1. Explain Image Preprocessing Steps

- Loading Images: The function load_and_preprocess_data reads images from specified directories for different categories.
- Resizing: Each image is resized to 150x150 pixels to standardize the input size. This ensures that all images have the same dimensions, which is crucial for batch processing and neural network input.
- Grayscale Conversion: Images are converted to grayscale using cv2.cvtColor. This reduces the complexity by eliminating color information, which might not be necessary for the classification task.
- Histogram Equalization: This step enhances the contrast of the images by redistributing the intensities. cv2.equalizeHist is used to achieve this, which can help in highlighting features in images that might otherwise be too dark or too bright.

2. Explain the Importance of Your Selected Feature Sets for This Image Classification Task

- Histogram of Pixel Intensities: Captures the overall intensity distribution in the image. Useful for distinguishing between images with different lighting or overall brightness.
- Sobel Gradients: Captures edge information in both horizontal and vertical directions. Edges are fundamental in recognizing shapes and boundaries within images.
- Histogram of Oriented Gradients (HOG): Captures the distribution of gradients or edge directions. This is particularly useful for object detection and capturing structural information in images.
- Local Binary Pattern (LBP): Captures texture information. It is robust to monotonic illumination changes and is useful in recognizing different textures in images.

3. Apply Appropriate Techniques for Dimensionality Reduction
Given the large size of the combined feature set, dimensionality reduction is essential to avoid the curse of dimensionality, improve model performance, and reduce computational costs. In the code, Principal Component Analysis (PCA) is used.

PCA reduces the number of features while retaining the most important variance in the data. By selecting 150 components, we ensure that we capture the most significant patterns in the data while reducing noise and redundancy.

4. Evaluation of the Trained Models Using Appropriate Metrics
The trained models are evaluated using accuracy and a classification report that includes precision, recall, and F1-score

Accuracy: Measures the proportion of correctly classified instances out of the total instances.

Classification Report: Provides detailed metrics such as precision (positive predictive value), recall (sensitivity), and F1-score (harmonic mean of precision and recall) for each class. These metrics give a more comprehensive evaluation of model performance, especially in cases of class imbalance.

5. Development of a Flask Application with Image Upload Functionality for Classification from Best Performing Model

Flask application is developed in app2.py file. From where you can upload a images and it will give the output based on the classification.

6. Guide on Setting Up a Flask Application for Local Image Classification

Create a virtual environment

Then run: pip install -r requirements.txt
It will install all the dependencies required for the project

Then run flask application: python app2.py
And then access the application on the browser using `http://127.0.0.1:5000`