Machine Learning Internship

Task for ML Intern

The Short Report is as follows:

1. Preprocessing Steps and Rationale

To ensure optimal model performance, we applied the following preprocessing steps:

- **Image Resizing**: All images were resized to (224, 224, 3) to maintain consistency with CNN input requirements.
- **Normalization**: Pixel values were scaled between 0 and 1 to stabilize training.
- **Data Splitting**: The dataset was divided into training and testing sets to evaluate model generalization.
- **Handling Class Imbalance**: Techniques such as data augmentation were considered to balance class distribution.

2. Insights from Dimensionality Reduction

- **Principal Component Analysis (PCA)** was initially explored to visualize high-dimensional data.
- **Findings**: PCA indicated that significant variance was retained within a few components, suggesting some redundancy in features.
- **Impact**: While PCA is not directly used in CNNs, insights guided architectural decisions to prevent overfitting.

3. Model Selection, Training, and Evaluation Details

- **Model Chosen**: A Convolutional Neural Network (CNN) was selected for feature extraction and classification.
- **Architecture**:
    - Conv2D layers with ReLU activation for feature extraction.
    - MaxPooling layers to reduce dimensionality.
    - Fully connected (Dense) layers for classification.

- **Training Details**:
  - Optimizer: Adam
  - Loss Function: Categorical Crossentropy
  - Epochs: 10
  - Batch Size: 32
- **Evaluation Metrics**:
  - Accuracy: 28%
  - Regression Metrics: MAE = 0.0, RMSE = 0.0 (Indicating an issue with evaluation settings)
  - Visualization: Scatter plot of actual vs. predicted values showed inconsistencies in classification.

4. Key Findings and Suggestions for Improvement

Key Findings:

- Model achieved **28% accuracy**, which is lower than expected for a CNN.
- There were potential **data inconsistency issues** affecting training.
- Regression metrics yielded **inconsistent results** due to incorrect evaluation settings.

Suggestions for Improvement:

1. **Increase Dataset Size**: Using data augmentation or collecting more images can improve performance.
2. **Optimize Hyperparameters**: Experimenting with learning rate, batch size, and architecture modifications can enhance accuracy.
3. **Use Transfer Learning**: Implementing pre-trained models like VGG16, ResNet, or EfficientNet can boost accuracy significantly.
4. **Fix Evaluation Issues**: Ensure correct application of regression metrics and classification evaluation.
5. **Enhance Preprocessing**: Exploring more advanced augmentation techniques could improve model robustness.