Assignment 1: Predict diabetes using Perceptron

Aditya Anant Deshpande - a1910571

***Abstract*—Diabetes requires early diagnosis for effective management. This study uses single-layer and multi-layer Perceptron models to predict diabetes using the Pima Indians Diabetes Database. After preprocessing, including imputing missing values and scaling, models were trained and evaluated. The single-layer Perceptron outperformed the multi-layer model, achieving 81.17% accuracy and 76% precision. The Random Forest classifier performed similarly with 81% accuracy and 77% precision. This paper presents the methodology, results, and potential improvements.**

I. Introduction

Diabetes is a significant global health issue that necessitates early detection to prevent severe complications. Machine learning models, particularly Perceptron-based algorithms, have gained attention for medical diagnostics, especially in predicting conditions like diabetes. In this paper, we applied both single-layer and multi-layer Perceptron models for diabetes prediction using the Pima Indians Diabetes Database. The dataset contains physiological data, allowing us to create binary classifiers to distinguish diabetic and non-diabetic patients.

II. Methodology

A. Dataset

The Pima Indians Diabetes Database was used for this study. It contains 768 instances with 9 attributes: 'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', and 'Outcome'. The data types for the features are as follows:

- **Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, Age, Outcome**: `int64`
- **BMI, DiabetesPedigreeFunction**: `float64`

The following table shows descriptive statistics of the dataset, including mean, standard deviation, and percentiles:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

Fig. 1. Summary statistics of the Pima Indians Diabetes Database.

The target variable, 'Outcome', is binary, indicating whether a patient has diabetes (1) or not (0). Below is the distribution of the target classes (Diabetes vs. No Diabetes), where 34.9% of the patients are diabetic, and 65.1% are non-diabetic.
To further understand the relationships between the features and the target variable, we also computed the correlation matrix. The following heatmap (Fig. 3) shows the pairwise correlation between all features. Higher values indicate a stronger correlation, either positive or negative. For example, 'Glucose' shows a strong positive correlation with the 'Outcome' variable, meaning higher glucose levels are indicative of diabetes.

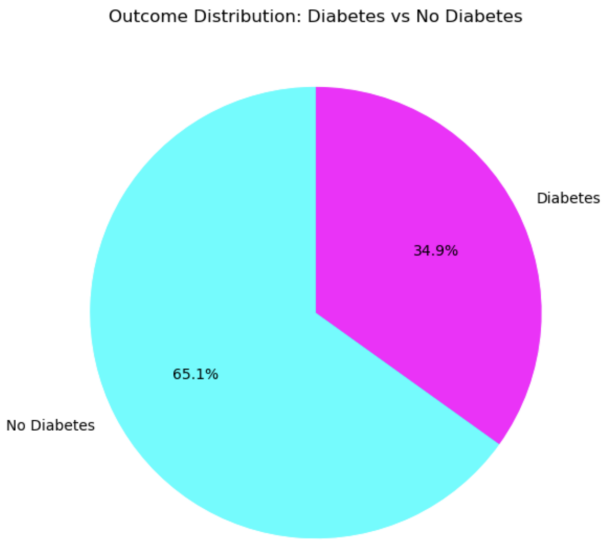

Fig. 2. Outcome distribution: Diabetes vs No Diabetes.

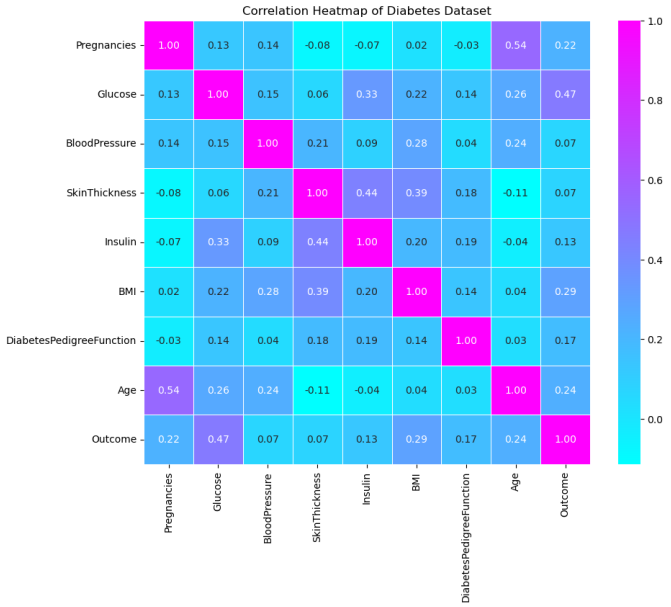

Fig. 3. Correlation heatmap of the Pima Indians Diabetes Database features.

This heatmap provides insights into the relationships between features such as ‘Age‘, ‘BMI‘, and ‘Pregnancies‘ with the target ‘Outcome‘ variable. These relationships were considered during the model training and feature selection process.

B. Data Preprocessing

Before training the models, several preprocessing steps were performed to clean the data and handle outliers:

- **Handling Outliers**: Box plots were created to visually inspect the distribution of features and identify outliers. The box plots for features such as ‘Glucose‘, ‘BloodPressure‘, ‘SkinThickness‘, ‘Insulin‘, and ‘BMI‘ (shown in Fig. 4) revealed the presence of outlier values, especially zeros in some of these features, which are not realistic values for the corresponding measurements.

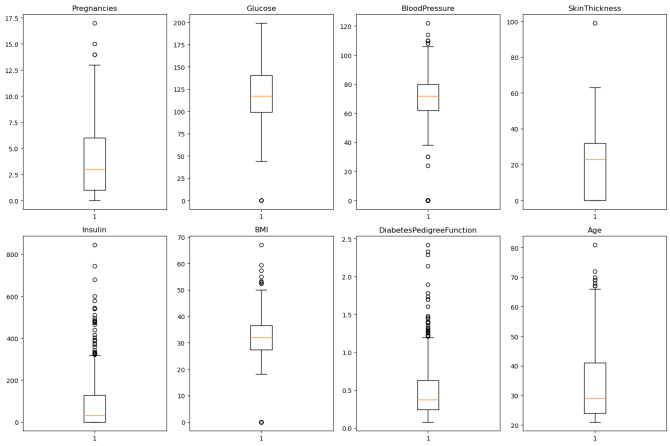

Fig. 4. Box plots of features to identify outliers in the Pima Indians Diabetes Database.

- **Imputing Missing Values (Outliers)**: Many instances in the dataset contained zeros for features like ‘Glucose‘, ‘BloodPressure‘, ‘SkinThickness‘, ‘Insulin‘, and ‘BMI‘, which are physiologically implausible. These values were treated as missing values and replaced with the median value of the respective feature. The imputation was applied to ensure that no unrealistic zeros remained in the dataset, which could distort the model training process. This was done using a loop that iterated through the relevant columns, replacing zeros with the median of each respective column.

After preprocessing, the dataset was standardized using `StandardScaler` to ensure all features were on a comparable scale. This was necessary for models like the multi-layer Perceptron (MLP) that are sensitive to feature scaling.

C. Model Architecture

Two Perceptron models were built for this study:

- **Single-Layer Perceptron (SLP)**: This model was created using the `MLPClassifier` from the `sklearn.neural_network` module. The hidden layer size was set to an empty tuple, which effectively makes it a single-layer Perceptron (i.e., no hidden layers). The model was trained for a maximum of 2000 iterations with a random state of 12 to ensure reproducibility. The model was fit to the scaled training data, and predictions were generated on the scaled test data.
- **Multi-Layer Perceptron (MLP)**: This model was also implemented using the `MLPClassifier`, but with two hidden layers. The first hidden layer had 50 neurons, and the second hidden layer had 20 neurons. Similar to the SLP, the model was trained for a maximum of 2000 iterations, with a random state of 12 to ensure reproducibility. After training on the scaled training data, predictions were made on the scaled test data.

Both models used the Adam optimizer, which adapts the learning rate during training, and employed early stopping to prevent overfitting. The models were evaluated using metrics such as accuracy, precision, recall, and F1-score, which will be discussed in the Experimental Analysis section.

D. Training and Evaluation

Once the data was preprocessed and scaled, the models were trained and evaluated using standard metrics such as accuracy, precision, recall, and F1-score. The following steps were taken for training and evaluation:

- **Standardization**: The training and testing datasets were standardized using `StandardScaler` to ensure that all features were on the same scale. This was particularly important for models like the multi-layer Perceptron (MLP), which are sensitive to feature scaling.
- **Single-Layer and Multi-Layer Perceptron Models**: - The single-layer Perceptron (SLP) and multi-layer Perceptron (MLP) models were trained using the scaled data. The SLP did not include any hidden layers, while the MLP had two hidden layers with 50 and 20 neurons, respectively. - After training, predictions were made on the test set. The performance of both models was evaluated using accuracy, precision, recall, and F1-score. These metrics were computed using `sklearn` functions to assess the quality of the predictions.
- **Random Forest Model**: - A Random Forest classifier was also trained, but only for comparison purposes to evaluate its accuracy relative to the Perceptron models. The model was initialized using the `RandomForestClassifier` from `sklearn` with a random state of 12 to ensure reproducibility. - The Random Forest model was trained on the scaled training data and evaluated on the test data. - Evaluation metrics, including accuracy, precision, recall, and F1-score, were calculated for the Random Forest model to compare its performance with the single-layer and multi-layer Perceptron models.

The comparison of evaluation metrics for all three models (SLP, MLP, and Random Forest) allowed us to analyze which model performed best for the diabetes prediction task. While

the Random Forest was only used as a benchmark to compare accuracy, the focus was on understanding the performance of the Perceptron models in detail.

III. Experimental Analysis

In this section, we describe the performance of the single-layer Perceptron, multi-layer Perceptron, and Random Forest models. The evaluation metrics used for comparison are accuracy, precision, recall, and F1-score. Additionally, confusion matrices were used to further analyze the classification performance of each model.

A. Evaluation Metrics

The performance of the single-layer Perceptron, multi-layer Perceptron, and Random Forest models was evaluated using accuracy, precision, recall, and F1-score. Below are the detailed results for each model:

- **Single-Layer Perceptron**:
  - Accuracy: 81.17%
  - Precision: 76%
  - Recall: 69%
  - F1 Score: 72.38%
- **Multi-Layer Perceptron**:
  - Accuracy: 75.32%
  - Precision: 66.04%
  - Recall: 63.64%
  - F1 Score: 64.81%
- **Random Forest**:
  - Accuracy: 81%
  - Precision: 77%
  - Recall: 67%
  - F1 Score: 72%

The comparison of the evaluation metrics demonstrates that the single-layer Perceptron outperformed the multi-layer Perceptron across all metrics. The Random Forest classifier achieved similar performance to the single-layer Perceptron, particularly in terms of precision.

B. Results

To better visualize the differences between the models, we generated a bar plot comparing the accuracy, precision, recall, and F1-score for the single-layer Perceptron and multi-layer Perceptron models.

The bar plot (Fig. 5) shows that the single-layer Perceptron consistently outperformed the multi-layer Perceptron across all evaluation metrics. This indicates that, for this particular dataset, the simpler single-layer architecture provided better results than the more complex multi-layer model.

Additionally, confusion matrices were generated for both the single-layer and multi-layer Perceptron models to further investigate their classification performance.

The confusion matrices (Fig. 6) show that the single-layer Perceptron had fewer misclassifications compared to the multi-layer Perceptron, particularly in predicting diabetic patients (class 1).

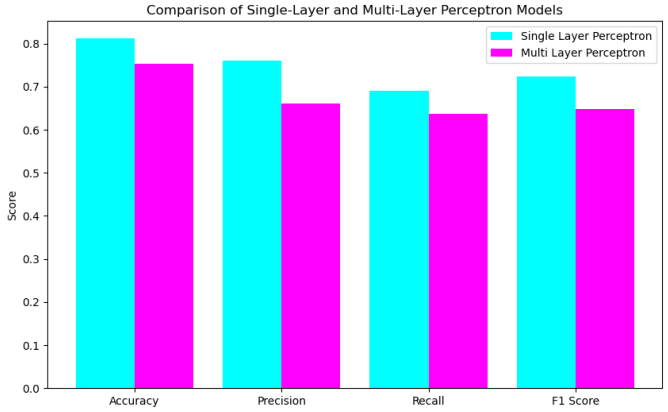

Fig. 5. Comparison of Single-Layer and Multi-Layer Perceptron Models.

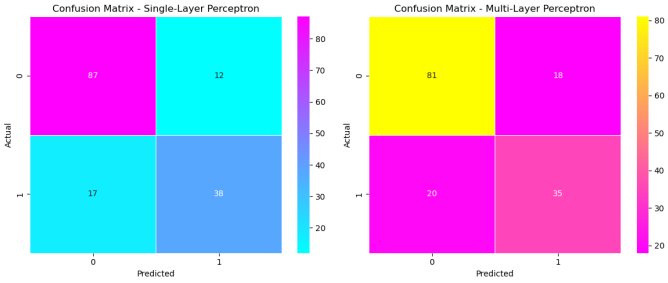

Fig. 6. Confusion Matrices: Single-Layer Perceptron (left) and Multi-Layer Perceptron (right).

Lastly, the Random Forest model's performance was also analyzed using a confusion matrix:

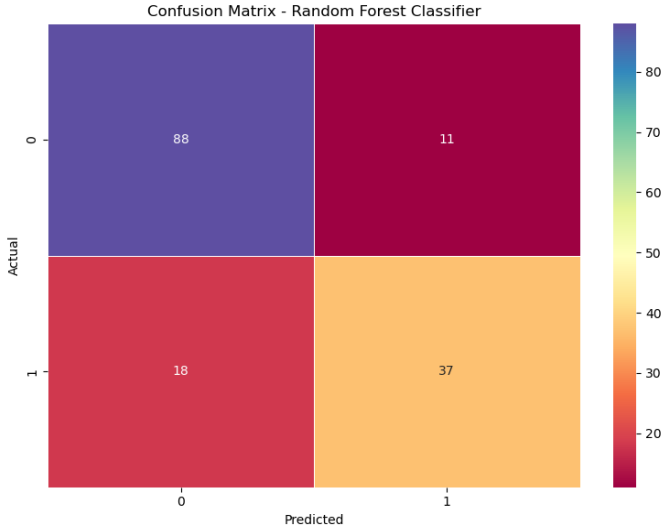

Fig. 7. Confusion Matrix for the Random Forest Classifier.

The confusion matrix for the Random Forest classifier (Fig. 7) reveals a similar performance to the single-layer Perceptron. The Random Forest model made a few misclassifications, with slightly better precision but similar recall and F1-score

compared to the single-layer Perceptron.

IV. Conclusion

In this study, we evaluated the performance of single-layer and multi-layer Perceptron models, as well as a Random Forest classifier, for predicting diabetes based on the Pima Indians Diabetes Database. The results showed that the single-layer Perceptron achieved the highest overall performance, with an accuracy of 81.17%, a precision of 76%, a recall of 69%, and an F1-score of 72.38%. The multi-layer Perceptron, while more complex, did not perform as well, achieving an accuracy of 75.32% with lower precision and recall scores. The Random Forest classifier performed similarly to the single-layer Perceptron, achieving an accuracy of 81% and a precision of 77%.

These results suggest that, for this specific dataset, simpler models like the single-layer Perceptron and Random Forest offer strong performance with lower computational complexity. The added complexity of the multi-layer Perceptron did not result in improved performance, highlighting the importance of model selection based on the problem at hand. Future work could explore further tuning of the multi-layer Perceptron or the use of other ensemble methods to potentially improve performance. Additionally, more advanced techniques for handling class imbalance could be explored to further enhance model performance in predicting diabetic patients.

V. Code and Repository

The full implementation of the models, data preprocessing steps, and evaluation metrics discussed in this paper can be found in the GitHub repository linked below. The code has been made available for reproducibility and further experimentation.

- **GitHub Repository:** https://github.com/aadi654/DLF_Assignments

VI. References

- Jahangir, M., et al. (2017). *An Expert System for Diabetes Prediction Using Auto-Tuned Multi-Layer Perceptron*. Intelligent Systems Conference.
- Krishnan, K. T., & Thaiyalnayaki, K. (2021). *Classification of Diabetes Using Deep Learning and SVM Techniques*.
- Mirshahvalad, R., et al. (2017). *Diabetes Prediction Using Ensemble Perceptron Algorithm*. 9th International Conference on Computational Intelligence and Communication Networks.