Project 8: Decision Trees and Neural Networks

Drew Watson and Treyson Grange Slides Repo

Introduction

For this project, we analyzed data from the MAGIC Gamma Telescope which holds data of high energy gamma particles using the Cherenkov Gamma telescope. Our analysis aims to predict whether the telescope is looking at a gamma signal or a hadron background.The stakeholders of this project are scientists studying Cherenkov radiation who wish to use a machine learning model to differentiate gamma ray signals from hadronic shower backgrounds. This is important and relevant to our shareholders, as if we can identify these signals automatically, we can save scientists lots of time.

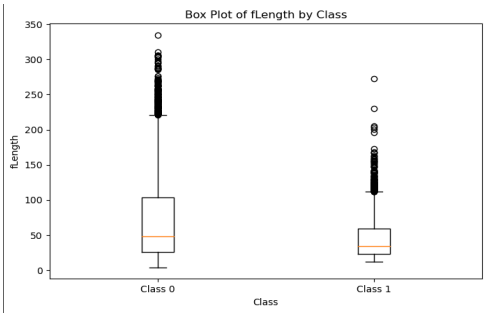
Dataset

We found our dataset, titled MAGIC Gamma Telescope from the UV Irvine Machine Learning Repository. The dataset was specifically grouped as a classification task, making it perfect for this project. This dataset has 19020 instances, with a ratio of 2:1 ratio of gamma signals to hadron backgrounds. The dataset includes features such as major and minor axis lengths of ellipses, logarithm of pixel content sum, pixel ratio measures, asymmetry distance, third moment roots along axes, angle of major axis, distance from origin to center, and object classification (gamma signal or hadron background). The dataset has 10 continuous, quantitative input features and 1 binary categorical variable “class”, which takes a value of either “h” for hadron background or “g” for gamma signal. This makes it perfect for classification using neural networks and decision trees.

Analysis Technique

Decision Tree

We visualized the relationship between each feature and the target variable using box plots, aiming to gain insights into how each feature contributes to the classification task. From these plots, we hoped to see any visual correlations between our features based on gamma or hadron. The more different the two box plots for each feature are, the more important that feature is. However, after plotting all of them, we noticed that all box plots were too similar to really specify each variable as all important. This is expected, because if there was one variable that basically told us what we were looking at, it would make the task of classification trivial. So we moved on, choosing all but our target for our features. From here, we tried different values of max_depth, and scored our model against our training data.



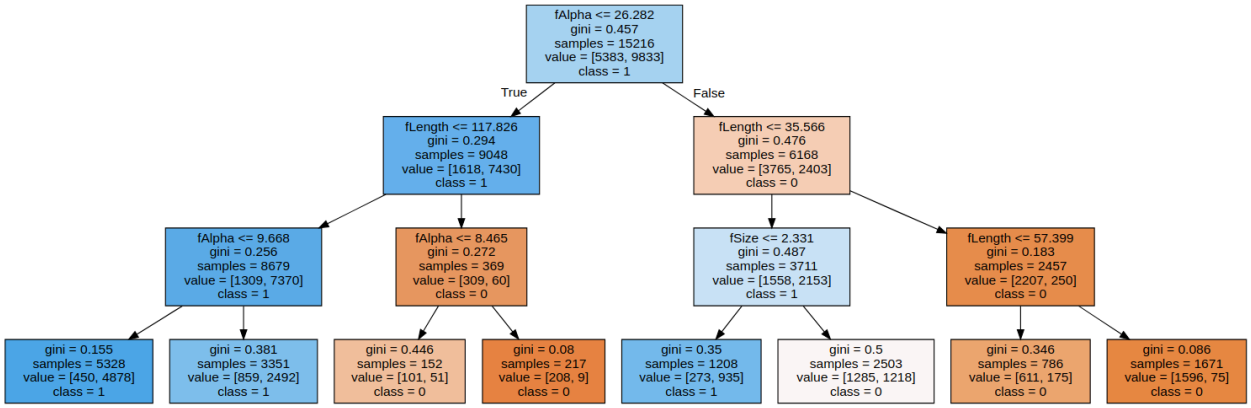
Neural Network

For feature selection for the neural network, we used scatterplots and Pearson correlation tests to determine if there were features which have low correlation with the target variable. We found that the features fM3Trans, fConc, and fConc1 had low correlation with the target variable. However, removing these features from our neural network did not result in better precision, recall, or f-score, so we opted to keep all of the features because neural networks gracefully handle large numbers of input features. We then defined three neural network architectures, **Model A**, **Model B**, and **Model C**. Model A consists of 10 input nodes (1 for each feature), 5 hidden layers with 30, 60, 30, 10, and 5 nodes respectively, and 1 output node. Model B consists of 10 input nodes, 6 hidden layers with 50, 100, 300, 100, 50, and 10 nodes respectively, and 1 output node. Model C consists of 10 input nodes, 101 hidden layers with the first 100 layers having 20 nodes and the 101st layers having 10 nodes, and 1 output node. Visualizations for each model are shown in Appendix 1 below. All models were trained for only 10 epochs to reduce the training time.

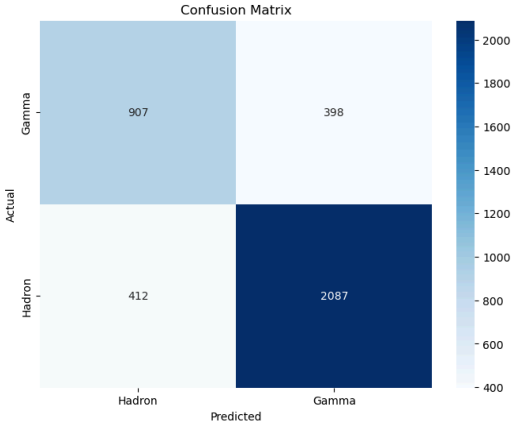
Results

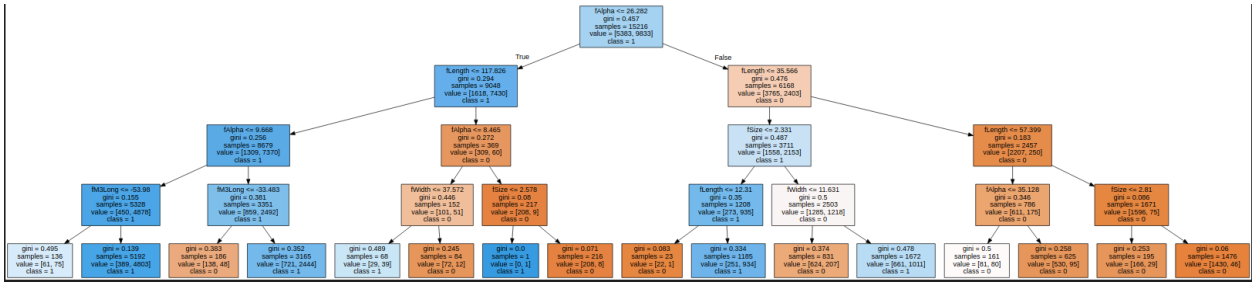
Decision Tree

We studied two different decision trees, one at max depth 5, and the other at max depth 3. Let's first investigate the first tree.

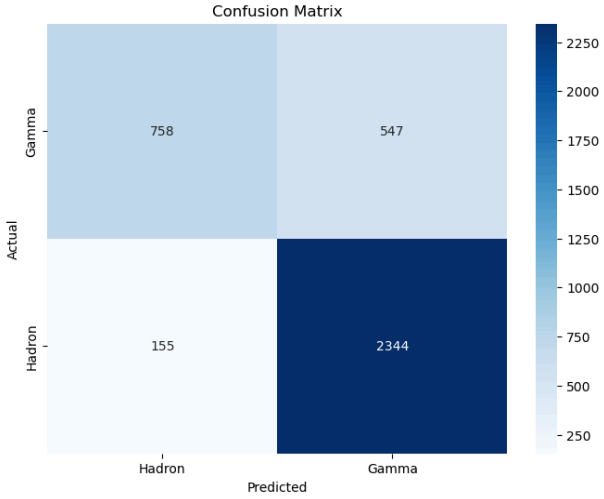


Shown above is a diagram of our decision tree. The decision tree first decides on fLength, and descends into choices with fAlpha. This is a pretty basic decision tree, with a pretty good F1 score. After testing, and calculating our confusion matrix we end with a F1 score of .83747. This is pretty good, even though our leaf nodes feature values that are a bit too similar, for example in the 6th leaf node, we see that there are 1285 gamma signals that fall here, and that there are 1218 hadron backgrounds. Also, we notice that in the second level, the different sides of the tree split on different attributes, meaning the algorithm makes the best choice recursively for each branch. Then we take a look at our next max_depth of 4.



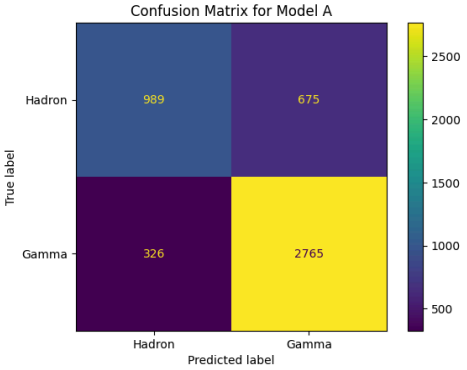


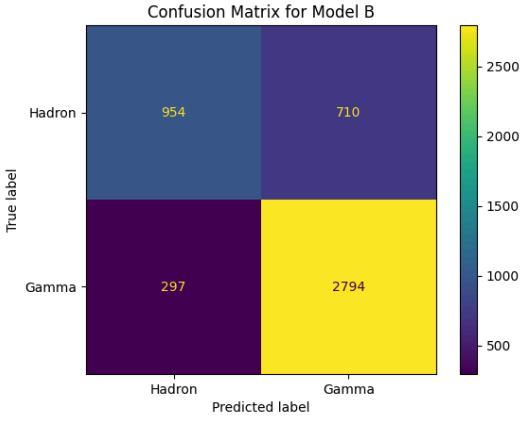
We can instantly note that the decision tree algorithm still kept the same decision boundaries for each level, but adding leaf nodes to the current leaves. This decision tree has a slightly better F1 score of 0.8697 And according to our confusion matrix we see that we are suffering the most when we predict Hadron when it is in actuality Gamma, or false positives. In fact, it is closer to the count of True positives. Our overall performance improved as we increased our max_depth, however as our max_depth increases, our performance does not go up enough to justify the extra work our model has to put in. And at a certain point it makes decisions that have little effect on anything. This is overfitting, and by keeping our max depth to around 3-4 we minimize it.



Neural Network

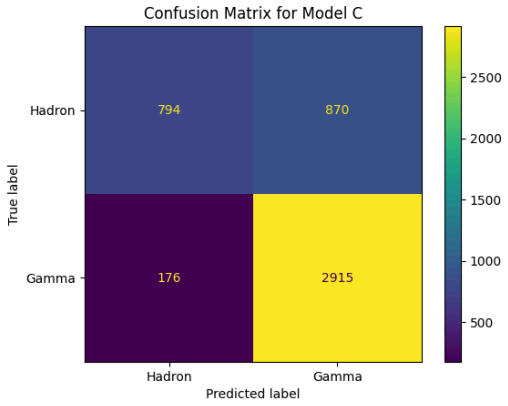
Despite having very different architectures, all three of our models perform very similarly. Model A had a precision of 0.803, a recall of 0.895, and an f-score of 0.847. Model B had a precision of 0.797, a recall of 0.904, and an f-score of 0.847. Model C had a precision of 0.770, a recall of 0.943, and an f-score of 0.848. Notice that each model has a recall which is much larger than its precision. This means that, just like the decision tree, our neural network models are much better at predicting gamma signals than predicting hadron backgrounds. This is beneficial for stakeholders if they are more interested in catching all of the gamma signals, but is detrimental if stakeholders are more concerned with misclassification of background signals. Confusion matrices for each of the neural network models are shown in figures to the right and below.





It should be noted that our results suggest that deeper models are more biased towards classifying a data point as a gamma ray. Model C is the most biased towards classifying

gamma rays, with the highest recall and lowest precision of the three models. During some fits of Model C, the model classified ALL the signals as gamma.



Regardless of the number of epochs used, and the number of hidden layers and nodes per hidden layer used, we failed to achieve an f-score of greater than 0.90. Our theory is that this is caused by the class imbalance in the data. We postulate that this might be remedied by supplying a bias to the network so that it favors classifying hadron background. We did not see any correlations between the edge weights of the neural nets and our decision trees.

Technical

To prepare the dataset for analysis, we first converted our target variable to be binary for easier use in our models. We gave hadron backgrounds a value of 0 and gamma signals a value of 1. We also standardized each input feature before supplying them to our models for the fitting process.

Decision trees are designed for classification problems, and are compatible with continuous quantitative input features, so they are a suitable model for this binary classification problem. Neural networks can be used for regression or classification, but there are choices we made for our networks that make them more suitable for this binary classification problem. For our neural network models, we used a sigmoid activation function for the output layer, and a binary cross entropy loss function. The reason for this is because the sigmoid activation function on the output layer is best for classification (because it outputs a probability), and binary cross entropy is a loss function catered specifically towards binary classification problems. All hidden layers use the ReLu activation function, because it is the most efficient and is not as susceptible to overfitting.

For both our decision tree and our neural networks, we experimented with several different features, but the best results were obtained by using all the features. During our analysis of the decision tree model, we experimented with different max_depths in order to see its effect on bias and variance, noticing a decrease in bias but an increase in variance as max_depth increased. During our analysis, we tried a plethora of neural network architectures, some with very large numbers of nodes per layer, and some with a large number of hidden layers. We even tried creating a random number of nodes per hidden layer. All of these models resulted in f-scores no better than 0.85. We also experimented with binary focal cross entropy and mean squared error for loss functions, both of which performed no better than binary cross entropy. As stated in the results section above, we might have been able to overcome this plateau in f-score by supplying a bias to the network so that it favors classifying hadron background. One way to do this is to adjust the threshold we used to classify the output of the neural net.

Appendix 1

This appendix contains the visualizations of each of the neural network models used in this project.