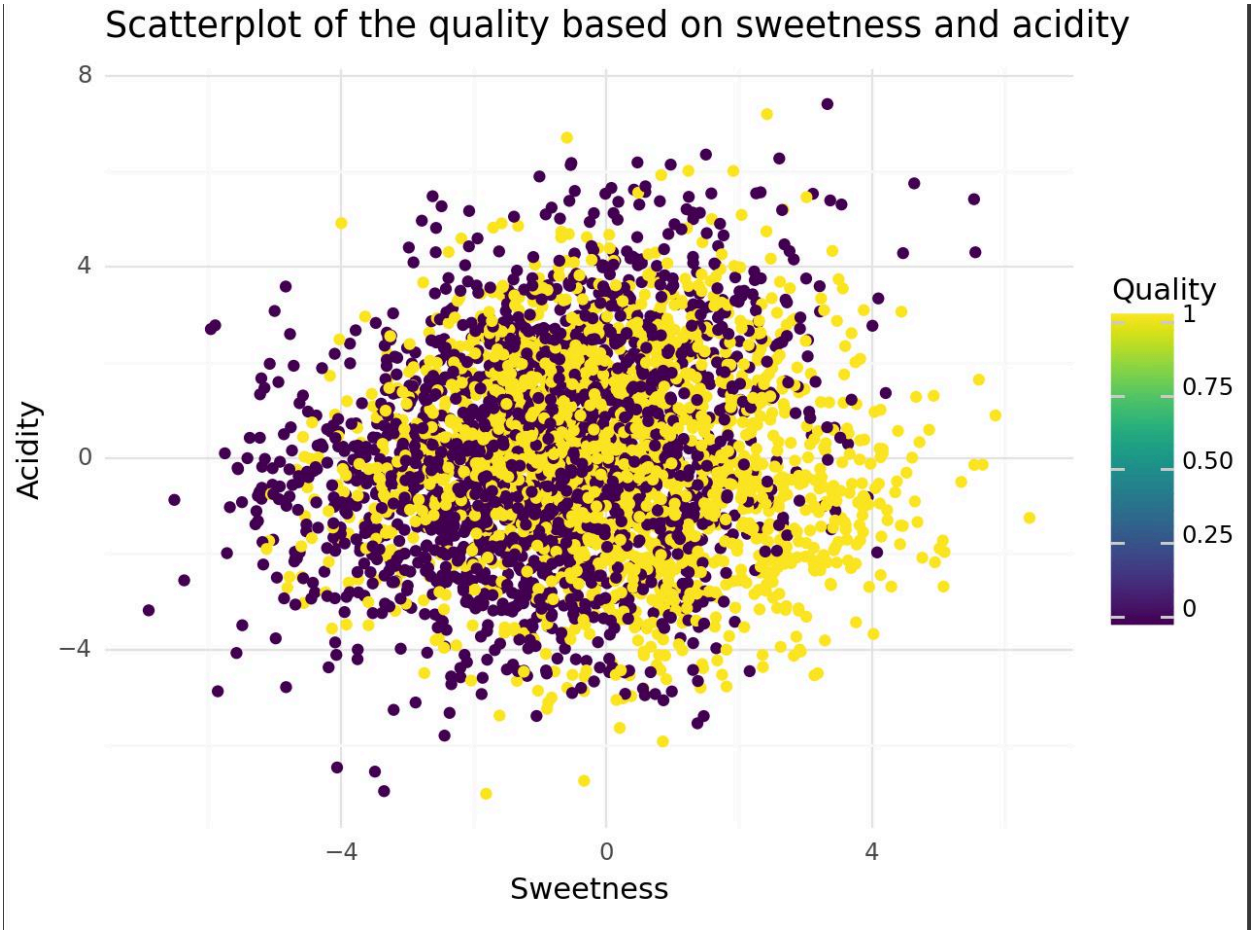
Introduction

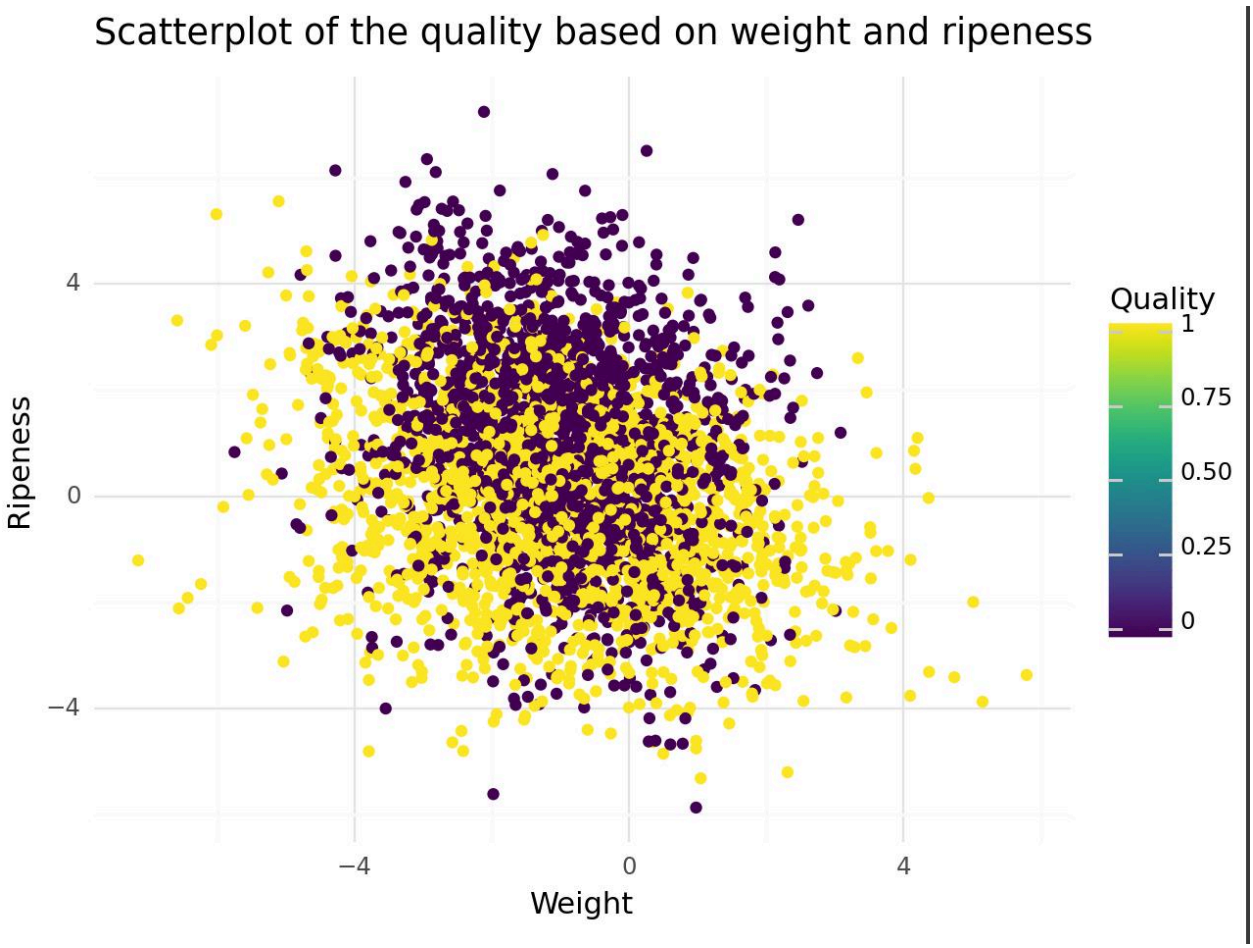
The data used for this project was the Apple quality data provided in the assignment brief. The features in this data are: Apple id, size of the fruit, weight, sweetness, crunchiness, ripeness, acidity and quality of the apple, being good or bad is what was predicted.

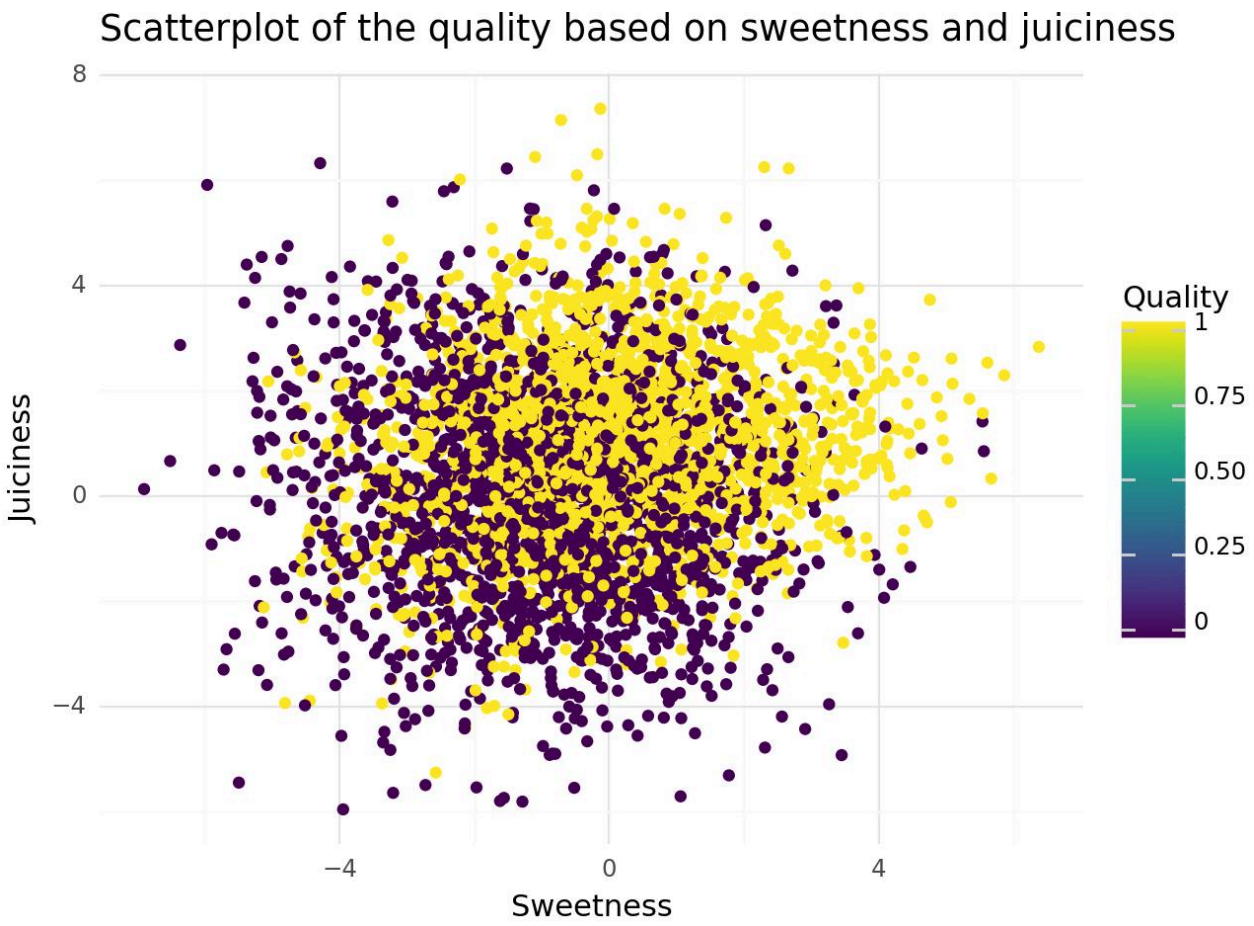
The data is z-scored from the beginning so to the human eye we don't really have a great concept of what the values mean. Additionally, there are no units mentioned for any of the metrics so we don't really have any point of reference. Although units can be assumed for some of the features, we really have no idea what features like crunchiness or sweetness are measured in. Although classification of fruits may not seem super practical, for a company that sells or grows apples, this data could be extremely important for quality assessment of apples. The data is decently hard to read and based on graphs, it is also difficult to find separation within the data, even when graphing features that I thought would be able to make some sort of separation in the data. In this scenario, I am comparing the performance of a deep neural network with this problem, to a simpler model, which I picked to be a logistic regression.

Analysis

The data, as mentioned in the introduction, was pretty difficult to interpret due to the fact that it is all pre z-scored and has no units. Additionally, there is no way to access the data before z-scoring, but even if I could, it still wouldn't bring much to the table without any units. Additionally, when examining various scatter plots created through gg plot, some findings are shown, but still not a whole lot.


Scatterplot of the quality based on sweetness and acidity
Acidity
8
4
0
−4
−4
0
4
Sweetness
Quality
1
0.75
0.50
0.25
0


Scatterplot of the quality based on weight and ripeness
Ripeness
4
0
−4
−4
0
4
Weight
Quality
1
0.75
0.50
0.25
0



Through these three graphs, I notice a similar pattern in all three, but aside from this specific pattern, I don't really feel like there is really anything further to gain. The only pattern that I noticed was that in the extremes, which are the corners of each graph, there seems to be a high density of either good or bad apples. Aside from that though, in the middle there is such a high density of good and bad apples that seems inseparable. I believe this makes sense because when you are a lower amount of standard deviations from the norm such as 0-2 range, there will be an extremely high density of points that seem inseparable because 95% of data is found within two standard deviations. This also explains why it seems like in the extremes, we can clearly see a majority of good or bad apples because so little data is found outside of 2 standard deviations and in the extremes of these features, we see either good apples only or bad apples only.

Methods

To begin this project, I began with a decently complex model as we had used in class just for baseline. It had 5-7 layers and started with 256 neurons and scaled down throughout the layers. With this model over no matter how many epochs the accuracy never got above .75 and I thought this was extremely odd.

After thinking about it for a little while and reviewing some of the video lectures I realized the model structure I chose for this problem was far too complex. This data only had 4000 points and

was predicting something relatively simple, if an apple's quality was good or bad based on various features such as, crunchiness, sweetness and other features that are relatively easy to understand for someone who has eaten an apple before. The reason the initial model wasn't able to predict if an apple had good or bad quality was because the model was too complex for the data which was not sufficiently complex.

The model I ended up using had only 3 layers and the first two had "relu" activation which is an activation function that returns 0 if the number is a positive number and the actual number if it is anything else. What this does is introduce non-linearity within the situation. Additionally, I used a dropout layer which makes every single node have a 20% chance of being reduced to 0. While this may seem counterproductive, this actually helps to reduce overfitting, which is when a machine learning model performs better on data that it has already seen and learned from compared to new, unseen data, which is the point of building these models, to be able to predict well on new and unseen data. For the second to last layer of the model, it also uses "relu" activation and for the final layer of the model, I used "sigmoid" activation which returns only a 0 or 1 which is exactly what is needed for this problem where we are identifying if an apple is good or bad, which we can portray with a 0 or 1. For the second model I used a logistic regression model. There isn't much to be said about this because I chose it because logistic regression itself is pretty simple and not computationally expensive either.

Results

My deep learning model performed very well for this problem. Much better than the more simple logistic regression. The deep learning model was relatively simple and not very computationally intensive as it didn't have many layers nor neurons and still performed about 20% better than the logistic regression model. This is a very significant difference and makes it clear that deep learning was necessary for this problem. While the metrics of the logistic regression model of about 75% accuracy is still pretty good, it gets blown out of the water by the deep learning model's 94% accuracy. The only downside of the deep learning model is that it is more computationally intensive than the logistic regression model, even though the deep learning model is very simple when compared to deep learning models in general.

Reflection

The most significant thing I learned from doing this assignment is that when building a deep learning model, extensive data exploration is needed in order to find the ideal structure for the deep learning model. Initially, the model I used was too complex for the data, which itself wasn't very complex and by creating a simpler model, I was able to get very good performance from the model. It was decently difficult since there isn't really a rule that can be universally applied to all deep learning models based on brief data exploration. What I would do differently is explore the data on my own first before making a model, and as a result I would hopefully be able to save some time.