Posionous or Edible?: Model Comparison with Mushroom Classification

Garrett McCue
Final Project Paper
DATA-550: Supervised Machine Learning
garrettsmccue@lewisu.edu

I. INTRODUCTION

The ability to determine if mushrooms are edible in the wild is a vital skill when it comes to foraging for food. The differences between a poisonous mushroom and an edible mushroom can be difficult to determine by an untrained eye, but with the help of machine learning, the ability to safely hunt for edible mushrooms can be made more accessible to everyone. The goal of the project is to evaluate the performance of two machine learning algorithms on classifying mushrooms as poisonous or edible when given a number of attributes that describe the characteristics of each mushroom sample. The dataset[1] used to train and evaluate the algorithms contains 8,124 mushrooms, which are labeled as either edible or poisonous. Each sample contains 22 categorical attributes which included: cap shape, color, odor, gill size, stalk shape, etc. and a binary target variable which represents whether the mushroom is edible or not. The binary nature of the target variables was taken into consideration when deciding which two algorithms to use. The first algorithm that was chosen was the logistic regression because of its use in binary classification and overall simplicity. The second algorithm chosen was a neural network that built off of the simple binary classifier by using many layers containing many logistic regression nodes. The second algorithm was chosen to see if expanding the complexity of the simple binary classifier, used in algorithm one, could result in a significant increase in model performance.

II. PREPROCESSING

Before processing the raw data, the main libraries, numpy, pandas, sklearn, and keras were imported. The data was read from an excel file and stored as a data frame. The target value column was then split from the rest of the data, resulting two new data frames. The attributes were then transformed into dummy variables since every feature is categorical. This is a crucial step because it is allowing the categorical data to be changed into a form that the algorithm will be able to use. The target data was then transformed into one hot encoded targets since they too are categorical. A function was then created that will be used to convert the one hot encoded targets back into the categorical values when evaluating the models performance. In the last step of preprocessing, the data was divided into training and testing data using an 80/20 split.

III. MODELS

Before the models were built, two evaluation functions were created to use when examining each models performance. The first function created generates a plot for the models training and testing loss over time. The second evaluation function is used for generating a confusion matrix after the model predictions, as well as a classification report. The function also returns a data frame for the model with its corresponding accuracy, precision, recall, and f1-score, which is later used for a model performance comparison.

The first model built was the logistic regression. This model used a stochastic logistic regression algorithm which was the sequential model from the keras library. The activation function chosen was the sigmoid function with an L2 regularizer tuned to 0.001. The model was then optimized using the SGD optimizer and categorical cross entropy was used for the loss function with the learning rate set at 0.1. The model was then fitted using batch sizes of 256 over 200 epochs.

The second model built was the neural network. This model also implements the stochastic logistic regression like the previous model, however this model utilizes 3 dense layers instead of being a single perceptron. The first layer of the model is made up of 6 nodes with a ReLU activation function and a L2 regularizer tuned to 0.001. The second layer has 3 nodes also with a ReLU activation function and the same regularizer parameters as the previous layer. The final dense layer uses 2 nodes, since this is a binary classification problem, and a sigmoid activation function with the same regularizer parameters as the previous layers. The model was compiled using the SGD optimizer and a binary cross entropy loss function with a learning rate of 0.1. It was then fit like the previous model, with batch sizes of 256 over 200 epochs.

IV. RESULTS AND DISCUSSION

The first model, logistic regression, performed very well, reporting a training accuracy of 0.999 and a testing accuracy of 0.999 with a loss of 0.048 after the final iteration (Fig. 1). The model resulted in 1 misclassification of poisonous when it was in fact edible out of a dataset with over 8,000 samples. The neural network performed just as well as the logistic regression, with a training and testing accuracy of 1.0 meaning no misclassifications. The loss of the model after the final iteration was 0.028, just beating out the loss of the logistic regression(Fig 2.). Comparing the models using the accuracy, precision, recall, and f1-scores, we find that the neural network outperformed the logistic regression in accuracy, recall, and f1-score (Table 1).

The separation between the two models is almost negligible, but it is important to remember when it comes to deciding if something is poisonous or edible any small difference is a matter of life and death. Putting that aside, the logistic regressions simplicity makes it computational more favorable while still yielding near identical results. It would be necessary to expose more unseen data to the model in order to ensure no overfitting has occurred and to get a more confident conclusion that using a neural network instead of a simple logistic regression does not result in a better performing model.

A. Figures and Tables

Logistic Regression Model Performance

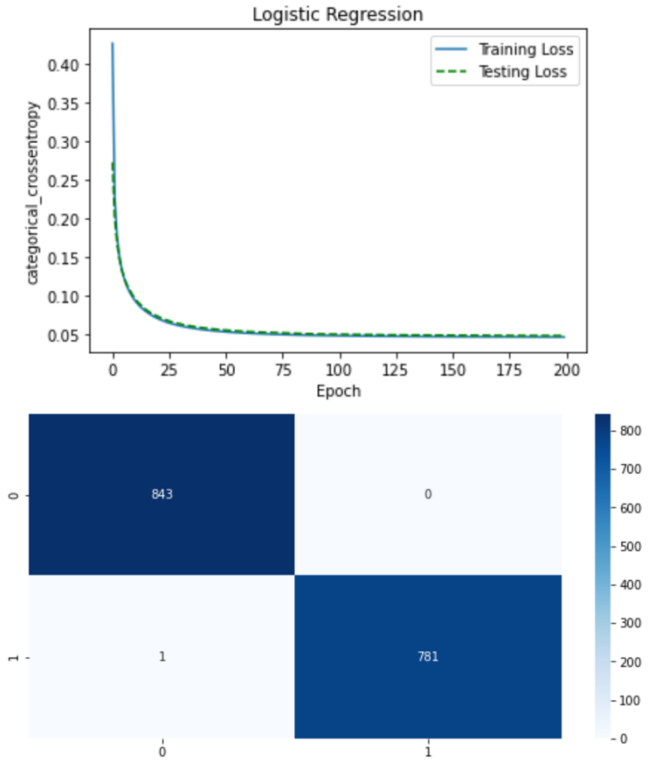

Fig. 1. Loss of the logistic regression model over 200 epochs, and the corresponding confusion matrix

Neural Network Model Performance

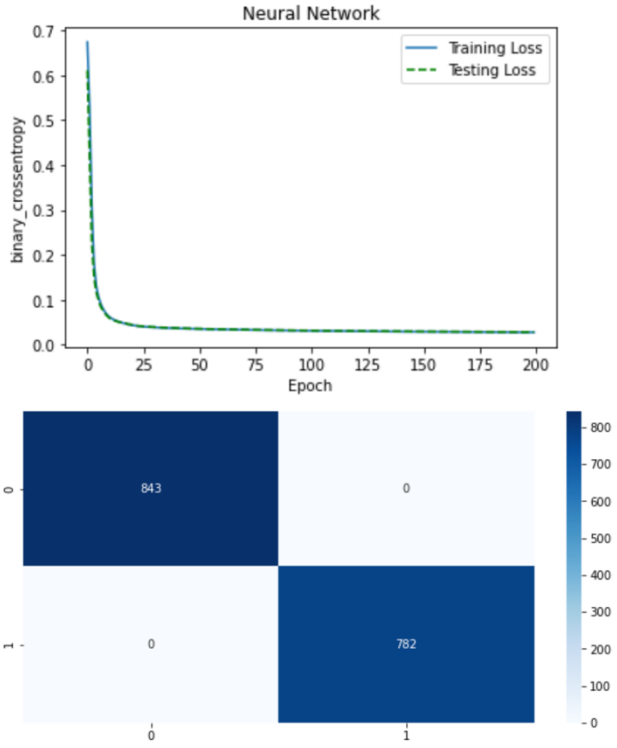

Fig. 2 Loss of the neural network model over 200 epochs, and the corresponding confusion matrix

Model Comparison Table

	Logistic Regression	Neural Network	Best Model
Accuracy	0.999385	1.0	Neural Network
Precision	1.000000	1.0	Logsitic Regression, Neural Network
Recall	0.998721	1.0	Neural Network
F1-score	0.999360	1.0	Neural Network

Table 1. A table comparing each model's performance metrics with the top scoring model in the final column.

REFERENCES

[1]https://www.kaggle.com/uciml/mushroom-classification