CS461
PROGRAM 3
Alex Dowell

Data preparation:

In order to prepare the data for analysis, before I first read in the Cleveland Clinic data and separated the input parameters from the output (column 14), I removed the incomplete observations. I also converted the output variables into 1-hot encoded variables. This process involves creating a new binary column for each unique value in the original categorical column. For this example, the original column had 5 unique values (0-4), 5 new columns would be created with values of 0 or 1 indicating the presence or absence of each value in the original column. I selected this method because it allows the model to more easily understand the relationships between the different categories.

I employed standardization, StandardScalar() from the library Sklearn, on the numerical variables in the dataset. This involves subtracting the mean from each value and dividing by the standard deviation in order to scale the values to have a mean of 0 and a standard deviation of 1. I chose to use standardization because it can help the model converge more quickly during training and can also help prevent some features from dominating the model due to their larger scale.

Network configuration:

My network is set up as a fully connected feedforward neural network with 3 hidden layers. The first hidden layer has 26 neurons, the second has 13 neurons, and the third has 7 neurons. Using ReLu activation functions on the hidden layers and softmax on the output layer. The output layer has 5 neurons, as this is a binary classification for the one hot encoded variables. I did not try any nonlinear features, such as connecting outputs to inputs more than one layer ahead, as this is not necessary for the type of data I am working with.

Validation strategy:

I divided my data into training, test, and validation sets in a ratio of 80:10:10. I used k-fold cross validation with k=8 in order to further validate the model's performance. This involves dividing the data into k equal folds and training the model k times, each time using a different fold as the validation set and the remaining folds as the training set. The final performance is then calculated as the average performance across all k iterations.

I used early stopping as a method to lower the risk of overtraining. This involves monitoring the performance on the testing set and stopping the training process when the performance starts to deteriorate. This process set the epoch count to 15 epochs.

Results:

The model achieved an accuracy of 62% for the training data, 57% on the testing set, and 50% for the validation set. I found that adding additional layers and neurons

generally impacted the model's performance, while decreasing the learning rate had a negative effect on performance.

Comments:

Overall, I am satisfied with the results of the model. It was able to accurately predict the target variable in 30%ish cases. However, I think that further improvement could be made by trying different network architectures or implementing more advanced techniques, such as regularization or optimization algorithms. Also This dataset is particularly small. I believe that If the dataset was bigger then it was be easier to achieve a higher accuracy.

There were a few surprises along the way, especially with the data preparation. Examining the models performance highlights the importance of careful experimentation and tuning of hyperparameters in order to find the best model configuration.

References:

- Team, Keras. “Keras Documentation: The Model Class.” Keras, https://keras.io/api/models/model/.

Sample Output:

x_train.shape, y_train.shape, x_test.shape, y_test.shape, x_val.shape, y_val.shape

(237, 13) (237, 5) (30, 13) (30, 5) (30, 13) (30, 5)

Epoch 1/15

8/8 [==============================] - 4s 97ms/step - loss: 1.6495 - accuracy: 0.1772 - val_loss: 1.6560 - val_accuracy: 0.1667

Epoch 2/15

8/8 [==============================] - 0s 20ms/step - loss: 1.5889 - accuracy: 0.2447 - val_loss: 1.6017 - val_accuracy: 0.2333

Epoch 3/15

8/8 [==============================] - 0s 15ms/step - loss: 1.5356 - accuracy: 0.3418 - val_loss: 1.5580 - val_accuracy: 0.4000

Epoch 4/15

8/8 [==============================] - 0s 15ms/step - loss: 1.4858 - accuracy: 0.4304 - val_loss: 1.5186 - val_accuracy: 0.4333

Epoch 5/15

8/8 [==============================] - 0s 15ms/step - loss: 1.4383 - accuracy: 0.4979 - val_loss: 1.4768 - val_accuracy: 0.4667

Epoch 6/15

8/8 [==============================] - 0s 17ms/step - loss: 1.3874 - accuracy: 0.5570 - val_loss: 1.4338 - val_accuracy: 0.4667

Epoch 7/15

8/8 [==============================] - 0s 15ms/step - loss: 1.3372 - accuracy: 0.5781 - val_loss: 1.3846 - val_accuracy: 0.5333

Epoch 8/15

8/8 [==============================] - 0s 18ms/step - loss: 1.2815 - accuracy: 0.5949 - val_loss: 1.3332 - val_accuracy: 0.5333

Epoch 9/15

8/8 [==============================] - 0s 17ms/step - loss: 1.2183 - accuracy: 0.6076 - val_loss: 1.2769 - val_accuracy: 0.5667

Epoch 10/15

8/8 [==============================] - 0s 15ms/step - loss: 1.1626 - accuracy: 0.6034 - val_loss: 1.2197 - val_accuracy: 0.5667

Epoch 11/15

8/8 [==============================] - 0s 17ms/step - loss: 1.1067 - accuracy: 0.6076 - val_loss: 1.1666 - val_accuracy: 0.5667

Epoch 12/15

8/8 [==============================] - 0s 16ms/step - loss: 1.0564 - accuracy: 0.6118 - val_loss: 1.1162 - val_accuracy: 0.5667

Epoch 13/15

8/8 [==============================] - 0s 14ms/step - loss: 1.0161 - accuracy: 0.6160 - val_loss: 1.0720 - val_accuracy: 0.5667

Epoch 14/15

8/8 [==============================] - 0s 15ms/step - loss: 0.9832 - accuracy: 0.6245 - val_loss: 1.0352 - val_accuracy: 0.5667

Epoch 15/15

8/8 [==============================] - 0s 15ms/step - loss: 0.9577 - accuracy: 0.6245 - val_loss: 1.0022 - val_accuracy: 0.5667

1/1 [==============================] - 0s 327ms/step

Validation dataset accuracy percentage: 50.0

predicted:	actual:
0	0
3	3
0	1
0	4
0	2
0	2
0	0
0	0
3	0
0	1
0	0
0	0
0	2
3	0
0	0
0	0
0	0
3	0
0	0
0	2
0	3
0	2
0	0
3	1
0	0
3	3
3	1
0	0

3 0

0 0

0 0

0 1

0 0

0 3

0 1

0 0

0 0

0 0