Task 1: Defining the Architecture of the Artificial Neural Network (ANN) Model

Objective

The main aim of this task is to create the architecture for predicting customer churn using Artificial Neural Network (ANN) model. That architecture consists of the input layer, hidden layers, output unit and activation functions that define how it sequences data to make predictions.

Steps Taken

1. Model Initialization

- **Input Layer:**
  - **Description:** The input layer (taking in the features of our customer data) These characteristics correspond to customer-relevant attributes, such as demographic data or the type and duration of service usage.
  - **Why we chose this configuration:**
    - The input shape is equal to the number of features in our dataset. This makes sure that all the features are treated and used to predict. We configure the input layer so that it has one neuron for each column in our dataset, meaning that this provides enough data to supply all of its neurons with inputs.
    - In this project the input_shape was defined as follows: input_shape=(X_train. shape[1],) where X_train. The second dimension, lowest_dimension=shape[1] (which is the total number of features) This ensures that the ANN takes into consideration all of these characteristics while predicting customer churn.

2. Designing the Hidden Layers

- **Hidden Layers:**
  - **Description:** The actual calculations and learning in the neural network is done by hidden layers, which take input data with weights and biases added to it, passes through an activation function.
  - **Layer Configuration:**
    - Two hidden layers were added to the model.
    - The first hidden layer consists of **64 neurons**, while the second hidden layer consists of **32 neurons**. Both layers use the **ReLU (Rectified Linear Unit)** activation function.
    - **Why we chose this configuration:**
      - **ReLU Activation:** ReLU is used in both hidden layers as it is computationally efficient and helps introduce non-linearity into the model, allowing it to capture complex patterns in the data.
      - **Layer size (64 and 32 neurons):** The number of neurons is a balance between complexity and computational efficiency. Too

few neurons may lead to underfitting, while too many neurons may result in overfitting. Based on experimentation, this configuration provided the best balance for our dataset.

3. Output Layer

- **Output Layer:**
  - **Description:** The output layer produces the final predictions. Since this is a binary classification problem — whether the customer churn or not hence our output layer has only 1 neuron.
  - **Activation Function:**
    - **Sigmoid Activation Function:** This function produces an output between 0 and 1, which is interpreted as the probability that a customer will churn.
    - **Why we chose this configuration:**
      - A **single neuron** was used in the output layer to represent the binary outcome (churn/no churn).
      - **Sigmoid activation** was chosen because it is apt for binary classification task as in our case this model predicts if a customer will churn or not based on the probability score and we can finally threshold at 0.5 to predict final class(calss=churn/no_churn).

4. Model Compilation

- **Optimizer:**
  - **Adam Optimizer:** Adam optimizer is a great choice for its handling of sparse gradients as well as adaptation of learning rate during training that allows faster convergence.
  - **Why we chose this configuration:**
    - The second most popular optimization algorithm behind SGD is Adam, which works well and has a simple implementation. With the kind of large datasets and complex models that we were dealing with, it would work great for our customer churn prediction model.
- **Loss Function:**
  - **Binary Cross-Entropy:** This loss function was used to measure the difference between the actual and predicted probabilities of churn.
  - **Why we chose this configuration:**
    - **Binary Cross-Entropy** — It is a loss function that computes difference between the actual and predicted probability of churn.
- **Metrics:**
  - **Accuracy:** This is the most used method of evaluating a model during training and monitoring its overall performance. Accuracy shows how many correct predictions model predicted.

Results Obtained

- The ANN architecture was successfully defined and compiled. This architecture included the input layer matching the dataset's features, two hidden layers with 64

and 32 neurons using ReLU activation, and an output layer with a single neuron using a sigmoid activation function for binary classification.
- The use of the Adam optimizer and binary cross-entropy loss function ensures efficient training and convergence.

Conclusion

While designing the ANN architecture, we were specifically focussed on how to do efficient customer churn prediction. Using hidden layers with ReLU activation and an output layer using a sigmoid function provides the model enough flexibility to learn complex patterns in customer data. Combined with Adam optimizer and binary_crossentropy loss, the model is highly trainable to recognize data patterns and also achieve less error in predicting.

With this architecture ready, model can be trained and evaluated against the customer churn dataset in the next steps.