Perceptron from Scratch for Diabetes Prediction

1 Introduction

This project implements a Perceptron neural network from scratch to predict diabetes outcomes. Instead of using high level machine learning libraries, the model logic including weight initialization, activation function, forward computation, and training loop is manually implemented. This approach helps build a strong conceptual understanding of binary classification.

2 Dataset Overview

The diabetes dataset contains medical attributes such as glucose levels, blood pressure, insulin values, body mass index, age, and other clinical indicators. The target variable represents whether a patient is diabetic or non diabetic.

3 Data Preprocessing

Data preprocessing is performed before training the perceptron. This includes separating features and labels, handling missing or zero values if present, and preparing the numerical input data for stable learning behavior. The dataset is then split into training and testing sets.

4 Perceptron Class Implementation

The core logic of the model is encapsulated inside a separate perceptron class file. This class is responsible for defining model parameters such as weights and bias, implementing the sigmoid activation function, computing predictions, and updating weights during training. By isolating the perceptron logic into its own class, the code becomes modular, reusable, and easier to maintain.

5 Using Data with the Perceptron Class

The main notebook imports the perceptron class and uses it to train on the diabetes dataset. Feature data is passed to the perceptron class as input vectors, while the corresponding labels are provided as target values. The perceptron class processes this data internally to perform forward propagation, calculate prediction errors, and update weights iteratively. This separation of data handling and model logic improves clarity and scalability.

6 Activation Function

A sigmoid activation function is used to map the weighted sum of inputs to a probability value between zero and one. This allows the perceptron to output interpretable probabilities suitable for binary classification.

7 Training Process

The perceptron is trained using gradient descent. For each training example, the model computes the prediction, calculates the error with respect to the true label, and updates weights and bias using the learning rate. This process is repeated for multiple epochs to minimize classification error.

8 Model Evaluation

After training, the model is evaluated on unseen test data. Prediction accuracy is used as the primary evaluation metric. This step helps assess how well the perceptron generalizes to new patient data.

9 Limitations

A single layer perceptron can only learn linear decision boundaries. Medical datasets often contain complex non linear relationships that may not be fully captured by this model.

10 Future Improvements

Future enhancements may include extending the model to multiple layers, experimenting with different activation functions, applying regularization, and comparing results with advanced machine learning models.

11 Conclusion

This project demonstrates how a perceptron operates internally by building it from scratch. Using a separate perceptron class improves code organization and reinforces core neural network concepts. The project serves as a solid foundation for deeper exploration into machine learning and deep learning.

Author

Satyam Gajjar