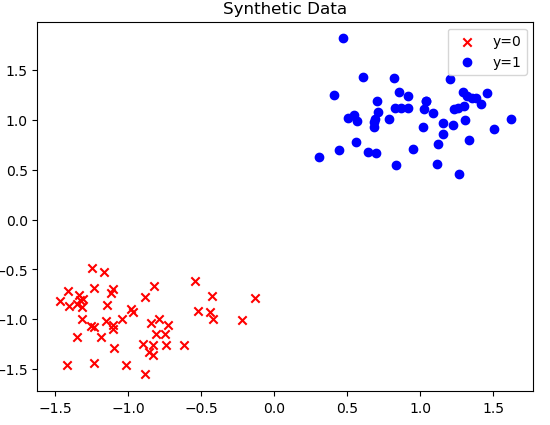
**Logistic Regression with TensorFlow - Report**

Jack Farah

**Option #2: Logistic Regression with TensorFlow**

In this project, it was requested to build a logistic regression model using tensorflow. I began this project by importing the necessary libraries, numpy, tensorflow, and matplotlib for visualization.

A computer code with text

Description automatically generatedA close-up of a computer screen

Description automatically generated I then synthetically created 100 data used for the logistic model. The zeros data was created to fit a gaussian distribution centered at (-1,1). Meanwhile, the ones data was created to also fit a gaussian distribution centered at (1,1). I then combined the zeros and ones dataset using numpys function vstack and concatenate. Finally, I cast the dataset into np.float32 in order for it fit tensorflows neural network properly, while systematically ensuring the model parameters are also float32 tensors. Next, I plotted the zeros and ones onto a scatter plot graph before the training of the model. The zeros are represented with a red ‘x’ and the ones are represented with a blue ‘o’. This graph is made just to visuzlize the data we are working with. A control for the experiment. Next, I began building the model used for the logistic regression model. This project provided the class of Logistic Regression model, leading me to use it for this project. Once the logistic regression class was created, I created the instance of the model.

A screen shot of a computer code

Description automatically generatedFollowing the creation of the Logistic regression class instance, the loss function, optimizer, and gradient decent was created. The loss function is used to measure the loss of each training session also known as epochs. The optimizer used was Adam with a learning rate of 0.01. The traiing loop was done for 20,000 iterations, measuring the loss at every 1000th iteration. This is A screenshot of a computer screen

Description automatically generateddone to make it easier to read the results of the training. The training takes advantage of the learning rate to optimize the gradient decend. The loss function was measured using the sigmoid cross entropy method for lositic regression, which was the most fitting loss function measurment for this project. After 20,000 iterations, the model has a loss function of 0.0036% which is an amazing result. The loss improved with every iteration. This was expected due to the nature of gradient decent. The constant improvement implies that the learning rate was at an accurate decission, and that the sigmoid cross entropy method for lositic regression was also the correct choice to measure the loss. The predictions in the end were very accurate and reliable.