**Assignment No. 7**

**Problem Statement** :Implement and analyze an Artificial Neural Network (ANN) classifier.

**Objective:** To understand and implement an ANN for classification, analyze its performance, and evaluate how different parameters affect its accuracy.

**Prerequisite :**

1. A Python environment set up with libraries such as numpy, pandas, matplotlib, seaborn, tensorflow (keras), and sklearn.
2. Internet connection (for fetching datasets if needed).
3. Basic knowledge of machine learning, deep learning, and artificial neural networks.

**Theory :**

An Artificial Neural Network (ANN) is a computational model inspired by biological neural networks. It consists of interconnected layers of neurons that process information using weighted connections.

### **Working of ANN Classifier**

1. **Input Layer:** Accepts features from the dataset.
2. **Hidden Layers:** Performs computations using weighted sums, activation functions, and backpropagation for learning.
3. **Output Layer:** Produces classification results (e.g., probabilities for different classes).
4. **Training Process:**
   * Forward propagation: Computes the predicted output.
   * Loss calculation: Measures error between predicted and actual values.
   * Backpropagation: Adjusts weights using an optimizer (e.g., SGD, Adam).
   * Repetition: Trains for multiple epochs to improve accuracy.

### **Choosing the Right Parameters**

* Number of Layers & Neurons: More layers capture complex patterns but increase computation.
* Activation Function: Common choices include ReLU, Sigmoid, Softmax.
* Optimizer: Adam, SGD, RMSprop for weight updates.
* Loss Function: Categorical Crossentropy (for multi-class) or Binary Crossentropy (for binary classification).

### **Advantages of ANN**

Handles complex patterns in data.  
Learns non-linear relationships.  
Can improve accuracy with sufficient training.

### **Disadvantages of ANN**

Computationally expensive (requires more processing power).  
Sensitive to overfitting (requires regularization).  
Requires large amounts of labeled data for effective training.

## **Implementation Steps**

### **1. Understanding the Dataset**

* Load the dataset using pandas.
* Check dataset dimensions using .shape.
* Display column data types using .info().
* Check for missing values using .isnull().sum().

### **2. Data Preprocessing**

* Handle missing values (imputation or removal).
* Encode categorical features if necessary (LabelEncoder, OneHotEncoder).
* Normalize numerical features using MinMax Scaling or Standardization.

### **3. Splitting Data into Training and Testing Sets**

* Use train\_test\_split from sklearn.model\_selection.
* Common split ratio: 80% training, 20% testing.

### **4. Implementing ANN for Classification**

* Use Keras Sequential API to define the ANN model.
* Add layers (Input layer, Hidden layers, Output layer).
* Choose activation functions (ReLU, Sigmoid, Softmax).
* Compile the model (define loss function, optimizer, and metrics).
* Train the model using .fit() method.
* Make predictions using .predict().
* Evaluate performance using accuracy, precision, recall, confusion matrix.

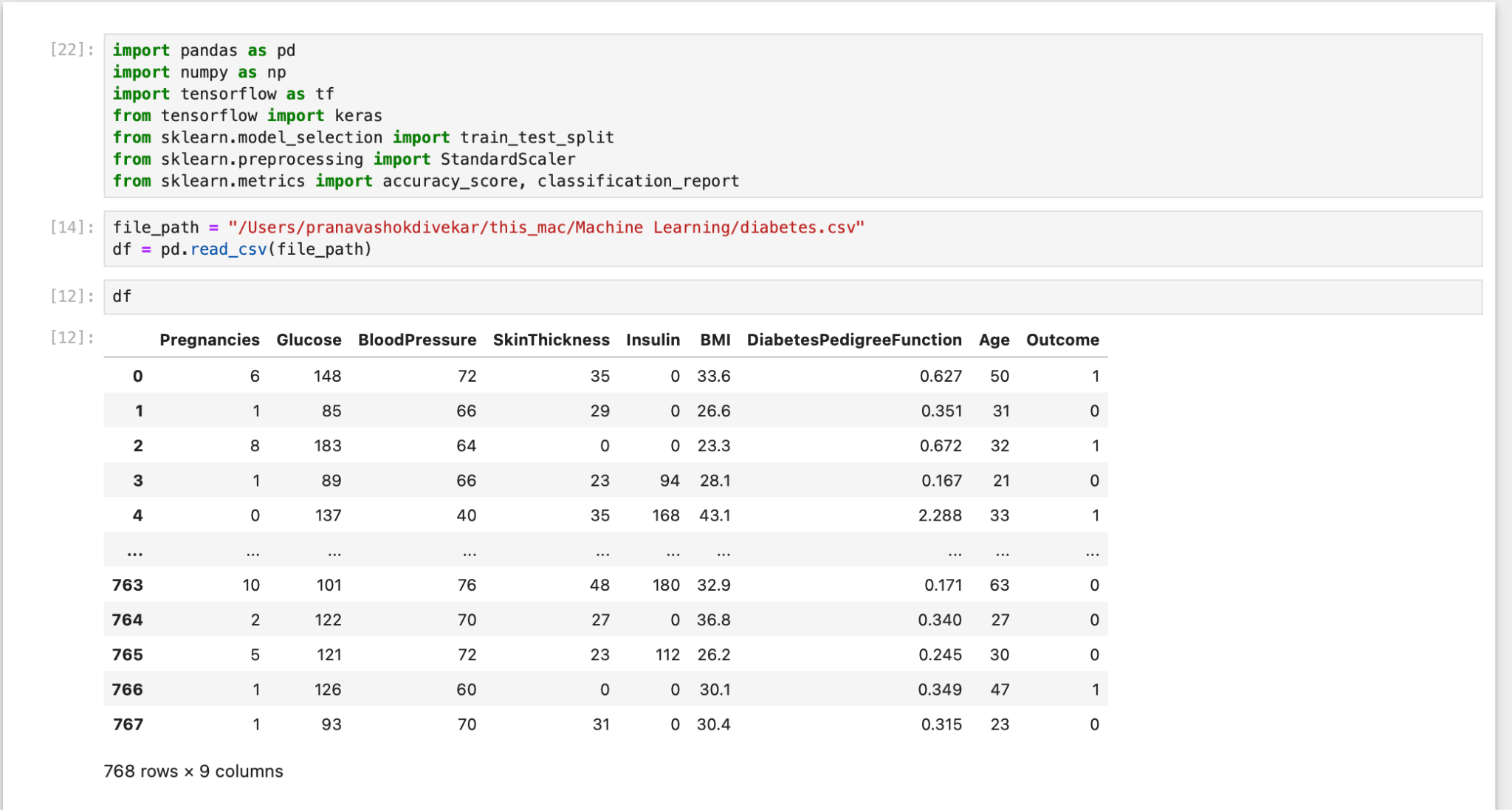
### **5. Hyperparameter Tuning**

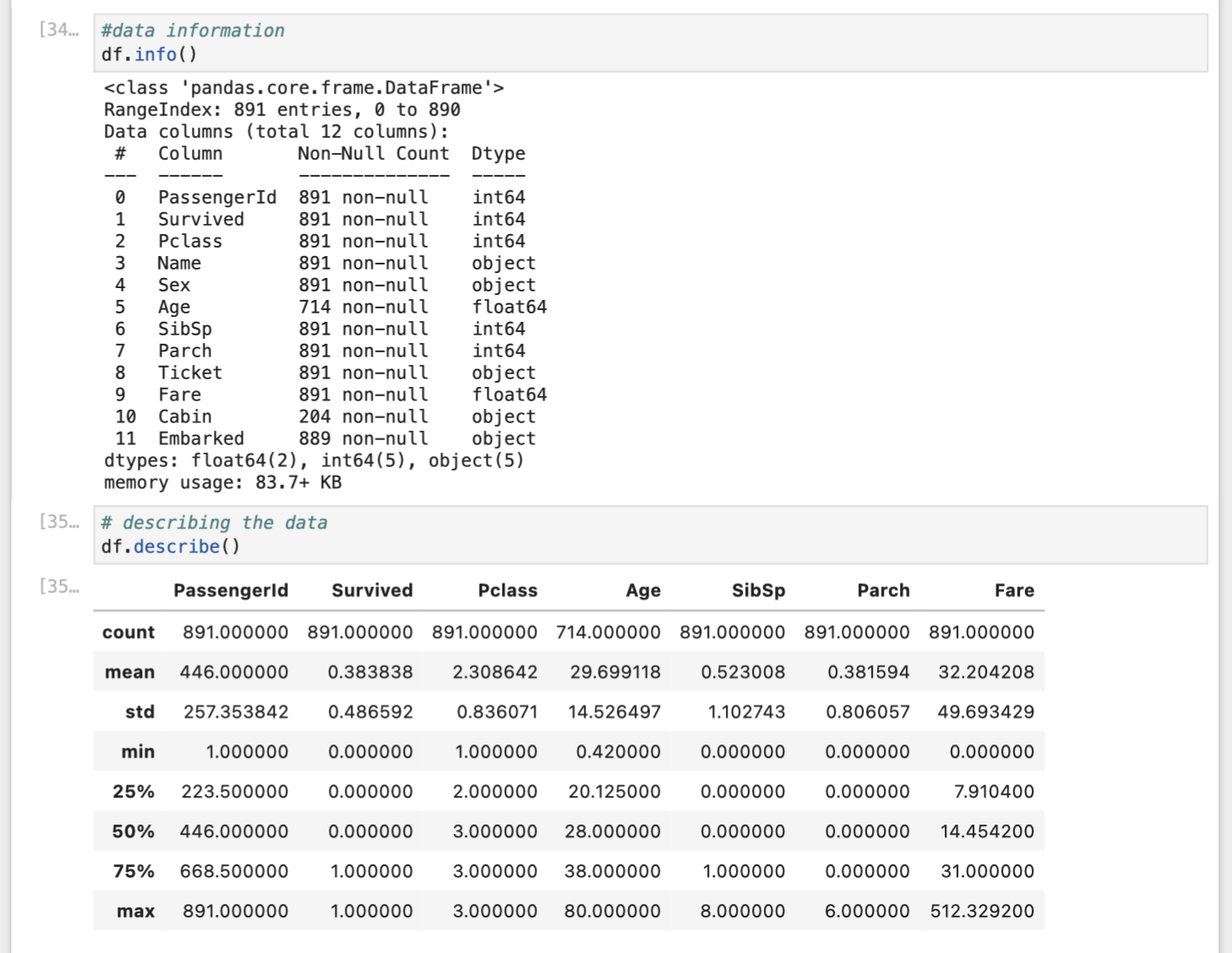
* Experiment with different numbers of layers and neurons.
* Try different optimizers (Adam, RMSprop, SGD).
* Test different activation functions (ReLU, Sigmoid, Tanh).
* Use early stopping to prevent overfitting.

### **6. Data Visualization**

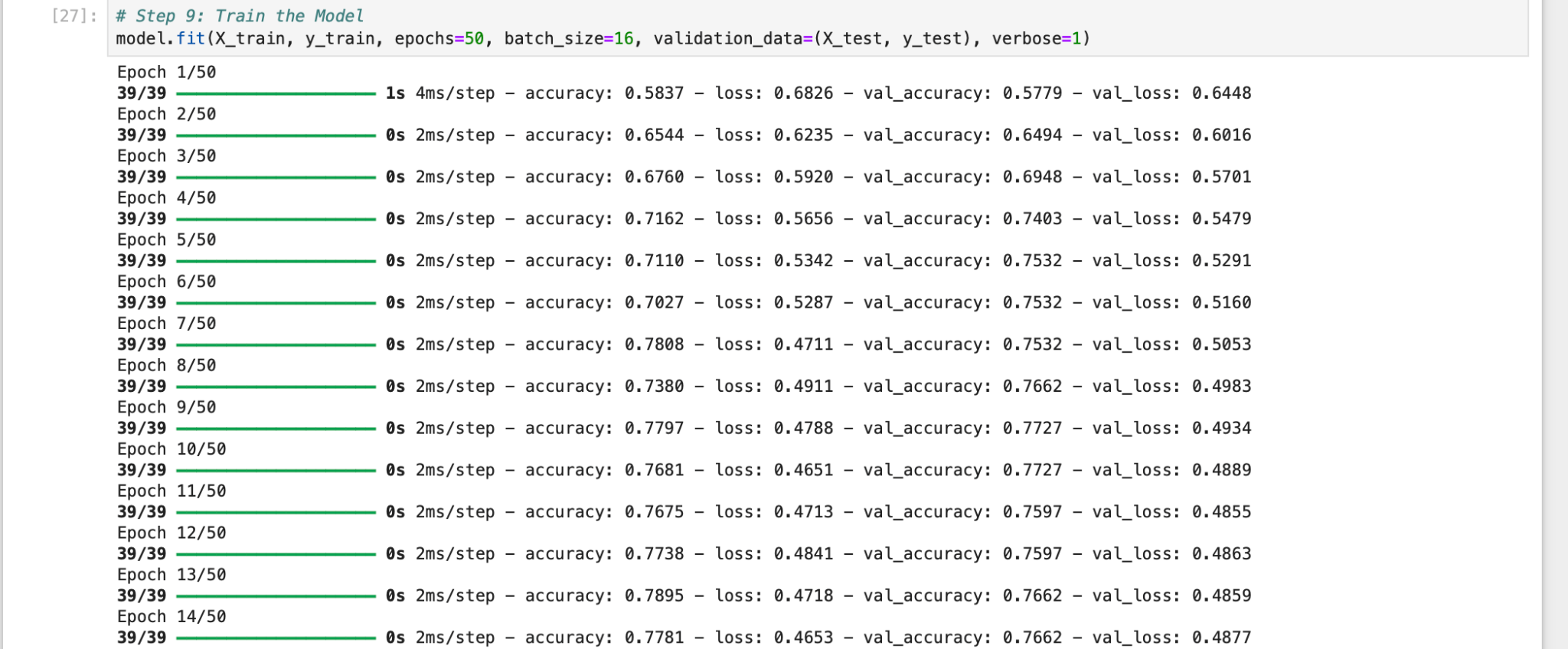
* Plot training loss and accuracy curves over epochs.
* Visualize confusion matrix for classification results.
* Compare accuracy for different architectures and hyperparameters.

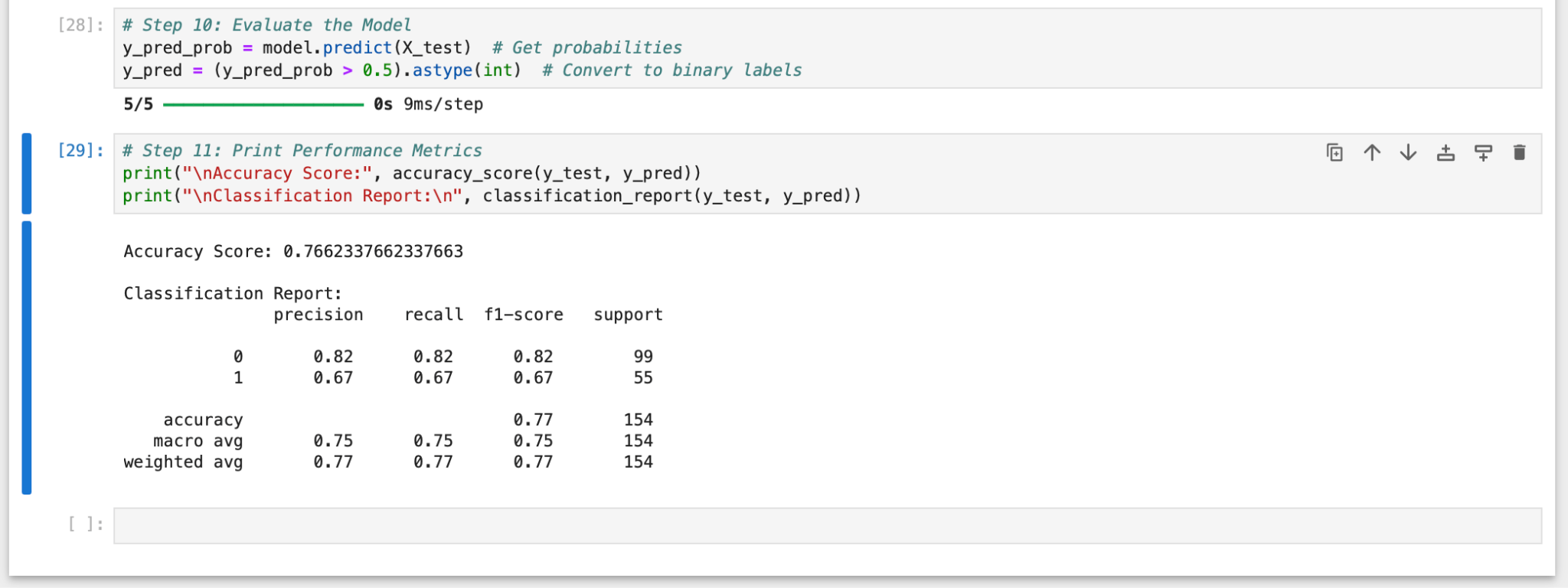
**Code & Output :**



****







**Github :- https://github.com/SagarGaneshkar/ML\_Codes**

**Conclusion:**

The ANN classifier achieved 76.62% accuracy, meaning it correctly predicted 77% of cases.

* Class 0: Good performance (82% precision & recall).
* Class 1: Weaker performance (67% precision & recall), likely due to class imbalance.
* The model works well but struggles with the minority class.
* Adjusting data balance or tuning parameters can improve results.