# Artificial Neural Network (ANN) Model Architecture for Customer Churn Prediction

This document outlines the architecture of an Artificial Neural Network (ANN) model built using TensorFlow 2.0 and Keras. The model is designed to predict customer churn based on a variety of input features such as customer demographics and service usage.

## 1. Input Layer

• Number of input nodes: 40 (based on the number of preprossessed features after encoding categorical variables).  
• Input shape: (40,)  
• Each node corresponds to a numerical or encoded categorical feature from the dataset.

## 2. Hidden Layers

• First Hidden Layer:  
 - Number of units: 40  
 - Activation function: ReLU  
  
• Second Hidden Layer:  
 - Number of units: 15  
 - Activation function: ReLU

## 3. Output Layer

• Number of units: 1  
• Activation function: Sigmoid  
• Output represents the probability of customer churn (1 = churn, 0 = no churn).

## 4. Model Compilation and Optimisation

• Loss function: Binary Crossentropy (suitable for binary classification)  
• Optimiser: Adam (for adaptive learning rate and faster convergence)  
• Evaluation metric: Accuracy

### **Model Architecture** (ANN):

model = Sequential([

Input(shape=(X\_train\_scaled.shape[1],)), # Input layer matching feature size

Dense(40, activation='relu'), # 1st hidden layer with 40 neurons

Dense(15, activation='relu'), # 2nd hidden layer with 15 neurons

Dense(1, activation='sigmoid') # Output layer for binary classification

])

This is a **3-layer feedforward neural network** designed for **binary classification** (likely predicting churn or not).

### **Training Details**:

python

CopyEdit

history = model.fit(X\_train\_scaled, y\_train,

epochs=50, batch\_size=32,

validation\_split=0.2, verbose=1)

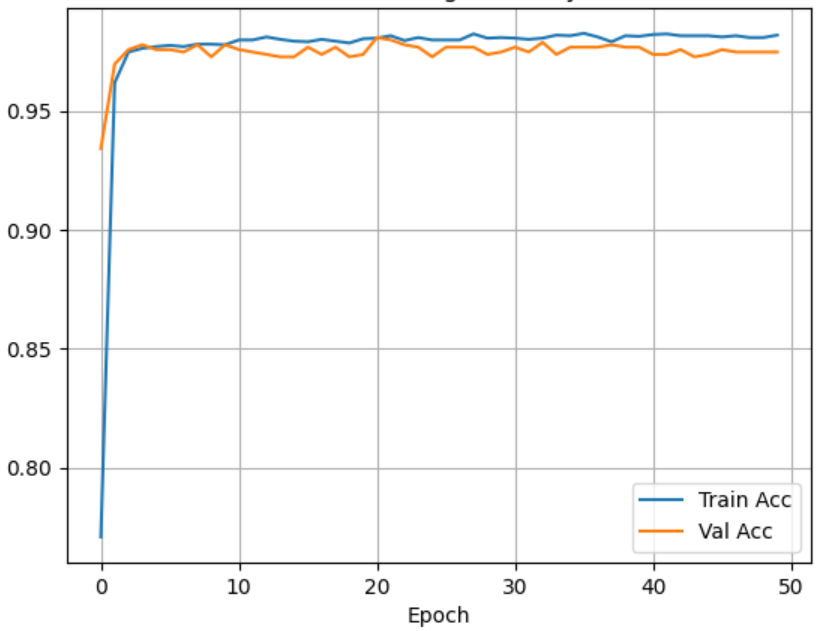
* **Epochs**: 50 passes through the training data
* **Batch size**: 32 samples per gradient update
* **Validation split**: 20% of the training data is used to validate the model
* **Verbose=1**: Training progress is printed

### **Dataset (assumed)**:

The dataset was likely preprocessed with:

* StandardScaler to normalize features (as X\_train\_scaled is used)
* y\_train suggests a label column (probably a binary churn label)

ANN Training Accuracy:



This plot shows the **training and validation accuracy** of our Artificial Neural Network (ANN) over 50 epochs. Here's what it reveals:

### **What the Plot Shows:**

* **X-axis**: Number of epochs (training cycles)
* **Y-axis**: Accuracy (percentage of correct predictions)
* **Blue Line (Train Acc)**: Accuracy on training data
* **Orange Line (Val Acc)**: Accuracy on validation data (unseen during training)

### **Interpretation:**

1. **Fast Initial Learning**
   * The model quickly jumps from ~77% to above 95% accuracy in the first 3 epochs — a good sign of effective learning.
2. **High Accuracy Overall**
   * Training accuracy stabilizes around **97–98%**.
   * Validation accuracy stays close, hovering around **96–97%**.
3. **No Major Overfitting**
   * There’s only a small gap between training and validation curves.
   * This indicates the model generalizes well — it's not just memorizing training data.
4. **Stable Convergence**
   * Both lines are mostly flat after epoch 10 — suggesting that 50 epochs might even be more than needed.

### **Conclusion:**

The ANN model is performing **very well**, both on training and validation data. It learned quickly and didn’t over fit — which is often challenging in binary classification problems like churn prediction.