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:

Figure 1: Image of the celestial objects

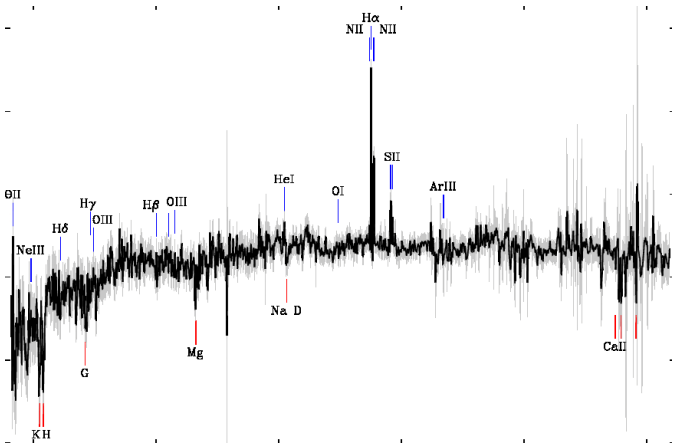


Figure 2: Image of the spectra of the celestial objects

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

vars	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 EDA

First we will check and deal with missing values in the dataset. There are 1 missing value for **i** and 3 for **z**. Then from Figure 1, we can see that **i** and **z** are quite scattered. So, we use regression imputation to impute the missing values.

Next, we give several visualizations to better understand the data. Figure 2 gives the spread of the celestial objects (galaxies, quasars, stars) across the universe. The x-axis is right ascension angle (at J2000 epoch), and the y-axis is declination angle (at J2000 epoch). We can see some interesting patterns, but it is unclear for classification.

Figure 3 gives the correlationship between varibales. There are some variables that have strong correlationships.

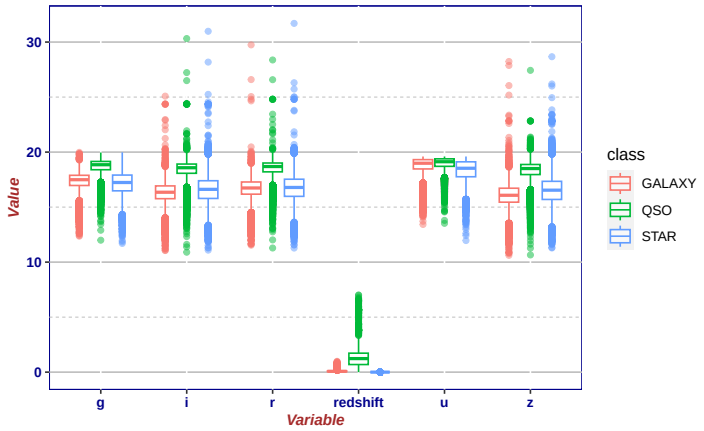


Figure 1: Boxplot

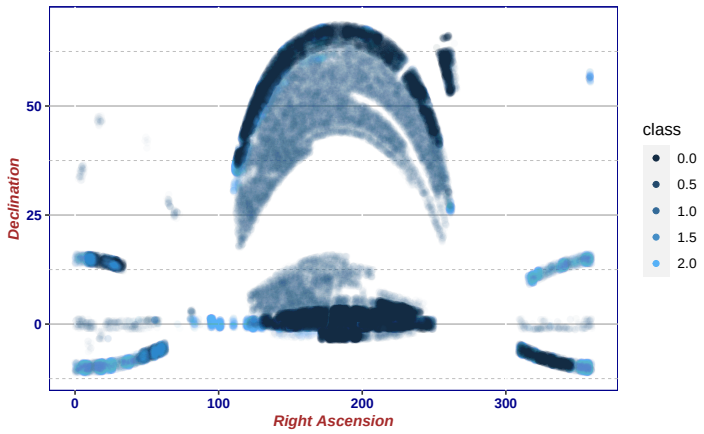


Figure 2: Spread of Stellars

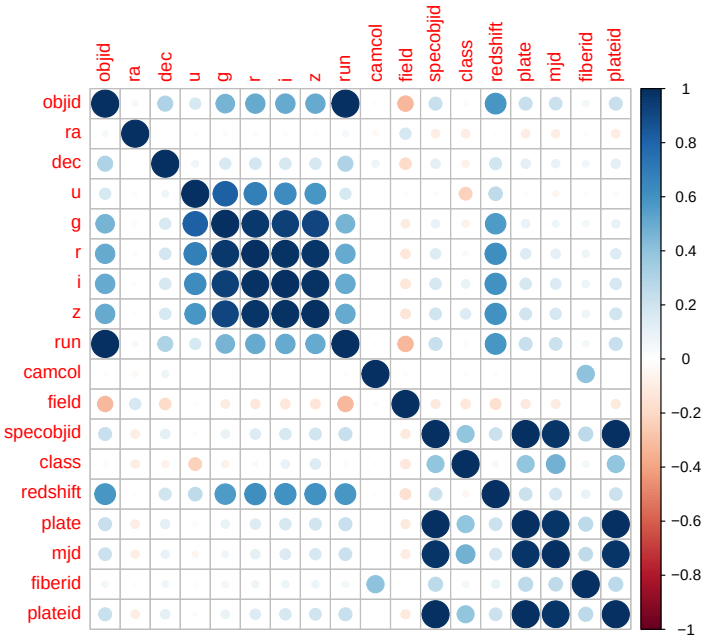


Figure 3: Correlation

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.

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

4.1 kNN

4.2 Decision Tree

4.3 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).

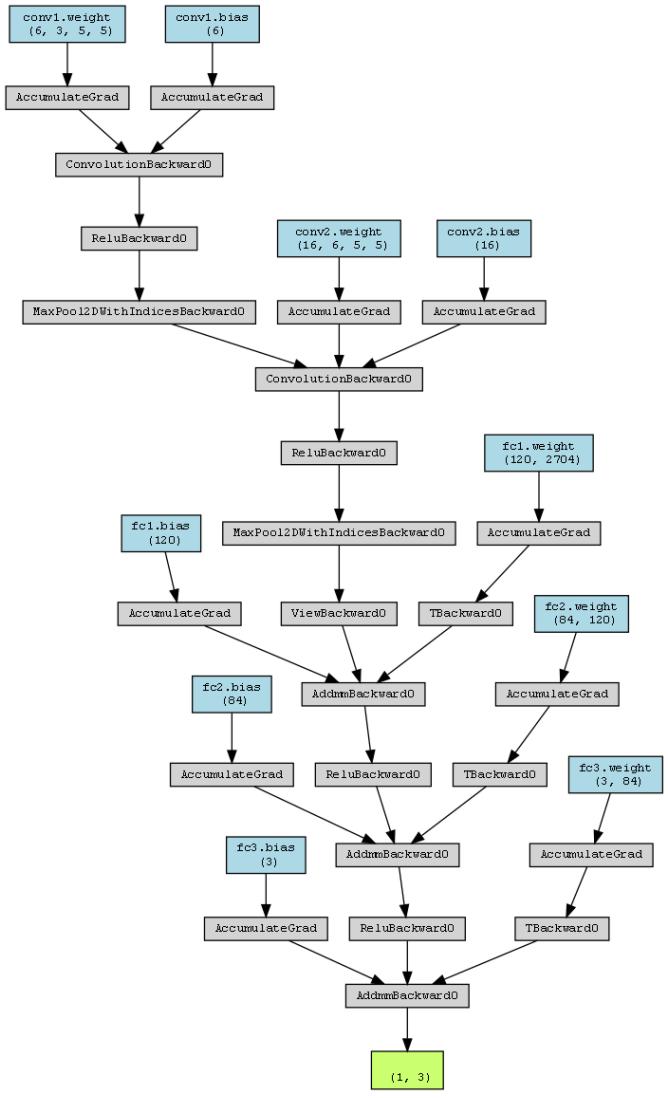


Figure 4: Structure of CNN

4.4 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.

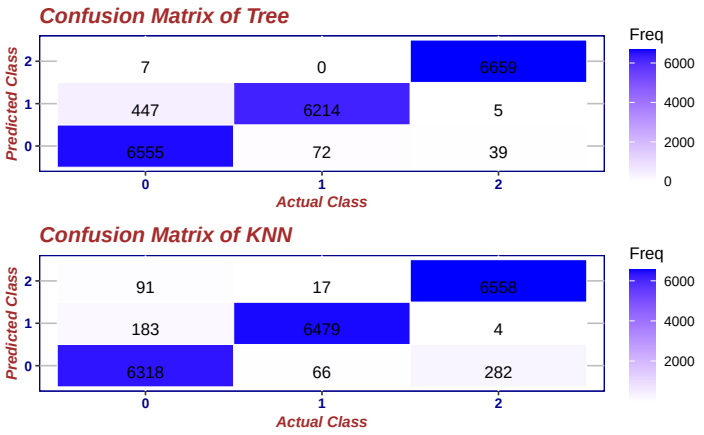


Figure 4: Confusion Matrix

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.

Table 2: Statistics from Confusion Matrix

Statistic	Tree	KNN
Accuracy	0.97150	0.96785
Kappa	0.95725	0.95177
AccuracyLower	0.96910	0.96531
AccuracyUpper	0.97376	0.97025
AccuracyNull	0.33333	0.33333
AccuracyPValue	0.00000	0.00000
McnemarPValue	0.00000	0.00000

Table 3: CNN comparison

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

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.

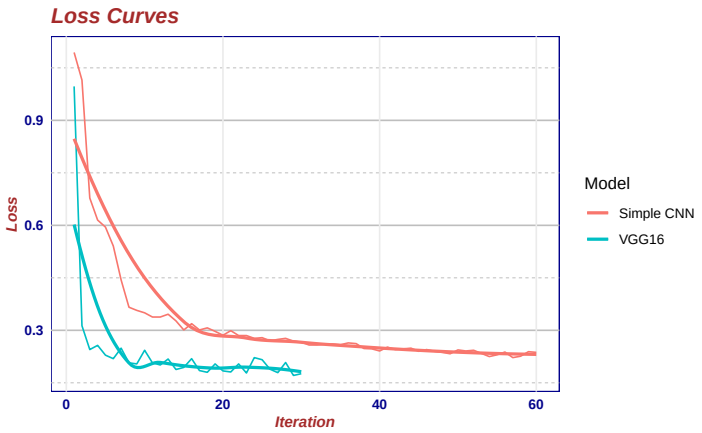
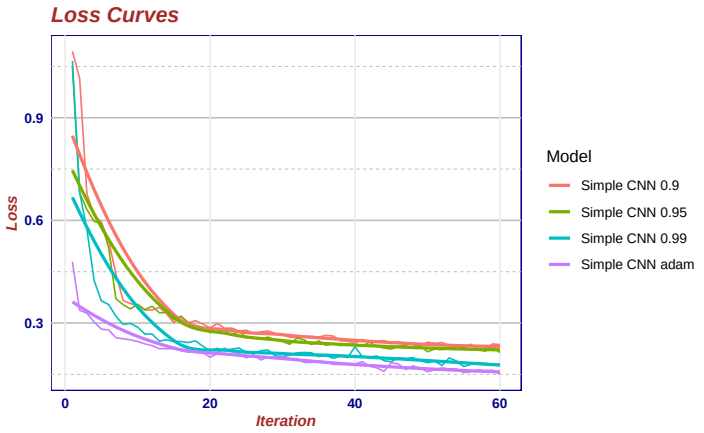


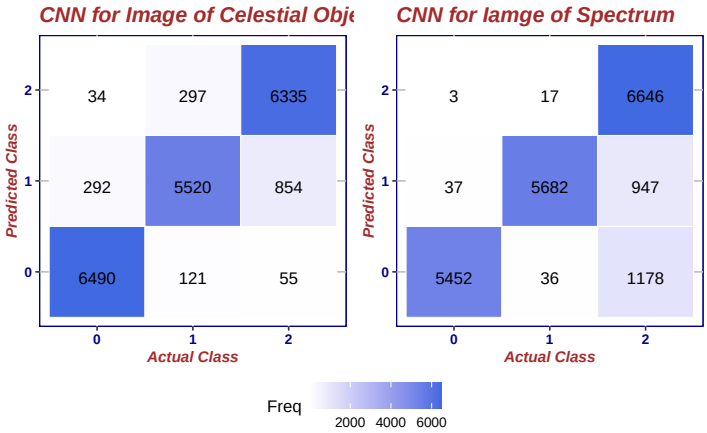
Figure 5: loss curves

Table 4: CNN tuning comparison

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

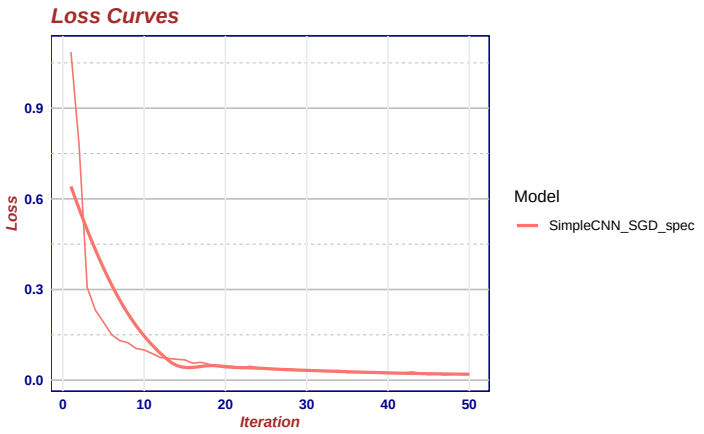


The accuracy of CNN of images 91.73%, CNN of spectrum is 88.91%



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

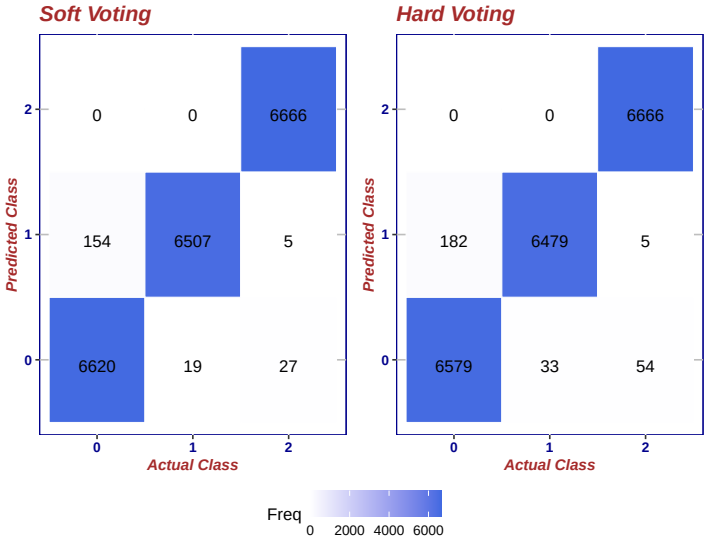


Figure 6: Confusion Matrices of Voting Classifier

(need a table)

6 Conclusion and future work

The voting classifier