Stellar Objects Classification Using Supervised Machine Learning Techniques

Deen Omat
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
dee20180947@std.psut.edu.jo

Jood Otey
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
joo20180505@std.psut.edu.jo

Amjed Al-Mousa
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
a.almousa@psut.edu.jo

***Abstract*—Machine Learning is used in many fields of study. This paper used machine learning to classify instances from the Sloan Digital Sky Survey Data Release 17 (SDSS DR17) as a galaxy, quasar, or star. Supervised learning was used to make the classification. Multiple machine learning models were built, Decision Trees, K-Nearest Neighbors, Multinomial Logistic Classification, Multilayer Perceptron, Naïve Bayes Classifier, Support Vector Classification, Random Forest, and Soft Voting Classifier. Random Forest performed the best with 98% accuracy and correctly classified all instances labeled as stars in the dataset. The worst-performing algorithm was Naïve Bayes, with 91% accuracy.**

Keywords—Machine Learning; Stellar Classification; Sloan Digital Sky Survey; Supervised Learning; Classification.

I. Introduction

Stars are made up of hydrogen and helium, the building blocks of galaxies [1]. Galaxies are also made of gas and dust; there are so many galaxies in the universe that scientists cannot count [2]. Quasars are found in some large galaxies with supermassive black holes at their centers and are considered active galaxies themselves. Five to ten percent of large galaxies are quasars [3].

The SDSS is one of the longest-running scientific research programs funded by the Sloan Foundation in astronomy. It has been running for 15 years and has succeeded in creating the most detailed three-dimensional maps ever made of the universe and in mapping one-third of the night sky. The data collected by the SDSS is available to the public and falls under open-source principles [4].

Stellar classification is the classification of stellar objects depending on their spectral characteristics. Due to the large amounts of data collected about the universe, scientists have begun applying machine learning algorithms to sift through the data [4]. This paper used eight classification algorithms to classify the instances in the SDSS dataset.

This paper was divided into five sections. The first section discussed similar work previously published. The second section elaborated on the dataset and the data preprocessing to prepare the data for the models used. The third section discussed the algorithms used, which hyperparameters for each algorithm were tuned, and the values chosen for those hyperparameters. The fourth section revealed the results obtained when the models were tested. Lastly, the fifth section concludes and summarizes what was done in this paper.

II. Related Work

Machine learning has been used successfully to predict medical, financial, and educational outcomes [5, 6, 7]. In addition, machine learning is widely used in astronomy; this section will cover three works done using machine learning. In a paper published, the authors used the Cumulative Kepler Object of Information dataset and machine learning to identify the exoplanets in the dataset. Four algorithms were used, Support Vector Machines, Random Forest, Ada Boost, and Neural Networks. The Ada Boost Classifier had the highest f1-score of 0.98. [8]

In another paper, the author used the fourth edition of the SDSS. The algorithms used were Decision Trees, Logistic Regression, and Naive Bayes Classifier to classify the instance into either galaxy, star, or quasar. The highest mean f1-score was achieved by their Decision Tree model, with a mean score of 97.8%. In addition, the authors did some feature engineering by adding the square of each feature, the product of all pairs of parameters, and the logarithm of each one. Their models could classify all the stars correctly, so the improved dataset was used to classify whether an instance was a galaxy or a quasar. The logistic regression classifier achieved the highest f-1 score of 98.2% [9].

Chuntama et al. used the Canada-France-Hawaii Telescope (CFHT) data archive [10]. Instances were classified into one of the following categories: star, globular cluster, rounded galaxy, elongated galaxy, or fuzzy object. To do the classification, they used Random Forest, Multilayer Perceptron, Weightless neural network (WiSARD), Deep Learning (Weka deep learning), Logistic Regression, Support Vector Machine, and Multiclass Classifier. Their Random Forest classifier achieved the best performance with an accuracy of 81.2%.

III. Experimental Setup

The dataset mentioned earlier was initially images taken by SDSS's cameras. The data was then processed and uploaded to Kaggle [11]. This paper aims to classify the instances into three categories: galaxy, quasar, or star.

A. Attribute Information

The dataset contained 100000 instances, distributed as follows: 59445 were galaxies, 21594 were stars, and 18961 were quasars. The pie chart in Figure 1 shows the class distribution.

2022 International Arab Conference on Information Technology (ACIT) | 979-8-3503-2024-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ACIT57182.2022.9994215

None of the features had missing or null values. Some of the features contained information about identifying the instance in the dataset. However, they were not useful for determining to which class the instance belonged and were therefore excluded. After excluding the irrelevant features, the dataset was reduced to 6 features and one label from the original 17 features and one label.

The original images were taken using a photometric system that consisted of five filters (u, g, r, i, and z). A photometric system is a set of defined ranges of wavelengths that can pass through the filters, with sensitivity to incident radiation. The sensitivity depends on the optical system, detectors, and filters used. Table 1 shows the features kept and a brief description of them.

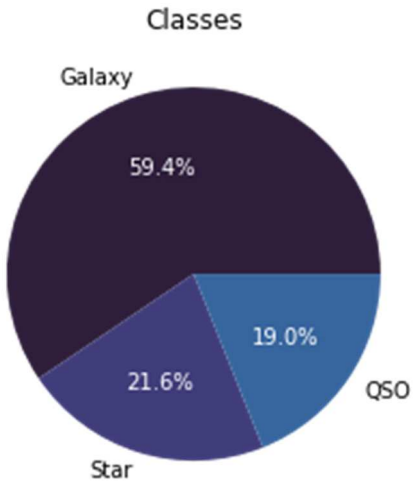

Fig. 1. Stellar Objects Class Distribution

TABLE 1. Attributes Relevant to the Task

Feature	Type	Description
u	float	Ultraviolet filter in the photometric systems
g	float	Green filter in the photometric system
r	float	Red filter in the photometric system
i	float	Near-infrared filter in the photometric system
z	float	Infrared filter in the photometric system
redshift	float	Redshift value based on the increase in wavelength
class	string	Instance's class

Before proceeding to data preprocessing, a histogram of each attribute was plotted to see how each feature was distributed. It was obvious that feature scaling would be necessary for more accurate predictions. The distribution of g, redshift, u, and z are shown in Figures 2, 5, 6, and 7, respectively might make it hard for the models to make predictions. Figures 3 and 4 follow a similar distribution to a normal distribution. Furthermore, every feature has a different range.

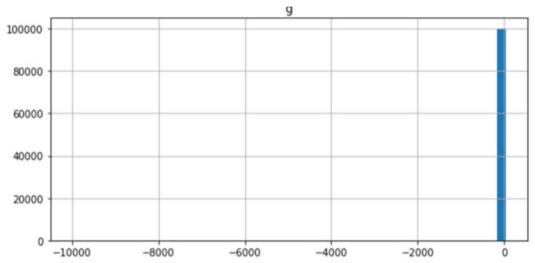

Fig. 2. Green Filter (g) Feature Pre-Scaling

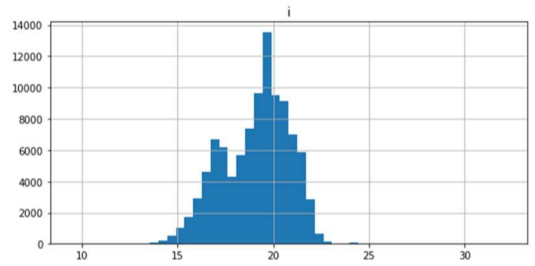

Fig. 3. Near Infrared Filter (i) Feature Pre-Scaling

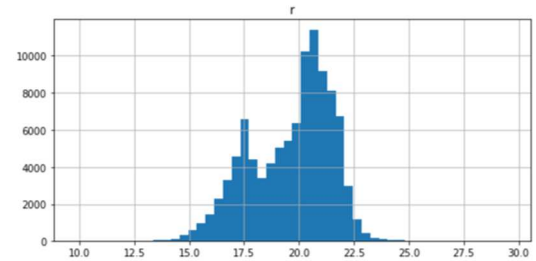

Fig. 4. Red Filter (r) Feature Pre-Scaling

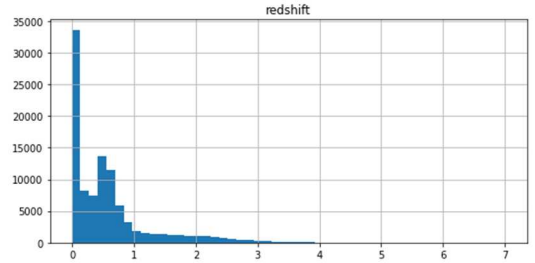

Fig. 5. Redshift Value Feature Pre-Scaling

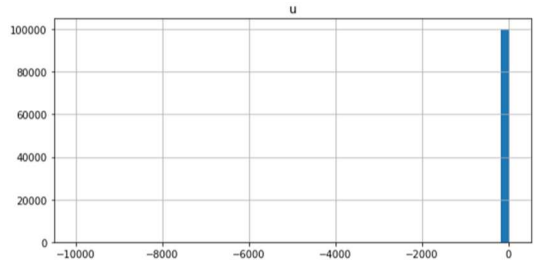

Fig. 6. Ultraviolet Filter (u) Feature Pre-Scaling

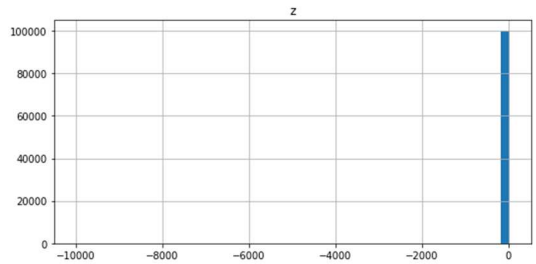

Fig. 7. Infrared Filter (z) Feature Pre-Scaling

B. Data Preprocessing

The first step of data preprocessing was feature scaling. Feature scaling is crucial to avoid bias towards any features since not all features have the same range of values. In addition, the Support Vector Classification, Multinomial Logistic Regression, and Gaussian Naïve Bayes models perform better when the data is normally distributed. Yeo-Johnson power transform method was applied to have the attributes follow a normal distribution [12].

Yeo-Johnson transformation is defined as:

$$y_i^{(\lambda)} = \begin{cases} \dfrac{(y_i+1)^{\lambda}-1}{\lambda} & if\ \lambda \neq 0, y \geq 0 \\ \log(y_i+1) & if\ \lambda = 0, y \geq 0 \\ -\dfrac{((-y_i+1)^{2-\lambda}-1)}{2-\lambda} & if\ \lambda \neq 2, y < 0 \\ -\log(-y_i+1) & if\ \lambda = 2, y < 0 \end{cases} \quad (1)$$

Figures 8-11 show the distribution of g, redshift, u, and z, respectively, after scaling. The features now follow a similar distribution to a normal distribution. There was no significant change in the distribution of the rest of the features.

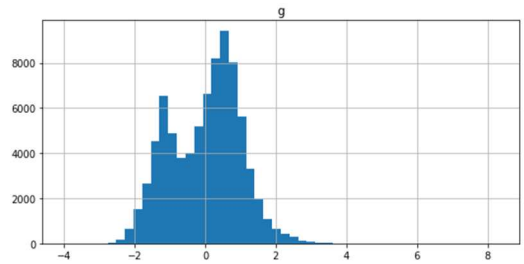

Fig. 8. Green Filter (g) Feature Post-Scaling

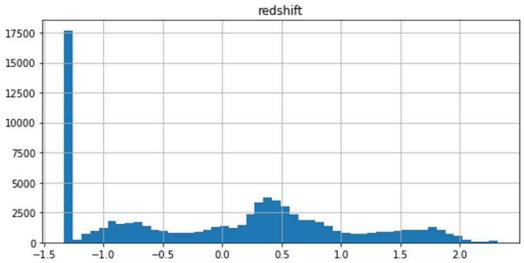

Fig. 9. Redshift Value Feature Post-Scaling

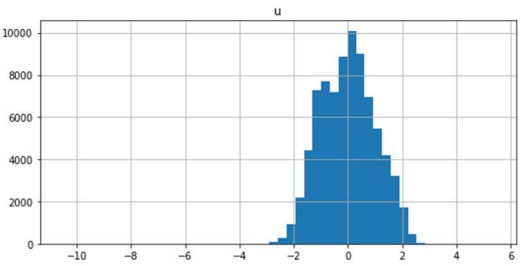

Fig. 10. Ultraviolet Filter (u) Feature Post-Scaling

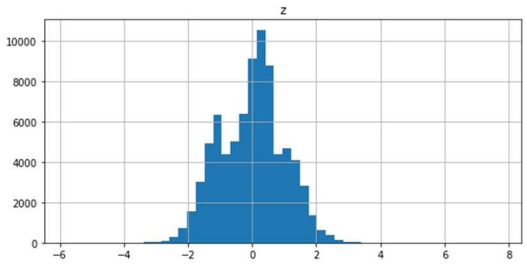

Fig. 11. Infrared Filter (z) Feature Post-Scaling

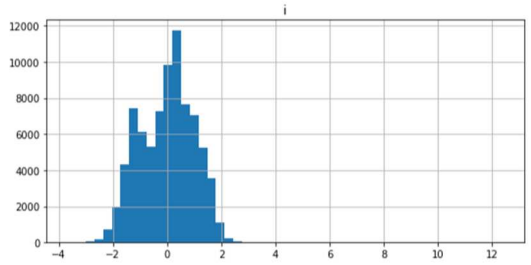

Fig. 12. Near Infrared Filter (i) Feature Post-Scaling

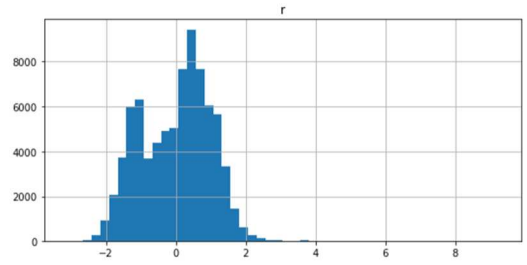

Fig. 13. Red Filter (r) Feature Post-Scaling

After scaling the features kept, to explore any correlations present between the features, a heatmap was plotted. There was a strong correlation of 0.83 between features g and u, a strong correlation of 0.91 between features g and r, a strong correlation of 0.95 between features r and i, and a strong correlation of 0.96 between features i and z.

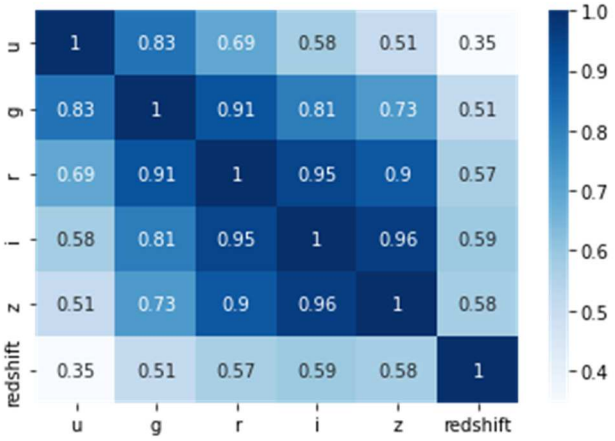

Fig. 14. Correlation Between Features

Before training the algorithms, the dataset was split into a train set and a test set which was necessary to see how well the model generalized to unseen data. The data was split randomly. However, a random seed was used to ensure the split's repeatability. This number ensures that the same rows were chosen for the training and testing sets in every run. The method provided by Sklearn (train_test_split) was used to do

the splitting. 80% of the data was used for training, and 20% was used for testing.

IV. Algorithms Used

Eight classification algorithms were used. Fivefold cross-validation was done on all algorithms. The following subsections discuss each algorithm used and how it works.

A. Decision Trees

The first algorithm trained was Decision Trees. Decision Trees are a widely used supervised learning algorithm to solve both classification and regression tasks. In this paper, the algorithm was used for multiclass classification.

Decision Trees classify instances based on whether or not they meet the node's condition; the process of classifying based on whether a condition was met or not gives the algorithm a tree-like structure. The node that does not branch is called a leaf node, and it has the final class label of the instance.

Decision Trees are prone to overfitting. Therefore, hyperparameter tuning was done, which resulted in a lower incidence of overfitting. Hyperparameter (max_depth) was not set to any value, which meant the algorithm would keep branching and cause overfitting. A randomized search was used to find the best value for the hyperparameters given a specific range. The hyperparameters regularized were (min_sample_leaf), (max_leaf_nodes), and (max_depth) and their values were set to 11, 11, and 6 respectively.

TABLE 2. Decision Trees Hyperparameters Tuned

Hyperparameter	Value
Min Sample Leaf	11
Max Leaf Nodes	11
Max Depth	6

B. K-Nearest Neighbors

The following algorithm used was K-Nearest Neighbors. Another popular supervised learning algorithm can be used for both regression and classification.

This algorithm classifies an instance based on the majority class of the K-nearest neighbors. An odd value for K is typically used to avoid a tie between classes. Hyperparameter Tuning was also done to increase the accuracy. The hyperparameters that were set were (n_neighbors), (weights), (p), and (leaf_size). After performing a randomized search the (n_neighbors) was set to 19, (weights) was set to uniform, (p) was set to 2, and (leaf_size) was set to 49.

TABLE 3. K-Nearest Neighbors Hyperparameters Tuned

Hyperparameter	Value
N Neighbours	19
Weights	Uniform
Power Parameter	2
Leaf Size	49

C. Multinomial Logistic Regression

The third algorithm used was a regression algorithm that can also be used for classification. Multinomial Logistic Regression calculates the probability that the instance belongs to a class.

The probability is calculated using a sigmoid function, which is defined as:

$$\sigma(t) = \frac{1}{1 + \exp(-t)} \tag{2}$$

The output of a sigmoid function is a value between 0 and 1. The model makes its prediction based on whether the probability is greater than 50%. If the calculated probability of belonging to a class is greater than 50%, then the instance is classified as that class. The only hyperparameter set was (C), which was set to 1000.

TABLE 4. Multinomial Logistic Regression Tuned

Hyperparameter	Value
The inverse of Regularization Strength, C	1000

D. Multilayer Perceptron

The next supervised learning model trained was Multilayer Perceptron, an artificial neural network. The structure of the model is as follows: one input layer, one or more hidden layers, and an output layer. Every neuron in all layers is connected with every neuron in the next layer, except for the output layer. All neurons except for input neurons are passed through an activation function. The output of an activation function decides whether the neuron should fire or not. The activation function used in this model was the ReLu function, and it is defined as:

$$f(x) = \max(0, x) \tag{3}$$

Multilayer Perceptrons have many hyperparameters that can be optimized. However, since tuning many parameters is computationally intensive, a randomized search on two parameters was done. The (hidden_layer_size) was set to (100, 100, 1000), and (alpha) was set to 0.01.

TABLE 5. Multilayer Perceptron Hyperparameters Tuned

Hyperparameter	Value
Hidden Layer Sizes	(100, 100, 100)
Alpha	0.01

E. Naïve Bayes Classifier

The fifth algorithm used was Gaussian Naïve Bayes, a probabilistic classifier; it uses Bayes theorem of probability to predict the class of instances. The algorithm assumes independence among features. The Gaussian Naïve Bayes algorithm also assumes that the features follow a normal distribution. After performing a randomized search on the (var_smoothing) hyperparameter, it was set to 8.11308307896872e-05.

TABLE 6. Naïve Bayes Classifier Hyperparameter Tuned

Hyperparameter	Value
Smoothing	8.111308307896872e-05

F. Random Forest

Random Forest is a very popular supervised learning algorithm. It works on classification and regression tasks. It is an ensemble algorithm made up of decision trees. The algorithm decides to classify an instance as a certain class based on what class was selected the most by the decision trees. Random Forest algorithm has a lot of hyperparameters, but only (min_samples_split), (min_samples_leaf), (max_features), and (max_depth) were tuned. (min_samples_split) was set to 14, (min_samples_leaf) was set to 6, (max_features) was set to log2, and (max_depth) was set to 12.

TABLE 7. Random Forest Hyperparameters Tuned

Hyperparameter	Value
Max Depth	12
Min Samples Split	14
Min Samples Leaf	6
Max Features	log2

G. Support Vector Classification

Although Support vector classification is used when the output class is binary, it was used in this paper. To be able to use this classifier, multiple binary classifiers have to be used. SciKit's Support Vector Classifier automatically creates binary classifiers using a one-vs-one scheme.

Support Vector Machine separates classes using an N-dimensional hyperplane, where N is the number of features. The data points closest to the hyperplane are called support vectors. The algorithm maximizes the margin between the hyperplane and the support vectors. To increase accuracy, the randomized search was performed on selected hyperparameters. Hyperparameter (C) was set to 100, and (kernel) was set to RBF.

TABLE 8. Support Vector Classification Hyperparameters Tuned

Hyperparameter	Value
The inverse of Regularization Strength, C	100
Kernel	RBF

H. Voting Classifier

The last classification algorithm used was a soft voting classifier. It is an ensemble classifier based on the following classifiers: Decision Trees, Multinomial Logistic Regression, K-Nearest Neighbors, Naïve Bayes, and Multilayer Perceptron. The soft voting classification works by first taking the individual probability provided by each classifier and taking the average. The data point is assigned to the class that has the highest average probability.

V. Results and Analysis

The following section will discuss the results of all the algorithms used and compare them against each other. The comparison will be made based on the following metrics: precision, recall, f1-score, and accuracy.

Precision is defined as:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

TP = True Positives, FP = False Positives
The recall is defined as:

$$recall = \frac{TP}{TP + FN} \tag{5}$$

FN = False Negatives, F1-Score is defined as:

$$F_1 = \frac{2}{\left(\frac{1}{precision}\right) + \left(\frac{1}{recall}\right)} \tag{6}$$

Accuracy is defined as:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

TN = True Negatives

The Decision Tree algorithm had an accuracy of 96%. It could recall 100% of the star instances but misidentified some, so the precision score was 98%. Decision Tree could not recall all of the galaxy and quasar instances, which is why their recall score was 98% and 86%, respectively. It correctly classified 96% of the galaxy and quasar instances; the precision score was 96%.

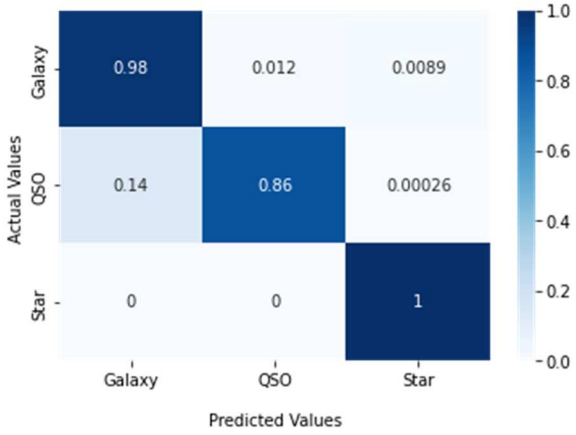

Fig. 15. Decision Tree Normalized Confusion Matrix

TABLE 9. Decision Tree Results

	Precision	Recall	F1-Score	Support
Galaxy	0%	98%	97%	11860
QSO	96%	86%	91%	3797
Star	98%	100%	99%	4343
Accuracy		96%		20000
Macro Average	96%	95%	95%	20000
Weighted Average	96%	96%	96%	20000

The K-Nearest Neighbors performed worse than Decision Trees at correctly classifying stars. The recall score was 99%, and the precision was 95%. However, it did perform better when it classified quasars, recall score was 92%, and precision was 97%. As for the galaxy class, the recall score was 97% which was lower than that of the decision tree at 98%. K-Nearest Neighbors had an accuracy of 97%.

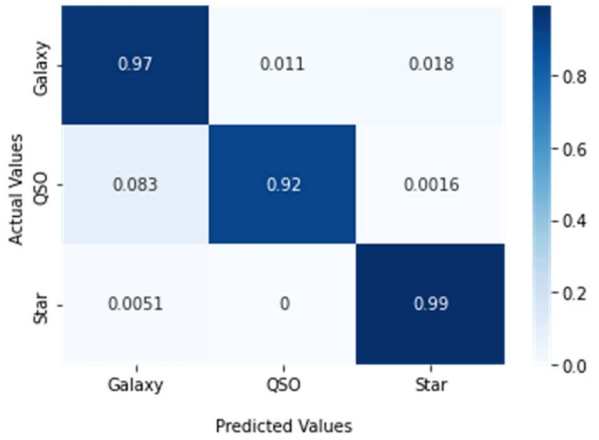

Fig. 16. K-Nearest Neighbors Normalized Confusion Matrix

TABLE 10. K-Nearest Neighbors Result

	Precision	Recall	F1-Score	Support
Galaxy	97%	97%	97%	11860
QSO	97%	92%	94%	3797
Star	95%	99%	97%	4343
Accuracy		97%		20000
Macro Average	96%	96%	96%	20000
Weighted Average	97%	97%	97%	20000

The Multinomial Logistic Regression algorithm had an accuracy of 96%. Like Decision Trees, it was able to recall all of the star instances, but it correctly classified 97% of the star instances. It performed the same as K-Nearest Neighbors when it classified galaxy instances. It had a recall score of 97% and the precision score of 97%. Multinomial Logistic Regression was better at recalling quasar instances than Decision Trees, recall score was 90%.

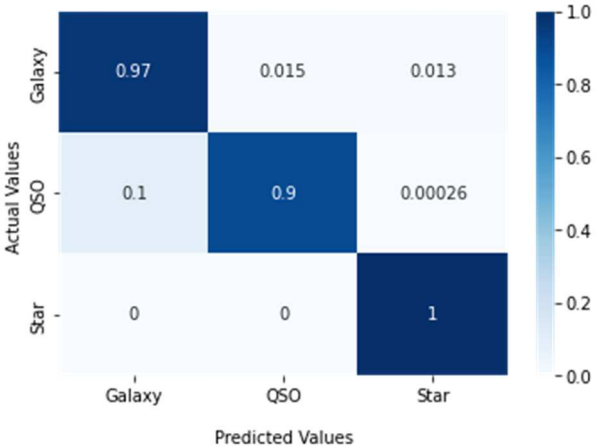

Fig. 17. Multinomial Logistic Regression Normalized Confusion Matrix

TABLE 11. Multinomial Logistic Regression Results

	Precision	Recall	F1-Score	Support
Galaxy	97%	97%	97%	11860
QSO	95%	90%	92%	3797
Star	97%	100%	98%	4343
Accuracy		96%		20000
Macro Average	96%	96%	96%	20000
Weighted Average	96%	96%	96%	20000

The Random Forest algorithm was the best-performing algorithm, with an accuracy of 98%. The algorithm correctly classified all the star instances, where both recall and precision scores were 100%. Recall and precision scores for class galaxy were 99% and 97%, respectively. Random Forest and Support Vector Classification were the best at recalling quasar instances, recall score was 92%.

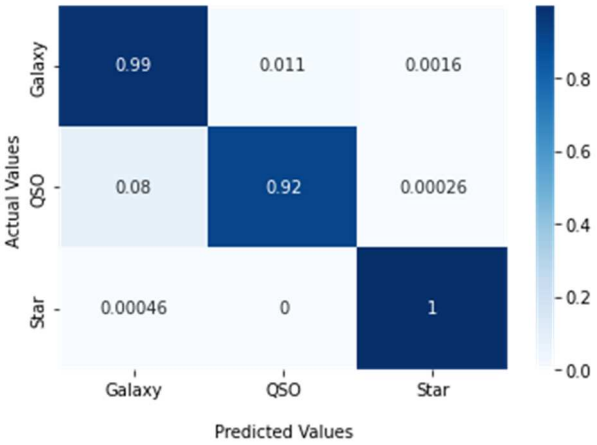

Fig. 18. Random Forest Normalized Confusion Matrix

TABLE 12. Random Forest Results

	Precision	Recall	F1-Score	Support
Galaxy	97%	99%	98%	11860
QSO	96%	92%	94%	3797
Star	100%	100%	100%	4343
Accuracy		98%		20000
Macro Average	98%	97%	97%	20000
Weighted Average	98%	98%	98%	20000

The Multilayer Perceptron had an accuracy of 97%. It performed the same as Multinomial Logistic Regression for the class star; recall and precision were 100% and 97%, respectively. For class quasar, it performed better than Multinomial Logistic Regression, recall score was 93%, and the precision score was 96%. Multilayer Perceptron was worse at recalling galaxy instances than Random Forest, recall score was 97%.

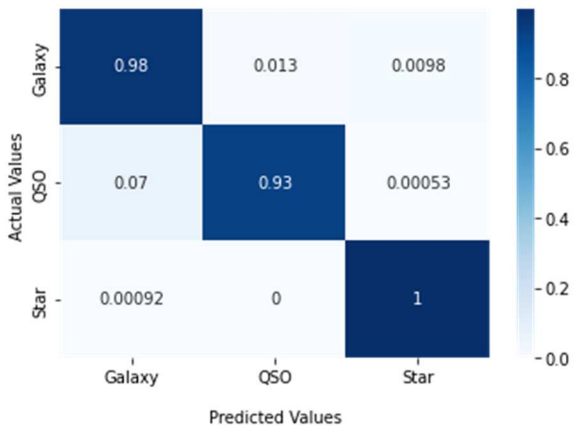

Fig. 19. Multilayer Perceptron Normalized Confusion Matrix

TABLE 13. Multilayer Perceptron Results

	Precision	Recall	F1-Score	Support
Galaxy	98%	98%	98%	11860
QSO	96%	93%	94%	3797
Star	97%	100%	99%	4343
Accuracy		97%		20000
Macro Average	97%	97%	97%	20000
Weighted Average	97%	97%	97%	20000

Naïve Bayes was the worst-performing algorithm, with an accuracy of 91%. It performed the worst at classifying quasar and galaxy instances, the recall was 74%, precision was 85% for class quasar, the recall was 90%, and precision was 95% for the class galaxy. Naïve Bayes had a recall score of 100% and 98% precision for star instances.

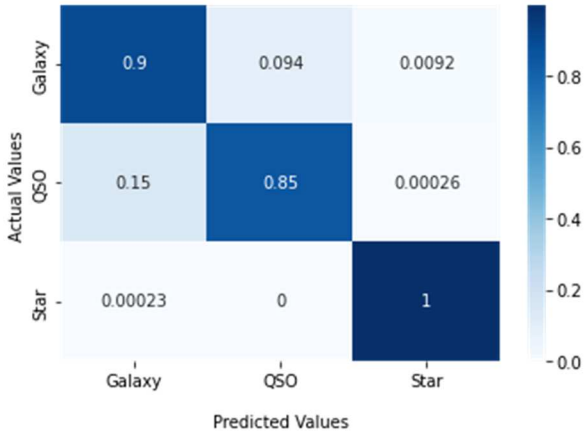

Fig. 20. Naive Bayes Normalized Confusion Matrix

TABLE 14. Naive Bayes Results

	Precision	Recall	F1-Score	Support
Galaxy	95%	90%	92%	11860
QSO	74%	85%	79%	3797
Star	98%	100%	99%	4343
Accuracy		91%		20000
Macro Average	89%	92%	90%	20000
Weighted Average	92%	91%	91%	20000

The Support Vector Classifier had an accuracy of 97%. The precision and recall score for class galaxy was 97% and 98%, respectively. For class quasar, the recall was 92%, and the precision was 97%. For class star, it had a recall score of 100% and the precision score of 97%.

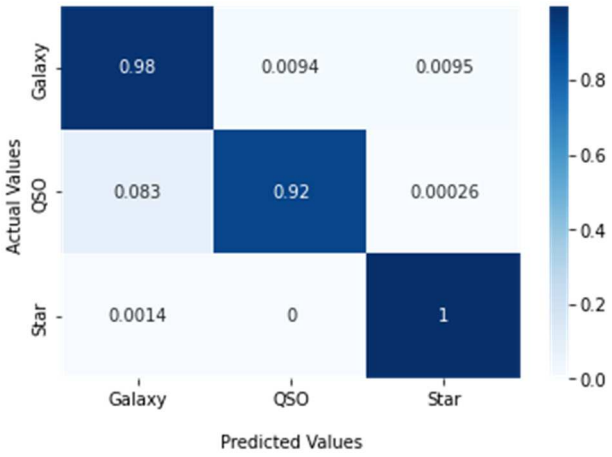

Fig. 21. Support Vector Classification Normalized Confusion Matrix

TABLE 15. Support Vector Classification Results

	Precision	Recall	F1-Score	Support
Galaxy	97%	98%	98%	11860
QSO	97%	92%	94%	3797
Star	97%	100%	99%	4343
Accuracy		97%		20000
Macro Average	97%	97%	97%	20000
Weighted Average	97%	97%	97%	20000

Lastly, Soft Voting Classifier. This algorithm also had an accuracy of 97%. Recall for classes star and galaxy was 100% and 98%, respectively. For class quasar, the recall was 91%. Classes star and galaxy had a precision score of 97%, and class quasar had a precision score of 96%.

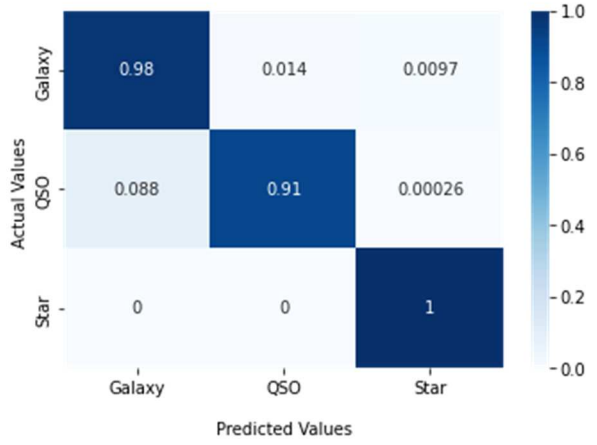

Fig. 22. Soft Voting Classifier Normalized Confusion Matrix

TABLE 16. Soft Voting Classifier Results

	Precision	Recall	F1-Score	Support
Galaxy	97%	98%	97%	11860
QSO	96%	91%	93%	3797
Star	97%	100%	99%	4343
Accuracy	97%			20000
Macro Average	97%	96%	96%	20000
Weighted Average	97%	97%	97%	20000

TABLE 17. Summary of the Accuracy of All the Algorithms Used

Algorithm	Accuracy
Decision Trees	96%
K-Nearest Neighbors	97%
Multinomial Logistic Regression	96%
Random Forest	98%
Multilayer Perceptron	97%
Naïve Bayes	91%
Support Vector Classifier	97%
Soft Voting Classifier	97%

VI. Conclusions

This paper used a supervised learning technique, classification, to classify instances from the SDSS DR17 dataset into either galaxy, quasar, or star. Eight algorithms were trained; the best classifier was Random Forest with an accuracy of 98%, and the worst classifier was Naïve Bayes with an accuracy of 91%. All of the algorithms performed best when classifying star instances. Random Forest could classify all the star instances in the dataset correctly. The lowest correct classification for all the algorithms was for instances labeled as quasars. This may be because quasar was the least represented class, and adding more instances with that class could help the algorithms perform better.

References

[1] C. Zuckerman, "Everything you wanted to know about stars," National Geographic, 20 March 2019. [Online]. Available: https://www.nationalgeographic.com/science/article/stars.

[2] "What Is a Galaxy?," NASA Science Space Place, [Online]. Available: https://spaceplace.nasa.gov/galaxy/en/.

[3] B. Peterson, "Quasar," Encyclopedia Britannica, [Online]. Available: https://www.britannica.com/science/quasar/Finding-quasars. [Accessed 29 May 2022].

[4] "The Sloan Digital Sky Survey," Max Planck Institute for Astronomy, [Online]. Available: https://www.mpia.de/en/research/collaborations/sdssiv.

[5] N. Abdulhadi and A. A. Al-Mousa, "Diabetes Detection Using Machine Learning Classification Methods," in *2021 International Conference on Information Technology (ICIT)*, Amman, 2021.

[6] A. Atwah and A. A. Al-Mousa, "Car Accident Severity Classification Using Machine Learning," in *2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, Zallaq, Bahrain, 2021.

[7] Z. Bitar and A. A. Al-Mousa, "Prediction of Graduate Admission using Multiple Supervised Machine Learning Models," in *IEEE SoutheastCon*, Raleigh, 2020.

[8] A. R. Bhamare, A. Baral and S. Agarwal, "Analysis of Kepler objects of interest using machine learning for Exoplanet Identification," in *International Conference on Intelligent Technologies (CONIT)*, 2021.

[9] D. A. Petrusevich, "Implementation of machine learning algorithms in the sloan digital sky survey DR14 analysis," *IOP Conference Series: Materials Science and Engineering,* vol. 862, no. 4, 2020.

[10] T. Chuntama, P. Techa-Angkoon and C. Suwannajak, "Multiclass Classification of Astronomical Objects in the Galaxy M81 using Machine Learning Techniques," in *2020 24th International Computer Science and Engineering Conference (ICSEC)*, 2020.

[11] fedesoriano, "Stellar Classification Dataset - SDSS17," January 2022. [Online]. Available: https://www.kaggle.com/fedesoriano/stellar-classification-dataset-sdss17.

[12] I.-K. Yeo and R. A. Johnson, "A new family of power transformations to improve normality or symmetry," *Biometrika,* vol. 87, no. 4, p. 954–959, December 2000.

[13] M. A. T. Rony, D. S. A. A. Reza, R. Mostafa and M. A. Ullah, "Application of Machine Learning to Interpret Predictability of Different Models: Approach to Classification for SDSS Sources," in *021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, 2021.

[14] "Machine Learning," Center for Astrophysics Harvard & Smithsonian , [Online]. Available: https://cfa.harvard.edu/research/topic/machine-learning.