STAT 679 Final Project Report

Xiaoyang Wang Ziang Zeng

1 Astronomical Challange

Our project focuses on classifying celestial objects into stars, galaxies or quasars using their spectral characteristics. With the advancement of astronomical technology, we can obtain a large amount of data, including images and spectral information, from telescopes and large-scale photometry.

The central question of our project is: "How can we effectively use image and spectral data to accurately classify different types of stellar objects?" There are many machine learning methods and statistical models can be applied to classify the celestial objects. However, different methods and models performs differently on same data, "All models are wrong but some are useful." Can we find a more "useful" model through combining several models together? Our solution is voting classifier.

2 Data

We plan to work with the astronomy data set containing three types of data: images of the celestial objects, images of the spectrum, and the metadata of the objects. Figure 1 displays images of a galaxy, a star, and a quasar, from left to right, respectively. Figure 2 displays images of the spectrum of a galaxy, a star, and a quasar, from left to right, respectively. Table 1 provides explanations of the variables within the metadata.

Figure 1: Image of Celestial Objects

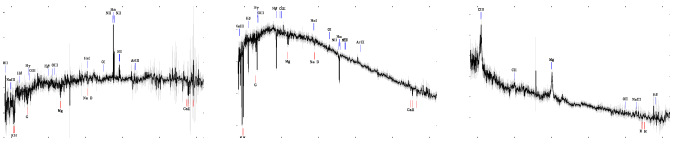


Figure 2: Image of Spectrum

They can be found in this link. All the data is obtained from https://www.sdss.org. Moreover, the distribution of classes of the celestial objects is 33333 samples each, which is selected from the website to make sure the dataset is balanced.

Table 1: Metadata of the celestial objects

Variables	Explanations
objid	Object Identifier
ra	Right Ascension angle (at J2000 epoch)
dec	Declination angle (at J2000 epoch)
u	Ultraviolet filter
g	Green filter
r	Red filter
i	Near Infrared filter
z	Infrared filter
run	Run Number
rerun	Rerun Number
camcol	Camera column
field	Field number
specobjid	Unique ID used for optical spectroscopic objects
class	Object class
redshift	Redshift value based on the increase in wavelength
plate	Plate
mjd	Modified Julian Date
fiberid	fiber ID
plateid	Plate ID

3 Exploratory Data Analysis

First, we will conduct exploratory data analysis on our dataset to better understand it and to find any possible errors. Figure 3 shows the distribution of variables that are meaningful for classification. We can see that the means of all variables across different classes are quite different. Moreover, there are no significant outliers.

Then, Figure 4 gives the correlationship between variables in the metadata. We can see that **redshift** has strong correlationship with the class of the object. Also, the filter variables (**u**, **g**, **r**, **i**, **z**) are correlated with each other.

Next, we will check and deal with missing values in the dataset. For images of the celestial objects, there is no missing value. For images of the spectrum, we have 14115 images that are unreadable. Considering that they are hard to impute, we just ignore them and conduct the analysis on the spectrum based on the rest of the images. For the metadata, there are 1 missing value for **i** and 3 for **z**. We use regression imputation with filter variables to impute the missing values as they are correlated with each other and quite scattered.

4 Methods

For our three types of data, we intend to use three seperate methods to build three different classification models. Then, we plan to use a voting classifier to combine three models and give our final model.

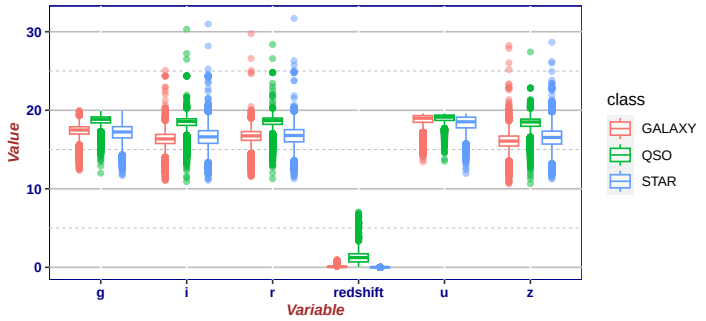


Figure 3: Boxplot

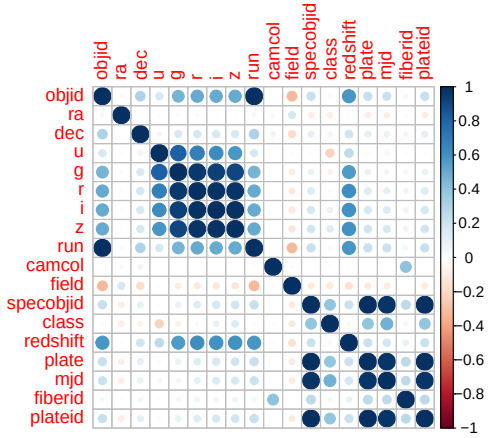


Figure 4: Correlation

In this section, we will give brief introduction to the methods that we have used.

4.1 kNN

The k-Nearest Neighbors (kNN) algorithm is a simple, but powerful machine learning technique that can be used for classification. At its core, kNN makes predictions about the classification of a data point based on the majority vote or average of its k nearest neighbors. With cross validation, we eventually selected $k = 3$.

4.2 Decision Tree

A Decision Tree is a machine learning algorithm that can be used for classification. It models decisions and their possible consequences as a tree-like structure, making it intuitive and easy to visualize. The decision-making process starts at the root node and splits the data on the feature that results in the most significant information gain (IG) or the greatest reduction in impurity (such as Gini impurity or entropy). The process continues recursively, creating decision nodes and leaf nodes. Decision nodes ask a question and branch based on the answers to those questions, leading to further splits or to leaf nodes. These nodes represent the outcome. With cross validation and consideration on the complexity of the tree, we set the maximum depth as 4, and use Gini impurity to prune the tree.

4.3 Logistic Regression

Multinomial logistic regression, extends the traditional logistic regression model to handle cases where the target variable categories are more than two. Unlike binary logistic regression, which uses one binary predictor per class, multi-class logistic regression models the probabilities of the multiple classes using a softmax function, which generalizes the logistic function for multi-class problems:

$$P(Y_i = k) = \frac{e^{\beta_k \cdot X_i}}{\sum_{j=1}^{3} e^{\beta_j \cdot X_i}}, i = 0, 1, 2.$$

Coefficient Estimation: The model estimates coefficients (or weights) associated with each feature in order to predict the log-odds of the outcomes for each class, except one. This excluded class serves as a "reference" or "baseline."

4.4 CNN

Before we get into complex Neural Networks, we firstly try to use a simple CNN to test the performance of NN in this problem. This CNN is quit shallow (337k parameters) with two convolutional layers and a maxpooling in between and followed by three fully connected layers. The competitor is VGG16 which is a much deeper NN with 13 convoluntional layers and 3 fully connected layers (138million parameters).

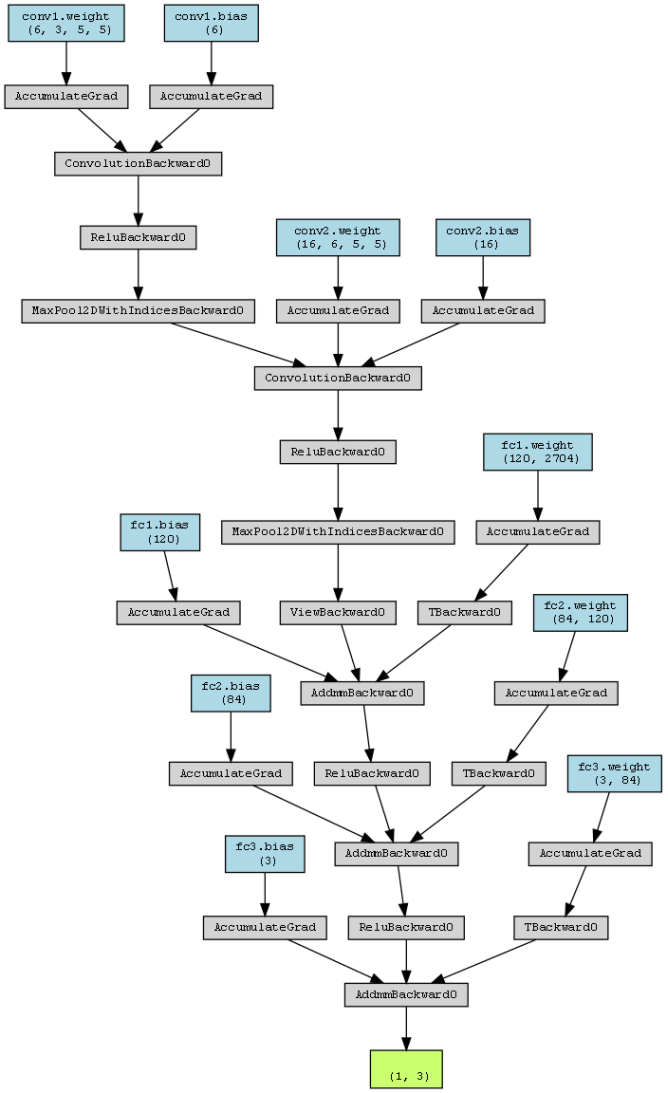


Figure 4: Structure of CNN

4.5 Voting Classifier

We tried to use two types of voting methods, soft voting and weighted hard voting. Their strategy are slightly different but the idea is similar: we want to combine the output of several model together to generate a more accurate one. Suppose we have models $\{C_1, \cdots C_n\}$, for a give input x, each model can have a prediction:$y^i_{pred}|x = (y_1, y_2, y_3)$ where one of y_j is 1 indicating the predicted class is j while others are 0. Or a predicting probability: $P_i(y_j|x)$ which is the predicting probability for x belong to class j for i_{th} model. For KNN and Tree model, their prediction and predicting probability are the same which means their predicting probability is one for predicting class. For soft voting, the prediction is made by average predicting probability of all candidate models and prediction of the voting model is the class with largest predicting probability. The probabilities for voting classifier given input x is $P(y_j|x) = \frac{1}{m}\sum_{i=1}^{m} P_i(y_j|x)$ so the prediction is $p(x) = \arg\max_{y_j} P(y_j|x)$ where m is the number of models. For weighted hard voting, the prediction is made by sum of weighted voting for each class for all the candidate models and prediction of the voting model is the class with largest number of weighted voting. For a given inputs, C_i has a predict $y^i|x : y^i_{k=j} = 1, y^i_{k\neq j} = 0$ and the prediction for weighted hard voting is $y_{pred} = \sum_i w_i \cdot y^i|x$, so the predict class is $argmax_j\ y_{pred}$.

5 Results

5.1 Metadata

Regarding statistical models, we intend to use either classification trees, or KNN models, incorporating ensemble learning methods such as bagging or boosting to improve performance. Our ultimate goal is to develop a hybrid model that combines image classifiers with meta-classifiers, aiming for superior accuracy.

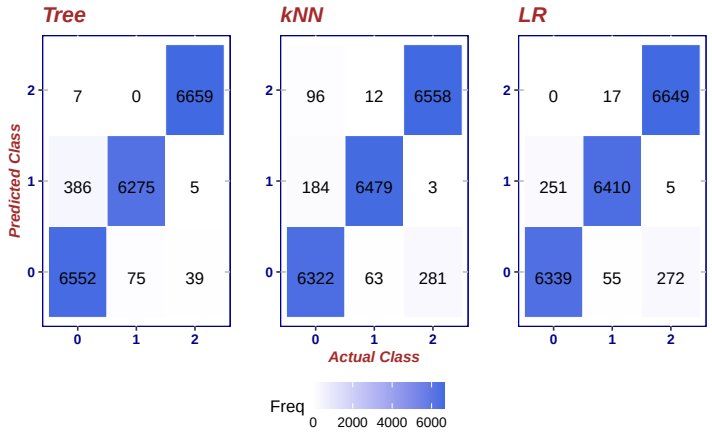


Figure 5: Confusion Matrices of Metadata Models

From the confusion matrix and statistics, we see a success in building model through the numerical data. By using several index and the red shift, our simple models reach more than 96% accuracy on validation data.

5.2 Image of the celestial objects

The results of this simple CNN is quite good. The cross entropy loss drop quickly after 10k iterations and get stable around 0.27. And the robustness can also be seen on its 90.2%(SGD)91.8%(Adam) average accuracy after 10 epochs training on validation data. Which actually discourages us to apply deeper NN which is quite time and energy consuming. The training took 20 min and the cross entropy loss is under 0.2 after 3 epochs training and stable at 0.18. On the validation data set, VGG has 94.23% which is better than our simple CNN. However this high accuracy not only consume lot of time but also make it more difficult to improve the performance through the combination of models. With these facts, we plan to modify the simple CNN a little without changing its main structure, and use this simple CNN to build up a final model which is expected to have similar performance as VGG16 while consume less time.

The accuracy of CNN of images 91.73%, CNN of spectrum is 88.91%

Table 2: CNN comparison

Name	Accuracy	train_time
SimpleCNN	91.68%	635
VGG16	94.11%	1281
Res18	94.89%	1324

Table 3: CNN tuning comparison

Optimizer	Accuray	train_time
SGD_mom0.9	91.68%	458
SGD_mom0.95	92.84%	463
SGD_mom0.99	93.78%	462
Adam	93.91%	463

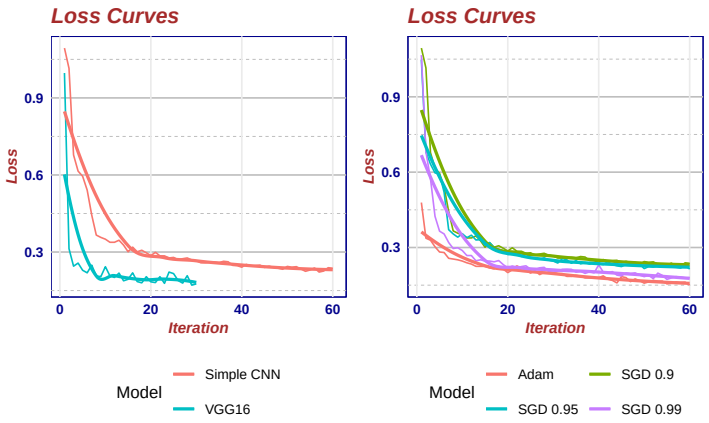


Figure 6: Loss Curve Plot for Different CNN

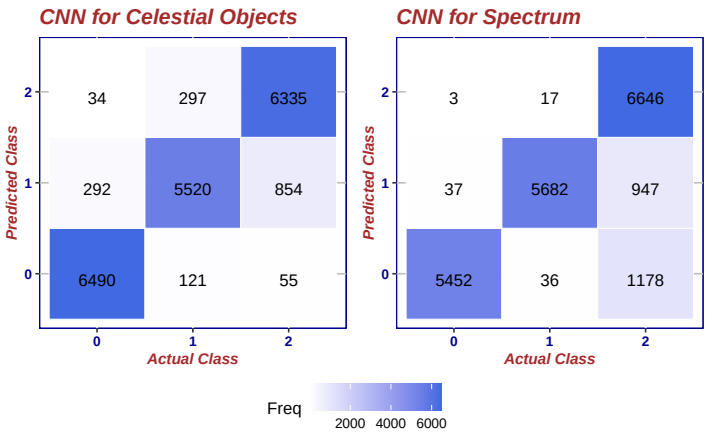
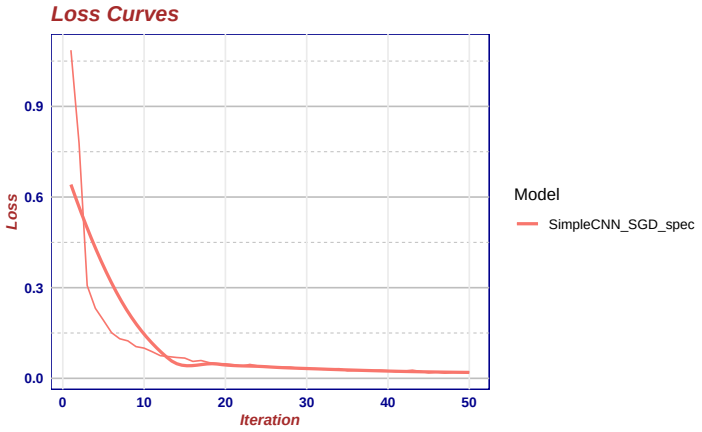


Figure 7: Confusion Matrices of CNN Models

5.3 Image of the spectra

The performance of SimpleCNN on spectrum image classification is also very good. We are able to get 0.021 cross entropy loss which is 99.14% accuracy on validation data set without missing (unreadable) images after 1467 seconds training. And if we include missing value for which the CNN randomly guess a class, the accuracy is 88.91%.



5.4 Voting Classifier

If we use voting strategy for only metadata model or CNN, we see a small improve of accuracy on both of them. For metadata models. the soft voting has 97.75%, weighted hard voting has 97.55% on validation data set which is better than KNN (96.80%), Tree model (97.44%) and Logistic Regression (97.00%). For CNN models, soft voting: 97.71%, weighted hard voting:91.55% are also better than CNN celestial: 91.73% and CNN spectrum 88.91%. The accuracy of voting classifiers are higher than any single model if we combine metadata models and CNN models together: Soft Voting:98.97%, Weighted Hard Voting: 98.63%. From the confusion matrix we can see that both of them classify star (class 2) perfectly followed by Galaxy (class 0) and Quasar (class 1).

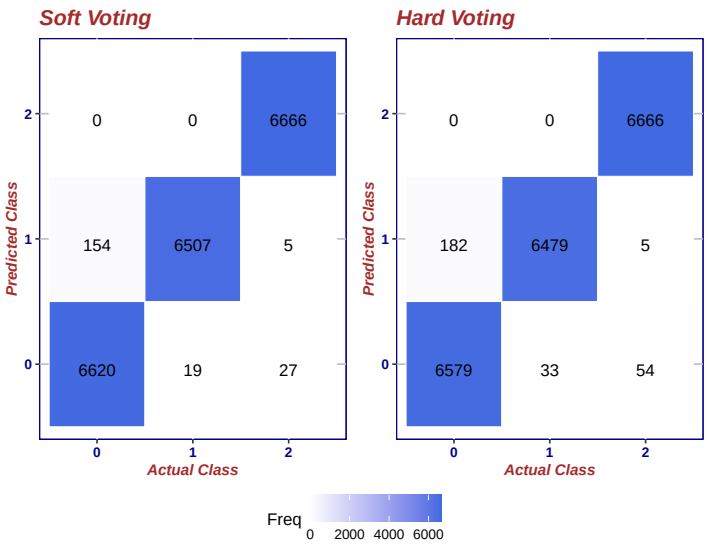


Figure 8: Confusion Matrices of Voting Classifier

6 Conclusion and future work

The voting classifier can improve the accuracy by choosing the best predict class from all the models. And it is more robust for missing data and outliers. From table 4 in appendix we can see that for accuracy, F1 score, precision and recall, soft voting performs better than weighted hard voting. Moreover, for most of the class, our voting model performs better than single model except for Galaxy and Quasar of image of spectrum data. CNN does a better job for these two class but much worse for Star. However, the price for better performance is more data used and model model trained. Although we use as simple model as possible for this job, the training and combining is also much more complicated for soft voting and weighted hard voting. In addition, the data we use is in very good quality and quit sufficient for training a well performed model which is not so common in real data. The corrupt or insufficient data may affect the accuracy of single model as well as voting classifier. According to these facts, our future work may focus on use less models with better voting strategies and use less data with worse data quality to test them.

(need table for coefficients of logistic regression, picture of tree model, intepretation of logistic regression (one unit change leads to ...) and tree model (an object with u<.. redshift > ... will be classified as ....))

7 Appendix

Table 4: Evaluation of Models

Data		M	M	M	IC	IS	M+IC+IS	M+IC+IS
Model		kNN	DT	LR	CNN	CNN	SVC	HVC
Accuracy		0.968	0.9744	0.97	0.9173	0.8891	0.9897	0.9863
Precision	Galaxy	0.9576	0.9434	0.9619	0.9522	0.9927	0.9773	0.9731
	Qso	0.9886	0.9882	0.9889	0.9296	0.9908	0.9971	0.9949
	Star	0.9585	0.9934	0.96	0.8745	0.7577	0.9952	0.9912
Recall	Galaxy	0.9484	0.9829	0.9509	0.9736	0.8179	0.9931	0.9869
	Qso	0.9719	0.9413	0.9616	0.8281	0.8524	0.9761	0.9719
	Star	0.9838	0.9989	0.9974	0.9503	0.997	1	1
F1	Galaxy	0.953	0.9628	0.9564	0.9628	0.8969	0.9851	0.98
	Qso	0.9802	0.9642	0.9751	0.8759	0.9164	0.9865	0.9833
	Star	0.971	0.9962	0.9784	0.9109	0.861	0.9976	0.9956

Note:

M: Metadata. IC: Image of Celestial Objects. IS: Image of Spectrum.