Final Project

Exploring the night sky

Sonali Pattnaik
University of Washington

March 16, 2018

Abstract

This paper describes automated classification of astronomical objects from Sloan Digital Sky Survey (SDSS) using different machine learning algorithms like Random forest, Support vector machine, Naive Bayes, and K-nearest neighbors. It also provides different exploratory insights into various aspects of the data. The classifiers are trained on a dataset comprising the u, g, r, i, z colors of 9416 galaxies, 8813 stars, and 1771 quasars. An analysis of the performance of the classifiers showed that random forest classifier was able to efficiently learn the non-linear boundaries separating the three classes.

1 Introduction and Overview

Modern astronomical sky surveys concerned with the study and characterization of distant objects such as stars, galaxies, or quasars certainly fits the concept of Big data. Such astronomical objects are often characterized through measurements of their optical spectrum. For example, the overall spectrum of a galaxy is simply the combined spectrum of all the stars and other radiating matter in the galaxy. As galaxies vary in structure and relative composition of star type and gas, their spectra will also vary.

In this project, I will use different machine learning classifiers in order to classify astronomical objects like stars, galaxies, and quasars. I will use the data from Sloan Digital Sky Survey (SDSS)[1] which can be downloaded freely from their website. Stars from our own galaxy look like point-sources, i.e., they have the shape of a point spread function. However, the galaxies being far away, can be resolved as an extended source object rather than a point source. The study of quasars, an amalgamation of the words "quasi-stellar radio source", has led to many advances in our understanding of fundamental physics. Quasars, also commonly referred to as QSOs (Quasi-Stellar Objects) or AGNs (Active Galactic Nuclei) are galaxies which contain supermassive black holes at their core. Their appearance on the sky is very similar to that of a normal star in our galaxy and hence have the shape of a point source.

2 Theoretical Background

The Sloan Digital Sky Survey or SDSS is a decade-plus multi-spectral imaging and spectroscopic redshift survey using a dedicated 2.5-m wide-angle optical telescope at Apache Point Observatory in New Mexico, United States. The telescope uses a camera of 30 CCD-Chips with 2048x2048 image points each. The chips are ordered in 5 rows with 6 chips in each row. SDSS photometric data are observed through five filters: u(ultraviolet), g(green), r(red), i, and z(infrared) at wavelengths of approximately 355.1, 468.6, 616.5, 748.1, 893.1 nm. The survey which began in 1998, covers 35% of the northen sky, with photometric observations of around 500 million objects and spectra for more than 3 million objects. A visualization of u,g,r,i,z filter curves as well as their central wavelength is shown in Figure 1.

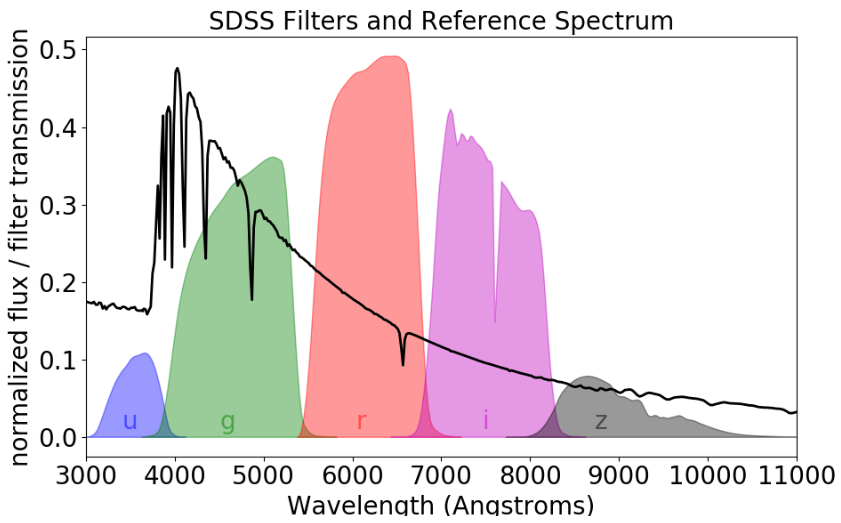

Figure 1: The spectrum of a star with the five filters (u,g,r,i,z) from the Sloan Digital Sky Survey (SDSS).

Examining high-resolution spectra from large, homogeneous datasets allow detailed investigations of the physical properties of these astronomical objects and provides new insights into the central engine powering these objects and their connections to their host galaxies. However, for long distance faint objects, obtaining these kind of spectra data can be very time consuming and expensive. Therefore, astronomers use broad-band filters and record the observations using the magnitude system. For the u-band filter shown above, the magnitude is defined as

$$u = m_{ref} - 2.5 \log_{10} \left[\int_0^{\infty} F(\lambda) S(\lambda) d\lambda \right] \tag{1}$$

Here F(λ) is the filter transmission, and S(λ) is the flux at wavelength λ. The constant m_{ref} encodes the calibration of the telescope. Subtracting two magnitudes reduces the uncertainty of calibrating m_{ref} from telescope to telescope. For example, an observation of a star will now consist of a vector four numbers: [u-g, g-r, r-i, i-z].

3 Algorithm Implementation and Development

The algorithm can be implemented in the following steps.

1. **Data preparation:** I read in the free SDSS data using a SQL query in Python with the help of the package astroquery. Here, I selected the top 20,000 objects from the SDSS PhotoObj database. The image table (PhotoObj) contains all image objects and the spectral table (SpecObj) contains the corresponding spectral data.

2. **Feature set:** SDSS has imaged the northern night sky in 5 bands :"u"(ultraviolet), "g"(green), "r"(red), "i" and "z"(infrared). In the table defined SpecObj, the class can be labeled as either star, quasar or galaxy depending on a response variable called 'redshift'. Redshift is the most common way to measure the distance to the galaxies.

3. **ML algorithms implementation:** Finally we train different models based on the spectral dataset. We will split the data into a training (80%) and a test part(20%). The models will be trained on the training data set and tested on the test data set and cross-validated several times to get the prediction accuracy.

4 Computational results

The spectrograph operates by feeding an individual optical fibre for each target through a hole drilled at the locations of the images in the telescope focal plane. Each hole is positioned specifically for a selected target, so every field in which spectra are to be acquired requires a unique plate which has a serial number. In spectroscopic mode, the telescope tracks the sky in the standard way, keeping the objects focused on their corresponding fibre tips. This can be visualized in the Table 1, which is derived from the data downloaded from SDSS server website. The dataset has 20000 examples with 9416 galaxies, 8813 stars, and

redshift	plate	mjd	fiberid
0.141838	294	51986	557
-0.000251	7456	56727	650
-0.000090	7456	56727	656
0.000368	7456	56727	660

Table 1: Spectral dataset with corresponding redshift, plateID (plate number of desired spectrum), mjd (mean julian date of desired spectrum) and fibreID (fiber number of desired spectrum)

1771 Quasars. The SDSS spectroscopic data is available as individual FITS files, indexed by three numbers: the plate, date, and fiber number[2]. The spectral data can be visualized in Figure 2.

The photometric data can be accessed directly using the SQL interface to the SDSS Catalog Archive Server (CAS). AstroML package contains a function which accesses this data directly using a Python SQL query tool. The data distribution as a function of 'u-g'(difference between u and g magnitudes) and 'g-r' (difference between g and r magnitudes) is shown in Figure 3.

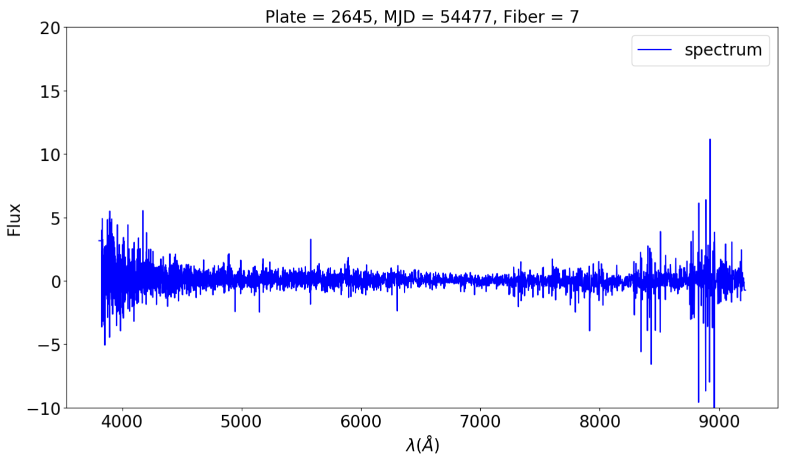

Figure 2: The spectrum for a particular plate number, mjd, and fiberid.

The most common way to measure the distance to galaxies is via redshift[3]. Light waves are compressed for the objects as they move towards you, the shorter wavelength meaning bluer light, or stretched as they move away, giving redder light. It's actually the expansion of the space that gives this redshift effect. This can be used to distinguish between stars, galaxies and quasars (Figure 4). For the star, the histogram has a zero-centered normal distribution. However, the galaxy has a left-shifted normal distribution and the quasars have an evenly distribution with some outliers around magnitude 4 to 5.

By analyzing all the 3 plots, we can say that the stars in this spectral data are closer than the quasars and galaxies. The quasars might have merged with many stars of different physical properties to give a more uniform distribution. Overall, redshift is a very important variable that speaks volumes about the characteristics of these three astronomical objects.

Now, let's take a look at the pdf's of the 5 PSF color magnitudes (u,g,r,i,z) for stars, galaxy, and quasars shown in Figure 5, 6, and 7, respectively We can see that stars and galaxies have overall brighter PSF magnitudes in all bands, especially the u band. Next, I will classify the sample based solely on this information. I split the data into a training set and test set (ratio 0.8/0.2), and then fit a different classifier models to classify the astronomical objects. I evaluate the score of different classifiers on the test set and cross-validate to produce the prediction accuracy (See Table 2).

Model	Score
KNN	79.7
Naive Bayes	78.83
Random Forest	98.83
SVM	46.5

Table 2: Machine learning models and their performance comparison on the spectral dataset.

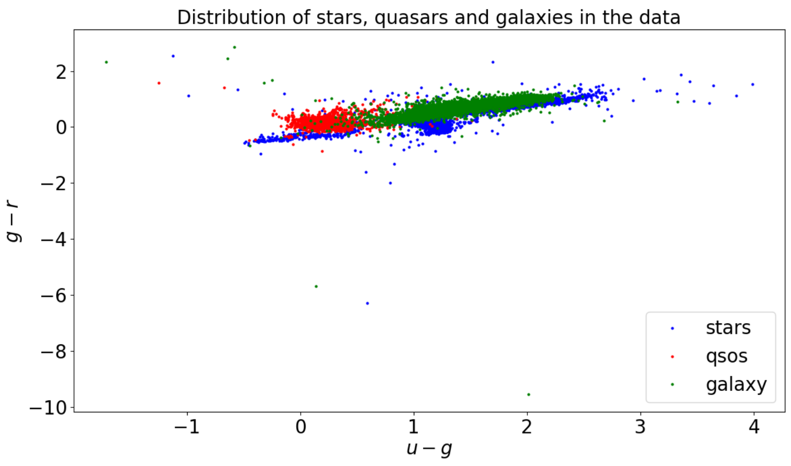

Figure 3: u-g and g-r colors of 16000 training data points

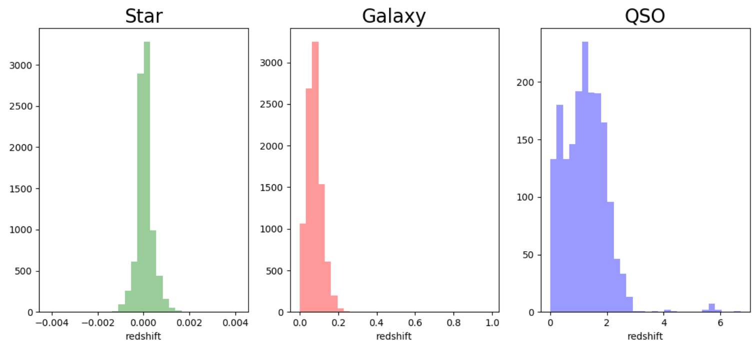

Figure 4: Redshift for Star, Galaxies, and Quasars.

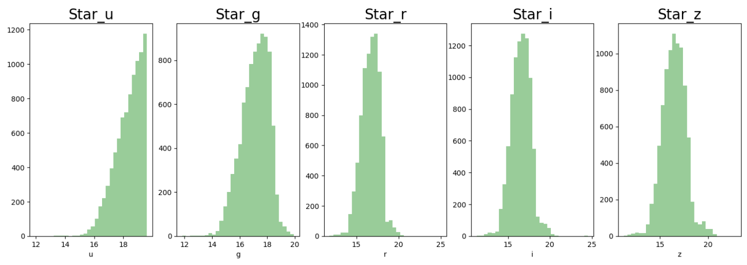

Figure 5: Stars observed though u,g,r,i,z band of filters

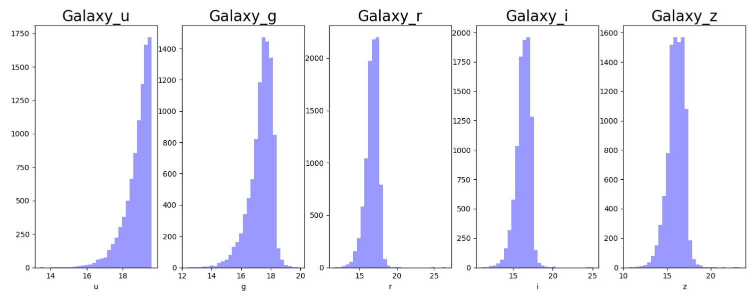

Figure 6: Galaxies observed though u,g,r,i,z band of filters

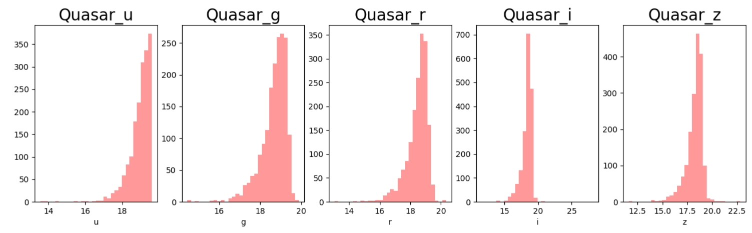

Figure 7: Quasars observed though u,g,r,i,z band of filters

Finally, I will plot confusion matrix (Figure 8). We see that most of the three classes are correctly identified (the diagonal elements are the most red). However, some of the quasars are miss-classified as galaxies.

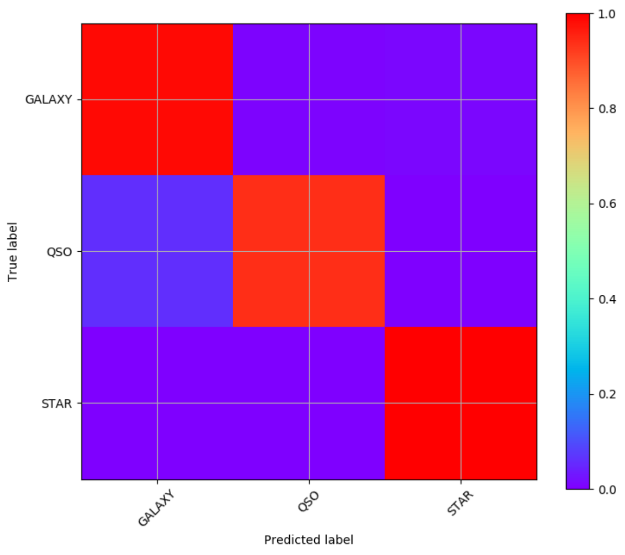

Figure 8: Confusion matrix

5 Summary and Conclusions

To summarize, random forest classifier could achieve high accuracy to classify SDSS objects. Gaussian Naive Bayes and KNN achieves similar accuracy levels but compute faster than random forest and SVM. Support Vector Machine Classifier gives a bad accuracy and takes a lot of time to classify. Also, analyzing the redshift data gives us a lot of insight into the distances of the astronomical objects from us.

References

[1] https://en.wikipedia.org/wiki/Sloan_Digital_Sky_Survey

[2] http://das.sdss.org/www/html/das2.html

[3] http://www.skytree.net/2016/04/12/sloan-digital-sky-survey-and-galaxies/

Appendix A – ML packages used and brief implementation explanation

1. scikit-learn : It is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, etc.

2. astroML : The goal of astroML is to provide a community repository for fast Python implementations of common tools and routines used for statistical data analysis in astronomy and astrophysics, to provide a uniform and easy-to-use interface to freely available astronomical datasets

3. panda : pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

4. seaborn : seaborn is a Python visualization library based on matplotlib.

Appendix B – codes

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.cross_validation import train_test_split
from astroquery.sdss import SDSS
from sklearn.ensemble import RandomForestClassifier
import seaborn as sns
from sklearn.neighbors import KNeighborsClassifier
from astroML.datasets import fetch_sdss_galaxy_colors
from astroML.plotting import scatter_contour
from astroML.datasets import fetch_sdss_spectrum
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC

query = """SELECT TOP 20000
p.objid,p.ra,p.dec,p.u,p.g,p.r,p.i,p.z, p.run, p.rerun, p.camcol, p.field,
s.specobjid, s.class, s.z as redshift, s.plate, s.mjd, s.fiberid
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 """

data = SDSS.query_sql(query).to_pandas()
k= data['class'].value_counts()
k1 = data.columns.values
k2 = data.head()
#print(k)
#print(k1)

################################
#plot the spectrum
# Fetch single spectrum
plate = 2645
mjd = 54477
fiber = 007
```

```
spec = fetch_sdss_spectrum(plate, mjd, fiber)
#ax = plt.axes()
#ax.plot(spec.wavelength(), spec.spectrum, '-b', label='spectrum')
#ax.legend(loc=1,fontsize =20)
#ax.set_title('Plate = %(plate)i, MJD = %(mjd)i, Fiber = %(fiber)i' % locals(),
fontsize =20)
#ax.set_xlabel(r'$\lambda (\AA)$',fontsize =20)
#ax.set_ylabel('Flux',fontsize =20)
#ax.xaxis.set_tick_params(labelsize=20)
#ax.yaxis.set_tick_params(labelsize=20)
#ax.set_ylim(-10, 20)

#plt.show()

###############################


X=data.drop("class",1)
y=data["class"]
train_X, test_X, train_y, test_y = train_test_split(X, y, train_size=0.7, random_state=0)

# KNN
clf=KNeighborsClassifier(n_neighbors=10)
clf.fit(train_X, train_y)
print(clf.score(test_X,test_y))

#Naive Bayes
gnb = GaussianNB()
gnb.fit(train_X, train_y)
print(gnb.score(test_X,test_y))

#Random Forest
rf = RandomForestClassifier(n_estimators=60)
rf.fit(train_X, train_y)
print(rf.score(test_X,test_y))


#SVM
svc = SVC()
svc.fit(train_X,train_y)
print(svc.score(test_X,test_y))




#plot data distribution
#data1 = fetch_sdss_galaxy_colors()
ug = train_X['u'] - train_X['g']
gr = train_X['g'] - train_X['r']
ri = train_X['r'] - train_X['i']
```

```
iz = train_X['i'] - train_X['z']
#spec_class = data['class']
#stars = (spec_class == 2)
#qsos = (spec_class == 3)
stars = (train_y == 'STAR')
glax = (train_y == 'GALAXY')
qsar = (train_y == 'QSO')
#red = train_X.redshift
#print(red[qsar])
#------------------------------------------------------------
# Plot stars, quasars, galaxies

#fig = plt.figure()
ax = fig.add_subplot(111)
ax.plot(ug[stars], gr[stars], '.', ms=4, c='b', label='stars')
ax.plot(ug[qsar], gr[qsar], '.', ms=4, c='r', label='qsos')
ax.plot(ug[glax], gr[glax], '.', ms=4, c='g', label='galaxy')
ax.xaxis.set_tick_params(labelsize=20)
ax.yaxis.set_tick_params(labelsize=20)
ax.set_title('Distribution of stars, quasars and galaxies in the data',fontsize =20)
ax.legend(loc=4,fontsize=20)
ax.set_xlabel('$u-g$',fontsize=20)
ax.set_ylabel('$g-r$',fontsize=20)

#plt.show()


from sklearn.metrics import confusion_matrix
cm = confusion_matrix(test_y,clf.predict(test_X))
cm1 = cm.astype(float)/cm.sum(axis = 1)[:,np.newaxis]

#plot the confusion matrix
plt.imshow(cm1, interpolation = 'Nearest', cmap = 'rainbow', vmin = 0, vmax = 1)
plt.grid()
plt.colorbar()
tick_names = ["GALAXY", "QSO","STAR"]
tick_marks = np.arange(len(tick_names))
plt.xticks(tick_marks, tick_names, rotation=45)
plt.yticks(tick_marks, tick_names)
plt.ylabel('True label')
plt.xlabel('Predicted label')
#plt.show()

# plot the redshift
fig, axes = plt.subplots(nrows=1, ncols=3,figsize=(16, 4))
ax = sns.distplot(data[data['class']=='STAR'].redshift,color="g", bins = 30,
ax = axes[0], kde = False)
ax.set_title('Star',fontsize=20)
ax = sns.distplot(data[data['class']=='GALAXY'].redshift,color="r", bins = 30,
ax = axes[1], kde = False)
```

```
ax.set_title('Galaxy',fontsize=20)
ax = sns.distplot(data[data['class']=='QSO'].redshift,color="b", bins = 30,
ax = axes[2], kde = False)
ax = ax.set_title('QSO',fontsize=20)

#plt.show()
#star
fig, axes = plt.subplots(nrows=1, ncols=5,figsize=(16, 4))
ax = sns.distplot(data[data['class']=='QSO'].u,color="r", bins = 30, ax = axes[0],
kde = False)
ax.set_title('Quasar_u',fontsize=20)
ax = sns.distplot(data[data['class']=='QSO'].g,color="r", bins = 30, ax = axes[1],
kde = False)
ax.set_title('Quasar_g',fontsize=20)
ax = sns.distplot(data[data['class']=='QSO'].r,color="r", bins = 30, ax = axes[2],
kde = False)
ax.set_title('Quasar_r',fontsize=20)
ax = sns.distplot(data[data['class']=='QSO'].i,color="r", bins = 30, ax = axes[3],
kde = False)
ax.set_title('Quasar_i',fontsize=20)
ax = sns.distplot(data[data['class']=='QSO'].z,color="r", bins = 30, ax = axes[4],
kde = False)
ax.set_title('Quasar_z',fontsize=20)
plt.show()
#galaxies
```