PCA+CNN+SVM IMPLEMENTATION

PSEUDO CODE

```
1. Import required libraries for image processing, machine learning, deep learning, visualization, and federated setup.

2. Mount Google Drive and define path to the dataset folder.

3. Define a function to:
   - Load images and resize to 224x224
   - Extract labels from folder names
   - Return image array and label array

4. Collect all image paths and apply the image loader function to get image data and labels.

5. Convert class labels to binary matrix format using LabelBinarizer.

6. Split dataset into training and test sets (90% training, 10% testing).

7. Reshape training images to 2D arrays and standardize using StandardScaler.

8. Apply PCA to reduce dimensionality to 20 components for both training and test sets.

9. Define a function to:
   - Create `num_clients` with randomized and equally sharded data
   - Return a dictionary mapping each client name to its data

10. Define a function to:
    - Convert each client’s data into batched TensorFlow datasets
    - Shuffle and batch each shard
```

11. Process and batch data for all clients and also prepare test set batches.

12. Define federated parameters like number of communication rounds, optimizer, loss, and metrics.

13. Define helper functions:

   a. Compute weight scaling factor for each client based on their data size.

   b. Scale model weights by scalar.

   c. Sum scaled weights across all clients.

   d. Evaluate global model on test data using accuracy and loss.

14. Define a class that:

   - Builds a CNN using dense layers
   - Appends a simulated SVM via a final softmax dense layer
   - Returns the combined model

15. Initialize the global CNN-SVM model.

16. One-hot encode `y_train` and `y_test` for multiclass classification.

17. Begin federated learning training loop:

   For each communication round:

   a. Copy global model weights

   b. Shuffle client list

   c. For each client:

      - Create a new CNN-SVM local model
      - Load global weights
      - Train on local data for fixed epochs
      - Scale and collect local weights
      - Clear session after each client

d. Average all local weights to update the global model

18. After each round:

a. Predict global model on test data

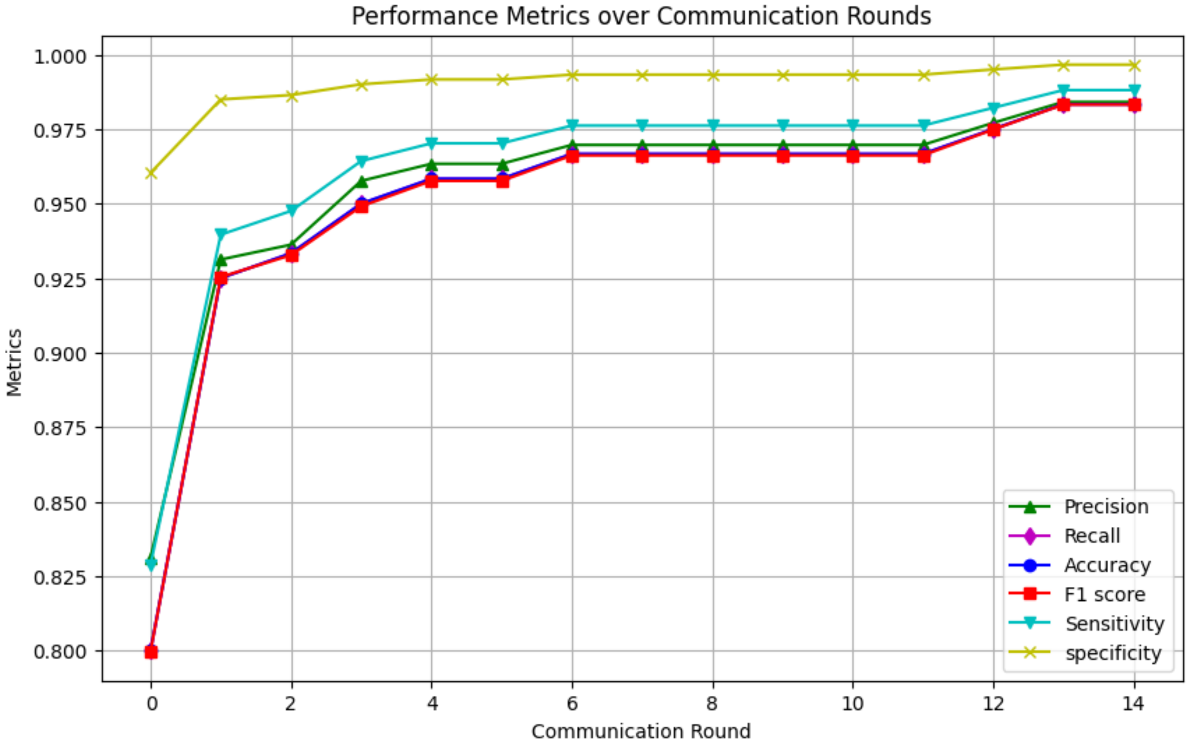
b. Calculate Accuracy, Loss, Precision, Recall, F1 Score

c. Use confusion matrix to compute Sensitivity and Specificity

d. Append all metrics to respective lists for plotting

19. Plot Accuracy, Precision, Recall, F1 Score, Sensitivity, and Specificity over communication rounds.

PCA+CNN+SVM


Performance Metrics over Communication Rounds
Metrics
Communication Round
Precision
Recall
Accuracy
F1 score
Sensitivity
specificity