BST691 Final Project Report
Classification of celestial objects based on the RD14 data from the Sloan Digital Sky Survey using Machine Learning Methods

Ruijie Yin
Division of Biostatistics
Department of Public Health Sciences
Miller School of Medicine
University of Miami

Abstract

Autonomous sky surveys have been becoming increasingly critical, and large data sets of astronomical images have been generated and become available, however, it always requires manual classification of the celestial objects in the images and this can be a bottleneck of quick analysis of massive sky survey data sets for the astronomers; This project implemented several machine learning methods such as Random Forest and Support Vector Machines and compared their performance in multi-class classification problem based on the RD14 data from Sloan Digital Sky Survey; Results shows that Random Forest has the best performance in terms of classification accuracy among all the methods implemented.

Keywords: Sloan Digital Sky Survey, Machine Learning, Multi-class classification

1. INTRODUCTION

An astronomical survey is a general map or image of a region of the sky which lacks a specific observational target; it allows astronomers to classify celestial objects and perform statistical analyses on them without making prohibitively lengthy observations; In some cases, an astronomer interested in a particular object will find that survey images are sufficient to make telescope time entirely unnecessary; it can also help astronomers obtain observation time on larger, more powerful telescopes. If previous observations support a hypothesis, a telescope scheduling committee is more likely to approve new, more detailed observations to test it. Some of the famous sky surveys are Catalina Sky Survey, National Geographic Society – Palomar Observatory Sky Survey (NGS–POSS), SAGES Legacy Unifying Globulars and GalaxieS (SAGES Legacy Unifying Globulars and GalaxieS Survey (SLUGGS) survey and Ohio Sky Survey [1].

In the past several years, autonomous sky surveys have been becoming increas-

ingly critical, and large data sets of astronomical images have been generated and become available to the public. The availability of these large data sets has introduced the need for machine learning tools that can automatically analyze these images. This includes the need for automatic morphological classification of celestial objects that appear inside an astronomical image. Galaxy Zoo project allows hobbyist volunteers to manually classify galaxies via the project web site. The galaxy images are acquired by the Sloan Digital Sky Survey (SDSS), and displayed by Galaxy Zoo as JPEG images scaled by $0.024R_p$, where R_p is the Petrosian radius for the galaxy [2]. With an increasingly larger amount of images generated nowadays, manual classification of these celestial objects will soon become infeasible. It is inevitable to make the pipeline of this kind of job automatic, so that it will no longer be the bottleneck of quick analysis of massive sky survey data sets for the astronomers.

This project implemented several machine learning methods such as Random Forest, Support Vector Machine, K-Nearest Neighbour and Logistic regression to compare their abilities on classifying the celestial objects; The project report is organized as follows: In section 2, the data that was used was introduced and the methods mentioned above were reviewed in detail; In section 3, results of the classification accuracy are presented; In section 4, the performance of these four methods were compared and discussed and in section 5, the conclusion of what we have found were demonstrated.

2. METHODS

2.1. DATA

The data used in this project was from the 14th Data Release of space observations results from Sloan Digital Sky Survey (SDSS), it consists of 10,000 observations of space taken by the SDSS. Every observation is described by 9 feature columns and 1 class column which indicates it to be either galaxy, quasar or a star. Among 10000 observations, 4998 observations are labeled as galaxy, 4152 are star and the rest 850 are quasar;

The feature ra stands for J2000 Right Ascension (r-band), it is the angular distance measured eastward along the celestial equator from the Sun at the March equinox to the hour circle of the point above the earth in question, when paired with declination, as the feature dec in the data, which is another angle that locate a point on the celestial sphere in the equatorial coordinate system, measured north or south of the celestial equator, along the hour circle passing through the point in question, these two astronomical coordinates can specify the direction of a point on the celestial sphere in the equatorial coordinate system [3];

The Thuan-Gunn astronomical magnitude system, as features denoted as u, g, r, i, z in the data, represent the response of the 5 bands of the telescope; the system is defined by a few dozen standard stars, and the star BD+17deg4708, a subdwarf F6 star with $B-V=0.43$, is defined to have colors equal to zero; The feature field describes a field within an image taken by the SDSS, it typically starts at 11 (after an initial ramp-up time), and can be as large as 800 for particularly long runs; Redshift,

denoted as redshift in the data, is a phenomenon where electromagnetic radiation (such as light) from an object undergoes an increase in wavelength. Whether or not the radiation is visible, "redshift" means an increase in wavelength, equivalent to a decrease in wave frequency and photon energy, in accordance with, respectively, the wave and quantum theories of light [4].

We randomly selected 80% of the original data as our training data and the rest as testing data, the proportion of the three classes were the same as their proportion in the original data in both the training and testing data.

2.2. Data Analysis

2.2.1. Cross-validation

Cross-validation is used to select the best parameter in a model or calculate the misclassification error of the fitted model when the size of our dataset is limited. It is a re-sampling procedure and has a single parameter called k that refers to the number of groups that a given data sample is to be split into, which is often called the k-fold cross-validation. Thus, when a specific value for k is determined, it is then be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation. Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.It is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split. The general procedure is as follows:(1)Shuffle the dataset randomly.(2)Split the dataset into k groups,let $v \subset \{1,...,n\}$ be a random set with distribution P_v, let $L_{(v)}$ be the learning set restricted to v (the testing data) and $L_{-(v)}$ be the learning set restricted to $\{1,...,n\} \setminus v$ (the training data): for each unique group: fit a model on the training set and evaluate it on the testing set; calculate a cross-validated misclassification error for each of the model:

$$Err_{cv}(\hat{\mu}) = E_v[\frac{1}{|L_{(v)}|}\sum_{i \in L_{(v)}} Q(Y_i, \hat{\mu}(X_i, L_{-(v)}))] \tag{2.1}$$

where $\hat{\mu}(X) = \hat{\mu}(X, L)$ is an estimator for the unknown target function $\mu(X) = E(h(Y)|X)$ and $Q(Y, \hat{\mu}) \geq 0$ is the loss function. Then retain the parameter in the model giving the smallest misclassification error and finally use this parameter to refit a model using all the training dataset we have to obtain a final best model. Apparently, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. In this project, cross-validation was used to select the optimal k (the number of neighbours considered for each data point) when applying with the K-nearest neighbour method.

2.2.2. Principle Component Analysis

Principal component analysis (PCA, Pearson 1901, Hotelling 1933) is a mathematical procedure that transforms a number of (possibly) correlated variables into a (smaller) number of uncorrelated variables called principal components and is used by almost all scientific disciplines. Consider a vector $\boldsymbol{X} = (X_1, X_2, ..., X_p)^T$ comprised of different variables and the population variance $\Sigma = var(\boldsymbol{X})$. PCA intends to find a few standardized linear combination ($l^T X$, where $l^T l = 1$) that explain most of the variation in the data as measured by Σ. This is done by using orthogonal transformation that transforms X into a new uncorrelated Y. Let $\Sigma = P\Gamma P^T$ be the spectral decomposition of Σ, where Γ=diag$\{\lambda_j\}$,j=1...p are the eigenvalues of Σ and P_{pxp} is the orthogonal matrix comprised of the unit eigenvectors $e_1, e_2, ..., e_p$ of Σ. The principal component transformation of X is:

$$Y = P^T(X - \mu) = \begin{pmatrix} e_1^T \\ \vdots \\ e_p^T \end{pmatrix} (X - \mu) \tag{2.2}$$

where the variable

$$Y_j = e_j^T (X - \mu) \tag{2.3}$$

is the jth principal component score and e_j is the jth principal component loading. The first principal component Y_1 is the linear combination of x-variables that has maximum variance among all linear combinations, the second principal component Y_2 has the second largest variance, so on and so forth. This project retained the first principle components that account for more than 90% of the variation in the data as our new feature, and replaced the corresponding original features with it, in order to reduce multicollinearity and improve prediction accuracy.

2.2.3. Random Forest

Random Forest (Breiman, 2001) is a substantial modification of bagging that builds a large collection of de-correlated decision trees and then averages them. On many problems the performance of random forests is very similar to boosting, and they are simpler to train and tune. It utilizes decision trees, since they are ideal candidates for bagging, as they can capture complex interaction structures in the data, if grown sufficiently deep, have relatively low bias. The procedure of implementing Random Forest for classification is shown as below,

1. For $b = 1$ to B:
   (a) Draw a bootstrap sample Z^* of size N from the training data.
   (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, note that we do not prune the trees:
      i. Select m variables at random from the p variables.
      ii. Pick the best variable/split-point among the m variables.

iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_B\}_1^B$

To make a prediction at a new point x, let $\hat{C}_b(X)$ be the class prediction of the b^{th} random-forest tree. Then our prediction $\hat{C}_{rf}^B(x)$ is the result of the majority vote of $\{\hat{C}_b(X)\}_1^B$. The random forest can be directly used in multi-class classification problems, in this project, a total of 2500 classification trees in the forest were grown, results of different number of variables to be chosen for split were also briefly compared in section 4.

2.2.4. SUPPORT VECTOR MACHINE

Support vector machines (Vapnik, 1982) are supervised learning models with associated learning algorithms that can be used for classification and regression problems. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. It have strong theoretical foundations, excellent empirical successes, and have been applied to many tasks such as handwritten digit recognition, object recognition, and text classification. Take SVMs in the binary classification setting for example, a detailed implementation of hard-SVMs are illustrated as following:

We are given training data $\{x_1 ... x_n\}$ that are vectors in some space, for a decision hyper-plane $\mathbf{x}^T\mathbf{w} + b = 0$ (w is the slope and b is the intercept of the hyperplane) to separate the two classes $Class1 = \{(\mathbf{x}_i, 1)\}$ and $Class2 = \{(\mathbf{x}_i, -1)\}$, it has to satisfy

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 0 \tag{2.4}$$

for both $\mathbf{x}_i \in Class1$ and $\mathbf{x}_i \in Class2$. Among all such planes satisfying this condition, we want to find the optimal one hyperplane H_0 that separates the two classes with the maximal margin, which is the distance between the decision hyperplane and the closest sample points.

The optimal hyperplane should be in the middle of the two classes, so that the distance from the plane to the closest point on either side is the same. We define two additional hyperplanes H_+ and H_- that are parallel to H_0 and go through the point closest to the plane on either side:

$$\mathbf{x}^T\mathbf{w} + b = 1, \quad \text{and} \quad \mathbf{x}^T\mathbf{w} + b = -1 \tag{2.5}$$

All points $\mathbf{x}_i \in P$ on the positive side should naturally satisfy

$$\mathbf{x}_i^T\mathbf{w} + b \geq 1, \quad y_i = 1 \tag{2.6}$$

and all points $\mathbf{x}_i \in N$ on the negative side should satisfy

$$\mathbf{x}_i^T\mathbf{w} + b \leq -1, \quad y_i = -1 \tag{2.7}$$

These, however, can be combined into one single inequality:

$$y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1, \quad (i=1,\cdots,m) \tag{2.8}$$

The equality holds for those points on the planes H_+ or H_-. Such points are called *support vectors*, for which

$$\mathbf{x}_i^T\mathbf{w}+b = y_i \tag{2.9}$$

Moreover, the distances from the origin to the three parallel planes H_-, H_0 and H_+ are, respectively, $|b-1|/||\mathbf{w}||$, $|b|/||\mathbf{w}||$, and $|b+1|/||\mathbf{w}||$, and the distance between planes H_- and H_+ is $2/||\mathbf{w}||$.

Our goal is to maximize this distance, or, equivalently, to minimize the norm $||\mathbf{w}||$. Now the problem of finding the optimal decision hyperplane in terms of $\mathbf{w}$ and b can be formulated as:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} = \frac{1}{2}||\mathbf{w}||^2 \quad \text{(objective function)} \\ &\text{subject to} \quad y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1, \text{ or } 1-y_i(\mathbf{x}_i^T\mathbf{w}+b) \leq 0, \quad (i=1,\cdots,m) \end{aligned}$$

Since the objective function is quadratic, this constrained optimization problem is called a quadratic program (QP) problem. This QP problem can be solved by Lagrange multipliers method to minimize the following

$$L_p(\mathbf{w},b,\alpha) = \frac{1}{2}||\mathbf{w}||^2 + \sum_{i=1}^{m}\alpha_i(1-y_i(\mathbf{x}_i^T\mathbf{w}+b)) \tag{2.10}$$

with respect to $\mathbf{w}$, b and the Lagrange coefficients $\alpha_i \geq 0$ $(i=1,\cdots,\alpha_m)$. We let

$$\frac{\partial}{\partial W}L_p(\mathbf{w},b) = 0, \quad \frac{\partial}{\partial b}L_p(\mathbf{w},b) = 0 \tag{2.11}$$

These lead, respectively, to

$$\mathbf{w} = \sum_{j=1}^{m}\alpha_j y_j \mathbf{x}_j, \quad \text{and} \quad \sum_{i=1}^{m}\alpha_i y_i = 0 \tag{2.12}$$

Substituting these two equations back into the expression of $L(\mathbf{w},b)$, we get the *dual problem* (with respect to α_i) of the above *primal problem*:

$$\begin{aligned} &\text{maximize} \quad L_d(\alpha) = \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T, \mathbf{x}_j \\ &\text{subject to} \quad \alpha_i \geq 0, \quad \sum_{i=1}^{m}\alpha_i y_i = 0 \end{aligned}$$

Solving this dual problem we get α_i, from which $\mathbf{w}$ of the optimal plane can be found. Notice that those points $\mathbf{x}_i$ on either of the two planes H_+ and H_- (for which the equality $y_i(\mathbf{w}^T\mathbf{x}_i+b)=1$ holds) are called *support vectors* and they correspond

to positive Lagrange multipliers $\alpha_i > 0$. The training depends only on the support vectors, while all other samples away from the planes H_+ and H_- can be discarded once the model is obtained. Soft-SVM can be obtained in a similar way except that we add an extra slack term to relax the condition when two classes are not linearly separable.

More generally, SVMs allow one to project the original training data to a higher dimensional feature space $\mathscr{F}$ via a Mercer kernel operator K. In other words, we consider the set of classifiers of the form:

$$f(X) = (\sum_{i=1}^{n} \alpha_i K(x_i, x)) \tag{2.13}$$

When K satisfies Mercer's condition (Burges, 1998) we can write:

$$K(u, v) = \langle \Phi(u), \Phi(v) \rangle \tag{2.14}$$

where $\Phi : X \rightarrow F$; We can then rewrite f as:

$$f(x) = \langle w, \Phi(x) \rangle, w = \sum_{i=1}^{n} \alpha_i \Phi(x_i) \tag{2.15}$$

Thus, by using K we are implicitly projecting the training data into a different, usually higher dimensional feature space $\mathscr{F}$. In this project, one-vs-one approach was used to decompose the multi-class classification problem into multiple binary classification problems, hence, three SVM models were fit and the majority voting principle was used to predict the label.

2.2.5. K-Nearest Neighbour

The k-nearest neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. In both cases, the input consists of the k closest training examples in the feature space. In this case, k-NN is used for classification and the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors; k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. It is among the simplest of all machine learning algorithms. Euclidean distance is often used to measure the similarity between the data points. In general, it has low bias but potentially high variance and is robust against outliers; However, the accuracy of knn can be severely degraded by the presence of noisy or irrelevant features, or if the features scales are not consistent with their importance. The algorithm is shown as below:

1. Choose a K, that is the chosen number of neighbors.
2. For each data point in the data:
   (a) Calculate the distance between the new data point and every data points from the data.
   (b) Sort the distances from smallest to largest.
   (c) Determine the K neighbours of the new data point.
   (d) Return the majority vote of the class label of its K neighbours.

2.2.6. Logistic Regression

In logistic regression, a single outcome variable Y_i (i = 1,..., n) follows a Bernoulli probability distribution that takes on the value 1 with probability π_i and 0 with probability 1_i. Then π_i varies over the observations as an inverse logistic function of a vector x_i, which includes a constant and $k-1$ explanatory variables:

$$Y_i \sim Bernoulli(Y_i|\pi_i) \tag{2.16}$$

$$\pi_i = \frac{1}{1+e^{-x_i\beta}} \tag{2.17}$$

In which the parameter $\beta = (\beta_0, \beta_1^T)^T$ is a k × 1 vector, where β_0 is a scalar constant term and β_1 is a vector with elements corresponding to the explanatory variables. One-vs-all approach was used so we ended up with three binary logistic classifiers; Our prediction for each observation in the testing data was the class that has the largest one-vs-all probability.

3. Results

In this section, the prediction accuracy of Random Forest, Support Vector Machine, Logistic regression and K-nearest neighbour method are compared in classifying galaxies, stars and quasars on the test data; Exploratory data analysis shows that the variables g, r, i, z are highly correlated, as shown in Figure 3.1 and the correlation matrix (Figure 3.2, correlations are rounded to four decimal places):

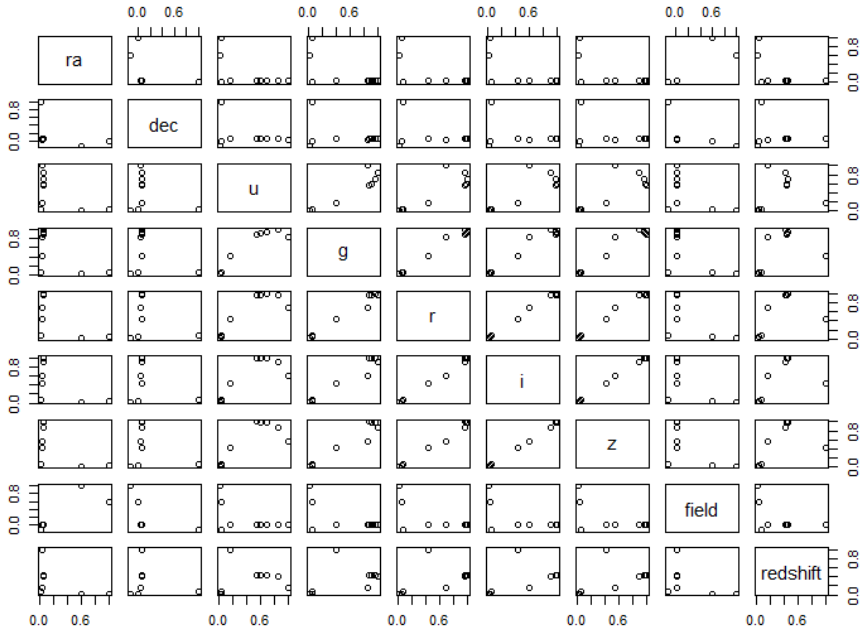

Figure 3.1: The correlation plot.

	ra	dec	u	g	r	i	z	$field$	$redshift$
ra	1.0000	0.0036	0.0312	0.0439	0.0471	0.0457	0.0429	0.5947	0.0303
dec	0.0036	1.0000	0.0353	0.0619	0.0634	0.0583	0.0569	0.1315	0.0670
u	0.0312	0.0353	1.0000	0.8492	0.6924	0.6026	0.5515	0.0085	0.1637
g	0.0439	0.0619	0.8492	1.0000	0.9581	0.9074	0.8796	0.0147	0.4076
r	0.0471	0.0634	0.6924	0.9581	1.0000	0.9777	0.9692	0.0171	0.4411
i	0.0457	0.0583	0.6026	0.9074	0.9777	1.0000	0.9815	0.0198	0.4314
z	0.0429	0.0569	0.5515	0.8796	0.9692	0.9815	1.0000	0.0182	0.4240
$field$	0.5947	−0.1315	0.0085	0.0147	0.0171	0.0198	0.0182	1.0000	0.0154
$redshift$	0.0303	0.0670	0.1637	0.4076	0.4411	0.4314	0.4240	0.0154	1.0000

Figure 3.2: Correlation Matrix

In order to reduce multicollinearity and improve prediction accuracy, a Principle Component Analysis was first performed to merge these four predictors into one new predictor, given that the first principle component has already explained 96.32% of the variation in the data, as shown in the two scree plots (Figure 3.3). The new variable is called 'coordinates', by the definition of principle component, it is a linear

combination of the variables g, r, i, z, and all these four variables are removed from our training and testing data, and replaced by the new variable.

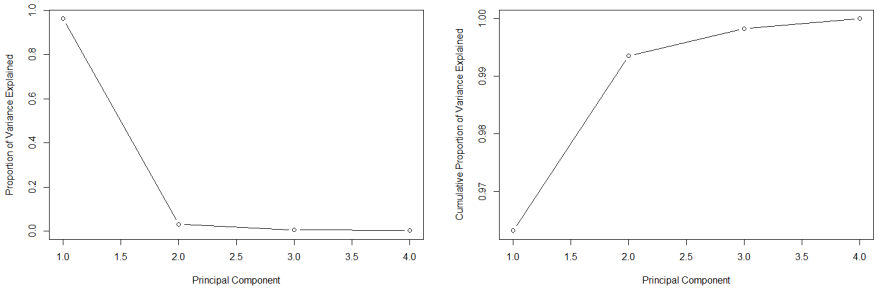

Figure 3.3: The scree plots.

Random Forest was first trained on the training data, the R package `randomForest` was used and a total of 2500 trees have been grown in the forest, the number of variables to be split at each node is set to be 2, the Out-of-Bag estimate of training error rate is only 0.99%; Figure 3.4 shows when the number of trees increases, how OOB error and in-class OOB error evolves; the black solid line is for overall OOB error, the three coloured lines, are for each class' OOB error:

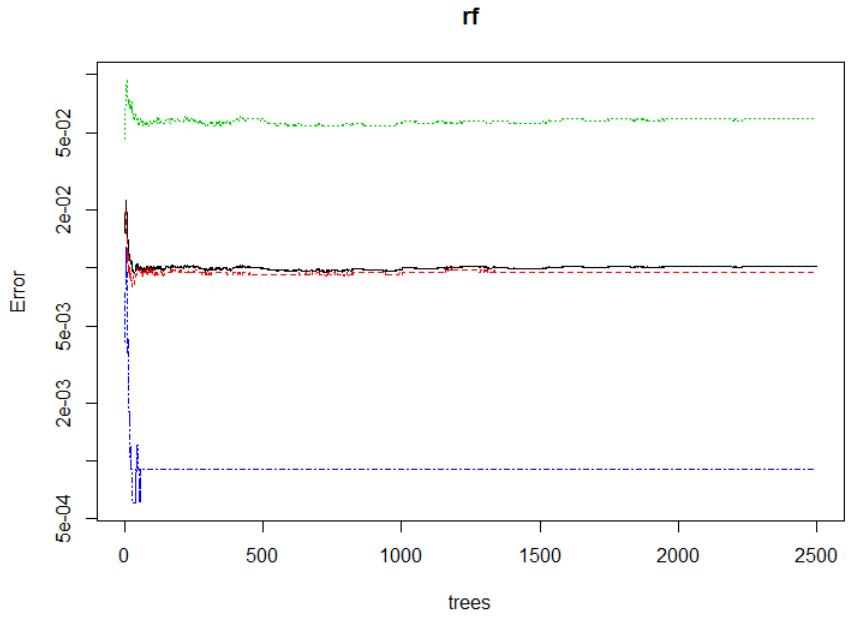

Figure 3.4: OOB evolves with increasing number of trees

The prediction accuracy of Random Forest on the testing data is 0.9835, the corresponding confusion matrix is shown as below (Table 3.1):

	Galaxy	Quasars	Stars
Galaxy	985	181	1
Quasars	7	151	0
Stars	6	1	831

Table 3.1: Confusion Matrix of Random Forest

The second method is Support Vector Machine, "one-vs-one" approach was implemented when finding the optimal separating hyperplane between classes; kernel of 'radial' was used and a total of 2723 support vectors were identified, more specifically, 1283, 134 and 1306 support vectors from classes galaxy, quasars and stars are identified, respectively. As an example on visualizing the support vectors, the decision boundary, and the margin for the model, a two-dimensional projection of the data (using the variables redshift and u) with three classes (shown in different shadings) and support vectors, are shown in Figure 3.5:

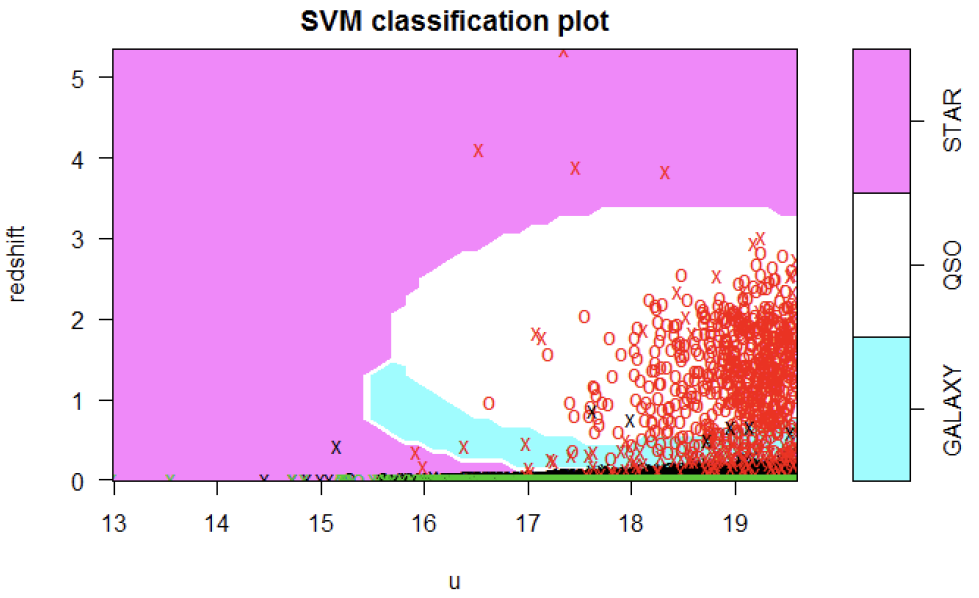

Figure 3.5: Projection of data on 2-dimension

R package `e1071` was used, we can also tune different values of parameters such as 'gamma (parameter needed for all kernels except linear kernel)' and 'cost (cost of constraints violation)' to find the best classification accuracy; When the default settings are used, the prediction accuracy is high enough as 0.9435; Its confusion matrix is shown in Table 3.2.

	Galaxy	Quasars	Stars
Galaxy	912	6	80
Quasars	20	149	1
Stars	6	0	826

Table 3.2: Confusion Matrix of SVM

The third method implemented was the Logistic regression; a 'one-vs-all' system of logistic classifiers was attempted, in this case, multiple binary classifiers, one for each of the three classes, was trained and we received three predictions and the class that has the largest one-vs-all probability is the our prediction. The prediction accuracy is 0.9805 and its corresponding confusion matrix is in Table 3.3:

	Galaxy	Quasars	Stars
Galaxy	983	23	0
Quasars	6	146	0
Stars	9	1	832

Table 3.3: Confusion Matrix of Logisitic Regression

The last method we experimented with was K-nearest neighbor classifier. The optimal k, which is the optimal number of neighbours to be considered for each data point, was found to be 5 by using 10-fold cross-validation, as shown in the Figure, it gave us the largest accuracy of 0.6752 on the training data:

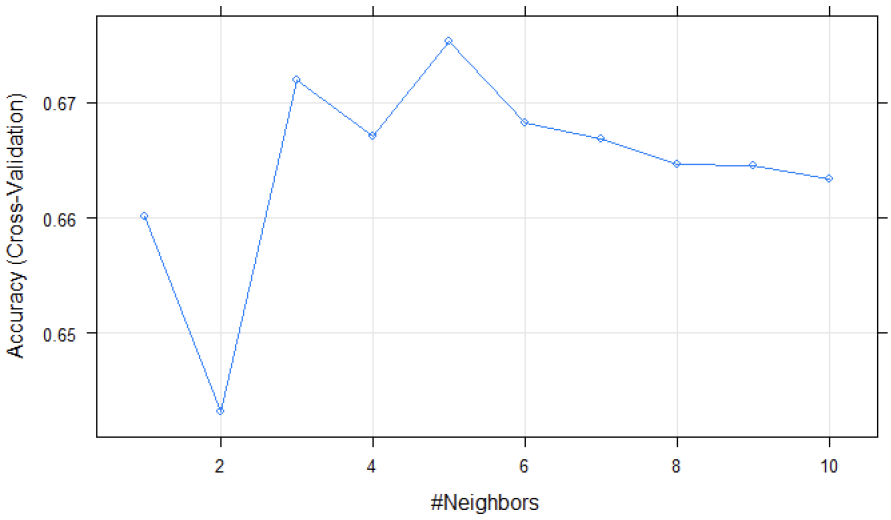

Figure 3.6: K v.s. Accuracy on training data

The prediction accuracy on testing data is 0.6625, and the confusion matrix is shown as below (Table 3.4):

	Galaxy	Quasars	Stars
Galaxy	748	62	296
Quasars	4	58	17
Stars	246	50	519

Table 3.4: Confusion Matrix of KNN

In summary, the classification accuracy of the four methods are listed in the Table 3.5:

Methods	Prediction Accuracy
Random Forest	.9835
Support Vector Machine	.9435
Logistic Regression	.9805
K-nearest neighbour	.6625

Table 3.5: Prediction accuracy of the four methods

4. DISCUSSION

From the Table 3.5, we see that Random Forest has an accuracy rate as high as 98.35%, this result is not surprising given that Random Forest has long been demonstrated suitable for multi-class classification problems, as it is a ensemble method that can capture enough underlying structure of the data and at the same time does not overfit; The number of variables to be split at each node has been tested with different values and they all gave similar results (not listed in the Results section), in order to boost computing time and enjoy the advantage of random subspace method, this number was finally set to be 2.

In comparison to other three methods, KNN does not perform very well, this is probably because some of the features used in predicting the class labels are actually irrelevant; the feature 'field' is perhaps one of the irrelevant features, since it is related to the telescope and the camera; thus, its accuracy are degraded when such features are used; while Random Forest handles these irrelevant features pretty well, in fact, Random Forest can still perform very well even if most of the predictive features are irrelevant.

The performance of SVM can still be improved by perhaps using some other kernel function and tweaking the parameters. However,its computing time with the default setting is already relatively longer than the other three methods, it will cost more time and effort in finding the best SVM model. Logistic regression performs slightly worse than Random Forest, but it also provides us with regression models that can be easily interpreted and used.

5. CONCLUSION

Random Forest was found to have the best performance in classifying the celestial objects based on astronomical images, given its performance and availability, random forest should probably become part of the "standard tool-box" of methods for this kind of task.

6. REFERENCE

[1]Wikipedia contributors, "Astronomical survey," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Astronomical_surveyoldid =880644452

(accessed April 23, 2019).

[2]Lior Shamir, Automatic morphological classification of galaxy images, Monthly Notices of the Royal Astronomical Society, Volume 399, Issue 3, 1 November 2009, Pages 1367–1372, https://doi.org/10.1111/j.1365-2966.2009.15366.x

[3]Wikipedia contributors, "Right ascension," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Right_ascensionoldid=890345933 (accessed April 23, 2019).

[4]Wikipedia contributors, "Redshift," Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Redshiftoldid=893885024 (accessed April 22, 2019).