Task for AI/ML Internees

WEEK 1:

Disease Prediction Using Patient Data

1. Objective: Train and evaluate machine learning models to predict diseases like diabetes or heart disease.

2. Tasks:

- Download the dataset from the UCI Machine Learning Repository.
- Perform data preprocessing: Handle missing values, perform normalization,
 and encode categorical features.
- Exploratory Data Analysis (EDA):
- Analyze feature distributions and correlations.
- Visualize data using histograms, scatter plots, and heatmaps.
- o Train models using Logistic Regression, Random Forest, and SVM.
- o Evaluate models with metrics like accuracy, precision, recall, and F1-score.
- o Compare model performances and identify the best-performing algorithm.
- 3. Outcome: A report summarizing the analysis, model performances, and insights.

WEEK 2:

Cancer Detection Using Histopathological Images

1. **Objective:** Use deep learning to detect cancerous cells from histopathological images.

2. Tasks:

o Access the Breast Cancer Histopathological Dataset from Kaggle.

 Perform data augmentation to balance the dataset and enhance generalizationImplement a Convolutional Neural Network (CNN) using frameworks like

TensorFlow or PyTorch.

- Explore Transfer Learning with pre-trained models like ResNet or VGG16.
- o Highlight cancerous regions in the images using Image Segmentation or

Grad-CAM visualization techniques.

3. Outcome: A deep learning model that accurately detects and highlights cancerous Regions.

WEEK 3:

Objective

Develop practical skills in applying Deep Learning for medical image classification. The tasks focus on **Skin Cancer Detection** and **Pneumonia Detection** using CNNs. Internees will preprocess datasets, apply data augmentation, and fine-tune pre-trained models for accurate classification.

Tasks Overview

1. Skin Cancer Detection

Task: Classify skin lesions into categories (e.g., benign or malignant).
 Dataset: ISIC Skin Cancer Dataset.

Steps to Follow:

1. Data Preprocessing:

- o Load the dataset, normalize pixel values to [0, 1].
- Resize images to a fixed size (e.g., 224x224 for compatibility with CNNs).
- o Split the dataset into training, validation, and testing sets.

2. Data Augmentation:

 Apply techniques like random rotation, flipping, zooming, and brightness adjustment.

3. Model Development:

- Use a pre-trained CNN model (e.g., ResNet50 or EfficientNet) with transfer learning.
- Fine-tune the model by replacing the last fully connected layer with a dense layer for binary classification.
- o Use Binary Crossentropy as the loss function and Adam optimizer.

4. Evaluation:

• Evaluate the model using accuracy, precision, recall, and F1-score.

5. Deliverables:

- Code file (skin_cancer_detection.py or notebook).
- A short report (1-2 pages) detailing model performance and insights.

2. Pneumonia Detection from Chest X-Rays

• Task: Classify chest X-ray images into pneumonia-positive or negative. • Dataset: Chest X-Ray Images Dataset (Kaggle).

Steps to Follow:

1. Data Preprocessing:

- Load and preprocess images (resize to 224x224, normalize pixel values).
- o Divide the dataset into training, validation, and testing sets.

2. Data Augmentation:

 Apply augmentation techniques (e.g., random cropping, rotation, and histogram equalization).

3. Model Development:

- Implement a CNN architecture or fine-tune a pre-trained model (e.g., MobileNet or InceptionV3).
- Optimize the model with Categorical Crossentropy loss and Adam optimizer.

4. Evaluation:

- Use metrics like sensitivity, specificity, and ROC-AUC score.
- o Compare results on augmented vs. non-augmented datasets.

5. Deliverables:

- Code file (pneumonia_detection.py or notebook).
- A summary report (1-2 pages) on model performance and challenges.

Learning Goals

- Learn **Transfer Learning** for medical image classification.
- Gain hands-on experience with data preprocessing and augmentation.
- Understand evaluation metrics (e.g., sensitivity, specificity, F1-score).
- Explore the impact of fine-tuning pre-trained models.

Tools & Libraries

- Frameworks: TensorFlow/Keras or PyTorch.
- Libraries: OpenCV, NumPy, Pandas, Matplotlib, Scikit-learn.
- Hardware: Google Colab or a local GPU-supported system.

Submission Requirements

- 1. Code files or notebooks for each task.
- 2. Reports summarizing findings, challenges, and insights.
- 3. Model files (optional) and charts/graphs showing results.
- 4. Deadline: End of Week 4 (20 Dec).

Bonus Task

For enthusiastic internees, encourage:

- Experimenting with ensemble models combining multiple pre-trained CNNs.
- Hyperparameter tuning (learning rate, batch size, epochs) for performance improvement.