Qingyu Zhang / Andy

N-number: N19903322

Prof. Pascal Wallisch

Fundamentals of Machine Learning

HW4 Report: Neural Networks

Question 1:

1) What was done:

I trained a **single-layer perceptron without any hidden layer or activation function** to classify diabetes status (0/1). The model receives 21 input features and outputs a single logit, trained using BCEWithLogitsLoss, Adam Optimizer, learning rate of 0.001, and 20 epochs. Calculated AUC and plotted ROC curve. I used DataLoader to make the model train for all batches of dataset in each epoch (batch size 64)

2) Why this was done:

The perceptron serves as a linear baseline model for classification. It is the simplest possible neural architecture, and provides a benchmark for comparison with deeper models. By starting with no hidden layers or nonlinearities, we can observe how much performance can be achieved with only linear decision boundaries on this dataset.

We used BCEWithLogitsLoss because it is the most numerically stable and efficient way to handle binary classification tasks—it combines a sigmoid activation and binary cross-entropy loss in a single function, avoiding issues like log(0) or gradient saturation. The Adam optimizer was chosen because it is adaptive, robust to noisy gradients, and

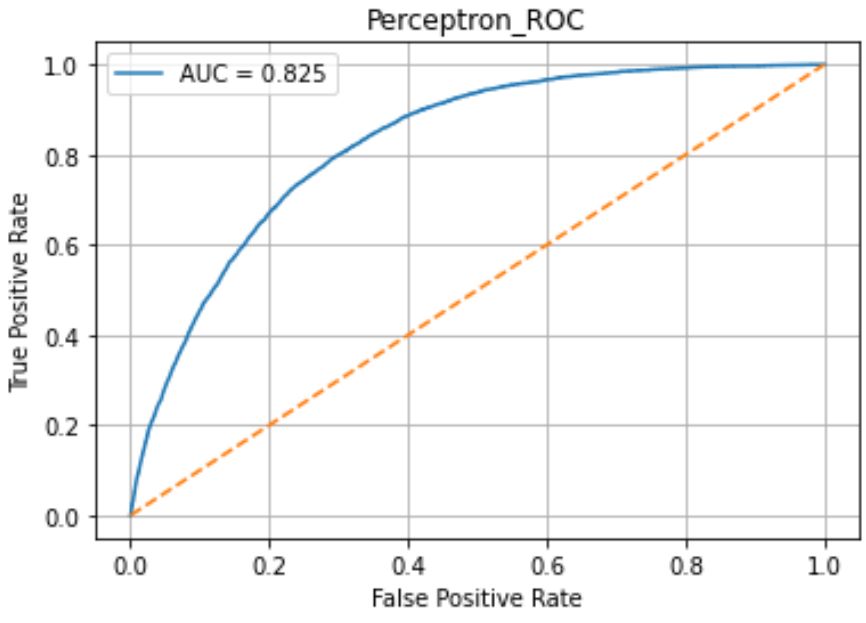
generally provides fast convergence without requiring much tuning. A learning rate of 0.001 is a safe and commonly effective default for Adam in small to medium neural networks.

We trained for 20 epochs to ensure the model had sufficient opportunity to converge, as this is a moderate and computationally efficient number of iterations given the dataset size. We used DataLoader with batch size 64 to enable mini-batch gradient descent, which balances between the efficiency of batch updates and the stability of stochastic optimization. It also ensures that the entire training set is covered in each epoch without overloading GPU or CPU memory.

3) Findings:

The **perceptron** achieved a test **AUC of 0.825**, which is competitive with shallow MLPs. The ROC curve showed a smooth and consistent tradeoff between true positive and false positive rates.

```
[Task 1] AUC (Perceptron): 0.825105046195604
```



4）Interpretation:

The **perceptron** achieved a test **AUC of 0.825**, which is competitive with shallow MLPs. The results suggest that the dataset has a reasonably strong linear structure for separating diabetic and non-diabetic individuals. This also indicates that while deeper models might yield small improvements, a simple linear classifier already captures the most predictive signal from the 21 input features.

Question 2:

1) What was done:

MLPs with 1 and 2 Hidden Layers + Different Activations.

I trained multiple Feedforward Neural Networks (MLPs) with:

- Hidden layers: 1 layer ([32]) and 2 layers ([64, 32])
- Activation functions: None, ReLU, and Sigmoid

Each configuration was evaluated using AUC on the test set. A comparison table (including the result of single Perceptron in Question 1) was printed.

2) Why it was done:

The goal was to assess how performance depends on architecture depth (number of hidden layers) and activation type. ReLU and Sigmoid are common nonlinearities, while "None" simulates a deeper linear model.

(Models in this question also use the same Loss function, Optimizer, learning rate, and number of epochs as in Question 1. I also use DataLoader in this question. The reasons for using all of these are the same as in Question 1.)

3) Findings:

Test AUC results of different models are these:

Summary Table (Task 2):

Model	Hidden Size	Test AUC
MLP-1-None	(32,)	0.825
MLP-1-sigmoid	(32,)	0.833
MLP-1-relu	(32,)	0.833
MLP-2-None	(64, 32)	0.825
MLP-2-sigmoid	(64, 32)	0.833
MLP-2-relu	(64, 32)	0.830
Perceptron (Q1)	N/A	0.825

MLP-1-sigmoid, **MLP-1-relu**, and **MLP-2-sigmoid** show the best performance and their **AUC** scores are all about **0.833**.

4) Interpretation:

Deeper networks do not consistently outperform simpler ones. Those **Feedforward Neural Networks with AUC of 0.833** (no matter 1 hidden layer or 2, they all use activation functions) **slightly outperform** single Perceptron in Question 1. On the other hand, **activation functions matter**: both sigmoid and ReLU improve performance

slightly over the linear baseline. Although we can't tell which is better between sigmoid and ReLU, they are both better than no activation function. For the number of hidden layers, one-layer networks performed similarly compared with two-layer ones on this dataset, indicating that **adding depth provided limited benefit (AUC almost didn't change)**. For networks using ReLU, increasing depth even shows diminishing returns on this dataset (AUC from 0.833 to 0.830).

Question 3:

1) What was done:

   I trained and compared a 2-hidden layers deep FFN ([64, 32] with ReLU) with a 1D CNN (8 filters, kernel size 3) followed by a flattening layer and a final linear layer. Both models were trained using the Adam optimizer, a learning rate of 0.001,

BCEWithLogitsLoss, and 20 epochs. We evaluated their performance on the test set using ROC curves and AUC scores.

2) Why it was done:

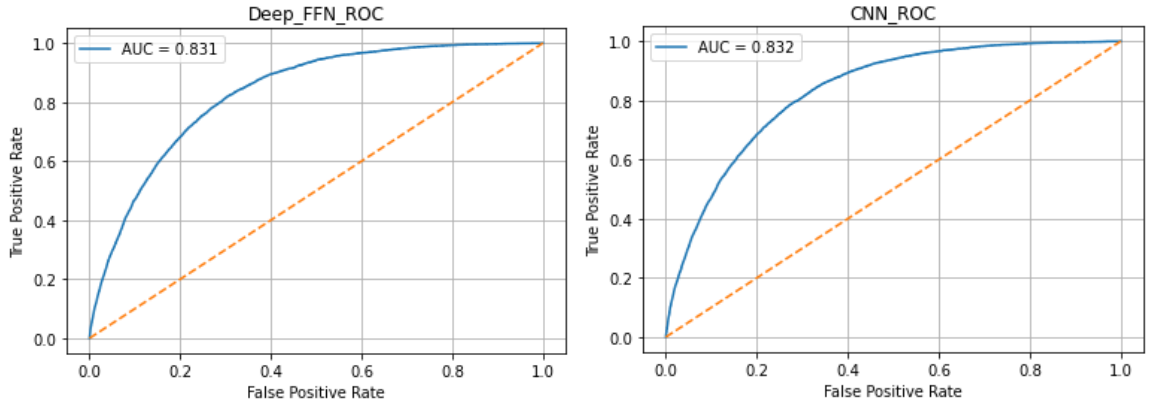
This task was designed to evaluate whether a convolutional architecture, which is typically used in image and sequence modeling, could outperform a deep feedforward network on tabular data. While CNNs are excellent at capturing local spatial patterns, tabular datasets like this one usually lack inherent spatial structure. By comparing CNN with Deep FFN, we assess whether local feature interactions across adjacent columns carry meaningful predictive information in this context.

3) Findings:

- **Deep FFN AUC**: 0.831
- **CNN AUC**: 0.832

The CNN achieved a marginally higher AUC than the deep FFN, but both models performed very similarly.

```
[Task 3] AUC (Deep FFN): 0.8306374737726581
[Task 3] AUC (CNN): 0.8316180719022932
```



4) Interpretation:

   The results suggest that although the CNN is capable of learning slightly more expressive local patterns by sliding over the feature space, its improvement over a traditional MLP is minimal on this tabular dataset. This aligns with expectations—tabular data typically does not have strong local dependencies like pixels in an image or words in a sentence. Therefore, while CNNs are theoretically more flexible, their practical advantage here is limited. A well-structured FFN remains a strong baseline for this type of classification task.

Question 4:

1) What was done:

   I trained FFNs with a single hidden layer (size [32]) to predict BMI. Activation functions compared: None, Sigmoid, and ReLU. All models were trained using Mean Squared Error (MSELoss), the Adam optimizer, a learning rate of 0.001, batch size 64, and for 20 epochs. Model performance was measured on the test set using **Root Mean Squared Error (RMSE)**.

2) Why it was done:

This task evaluates the impact of activation functions on the performance of shallow neural networks for regression tasks. Since MLPs become nonlinear function approximators only when activation functions are introduced, this comparison helps determine whether BMI prediction benefits from modeling nonlinear relationships. Using a shallow network allows us to isolate the effect of nonlinearity without conflating it with model depth. We also used RMSE as the evaluation metric because it penalizes larger errors more heavily, which is important in health-related predictions like BMI. The training setup (Adam, LR=0.001, batch size 64, 20 epochs) follows conventions established in Tasks 1–3 for consistency across tasks.

3) Findings:

The test RMSEs for each model variant were:

Activation	RMSE
None	6.130
Sigmoid	5.978
ReLU	6.058

```
[Task 4] RMSE (act=None): 6.130297180240645
[Task 4] RMSE (act=sigmoid): 5.977673316398939
[Task 4] RMSE (act=relu): 6.058406116130607
```

The model with sigmoid activation achieved the lowest RMSE, slightly outperforming both the linear model and the ReLU-activated version.

4) Interpretation:

These results indicate that introducing nonlinearity improves regression performance, even in shallow models. The sigmoid activation likely captures subtle, saturating relationships between inputs and BMI that a purely linear model (activation=None) misses. Interestingly, while ReLU is popular in deep networks, it performed slightly worse than sigmoid here—possibly due to the shallow network structure and the lack of zero-centered output.

From the RMSE values obtained (ranging from 5.978 to 6.130), we can conclude that the model demonstrates moderate predictive accuracy. An RMSE of ~6 means that, on average, the predicted BMI values deviate from the actual values by approximately 6 units. This level of error is acceptable for a simple one-hidden-layer network and indicates that the model captures a substantial portion of the variance in BMI based on the available features.

The overall RMSE differences are small (within ~0.15), suggesting that BMI may depend on largely linear combinations of features, but with some minor nonlinear interactions. A single hidden layer is sufficient to capture most of this signal.

Question 5:

1) What was done:

   I designed and trained a deeper feedforward neural network to predict BMI using the same input features as in Question 4. This time, the architecture included **three hidden layers** with sizes [64, 32, 16] and **ReLU** activation after each hidden layer. The model was trained using the Adam optimizer (learning rate = 0.001), MSELoss, batch size of 64, and 20 epochs. I evaluated the model's predictive performance using RMSE on the held-out test set, consistent with the setup in Question 4.

2) Why it was done:

   Question 5 aims to explore whether **deeper architectures** can improve regression performance compared to the shallower models evaluated in Question 4. While a single hidden layer is theoretically sufficient to approximate any continuous function (universal approximation theorem), deeper models can be more **parameter-efficient** and better at learning **compositional feature representations**. Using a deeper architecture with ReLU is a common practice in modern neural network design, and we seek to determine whether such a structure provides a meaningful performance gain on this BMI prediction task.

3) Findings:

   - **Test RMSE (Deep ReLU Network): 6.0327**

   This result was slightly better than the linear and one-layer ReLU models in Question 4, but not significantly lower than the best-performing model (sigmoid) from Question 4.

```
[Task 5] Best RMSE: 6.032730790429399
```

4) Interpretation:

The deeper FFN model (using ReLU) achieved a slightly lower RMSE of 6.03 compared to the ReLU-activated shallow network from Question 4 (RMSE = 6.06), indicating a marginal improvement in prediction accuracy. However, it is important to note that the best-performing model overall was actually the Task 4 MLP with sigmoid activation, which achieved an even lower RMSE of 5.98. This suggests that increasing model depth with ReLU activation does not necessarily yield better performance in this context. These findings imply that BMI prediction in this dataset may not benefit significantly from deeper architectures, and that careful selection of activation functions in shallow networks can be more impactful than simply increasing model complexity. The small improvement (~0.03) over the shallow ReLU model and inferior performance relative to the sigmoid model suggest that deeper FFNs with ReLU may add unnecessary complexity without clear benefit. Overall, the results reinforce that in small, tabular datasets, model simplicity paired with well-chosen nonlinearities can be just as effective, if not better, than deeper models.

Extra Credits

(a) I used mutual information to assess the relevance of each feature for predicting diabetes status. Mutual information quantifies the amount of shared information between a feature and the target variable, capturing both linear and nonlinear dependencies. The results showed that features such as **HighBP**, **GeneralHealth**, and **HighChol** had the highest mutual information scores, indicating they are most predictive of diabetes. On the other hand, features like **Zodiac, HasHealthcare,** and **MentalHealth** had very low scores, suggesting little to no relationship with diabetes. This ranking helps us understand which variables are most valuable for model building, and which can be potentially removed in simplified models.

```
Extra Credit (a) - Feature Importance via Mutual Information:
                Feature  Mutual Information
                 HighBP            0.055045
          GeneralHealth            0.050957
               HighChol            0.042060
      HardToClimbStairs            0.033558
                    BMI            0.028509
           PhysActivity            0.028306
             Myocardial            0.027691
                 Smoker            0.023763
          BiologicalSex            0.022835
                  Fruit            0.022279
         PhysicalHealth            0.021356
             AgeBracket            0.021170
          IncomeBracket            0.019225
       EducationBracket            0.017935
             Vegetables            0.016275
NotAbleToAffordDoctor            0.013761
                 Stroke            0.012585
           HeavyDrinker            0.009698
           MentalHealth            0.009553
          HasHealthcare            0.007328
                 Zodiac            0.003578
```

(b) Neural networks, especially multi-layer perceptrons (MLPs), provide a flexible framework for modeling nonlinear relationships. In this assignment, MLPs achieved solid performance in both classification (AUC $\approx$ 0.833) and regression (RMSE $\approx$ 5.98). However, their benefits over classical methods are not always substantial.

Classical models such as logistic regression, random forests, and boosting methods performed nearly as well, while being simpler, faster to train, and easier to interpret. These models proved to be strong baselines, particularly effective on structured tabular data like this.

In the end, while neural networks offer more modeling power, the additional complexity may not always be justified. For datasets like this one, classical methods remain efficient, interpretable, and highly competitive alternatives.