

**BAX 423 Final Project**  
**Grocery Shopping Recommendation Algorithm**

Group 11

Jiayi Jiang, Shulang (Simon) Ning, Xinyu Liu

## I. Business Objectives

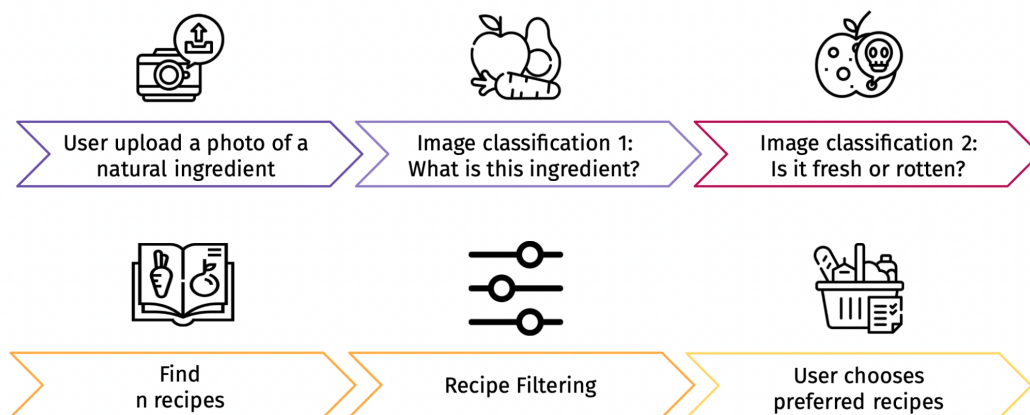
In the world of fitness and diet industry, we've noticed several limitations. One common struggle is in creating customized shopping experiences. Another area of concern is for unique dietary needs. Whether users have allergies, intolerances, or specific nutrition plans, current apps usually fall short in providing these personalized and suitable options. We desire to shed light on these unmet demands and provide a solution that caters to a broader range of fitness and diet enthusiasts.

To address these issues, our project objective is to develop a user-friendly website for recipe selection and grocery shopping. Our website aims to accurately identify food items, provide information on their safety and quality, and offer customized recipe suggestions that align with your dietary requirements.

The target audience for our website includes individuals who desire to eat more natural food, have a fast-paced lifestyle, have their own fitness and nutrition plan, and struggle with making recipe choices. The website aims to provide user-friendly interfaces and personalized solutions to meet their fitness and nutritional goals.

## II. Key Actionable Business Initiative

To accomplish our objective of creating a website that offers personalized recipe and grocery shopping recommendations, our project will incorporate ingredient classification deep learning models and recipe recommendation algorithms. Additionally, we will provide users with various customization options to enhance their experience.



The user journey on our website will consist of six stages. Firstly, users will have the ability to upload a photo of a natural ingredient, such as vegetables or herbs, to the website. Our first image classification CNN model will accurately identify the specific natural ingredient from the

uploaded photo. Following that, the second CNN model will assess the quality of the ingredient based on image features, distinguishing between good and rotten items.

Once the ingredient is identified, the system will fetch multiple recipes from our recipe dataset that include the identified ingredient. Users will then have the option to apply filters based on cuisine style preference, diet preferences, and allergies to refine the recipe selection. Ultimately, the filtered recipes, along with the complete list of required ingredients, will be displayed on the webpage.

To assist users with their grocery shopping, they will have the ability to take a screenshot of the recipe and ingredient list output. This will provide them with a convenient reference for their next shopping trip.

### **III. Metrics of Success/KPIS**

#### **1. Define key success metrics**

We have adopted distinct success metrics for each stage of our project, as different data requirements exist for each phase.

For ingredient classification, we measure success based on classification accuracy. This metric evaluates how accurately the system can classify ingredients. This is also the metrics referred to when selecting the model to be connected to the demo.

In the case of recipe recommendation, success is determined by the number of available recipes that match the ingredients recognized from the uploaded photo.

For the pre-designed recipe filtering option, the success metric considers the total combination of preferences available to users. Examples include combinations like "Mexican x Vegetarian" or "Chinese x High Protein." This metric assesses the flexibility of filtering options for users.

#### **2. Hypothesis of business initiative's impact on metrics**

By implementing our business initiative, which involves web scraping to collect recipe and ingredient image data, as well as deep learning techniques for ingredient image classification, we anticipate significant improvements in our defined success metrics.

Firstly, the web scraping process will allow us to gather sufficient data from various sources such as Google and popular recipe websites like allrecipes.com. This data accumulation is crucial as it ensures we have a sufficient quantity of high-quality data to train and test our deep learning models for image classification.

Additionally, this data will enable us to recommend a larger number of recipes to users based on the recognized ingredients, enhancing the overall user experience. Furthermore, the availability of diverse data will provide users with more customization options for filtering the recommended recipes.

Thirdly, by leveraging the TensorFlow framework, we will have the opportunity to experiment with different neural network structures, model tuning techniques, and utilize different dataset for model training. This aspect of our business initiative allows us to continuously evolve and improve our deep learning models over time.

## IV. Analytics Plan

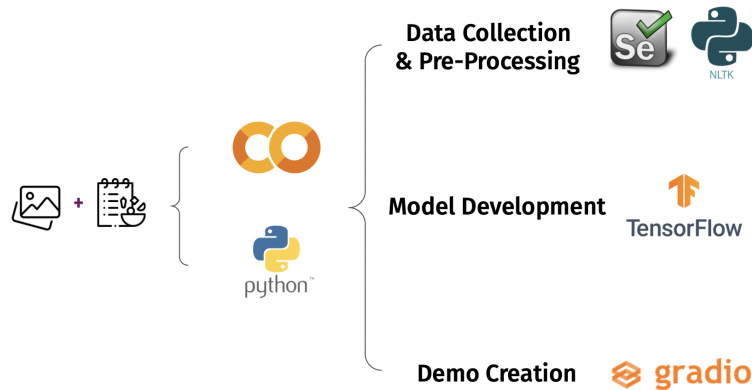
### 1. Role of Analytics

Analytics played a vital role in guiding and refining our business initiative at different stages. Initially, we leveraged analytics to identify the specific area to focus on. By examining the top diet apps and conducting an App Store search, we observed that most apps primarily emphasized calorie counting rather than recipe recommendation. Also, there is a gap in the intersection of natural food image classification, recipe recommendation and grocery shopping suggestions. Thus, we made an informed decision to concentrate our project on this underexplored niche.

As we started to collect image and recipe data and run different image classification models, analytics had helped us track our progress. For recipe and image dataset building, we were tracking the net number of recipes and images collected. For image classification models, the results were used not only to tune the models, but also to direct image dataset update and augmentation.

	Classify Image	Recommend Recipes	Customize Shopping List
Data Source	<b>Ingredient Classification</b> V1: 27k scraped pics V2: 27k handpicked+augmented pics V3: a backup set from kaggle	<b>457 recipes</b> scraped from allrecipes.com	<b>3 diet types:</b> Veggie, High Protein, Normal <b>7 styles:</b> CHN, IND, KOR, JPN, GRC, MEX, ITA
	<b>Fresh/Rotten Classification</b> 1 Image Set from Kaggle		
Success Metrics	Classification Accuracy	Number of Recipes Available per Ingredient	Number of Customized Combinations Available (e.g. Vegetarian x Mexican, High Protein x Korean, etc.)

## 2. Analytics implementation plan



To implement our analytics plan, we will stack several technologies. First, the recipe dataset will be built with web scraping, followed by an NLP preprocessing to format the recipe ingredients and finding the more popular ingredients to include in the image dataset. Then, the image datasets will be created by image scraping as well as image augmentation. The CNN model will be built under the framework of TensorFlow.

## V. Dataset Building

### 1. Data source and dataset development

For the three types of data utilized for the project, we collected them through different sources and with different methods.

#### a. Recipe dataset

To build the recipe dataset, we scraped 457 recipes from allrecipes.com. For each individual recipe, we retained essential details including the recipe name, the list of ingredients required, and a link to the original recipe page. These components enable users to easily identify recipes of interest and access additional information directly from the source.

#### b. Pre-designed customization options

To provide customization options for our users, we have incorporated the original options found on allrecipes.com as a foundation. These options serve as a starting point for personalizing recipes according to individual preferences.

In addition to the options available on allrecipes.com, we also included the three most common diet types and common allergies for further customization.

### **c. Three ingredient image datasets.**

#### **(1) Web-scraped ingredient image dataset (self-built)**

To create the initial version of our self-built image dataset, we utilized web scraping techniques to gather images from Google Image. This dataset consists of 30 distinct ingredients, with a total of 900 jpg images for each ingredient, resulting in a comprehensive collection of 27,000 images.

Subsequently, we divided the image dataset into three distinct groups for the purpose of running our image classification CNN model. The training set consisted of 600 images, the validation set consisted of 150 images, and the test set included another 150 images.

It is important to note that at this stage, only basic cleaning procedures were conducted on the dataset before employing it to train our deep learning classification model. Unfortunately, this limited cleaning approach resulted in subpar training outcomes, which did not meet our expectations.

#### **(2) Augmented ingredient image dataset (self-built)**

Following the training results of the initial dataset, we decided to construct a second image dataset using a different approach. Our methodology involved a multi-step process.

Firstly, we manually selected a subset of 200-300 high-quality images from the original image dataset. These images were carefully chosen to ensure their overall quality and suitability for training purposes.

Next, we applied various augmentation techniques to expand the size of the high-quality image pool. By employing these augmentation methods, we were able to create additional variations and augmentations of the original images, thereby enriching our dataset.

Lastly, we randomly sampled an equal number of images from the augmented image pool to form the training, validation, and test sets. This ensured a fair distribution of images across the different sets.

The utilization of this methodology proved to be highly effective, resulting in a significant improvement in the training results of our model.

#### **(3) Backup dataset (Kaggle)**

As a backup option, we obtained a supplementary dataset from Kaggle. This image dataset comprises 15 different ingredients and is sufficiently extensive to adequately support the fitting of our CNN model.

## **2. Explanatory variables and target outcome**

In this dataset, the explanatory variables are the vegetable images. Each image represents a specific type of vegetable and serves as input to the model for classification. The features of the images, such as color, shape, texture, and size, provide the information for the model to learn and make predictions. We expect that, in the later stage, the CNN model is trained to learn the visual features and patterns that distinguish one vegetable from another. These images are well-labeled into 15 distinct vegetable categories, and in JPG format and have a size of 224×224 pixels.

The response variable in this dataset is the class or label assigned to each vegetable image. The goal is to classify each vegetable image into one of the 15 predefined classes accurately, followed by a food quality (fresh or rotten) identification using CNN models. The response variable provides the ground truth for the model's training and evaluation process.

### **3. Explore and document the variation in the datasets**

The variation in the datasets can be identified in terms of explanatory and response variables.

For the explanatory variable of natural ingredient images (vegetables and herbs), all images are in jpg format, and the number of images for each ingredient category and their training, validation, and test group is equal. Meanwhile, these images vary in terms of shape, color, texture, and the size, backgrounds (white, kitchens, farms or markets), noise level, and lighting conditions. The consistency and variation altogether makes sure that the CNN model is trained on a balanced dataset.

On the other hand, the response variable consists of 30 classes, representing different types of vegetables, including highly utilized ingredients according to the recipes, such as garlic, onion, tomato, bean, broccoli, etc.. The dataset aims to provide a balanced distribution of images, ensuring all 15 classes are represented adequately.

## **VI. Analytics Methodology**

### **1. Type of Analytics and Methodology: Description of predictive analytics**

The main objective of this dataset is to facilitate predictive analytics through the utilization of advanced deep learning techniques, specifically Convolutional Neural Network (CNN) models. These models are trained to accurately classify vegetables and determine their quality based on the analysis of their respective images.

By harnessing the hierarchical structure inherent in CNNs, these models possess the ability to automatically learn and identify pertinent features from the visual data of the vegetables. This capability significantly enhances the accuracy of classification outcomes. The integration of convolutional, pooling, and fully connected layers, coupled with the meticulous training and

optimization procedures, empowers CNN models to effectively accomplish predictive analytics tasks within our project.

To provide more insight into our approach, we present a breakdown of the CNN layers that were employed in this project:

1. Convolutional Layers: detect local patterns and features in the images by performing convolutions across the image pixels. Multiple filters are used in our algorithm to capture different features, such as edges, textures, and shapes.
2. Pooling Layers: added to reduce the spatial dimensions of the feature maps after convolutional layers. Pooling helps to capture the most relevant features while reducing the sensitivity to small spatial variations. The pooling technique we used in our algorithm is max pooling.
3. Fully Connected Layers: serve as the classifier and are responsible for mapping the extracted features to the final class probabilities. Activation functions, such as ReLU (Rectified Linear Unit), are used in our algorithm between these layers to introduce non-linearity.

During training, the model learns the optimal combination of weights and biases that minimize a defined loss function. Backpropagation is used to propagate the error gradient from the output layer to update the model's parameters using optimization algorithms like Adam. The training process iterates over the dataset for multiple epochs, refining the model's parameters to improve its predictive capability.

Then, in the hyperparameter Tuning process, the hyperparameters we included in our algorithm are learning rate, batch size, number of layers, filter sizes. The validation dataset is instrumental in guiding the selection of optimal hyperparameters. The hyperparameter configuration that yields the best validation performance is

```
epochs=100,  
  
verbose=1,  
  
validation_data=val_image_generator,  
  
steps_per_epoch = 3000//6,  
  
validation_steps = 300//6,  
  
callbacks=early_stopping
```



Finally, Early stopping prevents overfitting and determines the optimal number of training epochs. During training, the model's performance on the validation dataset is monitored. For instance, the early stopping process is triggered when the validation performance fails to improve over a period of 5 epochs and the validation performance starts to deteriorate. The training stopped early based on the validation performance, saving our computational resources and preventing our CNN model from overfitting.

## **2. Model findings**

### **(1) Relationship between model output and explanatory variables**

As for the causal inferences, while our models can provide insights into the relationship between the vegetable images and their corresponding labels, they are not explicitly designed to establish a causal relationship. If we want to incorporate causal manners, additional experimental or observational studies involving manipulating specific factors (such as storage conditions or ripeness) and observing their effects on vegetable quality are needed.

While our models can contribute valuable insights and correlations, it is essential to recognize the limitations and the need for further research to establish causal relationships in the context of vegetable quality.

### **(2) Prediction output and informative features**

Ensuring the best possible prediction is a priority in our approach, and we achieve this by incorporating a validation set and employing early stopping techniques. The validation set is utilized to evaluate the model's performance during training. Early stopping is a technique that continuously monitors the validation performance and halts the training process if the model's performance on the validation set begins to deteriorate. This prevents overfitting and helps us select the model with the best generalization capability.

In a CNN model, the convolutional layers play a crucial role in automatically extracting relevant features from input images. These features encompass color variations, textures, and shapes that differentiate different types of vegetables or indicate their freshness levels.

As the network trains, the weights of the filters within the convolutional layers are adjusted to enhance the features that are more relevant to the objective of the task. This adaptive adjustment of weights allows the model to generalize and focus on the most salient features within the images. Our assumption is that by assigning higher weights to these informative features, the model becomes more adept at capturing and leveraging the key visual patterns associated with vegetable classification and quality assessment.

### **(3) Actions based on model training results**

The CNN algorithm mainly focuses on training to classify vegetables and evaluate its performance. After obtaining predictions, we incorporate a website demo on gradio with prescriptive analytics by suggesting specific actions or interventions based on the predicted vegetable category or freshness level. For example, if a vegetable is classified as semi-rotten, our website demo interface will recommend immediate consumption or proper storage to prevent further deterioration.

### **3. Model description**

(1) Vegetable classification CNN model and Fresh/rotten classification CNN model:

The CNN image classification models follows the sequential architecture with the following layers:

- a. Convolutional Layers: Two Conv2D layers with 32 and 64 filters, respectively, using a kernel size of 3x3 and a ReLU activation function. Each convolutional layer is followed by a MaxPooling2D layer with a pool size of 2x2.
- b. Flatten Layer: Flattens the output of the convolutional layers into a 1D vector.
- c. Fully Connected Layers: Two Dense layers with 128 units and ReLU activation functions. A Dropout layer with a rate of 0.25 is included between the dense layers to prevent overfitting.
- d. Output Layer: A Dense layer with 3 units and a softmax activation function, representing the three vegetable categories: fresh, semi-rotten, and rotten.

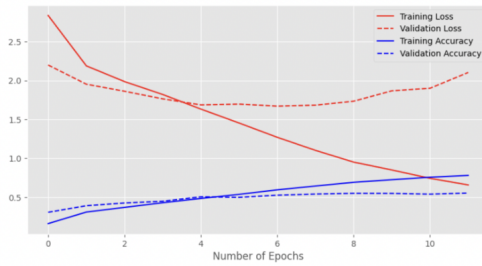
The model is trained using the Adam optimizer and categorical cross-entropy loss. Early stopping is used with a patience of 5 to monitor the validation performance and stop training if it does not improve.

Main analytics results

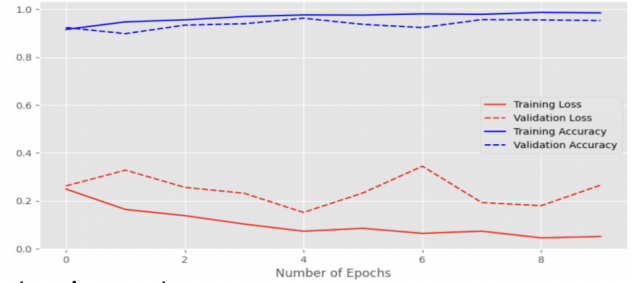
#### **1. Ingredient recognition CNN model**

The evaluation criteria for ingredient classification is centered around classification accuracy. After 10 epochs and utilizing scraped images, our model achieved a loss of 1.98 and an accuracy of 0.55. The accompanying graph provides a visual representation of the training loss, validation loss, training accuracy, and validation accuracy as they evolve over the course of the epochs.

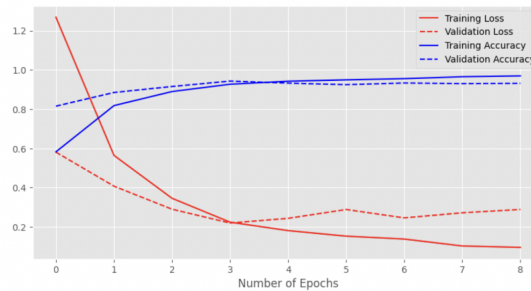
**Model 1 With scraped images**  
**Evaluation:** loss: 1.9843, accuracy: 0.5544



**Model 2 With handpicked and augmented images**  
**Evaluation:** loss: 0.2605, accuracy: 0.9568



**Model 3 With the backup image dataset**  
**Evaluation:** loss: 0.2669, accuracy: 0.9337

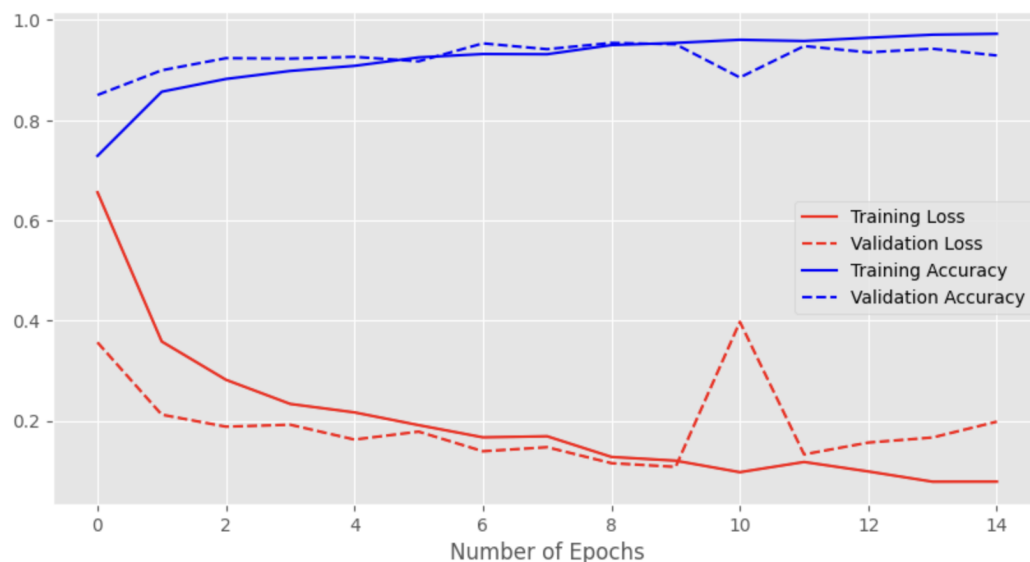


The above graphs indicate a comparison of the 3 models we used, the comparison indicates that Model 2 outperformed the other models with the lowest evaluation loss 0.26 and highest accuracy 0.95.

## (2) Fresh/rotten identification CNN model

The evaluation criteria for fresh/rotten classification is centered around classification accuracy. After 15 epochs and utilizing Kaggle dataset images, our model achieved a loss of 0.16 and an accuracy of 0.94. The accompanying graph provides a visual representation of the training loss, validation loss, training accuracy, and validation accuracy as they evolve over the course of the epochs.

### Fresh/rotten Classification Evaluation




## VII. Demo Development

### a. Ingredient recognition and recipe recommendations:

The website offers users the ability to efficiently identify ingredients through image recognition and receive personalized recipe recommendations tailored to their preferences. Here are the website instructions:

1. Image Upload: Capture a photograph of the desired ingredient, such as a tomato, and proceed to upload it to the system.
2. Cuisine Selection: Indicate your preference for a specific regional cuisine, for instance, Indian food, to refine the recipe recommendations accordingly.
3. Preference Specification: State your desired vegetable preferences. If you are pursuing weight loss, the system can identify "high-protein" options specifically suited to your objectives. If you have any allergies, kindly enter this information as well.

After submitting the provided information, the system will promptly process the data and promptly present the top recipe choices that align with your preferences.



region

Indian

food\_type

High Protein

allergy

mushroom

Clear

Submit

Recipe 1

Name: chef johns mulligatawny soup

Region: Indian

Ingredients: unsalted butter, vegetable oil, boneless, skinless chicken thighs, kosher salt, or more to taste, curry powder, garam masala, ground cumin, freshly ground black pepper, ground mustard, ground coriander, cayenne pepper, garlic, minced, finely grated fresh ginger, yellow onion, diced, carrots, diced, celery, diced, tomato paste, cubed Yukon Gold potatoes, seeded and diced tomato, Granny Smith apples - peeled, cored, and diced, dry red lentils, bay leaf, tamarind paste, chicken broth, coconut cream (Optional), plain Greek yogurt, or to taste, chopped fresh cilantro, or to taste, sliced green onion, or to taste, red pepper flakes, or to taste

Link: <https://www.allrecipes.com/recipe/8466000/chef-johns-mulligatawny-soup/>

Recipe 2

Name: indian butter chicken chicken makhani

Region: Indian

Ingredients: garam masala, tandoori masala powder, Madras curry powder, ground cumin, ground cardamom, ground cayenne pepper, salt and ground black pepper to taste, boneless, skinless chicken thighs, cut into bite-size pieces, butter, divided, yellow onion, chopped, garlic, minced, lemon juice, chopped fresh ginger, tomato puree, half-and-half, plain yogurt, cashews, fresh cilantro, stems removed

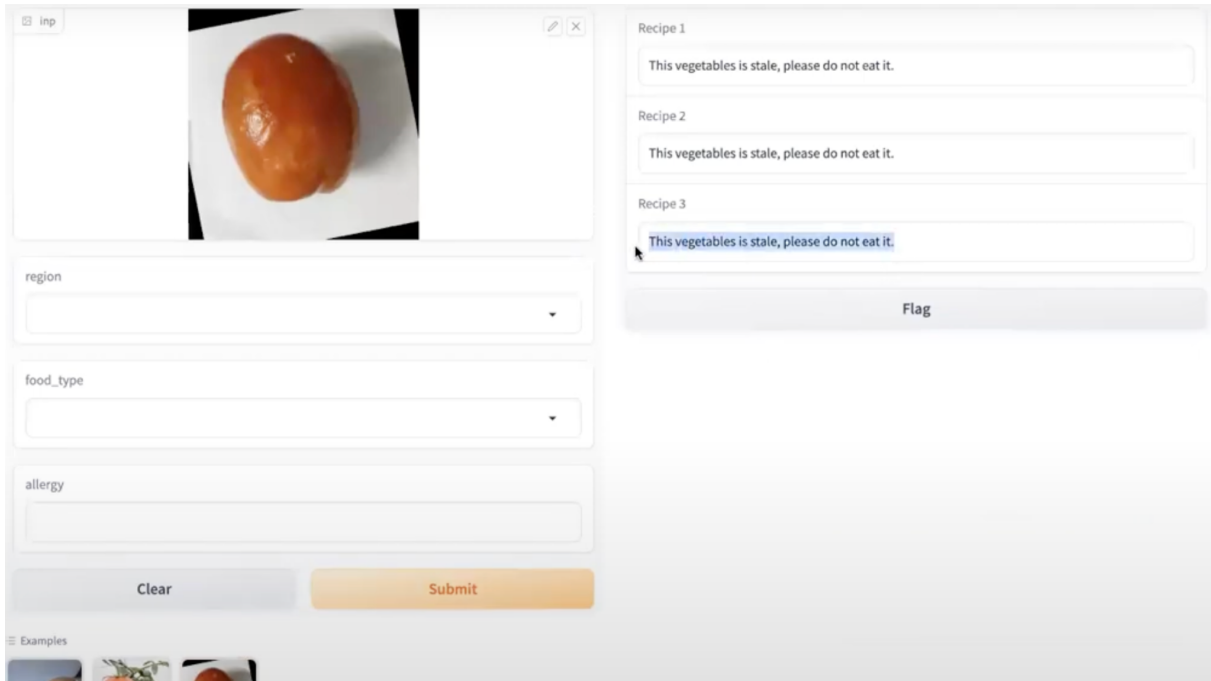
Link: <https://www.allrecipes.com/recipe/246717/indian-butter-chicken-chicken-makhani/>

Recipe 3

Name: easy indian butter chicken

b. Fresh/rotten food quality classification:

In addition, we have incorporated a food quality feature into our website. When a user uploads an image of a grocery item, such as a frostbitten tomato, the system will promptly generate warning messages to alert the user: “This vegetable is stale, please do not eat it.”

The screenshot displays a web application interface. On the left, there is an image upload section with a placeholder 'inp' and a small image of a tomato. Below the image are three dropdown menus labeled 'region', 'food\_type', and 'allergy'. At the bottom of this section are 'Clear' and 'Submit' buttons. To the right, there is a list of three recipes. Each recipe has a text box containing the message 'This vegetables is stale, please do not eat it.' Below the list of recipes is a 'Flag' button. At the bottom left, there is an 'Examples' section with three small images of different food items.

## VIII. Limitations and Potential Solutions

### 1. Single ingredient vs multiple ingredient in the image

The existing ingredient classification CNN model has a limitation in that it can only identify one ingredient in an image. The model's ability to recognize a specific ingredient depends on both its appearance in the image and the model's proficiency in recognizing that particular ingredient in that specific context. However, users may upload images that contain multiple ingredients, and if the model fails to recognize and classify all of them, it would result in a diminished user experience.

To address this challenge, we have devised a potential solution by incorporating two additional components into our current image classification model.

#### (1) Before image classification: Object detection and insurance segmentation

To determine the appropriate number of objects to aim for during the image classification process, as well as locate each object in the picture, it is necessary to incorporate an

object detection and instance segmentation algorithm before iterating the image classification.

This can be achieved by utilizing established object detection algorithms like Mask R-CNN. The algorithm can detect and locate objects within an image. The number of objects recognized will be stored as the number of iterations to run the image classification algorithm to recognize all ingredients captured in the picture. Locating the objects can be useful for precise segmentation and identification of individual fruits.

## (2) Iterate the image classification process of single ingredient

After obtaining the number of objects and segmenting the picture using the object detection algorithm, the image classification CNN model will be iterated for each detected segment. The purpose of these iterations is to ensure that all objects within the segments are accurately recognized and classified.

For each segment, the image classification CNN model will be applied to analyze and determine the specific ingredient or object present. By iterating the image classification process for each segment, the model can effectively classify all the objects that were initially detected in the image.

Once the identification and classification of ingredients are completed, the output will be cleaned to provide the user with the total number of a specific ingredient they have in the image. Each recognized ingredient will be accounted for only once. A counting mechanism will be applied to determine the number of occurrences for each ingredient so as to provide the user with an accurate count of how many instances of a specific ingredient were identified in the image.

## **2. Overfitting issue**

Since the source of image is similar for the best performing CNN model at the moment, it is highly possible that the model currently has overfitting issues. Potential solutions includes: (1) increase the source of image, including utilizing publicly available image datasets, such as ImageNet, CIFAR, or Open Images; (2) modify the current augmentation methods; (3) introduce regularization terms in training; (4) adjust value of hyperparameters; (5) try other model structures; (6) utilize pre-trained models that have been trained on larger and more diverse datasets.

## **3. Other potential options and features**

### (1) Multiple pictures at a time

Currently, the model allows users to upload only one image at a time. Another potential feature to consider is enabling users to upload multiple pictures simultaneously and

aggregating the recognized ingredients. This would result in more accurate and customized recipe recommendations.

(2) Recommending ingredients already in the pictures

The ingredients identified in the uploaded pictures will still be included in the ingredient list for the current version. By preprocessing the recipe ingredient text data, we can filter out ingredients that the user already has based on the uploaded pictures. This would make the shopping list more precise.

(3) Fresh/rotten classification as an embedded feature of classification process

The fresh/rotten classification currently operates as a subsequent step rather than an integrated part of ingredient recognition. Rotten ingredients will still be identified and considered for recipe recommendations in the current version. To address this issue, we can integrate the fresh/rotten classification model into the overall classification process to provide a comprehensive output for ingredient classification.

(4) Ingredient counting and calorie counting

The model does not currently account for numerical features of ingredients, such as the exact quantity of each vegetable owned or needed, as well as their calorie and nutrition information. This feature could be implemented once the model can recognize multiple ingredients, generate counting results, and connect with a separate dataset to calculate calories.

(5) More recipes and pre-designed filters

Currently, the available recipes and filtering options are limited. The reason for not expanding the selection is the lack of understanding regarding user demand. As the website is used and more user data is collected, it can serve as a reference to expand the existing recipe database and offer a wider range of pre-designed filters.

## Appendix

Recipe scraping, NLP preprocessing and ingredient frequency counting:

[https://colab.research.google.com/drive/1ntzbL0mYY0LFfmg5kx9k2j\\_NufSBsIrX?usp=sharing](https://colab.research.google.com/drive/1ntzbL0mYY0LFfmg5kx9k2j_NufSBsIrX?usp=sharing)

Image cramping and train/validation/test splitting:

[https://colab.research.google.com/drive/1og\\_CcnEzYFiG6K9tTBs1JqD8w2DWUyXv?usp=sharing](https://colab.research.google.com/drive/1og_CcnEzYFiG6K9tTBs1JqD8w2DWUyXv?usp=sharing)

Image augmentation:

[https://colab.research.google.com/drive/1j\\_BhVrAh\\_N-zvfXwGkwqBZKZq6Xarxvl?usp=sharing](https://colab.research.google.com/drive/1j_BhVrAh_N-zvfXwGkwqBZKZq6Xarxvl?usp=sharing)

Model with uncleaned data:

[https://colab.research.google.com/drive/1r3\\_wYTJnHiJb9Mf-NY\\_s0ocZ26mz2SHM?usp=sharing](https://colab.research.google.com/drive/1r3_wYTJnHiJb9Mf-NY_s0ocZ26mz2SHM?usp=sharing)

Model with cleaned data:

<https://colab.research.google.com/drive/1zXrwh523qDwb-MrUE3PJRP9nNPILgsx?usp=sharing>

Model with kaggle data:

<https://colab.research.google.com/drive/1cIvue1YnBall0XtnIoUyqDkot8vJzcOX?usp=sharing>

Model with small data (ingredient and fresh/rotten) but with demo:

[https://colab.research.google.com/drive/11Qms3sMtCmUJpXvcBcIbt0Fsbou\\_mVuK?usp=sharing](https://colab.research.google.com/drive/11Qms3sMtCmUJpXvcBcIbt0Fsbou_mVuK?usp=sharing)

Image dataset 1 (self-built; original):

[https://drive.google.com/drive/folders/1i9n2ZDd\\_TldwgcItYJXSef-g6wiNDGYs?usp=sharing](https://drive.google.com/drive/folders/1i9n2ZDd_TldwgcItYJXSef-g6wiNDGYs?usp=sharing)

Image dataset 2 (self-built; augmented):

[https://drive.google.com/drive/folders/1i9n2ZDd\\_TldwgcItYJXSef-g6wiNDGYs?usp=sharing](https://drive.google.com/drive/folders/1i9n2ZDd_TldwgcItYJXSef-g6wiNDGYs?usp=sharing)