Classifying Quasars, Galaxies And Stars Using Photometric Data

Ans IMRAN

Contents

1 Overview

The classification of galaxies, quasars, and stars is a foundational aspect of astronomy. Early star catalogues showed stars as part of our galaxy, with Andromeda later identified as a separate galaxy. Quasars were discovered through their radio emissions and high redshifts. Classifying these sources is essential for scientific research, driving the development of large-scale facilities like the SKA and LSST. These projects aim to catalogue billions of sources, making manual classification of catalogued objects impractical, hence necessitating machine learning for efficient classification. Photometric data, has proven effective for distinguishing between stars, galaxies, and quasars. This report demonstrates that machine learning models, particularly Random Forest, can very accurately classify these astronomical objects using photometric data from the Sloan Digital Sky Survey.

2 Introduction

The classification scheme of galaxies, quasars, and stars is foundational in astronomy. Initial star catalogues revealed that stars form our galaxy (Herschel [13]). Later, Andromeda was identified as a separate galaxy (Opik [23]; Hubble [14]), and then more galaxies were catalogued with advanced telescopes. Quasars were identified when radio emissions were detected from unresolved star-like sources with high redshifts (Smith and Hoffleit [27]; Greenstein and Matthews [11]), attributed to accretion disks around supermassive black holes (Burbidge, Burbidge, and Sandage [6]). Quasars' emissions primarily come from active galactic nuclei (AGN) (Urry and Padovani [28]), often identified by their high-ionisation emission lines (e.g., C IV, Mg II) in optical spectra (Francis et al. [10]; Berk et al. [4]). Nearby resolved galaxies with bright, compact cores and AGN emissions, known as Seyfert galaxies (Weedman [29]; Antonucci [3]), are also categorized as quasars.

Classifying astronomical sources is crucial for individual system studies and statistical population analyses. Large quasar samples are essential for various scientific objectives, driving the development of facilities like the Square Kilometre Array (SKA; Bourke et al. [5]) and the Large Synoptic Survey Telescope (LSST; Ivezić et al. [15]). These objectives include Lyman- α forest surveys (Rauch [25]), cosmic magnetism studies (Scranton et al. [26]), general cosmology (Leistedt and Peiris [19]), and galaxy evolution (Kauffmann and Haehnelt [18]).

Millions of sources have been catalogued by surveys such as the Sloan Digital Sky Survey (SDSS; Aguado et al. [2]), the Wide-field Infrared Survey Explorer (WISE; Wright et al. [30]), and the LOw Frequency ARray (LOFAR; Haarlem et al. [12]). Future telescopes are expected to exponentially increase these catalogues, with the LSST projecting around 20 billion stars and a similar number of galaxies (Ivezić et al. [15]). The SKA aims to detect billions of sources (Bourke et al. [5]), making manual verification and labelling of these sources impractical. Projects like Galaxy Zoo (Lintott et al. [20]) help in labelling, but slone, they are not expected to be able to keep up with the source counts anticipated for the next generation of telescopes. Therefore, machine learning algorithms are increasingly being used for large-scale data analysis (Jones [16]).

Distinguishing astronomical source types is straightforward with detailed data like spectroscopy and multi-wavelength observations. However, obtaining such detailed observations for millions of sources is time-consuming and impractical with current telescope survey speeds. Instead, photometry, measuring light in multiple wavebands, provides a faster classification method based on colours (u-g and g-r for SDSS; w1-w2 and w2-w3

for WISE) (Nikutta et al. [22]; Peters et al. [24]). Photometry captures the overall spectral shape, distinguishing stars, galaxies, and quasars. Studies have shown photometry's effectiveness as a machine learning feature for source classification (Carrasco et al. [7]; Kang et al. [17]).

This report uses the methodologies used in the study Clarke et al. [8], to reproduce the result that Quasars, Galaxies and Stars can indeed be classified based on photometric data, using machine learning models.

This report replicates some of the methodologies used in the study Clarke et al. [8] to demonstrate that quasars, galaxies, and stars can indeed be effectively classified by machine learning models using photometric data.

I utilized data from the Seventeenth Data Release of the Sloan Digital Sky Surveys, sourced from Kaggle. Irrelevant features were removed, and data cleaning involved filtering out erroneous values and addressing class imbalance through undersampling. The dataset was split into training, validation, and test sets (60%, 20%, 20%), and standardized using z-score normalization. Visualization using PCA demonstrated clear class separation, indicating good potential for classification. Various machine learning models were trained, with Random Forest achieving the highest accuracy (97.5%). Model ensembling did not enhance performance, making the Random Forest the best standalone model.
The github link of the scientific project: https://github.com/AnsImran/Classifying-Quasars-Galaxies-Stars-using-Photometryhttps://github.com/AnsImran/Classifying-Quasars-Galaxies-Stars-using-Photometry

3 Scientific Concepts

3.1 Random Forest

Random Forest is a widely used machine learning algorithm. To understand Random Forest, we first need to understand "Decision Trees".

3.1.1 Decision Trees

A Decision Tree is a supervised machine learning algorithm, meaning it requires labeled data for training. It is used for both classification and regression tasks.

Let's assume we have 100,000 unlabeled objects that we want to classify into one of two groups/classes: A or E. Each object belongs to only one class and has n features, $x_1, x_2, \ldots, x_n$. Each feature describes a specific property of the objects. For example, feature x_1 might indicate whether the given object has four corners or not. If feature x_1 is present in object-a, then x_1 takes a value of 1 for object-a. If feature x_1 is not present, x_1 takes a value of 0 for object-a. For example, if our objects are polygons and object-a is a triangle, then x_1 will be 0 for the triangle. On the other hand, if object-a were a tetragon, x_1 would be 1 for object-a. Notice, for the sake of simplicity, we are using only binary-valued features here.

One way to solve this problem, would be, to look at each of the 100,000 objects and manually classify them as triangles, tetragons or pentagons etc. But this would be a very very time consuming process. Luckily, we have a solution! Instead of manually labelling the objects, we can use a Decision tree to classify these objects for us.

Now, let's assume we have several interns working under our supervision. We randomly pick 5000 objects and give them to the interns for classification. These 5000 human-labeled objects will become our labeled data. We'll use this data to train and test our machine learning algorithm. Out of the 5000 labeled objects, we'll use 3000 to train our decision tree algorithm. The remaining 2000 objects will be used for testing and cross-validation.

First, we'll classify the 3000 objects based on each of the n features. We'll start with feature x_1 and classify the 3000 training objects based on whether x_1 is present in a given object. We can visualize this process as a node with one input at the top and two outputs at the bottom: a left bucket and a right bucket. The node acts as a filter, taking the training objects as inputs. If x_1 is present in an object, it sends it to the left bucket. If x_1 is not present, it sends it to the right bucket. In the terminology of Decision Trees,w e call this object as a **root node**.

Now, we repeat the procedure for all n features. For each feature, we again place all 3000 examples at the top of the corresponding node and filter them into left or right buckets, depending on whether the current feature is present or absent in a given object. This process results in n different root nodes, one for each feature.

Now, out of the n root nodes, we'll select the one that provides the maximum information gain to be "THE ROOT NODE" of our decision tree. We'll discard all other root nodes.

Information Gain:
Before understanding the information gain we first have to understand the entropy of a bucket/branch.

Recall that we are conducting a binary classification. Suppose our root node directs 1,000 objects to the left branch and classifies them as class A. The remaining 2,000 objects go to the right branch and are classified as class E. For both branches, to calculate the entropy, we count the total number of class-A objects in that branch and divide it by the total number of objects in the same branch. This value represents the probability '**p**' of finding a class-A object in that branch.

Now, we calculate the entropy of bucket/branch using the following formula:

$$H(p) = -plog_2(p) - (1-p)log_2(1-p)$$

Note: If we encounter $0\log_2(0)$ in the above formula, the computer should automatically set it to zero, i.e., $0\log_2(0) = 0$. This adjustment will have to be coded into the program to handle such cases.

where:

$H(p)$: Entropy of the bucket/branch

p : The probability of finding an object of class-A in the bucket/branch

$$p = \frac{\text{total number of objects belonging to class-A in the current branch}}{\text{total number of objects in the current branch}}$$

Now that we have an understanding of the entropy of a branch, we can go on to the definition of Information Gain. Information Gain is defined as follows:

$$\text{Information Gain} = H(p^{root}) - \left(w^{left}\ H(p^{left})\ +\ w^{right}\ H(p^{right}) \right)$$

where

$$w^{left} = \frac{\text{Number of total objects in the left branch}}{\text{Number of total objects at root node}}$$

$$w^{right} = \frac{\text{Number of total objects in the right branch}}{\text{Number of total objects at root node}}$$

$H(p^{root})$: The entropy of the data at the root node. It refers to the entropy calculated prior to the objects being partitioned into left or right branches.

$H(p^{left})$: The entropy of left branch

$H(p^{right})$: The entropy of right branch

So that's how we compute the information gain, and based on this computation, we determine which feature to select at the root node.

The left and the right branches are called Decision Nodes. Once we have partitioned the original data to the left and right decision nodes, each node is treated as a new root node (but now we call it a decision node), and we repeat the entire procedure as discussed. The only difference is that this time, instead of selecting a feature for the node from all n features, we exclude the feature that has already been used. This results in two new branches from each of the original left and right decision nodes.

For these four new decision nodes, the process is repeated for each new node, always excluding features that have already been used in that part of the tree. This recursive algorithm continues until all features have been used, or another stopping criterion is met, such as achieving pure nodes (all the objects that were partitioned to this node were either class-A or class-B) or reaching a maximum tree depth. By following this recursive process, we build an entire decision tree, stemming from the root node. The nodes at the end of the tree are called leaf nodes. Figure 1 shows an illustration of a decision tree.

3.1.2 Sampling With Replacement

Decision trees are highly sensitive to the training dataset, meaning even a small change in the data can result in a significantly different tree. To reduce this sensitivity, one effective method is to train multiple decision trees. This creates an ensemble of trees. When making predictions, an object is passed through the ensemble, and each tree provides its prediction. The final prediction is determined by selecting the one that receives the highest number of votes from the individual trees.

To obtain multiple trees, we need to alter the training data. One effective method is to generate new datasets from the original dataset through sampling with replacement. This technique, allows us to create diverse training datasets, each contributing to the development of a unique decision tree.

To understand sampling with replacement, imagine placing all the objects from the training dataset into a virtual bag. We shake the bag well and then randomly pick one object from it, noting its name and corresponding properties in a list. We then put the object back into the bag. Shaking the bag again, we pick another object, record its name

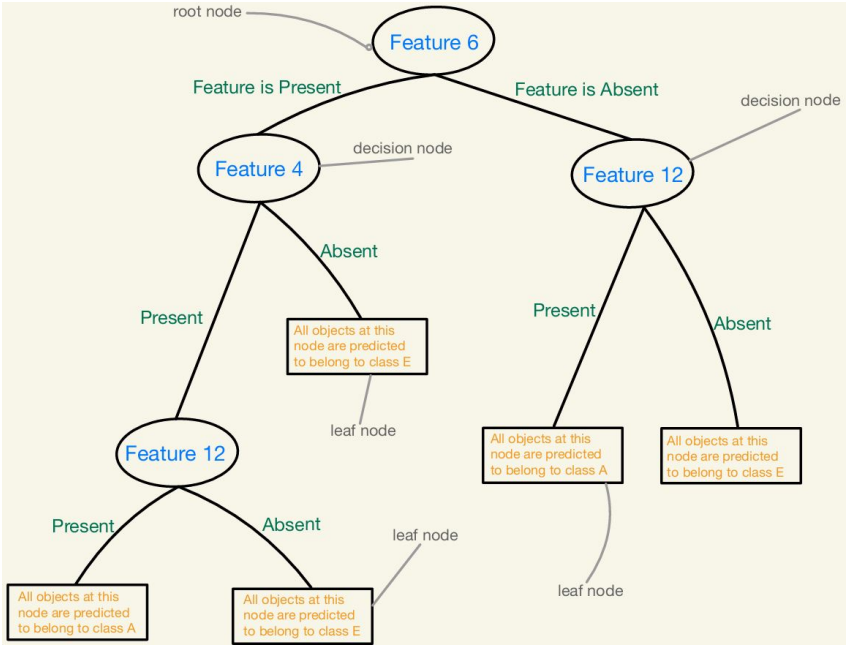

Figure 1: **Illustration of a decision tree**

and features on the same list, and return it to the bag. Notice that there is a non-zero probability of picking the same object multiple times. By repeating this procedure 3,000 times, we create a new training dataset, equal in size to our original dataset. Although this new dataset is generated from the original, it will be somewhat different due to the sampling with replacement. This process helps in generating multiple diverse training datasets from the original one. These newly generated training datasets can then be used to train new decision trees and thus creating an ensemble of decision trees.

3.1.3 Random Forest

We previously explored how to create an ensemble of decision trees using sampling with replacement. However, this method can sometimes result in decision trees that are quite similar, especially near the root node, which can negatively impact the performance of the algorithm. To enhance the algorithm and transform it into a Random Forest, we have to introduce an important modification.

Randomizing the feature choice: *At each node, instead of considering all the available features m when choosing a feature to split on, we randomly select from a subset of $k < m$ features.*

This approach increases the diversity among the trees in the ensemble, leading to more accurate results when aggregating their predictions. When m is large, such as in the dozens or thousands, it is recommended to use $k = \sqrt{m}$. With this modification, our algorithm becomes a Random Forest, which is significantly more robust than a single decision tree or even a simple ensemble of trees. For further details of Random Forest see Louppe [21].

4 Descripition of my Work

4.1 Data

The data originally came from **The Seventeenth Data Release of the Sloan Digital Sky Surveys: Complete Release of MaNGA, MaStar and APOGEE-2 Data** see Accetta et al. [1]. I got a subset of this data from Kaggle, see fedesoriano [9].

The relevant features of the data are as follows:

Feature Name	Description
u	Ultraviolet filter in the photometric system
g	Green filter in the photometric system
r	Red filter in the photometric system
i	Near Infrared filter in the photometric system
z	Infrared filter in the photometric system
redshift	Redshift value based on the increase in wavelength
class	Object class (galaxy, star, or quasar object)

Table 1: Description of features relevant for classification

4.1.1 Data Cleaning and Data Preparation

Data Cleaning:

First, I removed all irrelevant features from the dataset, such as the object ID and the date on which the an object was catalogued etc. I selected only the features described in Table 1.

I then applied filters to the data to identify any entries containing null, unusually high/low, or unphysical values. One of the records contained an unrealistic value of -9999 in the entries corresponding to the u, g, and z features. I removed this record/row from the dataset.

Addressing Class Imbalance:

When dealing with a multiclass classification problem, it is crucial to consider the distribution of samples across different classes in the dataset. If the number of samples corresponding to each class is not equal, it is referred to as class imbalance. In such scenarios, the machine learning model tends to perform better in predicting the class with a higher number of samples while struggling to accurately classify objects from other classes with fewer samples. Thus, class imbalance can significantly impact the performance of a machine learning model. Therefore removing class imbalance from the dataset is necessary.

There are several methods to address class imbalance. The technique I used is straightforward and is known as undersampling. In this approach, we focus on the class with the fewest samples and retain its sample count as it is. Then, we randomly remove the excess samples from the other classes until their counts match the number of samples in the minority class. This way, we balance the dataset by reducing the number of samples in the overrepresented classes.

In the dataset, Quasars were the least represented class, with 18,961 labeled instances. So, I performed undersampling, reducing the number of samples in the other classes

(Galaxies and Stars) to match the 18,961 samples of Quasars. This resulted in a balanced dataset with a total of $18,961 \times 3 = 56,883$ samples, distributed equally among the three classes: Quasars, Galaxies, and Stars, each having 18,961 samples.

Creating Training, Validation, and Test Data Splits:
Spitting the original dataset into training, validation, and test datasets is crucial in machine learning projects for several reasons:

Training data is used to teach the model, allowing it to learn patterns and relationships. Validation data helps tune hyperparameters and assess the model's performance during development, preventing overfitting to the training data. Test data provides an unbiased evaluation of the final model's performance on completely unseen data, simulating real-world application.
This division ensures the model can generalize well to new, unseen data and helps detect issues like overfitting (A situation when the model performs well when making predictions on the data that was used to train it, but performs poorly on unseen-data).

To create the training, validation, and test datasets, I randomly split the data, allocating 60% to the training dataset, 20% to the validation dataset, and 20% to the test dataset.

Standardizing the data:

Standardizing data is crucial for faster and smoother convergence in machine learning algorithms. To standardize the data, I used z-score normalization. In z-score normalization, for each data point x of each feature in the original dataset, the following formula is applied:

$$z = \frac{x - \mu}{\sigma}$$

where
z : The standardized data point, corresponding to point x.
μ : The mean of all data points for a given feature.
σ : The standard deviation of all data points for a given feature.

4.2 Data Visualization

Visualizing the data is crucial because it helps in understanding the data better. It reveals patterns and distributions that might not be obvious otherwise. Visualization gives an early indication of how well our machine learning model might perform.

For example, in our multiclass classification problem, we can use dimensionality reduction techniques to create a 2D or 3D representation of the data. By plotting this, we can visualize the different classes (quasars, galaxies, stars) in 2d/3d. If the different classes are visually separate, we can expect our ML algorithms to perform well. If the distinctions are unclear, high performance from machine learning models is less likely.

Therefore, I utilized principal component analysis (PCA) as a dimensionality reduction technique to generate a 3D representation of the photometric data. Figure 2 showcases various screenshots captured from different angles of this 3D plot. The clear separation between different classes indicates that we can definitely expect our machine learning models to perform very well on this data.

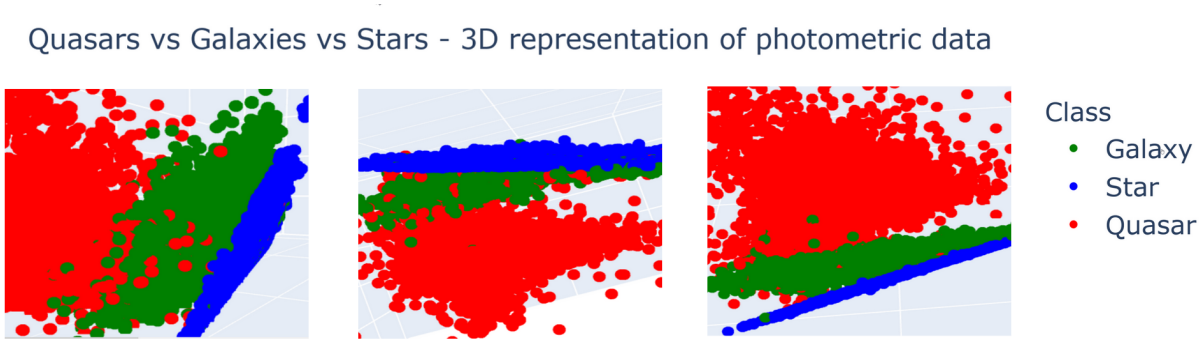

Figure 2: **Quasars, Galaxies and Stars in photometric space**...

4.3 Training Machine Learning Models

I trained different machine models to see which one works best for us. In the following sections, I will provide an overview of the training process for each model.

4.3.1 Gaussian Mixture Model - Unsupervised Learning Approach

Gaussian Mixture Model (GMM) is an unsupervised machine learning algorithm. Given a training dataset and the number of clusters, it can group the data into the specified number of clusters. Once trained, it can also return predictions in the form of probabilities - the probability of an input sample belonging to the quasar cluster, galaxy cluster, or star cluster.
Its accuracy was 73.47% on the validation dataset and 72.89% on the test dataset, which is a very good result as it tells us that this problem is indeed solvable by machine learning, as we expected from the data visualization.

4.3.2 Dense Neural Network - Supervised Learning Approach

A dense neural network, also known as a fully connected neural network, is a type of artificial neural network where each neuron in one layer is connected to every neuron in the subsequent layer.

Structure:

Input layer : Receives initial data

Hidden layers : Process information

Output layer : Makes Predictions

The details of neural network's implementation can be found on the https://github.com/AnsImran/C Quasars-Galaxies-Stars-using-Photometrygithub repo of this scientific project.

The accuracy of the neural network was around **96.51 - 96.67 %** for the validation, and test datasets repectively. A huge increase in accuracy over the Gaussian Mixture Model. This indicates a well-generalized model, as it performed very well on test dataset i.e. unseen dataset - not seen during training phase.

4.3.3 XGBoost Model - Supervised Learning Approach

XGBoost is a supervised learning algorithm that is an enhanced version of the Boosted Trees algorithm, which in turn is an adaptation of the Random Forest algorithm.

Boosted Trees: The Boosted Trees algorithm resembles the Random Forest algorithm with one key difference. In Boosted Trees, the sampling with replacement procedure of Random Forest is altered. The first decision tree in Boosted Trees is built in the same manner as in Random Forest. However, from the second tree onwards, the algorithm increases the probability of selecting samples that were misclassified by previous trees during the ***sampling with replacement*** process. This adjustment compels the new decision trees to focus more on correctly classifying the samples that earlier trees failed to classify accurately.

XGBoost, which stands for Extreme Gradient Boosting, is an advanced version of the Boosted Trees algorithm. The specifics of XGBoost are beyond the scope of this report.

In my project, I also trained an XGBoost model, using all the default hyperparameters. The XGBoost model achieved an accuracy of **97.24% and 97.36%** on the validation and test datasets respectively, surpassing the performance of the dense neural network!

4.3.4 Random Forest Model - Supervised Learning Approach

For the details of Random Forest Algorithm, kindly refer to section 3.1.3.

I used the Random Forest Algorithm with all of it's default hyper parameters. The accuracy of the Random Forest model was so far the best. It was **97.38%** on validation dataset **97.47%** on test dataset.

4.4 Model Ensembling

Different machine learning models analyze problems differently. For instance, a random forest algorithm evaluates a problem differently than a neural network. Some approaches may yield correct solutions, while others may not. To enhance the accuracy of our solutions, it's beneficial to consider a diverse range of methods before making a decision.

In the context of multiclass classification, we can achieve this by training multiple machine learning models, such as a gaussian mixture model, random forest, a neural network, and an XGBoost model. Once trained, we can aggregate their predictions to form the final output. This is called model ensembling, aggregating the outputs of multiple models. This method leverages the strengths of various models, ensuring that the final prediction benefits from the collective insights of all models involved. By considering multiple perspectives, we increase the likelihood of arriving at a more accurate and robust solution.

There are multiple ways to perform model ensembling. The approach I have used is as follows:

When determining whether a given sample is a quasar, star, or galaxy, you input this sample into all the machine learning models you have trained. Each model then provides predictions in the form of probabilities for each class. For example, the predictions might look like this:

The probability that the input sample is a quasar is 0.7

The probability that the input sample is a star is 0.2

The probability that the input sample is a galaxy is 0.1

Notice, the sum of probabilities for each prediction equals 1.

Then I associated weights with the predictions of each model. The sum of weights must be equal to 1. And then I averaged over all weighted predictions. The overall equation looks like this.

$$\text{final prediction} = \frac{w_1 \times \text{gmm} + w_2 \times \text{random-forest} + w_3 \times \text{xgboost} + w_4 \times \text{neural-net}}{4}$$

where:

$$w_1 + w_2 + w_3 + w_4 = 1$$
$$gmm : \text{The predictions by Gaussian Mixture Model}$$
$$\text{random-forest} : \text{The predictions by Random Forest Model}$$
$$\text{xgboost} : \text{The predictions by XGBoost Model}$$
$$\text{neural-net} : \text{The predictions by the Dense Neural Network}$$

Note: In the above equation, the predictions from each model are in the form of a tuple (0.7, 0.2, 0.1). When summing the weighted predictions, only the corresponding entries of each tuple are summed. At the end, each of the final three probabilities is divided by 4 to ensure the probabilities remain normalized.

At this point, a natural question arises: how do we decide how to weigh the predictions, i.e., how do we determine the strength of weight to associate with each model? There are various methods for optimizing these weights. The one I used is called Simple Random Search for Weight Optimization. The details of this method are beyond the scope of this report.

The weights assigned to each machine learning model by the Simple Random Search algorithm are as follows:

w_1 for gmm $= 0.0038$
w_2 for random-forest $= 0.6918$
w_3 for xgboost $= 0.2786$
w_4 for neural-net $= 0.0255$

The weight associated with the predictions of the random forest is the highest. This is understandable, as the random forest had the highest accuracy on the validation and test datasets. The weight associated with the predictions of the GMM is the lowest, which was expected since it had the lowest accuracy.

However, the weight associated with the neural network is unusually low compared to XGBoost and the random forest, despite having comparable accuracy. This suggests that although the neural network is producing the right results, it is doing so for the wrong reasons.

4.5 Final Results

The final results after aggregating the outputs of all the models are as follows:

- The ensemble of all models produced an accuracy of **97.61%** on the validation dataset, which is better than the random forest alone.
- The ensemble of all models produced an accuracy of **97.32%** on the test dataset, slightly worse than the random forest alone.

These results make sense, as I used the validation dataset in the Simple Random Search to optimize the prediction weight of each model. The weights likely overfitted to the

validation dataset, which is why the model performs slightly better on the validation dataset and slightly worse on the test dataset.
In conclusion, model ensembling in this case is not working well. Instead of improving the overall accuracy, the weaker models are pulling down the accuracy of the random forest. Therefore, using only the random forest model would be a better approach.

Figure 3 shows two confusion matrices corresponding to the predictions by the Random Forest on the validation and test datasets. The algorithm does not confuse quasars with stars, but it does confuse galaxies with both quasars and stars. This observation suggests that making threshold adjustments (details of which are beyond the scope of this report) between distinguishing quasars from galaxies and galaxies from stars could potentially improve the overall accuracy of the model.

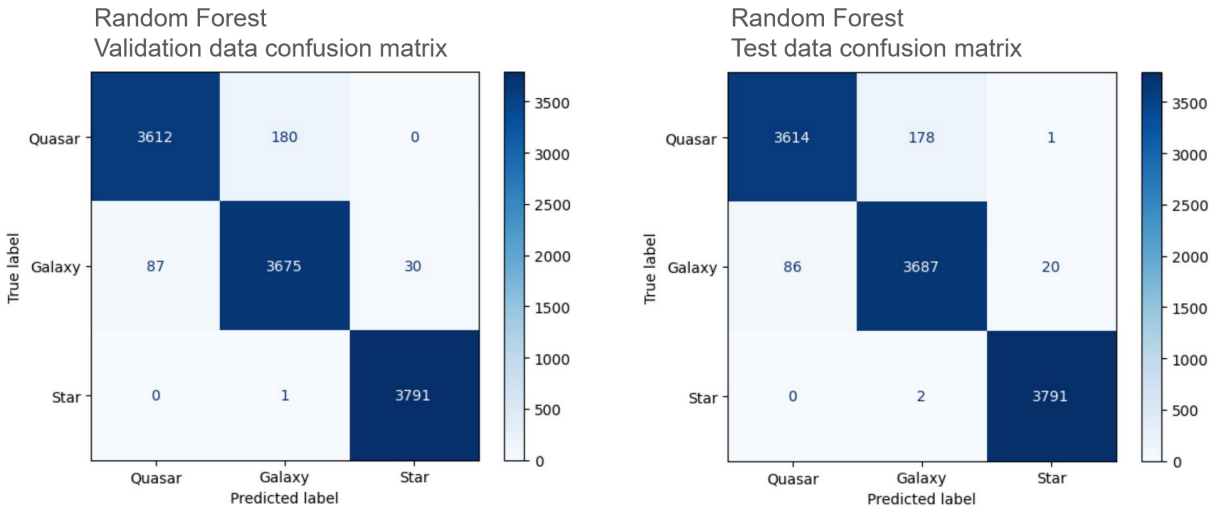

Figure 3: **Confusion matrices for the predictions by Random Forest**

5 Conclusion and Future Work?

- First, we cleaned the data to remove problematic samples. Next, we normalized the data to accelerate convergence of the machine learning models and to ensure equal importance among all features for solving the problem.
- Following normalization, we visualized the data in three dimensions and observed distinct clusters for the three classes: quasars, stars, and galaxies.
- Subsequently, we trained various machine learning models to identify the best performer for our scenario, with Random Forest proving most effective having an accuracy close to 97.5 %.
- Although we attempted model ensembling, it did not yield improved results compared to the Random Forest alone. We also generated confusion matrices for the Random Forest predictions, highlighting potential benefits from adjusting thresholds to enhance performance.

The key finding is that quasars, stars, and galaxies can indeed be classified using photometric data, achieving an impressive accuracy of over 97% with our initial analysis. Further enhancements in machine learning models and additional data preprocessing

could potentially boost accuracy significantly. By analyzing misclassified samples and identifying common issues, we may refine our preprocessing steps to address these challenges and improve model performance.

References

[1] Katherine Accetta et al. "The seventeenth data release of the Sloan Digital Sky Surveys: Complete release of MaNGA, MaStar, and APOGEE-2 data". In: *The Astrophysical Journal Supplement Series* 259.2 (2022), p. 35.

[2] David Sánchez Aguado et al. "The fifteenth data release of the Sloan Digital Sky Surveys: first release of MaNGA-derived quantities, data visualization tools, and stellar library". In: *The Astrophysical Journal Supplement Series* 240.2 (2019), p. 23.

[3] R. Antonucci. "A panchromatic review of thermal and nonthermal active galactic nuclei". In: *Astronomical and Astrophysical Transactions* 27.4 (Jan. 2012), pp. 557–602. DOI: 10.48550/arXiv.1210.2716. arXiv: 1210.2716 [astro-ph.CO].

[4] Daniel E Vanden Berk et al. "Composite quasar spectra from the sloan digital sky survey". In: *The Astronomical Journal* 122.2 (2001), p. 549.

[5] Tyler Bourke et al. "Advancing Astrophysics with the Square Kilometre Array (AASKA14)". In: (2015).

[6] GR Burbidge, E Margaret Burbidge, and Allan R Sandage. "Evidence for the occurrence of violent events in the nuclei of galaxies". In: *Reviews of Modern Physics* 35.4 (1963), p. 947.

[7] D Carrasco et al. "Photometric classification of quasars from RCS-2 using Random Forest". In: *Astronomy & Astrophysics* 584 (2015), A44.

[8] AO Clarke et al. "Identifying galaxies, quasars, and stars with machine learning: A new catalogue of classifications for 111 million SDSS sources without spectra". In: *Astronomy & Astrophysics* 639 (2020), A84.

[9] fedesoriano. "Stellar Classification Dataset - SDSS17". In: *Kaggle.com* (2022).

[10] Paul J. Francis et al. "A High Signal-to-Noise Ratio Composite Quasar Spectrum". In: 373 (June 1991), p. 465. DOI: 10.1086/170066.

[11] Jesse L Greenstein and Thomas A Matthews. "Redshift of the Radio Source 3C 48." In: *Astronomical Journal, Vol. 68, p. 279* 68 (1963), p. 279.

[12] Michael P van Haarlem et al. "LOFAR: The low-frequency array". In: *Astronomy & astrophysics* 556 (2013), A2.

[13] William Herschel. "XX. Catalogue of a second thousand of new nebulæ and clusters of stars; with a few introductory remarks on the construction of the heavens". In: *Philosophical Transactions of the Royal Society of London* 79 (1789), pp. 212–255.

[14] Edwin P Hubble. "A spiral nebula as a stellar system, Messier 31." In: *Astrophysical Journal, 69, 103-158 (1929)* 69 (1929).

[15] Željko Ivezić et al. "LSST: from science drivers to reference design and anticipated data products". In: *The Astrophysical Journal* 873.2 (2019), p. 111.

[16] Nicola Jones. "The learning machines". In: *Nature* 505.7482 (2014), p. 146.

[17] Shi-Ju Kang et al. "Evaluating the optical classification of Fermi BCUs using machine learning". In: *The Astrophysical Journal* 872.2 (2019), p. 189.

[18] Guinevere Kauffmann and Martin Haehnelt. "A unified model for the evolution of galaxies and quasars". In: *Monthly Notices of the Royal Astronomical Society* 311.3 (2000), pp. 576–588.

[19] Boris Leistedt and Hiranya V Peiris. "Exploiting the full potential of photometric quasar surveys: optimal power spectra through blind mitigation of systematics". In: *Monthly Notices of the Royal Astronomical Society* 444.1 (2014), pp. 2–14.

[20] Chris J Lintott et al. "Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey". In: *Monthly Notices of the Royal Astronomical Society* 389.3 (2008), pp. 1179–1189.

[21] Gilles Louppe. "Understanding random forests: From theory to practice". In: *arXiv preprint arXiv:1407.7502* (2014).

[22] Robert Nikutta et al. "The meaning of WISE colours–I. The Galaxy and its satellites". In: *Monthly Notices of the Royal Astronomical Society* 442.4 (2014), pp. 3361–3379.

[23] Ernst Opik. "An estimate of the distance of the Andromeda Nebula." In: *Astrophysical Journal, 55, 406-410 (1922)* 55 (1922).

[24] Christina M. Peters et al. "Quasar Classification Using Color and Variability". In: 811.2, 95 (Oct. 2015), p. 95. DOI: 10.1088/0004-637X/811/2/95. arXiv: 1508.04121 [astro-ph.GA].

[25] Michael Rauch. "The Lyman alpha forest in the spectra of quasistellar objects". In: *Annual Review of Astronomy and Astrophysics* 36.1 (1998), pp. 267–316.

[26] Ryan Scranton et al. "Detection of cosmic magnification with the Sloan Digital Sky Survey". In: *The Astrophysical Journal* 633.2 (2005), p. 589.

[27] Harlan J Smith and Dorrit Hoffleit. "Photographic History and Suggested Nature of the Radio Source 3C 48". In: *Publications of the Astronomical Society of the Pacific* 73.434 (1961), pp. 292–300.

[28] C Megan Urry and Paolo Padovani. "Unified schemes for radio-loud active galactic nuclei". In: *Publications of the Astronomical Society of the Pacific* 107.715 (1995), p. 803.

[29] Daniel W Weedman. "Seyfert galaxies". In: *Annual review of astronomy and astrophysics* 15.1 (1977), pp. 69–95.

[30] Edward L Wright et al. "The Wide-field Infrared Survey Explorer (WISE): mission description and initial on-orbit performance". In: *The Astronomical Journal* 140.6 (2010), p. 1868.