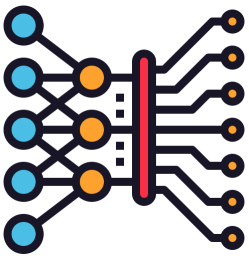
Assignment 1

Tuesday, September 17, 2024

Wayne Rudnick
Jeremy Middleman
Brian High
Andrei Phelps
Thomas Hynes



CS 491/591: Neural Networks

1 How to Run the Program

You can run the program by executing the main file using the python command.

```
python main.py
```

2 Code Organization

We structured our project into two files: main and perception. The main file is responsible for generating the training and test datasets, training the model, and adjusting model parameters. It also uses Matplotlib to visualize the training and testing results. The Perception file contains the Perception class and its member functions. After running the code, the command line displays the number of misclassified samples.

Note: Case 3 was not plotted because it is 4 dimensional data and not part of the assignment instructions.

3 Methodology and Model Evaluation

We implemented both the perceptron learning algorithm and gradient descent variants of the perceptron based on the material presented in lecture. To evaluate the performance of these two models under the conditions specified in the assignment, we conducted multiple tests, adjusting the learning rate and the number of epochs.

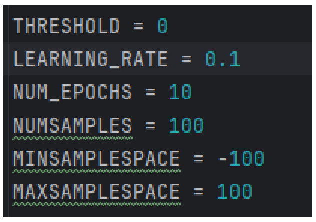
Additionally, we decided to plot both the training and test data to evaluate how well our model's parameters were tuned for the test cases. This comparison allowed us to assess the model's performance on the training data versus the test data. According to the Perceptron Convergence Theorem, a perceptron will perfectly classify a linearly separable dataset given sufficient epochs and the use of the perceptron learning algorithm. Therefore, if we observed the perceptron failing to perfectly classify the training data, it indicated that parameter adjustments were necessary.

4 Perceptron

4.1 Perceptron Learning Algorithm

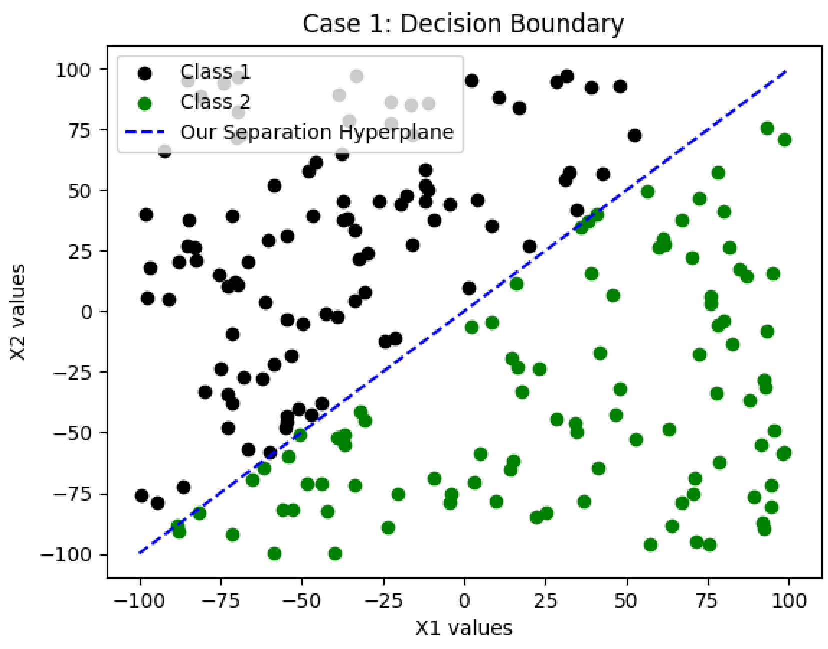
For parts 1-3 of the assignment, we chose to run our test with 10 epochs at a learning rate of 0.1. Due to the relatively simple nature of this test case, we wanted to see how the model performed with parameters that we thought were conservative. Even with this relatively low number of epochs, the perceptron effectively classified the data. Our post-training perceptron model correctly classified all test samples for Case 1 and missed 5 test samples for case 2. This near 100% accuracy in both cases is likely because the data is completely linearly separable. Had the data been less linearly separable, we expect that the model would not have been as accurate.

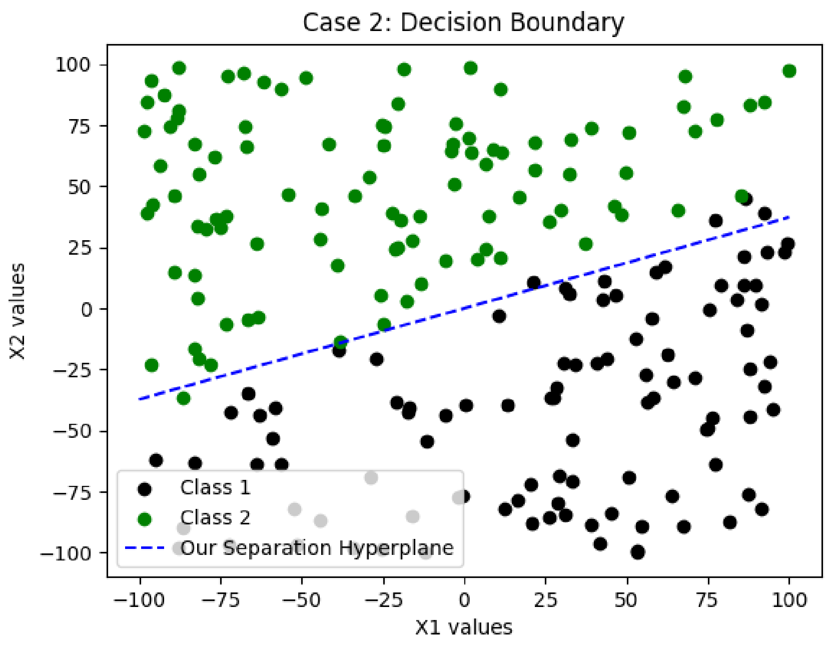
4.1.1 Test 1

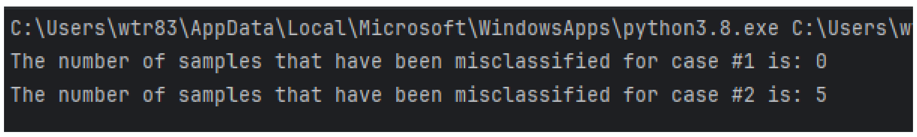


```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 10
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





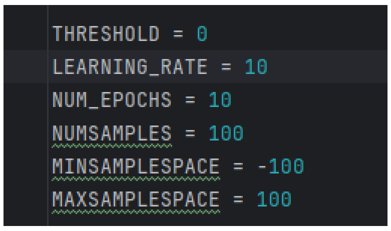


```
C:\Users\wtr83\AppData\Local\Microsoft\WindowsApps\python3.8.exe C:\Users\w
The number of samples that have been misclassified for case #1 is: 0
The number of samples that have been misclassified for case #2 is: 5
```

The model performs well under with what determined to be reasonable values.

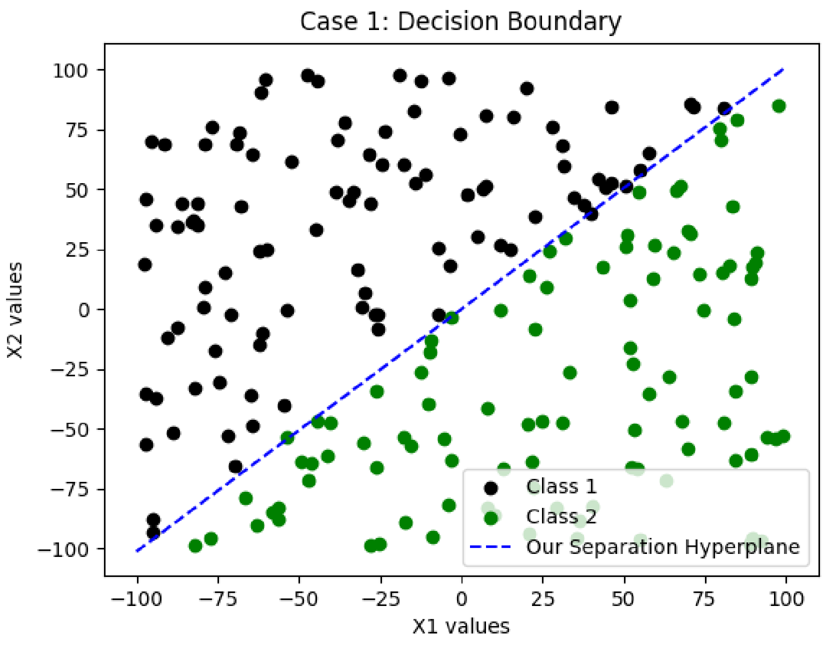
4.1.2 Test 2

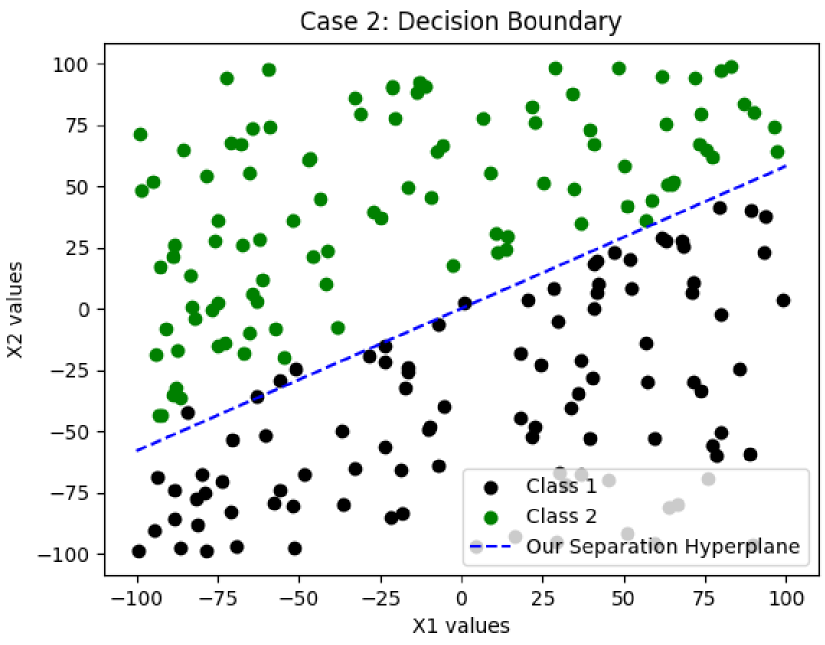
We increased the learning rate by 100-fold to see if we could cause the model to exhibit unstable behavior.

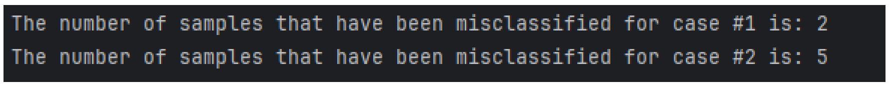


```
THRESHOLD = 0
LEARNING_RATE = 10
NUM_EPOCHS = 10
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





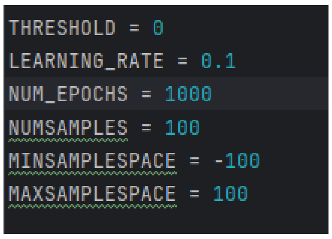


```
The number of samples that have been misclassified for case #1 is: 2
The number of samples that have been misclassified for case #2 is: 5
```

With these parameters, the model's performance was comparable to that of Test 1.

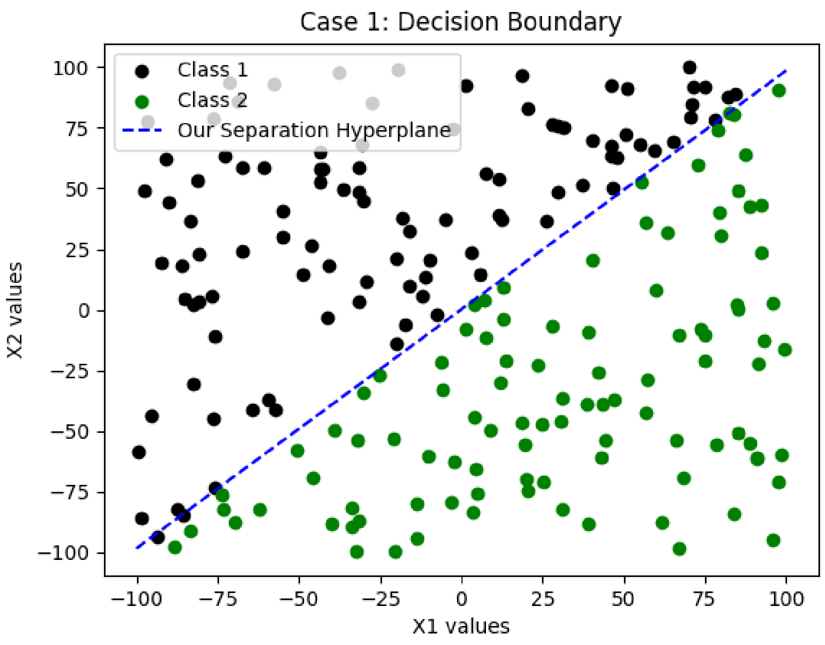
4.1.3 Test 3

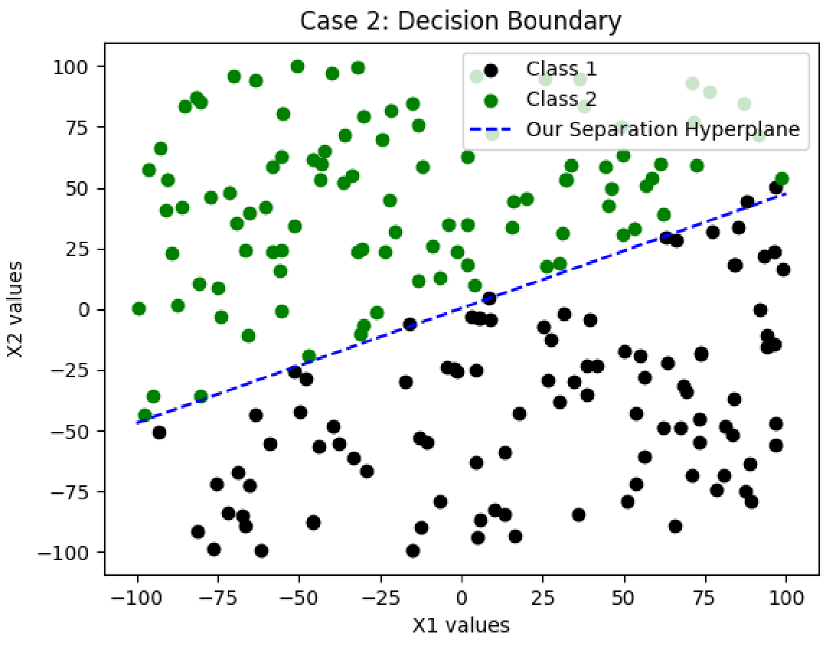
For our additional test with 10,000 samples, we observed a very defining line between class 1 and class 2. We wanted to include this test in our document because we believe it does a good job of showing how well the separation hyperplane is positioned when it has lots of training data. In this example, the hyperplane only has a mistake rate of approximately 1%.

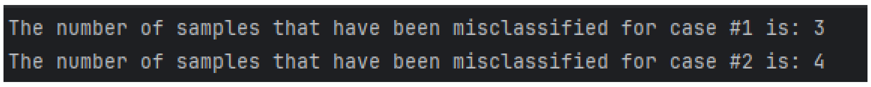


```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 1000
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters

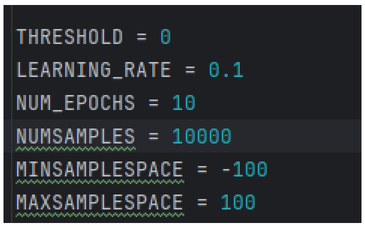






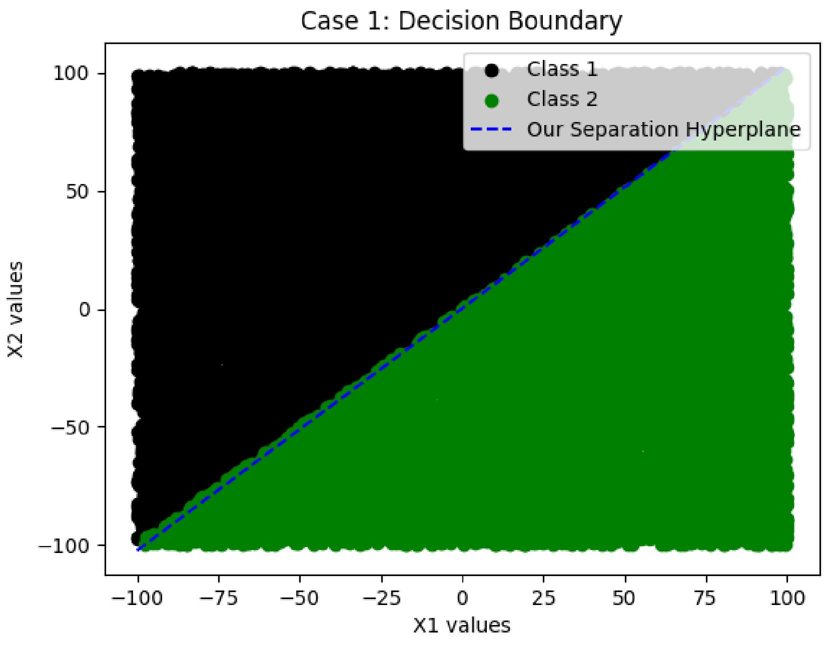
```
The number of samples that have been misclassified for case #1 is: 3
The number of samples that have been misclassified for case #2 is: 4
```

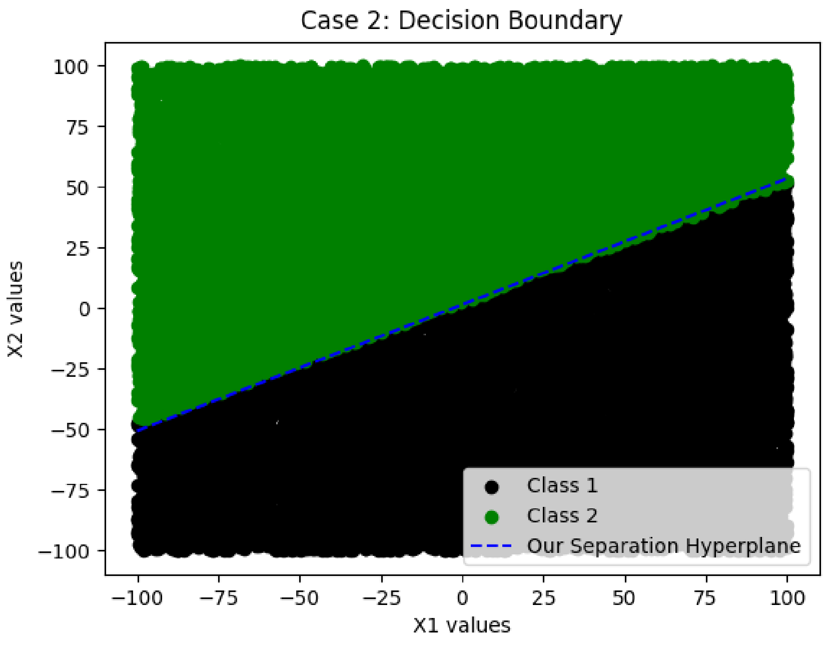
4.1.4 Test 4

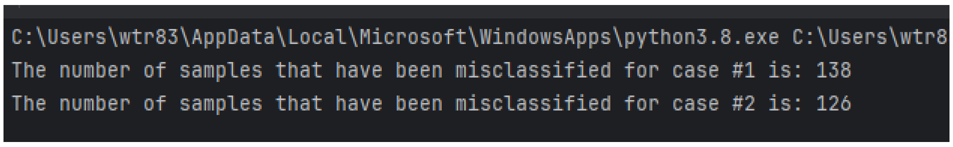


```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 10
NUMSAMPLES = 10000
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





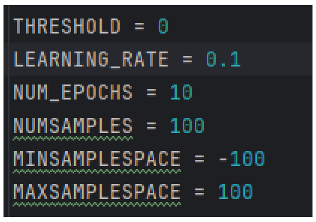


```
C:\Users\wtr83\AppData\Local\Microsoft\WindowsApps\python3.8.exe C:\Users\wtr8
The number of samples that have been misclassified for case #1 is: 138
The number of samples that have been misclassified for case #2 is: 126
```

4.2 Gradient Descent

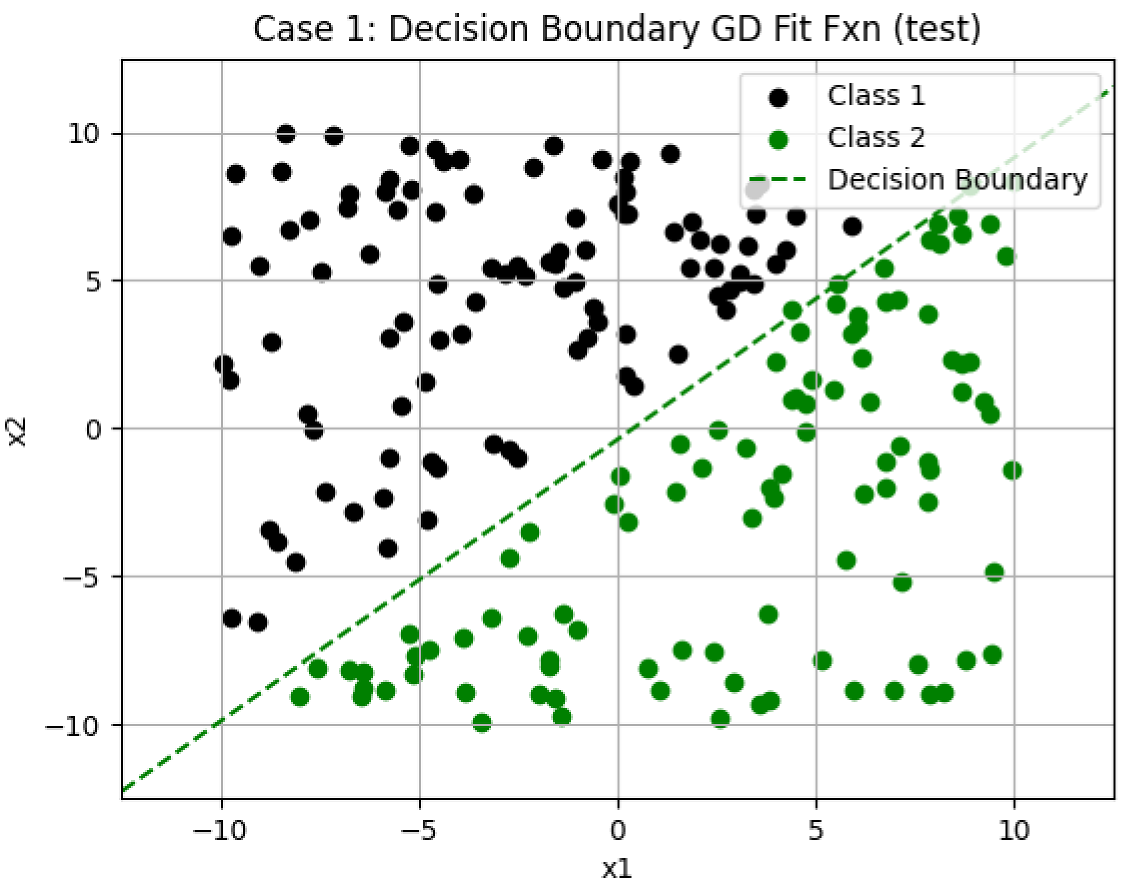
4.2.1 Test 5

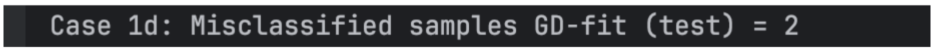
Test 5 has the same set up as test one to act as a comparison to perceptron learning algorithm, however this time using the gradient descent (GD) algorithm. The GD algorithm equalled or improved upon the accuracy of the perceptron learning algorithm as reflected in the data for case 1 (case 3 not pictured, but visible if the program is run). However it wasn't a massive improvement. In one run of the program in case 3, 13 samples were misclassified in the training data using the PLA while two were misclasssified with the GD algorithm. The PLA is still capable of separating the data even with higher dimensionality sample data. However the more dimensions that are added or the less linearly-separable that the data becomes, we may predict that there would be a significant decrease in accuracy for the PLA but not nearly as great of a decrease in the GD algorithm. This is because of the advantages of the GD algorithm over PLA, namely that the GD loss equation factors in the overall inaccuracy of the sample instead of just whether the sample was classified correctly or not.



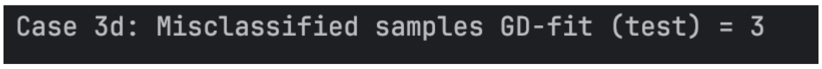
```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 10
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





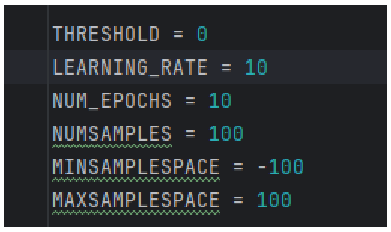
```
Case 1d: Misclassified samples GD-fit (test) = 2
```



```
Case 3d: Misclassified samples GD-fit (test) = 3
```

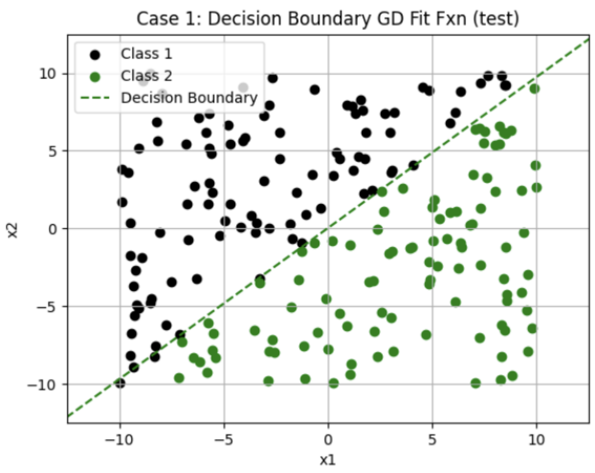
4.2.2 Test 6

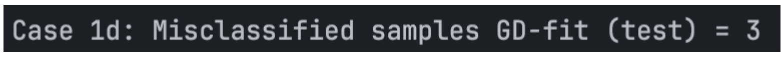
Again increasing the learning rate did not effect the stability of the model in this case.



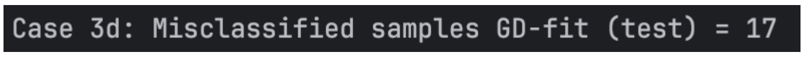
```
THRESHOLD = 0
LEARNING_RATE = 10
NUM_EPOCHS = 10
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





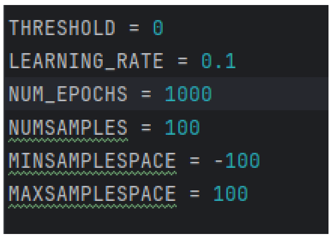
```
Case 1d: Misclassified samples GD-fit (test) = 3
```



```
Case 3d: Misclassified samples GD-fit (test) = 17
```

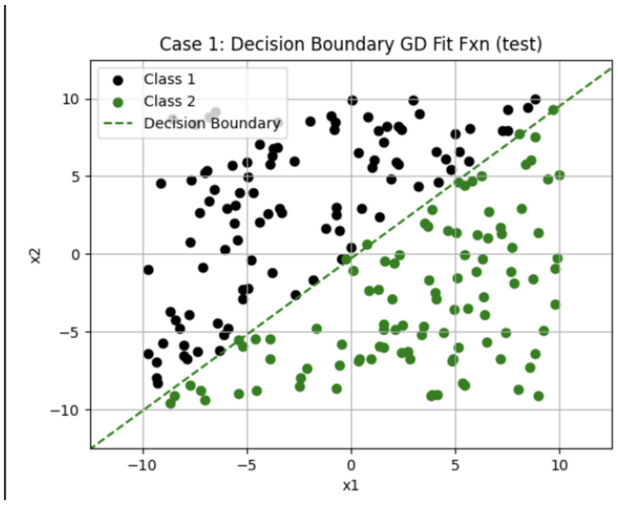
4.2.3 Test 7

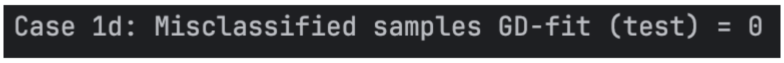
With a large number of epochs the model performs very well.



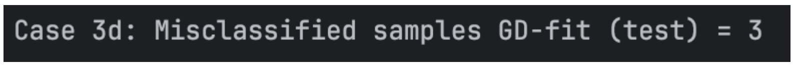
```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 1000
NUMSAMPLES = 100
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters



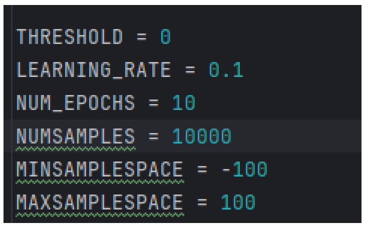


```
Case 1d: Misclassified samples GD-fit (test) = 0
```



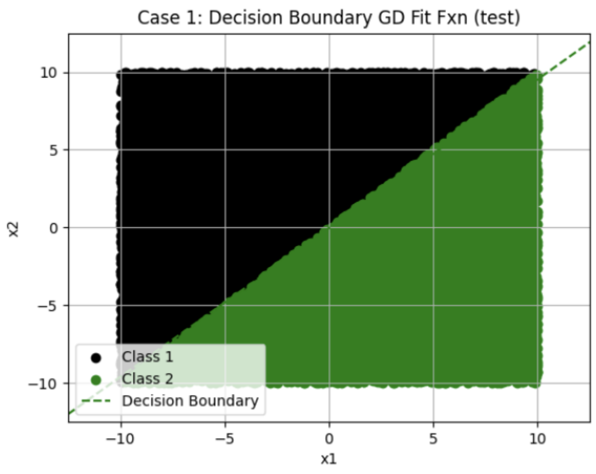
```
Case 3d: Misclassified samples GD-fit (test) = 3
```

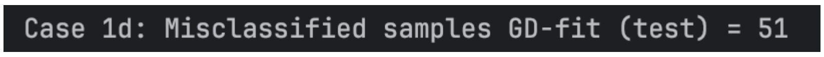
4.2.4 Test 8



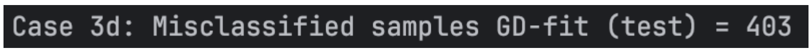
```
THRESHOLD = 0
LEARNING_RATE = 0.1
NUM_EPOCHS = 10
NUMSAMPLES = 10000
MINSAMPLESPACE = -100
MAXSAMPLESPACE = 100
```

Model Parameters





```
Case 1d: Misclassified samples GD-fit (test) = 51
```



```
Case 3d: Misclassified samples GD-fit (test) = 403
```

5 Individual Contributions

5.1 Wayne Rudnick

1) Initialized the GitHub repository and guided the team in starting the project.

2) Developed the perceptron class, including the fit function, test cases, and training data. Plotted the separation hyperplane and analyzed misclassified samples.

3) Assisted with parts 4 and 5 by adapting earlier code, providing guidance, and generating additional tests and graphs.

5.2 Jeremy Middleman

1) Helped with part 5 of the assignment, including writing sample generation, sample testing, and plot generation.

2) Helped write the report, specifically the section on gradient descent.

5.3 Brian High

1) Worked on the document.

2) Helped with Gradient Decent.

3) Helped with test cases.

5.4 Andrei Phelps

1) Contributed to the development of parts 4 and 5 of the code.

2) Created the LaTeX document and provided support in writing the final report.

3) Assisted in creating test cases and refining the code and report for submission.

5.5 Thomas Hynes

1) Added comments in code to enhance readability.

2) Assisted in finalizing the report.