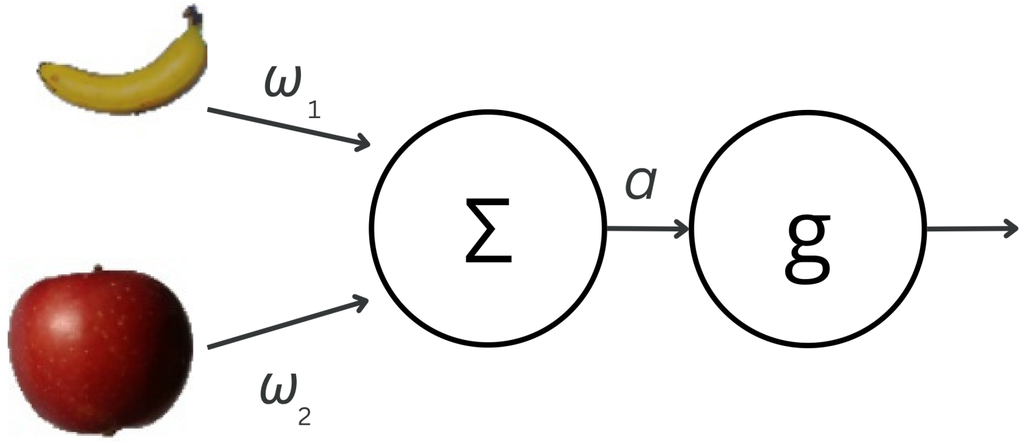
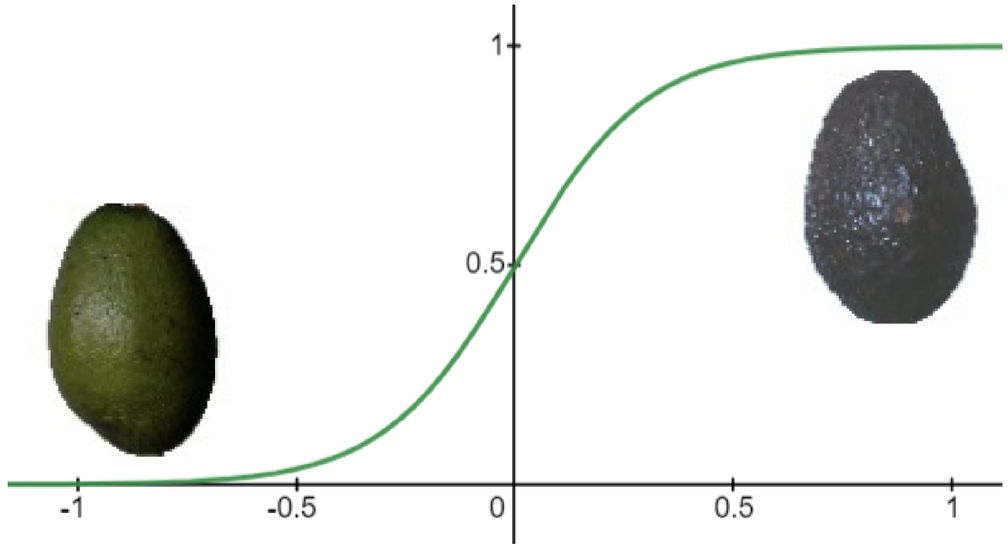
ACTIVITY 8. MACHINE LEARNING: PERCEPTRON AND LOGARITHMIC REGRESSION

Julian Christopher L. Maypa
2020-07587
App Physics 157 WFY-FX-1

Objectives

- Perceptron: Make a perceptron model with a binary piece-wise activation function and use it to classify two distinct types of fruit.
- Logistic Regression: Make a perceptron model with a sigmoid activation function and use it to determine the ripeness of a fruit.



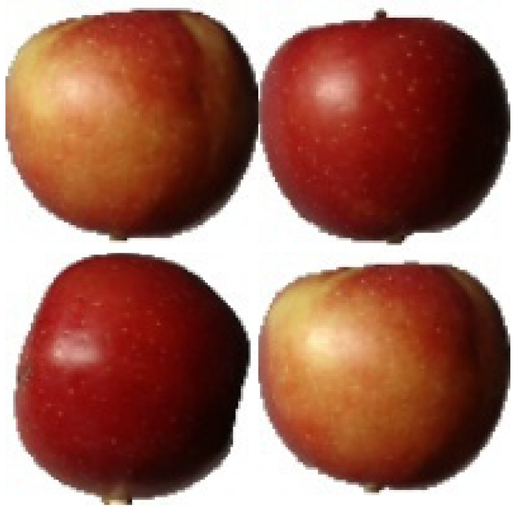


Preparation

The data that I used for this activity were obtained from the Fruits 360 and Fruits Classification datasets in Kaggle [1,2]. For Perceptron, I classified Braeburn apples and Cavendish Bananas, and the features that I used were their eccentricity and average *g* values. For logistic regression, I determined the ripeness of an avocado using their average *r*, *g*, and *b* values. The manual said to use the average R, G, and B values but when I looked at my dataset, I saw that the lighting was uneven, so I used their normalized chromaticity coordinates instead.

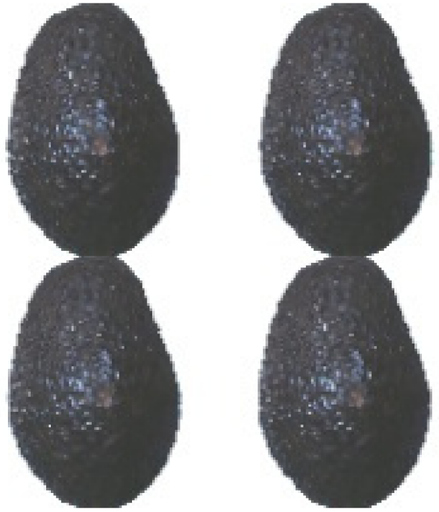
Cavendish Banana

Braeburn Apple



Unripe Avocado

Ripe Avocado

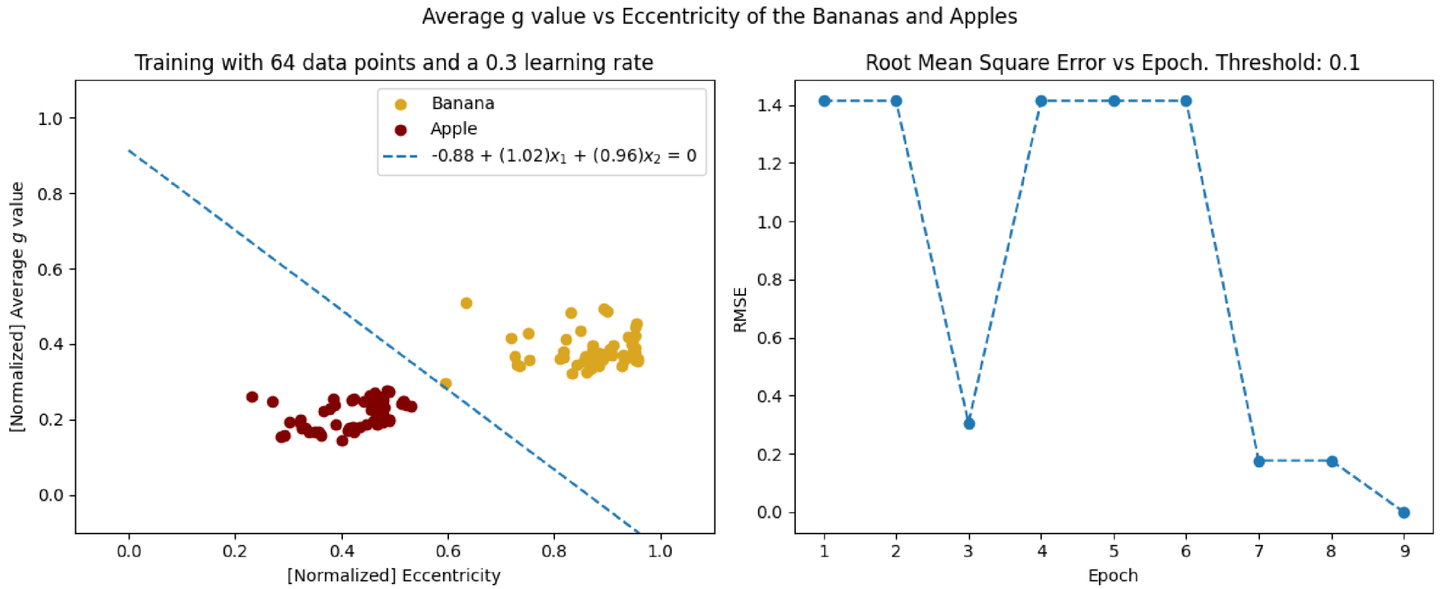


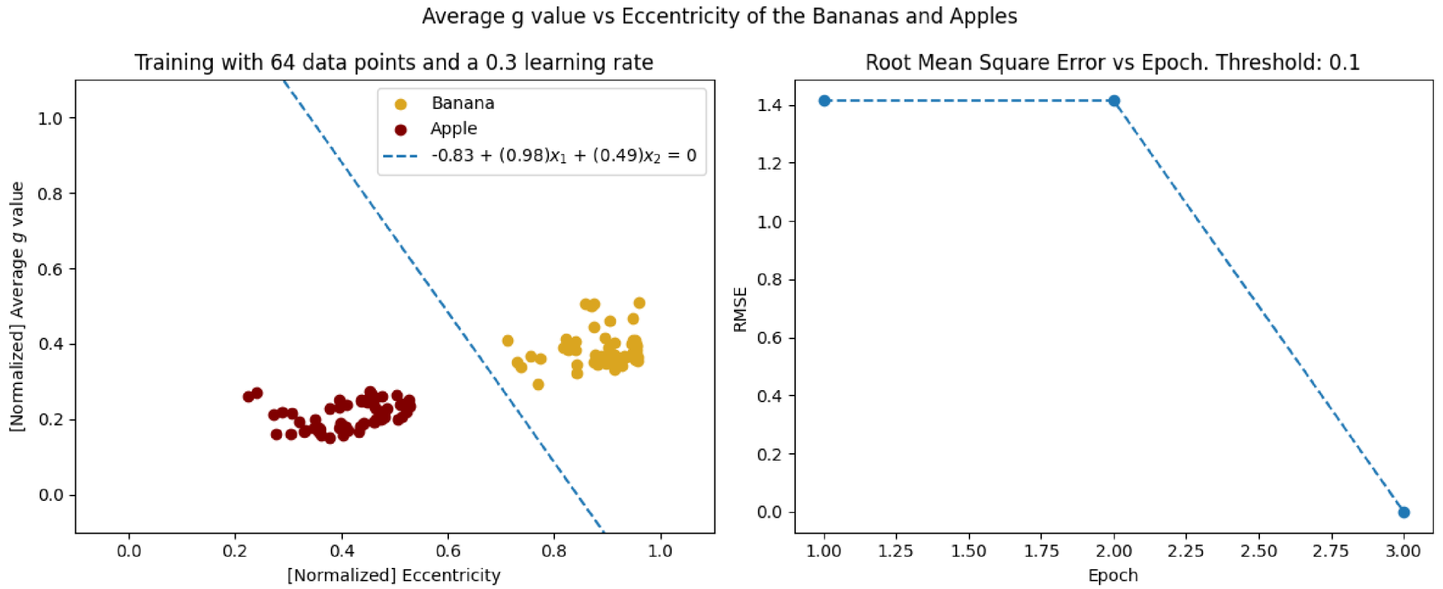
Perceptron

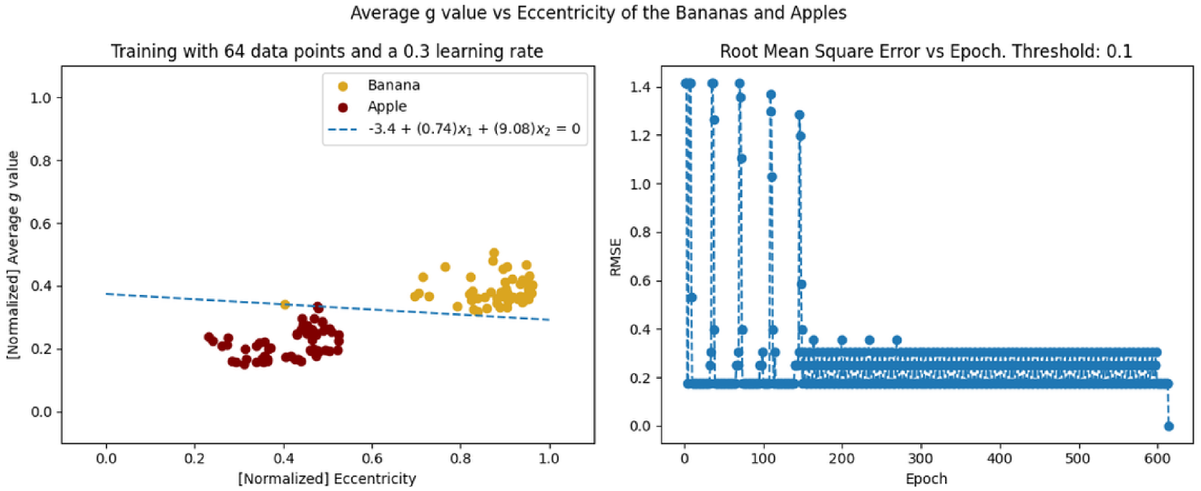
I made a perceptron algorithm which classified apples and bananas. The features that I used for my axes were eccentricity (*x*-axis) and average *g* value (*y*-axis). I first tested my algorithm by using 64 randomly sampled data points (64 apples, and 64 bananas), 0.3 learning rate, and an error threshold of 0.1. The error that I used was the Root Mean Square Error (RMSE). My program stops when the error falls below the threshold.

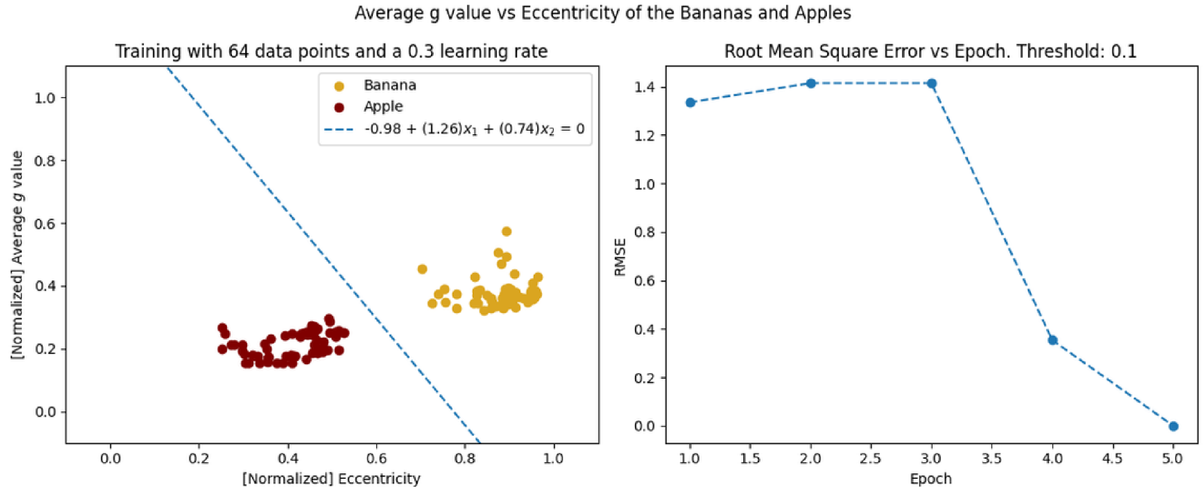
I ran my code multiple times. After each run, new testing data was randomly sampled and the initial weights were also randomized. By looking at the different runs, it can be seen that it takes more epochs to reduce the error when the data points of the different fruits are near each other and it takes less epochs when the data points are far away from each other. This is to be expected because the program has a harder time fitting a decision line when the points are overlapping. But on the other hand, if the points of the different fruits are far away from each other, then it is very easy to create a decision line separating the two [3,4].

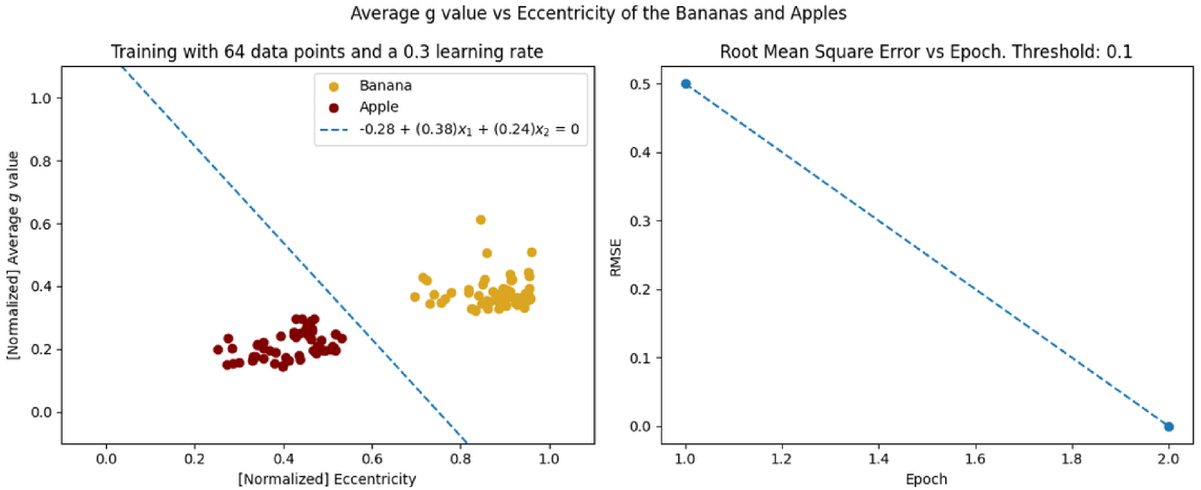
There are more results in the succeeding slide.

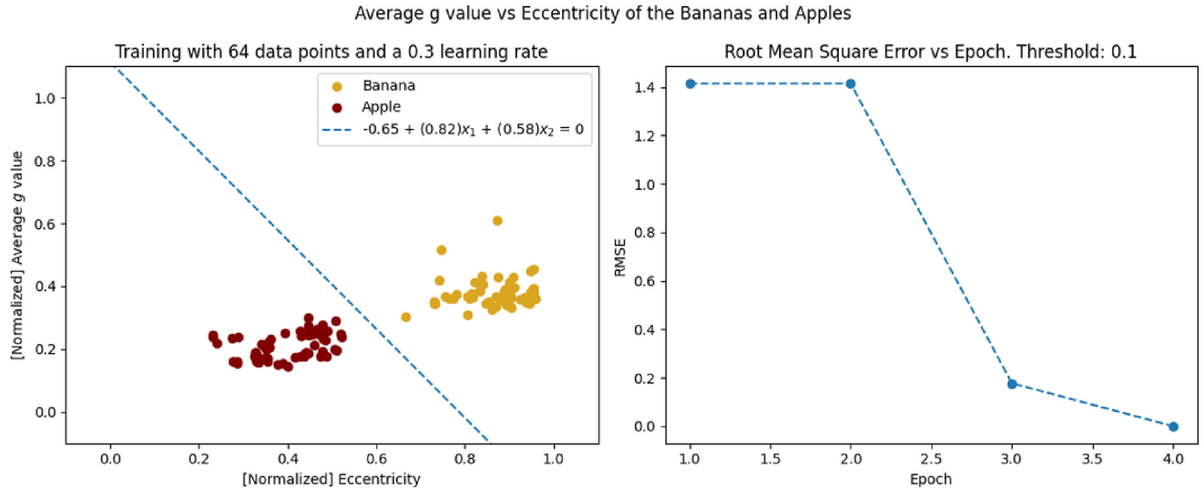





Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.3 learning rate
Banana
Apple
-3.4 + (0.74)x_1 + (9.08)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch


Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.3 learning rate
Banana
Apple
-0.98 + (1.26)x_1 + (0.74)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch


Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.3 learning rate
Banana
Apple
-0.28 + (0.38)x_1 + (0.24)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch


Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.3 learning rate
Banana
Apple
-0.65 + (0.82)x_1 + (0.58)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch

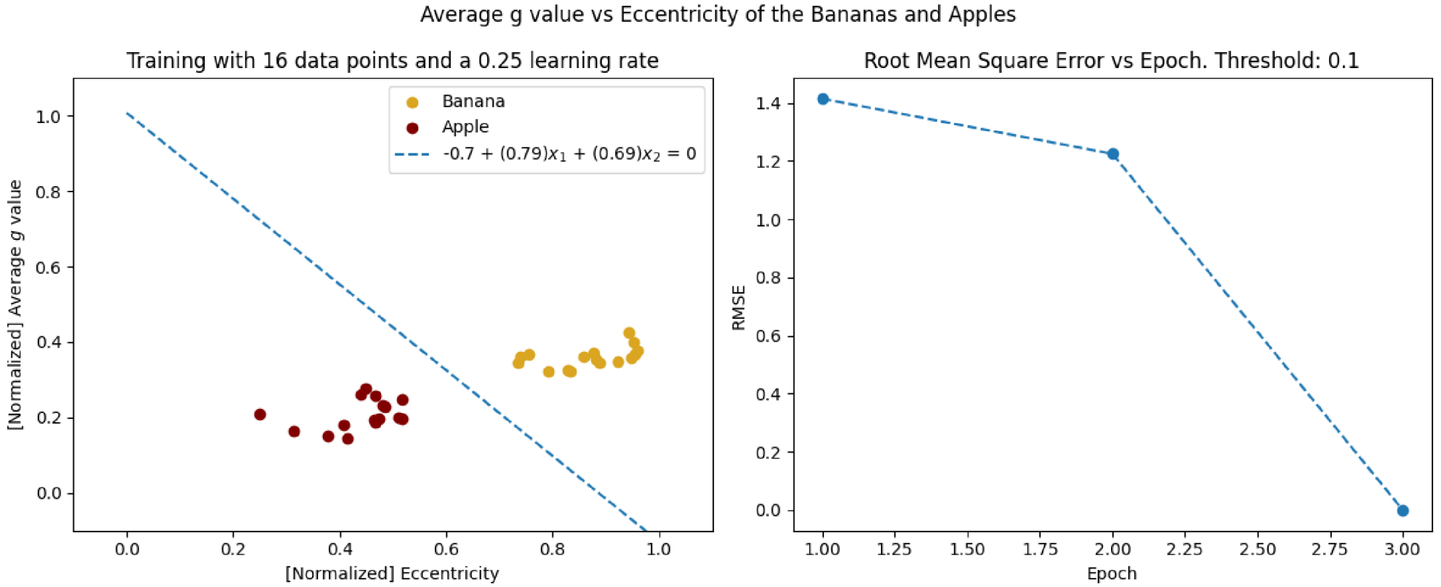
Data Points

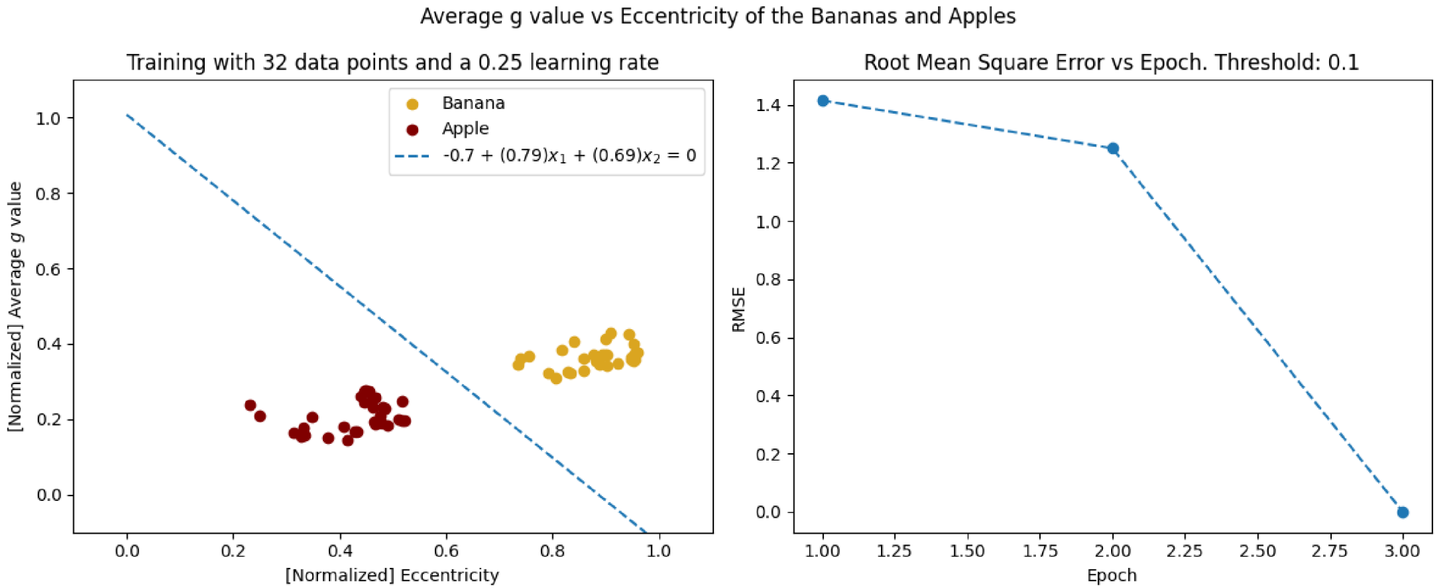
I then investigated the different parameters for the perceptron model. I investigated the effects of the number of data points and the effects of the learning rate. To investigate the effects of varying the data points, I used a constant learning rate of 0.25 and a constant error threshold of 0.1. I then varied the data points by using 16, 32, 64, 128, and 256 randomly sampled data points. But to be consistent, I set a random seed. Thus increasing the data points means new randomly sampled data points were added to the previously selected data points.

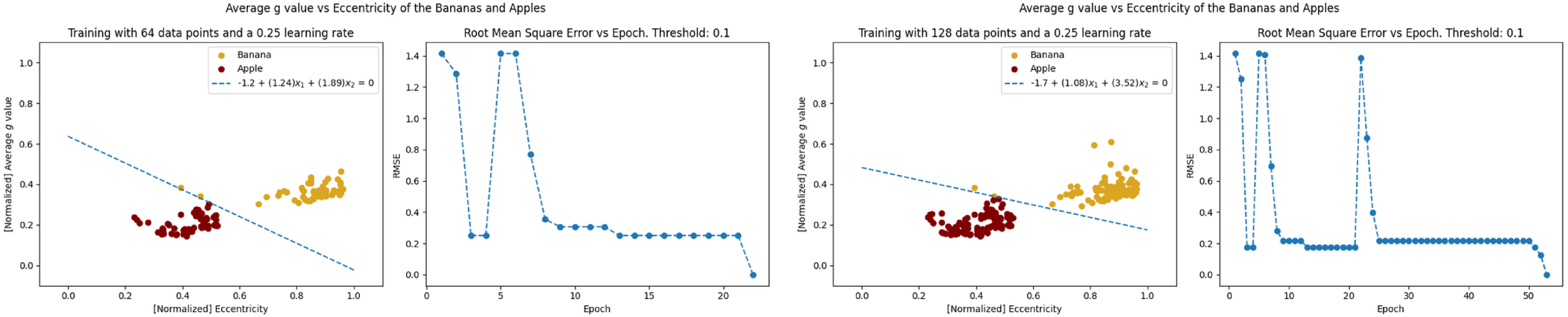
Looking at my results, it shows that increasing the number of data points does not have a significant impact on the learning time of the perceptron. This is evident when I used 16 and 32 data points. The reason why it took more epochs for the 64, 218, and 256 data points is because of the overlap between the banana and apple data points. This overlap is caused by some pictures in the training data. Some pictures have angles of the banana that made it look circular. Which decreased its eccentricity and making it overlap with the apple data.

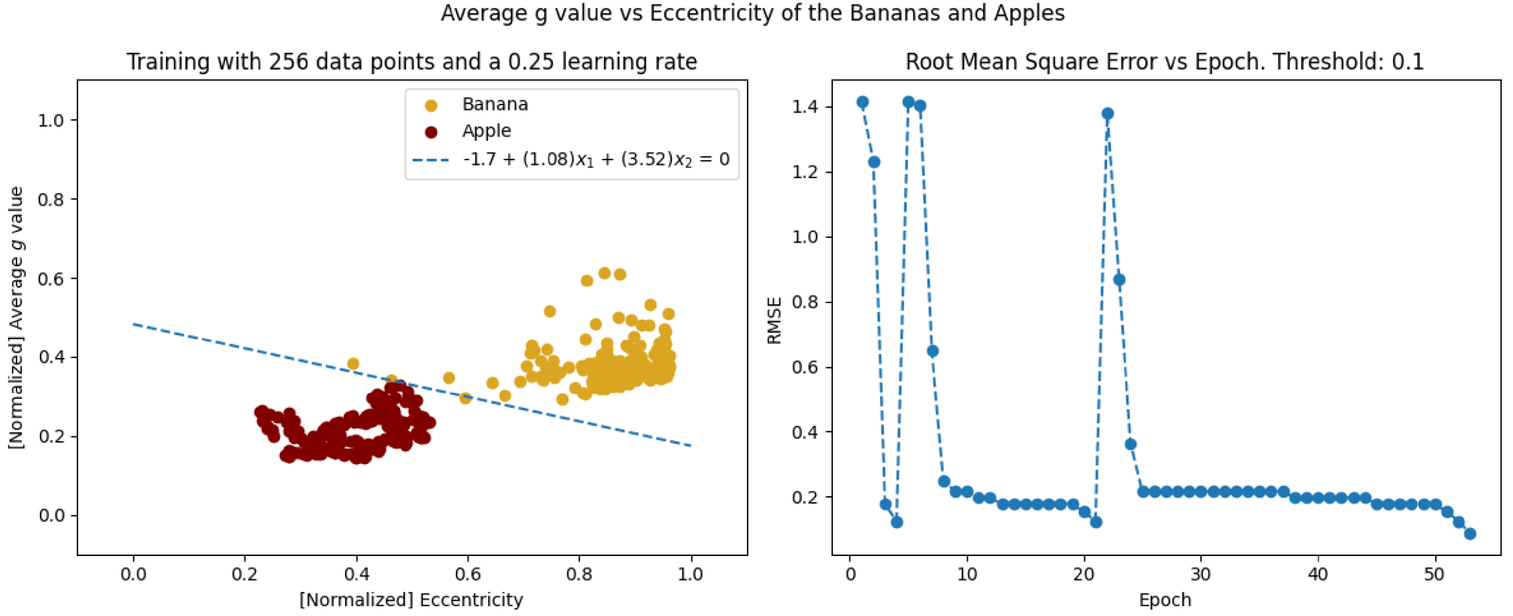
But in a more general sense, having more data makes the decision line more robust [5].

There are more results in the succeeding slide.






Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.25 learning rate
Banana
Apple
-1.2 + (1.24)x_1 + (1.89)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch
Average g value vs Eccentricity of the Bananas and Apples
Training with 128 data points and a 0.25 learning rate
Banana
Apple
-1.7 + (1.08)x_1 + (3.52)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch

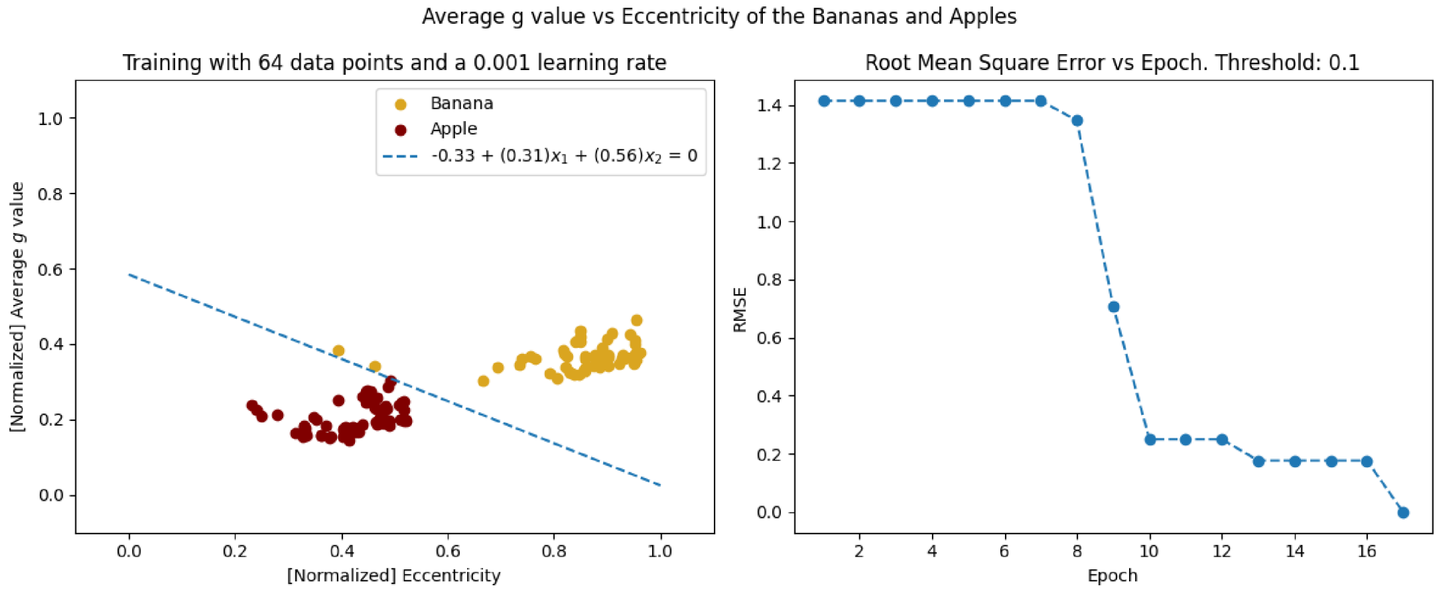

Average g value vs Eccentricity of the Bananas and Apples
Training with 256 data points and a 0.25 learning rate
Banana
Apple
-1.7 + (1.08)x_1 + (3.52)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch

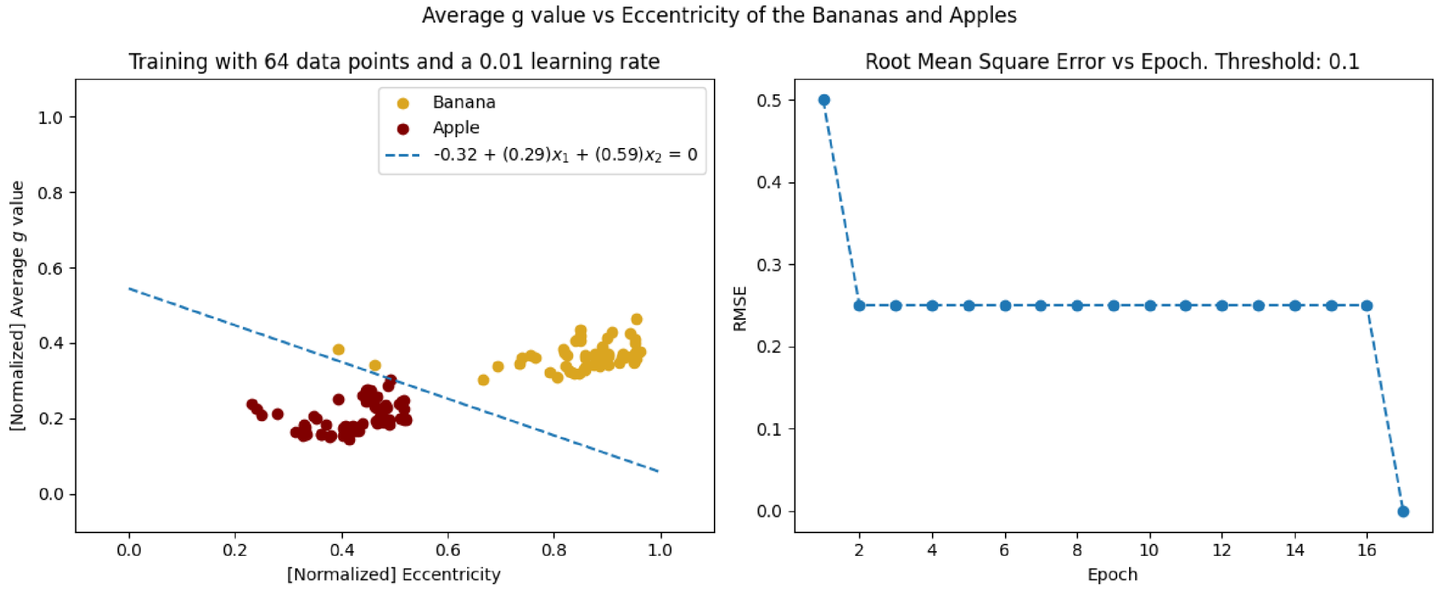
Learning Rate

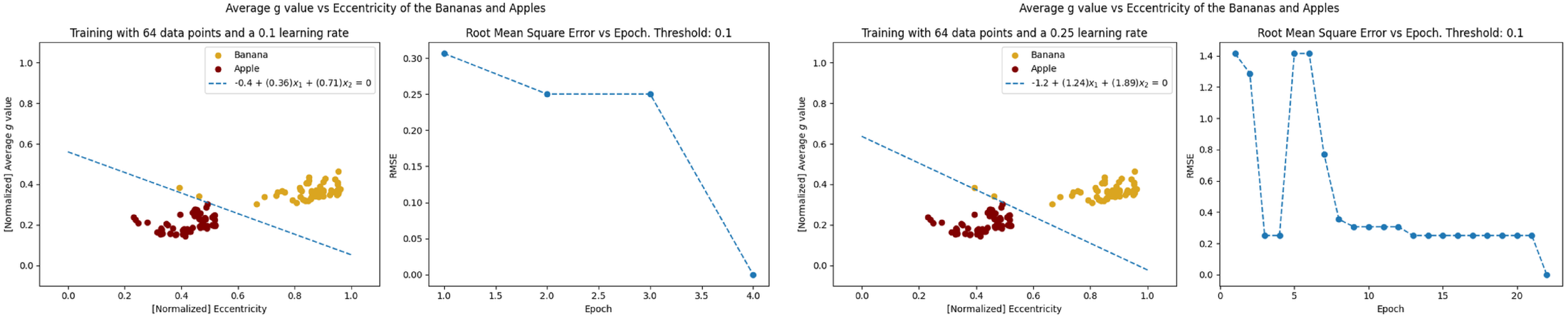
For the effects of the learning rate, I used a constant, randomly sampled 64 data points and a constant error threshold of 0.1. I then varied the learning rate by 0.001, 0.01, 0.1, 0.25, and 0.35.

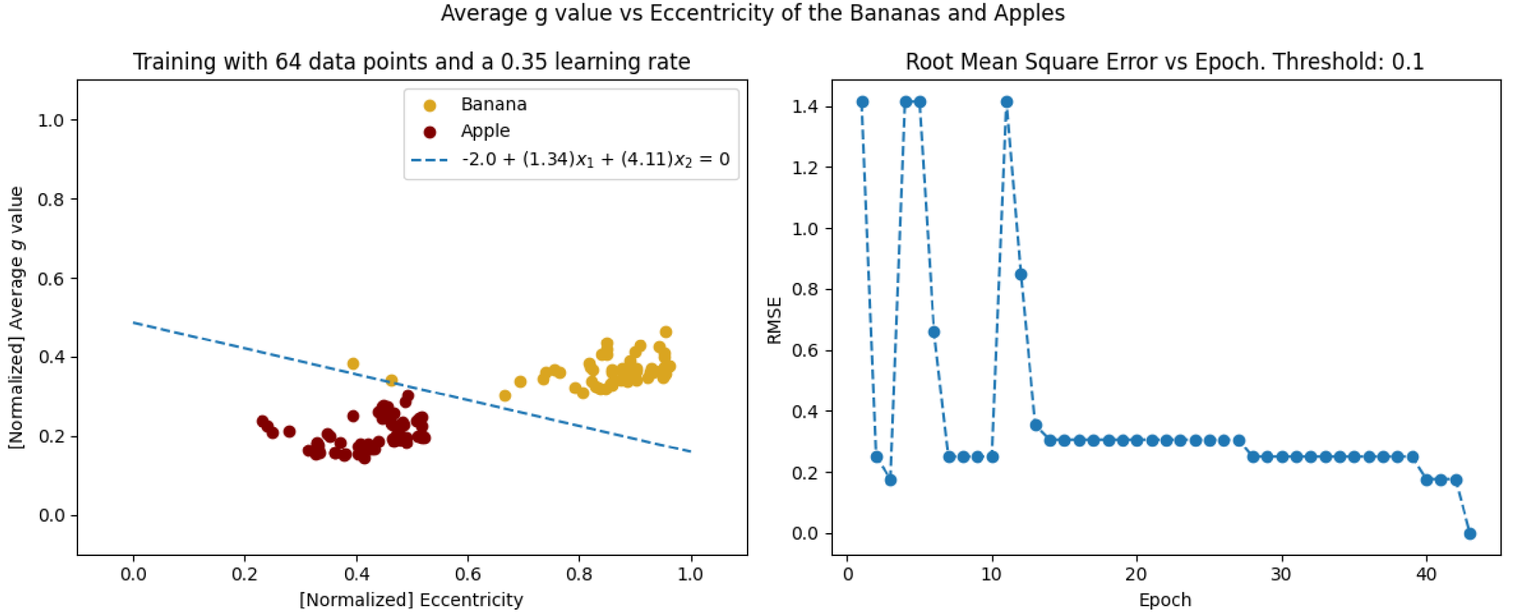
Looking at my results, it shows that increasing the learning rate decreases the learning time. But further increasing the learning rate increases the learning time. This happens because the learning rate controls the rate of change of the weights. If the learning rate is too small, then the weights converge at a very slow pace. But if the learning rate is too big, then the weights will have difficulty converging to a single value because they could miss their point of convergence due to their big adjustments [6]. But for the case these specific 64 data points, it can be seen that the optimal learning rate is 0.1.

There are more results in the succeeding slide.



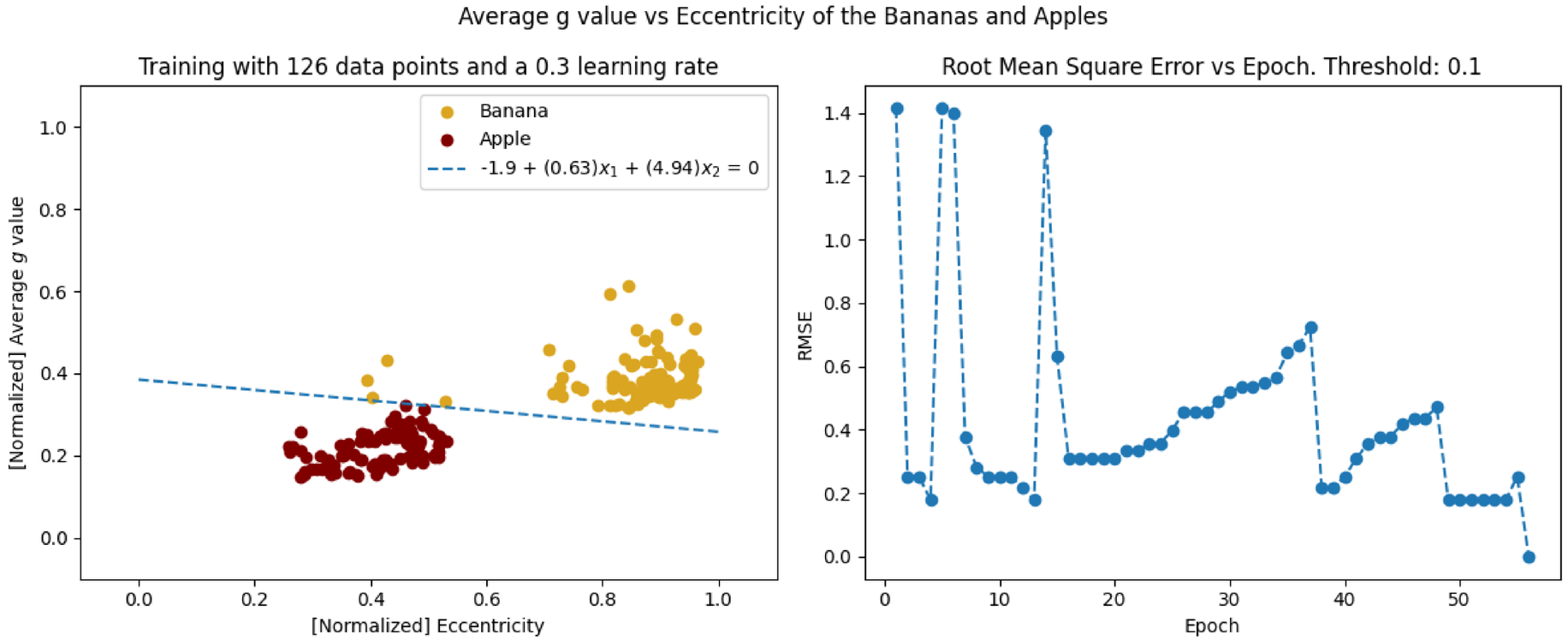



Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.1 learning rate
Banana
Apple
-0.4 + (0.36)x_1 + (0.71)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch
Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.25 learning rate
Banana
Apple
-1.2 + (1.24)x_1 + (1.89)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch


Average g value vs Eccentricity of the Bananas and Apples
Training with 64 data points and a 0.35 learning rate
Banana
Apple
-2.0 + (1.34)x_1 + (4.11)x_2 = 0
[Normalized] Average g value
[Normalized] Eccentricity
Root Mean Square Error vs Epoch. Threshold: 0.1
RMSE
Epoch

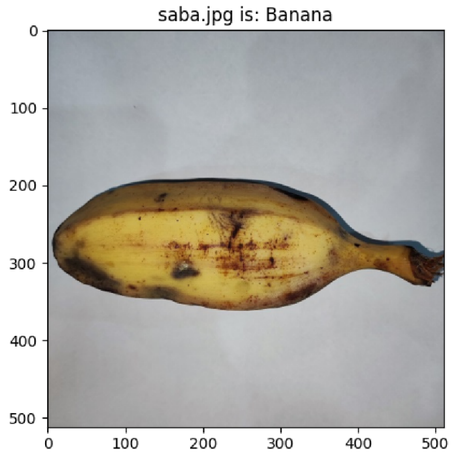
Testing

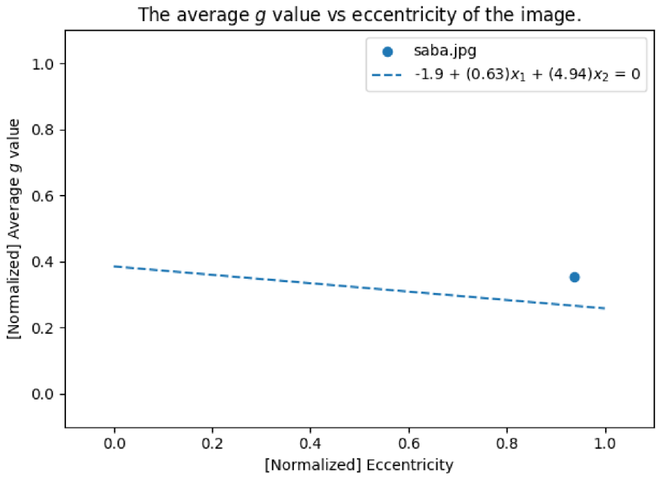
I then tested my perceptron algorithm using never before seen images. I will first recalculate the weights using 126 randomly sampled training data, a learning rate of 0.3, and an error threshold of 0.1. Below are the calculated weights, decision line, and the training time.



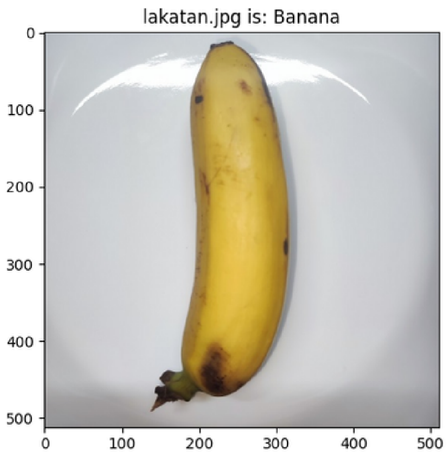
Recall that I used Cavendish Bananas and Braeburn Apples. So I want to test if my perceptron model works with different species of bananas and apples. Using the same feature extraction process, I tested my program using a picture of a lakatan and saba banana, and a fuji apple. Below are my results:

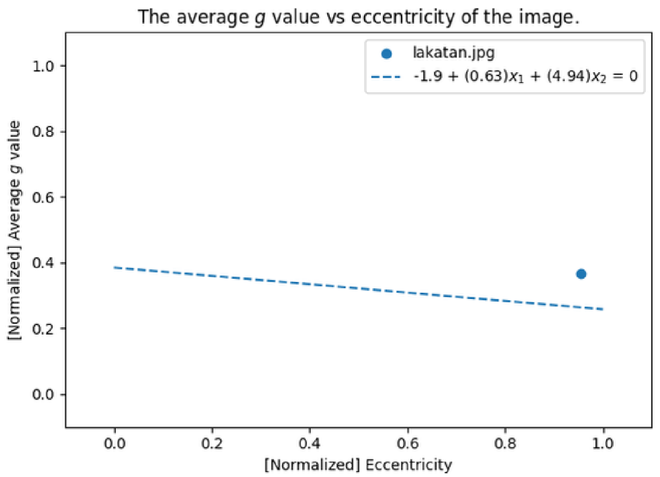
Testing with a never before seen image



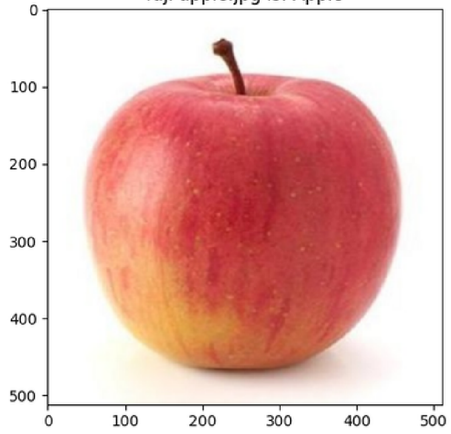


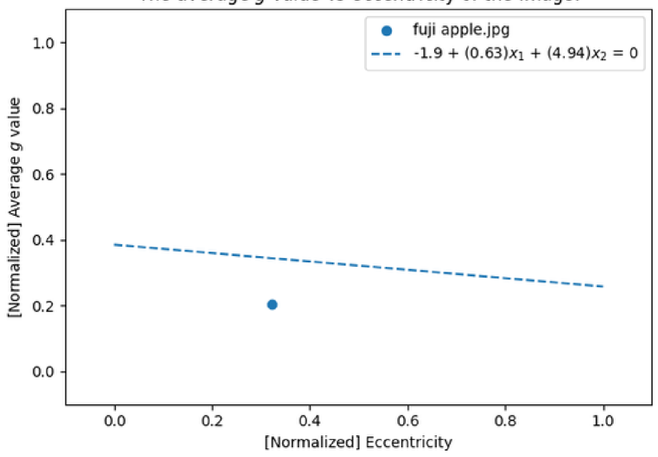
Testing with a never before seen image





Testing with a never before seen image

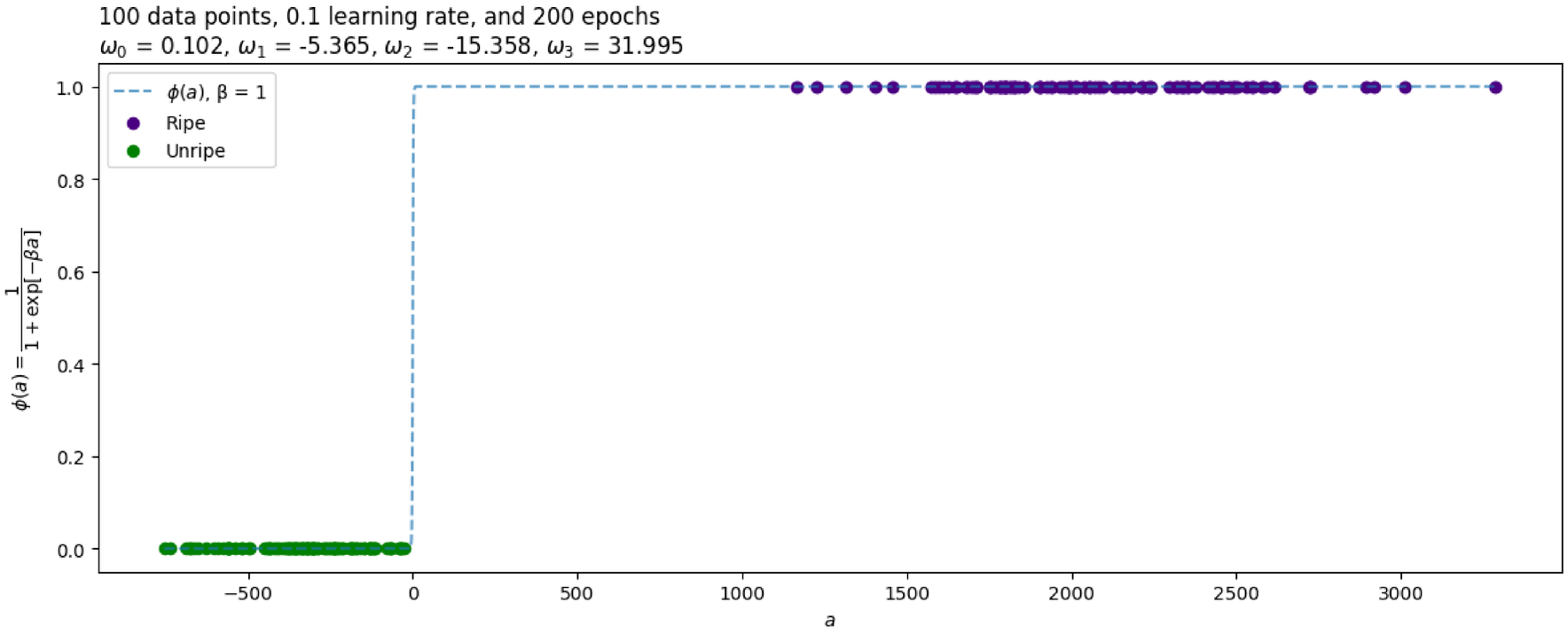




Looking at my results, it can be seen that my code has successfully classified the never before seen images. This proves that my perceptron algorithm works and the parameters that I used for training are adequate in classifying bananas and apples.

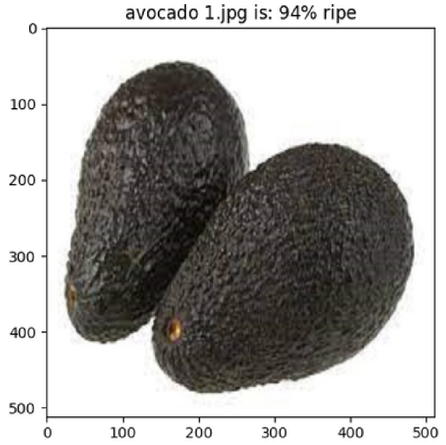
Logistic Regression

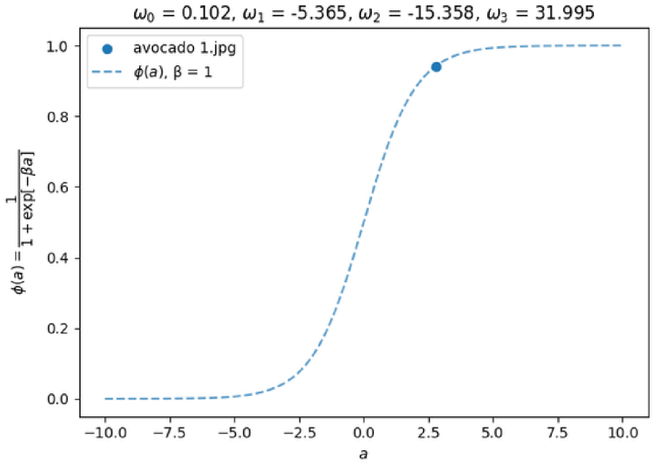
For logistic regression, I used the same algorithm for the weight change. But I changed the activation function to a sigmoid function and I set the number of epochs for the training. I used 100 randomly sampled data points (100 unripe and 100 ripe), 200 epochs, and a learning rate of 0.1. My results can be seen on the right.



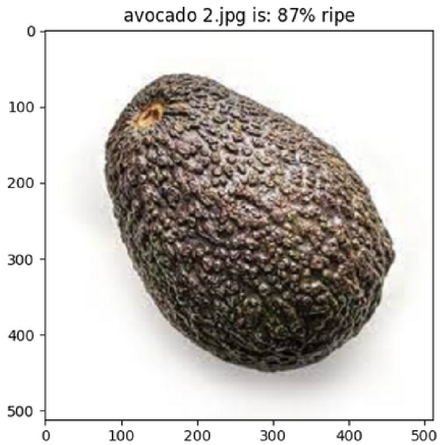
I then tested my program using never before seen images. Below are my results:

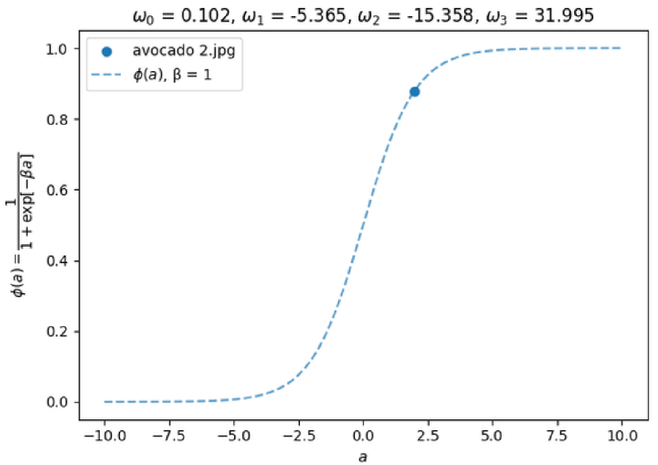
Testing with a never before seen image



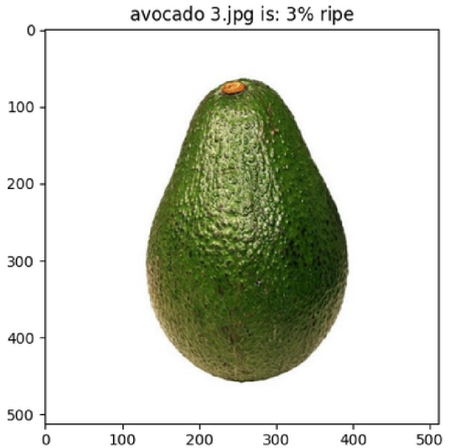


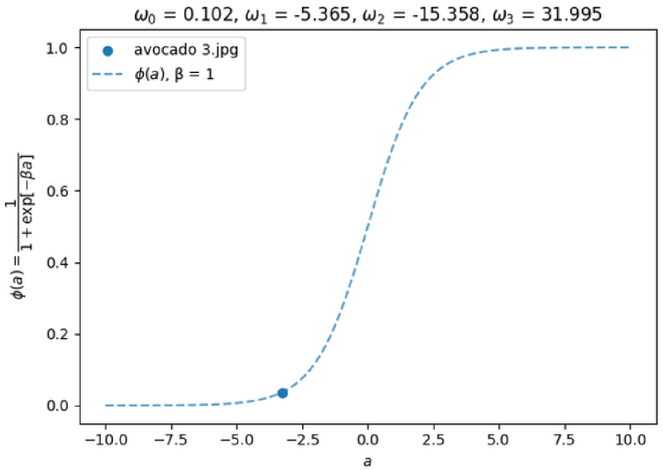
Testing with a never before seen image





Testing with a never before seen image





Looking at the results and visually verifying them, it can be seen that my results are accurate. This is because if you take a look at the training data back in page 3, it can be seen that the ripe avocados are very dark and even appears blackish. Since there are still very faint green spots in avocado 1 and avocado 2, then they are not yet 100% ripe. For avocado 3, it can be clearly seen that it is fully green, so 3% ripe is accurate. With this, my program has successfully implemented logistic regression.

Reflection

Overall, I believe that the results I got are correct. My perceptron was able to classify apples and bananas correctly, even though the testing images were a different species than the training images. My code for logistic regression also worked and it was able to determine the ripeness of an avocado. Upon visually inspecting the images, the resulting ripeness was consistent with the color of the avocados.

The most tedious part for me in this activity was understanding the perceptron algorithm. It took a long time, but I was able to fully understand it in the end.

I'd like to thank my instructors, Sir Rene Principe Jr. and Sir Kenneth Leo, for guiding me throughout the activity. I would also like to thank my professor, Ma'am Jing, for guiding me in my coding while my classmates and I worked in R202. I would also like to acknowledge my classmates: Abdel, Johnenn, Jonabel, Richmond, Lovely, Hans, Genesis, Jeruine, Rusher, and Ron for helping me complete this activity.

Self Grade

Technical Correctness	I understood the lesson and met all the objectives. My results are complete and I got the expected results.	35
Quality of Presentation	The images I added to this report are of good quality and all the graphs are properly labelled. My code is also properly organized and labelled.	35
Self Reflection	I got the expected results, and acknowledged the contributions of my peers while doing this activity. I also properly cited online references.	30
Initiative	Apart from doing the required tasks, I also helped my classmates with their code and helped them by cross-referencing my results with theirs.	10
	Total	110

References

[1] Zhang, E., et. al., (n.d.). Fruit Classification Dataset. Kaggle. Retrieved from https://www.kaggle.com/datasets/sshikamaru/fruit-recognition

[2] Oltean, M., et. al., (n.d.). Fruits 360 Dataset. Kaggle. Retrieved from https://www.kaggle.com/datasets/moltean/fruits
2023, from https://www.bsp.gov.ph/Coins%20and%20Notes/Coins/NGCCS/NGCCoins.pdf

[3] Syed, Mujahid & Hassan, Md & Ahmad, Irfan & Hassan, Mohammad & Albuquerque, V.H.C.. (2020). A Novel Linear Classifier for Class Imbalance Data Arising in Failure-Prone Air Pressure Systems. IEEE Access. PP. 1-1. 10.1109/ACCESS.2020.3047790.

[4] Chunbo Liu, Yitong Ren, Mengmeng Liang, Zhaojun Gu, Jialiang Wang, Lanlan Pan, Zhi Wang, "Detecting Overlapping Data in System Logs Based on Ensemble Learning Method", Wireless Communications and Mobile Computing, vol. 2020, Article ID 8853971, 8 pages, 2020. https://doi.org/10.1155/2020/8853971

[5] Dorfman, E. (2022). How Much Data Is Required for Machine Learning?. Postindustria. Retrieved from: https://postindustria.com/how-much-data-is-required-for-machine-learning/

[6] Brownlee, J. (2019). Understand the Impact of Learning Rate on Neural Network Performance. Machine Learning Mastery. Retrieved from: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/