HIGH SCHOOL EDITION
Journal of Student Research
Volume 13 Issue 3 (2024)

Classification of Stellar Formations with Machine Learning

Alexander Miller

Woodbridge High School, USA

ABSTRACT

Among the stellar formations observed with modern telescopes are stars, galaxies, and quasars. Depending on what formation they are, they emit different types of light. We use data on the light emitted by each formation emitted to train a neural network, which will classify each formation as either a star, galaxy, or quasar. We explain our process as we set up each element of the neural network.

Introduction

In this article, we utilize machine learning to classify different stellar formations based on the light we observe from them. Machine learning is a subfield of computer science that uses data to create models that can be used to make predictions [1, 2].

We consider the Sloan Digital Sky Survey, which has 100,000 observations of stellar formations, each of which can be classified as a star, galaxy, or quasar. All formations are observed as point sources of light. While the stars are true point sources, each galaxy contains 100 million to a few trillion stars, but appear as a point source. Quasars are a subtype of active galactic nuclei that formed as dust and gas fell into supermassive black holes, creating immense frictional forces and releasing electromagnetic radiation. The nearest quasars observed are all far away, as they only existed in the earlier universe and their light is just now reaching us [3].

Our goal is to create a model that can categorize each observation as a star, galaxy, or quasar. In order to distinguish between these observations, we must consider the many different types of light received from each object. This is a supervised learning problem, where the data is a collection of examples with features as the input and labels as the output.

In our case, the examples are the stellar observations, the information about their light forms the features, and the type of stellar formation is our label. We use six features, quantifying the ultraviolet, green, red, near infrared, and infrared light, as well as the redshift from each observation. The machine learning model we create will be a function that inputs such light information and outputs predictions on what type of astronomical formation produced that light.

Table 1. Five examples from our Data Set

Ultraviolet	Green	Red	Near-infrared	Infrared	Redshift	Class
23.48827	23.33776	21.32195	20.25615	19.54544	1.424659	Quasar
21.46973	21.17624	20.92829	20.60826	20.42573	0.586455	Quasar

22.24979	22.02172	20.34126	19.48794	18.84999	0.477009	Galaxy
24.40286	22.35669	20.61032	19.46490	18.95852	0.660012	Galaxy
21.74669	20.03493	19.17553	18.81823	18.65422	-0.000008	Star

In Section 2, we describe how we split and normalize our input data to prevent incorrect biases within our model. In Section 3, we explain the conversion of categorical variables into numerical data that we need to form the output of a function. In Section 4, we create our neural network. In Section 5, we show how we get our neural network to output predicted probabilities of each possible label. In Section 6, we introduce categorical cross entropy error and use it to train our neural network. We conclude our article in Section 7 with the results from our model.

Splitting and Normalizing Data

We split our data set into a training set and a validation set, with half of the dataset going to each.

We give our model all the features and labels from the training set, so it can learn from the training set.

Afterwards, we use the validation set to test our model's accuracy on predicting the formation type of a list of observations that it has never seen before. The validation set is important because it will detect and avoid overfitting. Overfitting is when our model begins detecting patterns in the training data that are merely coincidental, rather than the legitimate patterns we are interested in studying. This usually happens when we give our model too much freedom to make complex functions that aggressively fit the training data rather than finding the overall pattern, making our model seem better than it actually is.

Any coincidental patterns that appear in the training data will not also be in the validation data. Thus, we get from the validation set an accurate statistic on the true efficacy of our model. If our model performs significantly worse on the validation data than on the training data, it is being affected by overfitting, and we should reduce the number of parameters that can be changed by the model.

It is important to note that the exact numerical values given to each feature are arbitrary, and each feature can have a different mean and standard deviation across all examples. This can create bias in a model to prefer features that have a larger swing, since they generate larger numbers in the model's functions. Thus, we normalize our data by scaling and shift our data for each feature so that the mean value of that feature is 0 and the standard deviation is 1. While the model could have corrected for this, normalizing the data first starts all features off as equal, reducing the number of things our model must do, and making it easier to understand if we wanted to examine the details of the model's function.

Table 2. Five examples with normalized features

Ultraviolet	Green	Red	Near-infrared	Infrared	Redshift	Class
0.625302	1.328267	0.903717	0.666301	0.439674	1.160513	Quasar
-0.271405	0.267333	0.691473	0.866604	0.938147	0.013395	Quasar
0.075125	0.682318	0.374973	0.229295	0.045868	-0.136386	Galaxy

1.031596	0.846731	0.520038	0.216188	0.107324	0.114062	Galaxy
-0.148370	-0.292854	-0.253536	-0.151679	-0.064989	-0.789203	Star

Multi-Classification of Stellar Formations

Each observation belongs to one of three categories: star, galaxy, or quasar. This format of information is not very helpful to our model, as it is not a numerical output that our model can give. Furthermore, we cannot simply assign star to '0', galaxy to '1', and quasar to '2' because such a labeling would put the galaxy prediction in the middle of a scale from star and quasar, which is not necessarily accurate, and we want our model to be able to independently evaluate which of the three formations to predict.

In machine learning this is known as a multi classification problem. To properly reformat our labels, we give each possible label value its own variable indicating whether or not the label is that value, with '0' for no and '1' for yes. In our case, each label becomes three variables, with one indicating whether the formation is a star, one indicating whether it is a galaxy, and one indicating whether it is a quasar.

Table 3. Labels after replacing *star*, *galaxy*, and *quasar* with a 1 in their respective columns

Class	Star	Galaxy	Quasar
Quasar	0	0	1
Quasar	0	0	1
Galaxy	0	1	0
Galaxy	0	1	0
Star	1	0	0

Our model will then be a function that predicts based on the features a value for each of the three labels. Instead of just indicating '0' or '1' for each possible label, it will give a probability of each possible label. Using $\hat{y}_1$, $\hat{y}_2$, and $\hat{y}_3$ to denote our three labels as a probability vector, and using x_1, x_2, x_3, x_4, x_5, and x_6 to denote our features, our model can be represented as the function $(\hat{y}_1, \hat{y}_2, \hat{y}_3) = \hat{y}(x_1, x_2, x_3, x_4, x_5, x_6)$. When evaluating the accuracy, we find the largest $\hat{y}_n$, and predict that label as the true label.

Neural Network Creation

A neural network is a sequence of functions, organized in layers, that are applied to the features in order to produce predicted labels. The first and last layers of any neural network are the input and output layers.

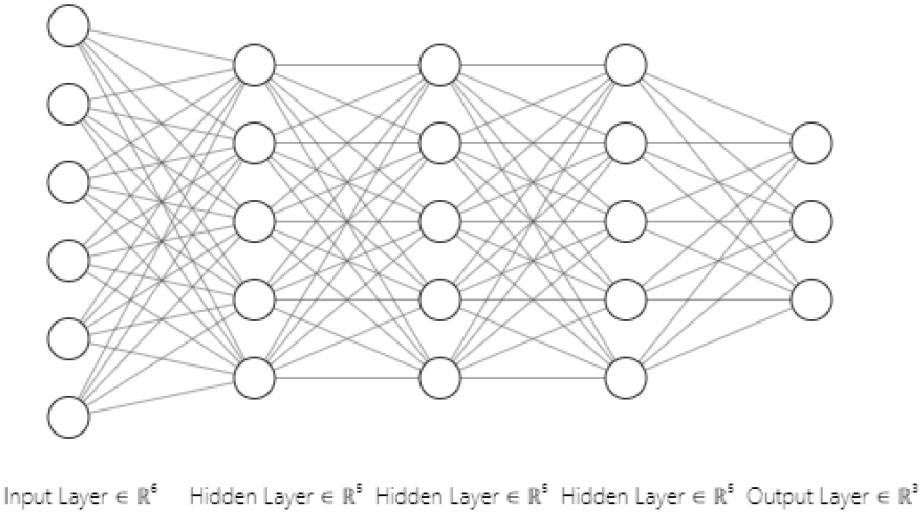

Figure 1. Diagram of our Neural Network

The input layer contains the normalized features as obtained in Section 2. Each of these features is referred to as a node in the input layer. We have six nodes in the input layer, one for each of our features.
The layers in between the input and output layers are referred to as hidden layers, and each contain some number of nodes and type of function that we can select when we create our model. We are using dense layers, where each node is a function comprised of two parts: a linear function of all the nodes in the previous layer, with different weights assigned to different nodes, and a nonlinear activation function applied after the linear function to allow our model to develop nonlinear behavior.

Thus, the value of each node is:

$A(w_1x_1 + w_2x_2 + \dots w_nx_n + b)$

Where x_n represents the nth node in the previous layer, and A represents the activation function. Each w_n and b are parameters that our model can tweak during training.

The last layer is called the output layer. This contains the predicted labels. Like the other layers, each node in the output layer is also a function of all the nodes in the previous layer.

In our stellar classification problem, we use three hidden layers of five nodes each. In our first and third layers, we use the tanh activation function, and in our second layer, we use the relu activation function. The details of these are not too important, as all they do is give our model flexibility to create different relationships, but we will go over the final activation function in the next section.

Conversion to Probability Values

The input to our final activation function can have a large range of positive and negative values, but as described in Section 3, we want to create a probability for each possible label. The probabilities should all be positive and should sum to 1.

Thus, we use the softmax function as our final activation function. The softmax function takes in a vector of input and outputs a vector with the associated probabilities. It exponentiates each input value x_n as a power of e, making each value positive, and then divides each value by the sum of all values so that the values add up to 1.

Training the Model

We want to select the optimal parameters in our neural network to minimize the error of its predictions, but first we must choose a definition of error. We will evaluate errors based on the probabilities given to each label and how much they differ from the true values, which will be either 0 or 1. To do this, we use the Categorical Cross Entropy Error function:

$$CE(y, \hat{y}) = -[\ y_1\log(\hat{y}_1) + y_2\log(\hat{y}_2) + y_3\log(\hat{y}_3) + \ldots\ y_n\log(\hat{y}_n)\ ]$$

The error is minimized when each y_n and $\hat{y}_n$ are close in value, reaching zero when all $y_n = \hat{y}_n$ and approaching infinity when $\hat{y}$ gives a high probability of the wrong guess. When we train our model, it constantly tweaks the parameter values in the direction that will give lower error.

We train our model for 100 epochs, where each epoch represents one pass through the entire training dataset of 50,000 observations. While we are training, the error decreases as our model continually selects better parameters, but the error levels out by 100 epochs.

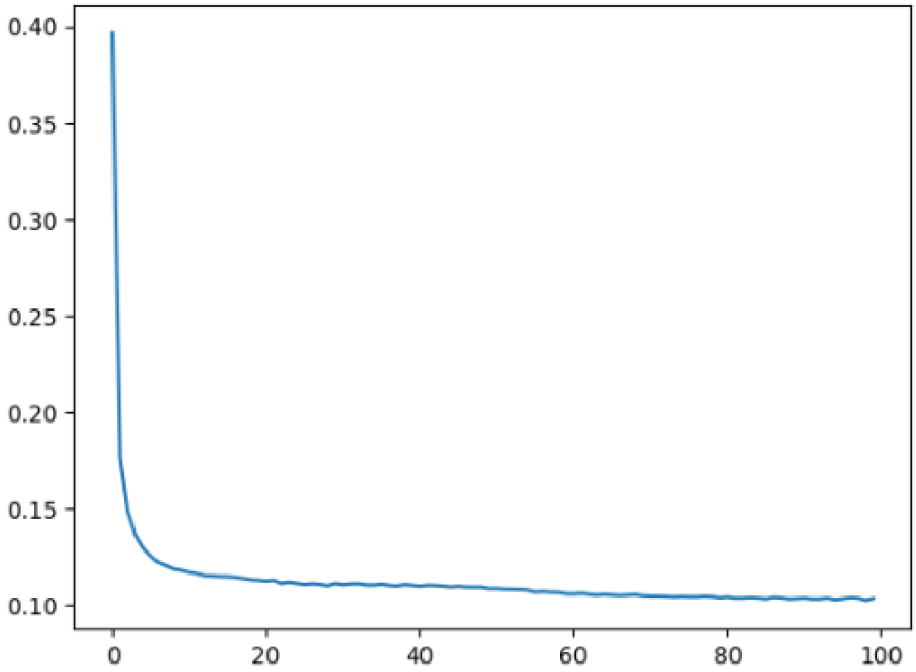

Figure 2. Categorical Cross Entropy Error of our model (y axis) as a function of epoch number (x axis)

After minimizing the cross-entropy error, our model is well fit to give predictions with probabilities close to the true labels. Thus, it should also have high accuracy because most of the time it will be closer to the true label values (either 0 or 1 for each label) than the incorrect values and will give a correct prediction.

Results and Conclusion

Our model has 97% accuracy on the training data and 97% accuracy on the validation data, meaning it made the correct prediction 97% of the time. The equivalence of accuracy between training and validation data shows

that overfitting was not an issue, which is to be expected because we had 50,000 data points in our training set and were thus not very vulnerable to random noise and coincidental patterns.

Overall, our model serves as a function that inputs six different variables that describe the light from each object, and outputs labels with the predicted probabilities of each type of formation. Since we have 113 parameters, we will not cover the actual function found by the neural network to predict stellar formation type. Our neural network was able to detect patterns in the six-dimensional input space, and effectively predicted the type of stellar formation that produced each observation of a light source.

Acknowledgments

I would like to thank my advisor for the valuable insight provided to me on this topic.

References

[1] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. The elements of statistical learning, volume 1. Springer series in statistics New York, 2001.

[2] Andriy Burkov. *The hundred-page machine learning book*, volume 1. Andriy Burkov Canada, 2019.

[3] Schade, David. The Space Distribution of Quasars. Annual Review of Astronomy and Astrophysics, 1990. https://www.annualreviews.org/content/journals/10.1146/annurev.aa.28.090190.002253