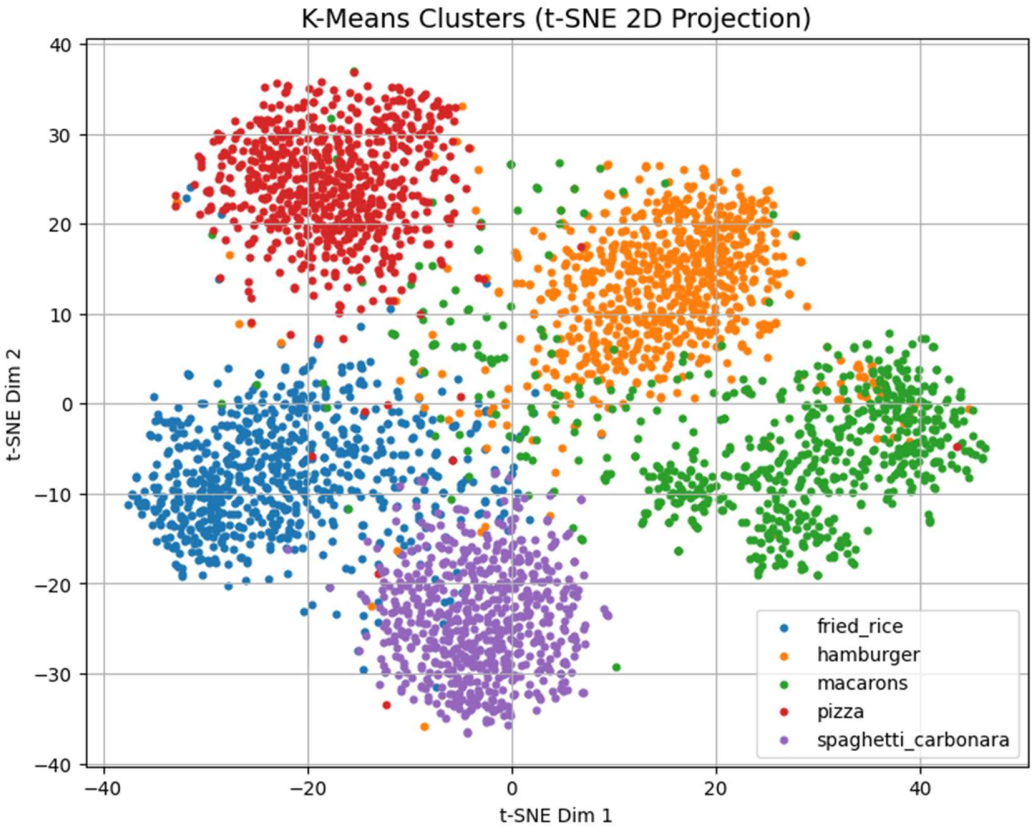
K-Means is an unsupervised clustering algorithm that groups data points based on feature similarity. Unlike Logistic Regression, K-Means does not use class labels during training and does not explicitly optimize class separation.

An elbow method was applied to determine the optimal number of clusters. The inertia curve was analyzed, and the KneeLocator algorithm was used to automatically select the best value of *k*. After clustering, cluster indices were mapped to true class labels using majority voting in order to compute classification accuracy.



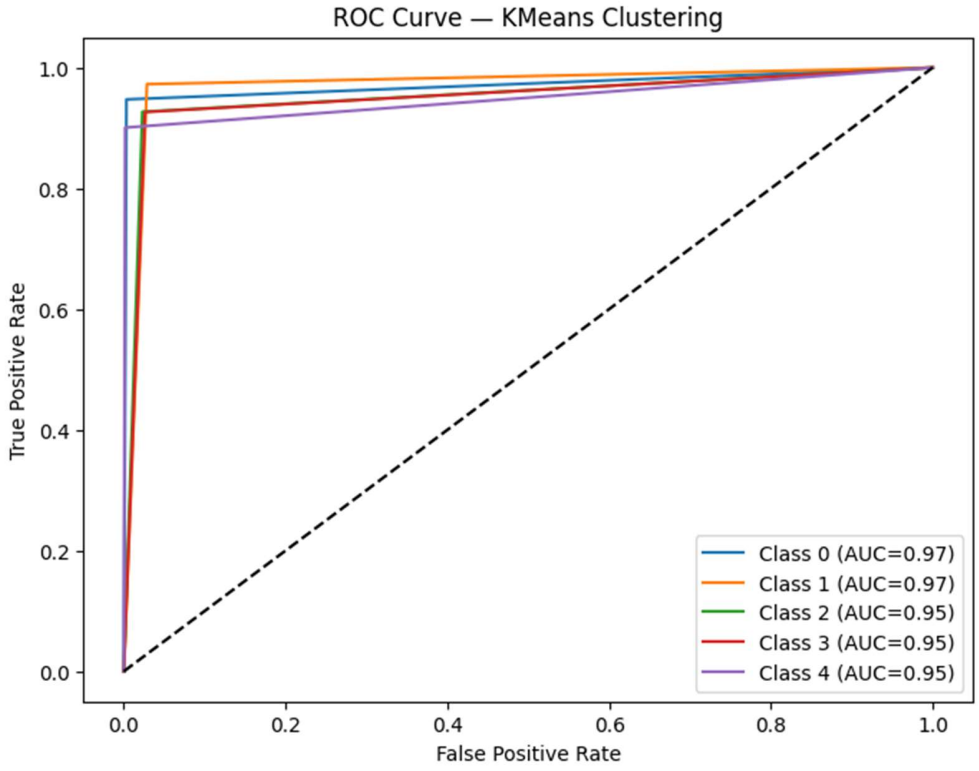
With MobileNetV2 features, K-Means achieved a **training accuracy of 92.09%** and a **test accuracy of 93.43%**. While this performance is strong for an unsupervised method, it is still lower than that of Logistic Regression. This gap is expected because K-Means assumes spherical clusters and does not take class boundaries into account.

t-SNE visualizations of the feature space showed partial separation between classes, along with overlapping regions.



Log loss:

ROC Curve:

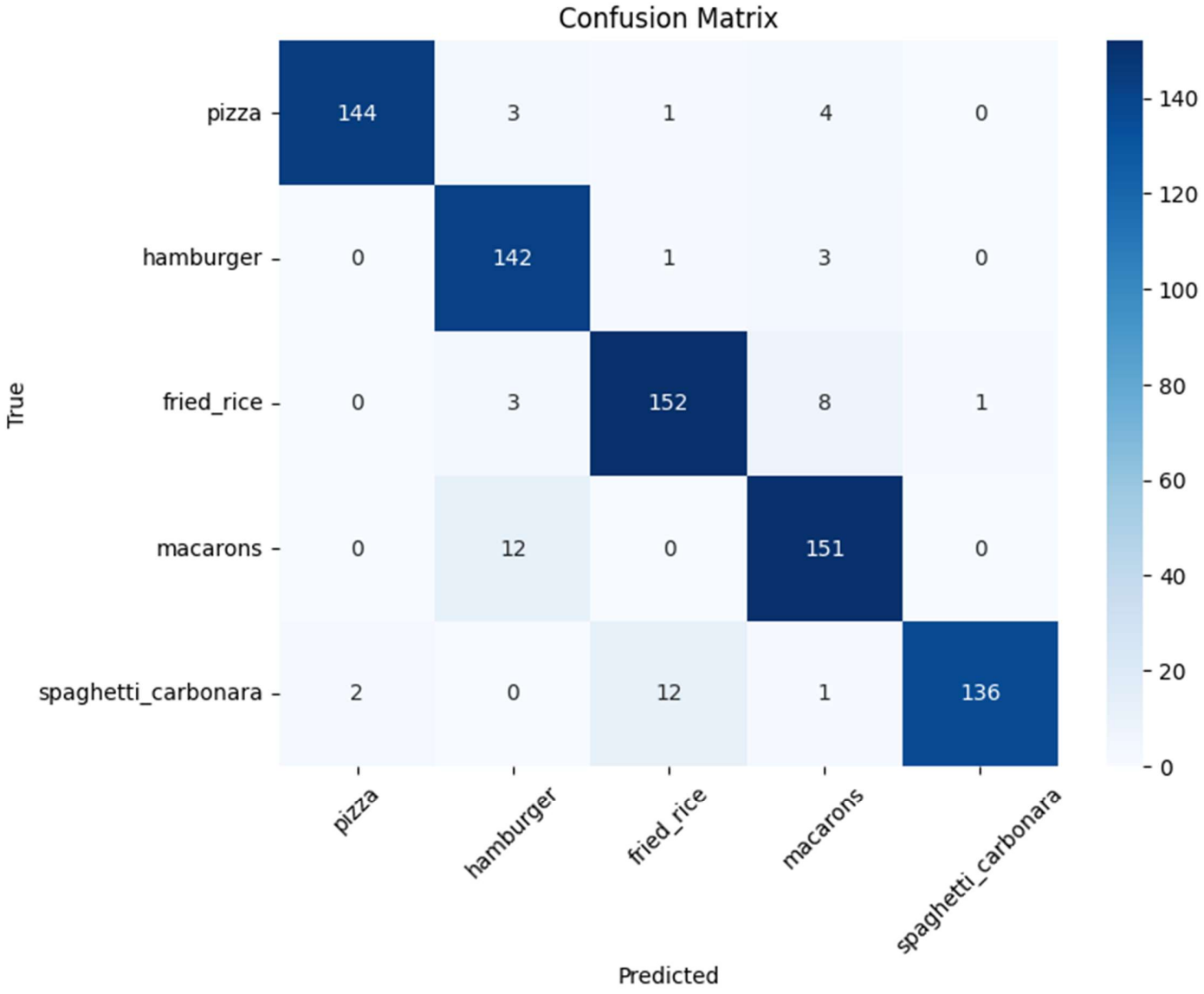


Classification Report:

Classification Report (Test):

	precision	recall	f1-score	support
pizza	0.99	0.95	0.97	152
hamburger	0.89	0.97	0.93	146
fried_rice	0.92	0.93	0.92	164
macarons	0.90	0.93	0.92	163
spaghetti_carbonara	0.99	0.90	0.94	151
accuracy			0.93	776
macro avg	0.94	0.93	0.94	776
weighted avg	0.94	0.93	0.93	776

These overlaps explain the misclassifications observed in the confusion matrix.



2.8 Experiments Without MobileNetV2

In a separate notebook, additional experiments were conducted without using MobileNetV2 in order to evaluate the effectiveness of handcrafted features compared to deep features. In this setup, feature extraction relied on traditional computer vision techniques, mainly **Histogram of Oriented Gradients (HOG)** combined with color histograms. Because handcrafted features are generally less expressive than deep features, **data augmentation** played a critical role in attempting to improve model generalization.

2.8.1 Data Augmentation Strategy

Data augmentation was applied at the image level before feature extraction. The main goal of augmentation was to artificially increase data diversity and simulate real-world variations that the model might encounter during inference. Each image had a probability of undergoing one or more augmentation operations, ensuring that the augmented dataset preserved variability without becoming unrealistic.

Several augmentation techniques were applied:

- **Image flipping** was performed horizontally with a random probability. This operation simulates changes in camera viewpoint and food orientation. Since food objects do not have a fixed left-right orientation, horizontal flipping is a valid transformation that helps the model become invariant to mirror views.
- **Random rotation** was applied within a limited angle range. Small rotations simulate changes in camera angle and imperfect alignment during image capture. This helps the model become more robust to slight orientation changes while preserving the semantic identity of the food item.
- **Brightness adjustment** was used to simulate different lighting conditions. Real-world food images are captured under varying illumination environments, such as indoor lighting, daylight, or shadows. By randomly increasing or decreasing brightness, the model is encouraged to focus on structural and texture-based cues rather than absolute pixel intensity.
- **Contrast adjustment** was also applied to account for differences in image quality and camera settings. Changing contrast alters the separation between light and dark regions in an image, which affects edge visibility. This augmentation helps reduce sensitivity to contrast variations across images.
- **Gaussian noise** was optionally added to some images. This simulates sensor noise and compression artifacts commonly found in real-world images, especially those captured with mobile devices. Adding noise improves robustness and discourages the model from overfitting to overly clean patterns.

After augmentation, features were extracted using HOG descriptors and color histograms. HOG captures local gradient and edge information, while color histograms provide global color distribution cues. These features were then concatenated into a single feature vector and standardized before being passed to the models.

2.8.2 Impact of Augmentation on Performance

Although data augmentation improved robustness compared to using raw handcrafted features alone, the overall performance remained significantly lower than the MobileNetV2-based pipeline. Despite these enhancements, the performance of models trained without deep features was significantly lower. Logistic Regression achieved a training accuracy of approximately **99.95%**, but the test accuracy dropped sharply to **48.6%**, indicating severe overfitting. Similarly, K-Means achieved a training accuracy of **27.95%** and a test accuracy of **30.1%**, reflecting poor clustering quality in the handcrafted feature space.

These results clearly demonstrate that handcrafted features, even when combined with augmentation, are not sufficient to capture the complex visual patterns present in food images.These results demonstrate that while augmentation is beneficial, it cannot fully compensate for the representational gap between handcrafted features and deep convolutional features. This further highlights the importance of deep feature extraction for complex image classification tasks such as food recognition.

2.9 Conclusion

This project highlights the effectiveness of transfer learning for image classification tasks. Deep features extracted using MobileNetV2 provide rich and discriminative representations that enable even simple models such as Logistic Regression to achieve excellent performance. While K-Means clustering benefits from high-quality features, supervised learning methods consistently outperform unsupervised approaches for this task.

The comparison between deep feature extraction and handcrafted features further emphasizes the superiority of convolutional neural networks for complex visual recognition problems. Overall, the project demonstrates that combining deep feature extractors with classical machine learning models is a powerful and computationally efficient strategy for food image classification.

	MobileNetV2 Feature Extractor		**HOG Feature Extractor**	
	Logistic	**K means**	**Logistic**	**K means**
Train Accuracy	100.00%	92.09%	99.95%	27.95%
Test Accuracy	96.91%	93.43%	48.6%	30.1%