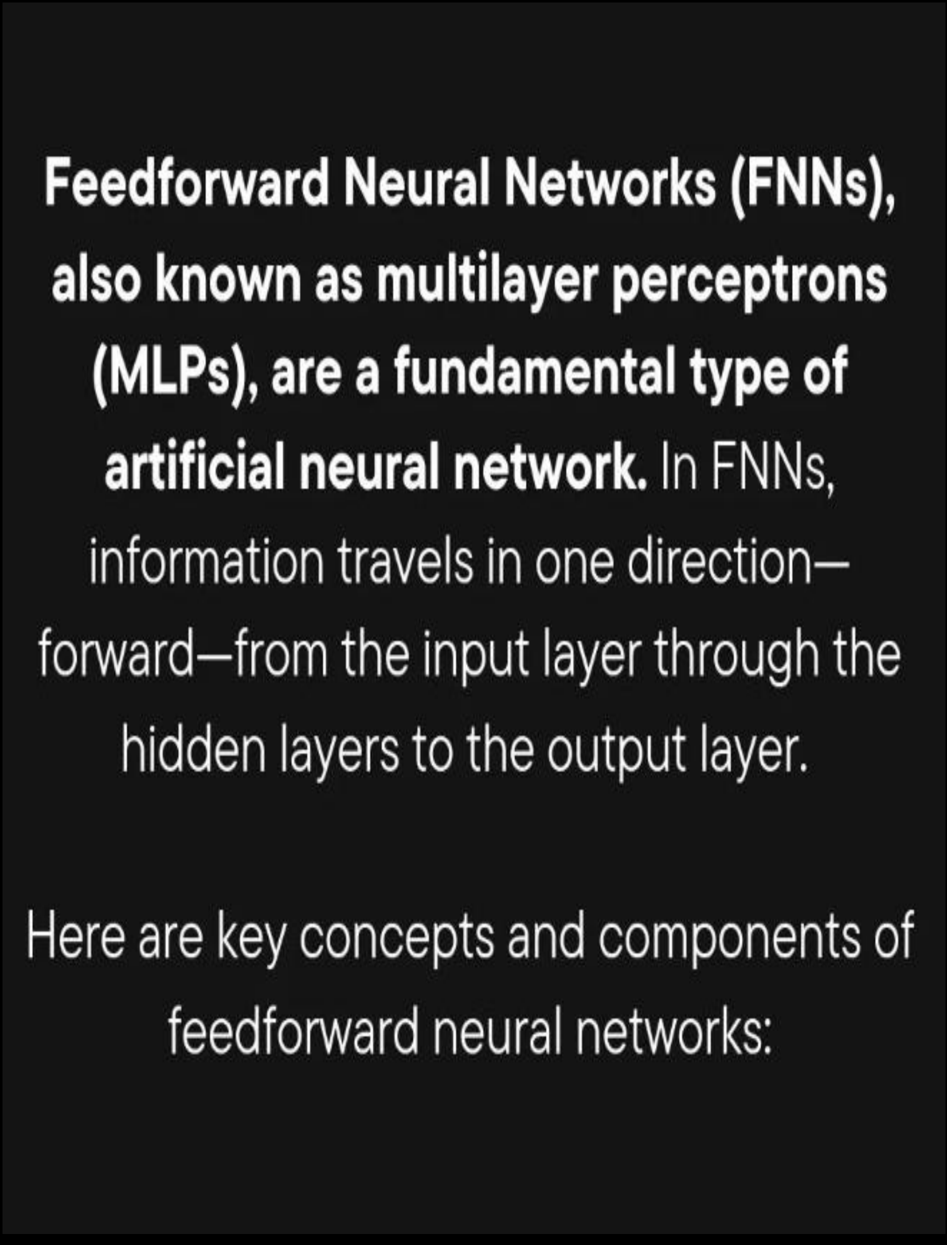
MACHINE LEARNING PART 28

Feedforward Neural Networks (FNNs)



Feedforward Neural Networks (FNNs), also known as multilayer perceptrons (MLPs), are a fundamental type of artificial neural network. In FNNs, information travels in one direction—forward—from the input layer through the hidden layers to the output layer.

Here are key concepts and components of feedforward neural networks:

Architecture

Input Layer:

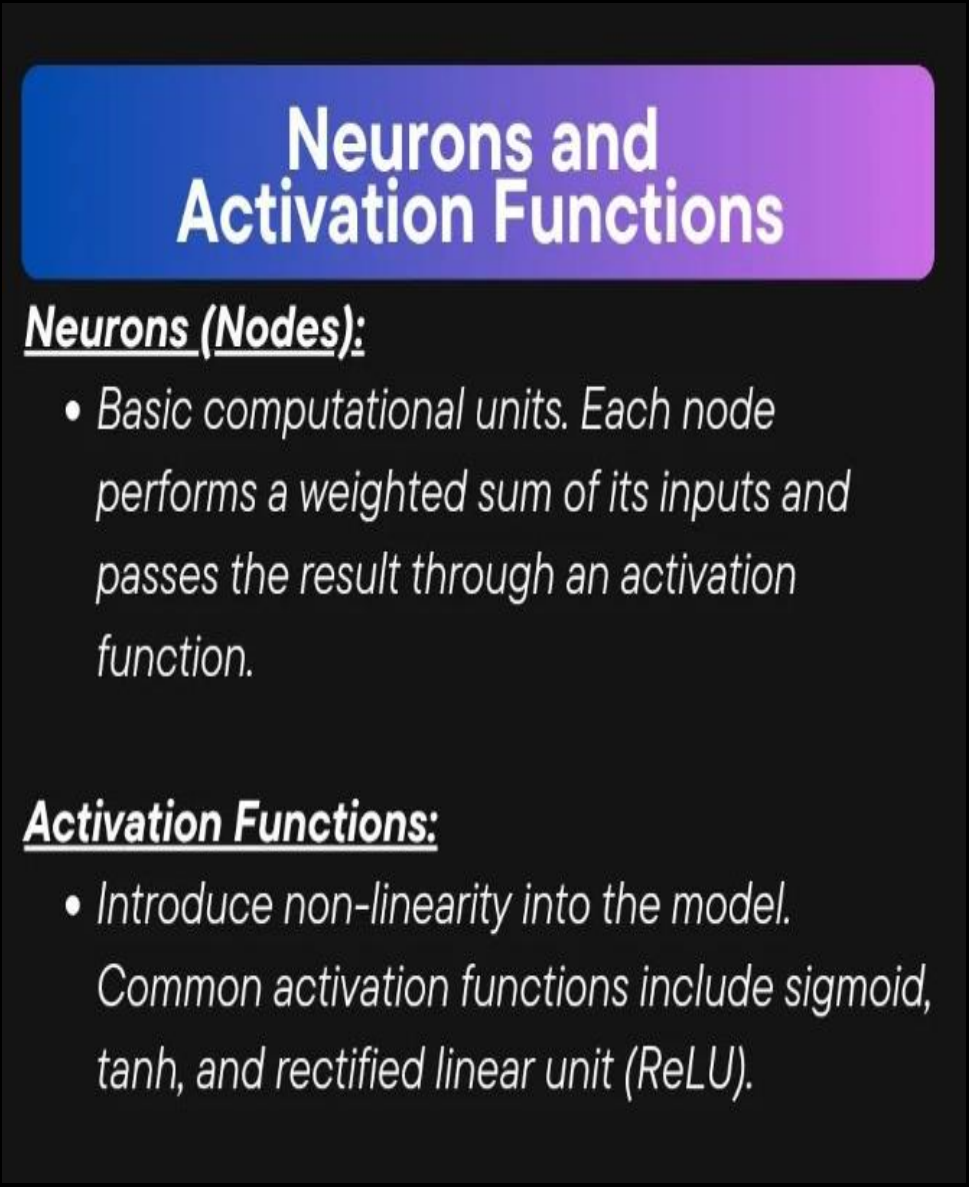
- *Receives input features. Each node represents a feature.*

Hidden Layers:

- *Layers between the input and output layers. Nodes in hidden layers apply weighted transformations to inputs using activation functions.*

Output Layer:

- *Produces the final output. The number of nodes in the output layer depends on the task (e.g., binary classification, multi-class classification, regression).*



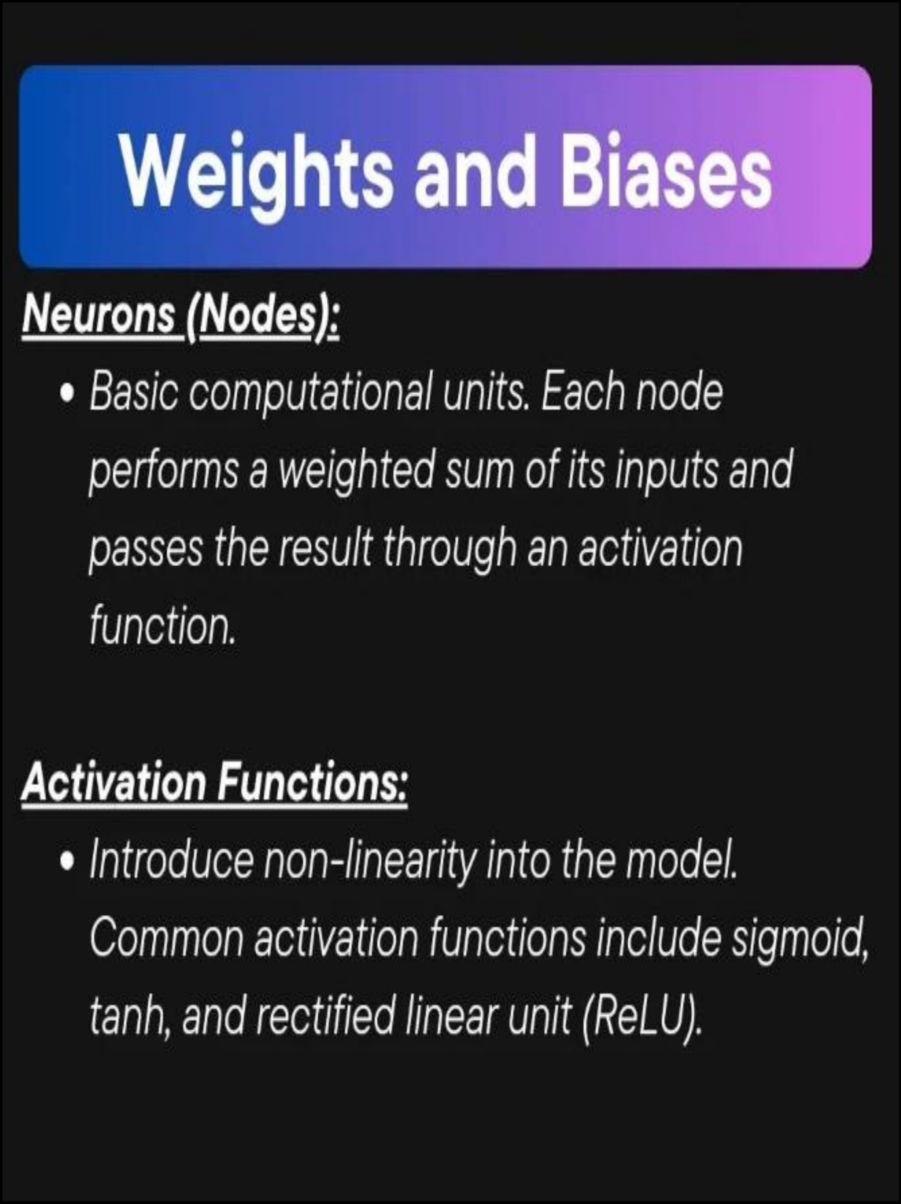
Neurons and Activation Functions

Neurons (Nodes):

- *Basic computational units. Each node performs a weighted sum of its inputs and passes the result through an activation function.*

Activation Functions:

- *Introduce non-linearity into the model. Common activation functions include sigmoid, tanh, and rectified linear unit (ReLU).*



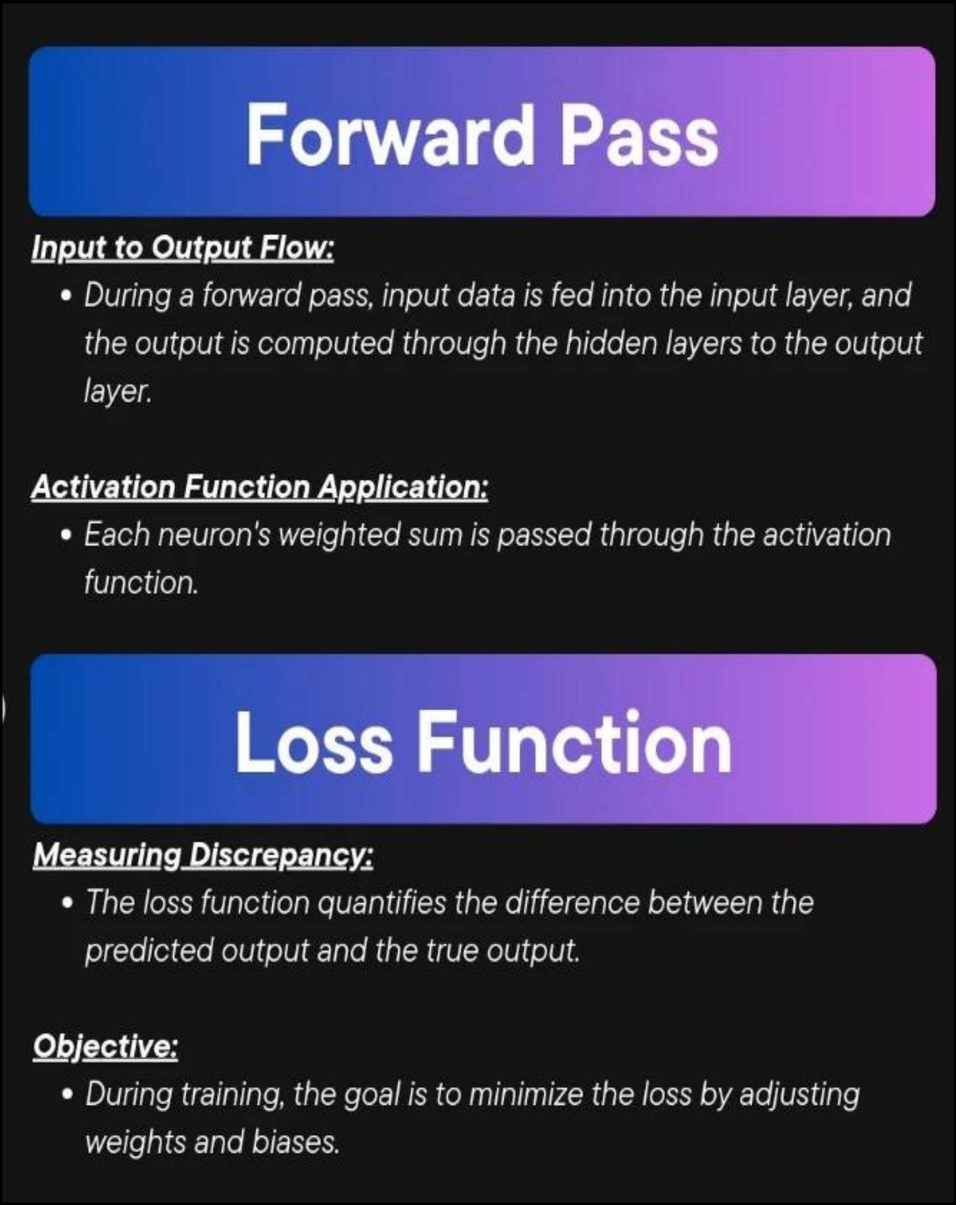
Weights and Biases

Neurons (Nodes):

- *Basic computational units. Each node performs a weighted sum of its inputs and passes the result through an activation function.*

Activation Functions:

- *Introduce non-linearity into the model. Common activation functions include sigmoid, tanh, and rectified linear unit (ReLU).*



Forward Pass

Input to Output Flow:

- *During a forward pass, input data is fed into the input layer, and the output is computed through the hidden layers to the output layer.*

Activation Function Application:

- *Each neuron's weighted sum is passed through the activation function.*

Loss Function

Measuring Discrepancy:

- *The loss function quantifies the difference between the predicted output and the true output.*

Objective:

- *During training, the goal is to minimize the loss by adjusting weights and biases.*

Training

Backpropagation:

- *The backpropagation algorithm is used to update weights and biases during training based on the calculated loss.*

Gradient Descent:

- *Optimization techniques like gradient descent are employed to find the minimum of the loss function.*

Overfitting and Regularization

Overfitting:

- *FNNs may memorize training data and perform poorly on new data. Regularization techniques, like dropout, help mitigate overfitting.*

Batch Training and Mini-Batch Training

Batch Training:

- *Updating weights based on the entire dataset.*

Mini-Batch Training:

- *Updating weights using smaller random subsets (mini-batches) of the dataset. Common in practice for efficiency.*

Applications

Image and Speech Recognition:

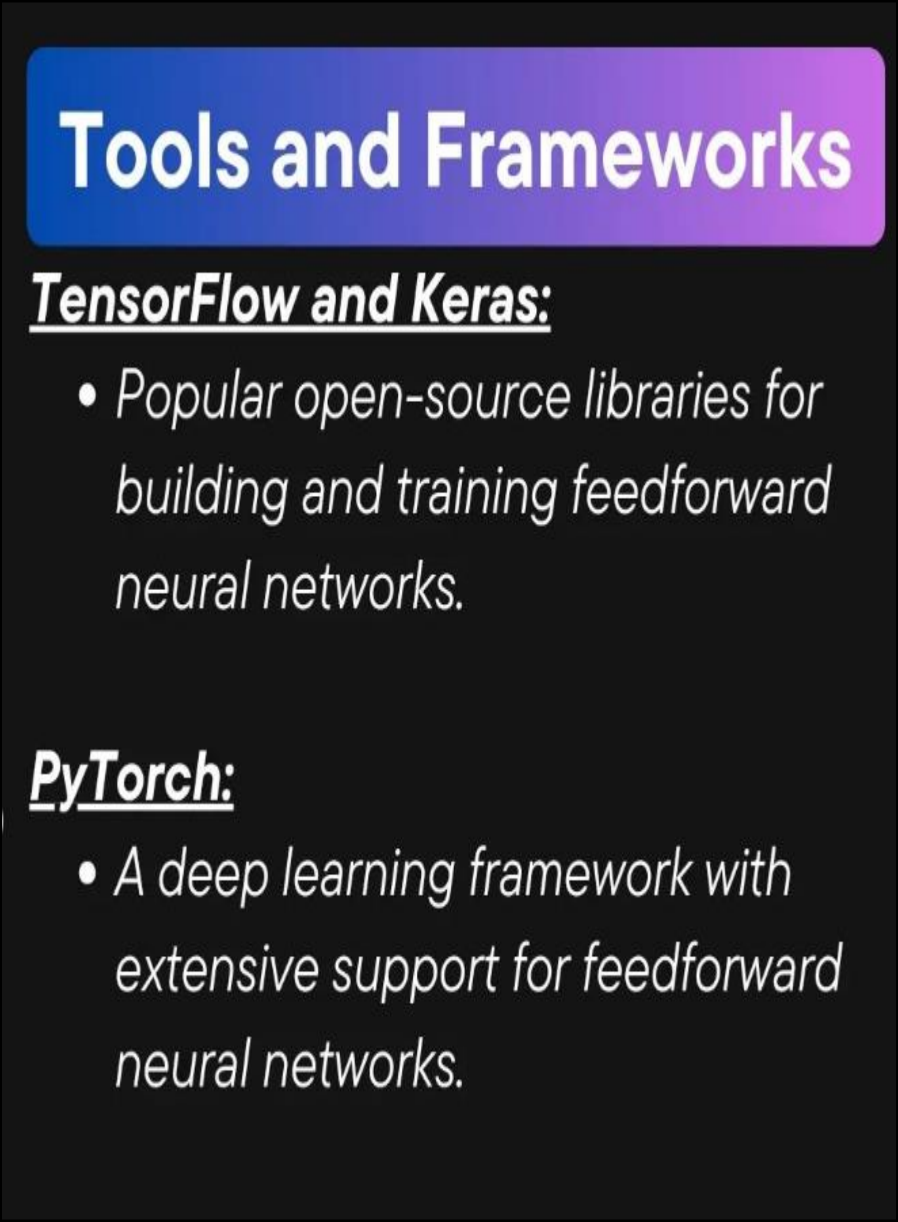
- *FNNs are used for tasks like image classification and speech recognition.*

Natural Language Processing:

- *Sentiment analysis, language translation.*

Regression Tasks:

- *Predicting numerical values.*



Tools and Frameworks

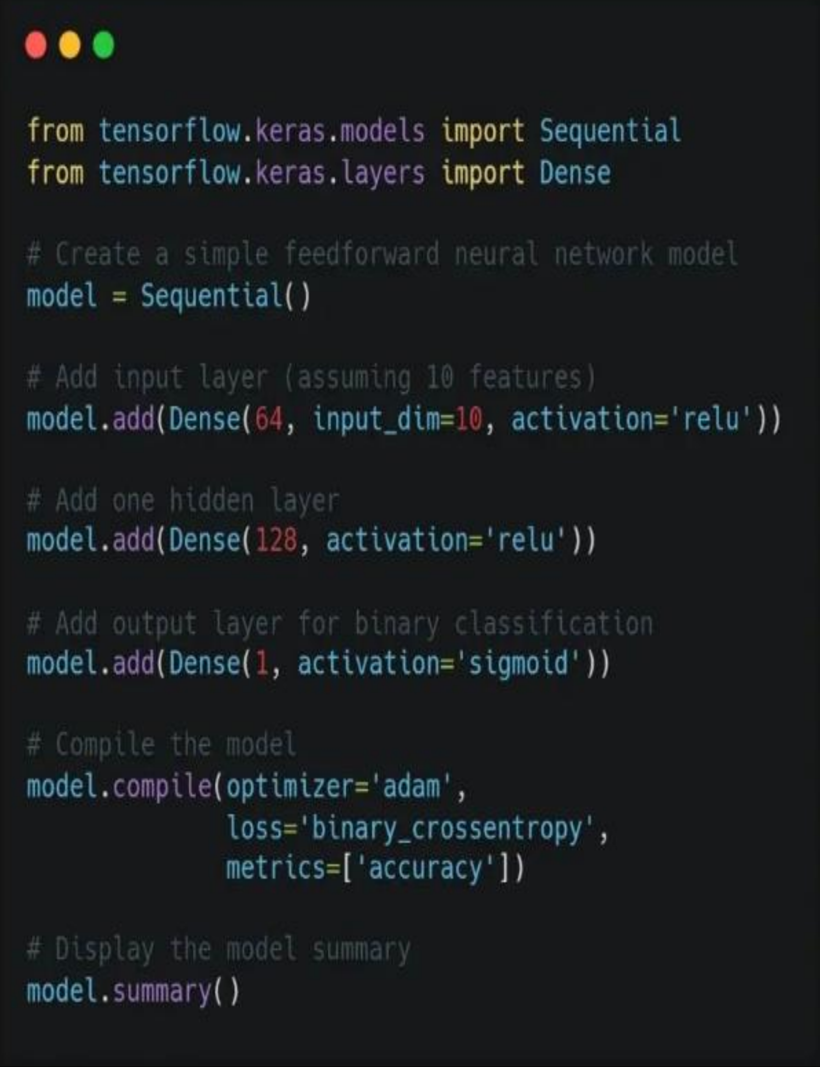
TensorFlow and Keras:

- *Popular open-source libraries for building and training feedforward neural networks.*

PyTorch:

- *A deep learning framework with extensive support for feedforward neural networks.*

Example Code



```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Create a simple feedforward neural network model
model = Sequential()

# Add input layer (assuming 10 features)
model.add(Dense(64, input_dim=10, activation='relu'))

# Add one hidden layer
model.add(Dense(128, activation='relu'))

# Add output layer for binary classification
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])

# Display the model summary
model.summary()
```