Prediction of Cure for Princess Disease

Mahesh Nalla (18259448)
University of Missouri-St. Louis

8 May 2023

Contents

1 Introduction

Once upon a time, a lovely princess who had developed a serious illness lived in a far-off realm. The nation's king and queen did everything in their ability to heal her, but none of the treatments they attempted proved to be effective. A group of local peasants told the king and queen about a mixture of supernatural ingredients that could be used to treat any illness. The villagers warned that because of recent droughts, there might only be a small amount of these

ingredients available at any given time, and that only an expert alchemist could determine whether a specific combination of these particularly volatile and rare ingredients would be able to heal the princess. The goal here is to in this project, how Artificial Intelligence can predict determine the amounts they had taken of each ingredient, along with whether or not it had led to a cure. I will try to implement neural networks

2 Dataset

The dataset was obtained from kaggle Data Science website called the "CLASSIFY Princess Cure". This dataset has been made publicly available. It contains with the given sets of ingredients, as mentioned they are volatile and scarce so we need to be ready for different combinations to be available at any given time.

2.1 Dataset Description

There are 2338 rows and 14 columns. I will extract 13 columns and add 1 column for target variable. The label can be '0' or '1'. To draw that disease can be "1" for cured or "0" for not. These ingredients (features we will use to predict) are as follows:

- Phoenix Feather
- Unicorn Horn
- Dragon's Blood
- Mermaid Tears
- Fairy Dust
- Goblin Toes
- Witch's Brew
- Griffin Claw
- Troll Hair
- Kraken Ink
- Minotaur Horn
- Basellisk Scale
- Chimera Fang

2.2 Input Data Visualization

The histogram plot of every input features showing their maximum and minimum value as well as how they are distributed can be seen in the images given below.

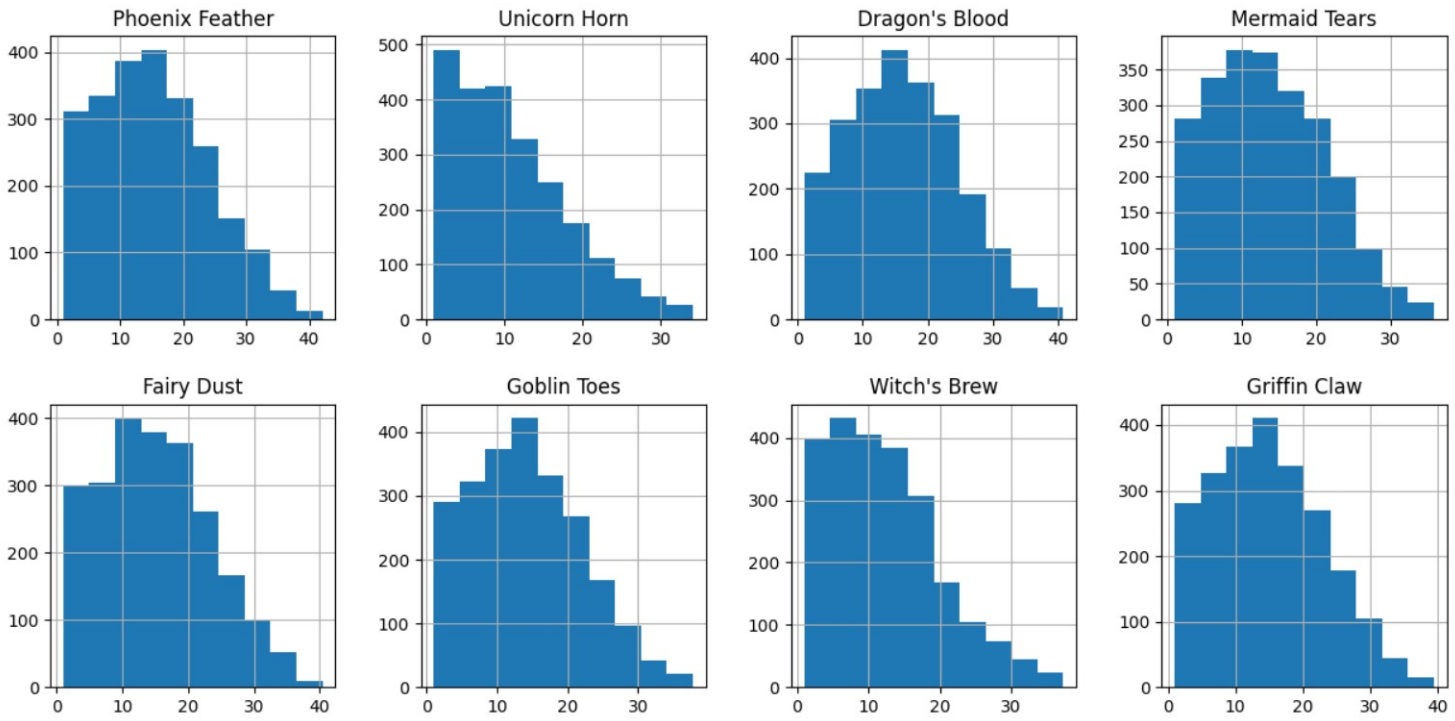

Figure 1: Input Data Distribution Histograms (Part One)

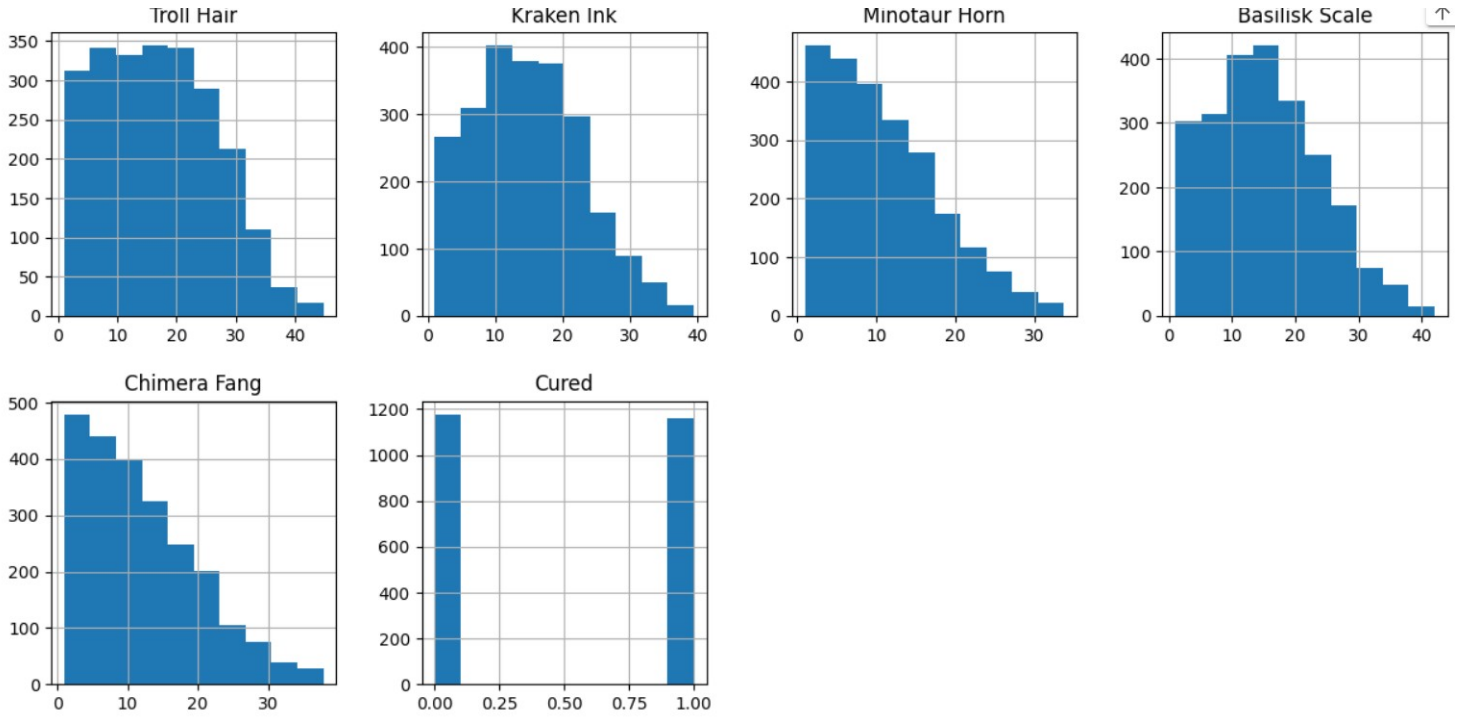

Figure 2: Input Data Distribution Histograms (Part Two)

Figure 3: Input Feature Statistics

	count	mean	std	min	25%	50%	75%	max
Phoenix Feather	2338.0	15.365697	8.669969	1.0	8.500	14.7	21.300	42.1
Unicorn Horn	2338.0	10.946749	7.225162	1.0	5.300	9.5	15.575	34.1
Dragon's Blood	2338.0	16.115654	8.372518	1.0	9.600	15.8	22.000	40.8
Mermaid Tears	2338.0	13.627973	7.545244	1.0	7.600	13.1	19.000	35.8
Fairy Dust	2338.0	15.069504	8.349340	1.0	8.625	14.5	20.700	40.4
Goblin Toes	2338.0	14.157271	7.831476	1.0	7.900	13.5	19.500	37.8
Witch's Brew	2338.0	12.328914	7.709753	1.0	6.325	11.2	16.900	37.3
Griffin Claw	2338.0	14.911206	8.132678	1.0	8.400	14.4	20.500	39.4
Troll Hair	2338.0	16.871685	9.579027	1.0	8.900	16.3	24.000	44.8
Kraken Ink	2338.0	14.890590	8.014197	1.0	8.800	14.4	20.400	39.5
Minotaur Horn	2338.0	10.916125	7.045195	1.0	5.200	9.7	15.375	33.7
Basilisk Scale	2338.0	15.371600	8.559139	1.0	8.800	14.8	21.100	42.0
Chimera Fang	2338.0	12.084003	8.047540	1.0	5.600	10.5	17.275	37.8
Cured	2338.0	0.496578	0.500095	0.0	0.000	0.0	1.000	1.0

2.3 Output Data Visualization

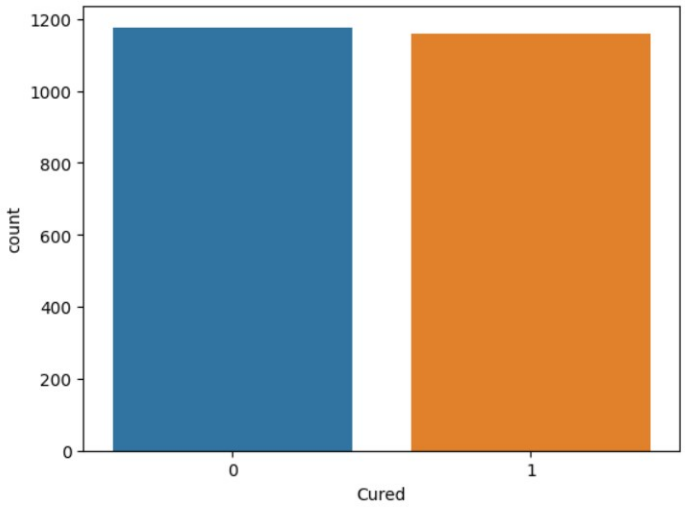

Figure 4: Output Label Bar graph

3 Data Processing

3.1 Data Normalization

The purpose of normalizing a DataFrame is to standardize the range of values in each column, making it easier to compare the relative sizes and magnitudes of different variables.
Normalization is especially useful when the variables in a dataset have different scales or units, as it ensures that each variable contributes equally to the analysis.

Mean Normalization Formula

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Z-Score Normalization

$$X_{normalized} = \frac{X - X_{mean}}{X_{standard_deviation}}$$

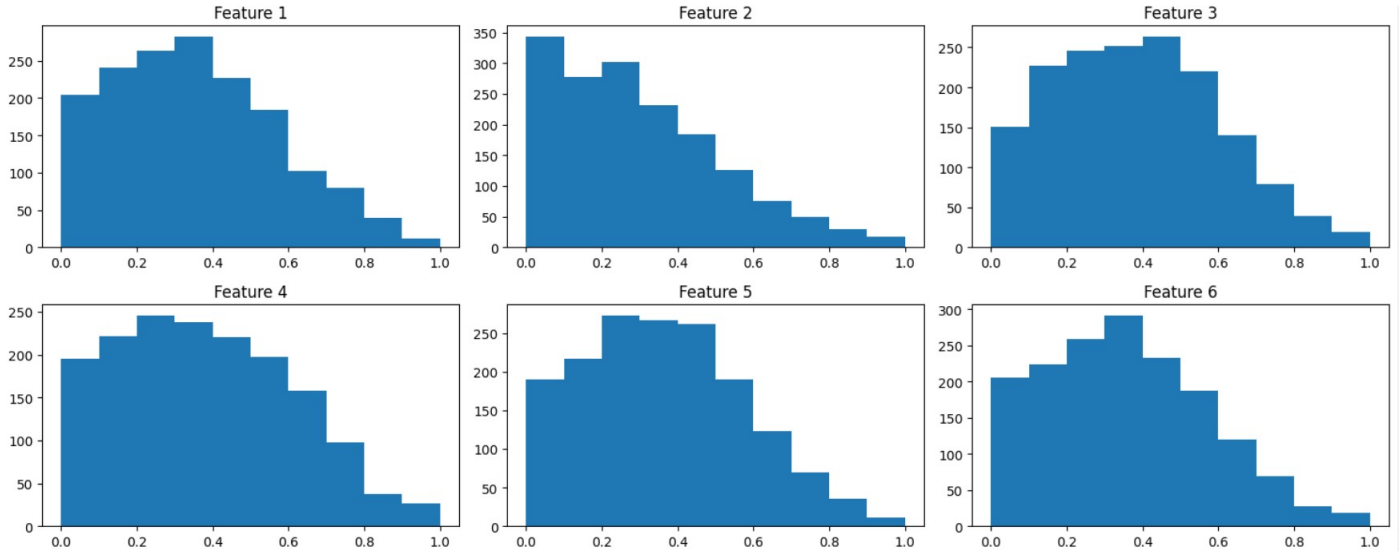

Figure 5: Data Normalization (Part one)

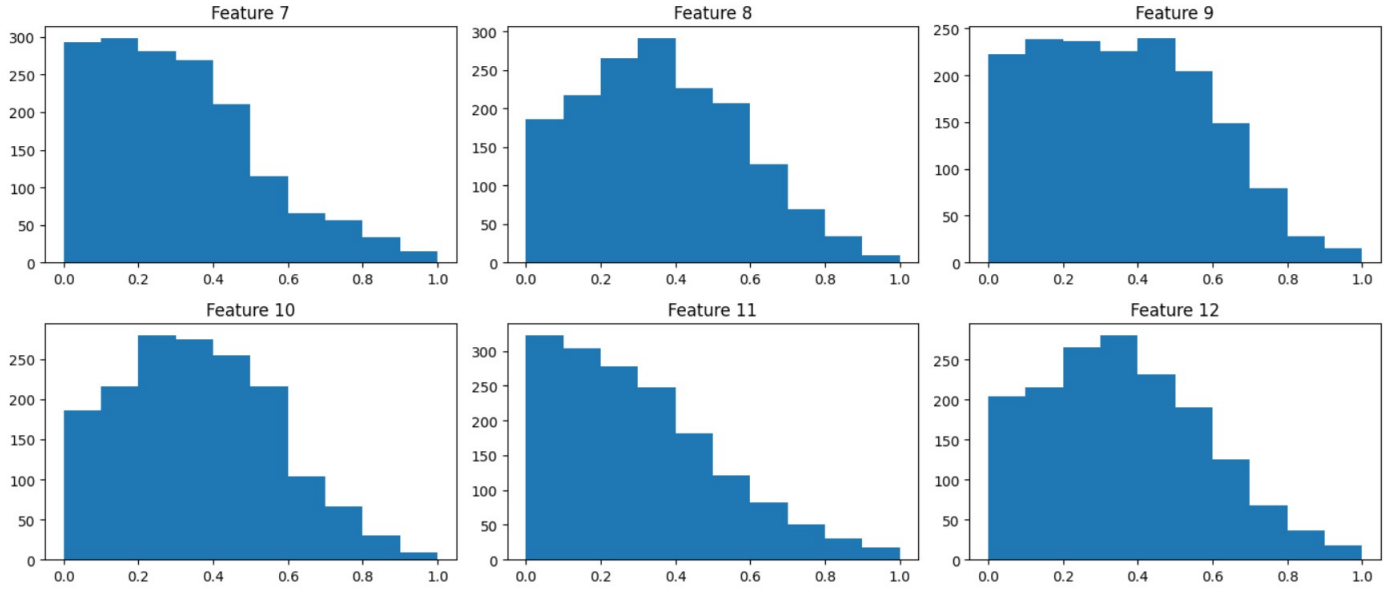

Figure 6: Data Normalization (Part Two)

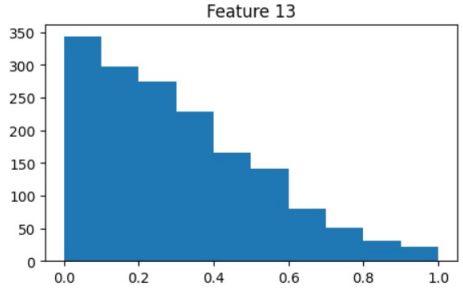

Figure 7: Data Normalization (Part three)

3.2 Data Splitting

A dataset is frequently divided into separate arrays or variables for the input or predictor variables (typically written as "X") and the output or response variable (typically denoted as "Y") in machine learning applications. The features or attributes that are utilized to predict the output variable are often contained in the input variables, whereas the output variable is the target or label that the model is attempting to predict.

4 Modelling

Artificial Neural Network (ANN) was used to create the model. I used a feed forward neural network. Here, data is shuffled and normalized for improving accuracy of the model.

4.1 Varying Neural Network Architecture

First I tried with basic architecture with one input layer, one hidden layer and one output layer. Later I increased the hidden layer from one to two and three.

4.1.1 Model Performance

Hidden	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
0 Layer	1.123	0.4679	7.3964	0.5049
1 Layer	0.3748	0.8588	6.1297	0.5920
2 Layer	0.3415	0.8649	4.4692	0.7018
3 Layer	0.1463	0.9413	6.1287	0.9129
4 Layer	0.0383	0.9938	0.3567	0.9329

Table 1: Performance comparison for different hidden layers

From the above table, layers 0,1 and 2 layers perform almost same. But coming to the layer 3 and 4 they perform more better than the previous layers. We can observe that the best fit model can be used for other architecture.

4.2 Learning Curve of Neural Network

The learning curve typically has the number of training examples on the x-axis and the performance metric on the y-axis, such as accuracy or loss

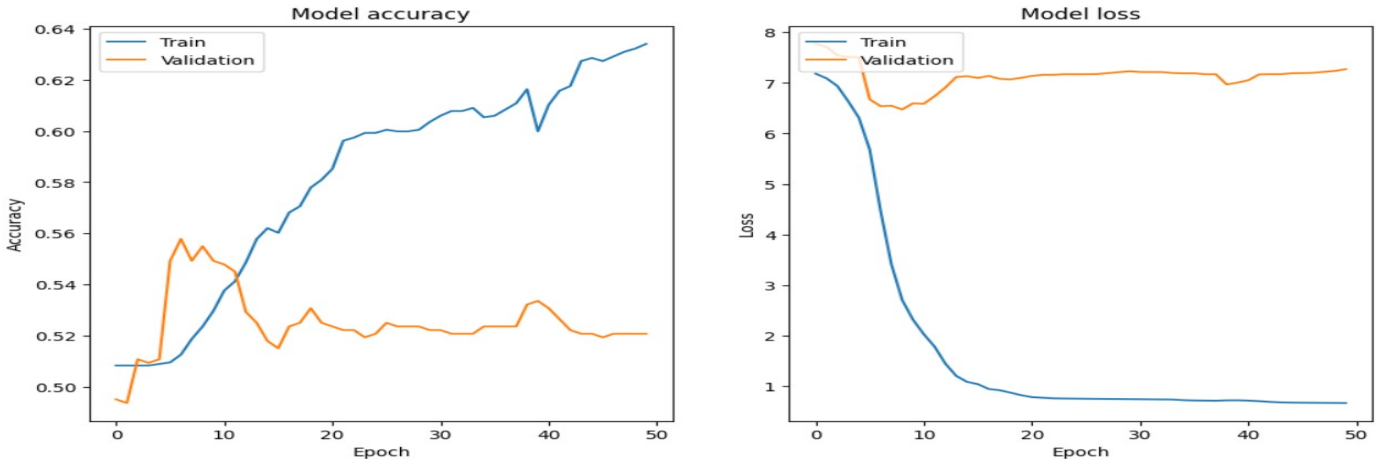

Figure 8: curve showing change in loss/accuracy vs epoch for model

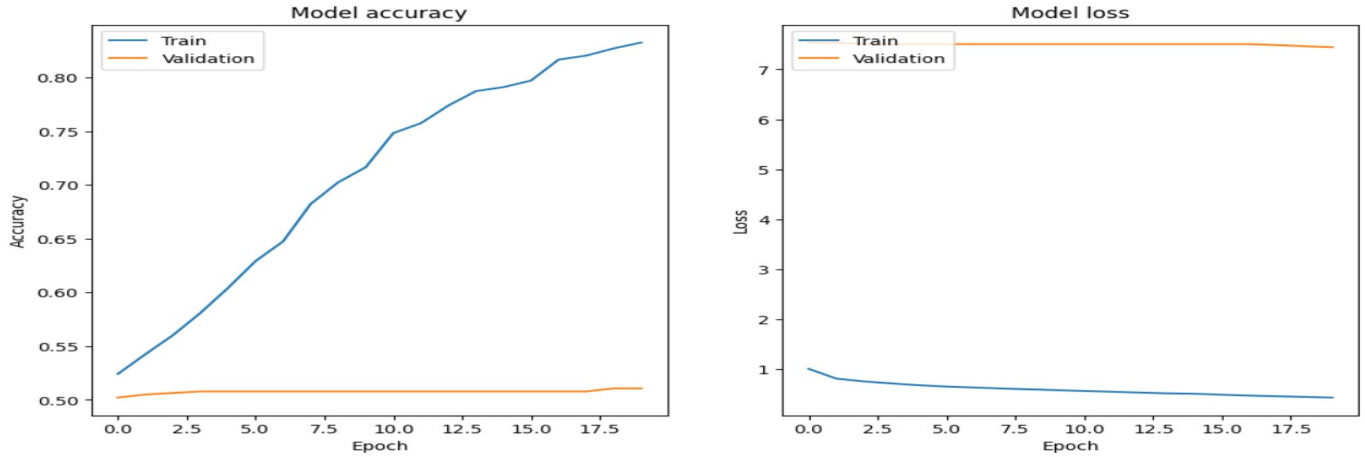

Figure 9: curve showing change in loss/accuracy vs epoch for 8,1 layer

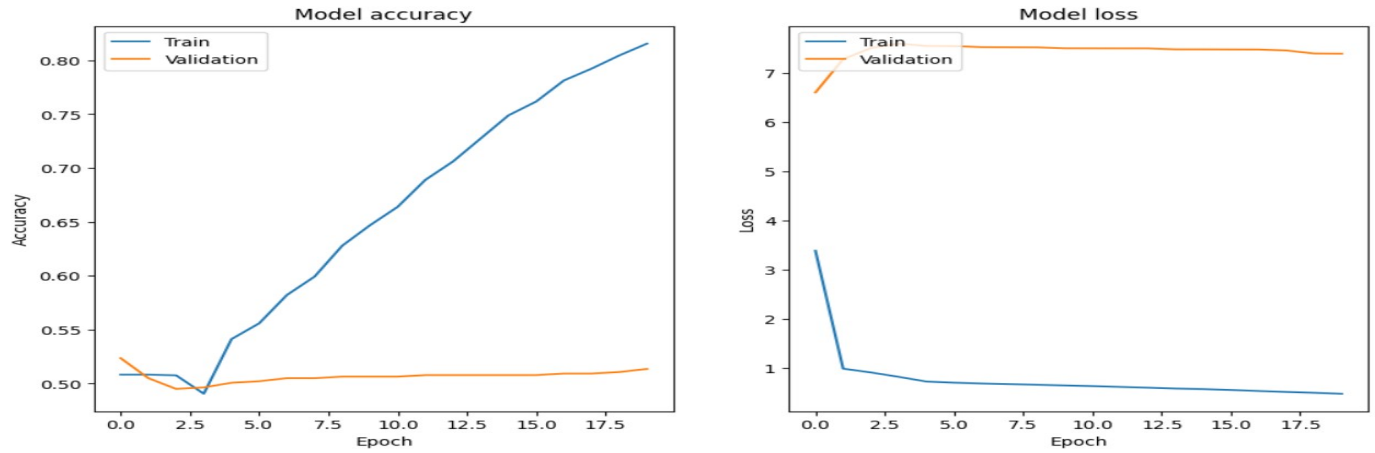

Figure 10: curve showing change in loss/accuracy vs epoch for 16,8,1 layer

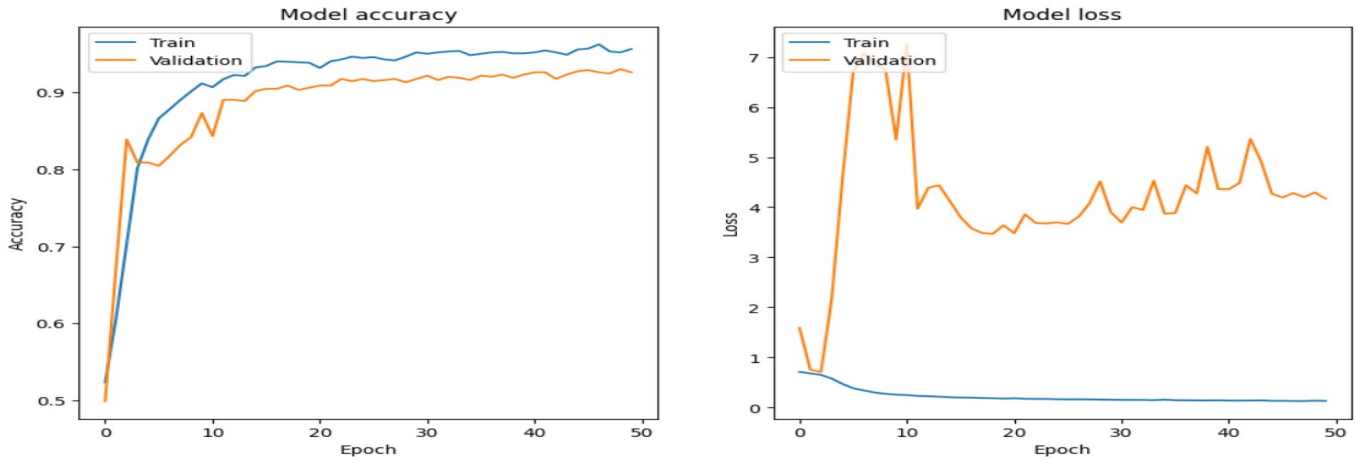

Figure 11: curve showing change in loss/accuracy vs epoch for 32,16,8,1 layer

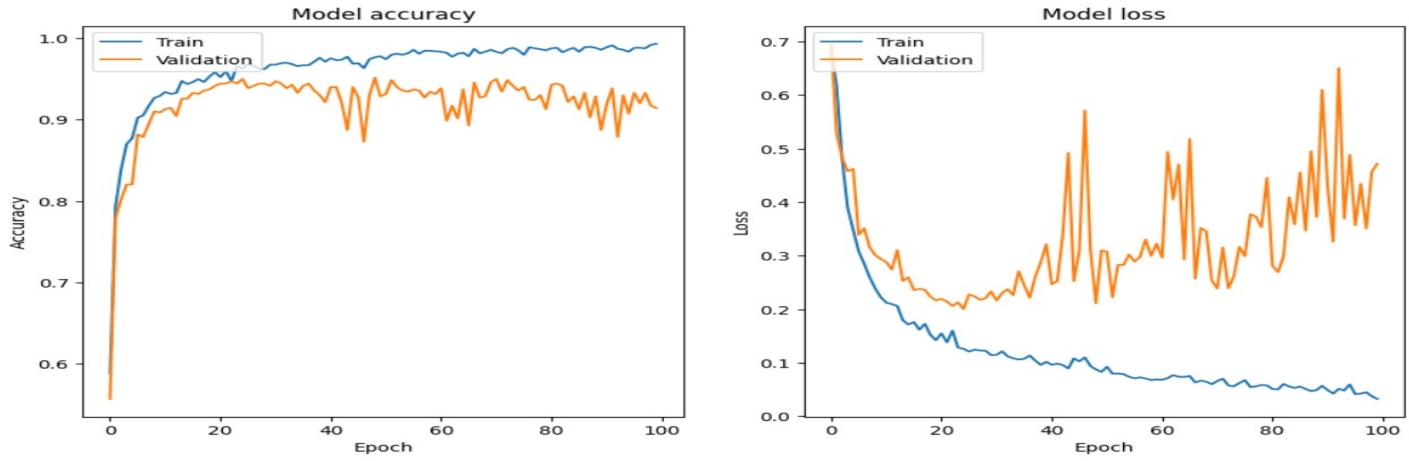

Figure 12: curve showing change in loss/accuracy vs epoch for 64,32,16,8,1 layer

5 Model Evaluation

There are some important metrics to evaluate the model and Here we use precision, recall, F-1 score for evaluation. Given below are some of the metrics of the model architecture.

1. Precision: what proportion of positive identifications was actually correct?
2. Recall: what proportion of actual positives was identified correctly?
3. F1-Score: evaluation metric for classification algorithms, Here I have rounded the values to 2 decimal places.

Model	Precision	Recall	F1-Score
Baseline Model	3.71	0.14	0.26
1 Layer	2.33	0.01	0.03
2 Layer	1.58	0.84	1.10
3 Layer	92.75	92.54	92.75
4 Layer	92.75	92.23	92.14

5.1 Custom Prediction Function

Implemented custom function for prediction got less validation accuracy when built model with multilayer architecture.

5.2 Accuracy Testing

The ROC (Receiver Operating Characteristic) curve is a graphical representation of the performance of a binary classifier that predicts whether something belongs to a certain class or not. The closer the ROC curve is to the top-left corner of the plot, the better the classifier's performance, as it indicates a high true positive rate (TPR) and low false positive rate (FPR).

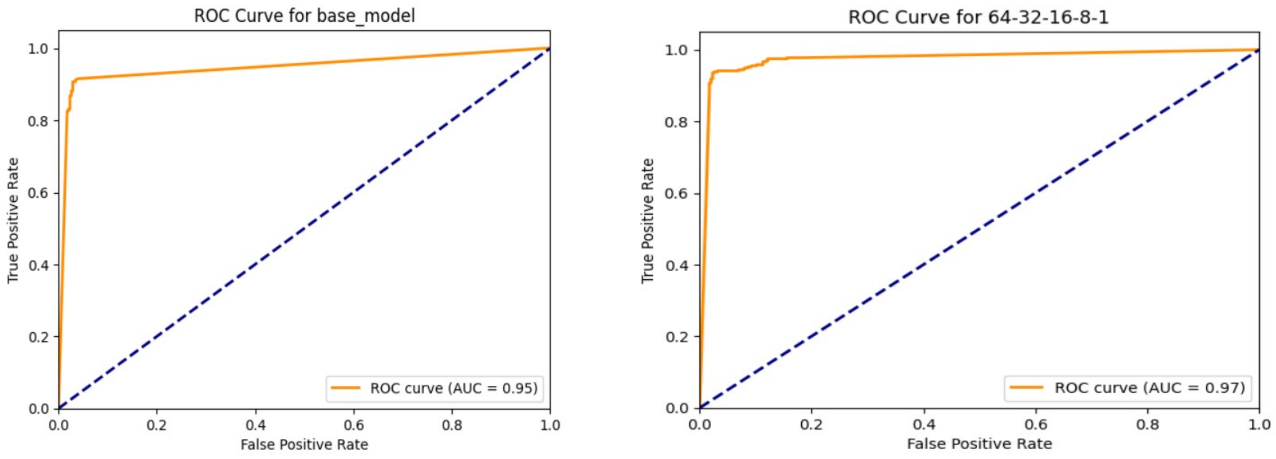

Figure 13: Comparison of model performance

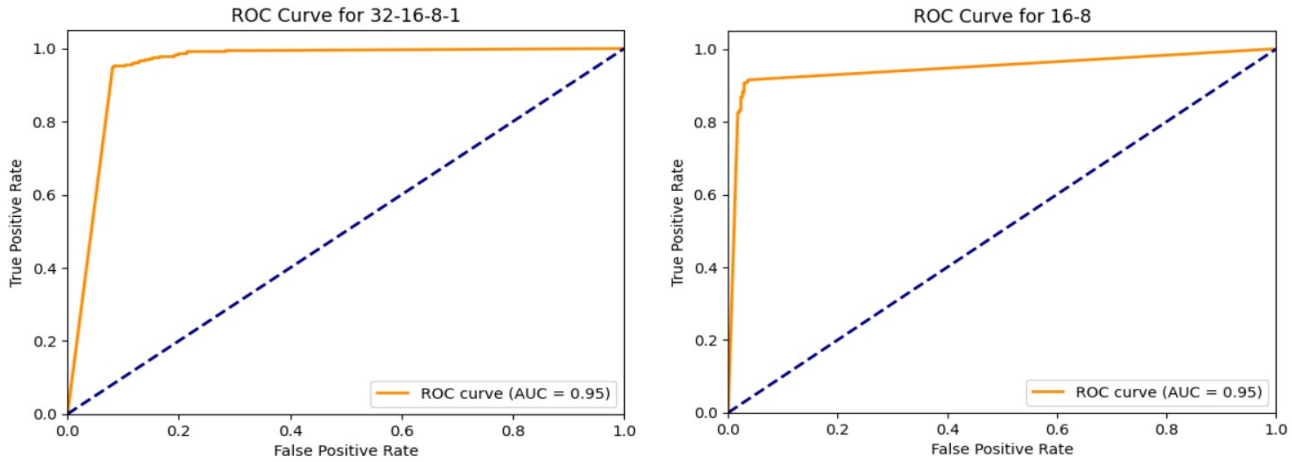

Figure 14: Comparison of model performance

6 Feature Importance Analysis

In previous phase, we built the models that are trained with using multilayer architecture increasing the layers and epochs. But, here we need to analyze the features that are more significance in building the model.

6.1 Verifying important Features

From the below graph, I have identified five important features(Troll Hair,Witch's Brew, Horn Dragons,Phoenix Feather, Chimera Fang) will be used only for the increasing the performance.

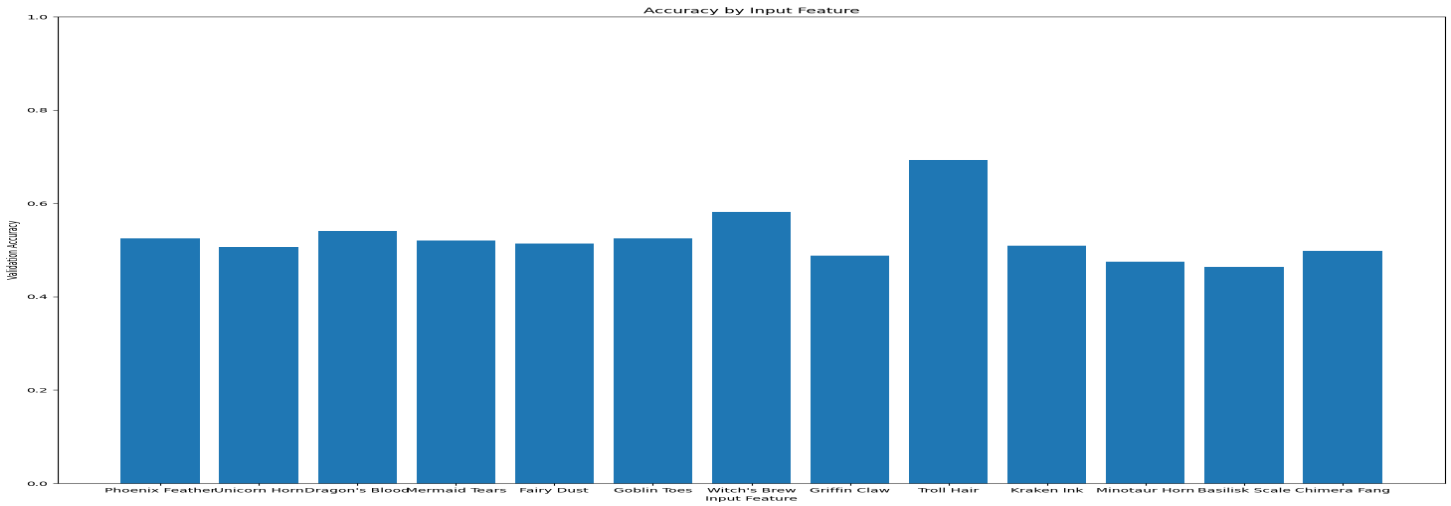

Figure 15: Verifying Important Features

6.2 Removing unimportant Features

In this observed some of the feature accuracies reduced after removing some unimportant features iteratively.

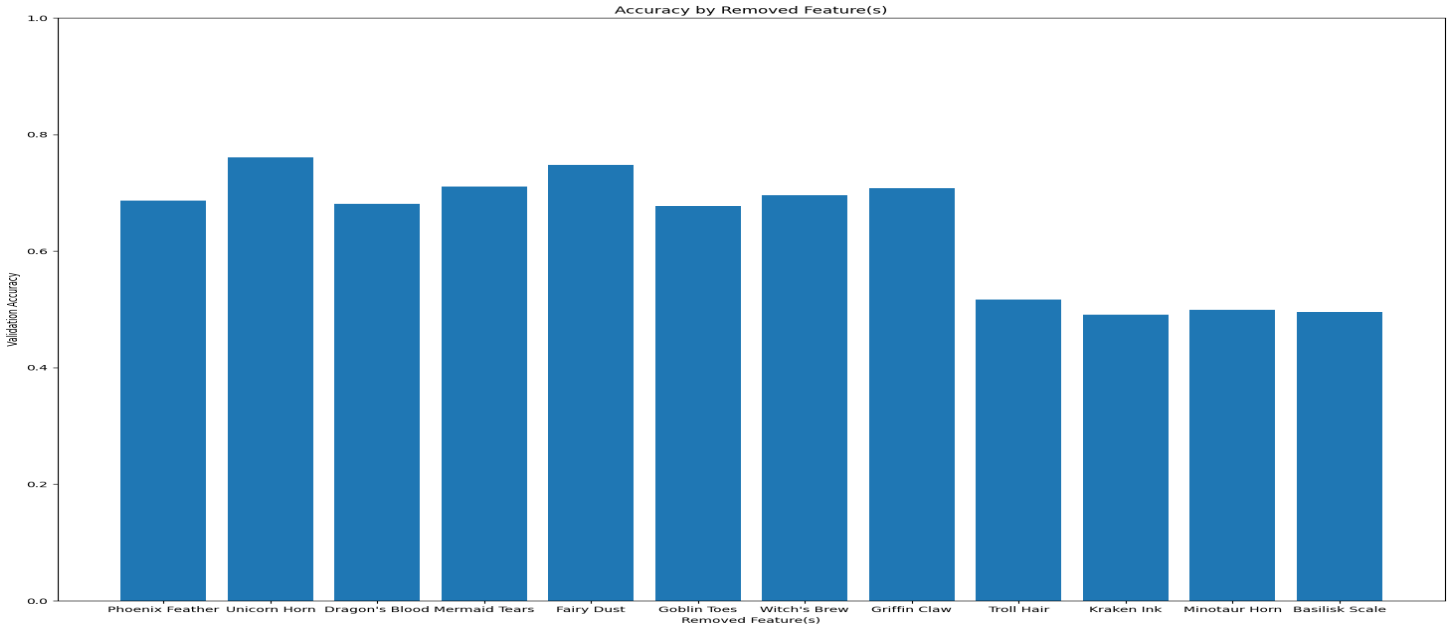

Figure 16: Performance after removing less important features

7 Conclusion

In this project, I developed a neural network model to predict whether the to cure the disease uses different ingredient statistics. Binary classification is a type of machine learning problem where the computer is trained to sort input data into one of two categories. These categories are usually named 0 and 1, or as positive and negative. The aim is to accurately categorize each input data point into the correct group. I have used different activation functions and for better improvement in validation accuracy.