DL-WEEK-3

Use scikit-learn/keras/tensorflow:

1. **Implement and demonstrate Multi-Layer Perceptron (MLP) model as a classifier with back propagation on synthetic dataset.**
2. You may prefer to create your own dataset (synthetic) using **make_classification()** fn., MLPClassifier()from scikit-learn library
3. Evaluate the **performance of the classifier by tabulating metrics:** Accuracy, precision, Recall, F1-score, Specificity, ROC curves, Confusion matrix, etc.,. Particularly check for under-fitting, over-fitting, and generalizability.
4. Improve the performance of the classifier with different techniques: check the training, validation errors. Apply different resampling techniques (k-fold cross validation, stratified sampling, LOOCV etc.,),change the hyperparameters, etc.,
5. Perform the above steps 3,4 on any standard dataset of your choice, and document all your observations carefully .

****

Ex: code snippet for generating synthetic dataset

```
from sklearn.datasets import make_classification, make_blobs

# 1. make_classification: Complex dataset with noisy features
X_class, y_class = make_classification(
    n_samples=100, n_features=2, n_informative=2,
    n_redundant=0, n_clusters_per_class=1,
    class_sep=1.0, random_state=42
)

# 2. make_blobs: Isotropic Gaussian blobs for clustering
X_blobs, y_blobs = make_blobs(
    n_samples=100, centers=3, n_features=2,
    cluster_std=1.0, random_state=42
)
```

Note:

There are two ways to generate synthetic data: make_classification(), make_blobs()

- make_classification is for testing classifiers; make_blobs is for testing clustering algorithms like K-Means.
- make_classification allows defining n_informative, n_redundant, and n_repeated features. make_blobs simply generates data around centers.
- make_classification can produce non-convex, overlapping, and complex boundaries, whereas make_blobs produces spherical (isotropic) clusters

****