Grading and sorting of fruits and vegetables using classical image processing techniques

Project overview:

I developed an automated system that grades and sorts fresh fruits and vegetables. specifically talking, lemons and mangoes, with the help of classical image processing techniques. The system processes images by enhancing brightness and contrast, removing backgrounds via HSV thresholding and morphological operations, extracting features, and finally classifying produce as "Good" or "Bad." The goal is to help farmers and companies quickly distinguish high-quality produce from defective items

Data acquisition and dataset preparation:

The two datasets I have obtained from Kaggle and merged them into one consolidated dataset:

- Mango dataset: Download from Fruit and Vegetable Disease Healthy vs. Rotten. I am using only the mango images from this dataset and have removed images of other fruits and vegetables.
- Lemon dataset: Download from Lemon Quality Dataset. I am using only the good quality and bad quality lemon images. I have renamed the subfolders to:
    - lemon__good
    - lemon__bad

I merged the mango and lemon images (with their respective quality labels) into a single folder named dataset. Each subfolder in dataset belongs to one class (lemon__good, lemon__bad, mango_good, mango_bad). I am including a small, representative sample of the dataset (about 5 images from each subfolder) for the submission.

Folder structure:

The files in this project is well organized which is as follows:

```
/dataset
    lemon_good      # Folder with Images of good quality lemons
    lemon_bad       # Folder with Images of poor quality lemons
    mango_good      # Folder with Images of good quality mangoes
```

mango_bad # Folder with images of poor quality mangoes
/projectcode.ipynb # Jupyter Notebook with all the source code
/projectcode.pdf # PDF version of the code
/README.pdf # This file

Note: I am not including the full original datasets in the repository. Instead, I provide these instructions so that anyone can download the datasets from Kaggle and reproduce the merging process.

Step by step instructions to run the code:

1. **Install all the Dependencies required:**
   first, Python 3 should be installed, install the required libraries by running:
   pip install opencv-python numpy matplotlib scikit-image scikit-learn seaborn pandas.

2. **Prepare the Dataset:**

   - Download the Lemon and mango Dataset from Kaggle and extract only the good quality and bad quality images.
   - Organize the images into subfolders by renaming them as below:
   - lemons to lemon__good and lemon__bad, similaly for mangoes
   - Merge these subfolders into a single folder and name it as dataset.

3. **Open the Notebook:**
   Launch Jupyter Notebook and open the file projectcode.ipynb.

4. **Run All Cells:**
   Execute all cells in the same order. Notebook will load the images, preprocess and extract the features. Then, split the data into training, validation, and test sets. Followed by evaluating the classifier.

5. **View the Outputs:**
   The notebook displays:
   Class distribution plots, Sample processed images (original, background-removed, edge maps, histograms, and final predictions). Evaluation metrics such as accuracy, a confusion matrix, and a classification report.

Libraries, Tools, and Code Releases:

- OpenCV: Used for image loading, resizing, color conversion, and background removal.
- NumPy: Used for numerical operations and array manipulation.
- scikit-image: Used for feature extraction (e.g., applying the Sobel filter and computing image entropy).
- scikit-learn: Used for splitting the dataset and evaluating the classifier.
- Matplotlib & Seaborn: Used for visualization and plotting graphs.
- Pandas: Used for handling data and plotting feature distributions.
- Jupyter Notebook: The project is implemented in projectcode.ipynb. All code is self-contained in this notebook.

Code Description:

- **Image loading & preprocessing**: I am loading images from subfolders and resizing them to 128x128 pixels. Gamma correction is applied to adjust brightness.

- **Background Removal**: I convert images to HSV. I use different HSV threshold ranges to capture green, yellow, orange, and red. This helped in separating the produce from the background. I improved the mask using Gaussian blur. These included opening, closing, and dilation. Later I used Canny edge detection to enhance the segmentation.

- **Feature extraction:** I extracted features by making histograms for the Hue, Saturation, and Value channels. I also calculated edge density using the Sobel filter and measured image entropy.

- **Dataset Splitting & Evaluation:** I split the dataset into training (60%), validation (20%), and test (20%) sets. The notebook optimized thresholds using the training set. It then evaluated a simple classifier based on the features.

- **Visualization**: I did plot class distributions, feature distributions, and hue histograms. Then displayed sample outputs from the test set showing the original images, one's with no backgrounds, edge maps, and final

Running Example:

Steps to run the project:

- Open the projectcode.ipynb notebook in Jupyter Notebook.
- Execute all cells sequentially. (Make sure the dataset is downloaded and placed in proper path.)
- The notebook will automatically load the dataset, preprocess the images, extract features, split the data, evaluate the model, and display results.