Astronomical Survey Data Classification

ECS-308, Data Science and Machine Learning
Course Instructor: Dr. Kushal K Shah

Vatsalya Sharan, 18307
Department of Physics
IISER Bhopal

Preface

The choice of this final project originally stemmed from my passion for utilizing Machine Learning techniques to understand and solve the various problems in the domain of physics. Having tackled a Physics problem of different nature in the previous semester, tackling classification problems in Astronomy using vast Data-sets and Surveys available seemed like a natural choice going further in this domain.

The completion of this project gives me much pleasure. I would like to show my gratitude to Dr.Kushal K Shah, Course Instructor, IISER Bhopal for giving me an opportunity to work on such an enriching project with clear guidelines and guidance throughout the course of this semester.
I would also like to thank him for being patient with the students and postponing the deadlines for the convenience of the students during the pandemic peaks and lock-downs.

I also thank the Indian Institute for Science Education and Research, Bhopal, for providing a conducive environment in the early stages, greatly facilitating the project and the decision making.

I would also like to expand my deepest gratitude to all those who have directly and indirectly guided me in completing this project.

-Vatsalya Sharan

Contents

1 Introduction 3

2 Data-Set 3
- 2.1 Training Data-set 3
- 2.2 Testing Data-Set 3
- 2.3 Data Extraction 3
- 2.4 SDSS 4
- 2.5 Features and Classes 4
- 2.6 Data Manipulations 4
  - 2.6.1 Data Scaling 5
  - 2.6.2 Data Encoding 5
  - 2.6.3 Data Splitting Into Test-Train 5

3 Data Visualization 6
- 3.1 Training Data 6
  - 3.1.1 SNS Plot 6
  - 3.1.2 2D-visualization 7
  - 3.1.3 3D-Visualization 7
- 3.2 Testing Data 9
  - 3.2.1 2D-Visualization 9
  - 3.2.2 3D Visualization 9
  - 3.2.3 SNS PLOT 9

4 Supervised Machine Learning Algorithms 10
- 4.1 Linear Regression 11
- 4.2 Logistic Regression 11
- 4.3 Support Vector Machine 13
- 4.4 Random Forest 15
- 4.5 Artificial Neural Network 15

5 Unsupervised Machine Learning Algorithms 17
- 5.1 K-Means Clustering 18

6 Comparative Advantages and Disadvantages of the various Methods 19

7 Bibliography 23

1 Introduction

Machine Learning and AI have brought in new paradigms in the world of data, and Physics and Astronomy have been no exception to them. This Project is one of the many example of how Machine Learning and AI are being used daily to solve a variety of new problems arising from the sheer magnitude and complexity of new data being generated.

2 Data-Set

Data-Set(s) which have been used here are taken from the SDSS (Slogan Digital Sky Survey). There are thousands of data-sets easily available on Internet for the purpose of machine learning, but SDSS provides individuals with access of vast amount of Astronomical data which can be used to approach multitudes of problems including classification and Regression Problems.
The Data-Set(s) used here are customized data-sets downloaded directly form SDSS website using SQL query.

2.1 Training Data-set

The training Data-Set is a $50,000$ row data-set with 6 features/columns in total which are labelled as u , g , r , i , z , $redshift$. There is a $'class'$ column attached with this which contains the labels. There are 3 classes namely that of 'STAR' , 'GALAXY' 'QUASAR'. The values of the features u and g in the training data-set range from 0 to 20 and rest of the features correspond to it.

2.2 Testing Data-Set

The Testing Data-set is again similar to the Training Data-set but has different feature values. The features u and g both range from 5 to 25 and the rest of the features corresponds to it. The idea behind introducing new data-set for Testing was basically to test the different model on the new data and check whether they are able to perform decent on the new data-set as well.

2.3 Data Extraction

In the code file/notebook, there are basically two methods for getting the data-set. One of them is using a predefined function 'get_data' from Astro-ML Library. Other one is directly extracting data from the SDSS website. The link ot the page is: `http://skyserver.sdss.org/dr16/en/tools/search/sql.aspx` and the SQL query is as follows:

```
- - This query does a table JOIN between the imaging (PhotoObj) and spectra
- - (SpecObj) tables and includes the necessary columns in the SELECT to upload
- - the results to the SAS (Science Archive Server) for FITS file retrieval.
SELECT TOP 50000
```

```
p.u,p.g,p.r,p.i,p.z,s.class, s.z as redshift,

    FROM PhotoObj AS p
JOIN SpecObj AS s ON s.bestobjid = p.objid
WHERE
p.u BETWEEN 0 AND 19.6
AND g BETWEEN 0 AND 20
```

The SQL Query for the testing Data-set is almost same but with the u and g values in the Query changed from 5 to 25.

2.4 SDSS

As mentioned Earlier, SDSS stands for Slogan Digital Sky Survey. It is a project to make a map of a large part of the universe. It contains data for a range of astrophysical phenomena including classification of galaxies, star clusters, planets, quasars, star formation etc.

2.5 Features and Classes

The features present in the data-set are basically logarithmic intensity of Light in various available spectrum. There are five rows of CCDs in the SDSS camera, labeled as u, g, r, i and z. These five filters together span the optical window. Each filter images a section of sky nearly, but not exactly, simultaneously (each filter is separated by 71.72 seconds). The filters always observe in the same time sequence: r, i, u, z and then g.
Apart from these features, there is also a feature named $redshift$ which is basically defined as the displacement of spectral lines towards longer wavelengths (the red end of the spectrum) in radiation from distant galaxies and celestial objects.It is interpreted as a Doppler shift that is proportional to the velocity of recession and thus to distance.
The Classes in the dataset are as follow:

- STAR-A massive sphere of gas undergoing nuclear fusion
- GALAXY- A massive structure containing Trillions of Stars and Gas.
- QUASAR- A massive and extremely remote celestial object, emitting exceptionally large amounts of energy, and typically having a star-like image in a telescope.

2.6 Data Manipulations

The data we downloaded directly from the SDSS website using SQL query was in the csv format and was converted into excel format for some prior manipulations which then had to be converted into pandas DataFrame for working in python.After this various manipulations were done in order to visualize the data better and to fit the data into various models used in this project.

2.6.1 Data Scaling

The training data was first converted into a pandas DataFrame. The features of the data-set were taken and then scaled using the max scaling method and the standard scalar scaling method available in sklearn library. This scaled features part of the data was rejoined with the 'class' part of the data to form the scaled DataFrame. The classes in this scaled DataFrame were mapped to integer values (from actual strings) because many methods require this kind of conversion of the labelled classes.
The classes 'STAR' , 'GALAXY' , 'QSO' are mapped to integers 0 , 1 , 2 respectively. Note that the original version of the non scaled data was also kept ready for usage in ANN models. Similar thing was done for the Testing Data-set.

2.6.2 Data Encoding

For some of the models, the target variable was required to be present in the one hot encoded form. For this purpose "*from sklearn.preprocessing import LabelEncoder*" was used. Encoding was done for both training and testing data-set. Since there were 3 classes, the encoded target class was a 3-vector.

2.6.3 Data Splitting Into Test-Train

Train Test Split from the sklearn was used for splitting the training data-set into training and testing. The test-train split used was 0.25 and random state = 42 . This was repeated for the encoded data-set with same parameters. This function basically takes the data-set, randomizes it and then divides the data into training and testing parts. The testing data-set was not splitted and was used only for testing purpose and not for any form of training.

3 Data Visualization

A lot of graphs were plotted to gain as much insight as possible into the data-set. Various ways of plotting and the inferences are listed below.

3.1 Training Data

3.1.1 SNS Plot

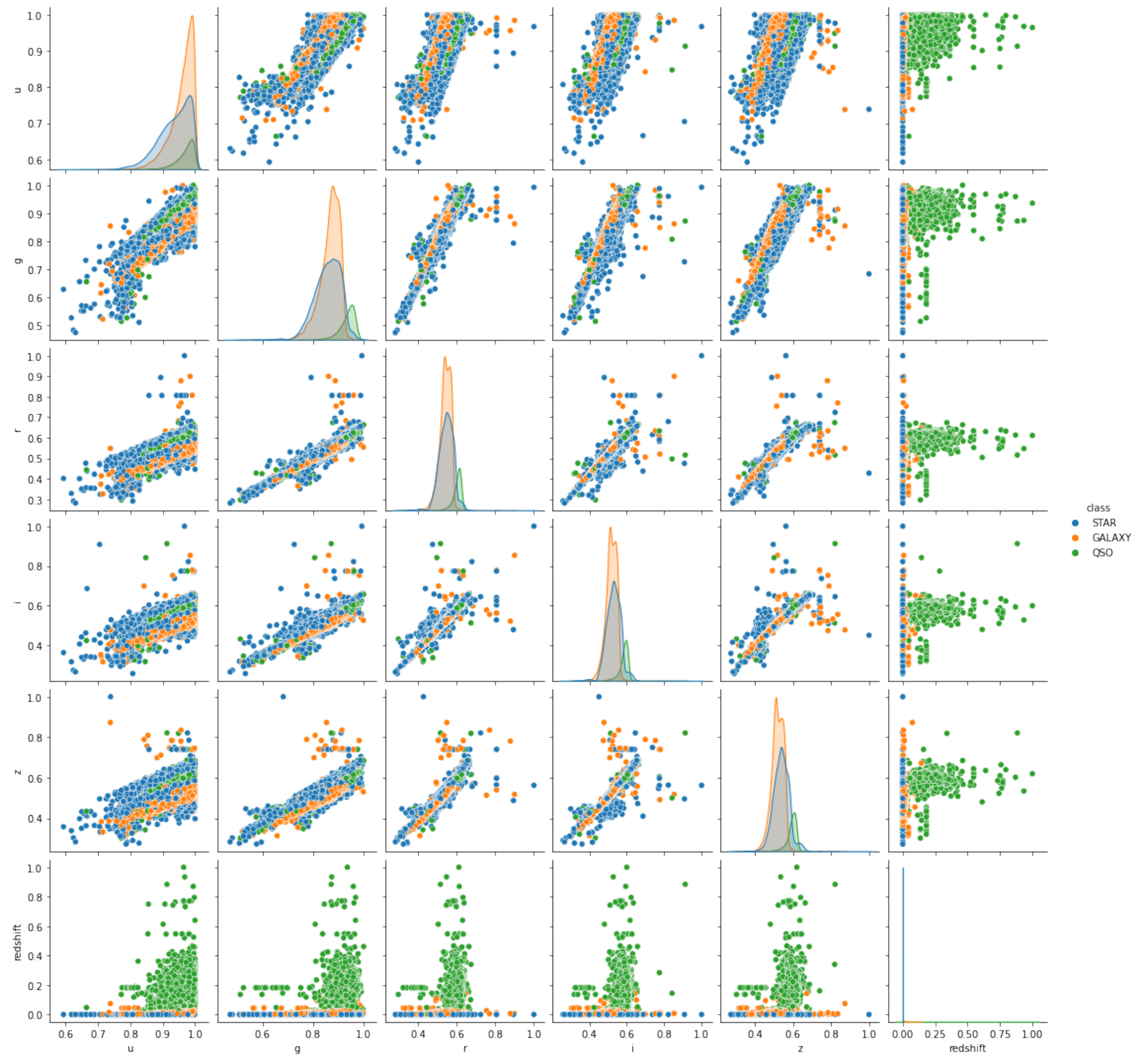
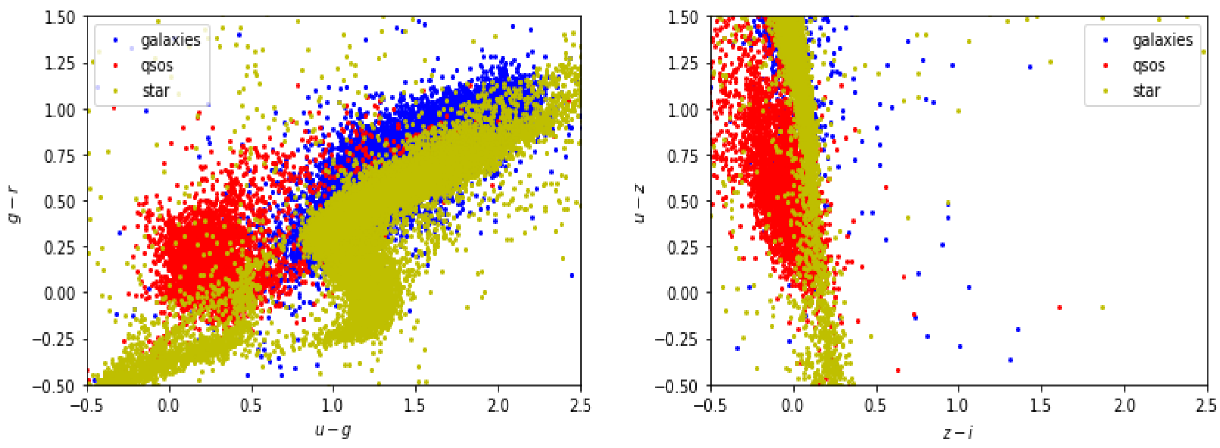


Figure 1: SNS plot of the training data

If we notice carefully in 'figure 1', the matrix of the graphs obtained is symmetric in nature. So just considering the elements below the diagonal, we can see that there is somewhat of a linear relationship between different features along with non-linearities, but the classes are clearly not separable and there is a lot of intermixing. The intermixing can also be seen on the diagonal elements with each of the classes overlapping with each other.



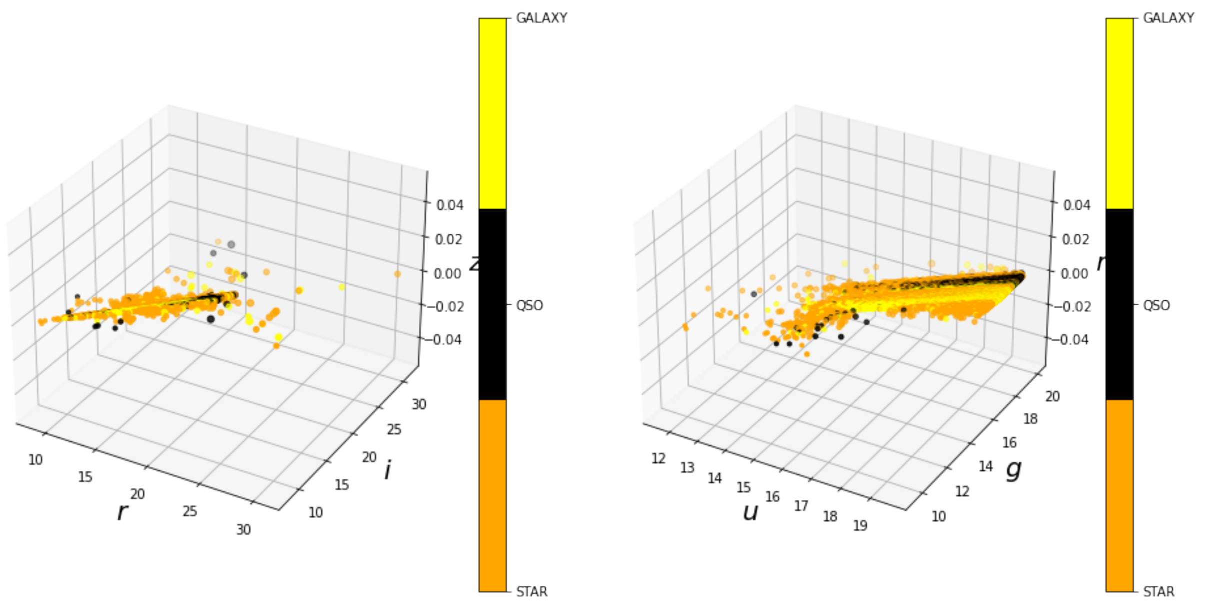
(a) g-r vs u-g (b) u-z vs z-i

Figure 2: 2D visualization of training dataset

3.1.2 2D-visualization

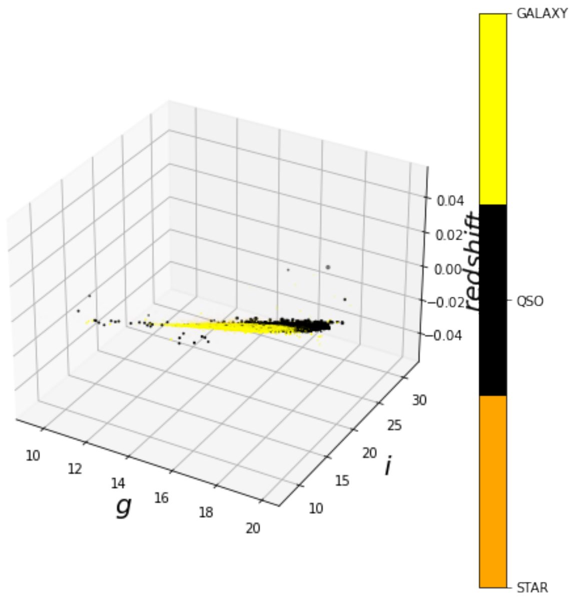
Here are some of the graphs (Figure 2) which try to show the nature of the data-set and the relationship between various classes. From the graphs, it is clear that that even though the classes have distinct regions, these regions overlap a lot and thus no clear separation is present here.

3.1.3 3D-Visualization

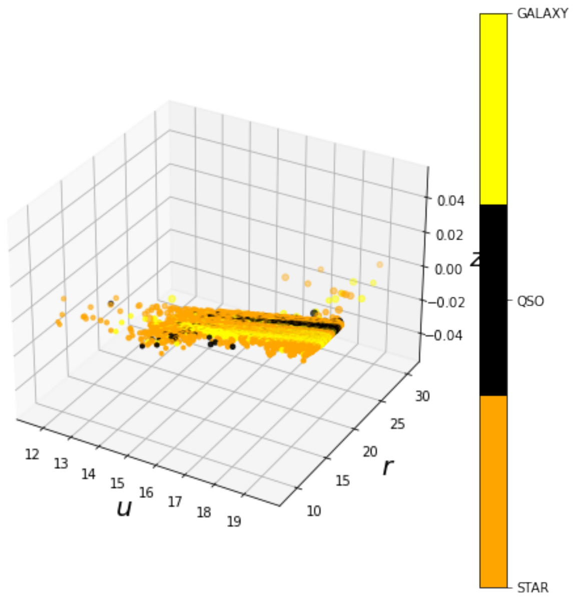


(a) r vs i vs z (b) u vs g vs r

Figure 3: 3D visualization of training data-set (1)



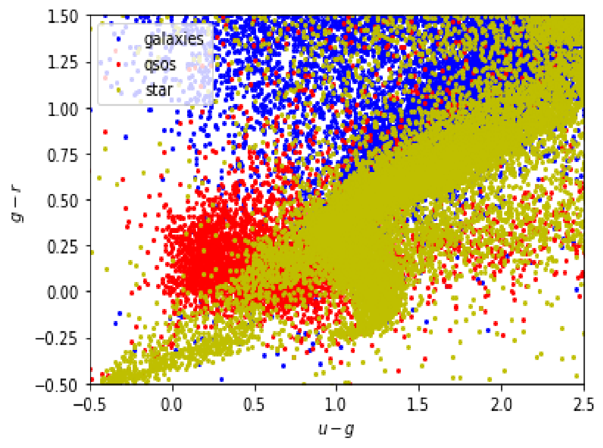
(a) g vs i vs redshift



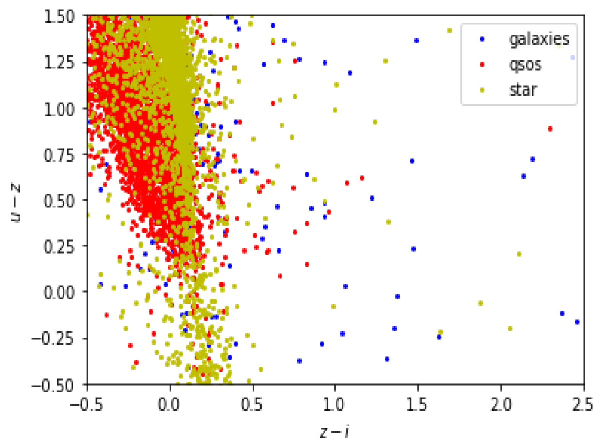
(b) u vs r vs z

Figure 4: 3D visualization of training data-set (2)

Here is an attempt (figure 3 and 4) to visualize the data-set in 3-D. Careful observation clearly implies the existence of somewhat linear relationships along with non-linearities between some of the features but it also shows that the classes are not linearly separable. The overlapping of colours indicate overlapping of classes.

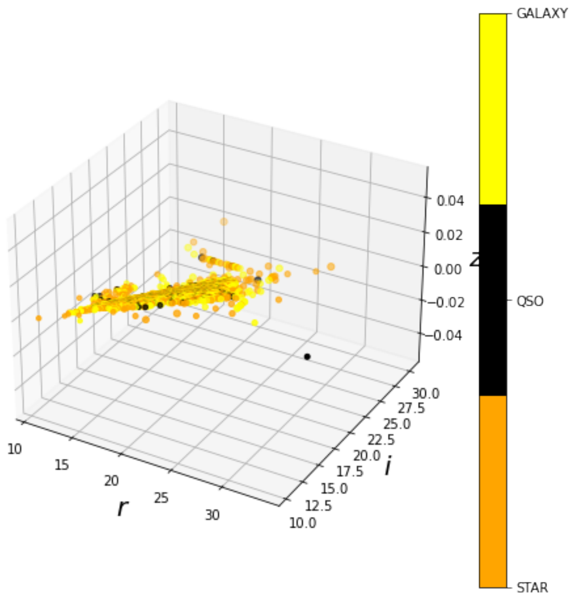


(a) g-r vs u-g

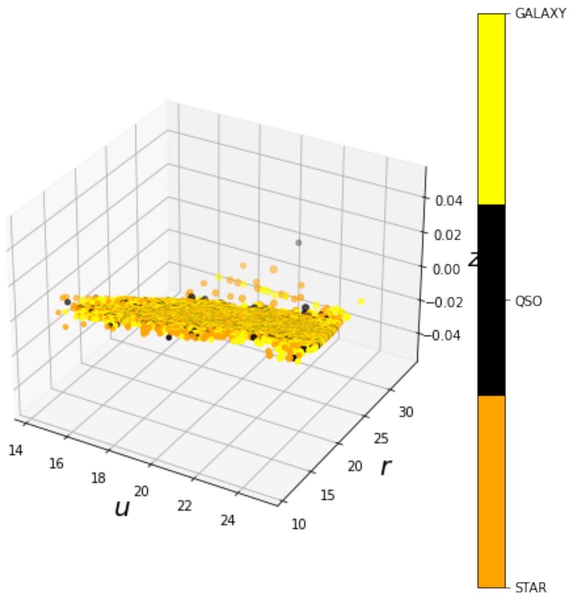


(b) u-z vs z-i

Figure 5: 2D visualization of testing dataset



(a) r vs i vs z



(b) u vs r vs z

Figure 6: 3D visualization of testing dataset

3.2 Testing Data

Below is a similar analysis for the testing data-set.

3.2.1 2D-Visualization

A careful analysis clearly (Figure 5) shows that the testing data-set is very different from the training which can be compared by looking at the same graphs under the training data-set section. One of the most evident features is the presence of large number of outliers as compared to the Training data-set.

3.2.2 3D Visualization

Including one of the 3-D plots for the sake of completeness. Figure 6 clearly depicts how the testing and the training data-set are different from each other. Testing data-set seems to have much more overlapping among the different classes.

3.2.3 SNS PLOT

SNS plot for the testing data (Figure 7) depicts the one vs one relation of all the featuers

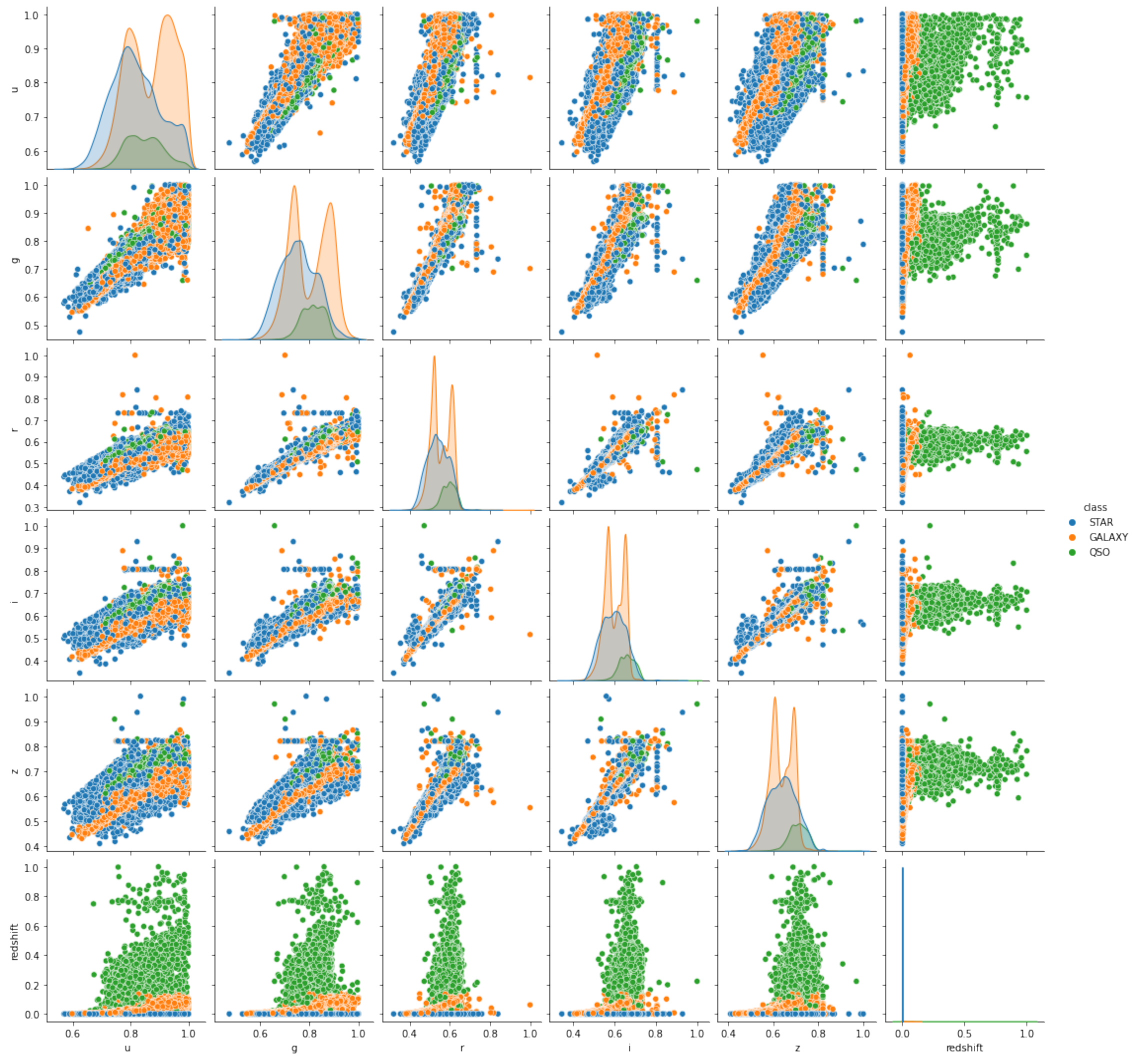


Figure 7: SNS plot of the training data

Machine Learning Methods

This section of the Report contains all the Methods which were used on the Training and Testing Data-set. Machine Learning Methods provides us with powerful tools to manage and classify large quantities of data. With the improvement in algorithms, better and more efficient models are being developed which are helping in ways not imagined before.

4 Supervised Machine Learning Algorithms

Supervised Machine Learning Process is a kind of Curve Fitting Process in which we don't have the explicit knowledge of the curve. In these set of Algorithms, models are fitted on a given data set, and the predictions are made. Different predictions are matched with the original values and a loss function is generated which is minimized using various methods over a number of iterations and then the models are tested over new data to make predictions.

4.1 Linear Regression

Linear Regression is a powerful tool in the research arena and is widely used for making sense of the data we have. But it is not used for classification problems, even more less when the data/labels are not continuous.

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i + \hat{\epsilon}_i \tag{1}$$

The way classification has been implemented here is by putting the feature data into the Linear Regression Model and getting an output based on that. Since the outputs were bound to be continuous, those values were rounded to nearest target data labels (0,1,2 in this case) and thus the predicted values were generated and this was used for testing the above model. Surprisingly, this ad-hoc method did achieve decent accuracy for the training data-set. It achieved an accuracy of 84.88% on the testing part of the training data set which is not a bad result. The loss function used here was 'Mean Squared Error'. There are few regularization techniques available to check for the case of over-fitting, which were implemented but didn't improve the result drastically.

- L1 Regularization or the Lasso Regularization gave an accuracy of 83.85%. Value of alpha used was 0.001
- L2 Regularization or the Ridge Regularization gave an accuracy of 85.32%

$$Ridge = RSS + \lambda \sum_{i=1}^{k} (\beta_i)^2 \tag{2}$$

The model failed miserably on the testing data-set with an abysmal accuracy of just 1.4%. This means that the coefficient approximated using the training data-set don't fit the testing data-set that well at least for the linear regression model.
The fact that it gives decent accuracy on the training data-set is maybe because it can be approximated to how a single neuron $wx + b$ works.

Alternatively, trying the linear regression model in a way that it takes all the features except 'redshift' as input and redshift as output (Figure 8), and then trying to compare the results, didn't turn out splendid as well, with a score of just 26.17 and rms value 12.3 on the testing part of the training data with max scaled training data. Using Standard scalar training set further reduces the accuracy.
The outcome of this model should come as no surprise because as we have seen in the data analysis section, data is quite complex and has some non-linearities, so a simple model like Linear Regression should not be able to capture all of that information.

4.2 Logistic Regression

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. It is a Linear Model of Classification and thus works well when the features have linear relationships. It uses a given logit function to make a decision boundary which divides the data-set into two classes. While the Linear Regression is used for Continuous data-sets, Logistic Regression is easily applied to the discreet data set. It uses methods

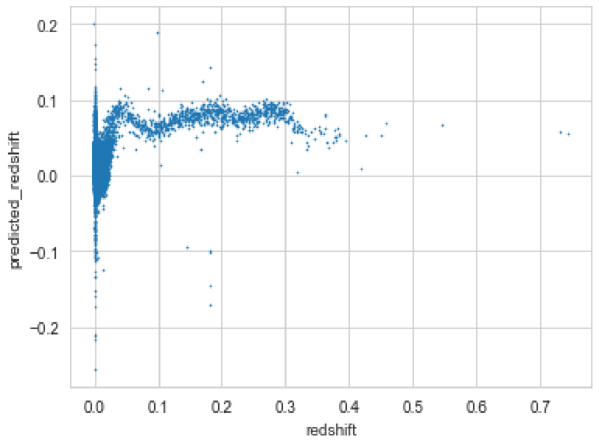


Figure 8: Lin Reg on redshift

like *onevsone* and *onevsall* for multiple class classification. It is also possible to use one hot encoding and directly predict the class of the given data point. Since the problem at hand is Multiclass Classification, the default value of 'multi_class' parameter is 'multinomial' and the loss function is 'categorical cross-entropy'.

For the Logistic Regression Model with default Parameters, the observed accuracy was 85.55%. Knowing the nature of the Data through the analysis in the previous section, greater accuracy can always be achieved using by tweaking the parameters.The Train-Test Split used for this model was normalized using max scaling method. The model took some time to give the output results. Some of the changes which were applied as follows:

- Changing the Norm/regularization from l2 to l1 immediately improved the results. This may point to the fact that the data had many outliers and the data is predominantly non Gaussian.
  (RSS = Residual Sum of Squares)

$$Lasso = RSS + \lambda \sum_{i=1}^{k} |\beta_i| \tag{3}$$

- Changed the Solver to SAGA. SAGA is an incremental gradient algorithms with fast linear convergence rates."SAGA supports non-strongly convex problems directly, and is adaptive to any inherent strong convexity of the problem. We give experimental results showing the effectiveness of our method."
- Increased the Max Iteration to 10000 so that the loss function reaches convergence.
- Changed the 'class_weight' parameter to *balanced*.
- Changing tolerance didn't have much impact apart from increasing time for decreasing values. random_value was set = 0.

All these changes resulted in a much better accuracy of 98.22%. To check if the model had the problems like over-fitting, Repeated K-Fold Cross Validation Method was implemented with the parameters like number of folds as 10 and number of repeats as 3. Implementing Cross Validation can help ensure that the model is performing well irrespective of the data-split and is not prone to over-fitting. The Cross Validation Accuracy obtained was 97.9% which shows model was not over-fitting.

Implementing this model on the testing part of the training data-set yielded an accuracy of 98.23%.
Upon Testing the model on the Testing Data-set, the accuracy obtained was 85.6% which goes on to show that even though the model is very simple and performs very well in the training data-set, it cannot predict good results with increasing number of outliers in the Testing Data-set.

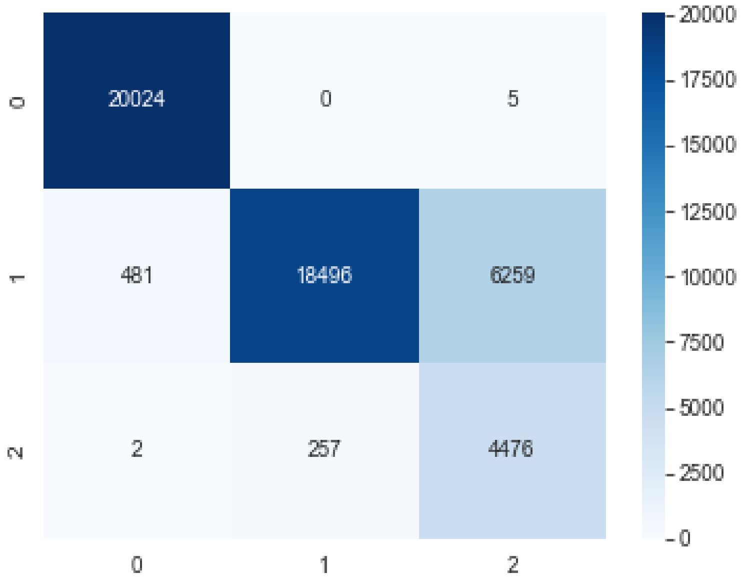


Figure 9: Logistic Regression Confusion Matrix on Testing data

The weighted f1 scores for all the classes were not same, which can also be seen from the confusion matrix. A lot of objects of the class 'GALAXY' were miss-classified as 'QUASAR' and vice versa. The f1 score was worst for the class QUASAR and this may be due the two above classes overlapping with each other and the model not being able to distinguish them, especially in the testing data.

4.3 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features in the data-set) with the value of each feature being the value of a particular coordinate. Classification is done by finding the hyper-plane that differentiates the two classes very well. The algorithm maximizes the distance from the set of nearest points of a particular class from this hyper-plane.

SVM can be used both as a linear or non linear model for classification by using different kinds of Kernels available. For the SVM model with default Parameters, the observed accuracy was 94.92%. The Default value for kernel is rbf (radial basis function) which allows for small non linearity in the data-set. Default C value is 1 and class_weight set to *none*. The loss function depends on the kernel used and the optimizer is based on Sequential minimal optimization and Quadratic Programming. The default function shape was 'ovr' meaning $one - vs - rest$ implementation. Train-Test Split used was normalized (max-scaling).

The Complete Grid Search over all the possible parameters didn't execute completely on both Google Colab and Jupyter Notebook Environment. But the Grid search method without the Gamma values did provide with fine tuned parameters and the model built using them performed extremely well on both the training and testing data. The hyper-tuned parameters are discussed below.

Value of C=500, $kernel = polynomial$, and $class\_weight = balanced$. The Classification Method was again $one - vs - rest$. Applying Different values of Gamma(with rbf kernal) reduced the model accuracy heavily. The above optimizations were obtained from Grid search and were good enough for the model to achieve and accuracy of 99.16% on the training data.

K fold Cross Validation was also performed with number of folds = 10 and number of repeats= 3, to check the possibilities of over-fitting, but the Cross validation scores were identical.
Testing this model on the testing data yielded an accuracy of 96.32% which was the best among the non-artificial Neural Network ML methods.

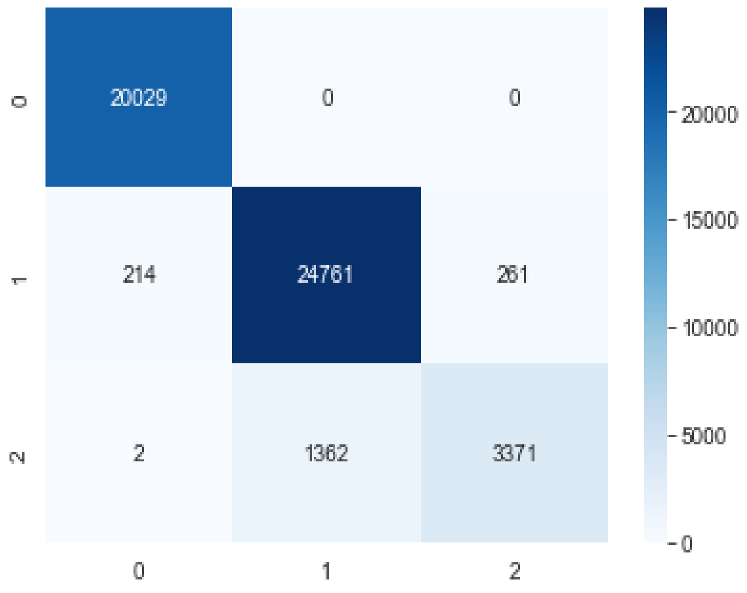


Figure 10: SVM Confusion Matrix on Testing data

The F1 score for the class 2 (QUASAR) was the least among all the classes at 81%. QUASAR objects were misclassified with the objects of class GALAXY. One-vs-rest method of classification also meant the model will have difficulty in distinguishing between the classes

’QUASAR’ and ’GALAXY’ because these classes are related to the features in a similar manner. But the F1 scores on the Other classes (0 and 1) were very high (98%).

4.4 Random Forest

Random Forest is a supervised learning algorithm. The ’Forest’ here, is basically an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models improves the overall result. It is one of the most used algorithms, because of its simplicity and diversity, as in, it can be used for both classification and regression tasks.

Coming to the model used in the analysis, the default model with ’n=100’ estimator and ’random_state=42’ was implemented and this turned out to be a great fit for the training data-set with a staggering accuracy of 99.208% on the testing part of the training data.Reducing the number of estimators slightly decreased the accuracy of the model.
The train-test split data used for this method was max-scaled with train-test-split of 0.25 and a random_state =42.

Slightly tweaking the default parameters like changing the ’criterion’ parameter, which is used for the final classification , from ’*gini*’ (default value) to ’*entropy*’ only reduced the overall accuracy. Changing other parameters resulted in lowering of accuracy on both the training and testing data.
Cross validation was applied to the above model to ensure that the model is not over-fitting. With a repeat value of 3 and K-fold=10, the accuracy obtained from this was around 99% Which is the same accuracy obtained on the normal train-test split of training data. Random Forest Method Fitted the training data very well with minimal hyper-parameter tuning.
This stark difference between the training and testing results is not very easy to infer. One of the reason why this happened could be because random forest is inherently non-linear, and we saw in the data-visualization section that some of the feature sets were related linearly with each other. Also Random Forest is very susceptible to the changes in data. Add to this the increased number of outliers and thus we get why model doesn’t perform that well on testing data.
The F1 scores on the training dataset were in line with the model accuracy. With the testing data-set a lot of class 1(GALAXY) objects were mis-classified as class 2(QUASAR) objects. The model failed on all classes expect class 0 objects (STAR) while being evaluated on testing data-set.

4.5 Artificial Neural Network

Artificial Neural Networks are a special type of machine learning algorithms whose units are modeled after the structure of neurons present inside human brain. ANNs are nonlinear statistical models which can show a complex relationship between the inputs and outputs to discover a new pattern.
For the ANN analysis, data-encoding was required and thus it was done as mentioned in the ’Data Encoding’ sub-section. Both Standard Scaling and Max scaling Methods were used. The Normalized Training data gave very good results on the training data and the testing

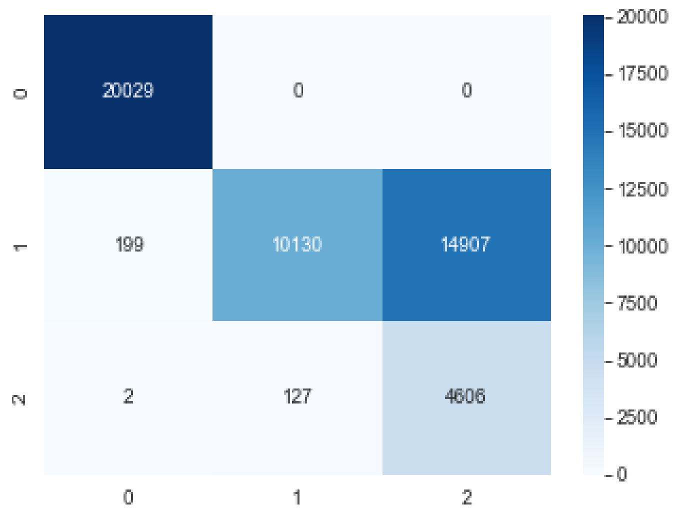


Figure 11: Random Forest Confusion Matrix on Testing data

split of the training data but failed to show similar results on the testing data. But the data without normalization gave comparable results on the training data and much better results on the testing data. The general model structure and optimization process followed is listed below:

- The 1st layer starting from 256 nodes, *relu* activation function and input_dim=6. This was varied with the 1st layer with only 128 nodes, but the results on the training data were comparable.
- The subsequent layers had 64 and 32 nodes with *relu* activation and the last output layer had 3 nodes with *softmax* activation. The overall structure of the model was funnel shaped.
- A plot_training_training_history function was defined, which plots the model performance w.r.t training and validation data.It also gives the model accuracy on the testing data.
- Both the data-set using standard scalar method and max scaled method were used and they both performed well
- With the set of loss functions namely- *categorical_crossentropy* , *categorical_hinge* , *kullback_leibler_divergence* , *mean_squared_error* and the set optimizers namely- *sgd*, *adam*, *nadam*, *adagrad*, *rmsprop* , two loops were run with the model being fitted with additional parameters like batch_size=128 and early stopping callbacks. The testing part of the train data was made into the validation_data.
- While most combinations gave good result with standard scalar and max scaled values, categorical_crossentropy showed best results followed by categorical_hinge.

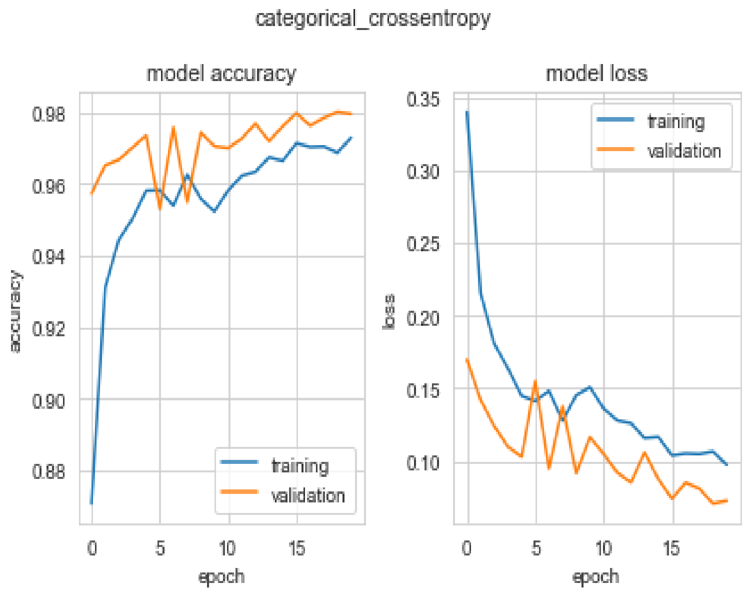


Figure 12: adam with categorical cross-entropy in the plot training history

- On the non scaled training data, it was mainly the categorical_crossentropy loss function with optimizers 'adam' , 'nadam' , 'adagrad' , 'rmsprop' showing good results.Since the data had multiple classes, this outcome was expected.
- Using the Model Check Point Call back with monitor parameter equaling val_accuracy and saving only best weights and early stopping monitoring val loss and patience = 10, models with different parameters were tested. These models were compiled with categorical_crossentropy and all the optimizers.
- With above settings and using Dropout = 0.2 after the 1st 256 neuron layer and batch_size =256 or 128, gave an accuracy of around greater than 95% on the testing data on multiple trials (5 runs) which is the highest among all the models.(This changes with runs, but ANN models further have lots of scope to improvise)

The f1 scores obtained using the model optimizations above for the all the classes was around 90%. This is the best result obtained among all the methods used for the classification. The encoding changes the integer labels of the different class but still its easy to infer which numbers belongs to which class in the confusion matrix in Fig 13 (0: GALAXY; 1:QUASAR ; 2: STAR).

5 Unsupervised Machine Learning Algorithms

Unsupervised Learning is a machine learning technique in which the users do not need to supervise the model. Instead, it allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with the unlabelled data. It can be in principle used for classification of the data points as well.
The Data-set we have, both training and testing, are labelled and thus are conducive for Supervised Machine Learning Algorithms and Classification Problems.

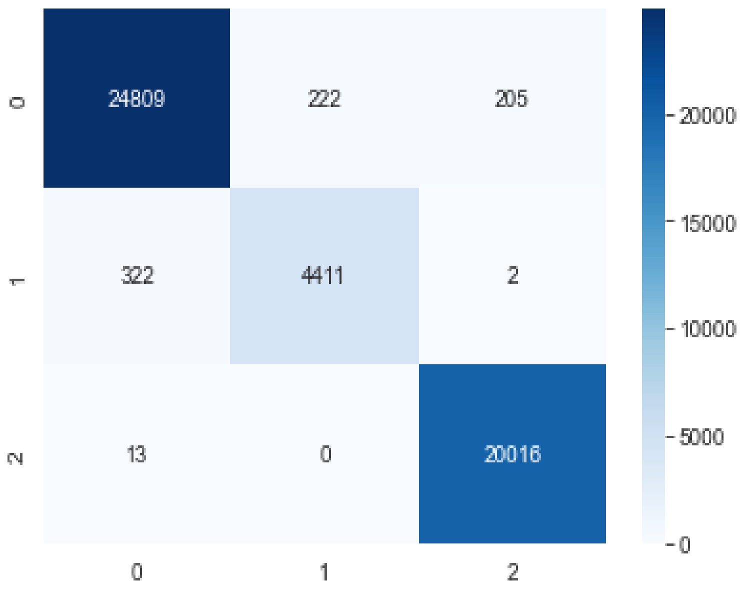


Figure 13: Confusion Matrix for ANN Model

5.1 K-Means Clustering

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process. The algorithm aims to minimize the squared Euclidean distances between the observation and the centroid of cluster to which it belongs.
Since the data-set was a labelled, it was not so straightforward to apply Unsupervised Machine Algorithms like K-Means Clustering on the given Data-set.
Mini Batch K-Means which has been used here, works similarly to the K-Means algorithm, difference being that in mini-batch k-means the most computationally costly step is conducted on only a random sample of observations as opposed to all observations.This approach significantly reduce the time required for the algorithm to find convergence with only a small cost in quality.
For the application, naturally the 'class' column was removed and the model was initially trained with only '3' clusters and the data was labelled into 3 different clusters by the algorithm itself. Now these 3 clusters may or may not be the actual clusters of the labelled data. So these clusters are assigned labels according to their closeness to a given class using a function. Each object was thus classified into different clusters by the algorithm and was now given a label with respect to the above function and then this output when compared with the test part of the training data gives and accuracy of 53.4%.
To optimize the model for better classification, use of metrics like Inertia, Homogeneity score and Accuracy score was implemented. Inertia is a measure of how internally coherent clusters are and is inversely proportional to sum of squares of distances between data points and their respective cluster's centroid. It decreases as the number of clusters increases.
The data points at borders of clusters can belong to more than one cluster with some probability or likelihood value. Homogeneity is a measure of data points of a particular cluster belonging to a single class.

Accuracy score is the percentage of correctly predicted values as compared to the testing part of the training data.

With the above metrics, a loop was run with increasing number of clusters. As the number of clusters increased, accuracy , homogeneity increased and Intertia decreased till the number of clusters was 512. Now the best model was used for testing on the testing data which gave an accuracy of about 92.07%. The f1 scores obtained for all the classes were similar (92%).

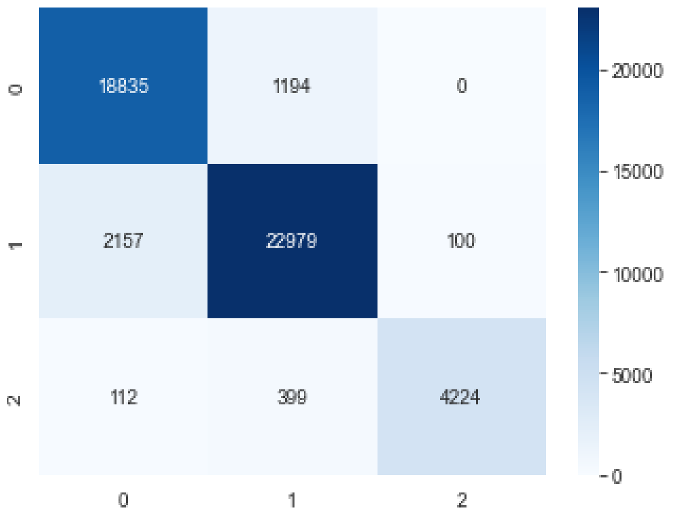


Figure 14: Confusion Matrix for K Means Model

6 Comparative Advantages and Disadvantages of the various Methods

This section compares the advantages and disadvantages of the different methods used for the analysis in the project.
Both the data-sets displayed some form of linear relationships between the various features and the testing data containing many outliers or new data points. Most of the models fitted very well on the training data-set with less amount of overfitting. Models which captured the general nature of both the data-sets performed relatively well while others did not.
In problems involving Astronomical Classification, usually the most important class is not present in large quantities and this data-set was also an example of this. The most important class which needed to be classified was QUASARS and only few models tackled this task decently.

- **Linear Regression**-This method is mostly used for finding out the relationship between variables and target and is generally easy to implement, both with codes and computationally. The model fails for our data-set because the relationship between the features is not exactly linear, which is a requirement for this model to work.The loss function looks at the mean of the dependent and independent variables which is not the complete description of the data in most of the cases. The model fails at even the regression task as discussed earlier.

- **Logistic Regression**-Although Logistic Regression introduces non-linearity in the form of sigmoid function,it still constructs linear decision boundaries which is not conducive for the data used. This model is easy to implement and efficient to train and makes no assumption about the distribution of classes in the feature space and has scope for regularization to prevent over-fitting. It has provision for multi-class classification using multi-class parameter and one hot encoding.
  But a major flaw in the method is that it assumes independence among the variables and needs independent variables to be linearly related to the log odds $log(p/(1-p))$. The logit function is also very susceptible to the outliers in the data and they change the shape of the sigmoid and hence shift the decision boundary.The decision boundary predicted by Logistic is also linear in nature, which doesn't fit with our testing data well which has some overlapping and some nonlinear distribution of classes in the feature space.
  Hence this model performed poorly on the testing data.

- **SVM**-This method with the advantages of different available kernels for different situations (for example tackling the non linearities with polynomial or rbf kernal) and loss function optimization defined by a convex optimization problem proved to be efficient in the classification of the testing and training data-sets. The introduction of slack variables and the ability to relax the margin constraints using a tuning parameter allows for greater flexibility in defining the model according to our needs. It is also one of the computationally fastest methods available and handles outliers much better than logistic regression, even though it gives up upon the actual probability estimates. Changing the data by small amounts do not change the hyper plane of separation by a huge amount, and thus this model is less prone to over-fitting and performs well on the testing data.
  Coming to some of the drawbacks, SVM doesn't perform well when the data has more noise or when the target classes overlap, which can be seen with the low f1 score for the objects of the class 'QUASAR'. Even though there are a lot Kernels and other tuning options available, its a difficult and time consuming process to choose the right parameters for the problem. SVM is also inherently built for binary class classification and uses 'ovo' or 'ovr' methods for multi-class which can give error at times. The data generally requires feature scaling and the models are difficult to interpret as opposed to simple models like Logistic Regression.

- **Random Forest**-The Forest, constructed out of the Decision Trees with the added layer of complexity of bagging to improve the overall results and to reduce the over-fitting problems of the Decision Tree method, finds its usage in many of the ML problems and even works for the training data-set used in the project. It gives great f1 scores for all the classes on the training data. Usually this method is easy to implement and works well with minimal hyper parameter tuning as was the case with training data.
  The demerits lies in the fact that small changes in the data can considerably effect the results, which was the case with testing data. The model failed to classify even the objects of the class 'GALAXY', where other models performed relatively better. Also the

model is inherently non-linear and may fail to capture some of the linear relationships present between the features.
Since Random Forest gives up on the probability aspects of the classification, the objects of different classes with lie close to each other or even overlap with each other in the feature space are prone to get misclassified into different classes which was exactly the case with the objects of the classes GALAXY and QUASAR.

- **ANN**-ANNs, inspired from human neurons sure do a lot of 'magical work' in machine learning. The ANN algorithms are computationally very heavy and require capable hardware to perform calculations. The Algorithms take ample time to compute and fine tuning for the models and other parameters is an arduous task to say the least and similar thing was observed for this project as well. The models and the results are very hard to interpret and mostly its just trial and error with some educated guesses that solve the problem at hand. The problem also needs to be in a language suitable for the ANN algorithm to understand, for example, the softmax one hot encoding which was done prior. The results change with iterations and on different systems, so that also needs to be taken into account.
  Now being done with the complaints, the power of ANNs is for everyone to see. It can solve for any problem, with arbitrary complex non linear and linear decision boundaries.It doesn't impose any kind of restriction on the input variables like their distribution, etc. ANN models are quiet robust to outliers and noise in the training data and was thus able to predict all 3 classes with excellent accuracy on both the training and testing data.
  ANN models have wide variety of features for different scenarios(Different optimizers and loss functions) and even though its a difficult job to find best parameters, the excellent results obtained makeup for the effort. ANN models handle the heteroskedasticity of the data well and generalize the outcomes so that new data can be predicted accurately, and thus ANN turned out to be the best model in the analysis.

- **K-Means Clustering**-This Unsupervised Machine Learning Method is very easy to implement (just requires calculation of Euclidean distances) and scales well with larger data. Although, this method is not meant for classification, certain modifications which are allowed, like initializing the centroids, assigning right lables etc., helped achieve decent accuracy which was observed in the project with model giving 92% f1 scores for all the classes. This method also guarantees convergence but may not always converge to global minima. The results obtained are easy to interpret using various plotting tools available.
  Although there are some advantages for using this model, the biggest disadvantage is that its not meant for handling classification problems. Increase in dimensionality can affect the predictions considerably. Its also not the easiest to predict what K values will lead to best results. Model can get distorted by the presence of outleirs in the data because the centroids can get dragged out of their actual positions.

From the above analysis and taking into account the nature of the data-sets, the performance of the models, its easy to conclude that ANN and SVM are the preferred models

for classification on the given data. SVM is preferred when only general classification is needed in quick time with easy implimentation.
ANN is preferred when specificity of the class predictions becomes important and high accuracy is desired. If computation barriers are not an issue, then ANN seems the best choice overall for the given data-sets.

7 Bibliography

- Dr.Kushal Shah's Youtube Channel 'Evolutionary Intelligence' https://www.youtube.com/c/EvolutionaryIntelligence/featured
- http://www.astroml.org/
- https://stats.stackexchange.com/questions/430341/linear-regression-for-multi-class-classification
- https://arxiv.org/abs/1407.0202
- https://www.sdss.org/dr12/imaging/imaging_basics/
- https://www.nbshare.io/notebook/819279082/Regularization-Techniques-in-Linear-Regression-With-Python/
- https://medium.com/@joel_34096/k-means-clustering-for-image-classification-a648f28bdc47
- https://machinelearningmastery.com/how-to-stop-training-deep-neural-networks-at-the-right-time-using-early-stopping/
- https://medium.com/@joel_34096/k-means-clustering-for-image-classification-a648f28bdc47
- https://medium.com/analytics-vidhya/l1-vs-l2-regularization-which-is-better-d01068e6658c
- https://towardsdatascience.com/confusion-matrix-for-your-multi-class-machine-learning-model-ff9aa3bf7826
- https://www.datacamp.com/community/tutorials/random-forests-classifier-python
- https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6
- https://machinelearningmastery.com/multinomial-logistic-regression-with-python/
- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- https://towardsdatascience.com/cross-validation-in-machine-learning-72924a69872f
- https://machinelearningmastery.com/repeated-k-fold-cross-validation-with-python/
- https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

- https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/
- https://nhorton.people.amherst.edu/ips9/IPS_09_Ch14.pdf
- https://www.asquero.com/article/advantages-and-disadvantages-of-artificial-neural-networks/
- https://developers.google.com/machine-learning/clustering/algorithm/advantages-disadvantages