LP5 GROUP B

1.Linear Regression :

A statistical method to model the relationship between a dependent variable and one or more independent variables using a straight line:

$y = mx + b$,

where y is the predicted value, x is the input, m is the slope, and b is the intercept.

2.Neural Network :

A neural network is a computer system inspired by the human brain, made up of layers of connected nodes (“neurons”) that learn to recognize patterns and make decisions.

3.Neural Network Architecture:

The structure of a neural network includes:

1. **Input Layer**: Receives the raw data
2. **Hidden Layers**: Perform computations and pattern learning using neurons and activation functions.
3. **Output Layer**: Produces the final result or prediction.

Connections between layers have weights that are adjusted during training.

4.Classification using deep neural networks :

It is a popular approach to solve various supervised learning problems such as image classification, text classification, speech recognition, and many more. In this approach, the neural network is trained on labeled data to learn a mapping between the input features and the corresponding output labels.

5.Binary classification using deep neural networks :

It involves designing a neural network architecture with multiple layers of interconnected neurons, training the network on labeled data using a suitable loss function, and using the trained network to classify new data.

6.Convolutional Neural Networks (CNNs) :

CNNs are a class of artificial neural networks that are specially designed to analyze and classify images, videos, and other types of multidimensional data. They are widely used in computer vision tasks such as image classification, object detection, and image segmentation.

The main idea behind CNNs is to perform convolutions, which are mathematical operations that apply a filter to an image or other input data.By applying different filters at each layer, the network learns to detect different features in the input data, such as edges, shapes, and textures.

7.Layers in CNN :

1. **Convolutional Layer** – Extracts features from input using filters.
2. **ReLU Layer** – Applies activation function to introduce non-linearity.
3. **Pooling Layer** – Reduces spatial size (e.g., max pooling).

4. **Fully Connected Layer** – Connects all neurons for final classification.
5. **Output Layer** – Produces the final predictions (e.g., softmax)

8.Architecture of CNN :

1. **Input** – Raw image data (e.g., 32x32 pixels, RGB).
2. **Conv (Convolution)** – Detects patterns/features using filters.
3. **ReLU (Activation)** – Adds non-linearity to the model.
4. **Pool (Pooling)** – Downsamples to reduce size and computation.
5. **(Repeat)** – Stacks more layers for deeper feature learning.
6. **Fully Connected** – Combines features to make decisions.
7. **Output** – Gives final class probabilities (e.g., via softmax).

9.What is a Recurrent Neural Network?

A recurrent neural network (RNN) is a type of neural network that is designed to work with sequential data. Unlike traditional feedforward neural networks that only process input data in a single pass, RNNs maintain an internal state or memory that allows them to process sequences of input data.

This makes RNNs well-suited for tasks such as natural language processing, speech recognition, and time series analysis.

10.Architecture of RNN :

RNN Architecture (Very Short):
Input → Hidden State (recurrent) → Output

1. **Input** – Sequence data (e.g., words, time series).
2. **Hidden State** – Remembers past info through recurrence.
3. **Output** – Predicts based on current input and memory.

11.Activation function :
Adds non-linearity to help neural networks learn complex pattern.

12. Sigmoid Activation Function :
Maps input values to a range between 0 and 1, often used for binary classification.

13.ReLU :

Outputs zero for negative values and the input itself for positives, enabling faster training.

14. Word Embedding :
Represents words as dense vectors in a continuous space, capturing semantic meanings.

15. To implement a deep neural network on the Boston Housing Dataset, we can follow these steps: (can be applied to all dl codes explainations)

1. **Load the dataset**: We can load the dataset using libraries like pandas or numpy.
2. **Preprocess the data**: We need to preprocess the data by scaling the input features so that they have zero mean and unit variance. This step is important because it helps the neural network to converge faster.

3. **Split the dataset**: We split the dataset into training and testing sets. We can use a 70/30 or 80/20 split for training and testing, respectively.
4. **Define the model architecture**: We need to define the architecture of our deep neural network. We can use libraries like Keras or PyTorch to define our model. The architecture can include multiple hidden layers with various activation functions and regularization techniques like dropout.
5. **Compile the model**: We need to compile the model by specifying the loss function, optimizer, and evaluation metrics. For regression problems like this, we can use mean squared error as the loss function and adam optimizer.
6. **Train the model**: We can train the model using the training data. We can use techniques like early stopping to prevent overfitting.
7. **Evaluate the model**: We can evaluate the model using the testing data. We can calculate the mean squared error or the mean absolute error to evaluate the performance of the model.