Pima Indians Diabetes Dataset

Q1. Dataset Selection: What dataset did you choose? Describe its purpose and target.

A1. I selected the Pima Indians Diabetes Dataset, which contains medical diagnostic measurements of female patients of Pima Indian heritage, aged 21 or older. The dataset's purpose is to predict the likelihood of diabetes based on various health parameters such as Glucose, Blood Pressure, Skin Thickness, Insulin Level, BMI, Pregnancies, Diabetes Pedigree Function and Age. The target variable is binary:

- **1:** Patient is likely to have diabetes.
- **0:** Patient is not likely to have diabetes.

Q2. EDA Findings: What key insights did you gain during EDA?

A2. Upon conducting EDA on this dataset, several important observations were made:

1. Certain medical features, such as Glucose, Blood Pressure, Skin Thickness, Insulin, and BMI, contained zero values, which are physiologically implausible. These were considered as missing values requiring treatment.

2. The target variable was found to be **imbalanced**, with a greater number of non-diabetic cases compared to diabetic cases.

3. Correlation analysis revealed that *Glucose level* had the highest positive correlation with diabetes occurrence, followed by BMI and Age.

4. Outliers were detected in features such as Insulin and Skin Thickness, which could potentially affect model performance if left untreated.

5. Several features displayed skewed distributions, particularly Insulin and Diabetes Pedigree Function.

Q3. Feature Engineering: What transformations did you apply? Why?

A3. The following transformations were applied to enhance data quality and improve model performance:

- Missing Value Handling
- Outlier Removal by IQR method.
- Feature Scaling by applying StandardScaler
- Dataset was split into **80% training** and **20% testing** to evaluate model performance on unseen data.

Q4. ANN Architecture: List number of layers, neurons, activation functions.

A4. The Artificial Neural Network (ANN) was implemented using the Sequential API in Keras with the following structure:

1. Input Layer:
    - Neurons: 12
    - Input Features: 8
    - Activation Function: ReLU
2. **Hidden Layer 1:**
    - **Neurons:** 8
    - **Activation Function:** ReLU to introduce non-linearity and handle complex feature interactions.
3. **Output Layer:**
    - **Neurons:** 1
    - **Activation Function:** Sigmoid to output probabilities between 0 and 1 for binary classification (diabetic or non-diabetic).

Q5. Performance Summary: What were the final accuracy and loss?

A5. The model was evaluated on the test dataset after training. The final results were as follows:

- Test Accuracy: 72.07%
- Test Loss: 0.4868

The accuracy indicates that the model correctly classified approximately 72% of the instances in the test set.

The loss value reflects the error in prediction probability compared to the actual labels, with a lower value signifying better fit.

Q6. Confusion Matrix Analysis: What did it reveal about model behavior?

A6. The confusion matrix provided a detailed breakdown of the model's classification performance:

- **True Positives (TP):** Cases where the model correctly predicted the positive class (diabetic patients).
- **True Negatives (TN):** Cases where the model correctly predicted the negative class (non-diabetic patients).
- **False Positives (FP):** Non-diabetic patients incorrectly predicted as diabetic.
- **False Negatives (FN):** Diabetic patients incorrectly predicted as non-diabetic.

From the confusion matrix, it was observed that:

1. The model demonstrated relatively stronger performance in identifying non-diabetic cases (high TN count i.e., 81).

2. The number of false negatives was comparatively higher than desired, indicating that the model sometimes fails to detect actual diabetic cases, which is critical in medical diagnosis.

Q7. Hyperparameter Tuning: What parameters did you change and what was the result?

A7. I tuned the ANN by modifying three key parameters:

- **Epochs**
- **Batch Size**
- **Learning Rate**.

The final configuration used 100 epochs, a batch size of 32, and the Adam optimizer with a learning rate of 0.0005. This combination provided the best balance between convergence speed and generalization, yielding a **test accuracy of 72.07%** and a **test loss of 0.487**. Further improvement was limited by dataset quality and inherent complexity of the problem.

Q8. Improvement Ideas: Suggest one way to improve performance.

A8. By increasing the dataset size through data augmentation or acquiring more real-world samples could help the ANN learn more robust patterns boosting accuracy significantly.

Q9. Overfitting/Underfitting: Did you face it? How did you address it?

A9. The model exhibited signs of underfitting, as both training and testing accuracy remained around 71–72%, indicating that it could not fully capture the underlying patterns in the dataset. To address this, I tuned hyperparameters (epochs, batch size, learning rate) and experimented with the network architecture. However, due to possible limitations in dataset complexity and size, the performance gain was minimal.