Assignment No 1

**Feed-Forward Neural Network**

Name : Samiksha More

PRN : 22420127

Roll : 381070

**Problem Statement:**

Implementing Feedforward neural networks in Python using Keras and TensorFlow.

**Objective:**

The aim is to design and implement a Feedforward Neural Network (FNN) using Keras and TensorFlow to solve supervised learning problems such as classification and regression. The primary objective is to understand how data flows through layers, how the model learns using backpropagation, and how Keras simplifies neural network development.

**Theory:**

A Feedforward Neural Network is the most fundamental type of artificial neural network. It consists of:

* Input Layer: Receives the raw data.
* Hidden Layers: Perform transformations and extract features using weighted connections and activation functions.
* Output Layer: Produces the final prediction (e.g., class label or numerical value).

**Key concepts:**

* Activation Functions introduce non-linearity (ReLU, Sigmoid, Tanh).
* Forward Propagation is the process where input data passes through layers to produce predictions.
* Loss Function measures the error between predictions and true values.
* Backpropagation adjusts the weights to minimize loss using optimization algorithms such as Gradient Descent or Adam.
* Epochs and Batches define how many times the dataset is passed through the model and in what size.

**Methodology:**

1. Data Collection and Preprocessing: Import dataset, normalize values, and split into training and testing sets.
2. Model Design: Define the architecture (number of layers, neurons per layer, activation functions).
3. Compilation: Choose optimizer (e.g., Adam), loss function (e.g., cross-entropy or mean squared error), and evaluation metrics (e.g., accuracy).
4. Training: Feed data into the model, perform forward propagation, compute loss, and adjust weights using backpropagation.
5. Validation and Evaluation: Test the model on unseen data to check accuracy and generalization.
6. Prediction: Use the trained model for classification or regression tasks.

**Advantages**

* Easy to implement using high-level APIs like Keras.
* Capable of modeling complex non-linear relationships.
* Flexible for both regression and classification tasks.
* Forms the foundation of advanced deep learning models.

**Limitations**

* Requires large datasets for good performance.
* Computationally expensive, especially with many layers.
* Can suffer from overfitting without proper regularization.
* Not suitable for sequential or spatial data compared to RNNs and CNNs.

**Applications**

* Classification: Image recognition, spam detection, disease prediction.
* Regression: House price prediction, demand forecasting.
* Recommendation Systems: E-commerce product recommendations.
* Basic Natural Language Processing: Sentiment analysis, keyword classification.

**Working / Algorithm**

1. Input data is passed into the input layer.
2. Each neuron calculates a weighted sum of inputs and applies an activation function.
3. Data moves forward through hidden layers to extract features.
4. The output layer generates final predictions.
5. A loss function calculates the error between predicted and actual output.
6. Using backpropagation, weights are updated to minimize error.
7. Steps are repeated for multiple epochs until the model converges.

**Conclusion:**

Feedforward Neural Networks are the backbone of deep learning. Implementing them in Python using Keras and TensorFlow makes the process easier, faster, and more efficient. While they have limitations in handling sequential and spatial data, they remain powerful tools for general classification and regression tasks and serve as a foundation for more complex architectures such as CNNs and RNNs.