Homework 6

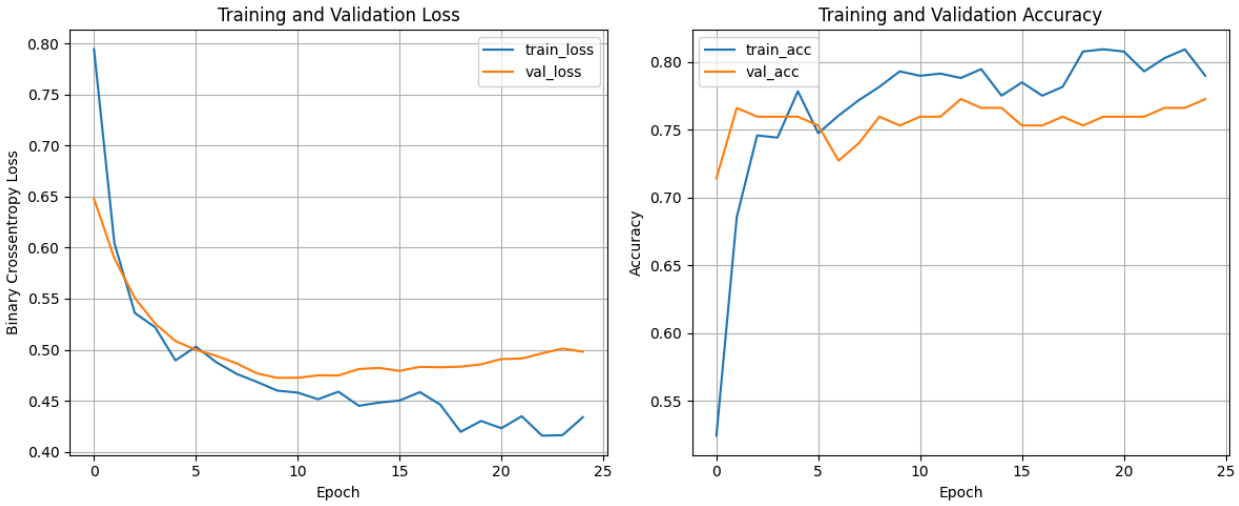
Sophia Godfrey

801149485

Github Link: https://github.com/QueenSophiaLo/Intro-To-ML/tree/main/Homework%206

Problem 1

Using the diabetes.csv file featured in previous homework assignments, three machine learning models (a Deep Neural Network (DNN), Logistic Regression (LR), and Support Vector Machine (SVM)) will be implemented and their results compared. The dataset was divided into an 80% training set and a 20% validation set. The split was performed using stratified sampling to ensure that the ratio of positive (diabetic) to negative (non-diabetic) outcomes was consistent across both partitions. All input features were Standardized using sklearn.preprocessing.StandardScaler. The scaler was fitted only on the training data to prevent data leakage and then used to transform both the training and validation sets. The model was compiled using the Adam optimizer and binary_crossentropy loss. Training was regulated using Early Stopping with patience=15 on the validation loss to restore the best weights and prevent overfitting.



The performance of all classification algorithms (Neural Network, Logistic Regression, and SVM) was evaluated using four metrics.

1. **Accuracy** provides the general measure of model correctness by considering the number of sucessful diabetes diagnoses.
2. **Precision** measures the model's exactness, indicating how often a patient predicted to have diabetes is actually diabetic, thereby minimizing false alarms (False Positives).
3. **Recall** (or Sensitivity) measures the ability of a model to identify all relevant instances within a dataset. High Recall is important when classifying medical ailments (like diabetes in this scenario) even if there are some false positives. This is becuase a false negative (failing to identify diabetes) can have severe consequences for patients.
4. **F1 Score** is used to provide a single, balanced measure of a model's robustness, and is the average of Precision and Recall.

```
Summary comparison (validation set):
           accuracy  precision    recall        f1
NeuralNet  0.753247   0.700000  0.518519  0.595745
Logistic   0.714286   0.608696  0.518519  0.560000
SVM        0.753247   0.660000  0.611111  0.634615
```

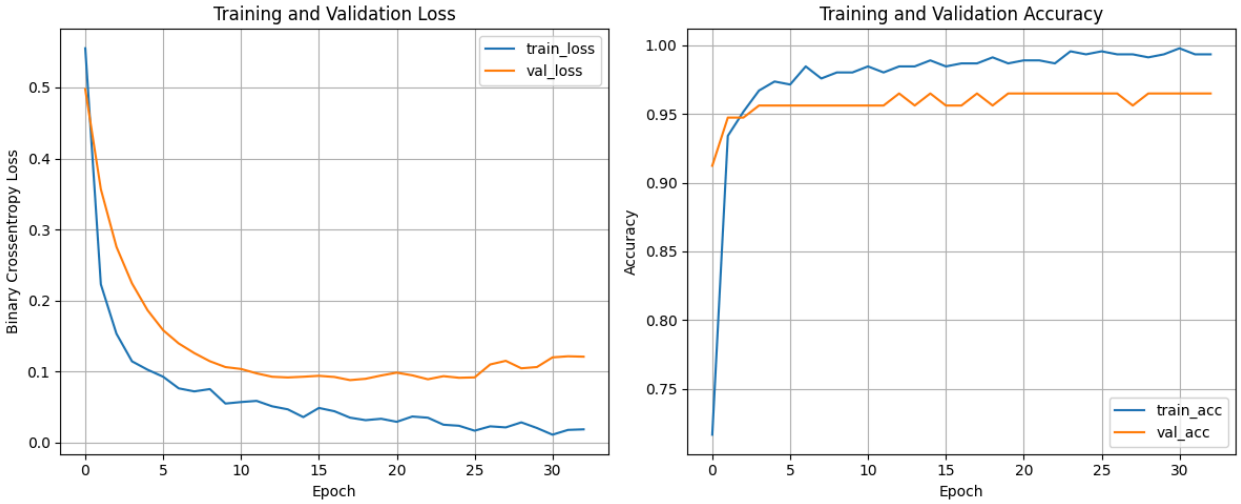
```
--- Model Comparison: Raw Confusion Matrix Component Counts ---
                        TN (True Negative)  FP (False Positive)  \
Model
Neural Network (Dense)                  88                   12
Logistic Regression                     82                   18
SVM (RBF)                               83                   17

                        FN (False Negative)  TP (True Positive)
Model
Neural Network (Dense)                   26                   28
Logistic Regression                      26                   28
SVM (RBF)                                21                   33
```

The Deep Neural Network (DNN) performed well in overall accuracy (0.7532) and achieved the highest Precision (0.7000). However, the DNN struggled significantly with Recall (0.5185), which resulted in 26 False Negatives (missed diagnoses). This suggests that while the DNN was careful about predicting the positive class (high precision), it was too conservative, missing nearly half of the actual diabetic patients. Because diagnostic safety requires minimizing False Negatives, Recall and F1 Score are the most important metrics. The Support Vector Machine (RBF) model proved to be the most reliable classifier, achieving the highest F1 Score (0.6346) and the highest

Recall (0.6111), with the lowest count of False Negatives (21). The Support Vector Machine gave the most clinically reliable performance by reducing the number of missed diabetic cases. These results show that on this relatively small dataset, a well-tuned classical model like SVM can still outperform a standard deep learning architecture.

Problem 2

Using the Wisconsin Breast Cancer (Diagnostic) dataset from Scikit-learn featured in previous homework assignments, three machine learning models (a Connected Neural Network (NN), Logistic Regression (LR), and a Support Vector Machine (SVM)) will be implemented and the results analyzed. The dataset, consisting of 569 instances and 30 numerical features derived from tumor cell nuclei characteristics, was used. All 30 features were standardized using StandardScaler, fitting only on the training data to prevent data leakage. The Fully Connected Neural Network was designed with three dense hidden layers, incorporating Batch Normalization and Dropout to improve stability and prevent overfitting.



The training log for the Fully Connected Neural Network (FCNN) reveals three phases: rapid convergence, stabilization, and the onset of overfitting. Initially, the model showed powerful learning, with the training loss plummeting by 90% (from 0.5551 to 0.0548) within the first ten epochs, and validation accuracy quickly stabilizing at approximately 0.9561. The model achieved its optimal generalization and minimum validation loss (0.0878) around Epoch 18. Following this peak, the gap between the rapidly decreasing training loss and the plateaued/rising validation loss widened, indicating the beginning of overfitting. The Early Stopping mechanism successfully monitored this trend, correctly halting training after Epoch 33 (15 epochs of non-improvement) and restoring

the model to the optimal weights achieved at Epoch 18, ensuring the final deployed model is robust and well-generalized with a validation accuracy of around 0.965.

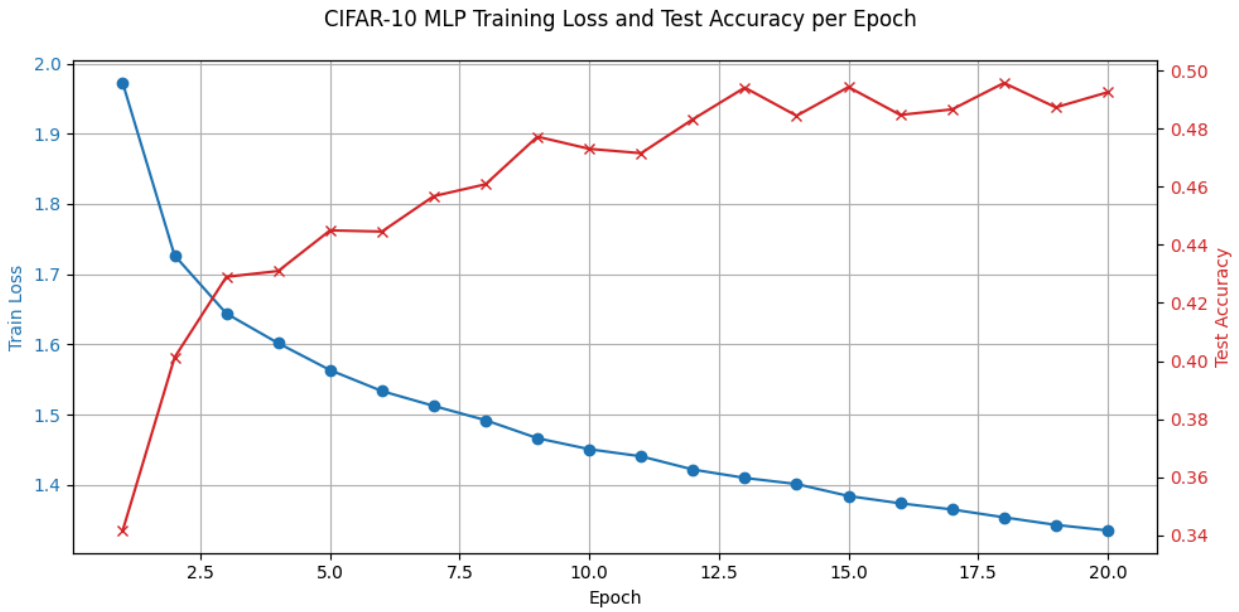
Summary comparison (validation set):

	accuracy	precision	recall	f1
NeuralNet	0.964912	0.972222	0.972222	0.972222
Logistic	0.982456	0.986111	0.986111	0.986111
SVM	0.982456	0.986111	0.986111	0.986111

The performance of all three models on the 20% validation set is summarized below. The target positive class is Benign. All three models performed well on the Cancer dataset, with Logistic Regression and SVM having the same high accuracy levels for accuracy, precision, recall, and f1 score. The Neural Network had the worst performance out of the models, but still had high levels of accuracy, precision, recall, and f1 score.

Problem 3 Part a

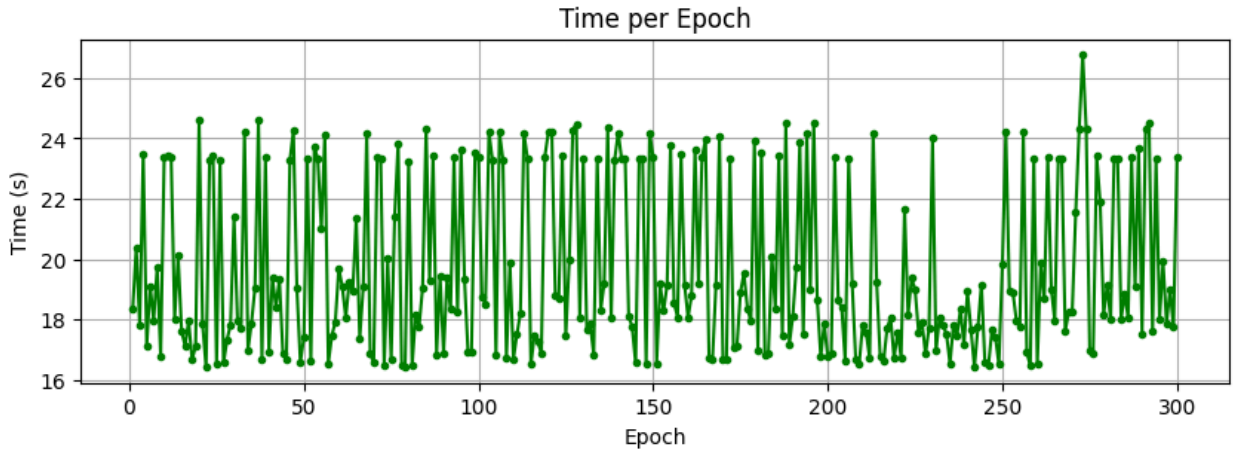
A minimal Fully Connected Neural Network (FCNN, or Multilayer Perceptron, MLP) was implemented and trained on the 10-class image classification task on the CIFAR-10 dataset. The objective was to analyze the performance of a shallow network architecture on a complex visual recognition task. The training time was consistent, averaging around 2.40 seconds per epoch on the 50,000 training images. The network's learning was slow, stabilizing with a relatively high final loss and low accuracy (refer to the full log for epoch-by-epoch values).



The training results for the single-hidden-layer Fully Connected Neural Network (FCNN) on the CIFAR-10 dataset shown in the chart above clearly demonstrate the architectural limitations of using an MLP for complex image recognition. While the training loss (blue line) showed rapid initial descent and continued to decrease consistently over 20 epochs, successfully minimizing error on the training data, the test accuracy (red line) rapidly stalled after initial gains. Accuracy quickly rose to approximately 44% but then severely plateaued, fluctuating around the 48%-50% mark and reaching a final value of approximately 50%. This significant divergence shows overfitting of the model.

Problem 3 Part b

This experiment extends the previous implementation by converting the shallow Multilayer Perceptron (MLP) into a deeper network with three hidden layers, trained over 300 epochs on the CIFAR-10 dataset. The objective was to determine if increased model capacity (depth and total parameters) could overcome the fundamental limitations of using an MLP for image classification. The training was conducted for 300 epochs. The time per epoch remained highly consistent, typically fluctuating between 17 seconds and 24 seconds, with an average time of approximately 20 seconds per epoch.



The model was evaluated after completing all 300 epochs of training. The training log indicates that the deeper network likely experienced significant overfitting. This can be seen by looking at the difference between the training loss and the test accuracy. The training loss (blue line) decreased very well and consistently throughout all 300 epochs, reaching a very low final value of about 0.15. This suggests the network got very good at fitting the training data. However, the test accuracy (red line) quickly rose to around 0.52 in the first 50 epochs but then mostly stopped improving and began to fluctuate slightly around the 50% mark. The growing separation between the very low training loss and the stagnant test accuracy is a common sign of overfitting: the model is effectively memorizing the training set without improving its ability to generalize to new images.