# Day75_ANN_Implementation_ChurnPrediction

August 29, 2025

## 1 ANN Practical Implementation (Churn Prediction)

In the previous notebook, we learned about **Vanishing Gradient, Dropout, Optimizers, and Loss Functions**.

Today, we will **apply those concepts in practice** using an **Artificial Neural Network (ANN)** to predict customer churn.
Dataset: `Churn_Modelling.csv`

## 2 Importing Libraries

[1]:
```
# Artificial Neural Network

# Importing the libraries
import numpy as np
import pandas as pd
import tensorflow as tf

print("TensorFlow Version:", tf.__version__)
```

```
TensorFlow Version: 2.20.0-rc0
```

## 3 Data Preprocessing

We will:

1. Import dataset
2. Separate features (X) and target (y)
3. Encode categorical variables (Gender, Geography)
4. Perform Feature Scaling
5. Split into training & test sets

[4]:
```
# Import dataset
dataset = pd.read_csv(r'C:\Users\Lenovo\OneDrive\Desktop\Python Everyday␣
↪work\Class work\Deep_lerning\Day2\Churn_Modelling.csv')
X = dataset.iloc[:, 3:-1].values
y = dataset.iloc[:, -1].values
```

```python
print("X Sample:\n", X[:3])
print("y Sample:\n", y[:10])
```

```
X Sample:
 [[619 'France' 'Female' 42 2 0.0 1 1 1 101348.88]
 [608 'Spain' 'Female' 41 1 83807.86 1 0 1 112542.58]
 [502 'France' 'Female' 42 8 159660.8 3 1 0 113931.57]]
y Sample:
 [1 0 1 0 0 1 0 1 0 0]
```

### 3.1 Encoding Categorical Data

- **Label Encoding** for Gender
- **OneHot Encoding** for Geography

[5]:
```python
from sklearn.preprocessing import LabelEncoder
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

# Label Encoding Gender
le = LabelEncoder()
X[:, 2] = le.fit_transform(X[:, 2])

# One Hot Encoding Geography
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [1])],␣
 ↪remainder='passthrough')
X = np.array(ct.fit_transform(X))

print("After Encoding:\n", X[:3])
```

```
After Encoding:
 [[1.0 0.0 0.0 619 0 42 2 0.0 1 1 1 101348.88]
 [0.0 0.0 1.0 608 0 41 1 83807.86 1 0 1 112542.58]
 [1.0 0.0 0.0 502 0 42 8 159660.8 3 1 0 113931.57]]
```

### 3.2 Feature Scaling

ANNs converge faster with standardized inputs.

[6]:
```python
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
X = sc.fit_transform(X)

print("After Scaling:\n", X[:3])
```

```
After Scaling:
 [[ 0.99720391 -0.57873591 -0.57380915 -0.32622142 -1.09598752  0.29351742
  -1.04175968 -1.22584767 -0.91158349  0.64609167  0.97024255  0.02188649]
```

```
[-1.00280393 -0.57873591  1.74273971 -0.44003595 -1.09598752  0.19816383
 -1.38753759  0.11735002 -0.91158349 -1.54776799  0.97024255  0.21653375]
[ 0.99720391 -0.57873591 -0.57380915 -1.53679418 -1.09598752  0.29351742
  1.03290776  1.33305335  2.52705662  0.64609167 -1.03067011  0.2406869 ]]
```

### 3.3 Splitting Dataset

[7]:
```python
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,
 ↪random_state = 0)

print("Train Shape:", X_train.shape)
print("Test Shape:", X_test.shape)
```

```
Train Shape: (8000, 12)
Test Shape: (2000, 12)
```

## 4 Building the ANN

We will start simple, then **add more layers** and compare performance.

[8]:
```python
# Initializing the ANN
ann = tf.keras.models.Sequential()

# Input + First Hidden Layer
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

# Second Hidden Layer
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

# Output Layer
ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))
```

## 5 Training the ANN

- Optimizer: **Adam**
- Loss Function: **Binary Crossentropy** (since this is binary classification)
- Metric: **Accuracy**

[9]:
```python
# Compile ANN
ann.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics =
 ↪['accuracy'])

# Train ANN
history = ann.fit(X_train, y_train, batch_size = 32, epochs = 100, verbose=1)
```

```
Epoch 1/100
250/250            4s 4ms/step -
accuracy: 0.7159 - loss: 0.6086
Epoch 2/100
250/250            1s 3ms/step -
accuracy: 0.7966 - loss: 0.4713
Epoch 3/100
250/250            1s 3ms/step -
accuracy: 0.8035 - loss: 0.4367
Epoch 4/100
250/250            1s 3ms/step -
accuracy: 0.8116 - loss: 0.4275
Epoch 5/100
250/250            1s 3ms/step -
accuracy: 0.8146 - loss: 0.4242
Epoch 6/100
250/250            1s 3ms/step -
accuracy: 0.8177 - loss: 0.4214
Epoch 7/100
250/250            1s 3ms/step -
accuracy: 0.8191 - loss: 0.4190
Epoch 8/100
250/250            1s 3ms/step -
accuracy: 0.8210 - loss: 0.4165
Epoch 9/100
250/250            1s 3ms/step -
accuracy: 0.8223 - loss: 0.4137
Epoch 10/100
250/250            1s 3ms/step -
accuracy: 0.8231 - loss: 0.4099
Epoch 11/100
250/250            1s 3ms/step -
accuracy: 0.8279 - loss: 0.4047
Epoch 12/100
250/250            1s 3ms/step -
accuracy: 0.8353 - loss: 0.3932
Epoch 13/100
250/250            1s 3ms/step -
accuracy: 0.8447 - loss: 0.3775
Epoch 14/100
250/250            1s 3ms/step -
accuracy: 0.8504 - loss: 0.3658
Epoch 15/100
250/250            1s 3ms/step -
accuracy: 0.8534 - loss: 0.3593
Epoch 16/100
250/250            1s 3ms/step -
accuracy: 0.8555 - loss: 0.3550
```

```
Epoch 17/100
250/250             1s 4ms/step -
accuracy: 0.8556 - loss: 0.3523
Epoch 18/100
250/250             1s 4ms/step -
accuracy: 0.8579 - loss: 0.3505
Epoch 19/100
250/250             1s 4ms/step -
accuracy: 0.8581 - loss: 0.3494
Epoch 20/100
250/250             1s 4ms/step -
accuracy: 0.8577 - loss: 0.3480
Epoch 21/100
250/250             1s 4ms/step -
accuracy: 0.8583 - loss: 0.3470
Epoch 22/100
250/250             1s 4ms/step -
accuracy: 0.8587 - loss: 0.3463
Epoch 23/100
250/250             1s 4ms/step -
accuracy: 0.8581 - loss: 0.3454
Epoch 24/100
250/250             1s 3ms/step -
accuracy: 0.8593 - loss: 0.3452
Epoch 25/100
250/250             1s 4ms/step -
accuracy: 0.8581 - loss: 0.3449
Epoch 26/100
250/250             1s 4ms/step -
accuracy: 0.8587 - loss: 0.3443
Epoch 27/100
250/250             1s 4ms/step -
accuracy: 0.8595 - loss: 0.3437
Epoch 28/100
250/250             1s 4ms/step -
accuracy: 0.8604 - loss: 0.3433
Epoch 29/100
250/250             1s 4ms/step -
accuracy: 0.8593 - loss: 0.3429
Epoch 30/100
250/250             1s 4ms/step -
accuracy: 0.8597 - loss: 0.3422
Epoch 31/100
250/250             1s 4ms/step -
accuracy: 0.8599 - loss: 0.3421
Epoch 32/100
250/250             1s 4ms/step -
accuracy: 0.8595 - loss: 0.3420
```

```
Epoch 33/100
250/250             1s 3ms/step -
accuracy: 0.8585 - loss: 0.3415
Epoch 34/100
250/250             1s 4ms/step -
accuracy: 0.8596 - loss: 0.3412
Epoch 35/100
250/250             1s 4ms/step -
accuracy: 0.8585 - loss: 0.3409
Epoch 36/100
250/250             1s 4ms/step -
accuracy: 0.8590 - loss: 0.3408
Epoch 37/100
250/250             1s 3ms/step -
accuracy: 0.8605 - loss: 0.3404
Epoch 38/100
250/250             1s 4ms/step -
accuracy: 0.8601 - loss: 0.3403
Epoch 39/100
250/250             1s 4ms/step -
accuracy: 0.8604 - loss: 0.3401
Epoch 40/100
250/250             1s 4ms/step -
accuracy: 0.8609 - loss: 0.3403
Epoch 41/100
250/250             1s 4ms/step -
accuracy: 0.8591 - loss: 0.3392
Epoch 42/100
250/250             1s 3ms/step -
accuracy: 0.8605 - loss: 0.3391
Epoch 43/100
250/250             1s 3ms/step -
accuracy: 0.8600 - loss: 0.3388
Epoch 44/100
250/250             2s 4ms/step -
accuracy: 0.8610 - loss: 0.3390
Epoch 45/100
250/250             1s 4ms/step -
accuracy: 0.8625 - loss: 0.3387
Epoch 46/100
250/250             1s 3ms/step -
accuracy: 0.8610 - loss: 0.3381
Epoch 47/100
250/250             1s 3ms/step -
accuracy: 0.8618 - loss: 0.3382
Epoch 48/100
250/250             1s 3ms/step -
accuracy: 0.8626 - loss: 0.3379
```

```
Epoch 49/100
250/250              1s 3ms/step -
accuracy: 0.8602 - loss: 0.3379
Epoch 50/100
250/250              1s 3ms/step -
accuracy: 0.8606 - loss: 0.3373
Epoch 51/100
250/250              1s 3ms/step -
accuracy: 0.8625 - loss: 0.3368
Epoch 52/100
250/250              1s 3ms/step -
accuracy: 0.8616 - loss: 0.3372
Epoch 53/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3371
Epoch 54/100
250/250              1s 3ms/step -
accuracy: 0.8633 - loss: 0.3369
Epoch 55/100
250/250              1s 3ms/step -
accuracy: 0.8624 - loss: 0.3367
Epoch 56/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3363
Epoch 57/100
250/250              1s 3ms/step -
accuracy: 0.8622 - loss: 0.3368
Epoch 58/100
250/250              1s 3ms/step -
accuracy: 0.8614 - loss: 0.3364
Epoch 59/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3357
Epoch 60/100
250/250              1s 3ms/step -
accuracy: 0.8619 - loss: 0.3359
Epoch 61/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3354
Epoch 62/100
250/250              1s 3ms/step -
accuracy: 0.8641 - loss: 0.3356
Epoch 63/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3356
Epoch 64/100
250/250              1s 3ms/step -
accuracy: 0.8634 - loss: 0.3355
```

```
Epoch 65/100
250/250            1s 3ms/step -
accuracy: 0.8635 - loss: 0.3354
Epoch 66/100
250/250            1s 3ms/step -
accuracy: 0.8639 - loss: 0.3348
Epoch 67/100
250/250            1s 4ms/step -
accuracy: 0.8630 - loss: 0.3351
Epoch 68/100
250/250            1s 3ms/step -
accuracy: 0.8630 - loss: 0.3350
Epoch 69/100
250/250            1s 3ms/step -
accuracy: 0.8627 - loss: 0.3347
Epoch 70/100
250/250            1s 3ms/step -
accuracy: 0.8618 - loss: 0.3347
Epoch 71/100
250/250            1s 3ms/step -
accuracy: 0.8633 - loss: 0.3345
Epoch 72/100
250/250            1s 3ms/step -
accuracy: 0.8644 - loss: 0.3345
Epoch 73/100
250/250            1s 3ms/step -
accuracy: 0.8626 - loss: 0.3343
Epoch 74/100
250/250            1s 3ms/step -
accuracy: 0.8641 - loss: 0.3342
Epoch 75/100
250/250            1s 3ms/step -
accuracy: 0.8627 - loss: 0.3341
Epoch 76/100
250/250            1s 3ms/step -
accuracy: 0.8620 - loss: 0.3341
Epoch 77/100
250/250            1s 3ms/step -
accuracy: 0.8640 - loss: 0.3343
Epoch 78/100
250/250            1s 3ms/step -
accuracy: 0.8635 - loss: 0.3339
Epoch 79/100
250/250            1s 3ms/step -
accuracy: 0.8625 - loss: 0.3341
Epoch 80/100
250/250            1s 3ms/step -
accuracy: 0.8641 - loss: 0.3341
```

```
Epoch 81/100
250/250              1s 3ms/step -
accuracy: 0.8649 - loss: 0.3337
Epoch 82/100
250/250              1s 4ms/step -
accuracy: 0.8621 - loss: 0.3337
Epoch 83/100
250/250              1s 3ms/step -
accuracy: 0.8644 - loss: 0.3341
Epoch 84/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3337
Epoch 85/100
250/250              1s 4ms/step -
accuracy: 0.8641 - loss: 0.3336
Epoch 86/100
250/250              1s 3ms/step -
accuracy: 0.8643 - loss: 0.3335
Epoch 87/100
250/250              1s 3ms/step -
accuracy: 0.8643 - loss: 0.3337
Epoch 88/100
250/250              1s 3ms/step -
accuracy: 0.8648 - loss: 0.3332
Epoch 89/100
250/250              1s 3ms/step -
accuracy: 0.8649 - loss: 0.3334
Epoch 90/100
250/250              1s 3ms/step -
accuracy: 0.8639 - loss: 0.3333
Epoch 91/100
250/250              1s 3ms/step -
accuracy: 0.8639 - loss: 0.3331
Epoch 92/100
250/250              1s 3ms/step -
accuracy: 0.8644 - loss: 0.3336
Epoch 93/100
250/250              1s 3ms/step -
accuracy: 0.8648 - loss: 0.3335
Epoch 94/100
250/250              1s 3ms/step -
accuracy: 0.8649 - loss: 0.3330
Epoch 95/100
250/250              1s 3ms/step -
accuracy: 0.8655 - loss: 0.3328
Epoch 96/100
250/250              1s 4ms/step -
accuracy: 0.8643 - loss: 0.3330
```

```
Epoch 97/100
250/250              1s 4ms/step -
accuracy: 0.8651 - loss: 0.3330
Epoch 98/100
250/250              1s 3ms/step -
accuracy: 0.8639 - loss: 0.3327
Epoch 99/100
250/250              1s 3ms/step -
accuracy: 0.8658 - loss: 0.3328
Epoch 100/100
250/250              1s 3ms/step -
accuracy: 0.8654 - loss: 0.3328
```

## 6 Model Evaluation

We will predict on the **Test set** and evaluate using:

- Predictions
- Confusion Matrix
- Accuracy

[10]:
```
# Predicting the Test set results
y_pred = ann.predict(X_test)
y_pred = (y_pred > 0.5)

# Compare predictions vs actual
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.
 ↪reshape(len(y_test),1)),1)[:10])
```

```
63/63              0s 3ms/step
[[0 0]
 [0 1]
 [0 0]
 [0 0]
 [0 0]
 [1 1]
 [0 0]
 [0 0]
 [0 1]
 [1 1]]
```

[11]:
```
# Confusion Matrix
from sklearn.metrics import confusion_matrix, accuracy_score

cm = confusion_matrix(y_test, y_pred)
acc = accuracy_score(y_test, y_pred)

print("Confusion Matrix:\n", cm)
```

```python
print("Accuracy:", acc)
```

```
Confusion Matrix:
 [[1492  103]
 [ 186  219]]
Accuracy: 0.8555
```

# 7 Experimenting with More Layers

We now add more 2 hidden layers and compare performance.

[12]:
```python
# Build deeper ANN
ann_deep = tf.keras.models.Sequential()

# Input + 3 Hidden Layers
ann_deep.add(tf.keras.layers.Dense(units=8, activation='relu'))
ann_deep.add(tf.keras.layers.Dense(units=8, activation='relu'))
ann_deep.add(tf.keras.layers.Dense(units=8, activation='relu'))

# Output Layer
ann_deep.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

# Compile
ann_deep.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics =␣
 ↪['accuracy'])

# Train
history_deep = ann_deep.fit(X_train, y_train, batch_size = 32, epochs = 100,␣
 ↪verbose=1)
```

```
Epoch 1/100
250/250              4s 3ms/step -
accuracy: 0.6799 - loss: 0.6088
Epoch 2/100
250/250              1s 3ms/step -
accuracy: 0.7980 - loss: 0.4511
Epoch 3/100
250/250              1s 3ms/step -
accuracy: 0.8023 - loss: 0.4330
Epoch 4/100
250/250              1s 4ms/step -
accuracy: 0.8050 - loss: 0.4224
Epoch 5/100
250/250              1s 4ms/step -
accuracy: 0.8201 - loss: 0.4063
Epoch 6/100
250/250              1s 4ms/step -
accuracy: 0.8381 - loss: 0.3834
```

```
Epoch 7/100
250/250              1s 4ms/step -
accuracy: 0.8482 - loss: 0.3664
Epoch 8/100
250/250              1s 4ms/step -
accuracy: 0.8503 - loss: 0.3574
Epoch 9/100
250/250              1s 3ms/step -
accuracy: 0.8514 - loss: 0.3540
Epoch 10/100
250/250              1s 3ms/step -
accuracy: 0.8541 - loss: 0.3510
Epoch 11/100
250/250              1s 4ms/step -
accuracy: 0.8556 - loss: 0.3488
Epoch 12/100
250/250              1s 3ms/step -
accuracy: 0.8553 - loss: 0.3470
Epoch 13/100
250/250              1s 4ms/step -
accuracy: 0.8580 - loss: 0.3454
Epoch 14/100
250/250              1s 4ms/step -
accuracy: 0.8599 - loss: 0.3435
Epoch 15/100
250/250              1s 4ms/step -
accuracy: 0.8591 - loss: 0.3422
Epoch 16/100
250/250              1s 4ms/step -
accuracy: 0.8612 - loss: 0.3411
Epoch 17/100
250/250              1s 3ms/step -
accuracy: 0.8619 - loss: 0.3401
Epoch 18/100
250/250              1s 4ms/step -
accuracy: 0.8620 - loss: 0.3394
Epoch 19/100
250/250              1s 4ms/step -
accuracy: 0.8631 - loss: 0.3392
Epoch 20/100
250/250              1s 4ms/step -
accuracy: 0.8627 - loss: 0.3389
Epoch 21/100
250/250              1s 4ms/step -
accuracy: 0.8633 - loss: 0.3383
Epoch 22/100
250/250              1s 3ms/step -
accuracy: 0.8635 - loss: 0.3374
```

```
Epoch 23/100
250/250              1s 4ms/step -
accuracy: 0.8631 - loss: 0.3376
Epoch 24/100
250/250              1s 3ms/step -
accuracy: 0.8622 - loss: 0.3374
Epoch 25/100
250/250              1s 3ms/step -
accuracy: 0.8636 - loss: 0.3374
Epoch 26/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3361
Epoch 27/100
250/250              1s 4ms/step -
accuracy: 0.8633 - loss: 0.3364
Epoch 28/100
250/250              1s 4ms/step -
accuracy: 0.8626 - loss: 0.3362
Epoch 29/100
250/250              1s 4ms/step -
accuracy: 0.8633 - loss: 0.3359
Epoch 30/100
250/250              1s 3ms/step -
accuracy: 0.8618 - loss: 0.3358
Epoch 31/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3352
Epoch 32/100
250/250              1s 3ms/step -
accuracy: 0.8635 - loss: 0.3352
Epoch 33/100
250/250              1s 4ms/step -
accuracy: 0.8630 - loss: 0.3347
Epoch 34/100
250/250              1s 4ms/step -
accuracy: 0.8627 - loss: 0.3344
Epoch 35/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3345
Epoch 36/100
250/250              1s 4ms/step -
accuracy: 0.8629 - loss: 0.3347
Epoch 37/100
250/250              1s 3ms/step -
accuracy: 0.8635 - loss: 0.3345
Epoch 38/100
250/250              1s 4ms/step -
accuracy: 0.8643 - loss: 0.3338
```

```
Epoch 39/100
250/250              1s 4ms/step -
accuracy: 0.8627 - loss: 0.3338
Epoch 40/100
250/250              1s 3ms/step -
accuracy: 0.8649 - loss: 0.3335
Epoch 41/100
250/250              1s 3ms/step -
accuracy: 0.8636 - loss: 0.3335
Epoch 42/100
250/250              1s 4ms/step -
accuracy: 0.8625 - loss: 0.3332
Epoch 43/100
250/250              1s 3ms/step -
accuracy: 0.8622 - loss: 0.3333
Epoch 44/100
250/250              1s 4ms/step -
accuracy: 0.8627 - loss: 0.3330
Epoch 45/100
250/250              1s 3ms/step -
accuracy: 0.8648 - loss: 0.3327
Epoch 46/100
250/250              1s 3ms/step -
accuracy: 0.8630 - loss: 0.3322
Epoch 47/100
250/250              1s 4ms/step -
accuracy: 0.8629 - loss: 0.3327
Epoch 48/100
250/250              1s 3ms/step -
accuracy: 0.8631 - loss: 0.3319
Epoch 49/100
250/250              1s 4ms/step -
accuracy: 0.8627 - loss: 0.3323
Epoch 50/100
250/250              1s 3ms/step -
accuracy: 0.8649 - loss: 0.3321
Epoch 51/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3317
Epoch 52/100
250/250              1s 3ms/step -
accuracy: 0.8640 - loss: 0.3323
Epoch 53/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3317
Epoch 54/100
250/250              1s 4ms/step -
accuracy: 0.8640 - loss: 0.3320
```

```
Epoch 55/100
250/250             1s 3ms/step -
accuracy: 0.8646 - loss: 0.3319
Epoch 56/100
250/250             1s 4ms/step -
accuracy: 0.8634 - loss: 0.3314
Epoch 57/100
250/250             1s 3ms/step -
accuracy: 0.8644 - loss: 0.3322
Epoch 58/100
250/250             1s 4ms/step -
accuracy: 0.8648 - loss: 0.3312
Epoch 59/100
250/250             1s 3ms/step -
accuracy: 0.8644 - loss: 0.3314
Epoch 60/100
250/250             1s 4ms/step -
accuracy: 0.8654 - loss: 0.3316
Epoch 61/100
250/250             1s 3ms/step -
accuracy: 0.8643 - loss: 0.3311
Epoch 62/100
250/250             1s 3ms/step -
accuracy: 0.8636 - loss: 0.3314
Epoch 63/100
250/250             1s 4ms/step -
accuracy: 0.8643 - loss: 0.3309
Epoch 64/100
250/250             1s 3ms/step -
accuracy: 0.8634 - loss: 0.3309
Epoch 65/100
250/250             2s 4ms/step -
accuracy: 0.8643 - loss: 0.3312
Epoch 66/100
250/250             1s 4ms/step -
accuracy: 0.8640 - loss: 0.3310
Epoch 67/100
250/250             1s 3ms/step -
accuracy: 0.8643 - loss: 0.3310
Epoch 68/100
250/250             1s 4ms/step -
accuracy: 0.8669 - loss: 0.3310
Epoch 69/100
250/250             1s 3ms/step -
accuracy: 0.8649 - loss: 0.3311
Epoch 70/100
250/250             1s 3ms/step -
accuracy: 0.8639 - loss: 0.3307
```

```
Epoch 71/100
250/250              1s 4ms/step -
accuracy: 0.8644 - loss: 0.3306
Epoch 72/100
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3307
Epoch 73/100
250/250              1s 3ms/step -
accuracy: 0.8643 - loss: 0.3307
Epoch 74/100
250/250              1s 3ms/step -
accuracy: 0.8655 - loss: 0.3303
Epoch 75/100
250/250              1s 3ms/step -
accuracy: 0.8662 - loss: 0.3310
Epoch 76/100
250/250              1s 3ms/step -
accuracy: 0.8636 - loss: 0.3305
Epoch 77/100
250/250              1s 3ms/step -
accuracy: 0.8665 - loss: 0.3303
Epoch 78/100
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3305
Epoch 79/100
250/250              1s 4ms/step -
accuracy: 0.8669 - loss: 0.3301
Epoch 80/100
250/250              1s 3ms/step -
accuracy: 0.8658 - loss: 0.3297
Epoch 81/100
250/250              1s 3ms/step -
accuracy: 0.8644 - loss: 0.3303
Epoch 82/100
250/250              1s 3ms/step -
accuracy: 0.8640 - loss: 0.3295
Epoch 83/100
250/250              1s 3ms/step -
accuracy: 0.8656 - loss: 0.3298
Epoch 84/100
250/250              1s 3ms/step -
accuracy: 0.8662 - loss: 0.3298
Epoch 85/100
250/250              1s 3ms/step -
accuracy: 0.8662 - loss: 0.3297
Epoch 86/100
250/250              1s 3ms/step -
accuracy: 0.8664 - loss: 0.3304
```

```
Epoch 87/100
250/250            1s 3ms/step -
accuracy: 0.8651 - loss: 0.3302
Epoch 88/100
250/250            1s 4ms/step -
accuracy: 0.8645 - loss: 0.3301
Epoch 89/100
250/250            1s 3ms/step -
accuracy: 0.8661 - loss: 0.3293
Epoch 90/100
250/250            1s 3ms/step -
accuracy: 0.8656 - loss: 0.3299
Epoch 91/100
250/250            1s 4ms/step -
accuracy: 0.8650 - loss: 0.3299
Epoch 92/100
250/250            2s 5ms/step -
accuracy: 0.8660 - loss: 0.3300
Epoch 93/100
250/250            1s 4ms/step -
accuracy: 0.8662 - loss: 0.3292
Epoch 94/100
250/250            1s 4ms/step -
accuracy: 0.8652 - loss: 0.3295
Epoch 95/100
250/250            1s 4ms/step -
accuracy: 0.8660 - loss: 0.3292
Epoch 96/100
250/250            2s 5ms/step -
accuracy: 0.8658 - loss: 0.3296
Epoch 97/100
250/250            1s 4ms/step -
accuracy: 0.8660 - loss: 0.3292
Epoch 98/100
250/250            1s 4ms/step -
accuracy: 0.8655 - loss: 0.3292
Epoch 99/100
250/250            1s 4ms/step -
accuracy: 0.8650 - loss: 0.3286
Epoch 100/100
250/250            1s 4ms/step -
accuracy: 0.8655 - loss: 0.3293
```

[13]:

```
# Evaluate deeper model
y_pred_deep = ann_deep.predict(X_test)
y_pred_deep = (y_pred_deep > 0.5)
```

```
cm_deep = confusion_matrix(y_test, y_pred_deep)
acc_deep = accuracy_score(y_test, y_pred_deep)

print("Confusion Matrix (Deeper ANN):\n", cm_deep)
print("Accuracy (Deeper ANN):", acc_deep)
```

```
63/63              0s 4ms/step
Confusion Matrix (Deeper ANN):
 [[1525   70]
 [ 198  207]]
Accuracy (Deeper ANN): 0.866
```

We now add 3 hidden layers and compare performance.

[15]:
```
# Initializing the ANN
ann = tf.keras.models.Sequential()

# Input + Hidden Layers
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))
ann.add(tf.keras.layers.Dense(units=6, activation='relu'))

# Output Layer
ann.add(tf.keras.layers.Dense(units=1, activation='sigmoid'))

# Compile and Train
ann.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
ann.fit(X_train, y_train, batch_size=32, epochs=100)

# Evaluate
y_pred = (ann.predict(X_test) > 0.5)
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
acc = accuracy_score(y_test, y_pred)

print("Confusion Matrix (3 Hidden Layers):")
print(cm)
print("Accuracy (3 Hidden Layers):", acc)
```

```
Epoch 1/100
250/250              4s 3ms/step -
accuracy: 0.7691 - loss: 0.5501
Epoch 2/100
250/250              1s 4ms/step -
accuracy: 0.7962 - loss: 0.4785
Epoch 3/100
250/250              1s 3ms/step -
accuracy: 0.7989 - loss: 0.4537
Epoch 4/100
```

```
250/250            1s 4ms/step -
accuracy: 0.8030 - loss: 0.4381
Epoch 5/100
250/250            1s 3ms/step -
accuracy: 0.8096 - loss: 0.4277
Epoch 6/100
250/250            1s 4ms/step -
accuracy: 0.8161 - loss: 0.4196
Epoch 7/100
250/250            1s 3ms/step -
accuracy: 0.8209 - loss: 0.4125
Epoch 8/100
250/250            1s 3ms/step -
accuracy: 0.8265 - loss: 0.4064
Epoch 9/100
250/250            1s 3ms/step -
accuracy: 0.8285 - loss: 0.4013
Epoch 10/100
250/250            1s 4ms/step -
accuracy: 0.8320 - loss: 0.3971
Epoch 11/100
250/250            1s 3ms/step -
accuracy: 0.8340 - loss: 0.3942
Epoch 12/100
250/250            1s 4ms/step -
accuracy: 0.8335 - loss: 0.3914
Epoch 13/100
250/250            1s 3ms/step -
accuracy: 0.8351 - loss: 0.3890
Epoch 14/100
250/250            1s 4ms/step -
accuracy: 0.8390 - loss: 0.3864
Epoch 15/100
250/250            1s 4ms/step -
accuracy: 0.8415 - loss: 0.3836
Epoch 16/100
250/250            1s 4ms/step -
accuracy: 0.8434 - loss: 0.3816
Epoch 17/100
250/250            1s 3ms/step -
accuracy: 0.8447 - loss: 0.3794
Epoch 18/100
250/250            1s 3ms/step -
accuracy: 0.8470 - loss: 0.3762
Epoch 19/100
250/250            1s 4ms/step -
accuracy: 0.8490 - loss: 0.3729
Epoch 20/100
```

```
250/250              1s 3ms/step -
accuracy: 0.8494 - loss: 0.3699
Epoch 21/100
250/250              1s 3ms/step -
accuracy: 0.8516 - loss: 0.3670
Epoch 22/100
250/250              1s 3ms/step -
accuracy: 0.8529 - loss: 0.3646
Epoch 23/100
250/250              1s 3ms/step -
accuracy: 0.8551 - loss: 0.3621
Epoch 24/100
250/250              1s 3ms/step -
accuracy: 0.8558 - loss: 0.3601
Epoch 25/100
250/250              1s 3ms/step -
accuracy: 0.8530 - loss: 0.3580
Epoch 26/100
250/250              1s 4ms/step -
accuracy: 0.8560 - loss: 0.3561
Epoch 27/100
250/250              1s 3ms/step -
accuracy: 0.8561 - loss: 0.3544
Epoch 28/100
250/250              1s 4ms/step -
accuracy: 0.8599 - loss: 0.3527
Epoch 29/100
250/250              1s 3ms/step -
accuracy: 0.8609 - loss: 0.3508
Epoch 30/100
250/250              1s 3ms/step -
accuracy: 0.8610 - loss: 0.3501
Epoch 31/100
250/250              1s 3ms/step -
accuracy: 0.8619 - loss: 0.3483
Epoch 32/100
250/250              1s 3ms/step -
accuracy: 0.8620 - loss: 0.3469
Epoch 33/100
250/250              1s 3ms/step -
accuracy: 0.8614 - loss: 0.3457
Epoch 34/100
250/250              1s 3ms/step -
accuracy: 0.8606 - loss: 0.3454
Epoch 35/100
250/250              1s 3ms/step -
accuracy: 0.8625 - loss: 0.3448
Epoch 36/100
```

```
250/250              1s 3ms/step -
accuracy: 0.8622 - loss: 0.3437
Epoch 37/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3426
Epoch 38/100
250/250              1s 3ms/step -
accuracy: 0.8634 - loss: 0.3418
Epoch 39/100
250/250              1s 3ms/step -
accuracy: 0.8612 - loss: 0.3416
Epoch 40/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3409
Epoch 41/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3405
Epoch 42/100
250/250              1s 3ms/step -
accuracy: 0.8645 - loss: 0.3399
Epoch 43/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3397
Epoch 44/100
250/250              1s 3ms/step -
accuracy: 0.8660 - loss: 0.3387
Epoch 45/100
250/250              1s 3ms/step -
accuracy: 0.8654 - loss: 0.3386
Epoch 46/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3386
Epoch 47/100
250/250              1s 3ms/step -
accuracy: 0.8639 - loss: 0.3383
Epoch 48/100
250/250              1s 3ms/step -
accuracy: 0.8624 - loss: 0.3382
Epoch 49/100
250/250              1s 3ms/step -
accuracy: 0.8633 - loss: 0.3380
Epoch 50/100
250/250              1s 3ms/step -
accuracy: 0.8635 - loss: 0.3379
Epoch 51/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3377
Epoch 52/100
```

```
250/250              1s 4ms/step -
accuracy: 0.8639 - loss: 0.3371
Epoch 53/100
250/250              1s 3ms/step -
accuracy: 0.8641 - loss: 0.3365
Epoch 54/100
250/250              1s 3ms/step -
accuracy: 0.8619 - loss: 0.3369
Epoch 55/100
250/250              1s 3ms/step -
accuracy: 0.8631 - loss: 0.3369
Epoch 56/100
250/250              1s 3ms/step -
accuracy: 0.8627 - loss: 0.3364
Epoch 57/100
250/250              1s 3ms/step -
accuracy: 0.8634 - loss: 0.3354
Epoch 58/100
250/250              1s 3ms/step -
accuracy: 0.8629 - loss: 0.3366
Epoch 59/100
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3356
Epoch 60/100
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3358
Epoch 61/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3360
Epoch 62/100
250/250              1s 3ms/step -
accuracy: 0.8643 - loss: 0.3356
Epoch 63/100
250/250              1s 3ms/step -
accuracy: 0.8644 - loss: 0.3357
Epoch 64/100
250/250              1s 3ms/step -
accuracy: 0.8631 - loss: 0.3354
Epoch 65/100
250/250              1s 3ms/step -
accuracy: 0.8630 - loss: 0.3360
Epoch 66/100
250/250              1s 3ms/step -
accuracy: 0.8633 - loss: 0.3353
Epoch 67/100
250/250              1s 3ms/step -
accuracy: 0.8640 - loss: 0.3361
Epoch 68/100
```

```
250/250              1s 3ms/step -
accuracy: 0.8665 - loss: 0.3354
Epoch 69/100
250/250              1s 3ms/step -
accuracy: 0.8633 - loss: 0.3358
Epoch 70/100
250/250              1s 3ms/step -
accuracy: 0.8636 - loss: 0.3353
Epoch 71/100
250/250              2s 4ms/step -
accuracy: 0.8635 - loss: 0.3351
Epoch 72/100
250/250              1s 3ms/step -
accuracy: 0.8658 - loss: 0.3349
Epoch 73/100
250/250              1s 3ms/step -
accuracy: 0.8631 - loss: 0.3347
Epoch 74/100
250/250              1s 3ms/step -
accuracy: 0.8645 - loss: 0.3347
Epoch 75/100
250/250              1s 3ms/step -
accuracy: 0.8639 - loss: 0.3352
Epoch 76/100
250/250              1s 3ms/step -
accuracy: 0.8641 - loss: 0.3349
Epoch 77/100
250/250              1s 3ms/step -
accuracy: 0.8635 - loss: 0.3346
Epoch 78/100
250/250              1s 3ms/step -
accuracy: 0.8643 - loss: 0.3352
Epoch 79/100
250/250              1s 3ms/step -
accuracy: 0.8651 - loss: 0.3348
Epoch 80/100
250/250              1s 3ms/step -
accuracy: 0.8637 - loss: 0.3342
Epoch 81/100
250/250              1s 3ms/step -
accuracy: 0.8654 - loss: 0.3341
Epoch 82/100
250/250              1s 3ms/step -
accuracy: 0.8633 - loss: 0.3357
Epoch 83/100
250/250              1s 3ms/step -
accuracy: 0.8626 - loss: 0.3347
Epoch 84/100
```

```
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3351
Epoch 85/100
250/250              1s 3ms/step -
accuracy: 0.8646 - loss: 0.3349
Epoch 86/100
250/250              1s 3ms/step -
accuracy: 0.8652 - loss: 0.3353
Epoch 87/100
250/250              1s 3ms/step -
accuracy: 0.8645 - loss: 0.3345
Epoch 88/100
250/250              1s 3ms/step -
accuracy: 0.8636 - loss: 0.3355
Epoch 89/100
250/250              1s 4ms/step -
accuracy: 0.8644 - loss: 0.3345
Epoch 90/100
250/250              1s 3ms/step -
accuracy: 0.8655 - loss: 0.3342
Epoch 91/100
250/250              1s 3ms/step -
accuracy: 0.8668 - loss: 0.3344
Epoch 92/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3338
Epoch 93/100
250/250              1s 3ms/step -
accuracy: 0.8652 - loss: 0.3346
Epoch 94/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3342
Epoch 95/100
250/250              1s 3ms/step -
accuracy: 0.8648 - loss: 0.3343
Epoch 96/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3345
Epoch 97/100
250/250              1s 3ms/step -
accuracy: 0.8652 - loss: 0.3336
Epoch 98/100
250/250              1s 3ms/step -
accuracy: 0.8659 - loss: 0.3343
Epoch 99/100
250/250              1s 3ms/step -
accuracy: 0.8650 - loss: 0.3341
Epoch 100/100
```

```
250/250            1s 3ms/step -
accuracy: 0.8643 - loss: 0.3343
63/63             0s 4ms/step
Confusion Matrix (3 Hidden Layers):
[[1526   69]
 [ 210  195]]
Accuracy (3 Hidden Layers): 0.8605
```

# 8 Comparison of Models

We experimented with **different ANN architectures** (1, 2, and 3 hidden layers) and compared their performance.

## 8.1 Results Summary

| ANN Architecture | Confusion Matrix | Accuracy |
|---|---|---|
| **1 Hidden Layer** | [[1492 103] [ 186 219]] | **0.8555** |
| **2 Hidden Layers** | [[1525 70] [ 198 207]] | **0.8660** |
| **3 Hidden Layers** | [[1526 69] [ 210 195]] | **0.8605** |

## 8.2 Interpretation

- **Hidden Layer:**
  - Accuracy: ~85.5%
  - Performs reasonably well, but leaves some misclassifications.
- **Hidden Layers:**
  - Accuracy: ~86.6%
  - Best overall performance in terms of accuracy.
  - Fewer false positives compared to 1 hidden layer.
- **Hidden Layers:**
  - Accuracy: ~86.0%
  - Slightly worse than 2 layers, indicating **adding more layers did not help**.
  - More false negatives (customers leaving were predicted as staying).

## 8.3 Conclusion

- Increasing from **1 → 2 hidden layers** improved performance.
- Adding a **3rd hidden layer** did **not** improve accuracy — in fact, performance dropped slightly.

- **Best Model:** ANN with **2 hidden layers**, giving ~**86.6% accuracy**.
- More layers are not always better — they may cause **overfitting** or unnecessary complexity.
- To further improve performance, we should explore:
  - **Hyperparameter tuning** (units per layer, learning rate, batch size).
  - **Regularization techniques** (Dropout, L2).
  - **Feature engineering** (new features, removing noisy ones).