HIGH SCHOOL EDITION
Journal of Student Research
Volume 13 Issue 4 (2024)

Classifying the Objects of the Universe with Machine Learning

Justin Wu[1], Abdulla Kerimov[#] and Steve Szabados[#]

[1]Greenhill School, USA
[#]Advisor

ABSTRACT

With the increase in space exploration and the search for other planets by many across the globe, identifying astronomical objects is an incredibly important task. It is one that will allow us to potentially find habitable planets around stars or asteroids with important minerals. Thus, the goal of my research was to discover the best way to use machine learning in order to identify these celestial objects. The dataset from the Sloan Digital Sky Survey from 2017 was used in this study. The key features of this data were the photometric values of each object, its redshift, and its label as a Galaxy, Quasar, or Star. Different baseline models were trained, tuned, and tested including logistic regression, decision tree, random forest, ridge classifier, and neural network. The best performing model was the tuned random forest model which had the highest f1-score, precision, and accuracy. The average accuracy was 99%, the f1 score for galaxies was 99%, for quasars was 97% and for stars was 100%. Different neural network architectures were trained and tested as well. However, none of the designed architectures could beat the hyperparameter tuned random forest. Thus, I achieved my goal by discovering that the random forest was incredibly accurate in identifying astronomical objects. This model could be potentially used for aiding astronomers in identifying objects across the universe.

Introduction

When viewing celestial objects from Earth, they often all appear as simply points of light and are incredibly hard to identify even with data from telescopes. These objects are vital in the study of our universe and could teach us about the formation of our own galaxy and universe. Thus, in our research, we did the important task of differentiating objects as stars, quasars, and galaxies for astronomers. A galaxy is a large collection of stars, dust, and gas which can appear in many different shapes. A star is a massive, illuminated collection of hot gas which glows due to internal nuclear fusion. Stars usually appear in a spherical shape with different colors depending on factors such as age and size. There are many types of stars in the universe, some being billions of years old, and studying them will teach us a lot about the past. A quasar is an incredibly luminous object found at the center of a galaxy. It is formed as lots of excess gas and dust fall into a super massive blackhole and form an accretion disk. These gases feel extreme frictional and gravitational forces and emit large amounts of radiation.

When looking at the achievements of others in this task, we can see there has been much progress, but also that there is still room to grow. Yulun Winston Yu (2021) compared the logistic regression and decision tree models in completing this task. They found that the decision tree model was the best, achieving a high accuracy score and f1 score of 99%. Makhijaa et al. (2019) used neural network models and got high accuracies of around 99%. In both these studies, they chose to only use one or two models. Thus, in our research, we hope to go further and conduct a systematic study by testing many models such as logistic regression, ridge classifier, decision tree, random forest, and neural network against each other. Additionally, the datasets in these referenced sets had only around 10000 to 30000 objects. In our study, we use 500,000 which helps with testing and

training to the best accuracy. This will especially help in the quasar identification as they encompass a small portion of all the datasets that were tested in these other studies.

Our objectives are to compare many different machine learning models and find the one that can achieve the highest accuracy in this task. On Earth, it is hard to find the patterns between celestial objects since they are large distances away and emit complex patterns of light. Thus, we want to find a way to identify these celestial objects so that astronomers can study them.

Dataset

We used the data from the Sloan Digital Sky Survey (SDSS) Data Release 16. The SDSS is an astronomical survey that aims to map the universe. This public dataset from the DR16 release has 500,000 observations. It had 18 features, but only 9 are relevant, so the rest were removed. Table 1 is an example of five objects in our dataset with only key features. Overall, we have 9 key features, *ra, dec, u, g, r, i, z, redshift*, and *class*. *Ra* and *Dec* are the right ascension angle and declination angle. These two values basically tell us where the object is in the sky. At the end, we have the *class* labels. This label describes the class of celestial object pre-determined by the Sloan Digital Sky survey. Next, the *u,g,r,i,z* and *redshift* values are what we use to identify the object. The *u,g,r,i,z* values are the photometric data of each object. The photometric system is one used to categorize the light emitted from stellar objects. It gives us information on the magnitude of each type of electromagnetic frequency. We used the *u,g,r,i,z* system for our data. The *u* represents the ultraviolet emissions (300-400 nm), the *g* represents the green emissions (400-550 nm), the *r* represents the red emissions (550-700 nm), the *i* represents the near infrared emissions (700-850 nm), and the *z* represents the infrared portion of the spectrum (850-1000 nm). Another piece of information we used was the *redshift* values of each object. Redshift is a phenomenon which causes the spectral output of an object to shift because of its movement relative to the viewer. We can assign a number to the redshift of each object which tells us how far its spectral lines are shifted. Both the *u,g,r,i,z* values and *redshift* will be incredibly helpful in identifying what the object is.

Table 1. Sloan Digital Sky Survey (SDSS) dataset. Example of five objects in our dataset with 9 key features, ra, dec, u, g, r, i, z, redshift, and class.

index	ra	dec	u	g	r	i	z	redshift	class
148842	256.353	34.0657	18.0125	17.5333	17.4326	17.2701	17.3514	0.55927	**QSO**
66346	249.771	44.193	19.0041	18.1252	17.9322	17.8378	17.8381	-0.0006	**STAR**
456003	118.133	18.1451	18.9787	17.4355	16.887	16.7048	16.6382	0.00015	**STAR**
352658	336.576	0.73947	19.3537	17.4466	16.547	16.127	15.8022	0.05799	**GAL-AXY**
121787	250.936	13.0039	19.5233	18.3987	17.7431	17.3419	17.1417	0.14221	**GAL-AXY**

Exploratory Data Analysis

First, we explored the class label distribution. The goal is to understand whether or not our dataset is balanced or imbalanced. Figure 1 illustrates the count of observations for each of the three labels. In total, we have 500,000 observations. 50% of those are of the galaxy label, 40% are of the star type, and the last 10% are for quasars. It is important to note that our dataset is mildly imbalanced. There is a large number of samples for

each type, but there is much more for stars and galaxies than for quasars. We kept this in mind for preprocessing the data so that we could correct any errors that arose from this issue.

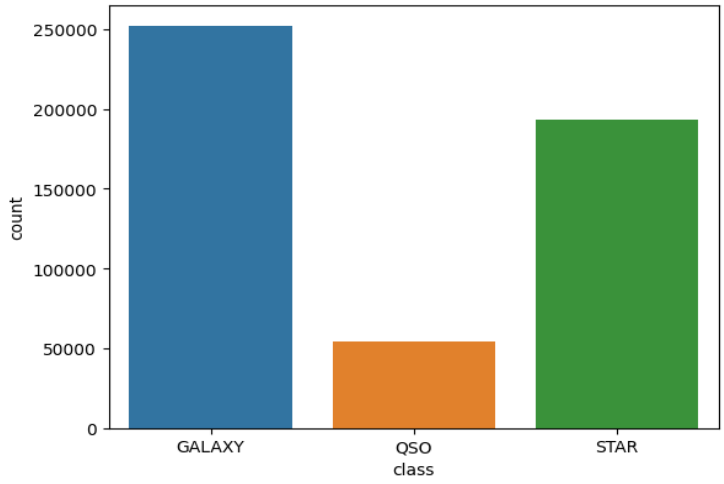

Figure 1. Imbalanced distribution of class labels (i.e. galaxy, star, quasar)

Next, we explored the correlations between features in our dataset. The goal is to understand the relation and correlation between features. The features *g* and *u* have a correlation of around 0.9 and *g* and *r* have a correlation of almost 1. Features *r* and *u* also have high correlation of around 0.7. The *ra* and *dec* angle values appear not to have any correlation with anything which makes sense since where the object is in the sky should not affect the light it emits. However, they were still kept as inputs to the model as there may have been some slight correlation that the model would be able to observe. *Redshift* does seem to have some correlation with the *u*, *g*, and *r* features with it being highest with *r*. *i* and *z* also seem to have a faint relationship.

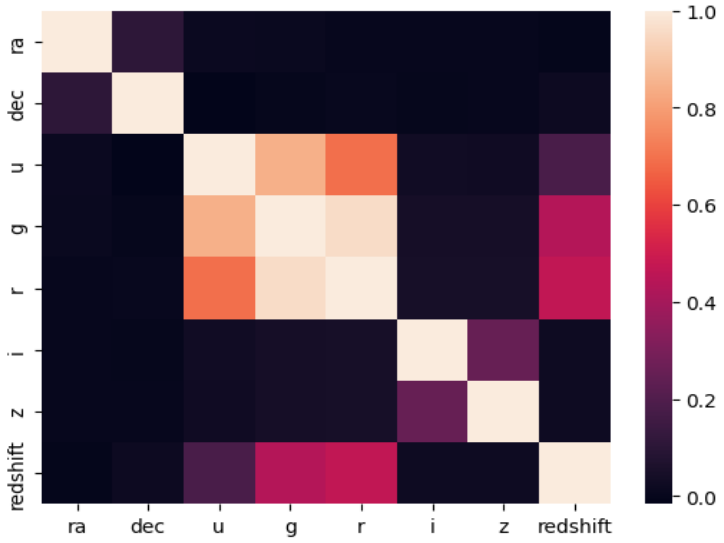

Figure 2. Correlation map between features

Figure 3 illustrates some of the correlations from Figure 2. All of the photometric relationships appear somewhat linear. However, the g-u graph has a very high slope when compared to the r-u and r-g lines. In each relationship, there seems to be somewhat distinct lines for each class. This might be useful in classification of class labels. In the redshift relationships, there seems to be a cloud of points around a horizontal line.

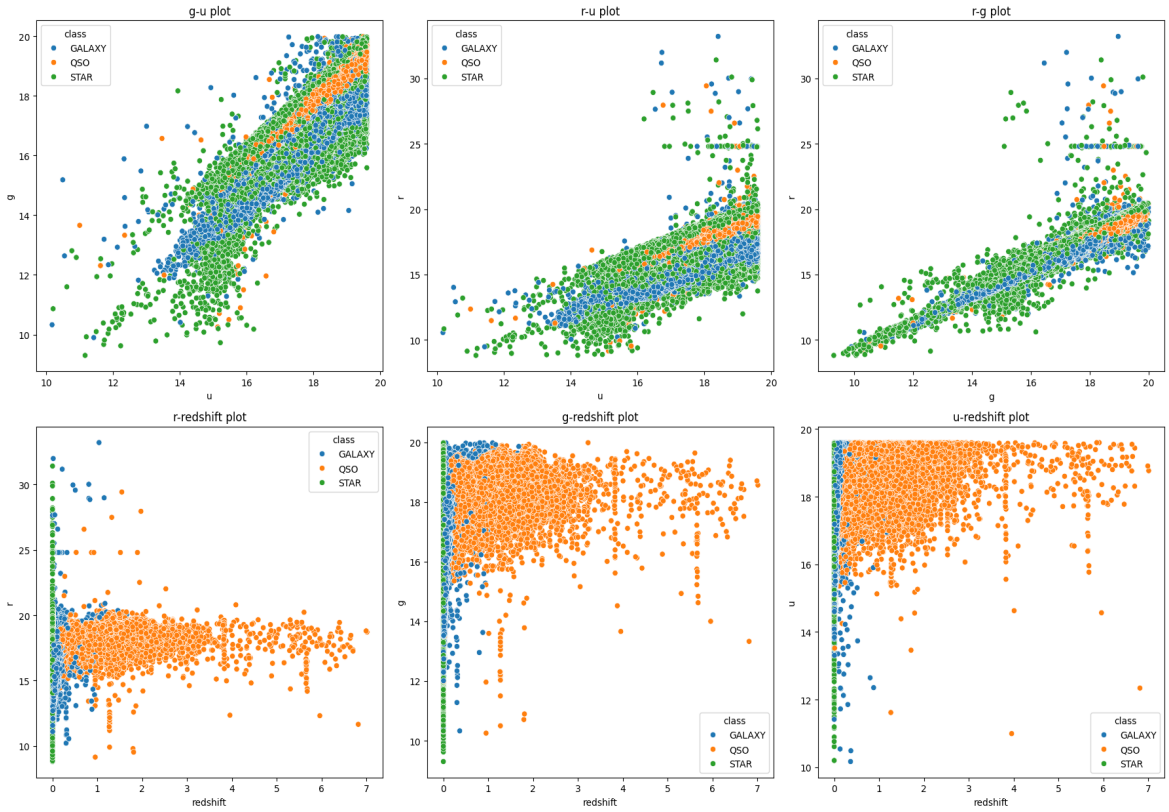

Figure 3. Relationship between some of the features

Last, we explored the features' distribution and ranges. The goal is to understand the ranges, lower and upper limits as well as the overlap between the features (Figure 4). Figure 4 shows that we can visually identify distinctions between the classes. For the *redshift* distributions, it is incredibly apparent that the quasar usually has a much higher value than both the galaxy and star. In addition, the star seems to have the lowest redshift values of all the other classes. This will be very helpful for classification. Another set of the important distributions are the ***u*** and ***z*** values. For the ***u*** data, there is not much distinction between galaxies and quasars. However, it is apparent that the star class usually has lower values. In the ***z*** dataset, there are differences between each of the classes. Galaxy usually has the lowest, quasar has the highest, and the star is between those values.

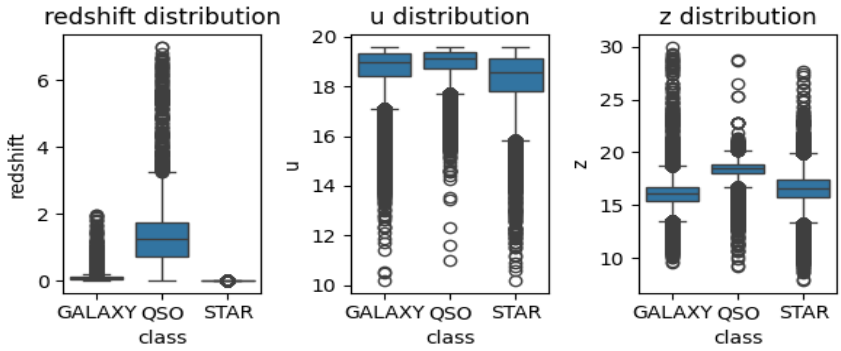

Figure 4. Red shift, u and z box plots

Methodology and Models

We split the data into train and test sets using the test size of 30%. By using a stratified split, we made sure that the test and train data had the same class distribution to the original data set. This helps minimize biases in the model. We also used a standard scaler to scale the features. Following, we used our train and test data to build five different baseline models, decision tree, random forest, logistic, ridge, and neural network.

A decision tree is a machine learning model that begins with a root node. This node splits the data based on the most important feature in the set following a criterion. Depending on how it is split, data will go down a branch to another node which will split the set based on another feature. Finally, the data will reach the end of the tree at a leaf node which will determine its class. The random forest model takes a random sample of data from the train set using bootstrap sampling and uses it to build a decision tree model. It does this many times in order to create a "forest" of decision trees. On the test data, it runs each observation through each tree. It then averages the result of each decision and uses that as a final result. The model was created to prevent overfitting seen in the decision tree.

Another model used was logistic classification. Logistic classification uses the logistic function to take in input features and map them out to one output value. The training algorithm finds the weight of each feature so that they can be summed to one input value. It also adds a constant term b. This input value is put into the logistic function which outputs the probability of which class it belongs too.

Ridge classifier, our fourth model, builds on the model of linear regression. This model uses a linear relationship between the features and output value. In training, it tunes the slopes of the function and the added constant so that it will output the right number that indicates the right class. Ridge classifier builds on this by trying to minimize overfitting. It does this by adding a penalty constant to the linear regression coefficients that helps generalize the predictive function.

Our final model is a neural network. This structure is a model made out of layers of neurons. Neurons are nodes that take input and will activate if that input value satisfies a certain function. These neurons are organized in layers with the input layer, hidden layers, and output layer. The input layer takes in the features into input nodes and sends them to the next neurons that they are connected to. Going through the connection, the input values are multiplied by weights and added by bias constants. This value is put into the next node's activation function which, if it fires, will send its output to the next nodes connected. Finally, the value will reach the output layer which will determine its class based on the arrived value. In training, the network determines the weights needed to achieve the desired classification output. However, we have to set the architecture which will be used. For our model, the best network was one with 3 hidden layers that each had 64 neurons.

After we had tested each baseline model, we found that the random forest had the best baseline detection. Thus, we hyper parameter tuned the baseline random forest model. Hyperparameter tuning is when you

change the initial values that describe the forming of the model so that you may get the greatest detection ability. For the random forest, we tuned the three most important parameters: the maximum features, minimum sample split, and number of estimators. The maximum features determine how many features a tree will be given and trained with. The minimum sample split determines the least required number of observations in a node to continue branching. The number estimators simply determine the number of decision trees in the forest.

During our hyperparameter tuning, we first used the random search method. In this method, we created a grid with many potential values for each parameter. Our grid had a minimum sample split of 2 to 6 with spaces of 2, a number of estimators of 200 to 2000 with spaces of 200, and a maximum feature of 1, 2, and 4. The algorithm would then randomly choose a number of combinations from this grid and validate each of them to find the best. To validate the parameters, we used k-fold cross validation. In this validation method, the training and test sets are split into K subsets of equal size. Then one subset is chosen as the test set while all the rest are chosen for training. This is repeated for every subset in the set and the results are averaged to get accurate validation metrics. Following the random search, we were able to get a general idea of the range of values for each parameter. We then switched to grid search with K-fold CV. Grid search is similar to random search except that it goes through every combination of hyperparameters using cross validation. Our best values for these three parameters resulted are number of estimators of 1000, minimum sample split of 10, and maximum features of 6.

Results and Discussion

We used confusion matrix to evaluate our models. A confusion matrix is a map that compares the predicted label by the model to the true label. For our data set this means that on the x axis it shows if it was predicted as a quasar, galaxy, or star, and on the y axis it shows its actual label.

Logistic Classification

As can be seen from Figure 5, baseline logistic regression yields very high accuracy for predicting stars. However, it was less accurate for predicting quasars and galaxies. In the test data, the model confused the star and galaxy the most times. There doesn't seem to be any overfitting as the train confusion matrix looks very similar to the test matrix.

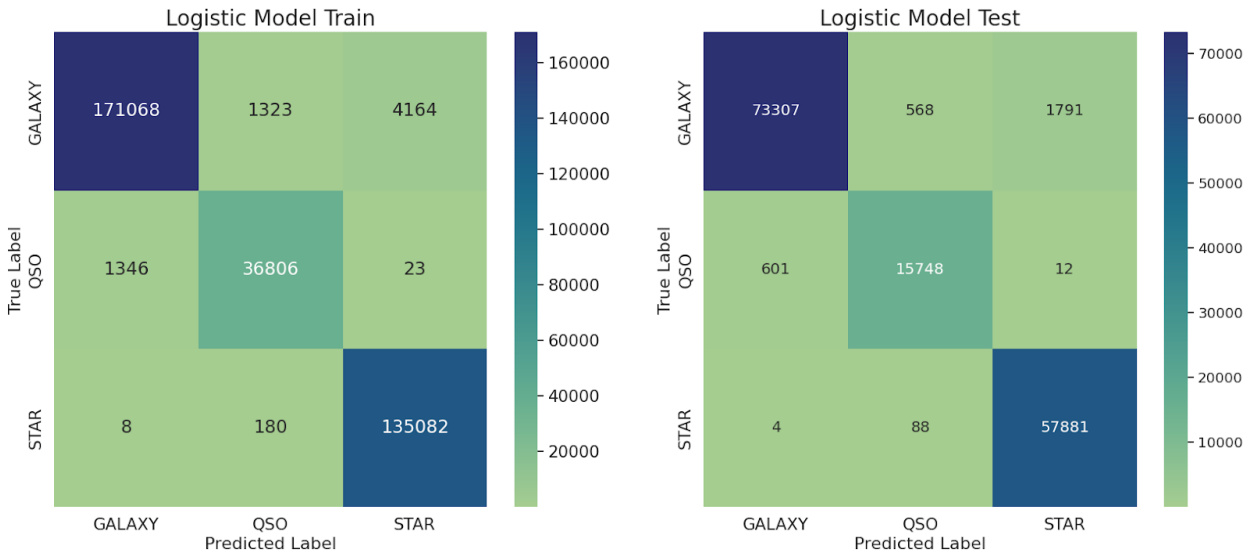

Figure 5. Train and test confusion matrix from the baseline logistic regression

Decision Tree

Figure 6 illustrates the confusion matrixes for the baseline decision tree model train and test datasets. In the train matrix, we can see that the model perfectly conforms to the dataset with every class getting correctly predicted. However, in the test set, we can see that it is off many times and that the accuracy is much lower than in the train set. This is evidence of overfitting which means that it conforms too tightly to the train set. The accuracy is still good, it predicted more correctly than the logistic model, but it could be improved by using a random forest instead. The biggest confusion in this model was between quasars and galaxies.

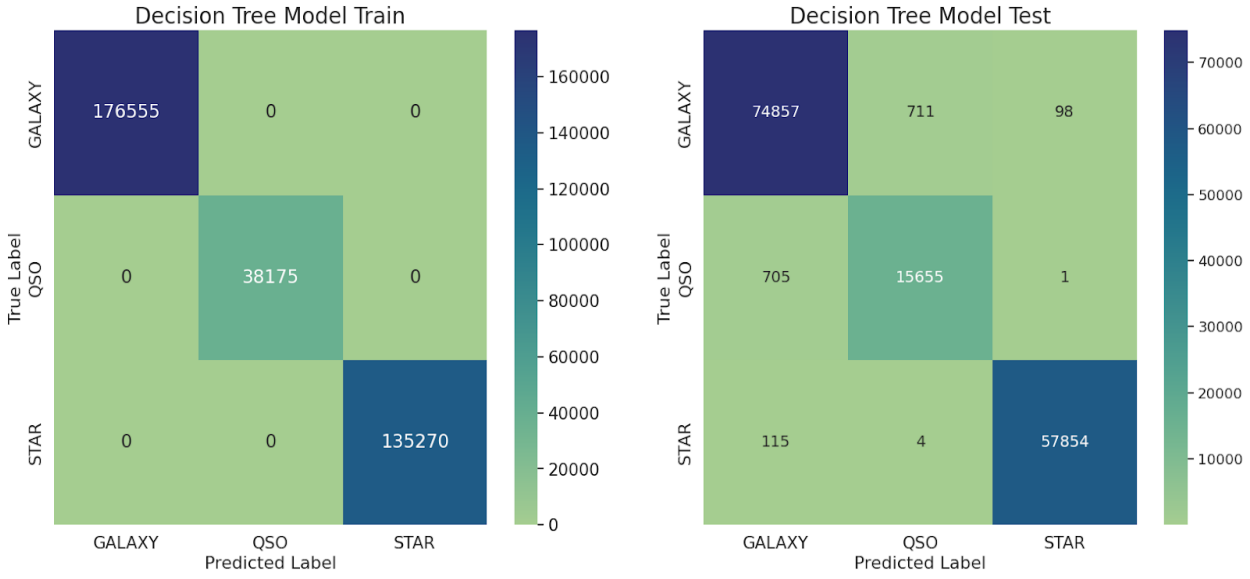

Figure 6. Train and test confusion matrix from the baseline decision tree

Random Forest

Figure 7 illustrates the confusion matrixes for the baseline random forest model train and test datasets. In the train confusion matrix, we can see there might again be some overfitting since the accuracy is almost 100 percent. However, we can also see that the test matrix shows much better results than the decision tree. The decision tree overfitting resulted in confusion of the galaxy class with the star and quasar class, but in the random forest, this has been largely minimized. The model also confuses less of the stars for galaxies when compared to the decision tree. It has high accuracy and f1-scores.

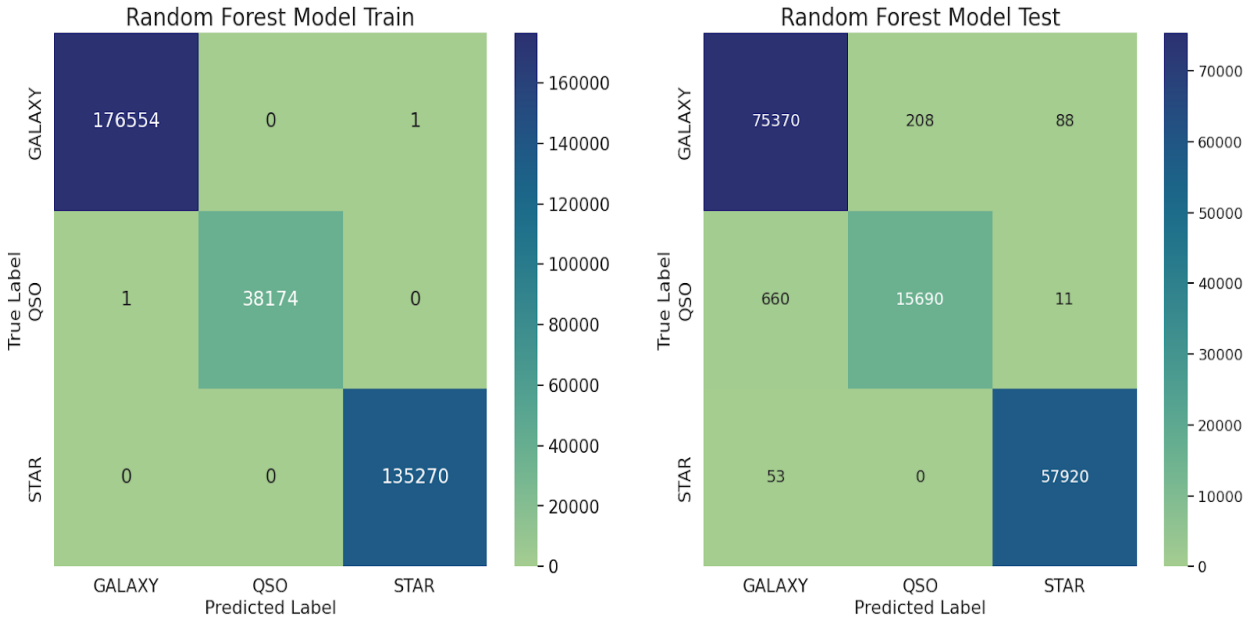

Figure 7. Train and test confusion matrix from the baseline random forest

Ridge Classification

The ridge classifier does the worst when compared to our other models (Figure 8). It confuses the star and galaxy class many times both in the train and test matrixes. Its accuracy and f1 score are clearly the lowest out of our models. Thus, it will not be useful.

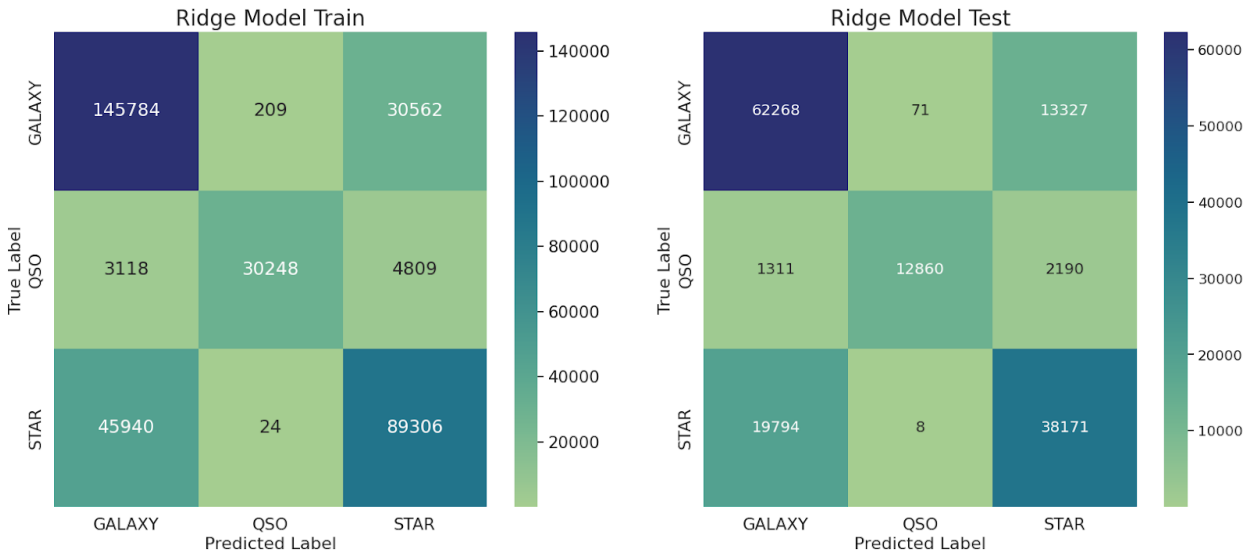

Figure 8. Train and test confusion matrix from the baseline ridge classifier

Neural Network

The Neural Network produces very good results similar to the random forest as depicted in Figure 9. The training model works well and predicts most correctly. The test model has good accuracy and f1 scores. Despite this the neural network confuses the star and galaxies much more than the random forest.

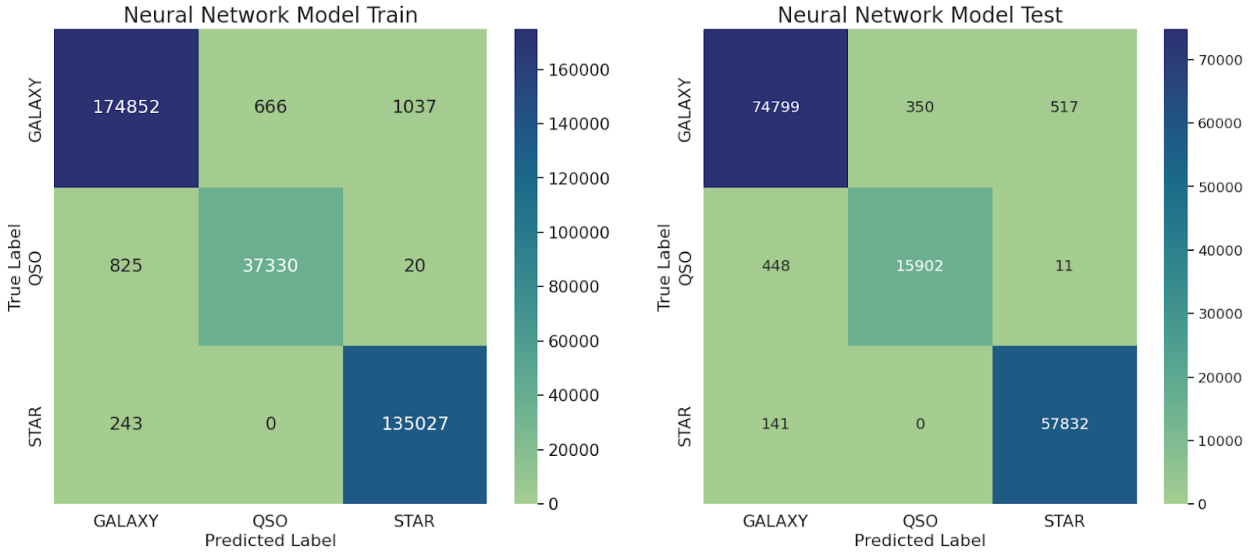

Figure 9. Train and test confusion matrix from baseline neural network

Hyper Parameter Tuned Random Forest

Out of all of baseline models, the baseline random forest was the best by far. After hyper parameterization, we achieved even better performance as shown in Figure 10. We evaluated our model using four main metrics,

recall, precision, f1-score, and accuracy. Recall is how many that should have been predicted as a class were predicted as that class, precision is how many predicted in a class are correct, accuracy is how many were predicted correct, and f1-score combines both the recall and precision metrics. F1-score, precision, and recall are binary metrics and thus can only be used in a class, not the whole set. The total accuracy of the model is 99.3% which is incredibly good. The individual metrics were also very good for each class. With many of the test precision, recall and f1-scores for each class near 1.00. The weakest point of this model is in identifying quasars. However, it still accurately finds many of them as it has the highest metrics compared to the other models with a f1-score of 0.97, recall of 0.96, and precision of 0.99. It is important to note that we rounded the models to the second decimal place so numbers of 1.00 are most likely near perfect, not actually always correct.

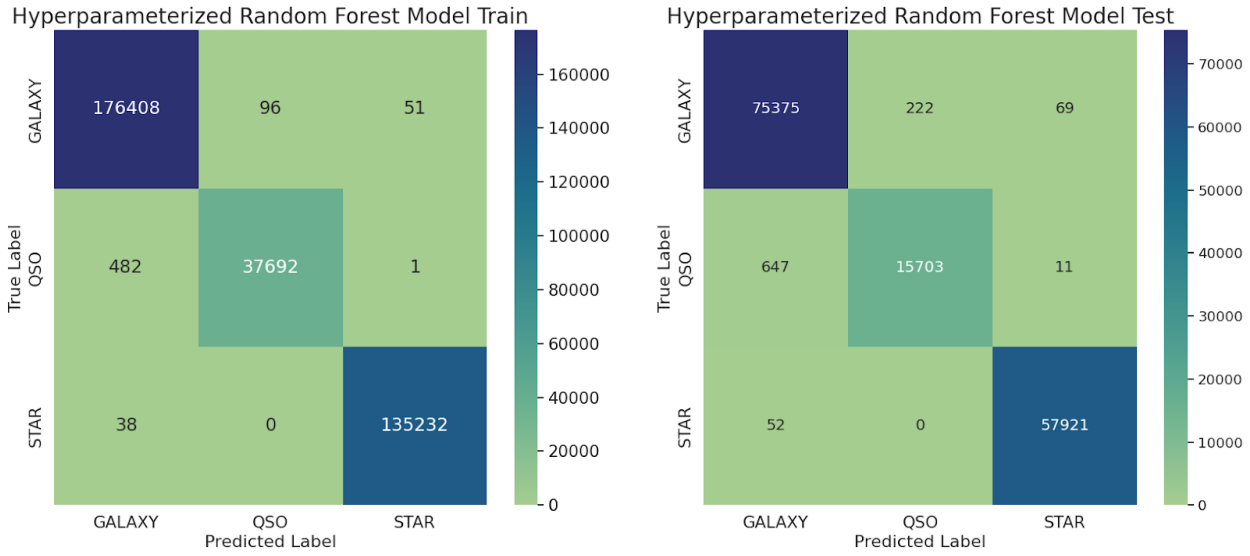

Figure 10. Train and test confusion matrix from hyperparameter tuned random forest

Table 2. Train classification report for hyperparameter tuned random forest model

	Precision	**Recall**	**F1-Score**	**Support**
Galaxy	1.00	1.00	1.00	176444
Quasar	1.00	0.99	0.99	38175
Star	1.00	1.00	1.00	135270
accuracy			1.00	350000
macro avg	1.00	1.00	1.00	350000
weighted avg	1.00	1.00	0.98	350000

Table 3. Test classification report for hyperparameter tuned random forest model

	Precision	Recall	F1-Score	Support
Galaxy	0.99	1.00	0.99	75666
Quasar	0.99	0.96	0.97	16361
Star	1.00	1.00	1.00	57973
accuracy			0.99	150000
macro avg	0.99	0.98	0.99	150000
weighted avg	0.99	0.99	0.99	150000

Conclusions

In this study, a wide range of machine learning models were used to classify astronomical objects. Comparing the random forest, logistic model, decision tree, neural network, and ridge models, we were able to find the best method to classify objects as stars, quasars, or galaxies. We identified the random forest as the best model and used the hyper parameterization methods of random and grid search to maximize its detection ability, achieving a high accuracy of 99.3%. Its metrics were also high with the precision of each class being 0.99 or 1.0, the recall of each class at 1.00 except for quasar at 0.96, and f1 scores of 0.97 for quasars, 0.99 for galaxies, and 1.00 for stars. This is higher than the other results that we saw in our literature review. This is probably due to changes in our dataset and the model comparisons. This model is incredibly good and could be applied to the astronomical field for identification purposes. However, there are still some things we could improve. First of all, there are more objects than the ones we classified such as comets and even blackholes. Our model does not recognize these, so in further studies it would be good to incorporate such data. Secondly, altering the distribution of the dataset so that there are more quasars would be beneficial to accuracy. It would help in improving that detection. Despite this, overall our model could aid astronomers in studying the universe and open doors to the knowledge behind galaxies, quasars, and stars. It is incredibly effective at identifying the objects we have taught it to.

Acknowledgments

I would like to thank my mentor Abdulla Kerimov , a data scientist at BP Energy, for helping me through the research process. He answered every question I had and taught me how to write and formulate a research plan. I would also like to thank the Sloan Digital Sky Survey which is run by the Astrophysical Research Consortium. Finally, I would like to thank Steve Szabados, Director of Machine Learning at iRhythm Technologies, for reviewing my article.

References

Wu, Y. (2021). MACHINE LEARNING CLASSIFICATION OF STARS, GALAXIES, AND QUASARS. MATTER: International Journal of Science and Technology, 6, 102–122. doi:10.20319/mijst.2021.63.102122

Makhija, S., Saha, S., Basak, S., & Das, M. (2019). Separating stars from quasars: Machine learning investigation using photometric data. Astronomy and Computing, 29, 100313. https://doi.org/10.1016/j.ascom.2019.100313
Hoerl, A. E., & Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. Technometrics, 12(1), 55–67. https://doi.org/10.2307/1267351
Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85–117.
Rokach, Lior & Maimon, Oded. (2005). Decision Trees. 10.1007/0-387-25465-X_9.
Sperandei S. (2014). Understanding logistic regression analysis. Biochemia medica, 24(1), 12–18. https://doi.org/10.11613/BM.2014.003
Breiman, L. (2001). Random Forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/a:1010933404324