**Detailed Report for Notebook V31**

**1. Overview**

**Objective:**  
The primary goal of Notebook V31 is to develop a binary classification model using DenseNet201 for the RSNA Pneumonia Detection Challenge. The notebook aims to classify chest X-ray images as either “Normal” or “Pneumonia.” In addition, the model addresses two major challenges:

* **Class Imbalance:** Only a fraction of images show pneumonia.
* **Overfitting:** The model is at risk of memorizing training data, so careful regularization and fine-tuning are necessary.

**2. Data Preparation**

**a. Dataset Characteristics**

* **Source:** RSNA Pneumonia Detection Challenge data.
* **Image Format:** The training images are provided in DICOM (.dcm) format.
* **Labels:**
  + **Bounding Box Information:** Provided in CSV files; if an image has at least one bounding box, it is labeled as Pneumonia (1), otherwise Normal (0).

**b. DICOM Conversion**

* **Issue:** Keras’ ImageDataGenerator does not support DICOM files.
* **Solution:**
  + A conversion cell was added that uses pydicom and OpenCV to convert all DICOM images to PNG format.
  + The conversion includes normalization and a conversion from grayscale to RGB to match DenseNet201’s expected input shape (224×224×3).
  + The file paths in the label DataFrame are updated to reference these newly created PNG images.

**c. Data Splitting & Augmentation**

* **Stratified Split:**
  + The dataset is split into training and validation sets using a stratified method (via train\_test\_split) so that both sets maintain a similar class distribution.
* **Data Augmentation:**
  + The training data is augmented with rotations, zoom, shear, width/height shifts, and horizontal flipping to increase image diversity.
  + The validation data is only rescaled.

**3. Model Architecture**

**a. Base Model: DenseNet201**

* **Pretrained on ImageNet:**
  + The DenseNet201 model is used as the base, excluding the top classification layers.
* **Feature Extraction:**
  + Initially, all layers of DenseNet201 are frozen to leverage the pretrained weights.

**b. Custom Classification Head**

* **Layers Added:**
  + **SpatialDropout2D (0.5):** Provides regularization by dropping entire feature maps.
  + **GlobalAveragePooling2D:** Reduces spatial dimensions.
  + **Dense Layer (64 units, ReLU activation) with L2 Regularization:** Further processes extracted features.
  + **Dropout (0.5):** Additional regularization.
  + **Output Dense Layer (1 unit, Sigmoid activation):** Provides the binary classification output.

**4. Loss Function & Handling Class Imbalance**

**a. Custom Focal Loss Function**

* **Reasoning:**
  + Focal loss focuses on hard-to-classify examples, which is particularly useful when handling class imbalance.
  + Instead of relying on TensorFlow Addons (which has compatibility warnings), a custom focal loss function is defined.
* **Implementation Highlights:**
  + **Clipping:** Ensures that predictions are clipped to avoid log(0) issues.
  + **Weighted Cross-Entropy:** Applies a modulating factor (gamma) and balance factor (alpha) to standard binary cross-entropy.

**b. Class Weights**

* **Computation:**
  + Class weights are computed using sklearn’s compute\_class\_weight function.
  + These weights are passed into the model.fit() call via the class\_weight parameter, ensuring that the minority class (Pneumonia) contributes more to the loss.

**5. Training Procedure**

**a. Callbacks**

* **EarlyStopping:**
  + Monitors val\_auc and stops training if there is no improvement for 5 epochs, restoring the best weights.
* **ReduceLROnPlateau:**
  + Reduces the learning rate by a factor of 0.2 if val\_auc does not improve for 3 epochs.
* **ModelCheckpoint:**
  + Saves the best model using the .keras file extension to comply with the new Keras model format requirements.

**b. Progressive Fine-Tuning**

1. **Initial Training (Custom Head Only):**
   * The base DenseNet201 model remains frozen.
   * The custom head is trained for an initial set of epochs (e.g., 5 epochs).
2. **Gradual Unfreezing:**
   * After initial training, the last 50 layers of the base model are unfrozen for fine-tuning.
   * The model is recompiled with a lower learning rate (e.g., 1e-5) and trained further (e.g., 10 epochs) with continued usage of class weights and callbacks.

**6. Evaluation**

* **Metrics:**
  + The model’s performance is evaluated using AUC (Area Under the ROC Curve), classification reports, and confusion matrices.
* **Observations:**
  + Evaluation cells provide loss and AUC values on the validation set.
  + A confusion matrix and classification report help identify precision, recall, and F1-score for both classes.

**7. Conclusions & Future Work**

**Conclusions:**

* **Data Preparation:**
  + Converting DICOM to PNG was essential to use Keras’ data generators effectively.
  + Stratified splitting and data augmentation contributed to a more robust training process.
* **Model Training:**
  + The progressive fine-tuning strategy (first training the custom head and then unfreezing parts of DenseNet201) is crucial to adapt the pretrained features to the RSNA dataset.
  + Custom focal loss combined with class weights helps mitigate the impact of class imbalance.

**Future Work:**

* **Hyperparameter Tuning:**
  + Further tuning of augmentation parameters, learning rates, dropout rates, and the focal loss parameters (gamma and alpha) might yield additional improvements.
* **Advanced Augmentation:**
  + Consider integrating techniques like MixUp or CutMix.
* **Model Interpretability:**
  + Implement Grad-CAM to visualize model attention and verify that the model is focusing on relevant regions in the X-rays.
* **Deployment:**
  + Once satisfied with performance, consider steps to deploy the model for real-world inference.

This report encapsulates the end-to-end process for Notebook V31—from data conversion to model evaluation—emphasizing practical solutions for handling class imbalance and overfitting in the RSNA Pneumonia Detection task.