**Assignment No. 1**

**Name: Pranav Zaware**

**PRN: 22311462**

**Batch: TY-A2**

**Problem Statement**

Implementing Feedforward neural networks in Python using Keras and TensorFlow

**Objective**

* To understand the structure and working of feedforward neural networks.
* To preprocess and prepare real-world datasets for training neural networks.
* To design and implement a multiclass FFNN using Keras and TensorFlow.
* To evaluate the model’s performance using metrics such as accuracy, confusion matrix, and classification report.
* To visualize the training and validation performance across epochs.

**Software and Hardware Requirements**

* **Operating System:** Windows/Linux/MacOS
* **Kernel:** Python 3.x
* **Tools:** Jupyter Notebook, Anaconda, or Google Colab
* **Hardware:** Minimum 4 GB RAM; GPU optional for faster training
* **Libraries:** TensorFlow, Keras, NumPy, Pandas, Matplotlib, Scikit-Learn

**Theory**

A **Feedforward Neural Network (FFNN)** is an artificial neural network where information flows in one direction—from input to output—without cycles.

* **Input Layer:** Accepts feature values (e.g., chemical properties of wine).
* **Hidden Layers:** Perform non-linear transformations using activation functions like ReLU.
* **Output Layer:** Produces predictions. For multiclass classification, a **softmax** function is used to output probabilities across multiple classes.

**Training** is performed using **backpropagation** and **gradient descent**, where weights are updated iteratively to minimize the loss function (categorical cross-entropy in this case).

**Methodology**

**1. Data Acquisition**

* Used the **Wine Quality dataset (winequality-red.csv)**, containing physicochemical properties of red wine and their quality ratings (3–8).

**2. Data Preparation**

* Split the dataset into **training (80%)** and **testing (20%)** sets.
* **Label Encoding:** Converted wine quality scores into numeric classes.
* **One-hot Encoding:** Transformed class labels into a categorical format for softmax output.
* **Feature Scaling:** Standardized numerical features using StandardScaler.

**3. Model Architecture**

* Sequential FFNN model with:
  + Input layer (128 neurons, ReLU activation)
  + Hidden layer (64 neurons, ReLU activation)
  + Dropout layers (to prevent overfitting)
  + Output layer (6 neurons with **softmax** activation for multiclass classification)

**4. Model Compilation**

* Optimizer: **Adam**
* Loss function: **Categorical Cross-entropy**
* Metric: **Accuracy**

**5. Model Training**

* Trained the model for up to 100 epochs with **early stopping** to avoid overfitting.
* Batch size: 32

**6. Model Evaluation**

* Evaluated the model on test data using the accuracy score.
* Generated **confusion matrix** and **classification report** (precision, recall, F1-score).

**7. Visualization**

* Plotted training vs validation accuracy/loss to track model performance.

**Advantages**

* Handles **non-linear relationships** between features and wine quality.
* Scales well with more features and larger datasets.
* Good **generalization** with dropout and regularization.
* Provides **probabilistic predictions** for multiple classes.

**Limitations**

* Requires **large amounts of data** for best performance.
* Computationally more expensive than simple classifiers.
* May **overfit** if not properly regularized.
* Limited interpretability (black-box model).

**Applications**

* Predicting wine quality in the **food and beverage industry**.
* Quality control in **manufacturing processes**.
* Classification tasks in healthcare, finance, and IoT.
* Any domain requiring **multiclass decision-making**.

**Conclusion**

In this assignment, a **Feedforward Neural Network** was successfully implemented to perform **multiclass classification** of wine quality. By preprocessing the data, applying one-hot encoding, and normalizing features, the network achieved good accuracy on unseen test data. The results showed that FFNNs can effectively capture non-linear relationships in real-world datasets. However, careful tuning of hyperparameters and the use of regularization techniques are necessary to improve performance further.