Feedforward Neural Network from Scratch

In [1]:
```
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, Input
from sklearn.datasets import make_classification
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
```

In [2]:
```
X, y = make_classification(
    n_samples=1000,
    n_features=20,
    n_informative=15,
    n_redundant=5,
    random_state=42
)
```

In [3]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [4]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [5]:
```
model = Sequential([
    Input(shape=(X_train.shape[1],)),
    Dense(64, activation='relu'), # Input layer
    Dense(32, activation='relu'),  # Hidden laye
    Dense(1, activation='sigmoid') # Output layer for binary classification
])
```

In [6]:
```
model.compile(
    optimizer='adam', # Optimizer
    loss='binary_crossentropy',
    metrics=['accuracy'] # Metrics to monitor during training
)
```

In [7]:
```
history = model.fit(X_train, y_train, # Training data and labels
                    epochs=20, # Number of epochs
                    batch_size=32, # Batch size
                    validation_split=0.2,
                    verbose=1)
```

```
Epoch 1/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.5218 - loss: 0.7088 - val_accuracy: 0.6812 - val_loss: 0.6157
Epoch 2/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.7610 - loss: 0.5682 - val_accuracy: 0.7812 - val_loss: 0.5122
Epoch 3/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.8366 - loss: 0.4574 - val_accuracy: 0.8625 - val_loss: 0.4320
Epoch 4/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.8545 - loss: 0.4035 - val_accuracy: 0.9000 - val_loss: 0.3680
Epoch 5/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9053 - loss: 0.3223 - val_accuracy: 0.9000 - val_loss: 0.3190
Epoch 6/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9086 - loss: 0.2830 - val_accuracy: 0.9187 - val_loss: 0.2823
Epoch 7/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9220 - loss: 0.2374 - val_accuracy: 0.9438 - val_loss: 0.2567
Epoch 8/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9267 - loss: 0.2172 - val_accuracy: 0.9375 - val_loss: 0.2376
Epoch 9/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9511 - loss: 0.1794 - val_accuracy: 0.9438 - val_loss: 0.2225
Epoch 10/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9452 - loss: 0.1702 - val_accuracy: 0.9500 - val_loss: 0.2105
Epoch 11/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9594 - loss: 0.1537 - val_accuracy: 0.9438 - val_loss: 0.2048
Epoch 12/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9576 - loss: 0.1363 - val_accuracy: 0.9438 - val_loss: 0.1891
Epoch 13/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9535 - loss: 0.1337 - val_accuracy: 0.9438 - val_loss: 0.1909
Epoch 14/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9710 - loss: 0.1193 - val_accuracy: 0.9312 - val_loss: 0.1836
Epoch 15/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9728 - loss: 0.1114 - val_accuracy: 0.9375 - val_loss: 0.1731
Epoch 16/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9766 - loss: 0.0986 - val_accuracy: 0.9312 - val_loss: 0.1711
Epoch 17/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9769 - loss: 0.0920 - val_accuracy: 0.9438 - val_loss: 0.1736
Epoch 18/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9756 - loss: 0.0854 - val_accuracy: 0.9375 - val_loss: 0.1681
Epoch 19/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 4ms/step - accuracy: 0.9793 - loss: 0.0855 - val_accuracy: 0.9375 - val_loss: 0.1687
Epoch 20/20
20/20 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - accuracy: 0.9790 - loss: 0.0877 - val_accuracy: 0.9500 - val_loss: 0.1649
```

In [8]:
```python
loss, accuracy = model.evaluate(X_test, y_test, verbose=0)
print(f"Test Loss: {loss:.4f}")
print(f"Test Accuracy: {accuracy * 100:.2f}")
```

```
Test Loss: 0.1524
Test Accuracy: 95.50
```

In [10]:
```python
import matplotlib.pyplot as plt
plt.figure(figsize=(12, 6))
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```