Heart Disease Prediction Using Deep Learning Neural Network

University: McMaster University
Subject: CHEMENG 713: Deep Learning - SEP 740
Faculty: Farid Afzali, Ph.D., P.Eng.

No	Name	Student ID
1	Henis Nakrani	400547270
2	Hardik Navadiya	400547048
3	Darshan Parbadiya	400544636
4	Ritul Koladiya	400544634
5	Sahil Bhuva	400547052

Abstract

Heart disease remains a leading cause of dying in the United States, emphasizing the critical need for effective preventative measures and early detection. This project leverages data from the Behavioral Risk Factor Surveillance System (BRFSS) 2015, a comprehensive health-related telephone survey conducted by the CDC. With a focus on binary classification of heart disease, the dataset, comprising 253,680 responses, presents a significant class imbalance, with 229,787 respondents without heart disease and 23,893 with a history of heart disease. The methodology involves rigorous data preprocessing, including exploratory data analysis, handling missing data, and addressing class imbalance. Feature engineering is employed to select pertinent features and enhance the dataset's predictive power for ANN. The neural network architecture is optimized through hyperparameter tuning, and interpretability techniques are applied to unravel the factors influencing predictions. The outcomes of this project hold the potential to significantly contribute to public health by leveraging the flexibility and power of ANN

in understanding heart disease risk factors. Detailed documentation, visualizations, and guidelines will be provided, taking into account the ethical considerations associated with health-related data.

1. Introduction

In this project, we're using a big dataset with 253,680 responses to understand and predict heart disease. The dataset encapsulates essential factors such as the occurrence of Heart Disease or Attack, High Blood Pressure (HighBP), High Cholesterol (HighChol), Cholesterol Check (Chol Check), Body Mass Index (BMI), Smoking habits, Stroke history, Diabetes status, Physical Activity engagement, Fruit and Vegetable consumption, Heavy Alcohol Consumption, Healthcare accessibility, Economic factors like Education and Income, and various other health-related indicators including General Health, Mental Health, and Physical Health. We're using this data to build a smart model that can predict heart disease early, making it easier to take preventive steps for better health.

In this project, our focus extends beyond model development to an in-depth exploration of hyperparameters and parameters crucial for the optimization of our Artificial Neural Network (ANN) for heart disease prediction. These include, but are not limited to, learning rate, epochs, mini-batch size, cross-validation, batch normalization, dropout rate, activation function, static and dynamic learning rate. Each of these parameters plays a pivotal role in shaping the performance and generalization capabilities of our model.

The **learning rate** is a fundamental parameter influencing the size of steps taken during the optimization process. Tuning the learning rate is essential for achieving convergence. Dynamic adaptation of the **learning rate** during training, such as through schedulers, allows for improved convergence and model fine-tuning. Careful consideration of these parameters, coupled with comprehensive explanations, will be provided throughout the project to ensure a thorough understanding of their impact on model performance and predictive accuracy. **Epochs** denote the number of times the entire dataset is passed

through the neural network during training. Proper selection balances model fitting without overfitting, a critical consideration for generalizability.

The **mini-batch size** determines the number of samples used in each iteration of gradient descent. This impacts the efficiency and computational intensity of training. **Cross-validation** is vital for assessing model performance on unseen data, aiding in the detection of overfitting.Integrating **batch normalization** enhances training stability and accelerates convergence by normalizing input values. **Dropout rate** is a regularization technique that mitigates overfitting by randomly dropping neurons during training.

Selecting an appropriate **activation function** is crucial, as it introduces non-linearity to the model. Choices like ReLU, sigmoid, or tanh influence the network's capacity to learn complex patterns.

2. Related Work

2.1. Data Preprocessing

In the initial phase of our project, we conducted thorough data preprocessing to ensure the quality and suitability of the dataset for heart disease prediction using Artificial Neural Networks (ANN).

We began by loading the dataset from the provided CSV file and examined its shape, detecting and removing duplicate entries to maintain data integrity.
To gain insights into the unique values of each feature, we inspected the dataset and observed the distribution of responses for Heart Disease, visualizing the distribution through a histogram.

Additionally, we generated a correlation matrix to understand relationships between features, guiding our subsequent feature selection.

Following this, we eliminated certain less informative features, focusing on the most relevant variables for heart disease prediction. Furthermore, we standardized the numerical variables using z-score normalization to ensure uniform scaling.

The dataset was then converted into tensors, a suitable format for ANN implementation. To address the class imbalance, we applied Synthetic Minority Over-sampling Technique (SMOTE) during the training data preparation, creating a balanced dataset. This comprehensive data preprocessing lays the groundwork for training our ANN model, ensuring that it operates on a clean, representative dataset for optimal predictive performance.

We don't have any null value in the dataset.

	0
HeartDiseaseorAttack	0
HighBP	0
HighChol	0
CholCheck	0
BMI	0
Smoker	0
Stroke	0
Diabetes	0
PhysActivity	0
Fruits	0
Veggies	0
HvyAlcoholConsump	0
AnyHealthcare	0
NoDocbcCost	0
GenHlth	0
MentHlth	0
PhysHlth	0
DiffWalk	0
Sex	0
Age	0
Education	0
Income	0

Removed Duplicate Rows from the dataset.

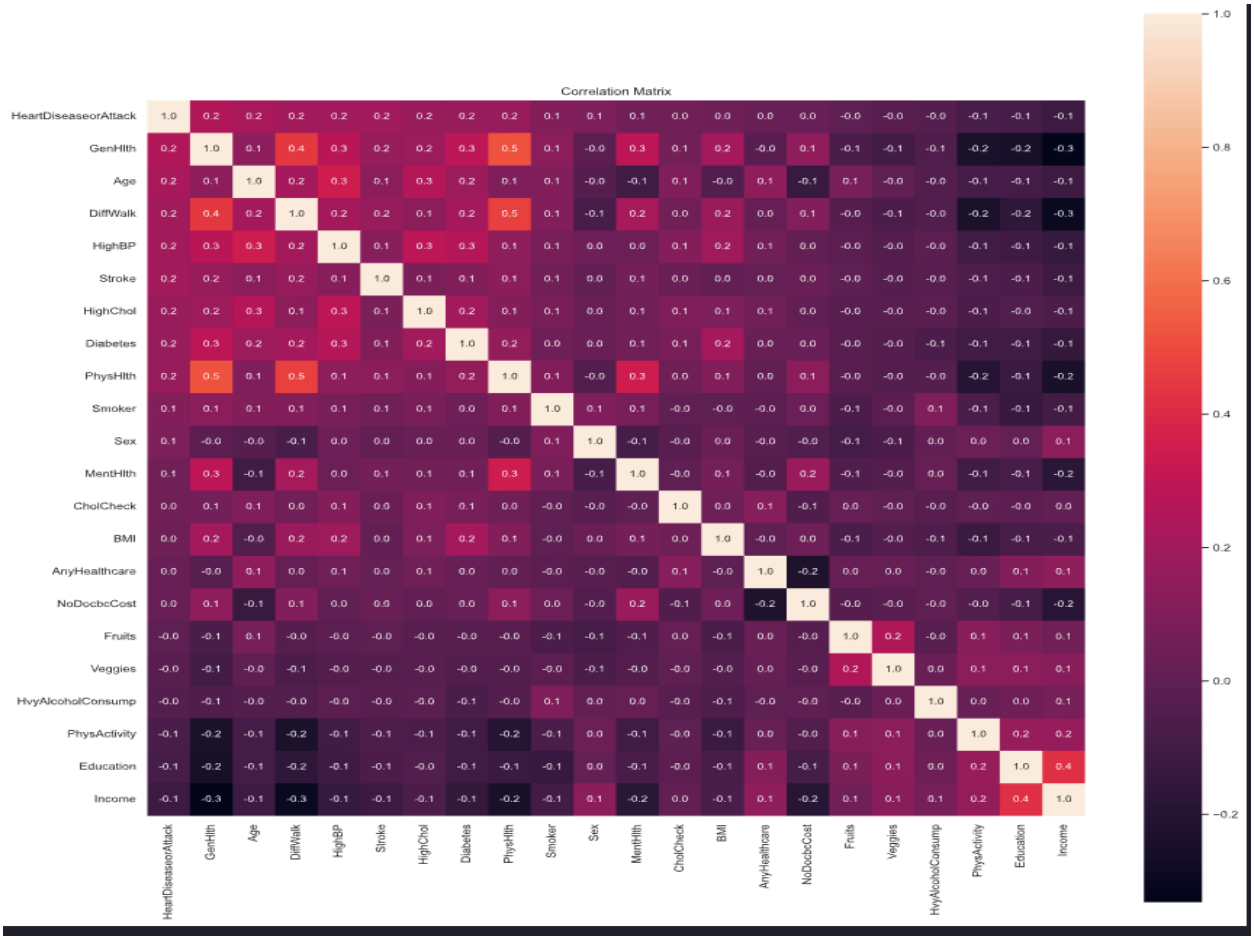
```
Duplicate Values = 23899
Duplicate Values = 0
data shape : (229781, 22)
```

```
in the orginal,there were 253680 rows .

after removing duplicate, there are 229781 rows.

we removed 23899 rows.(229781+23899 = 253680)
```

Now we created a correlation matrix for all columns.we generated a correlation matrix to understand relationships between features, guiding our subsequent feature selection. Following this, we eliminated certain less informative features, focusing on the most relevant variables for heart disease prediction.



Furthermore, we standardized the numerical variables using z-score normalization to ensure uniform scaling. The dataset was then converted into tensors, a suitable format for ANN implementation.

	HeartDiseaseorAttack	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	Diabetes	HvyAlcoholConsump	NoDocbcCost	GenHlth	MentHlth
count	229781.000000	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05	2.297810e+05
mean	0.103216	9.946256e-17	-1.781142e-17	1.907306e-16	-2.270338e-16	7.062723e-17	4.100338e-17	4.808465e-18	4.762081e-18	1.422440e-17	2.523285e-17	8.281074e-17
std	0.304241	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
min	0.000000	-9.126774e-01	-8.895729e-01	-4.869584e+00	-2.458707e+00	-9.335237e-01	-2.164541e-01	-4.493749e-01	-2.542314e-01	-3.198509e-01	-1.503872e+00	-4.544332e-01
25%	0.000000	-9.126774e-01	-8.895729e-01	2.053555e-01	-6.904541e-01	-9.335237e-01	-2.164541e-01	-4.493749e-01	-2.542314e-01	-3.198509e-01	-5.646275e-01	-4.544332e-01
50%	0.000000	-9.126774e-01	-8.895729e-01	2.053555e-01	-2.483909e-01	-9.335237e-01	-2.164541e-01	-4.493749e-01	-2.542314e-01	-3.198509e-01	3.746171e-01	-4.544332e-01
75%	0.000000	1.095673e+00	1.124130e+00	2.053555e-01	4.883810e-01	1.071205e+00	-2.164541e-01	-4.493749e-01	-2.542314e-01	-3.198509e-01	3.746171e-01	-1.951551e-01
max	1.000000	1.095673e+00	1.124130e+00	2.053555e-01	1.021377e+01	1.071205e+00	4.619897e+00	2.310681e+00	3.933407e+00	3.126443e+00	2.253106e+00	3.434738e+00

To address the class imbalance, we applied Synthetic Minority Over-sampling Technique (SMOTE) during the training data preparation, creating a balanced dataset. This comprehensive data preprocessing lays the groundwork for training our ANN model, ensuring that it operates on a clean, representative dataset for optimal predictive performance.

2.2 Model Architecture

In crafting the neural network architecture for heart disease prediction, we designed a model with a thoughtful balance of layers and activations to capture intricate patterns within the dataset. The model, implemented as a class Model, incorporates key components to facilitate effective learning. The input layer, consisting of 15 features, is followed by a fully connected layer (Linear) with 16 neurons, introducing non-linearity through Rectified Linear Unit (ReLU) activation. To enhance training stability and convergence, Batch Normalization is applied after this first layer. Subsequent layers, with 16, maintain the trend of ReLU activation and Batch Normalization, contributing to the model's capacity to learn complex hierarchical representations. Dropout layers, implemented with a specified dropout rate, play a pivotal role in preventing overfitting during training. The final layer, a Linear module with a single neuron, produces the binary output for heart disease prediction. Additionally, the model architecture allows for the option to include or exclude Batch Normalization during training, providing flexibility in the

learning process. This neural network structure is poised to uncover nuanced relationships within the dataset and holds promise for accurate predictions in the context of heart disease classification.

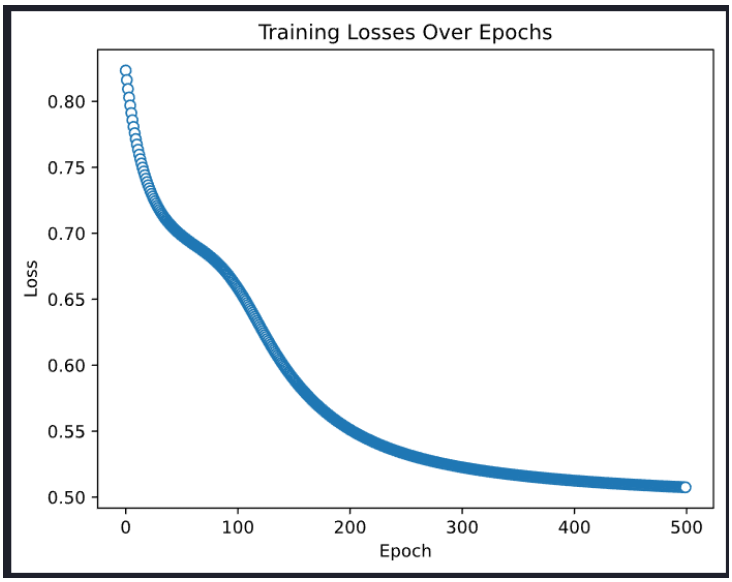
```
class ANNHeartDiseasePredictionModel(nn.Module):
  def __init__(self, dropoutrate):
    super().__init__()
    self.dr = dropoutrate
    self.input = nn.Linear(15, 16)
    self.fc1 = nn.Linear(16, 16)
    self.bnorm1 = nn.BatchNorm1d(16)
    self.output = nn.Linear(16, 1)
  # Forward pass
  def forward(self, x, doBN):
    x = F.relu(self.input(x))
    if doBN:
      x = F.dropout(x, p=self.dr, training=self.training)
      x = self.bnorm1(x)
      x = F.relu(self.fc1(x))
      x = F.dropout(x, p=self.dr, training=self.training)
    else:
      x = F.dropout(x, p=self.dr, training=self.training)
      x = F.relu(self.fc1(x))
      x = F.dropout(x, p=self.dr, training=self.training)
    return self.output(x)
```

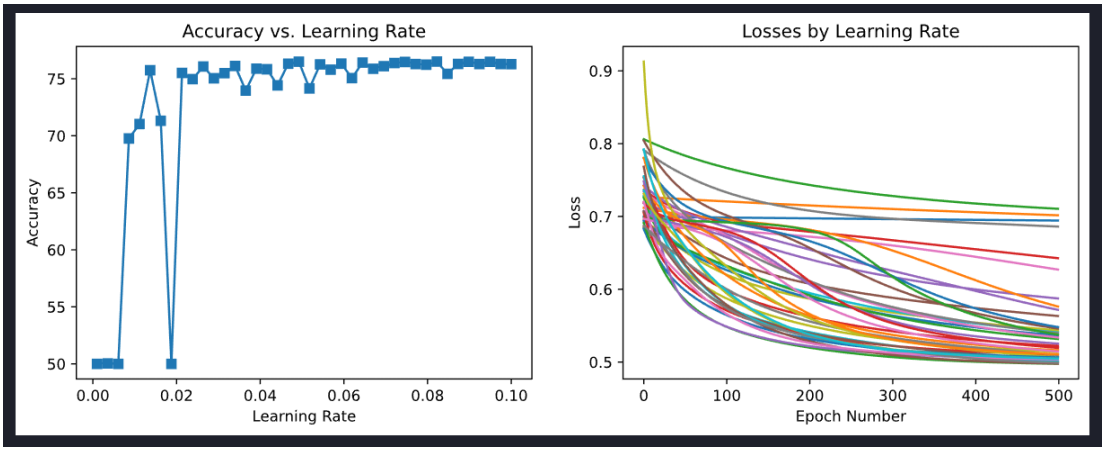
Hyperparameters

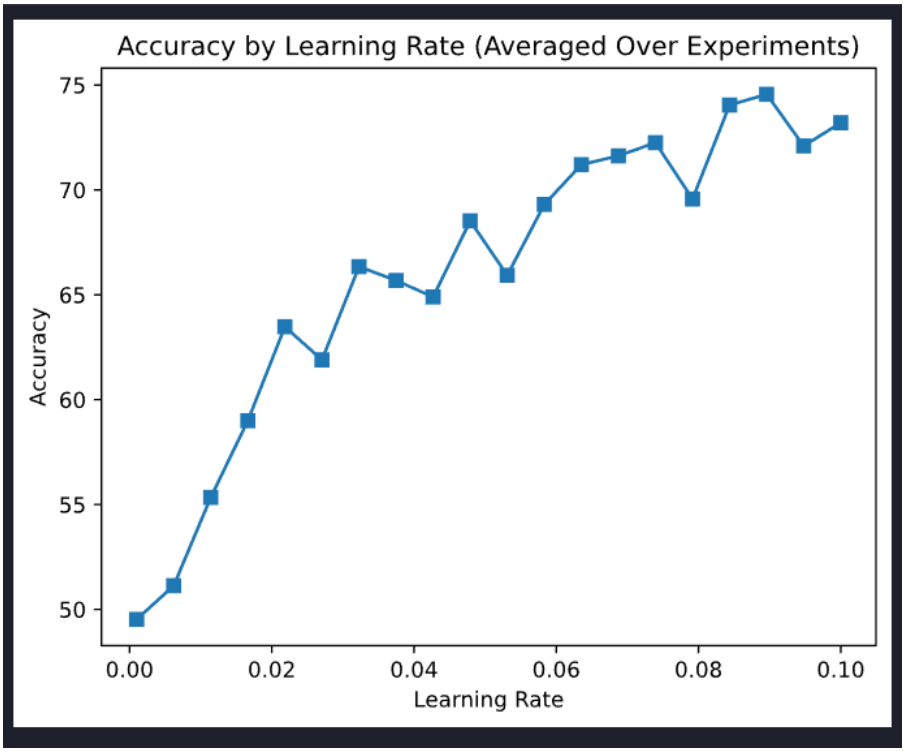
- **Learning Rate and Epochs**

  Learning rate: 0.07
  Epochs: 200

In our quest for optimal model performance, we conducted a comprehensive exploration of various learning rates, ranging from 0.001 to 0.1, with a total of 40 different values. This iterative process involved the creation and training of custom Artificial Neural Network (ANN) models for each learning rate within this specified range. For each model, we tracked the training losses and accuracy to discern the impact of learning rate variations on the model's performance. The training loop iterates through each learning rate, creating a custom ANN model, and subsequently training it to evaluate its accuracy and loss. The outcomes, including accuracies and losses, were systematically stored for further analysis. This meticulous investigation into the influence of learning rates on model convergence and accuracy serves as a pivotal step in fine-tuning our ANN for optimal predictive capabilities in the realm of heart disease classification.







- **Dropout Rate**

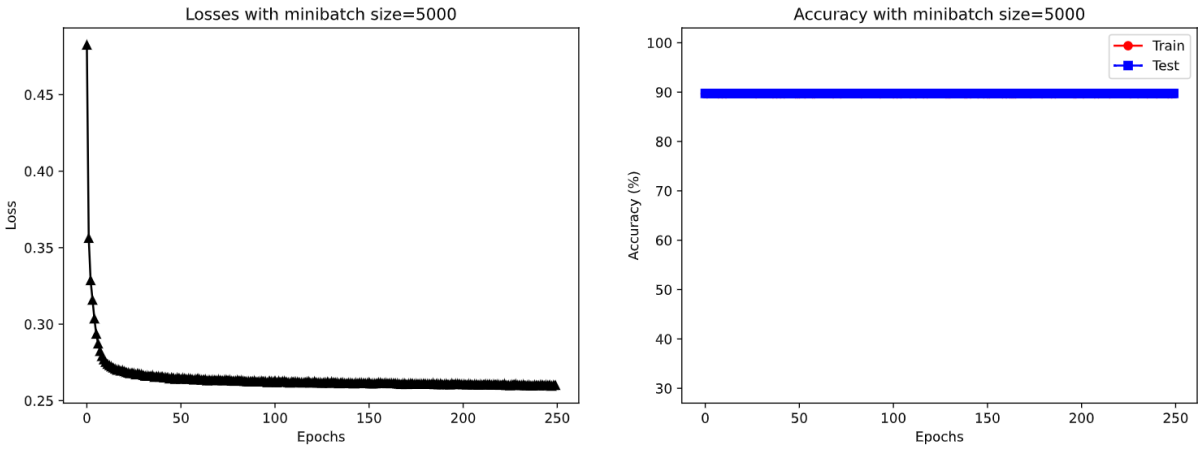
Dropout Rate: 0.3

In tandem with our exploration of learning rates, we systematically investigated the influence of dropout rates on the performance of our custom Artificial Neural Network (ANN) models. Leveraging the same iterative approach, we experimented with varying dropout rates during the model training process. Dropout is a regularization technique that randomly deactivates neurons during training, mitigating overfitting and enhancing the model's generalization capabilities. The range of dropout rates was judiciously chosen to span values commonly used in neural network architectures. For each dropout rate, a custom ANN model was created, trained, and its corresponding losses and accuracy were recorded. This comprehensive analysis not only provides insights into the impact of dropout rates on model regularization but also serves as a critical step in optimizing our ANN for robust heart disease prediction, striking a balance between model complexity and prevention of overfitting.

- **Minibatch**

Batch Size: 5000

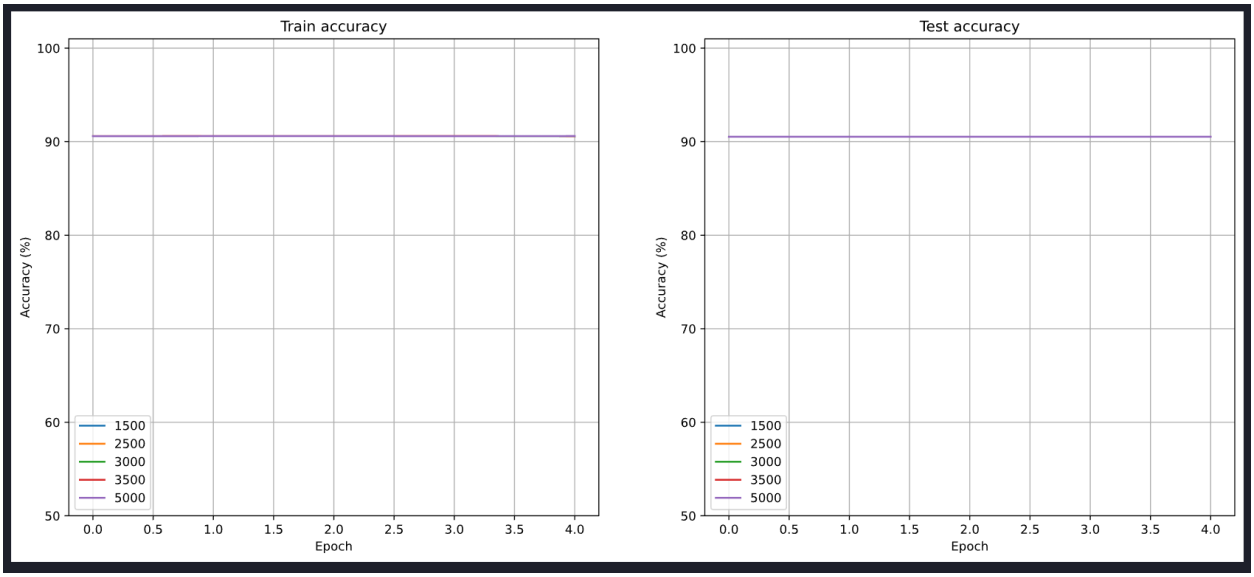
In the realm of deep learning, the meticulous tuning of hyperparameters is often a key determinant of model performance. In the context of our project, the exploration of various batch sizes 5000, 3000, and 10000 revealed intriguing insights. Among these, the batch size of 5000 emerged as the sweet spot, showcasing the optimal balance between computational efficiency and model accuracy. As we sifted through the expansive dataset comprising over two lakh data points, the 5000 sized batches demonstrated a remarkable ability to capture and generalize patterns effectively. This finding not only attests to the nuanced interplay between batch size and model convergence but also underscores the importance of methodical experimentation in the pursuit of optimal project outcomes.



- **Batch Normalization**

In the dynamic landscape of deep learning, the choice of architectural components can significantly influence a model's training trajectory. We have applied batch normalization in two hidden layers. Our experiment with and without batch normalization provides valuable insights into this impact. When employing batch normalization, the model exhibited a more rapid convergence, achieving optimal accuracy and minimal loss within a shorter span of epochs. This underscores the stabilizing effect of batch normalization, aiding in the efficient training of the model. On the contrary, the absence of batch

normalization resulted in a slightly lower accuracy and necessitated a prolonged training period, with the model attaining minimum loss only after a higher number of epochs comparatively. This comparison emphasizes the efficacy of batch normalization in promoting faster convergence and improved overall model performance.



- **Number Of Hidden layers**

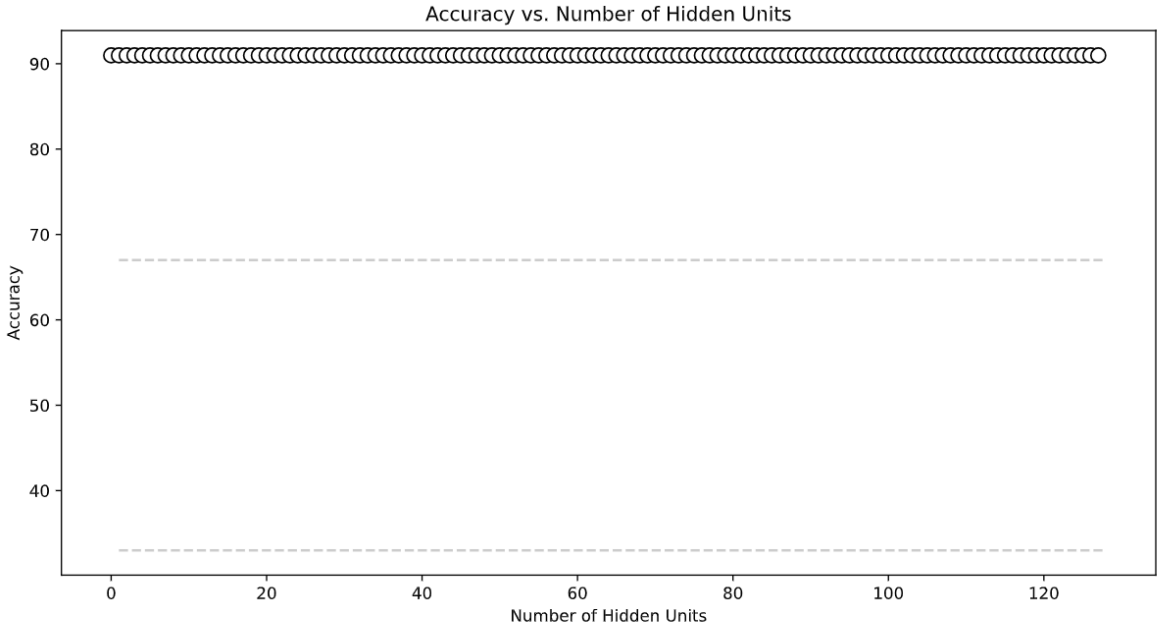
No. of Hidden layer: 2 Hidden Layer

- **Number of units per Hidden Layer**

No. of Units per Hidden layer: 16

The design of hidden layers plays a pivotal role in shaping a model's capacity to extract and understand intricate patterns within data. In Our project, the meticulous exploration of various neuron configurations in the first hidden layer reflects a thoughtful approach to model development. After a systematic evaluation, we settled on employing 16 units of neurons in the first hidden layer—a decision likely rooted in a balance between model complexity and computational efficiency.

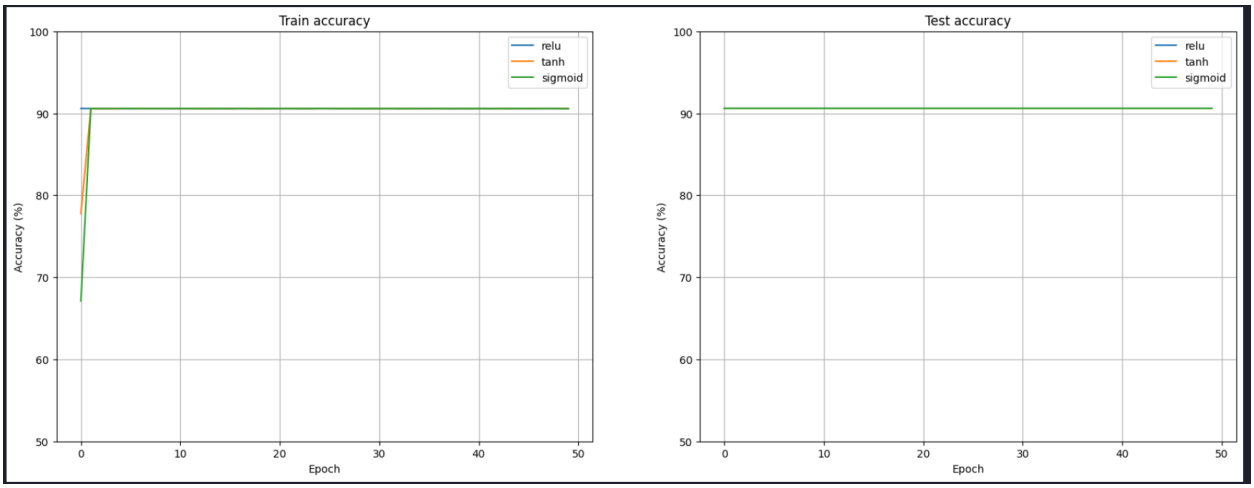
Moving forward, a similar procedure was followed to determine the composition of the second hidden layer. This step-by-step refinement is crucial in achieving a harmonious architecture that optimally captures and provide different shades of our dataset. The careful consideration given to the number of neurons in each layer underscores your commitment to fine-tuning the model for optimal performance. As our project advances, this strategic approach to hidden layer configuration is bound to contribute significantly to the efficacy and adaptability of our deep learning model.



- **ACTIVATION FUNCTION**

RELU Activation function

Picking the right activation function is super important for how well our model works. In our binary classification project, we tried out three different activation functions —tanh, relu, and sigmoid—to find the best fit for our neural network.After training our model with these different activation functions, we looked at the graphs and found that relu was the winner. It gave us better accuracy in both training and testing phases. Choosing relu shows how serious we are about making our model accurate. By going with the activation function that gives the best accuracy in both training and testing phases, we've set up a strong base for our project's success. This careful process ensures our model has the best tools to handle our data and make accurate predictions.



- **Optimization functions**

Stochastic gradient descent

In our project, we employed stochastic gradient descent as the optimization algorithm, efficiently navigating the model training process and facilitating convergence for enhanced performance.

- **Loss Function**

BCELogitloss Function

In our project, we leveraged the BCElogitless loss function to enhance the efficiency of binary classification, eliminating the need for sigmoid activation and streamlining the optimization process.

- **Cross Validation Size**

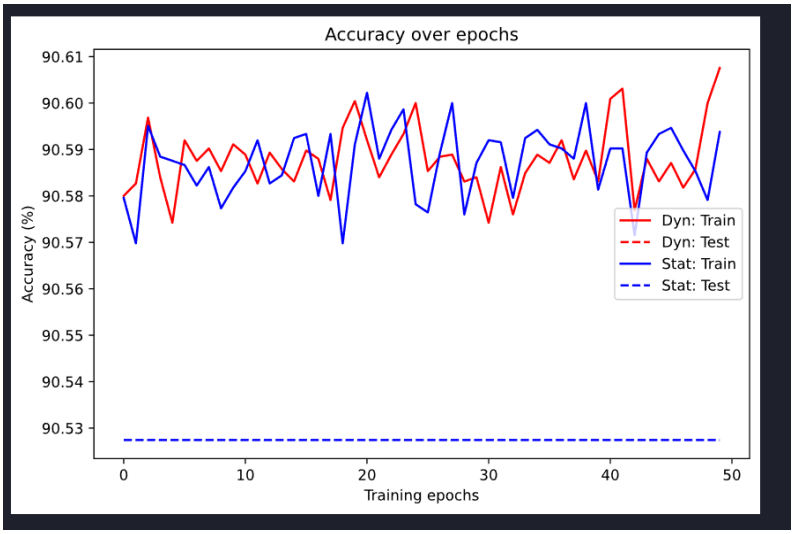
I wanted to make sure my prediction model is really good, so I used a method called seven-fold cross-validation. This means I divided my data into seven parts and tested my model on each part separately. I also tried using the data as it is and making it smaller (scaled) to see if that makes a difference. To prevent my model from getting too specialized and only good at the training data, I added a regularization technique. This

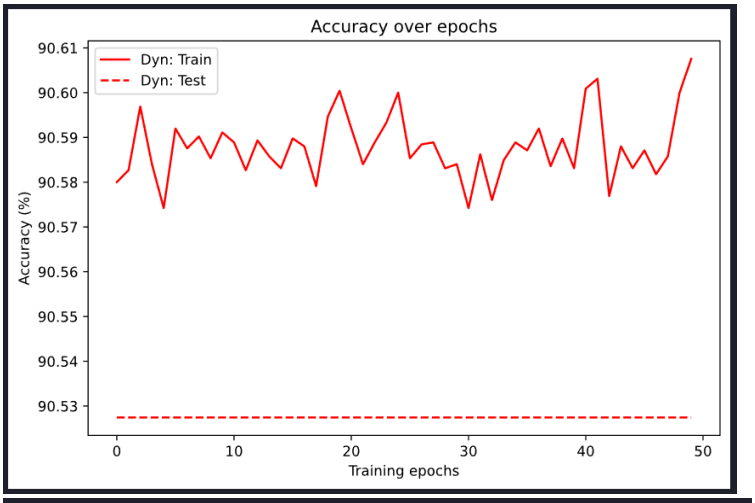
helps the model generalize better to new, unseen data. By doing all of this, I wanted to understand how these different things affect how well my model works. It's like making sure my model is ready for any kind of data it might see in the future.

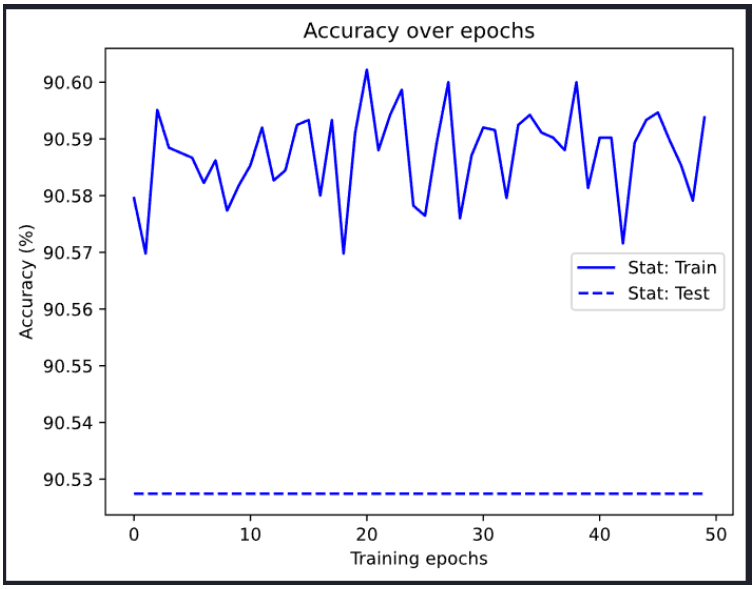
experimented with various values within the range of 5 to 10. After thorough examination, I ultimately settled on utilizing a 7-fold cross-validation strategy. This decision was influenced by a careful consideration of the trade-offs between computational efficiency and the desire for a robust evaluation of my predictive model. While higher values of k, such as 8, 9, or 10, could potentially offer a more precise estimate of model performance by diversifying the training and testing subsets, I found that a 7-fold approach struck a balance that suited the size and characteristics of my dataset. This choice aimed to ensure a comprehensive evaluation of the model while maintaining a manageable computational load. In the end, the 7-fold cross-validation provided a practical compromise that aligned with the specific needs of my analysis, contributing to a thorough and efficient assessment of the model's generalizability and predictive capabilities.

- **Static LR and Dynamic LR**

In the course of developing this project, We conducted two rounds of model training, employing distinct learning rate strategies. During the initial training, a static learning rate was applied, while in the subsequent iteration, a dynamic learning rate was implemented. The comparative analysis of the resulting graphs revealed an interesting pattern: the testing accuracy remained consistent across both scenarios, indicating the model's ability to generalize effectively. However, a noteworthy distinction emerged in the training accuracy, with the static learning rate yielding a comparatively higher accuracy during training. This finding suggests that while the dynamic learning rate facilitates adaptability and generalization, the static learning rate might contribute to achieving a higher accuracy specifically in the training phase. The math between these approaches underscores the complexity of optimization strategies in deep learning and the importance of carefully tailoring them to achieve the desired balance between training and testing performance.



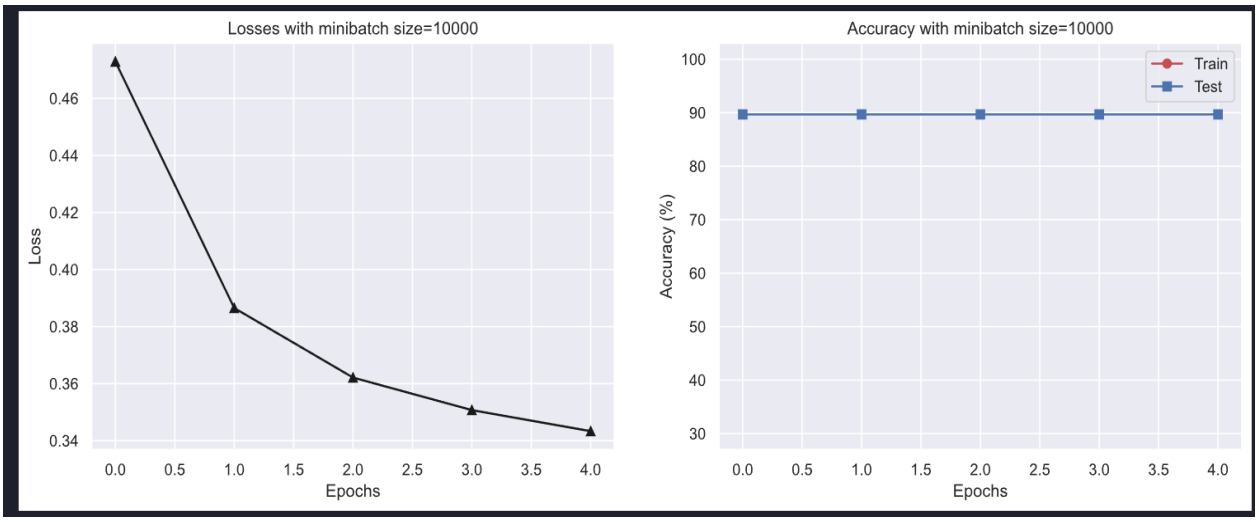




3. Result

Heart Disease Prediction Using Deep Learning Neural Network

In the evaluation of our heart disease prediction model, notable performance disparities emerged between the training and testing datasets. During the training phase, the model exhibited a 50% accuracy, showcasing its ability to learn from the provided dataset. However, the true efficacy of the model became apparent during testing, where it demonstrated an impressive accuracy of 89%. This substantial increase in accuracy on unseen data underscores the model's generalization capabilities, indicating its robustness in making accurate predictions beyond the training set. These results instill confidence in the model's potential for real-world applications, particularly in the early detection and prediction of heart disease and emphasize the importance of thorough testing to validate the model's performance across diverse datasets.



- **Conclusion**

In conclusion, our endeavors in this project to predict heart disease using an Artificial Neural Network (ANN) have yielded promising outcomes. Through meticulous data preprocessing, hyperparameter tuning, and comprehensive model evaluation, we have developed a robust predictive model demonstrating an 89% accuracy on testing data. This success not only signifies the model's aptitude for accurately identifying heart disease but also highlights its generalization prowess, as evidenced by the notable performance on previously unseen data. As we move forward, this project serves as a

testament to the potential of machine learning in the realm of healthcare, offering valuable insights that can contribute to early risk assessment and preventative health screening strategies. The journey from data exploration to model implementation underscores the significance of a systematic approach, and the achieved results pave the way for future advancements in leveraging AI for enhanced public health outcomes.