Assignment 3 – Report

NEERAJA VUDHANTHI

Task 1 – Regression:

For this regression task, the dataset named "space_ga" from OpenML has been used. It contains 3,107 observations on U.S. County votes cast in the 1980 presidential election. This dataset contains the total number of votes cast in the 1980 presidential election per county (VOTES), the population in each county of 18 years of age or older (POP), the population in each county with a 12th grade or higher education (EDUCATION), the number of owner-occupied housing units (HOUSES), the aggregate income (INCOME), the X spatial coordinate of the county (XCOORD), and the Y spatial coordinate of the county (YCOORD). The VOTES is the target for this dataset and the rest are considered as features.

Models' description:

Model 1:

- Architecture: Sequential neural network with one input layer, one hidden layer with 10 neurons activated by the sigmoid function, and one output layer.

- Training: Compiled using the Adam optimizer with mean squared error (MSE) as the loss function. Trained for 100 epochs with validation data for monitoring performance.

Model 2:

- Architecture: Sequential neural network with one input layer, one hidden layer with 10 neurons activated by the rectified linear unit (ReLU) function, and one output layer.

- Training: Compiled using the Adam optimizer with MSE as the loss function. Trained for 100 epochs with validation data for monitoring performance.

Model 3:

- Architecture: Sequential neural network with one input layer, one hidden layer with 12 neurons activated by the sigmoid function, followed by another hidden layer with 10 neurons activated by the sigmoid function, and one output layer.

- Training: Compiled using the Adam optimizer with MSE as the loss function. Trained for 100 epochs with validation data for monitoring performance.

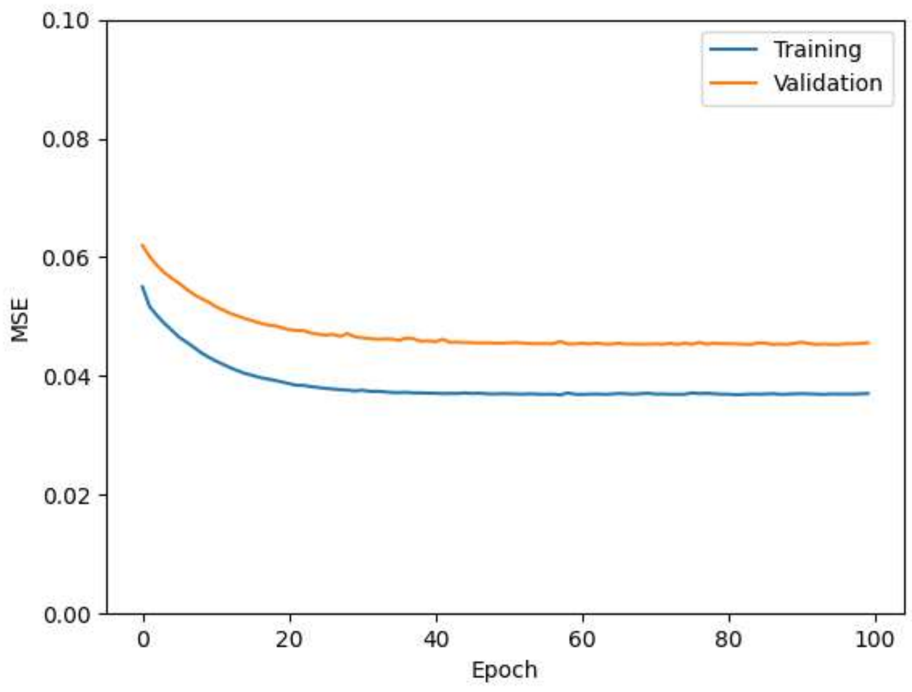
Model 4:

- Architecture: Sequential neural network with one input layer, one hidden layer with 10 neurons activated by the sigmoid function, and one output layer.

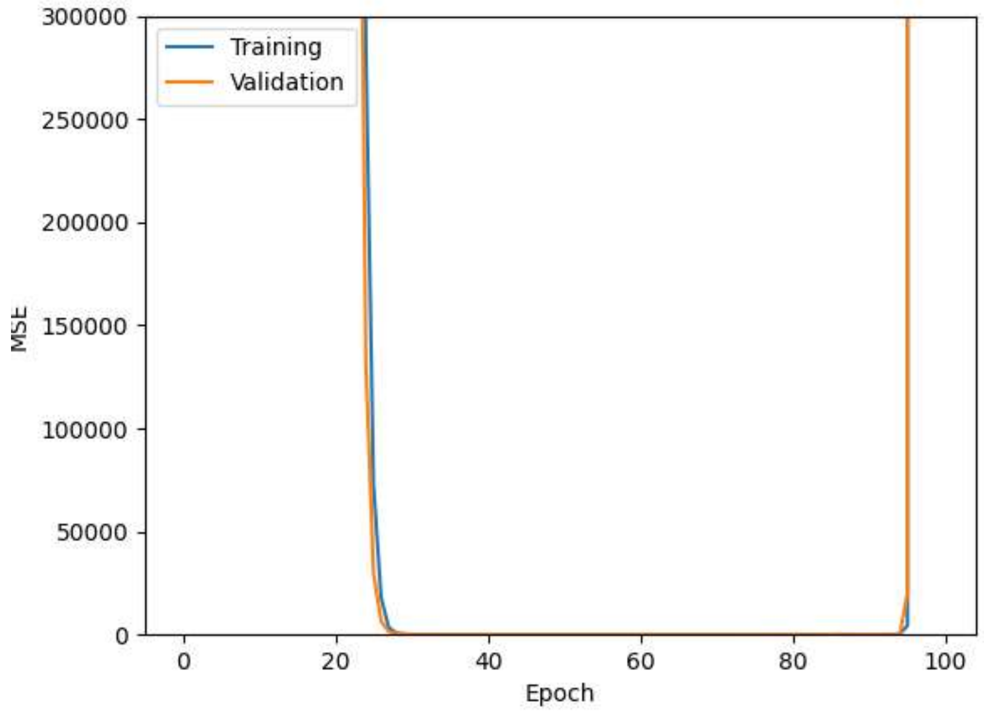
- Training: Compiled using stochastic gradient descent (SGD) optimizer with MSE as the loss function. Trained for 100 epochs with validation data for monitoring performance.

Graph Training errors and validation errors V/S the number of epochs for each model:

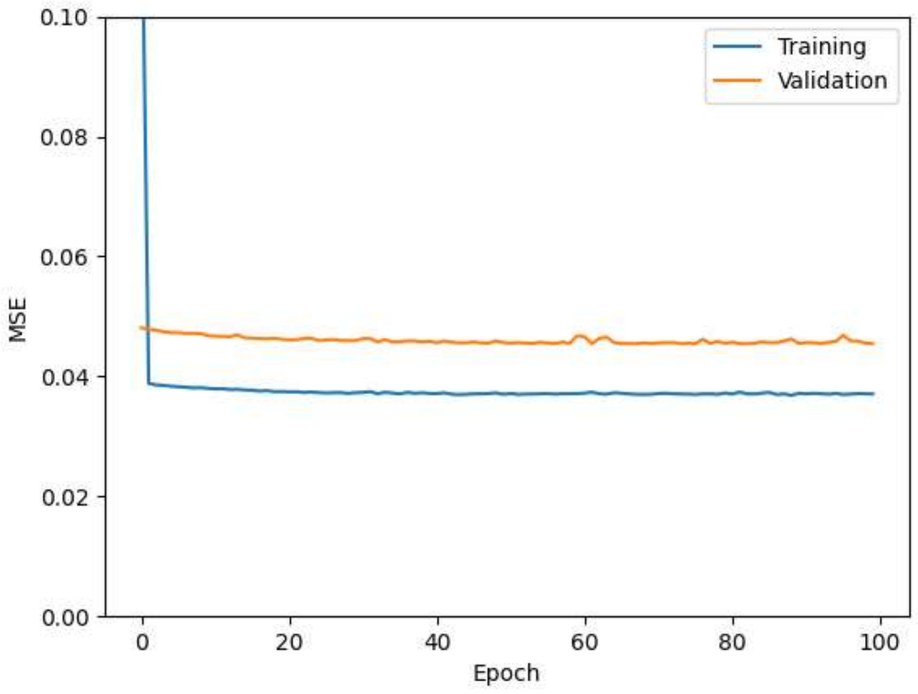
Graph for Model - 1:



Graph for Model - 2:



Graph for Model – 3:



Graph for Model – 4:

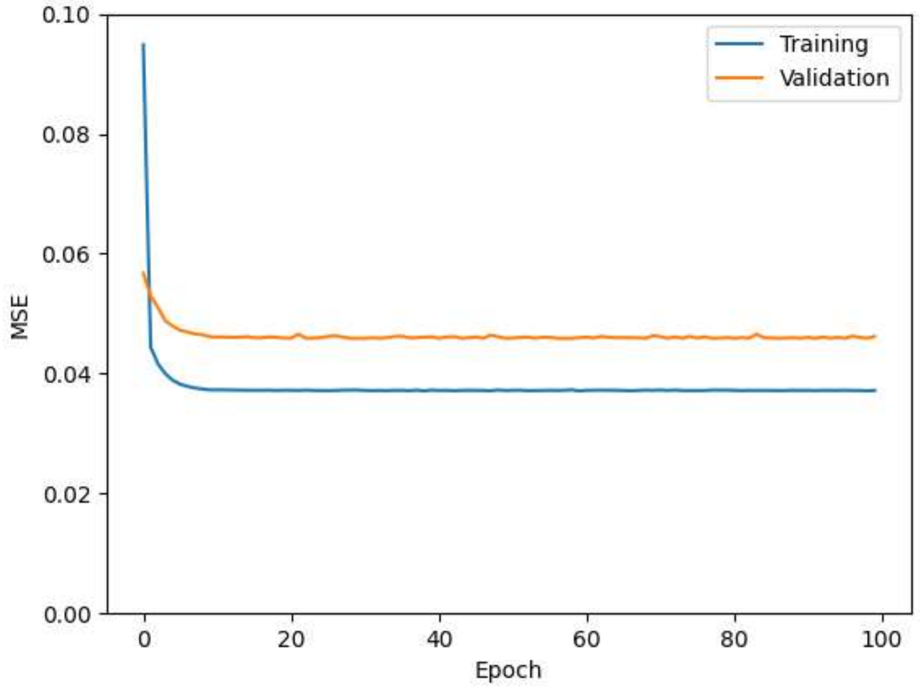


Table of minimum validation error for each model:

Model	Minimum validation error
Model – 1	0.045547496527433395
Model – 2	1.861162543296814
Model – 3	0.045900605618953705
Model - 4	0.046721767634153366

After analyzing the results of the models based on their minimum validation error, it's evident that Model 1 and Model 3 achieved the lowest validation errors, with values of 0.0455 and 0.0459 respectively. These models, utilizing sigmoid activation functions and varying numbers of neurons in the hidden layers, demonstrated robust performance in predicting the target variable.

Model 2, employing rectified linear unit (ReLU) activation function in the hidden layer, exhibited a notably higher minimum validation error of 1.8611 compared to Models 1 and 3. This suggests that the choice of activation function may have impacted the model's ability to learn and generalize from the training data.

On the other hand, Model 4, utilizing sigmoid activation functions like Model 1 but trained with Stochastic Gradient Descent (SGD) optimizer, also yielded a relatively low minimum validation error of 0.0467. While this model's performance is comparable to Model 1, the fluctuating nature of its learning process, as observed in the training visualization, may indicate less stable convergence during training.

Overall, Models 1 and 3 stand out as the most effective in terms of achieving the lowest validation errors. Their consistent performance across different architectures suggests that the choice of activation function and network depth played significant roles in model performance.

Task 2 – Classification:

For this regression task, the dataset named “quake” with ID: 772 from OpenML has been used. The earthquake dataset available on OpenML contains information about seismic activity recorded by seismographs. The dataset provides insights into the characteristics of seismic events, aiding in the analysis of earthquake patterns and trends.

Features:

- Latitude: Geographic coordinate specifying the north-south position of the earthquake's epicenter.

- Longitude: Geographic coordinate specifying the east-west position of the earthquake's epicenter.

- Focal - Depth: The depth at which the earthquake occurred below the Earth's surface.

Target Variable:

- Seismic Event Classification: Whether the earthquake is minor or major

Data Source:

The earthquake dataset sources its data from reputable seismic monitoring networks and agencies worldwide, including the United States Geological Survey (USGS), European-Mediterranean Seismological Centre (EMSC), and others.

Models’ description:

Model 1:

- Architecture: Sequential neural network with one input layer, one hidden layer with 20 neurons activated by the sigmoid function, and one output layer with 2 neurons activated by the softmax function.

- Training Procedure: Compiled using the Adam optimizer with categorical cross-entropy as the loss function. Trained for 100 epochs with validation data for monitoring accuracy.

Model 2:

- Architecture: Sequential neural network with one input layer, one hidden layer with 20 neurons activated by the rectified linear unit (ReLU) function, and one output layer with 2 neurons activated by the softmax function.

- Training Procedure: Compiled using the Adam optimizer with categorical cross-entropy as the loss function. Trained for 100 epochs with validation data for monitoring accuracy.

Model 3:

- Architecture: Sequential neural network with one input layer, one hidden layer with 20 neurons activated by the sigmoid function, followed by another hidden layer with 10 neurons activated by the sigmoid function, and one output layer with 2 neurons activated by the softmax function.

- Training Procedure: Compiled using the Adam optimizer with categorical cross-entropy as the loss function. Trained for 100 epochs with validation data for monitoring accuracy

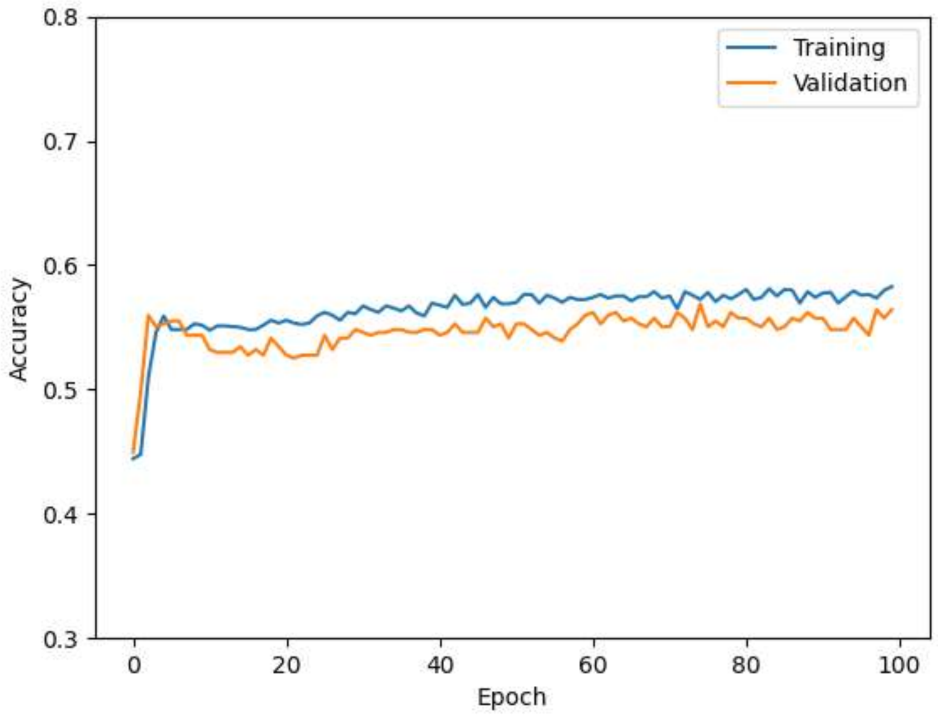
Model 4:

- Architecture: Sequential neural network with one input layer, one hidden layer with 20 neurons activated by the sigmoid function, and one output layer with 2 neurons activated by the softmax function.

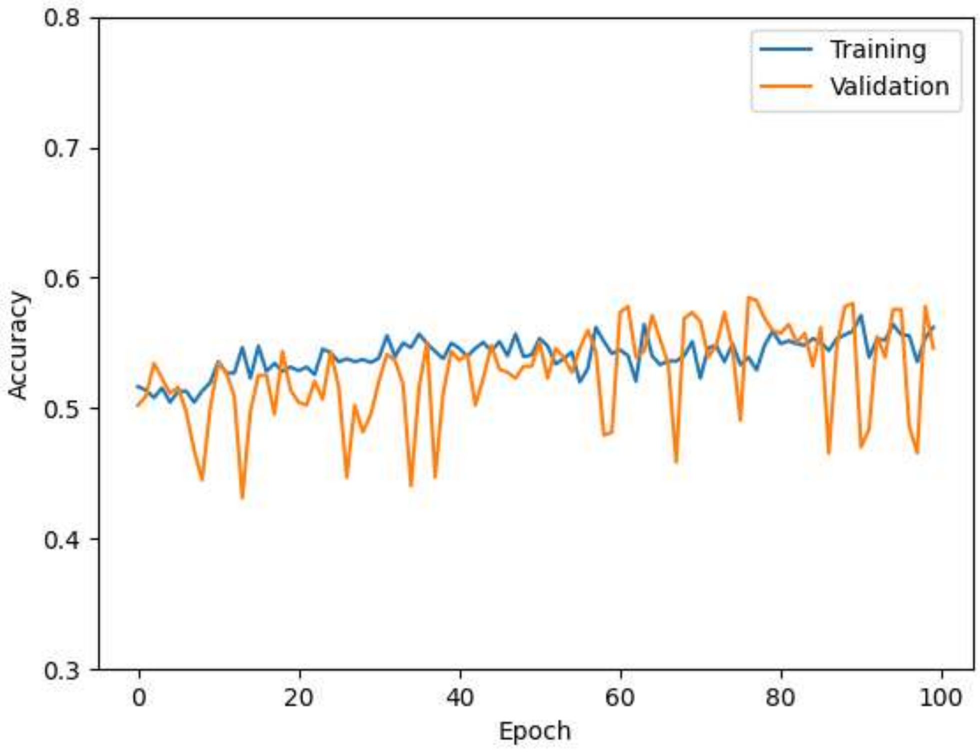
- Training Procedure: Compiled using stochastic gradient descent (SGD) optimizer with categorical cross-entropy as the loss function. Trained for 100 epochs with validation data for monitoring accuracy.

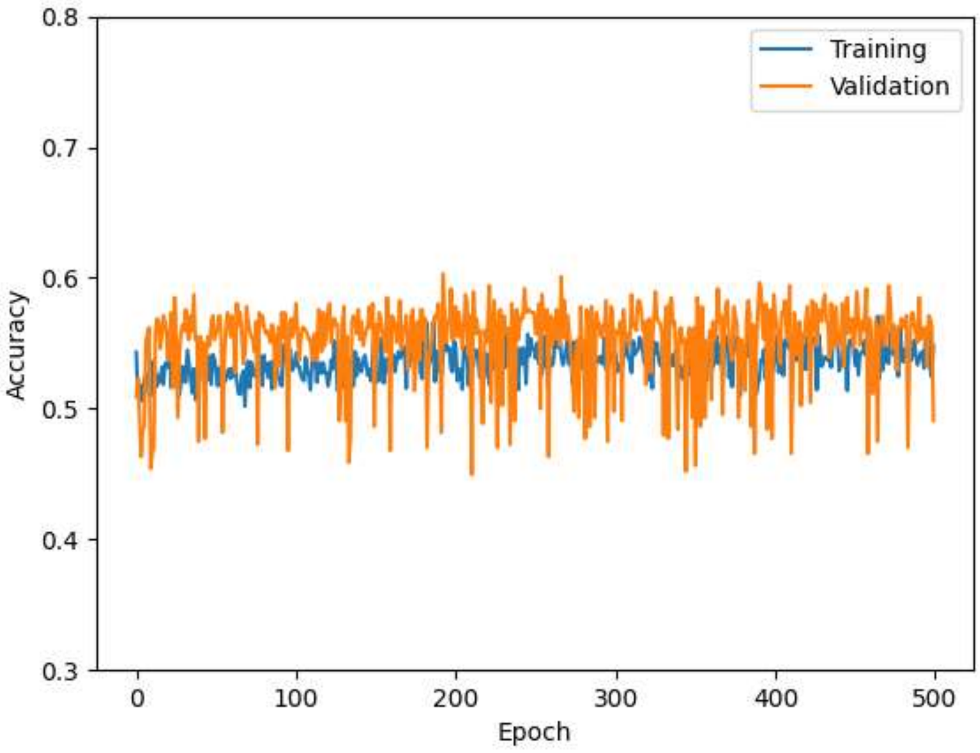
Graph Training accuracy and validation accuracy V/S the number of epochs for each model:

Graph for Model - 1:



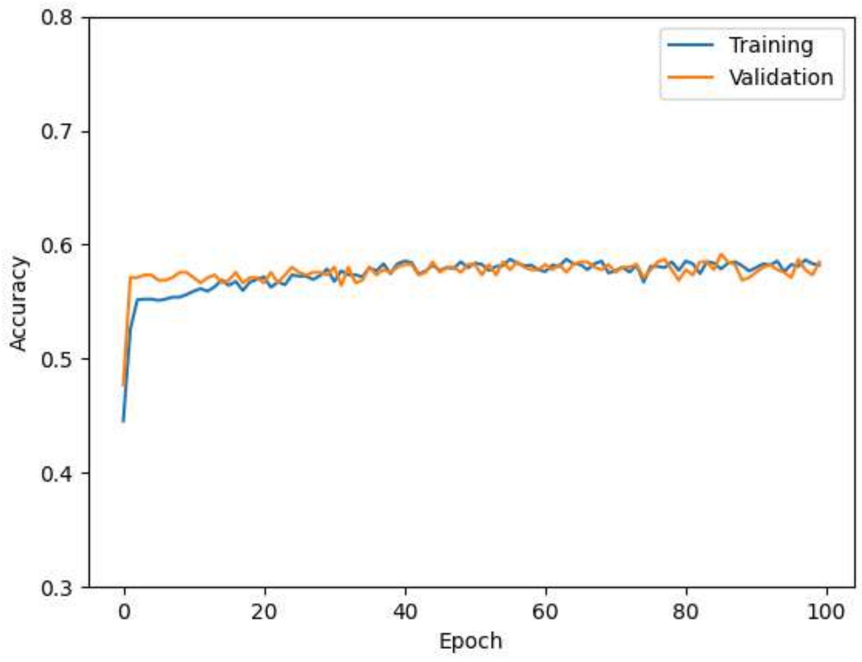
Graph for Model - 2:





Increasing the number of epoch did not help in plateauing the curve for model-2.

Graph for Model - 3:



Graph for Model - 4:

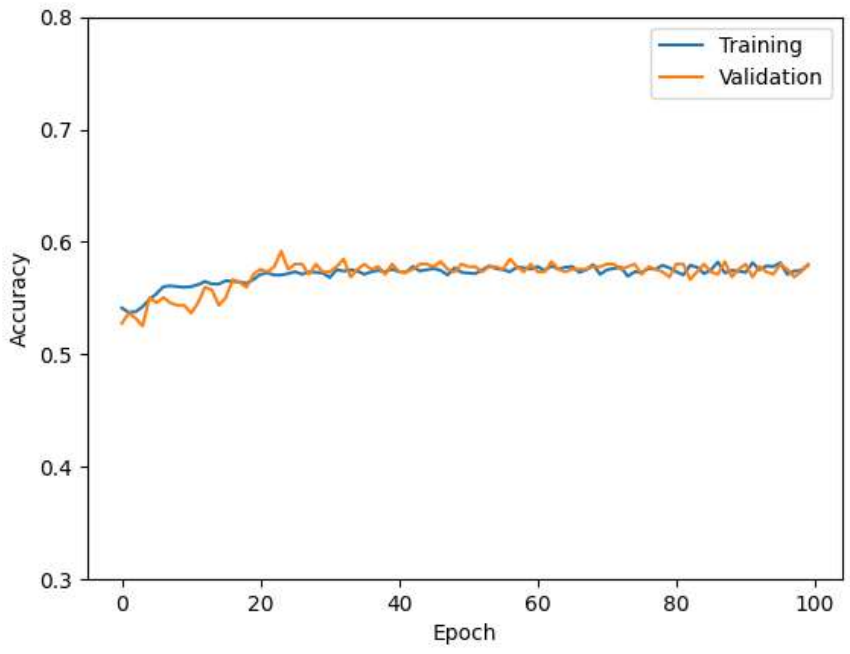


Table of maximum validation accuracy for each model:

Model	**Maximum validation accuracy**
Model – 1	0.5756880640983582
Model – 2	0.5963302850723267
Model – 3	0.5917431116104126
Model - 4	0.5688073635101318

After analyzing the results of the models based on their maximum validation accuracy, it's evident that Model 2 achieved the highest validation accuracy of 0.5963. This model used Rectified Linear Unit (ReLU) activation in the hidden layer, potentially allowing for faster convergence and better capturing of complex patterns in the data.

Similarly, Model 3 achieved a maximum validation accuracy of 0.5917. This model has an additional hidden layer compared to Model 1, aiming to capture more complex relationships within the data. Despite the additional layer, Model 3's performance remained comparable to Model 2, suggesting that the increased complexity did not significantly enhance classification accuracy in this context.

Model 1, made use of sigmoid activation functions throughout the network, achieved a maximum validation accuracy of 0.5757. While this model's performance is lower than Models 2 and 3, it still demonstrates a reasonable level of accuracy in classifying the data into the specified classes.

Lastly, Model 4, trained using Stochastic Gradient Descent (SGD) optimizer, achieved a maximum validation accuracy of 0.5688. Despite employing the same architecture as Model 1, the fluctuating nature of its training and validation accuracy suggests that the choice of optimization algorithm may have impacted its stability and convergence during training.

Overall, Models 2 and 3 stand out as the most effective in terms of achieving the highest validation accuracies. Their utilization of ReLU activation and additional hidden layers, respectively, contributed to their better performance in classifying the data.