Photometric Redshift Analysis using Supervised Learning Algorithms

A regression analysis with data taken from SDSS

Gautham Gururajan[1] and Shantanu Desai[2,⋆]

[1] Undergraduate, Indian Institute of Technology(IIT), Hyderabad
e-mail: ep17btech11008@iith.ac.in
[2] Professor, Indian Institute of Technology(IIT), Hyderabad
e-mail: shantanud@iith.ac.in

ABSTRACT

Aims. To explore some widely used feature selection algorithms and predictive models to measure photometric redshifts of galaxies with training information acquired from the SDSS Sky Server - Casjobs, from the DR10.
Methods. We use `MySQL` to query data with 49 easily measurable features.To select a few of them, we use Linear Correlation feature selection, Gini importance feature selection and `Boruta` Feature Selection and choose 20 features.We predict Photometric Redshifts before and after the feature selection and compare change in accuracy and runtime using Decision trees as our primary model, and then Two ensemble models built on our primary model-Random Forests(Bagging) and AdaBoost(Boosting).
Results. We attain accuracies of 95.79% for our decision tree model, 97.85% for our Random Forest model, 98.33% for our AdaBoost model after decreasing runtimes by 57.62%, 56.35% and 55.11% respectively through feature selection methods.We later do an analysis on outliers and find that the best performing algorithm produces an outlier rate of 55.65%, bias of 0.25500 and a precision of 0.1597
.

Key words. Methods – Machine Learning, Statistics Galaxies – Distances and Redshifts Cosmology – Observations

.

Introduction

For any meaningful extra-galactic analysis, the *distance* of the extra-galactic source in question is the most fundamental physical quantity that must be inferred. The most straightforward way to measure this distance would be through it's electromagnetic spectral energy distribution (SED), which is made of continuum and emission/absorption lines. Emission and absorption lines are sharp features which can be easily identified in the SED. Also, it is common knowledge that due to the expansion of the universe, the SED is shifted towards longer wavelengths by a factor of $1+z$ where z is the redshift(Other redshift mechanisms are made clear at [16]). The main difficulties in distance estimation is of finding a pair of characteristic features in the SED and measuring the amount by which they have been stretched. By using a necessary reference point, the measured redshift is then mapped to a corresponding distance value. Two well known features shape the SED continuum,

- The *Balmer break*(Below 4000Å), which is explained by absorption of photons more energetic than the Balmer limit at 3646Å and the combination of numerous absorption lines by ionised metals in stellar atmospheres.
- The *Lyman Break*(Below 1216Å), which is explained by absorption of light below the Lymann limit at 912Å and the absorption by the intergalactic medium along the line of sight.

When SEDs of sufficient wavelengths resolution are available, the emission/absorption lines can be identified, and the redshift precision can be measured to be better than 10^{-3} .
Despite having sophisticated multi-object spectrographs, only a few meaningful spectra for a few percent of the sources detected through deep imaging surveys are obtained. At least two well identified spectral features are required to obtain a robust redshift measurement. It is a fact that the success rate of measuring *spectroscopic redshifts* can be lower than 50-70% in deep spectroscopic surveys.

An alternative to this method is by measuring the flux of a source in broader filters. We can obtain a sample sufficient enough to give us details such as shape of the continuum, extra-galactic nature of sources and an estimate of redshift based on broader features such like the Lyman and Balmer breaks, or strong emission and absorption lines. This low resolution, less accurate distance estimate is called a *photometric redshift*. Although these measurements are relatively much cheaper with respect to computation cost, they pay the price by being much less precise(in the order of a 100 times factor).

In essence, a model to give us these photometric redshifts has a mapping between various fluxes/colors and the redshifts. In case of template fitting methods, the color-redshift mappings have been set based on parameters physically observed by scientists over a

⋆ Project Guide for EP3085

period of time.
In contrast, the Machine Learning methods use known values of redshifts and parameters to create a new mapping everytime it is applied, and is done through relevant 'training' samples, some of which we will see later on in this paper.

The data used in this analysis is from the publicly available archive, CasJobs, which we will talk about later. The data was surveyed through the *SDSS*.

SDSS

The *Sloan Digital Sky Survey*[22] began the operation phase at around May, 2000 and is currently operating in its fifth phase SDSS-V. The main intention for such a project was simple, to carry out a contiguous survey of all measurable galactic/extra-galactic objects.
This is being done through through imaging and spectroscopic surveys, with a dedicated 2.5m wide angle optical telescope equipped with a large format CCD camera to image the sky in five optical bands (Table 1).
Also it was equipped with two digital spectrographs to obtain the data of more than a million galaxies and more than a hundred thousand quasars selected from imaging data.
Spectra of more than three million astronomical objects have been obtained, along with documented deep multi-color images of more than one third of the sky.
It periodically updates the publicly accessible archives in phases known as *Data Releases*. Our paper makes use of the Data Release-10[1]

DR-10

As seen in Ref. [1], this data release includes the first spectroscopic data from the Apache Point Observatory Galaxy Evolution Experiment (`APOGEE`), along with spectroscopic data from the Baryon Oscillation Spectroscopic Survey (`BOSS`) [7] taken through July 2012 .
DR-10 also roughly doubles the number of `BOSS` spectra over those included in the ninth data release. DR10 includes a total of 1,507,954 `BOSS` spectra, comprising 927,844 galaxy spectra; 182,009 quasar spectra; and 159,327 stellar spectra, selected over 6373.2 square degrees.

Band	λ_{eff}
U	365 nm
G	475 nm
R	658 nm
I	806 nm
Z	900 nm

Table 1. Approximate Effective midpoints for the U-G-R-I-Z bands.

Getting the data

For every machine learning problem, the main concern is of getting a database large enough to provide

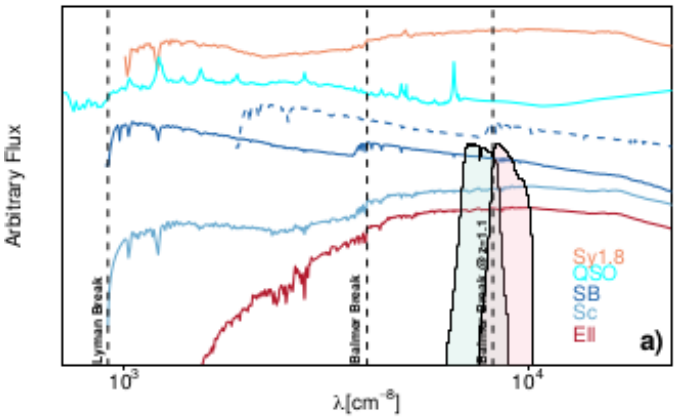

Fig. 1. Examples of SEDs for various type of galaxies(elliptical:Ell; Starburst:SB; spiral with small bulge: Sc) and AGN (luminous quasar: QSO; low luminosity, obscuredAGN: Sy1.8 (Seyfert 1.8)) The Lyman and Balmer breaks, among the key features in determining the redshifts, at rest-frame are indicated by vertical dashed lines. One template and the Balmer break are also plotted at redshift 1.1. For clarity, the transmission curves of i and z filters, covering the wavelength rage between 700 and 1100 nm are also indicated. Taken from [20]

results that are not very skewed towards a particular behaviour.
However, in our case it is relatively easy to obtain data, as associations like the SDSS make sure that all observed data is made available to the public through their sky server, named CasJobs[14].

Catalog Archive Server Jobs System allows you to access different types of schema. (A schema is a unique object that contains tables which contain information, A Catalog is a collection of schema : *To summarize, Cluster* > *Catalog* > *Schema* > *Table* > *Columns*/*Rows*), and on selecting one, you may issue your Query.
Since they would ideally need to have an arbitrarily large number of databases, `MySQL`[18] is provided to make queries instead of `SQL`.
We select the DR10-schema[1] and query out some features, most of which we think would be helpful to predict photometric redshifts (The code is available at the GitHub link specified). We subsequently issue our query and get 1,463,000 galaxies, of which only some will be chosen for the analysis(After pre-processing).

Acquiring and using a server

To the normal undergraduate researcher, running any prediction algorithm on a dataset of data exceeding 1 GB of space would be a difficult thing to do, due to lack of a server.
We have made use of Google Colaboratory[3], which is a free cloud service based on Jupyter Notebooks for machine learning education and research, and promotes ML/DL research along with free-of-charge access to a robust GPU. We have also made use of `PyDrive` (It is a wrapper library of google-api-python-client that simplifies many common Google Drive API tasks) to link our Google drive account with our Google Colab notebook, subsequently allowing us to access our data

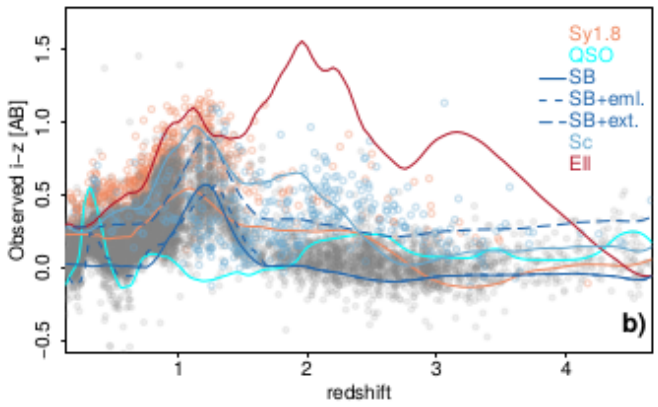

Fig. 2. (i-z) color as a function of redshift. Galaxies with reliable spectroscopic redshifts are represented with grey dots while AGN obscured/un-obscured by dust are represented with red/blue small dots. The solid lines represent the expected redshift evolution of the (i-z) color for the templates presented in the left panel, without any extinction. The star-burst galaxy (in blue) is also shown i)considering extinction (long-dashed line) and ii)considering the contribution from emission lines (short-dashed line).Taken from [20]

(The code to do so is given in the GitHub link).
If you have access to a server but wish to run a jupyter notebook on it while being able to edit it as you would on a normal environment, we perform an ssh tunneling procedure :
On your server's terminal :
`jupyter notebook --no-browser --port=ABCD`
which provides a url that has a token number within it.
On your local terminal :
`ssh -N -f -L localhost:XYZW:localhost:ABCD <username>@<address>`
And then simply go to your browser and access localhost:XYZW where you will be asked to enter the token number to proceed.

Supervised Machine Learning Terminology

Methods of Supervised learning

Supervised Machine Learning can be broadly divided into two types, those being *Classification* and *Regression*. Supervised classification is the task wherein there are established classes, and the data must be put into one of them depending on known conditions.
Similarly, we have supervised regression wherein some quantities of given data are predicted using previously collected information. Here, our problem requires us to use supervised regression techniques to predict photometric redshifts.

Training and Testing set

Our complete set is divided into a training set to help the model learn, and a testing set to verify the model's accuracy.
Here, we take our testing set as 40% of our complete set, and the rest as the training set.

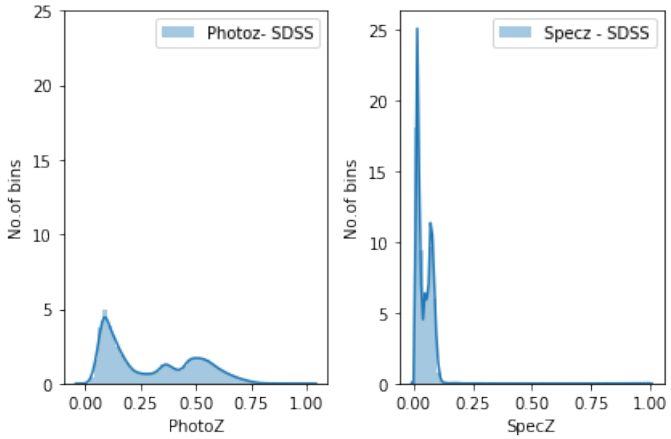

Fig. 3. As discussed earlier, we see that although they have peaks at similar redshifts, the distributions of photoz and specz differ considerably.

Z-score

A Z-score is a value which defined as:

$$Z = \frac{y_{obs} - \bar{y}}{\sigma} \tag{1}$$

Where y_{obs} is a chosen point, $\bar{y}$ is the mean of the distribution and σ is the standard deviation of that distribution.

Evaluation Metrics

Obviously, we would like to see how good our model does the task. For this reason, we will introduce some measures through which we can quantize goodness of the prediction.

- *MSE* :
  MSE or Mean Square Error is an evaluation metric that computes the quantity-

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - y_i^*)^2 \tag{2}$$

  Where y_i represents actual output and y_i^* represents predicted output.

  Or in other words, for each point, it calculates square difference between the predictions and the target and then averages those values. It is a good measure other than the fact that even if one prediction is wrong, the total MSE can be affected by a lot, and on the other hand, if the error is less than 1, the squared value will be even less.
- *RMSE* :
  The $RMSE$ or the Root Mean Square Error is defined as

$$RMSE = \sqrt{MSE} \tag{3}$$

  Which has similar properties as the MSE.

Description	Feature
Magnitudes	dered_u dered_g dered_r dered_i dered_z psfMag_u psfMag_g psfMag_r psfMag_i psfMag_z fiberMag_u fiberMag_g fiberMag_r fiberMag_i fiberMag_z
Radii	petroRadu petroRad g petroRad r petroRad_i petroRad_z expRad_u expRad_g expRad_r expRad_i expRad_z deVRad_u deVRad_g deVRad_r deVRad_i deVRad_z
Ellipticity	expAB_u expAB_g expAB_r expAB_i expAB_z
Extinction	extinction_u extinction_g extinction_r extinction_i extinction_z
Angles	expPhi_u expPhi_g expPhi_r expPhi_i expPhi_z ra dec

Table 2. The list of all input photometric features used in this work.A brief description of each of these features can be found on the SDSS Sky Server web page.

- R^2 :
  The R^2 or the Coefficient of Determination is a metric that is used to make sense of the error in an absolute scale. Our MSE or $RMSE$ may give us a certain number that can be understood only under the context of the values in our data, R^2 is a quantity that helps us figure out this quantity regardless of the scale of our data.

$$R^2 = 1 - \frac{MSE(Model)}{MSE(Baseline)} \tag{4}$$

  Where

$$MSE(Baseline) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \bar{y})^2 \tag{5}$$

  and $\bar{y}$ is the mean of all observed y_i.

  This shows us that R^2 can lie in between $-\infty$ and 1, where a value close to 1 would indicate a model with close to zero error, and a value close to zero would indicate a model very close to the baseline.
  A negative value would imply that the model's prediction is worse than the mean output value.

Decision trees

A decision tree (as the name suggests), is a tree like structure which has each node split into two or more nodes based on some predefined conditions. The starting node is known as the root node, other points of splitting are known as decision nodes, the split quantity is known as the parent node and the quantity that the parent node splits into is known as the child node.
A subsection of an entire tree is known as a branch or a sub-tree.
An collection of such trees whose results are used to make another final result (Through other case-wise predefined conditions which are discussed later on) is called a *Random Forest*. Here on, The feature space will be referred to as F_i, which is divided into τ branches β_τ, which have l leaf nodes per tree.

Gini importance

As mentioned in the previous sections, a branch is divided based on some predefined condition. This condition happens to be the *Gini Importance* (or *Mean Decrease Impurity*).
This quantity is constructed from the Gini Coefficient, which is the MSE of value of the item on each branch with the mean of the predicted value. Or more exactly, the Gini Importance is defined as the reduction of the Gini Coefficient from the parent branch to child sub branches.

Ensemble Learning methods

Ensemble Learning methods are algorithms that uses a variety of smaller, less accurate models to make one superior model (Sequentially - Boosting, Parallel - Bagging). In this paper we will be seeing ensemble models based on Decision Trees.

Pre-Processing

Our data consists of some `NaN` values as not all data points from the SDSS have been mapped.
To go through with our analyses, we are required to remove such NaN values, which is done with the help of the .drop() method available from `pandas`[17].
We subsequently remove other features that will not help with predictions like `objID` and `specObjID`(Which are just unique identities of objects/specimen and have no real meaning of use to redshift prediction).
We also scale all our features by calculating the mean of each feature, and then subtracting the mean of each feature from all other values of the feature and then dividing each difference by it's standard deviation, which has been a common standardization procedure.
We do this so that it is easy to compare variables that are initially in a different scale.

We drop all `NaN` values to end up with 935,531 data points for our analysis. All used features have been listed out in Table 2.

Feature Selection

Usually, in such machine learning problems, there are some features of the data set that may not be useful for task. Removing such features will help us increase the accuracy as well as decrease the run time of the model. We will try few such types of algorithms here, starting with an intuitive, vanilla method. More details on feature selection in the context of astrophysics can be found in Ref. [2].

Correlation Coefficients

One such method of filtering out features is checking using a simple metric - the *Pearson's correlation coefficient*, (will be referred to as $Corr_p$ from here on) which measures linear correlation between two variables. The resulting value lies in $[-1; 1]$, with -1 meaning perfect negative correlation (as one variable increases, the other decreases), $+1$ meaning perfect positive correlation and 0 meaning no linear correlation between the two variables. For two variables(Features) X and Y.

$$Corr_p[X, Y] = \frac{\mathrm{Cov}[X, Y]}{\sigma_X \sigma_Y} \tag{6}$$

Where σ_X and σ_Y are standard deviations of X and Y respectively and $\mathrm{Cov}[X, Y]$ is the Covariance between X and Y.

Here, $Corr_p$ is defined between each feature and photoz. We apply the above relation to obtain a correlation matrix and hence select the top 20 features (Which each have $|\ Corr_p\ | > 0.4$) The correlation heatmap can be seen in Figure 5. Although this method is seemingly

```
photoz          1.000000
deVRad_r       -0.425169
deVRad_i       -0.440057
deVRad_z       -0.411021
expAB_u        -0.541060
expAB_g        -0.441558
dered_u         0.818599
dered_g         0.915142
dered_r         0.889617
dered_i         0.842262
dered_z         0.807143
psfMag_u        0.798215
psfMag_g        0.895453
psfMag_r        0.845180
psfMag_i        0.765935
psfMag_z        0.724433
fiberMag_u      0.815311
fiberMag_g      0.909505
fiberMag_r      0.879347
fiberMag_i      0.824047
fiberMag_z      0.788084
Name: photoz, dtype: float64
```

Fig. 4. Top 20 features through $Corr_p$., the second column represents $Corr_p$

straightforward and quick. one obvious drawback is that this assumes only a linear dependency on the compared variables. In more complicated non-linear relationships, we notice that $Corr_p$ may be zero even if there is a 1-1 correspondence between the two variables. To make this correct, the first solution that one thinks of is a structure that easily deals with non-linearity - Decision Trees/Random Forests.

Gini Importance

From what was discussed before, we realize that the higher the Gini Importance, the more the feature is able

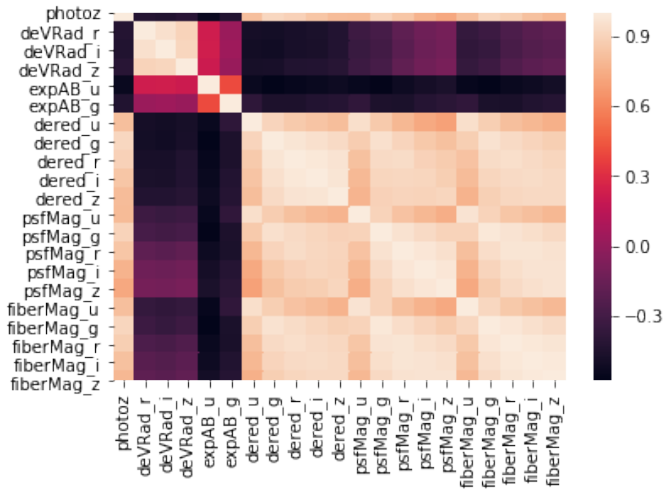

Fig. 5. Heatmap of the acquired top 20 features through $Corr_p$.

to split the training data into sub branches with similar values of redshift, and thus resulting in a higher predictive power of that particular feature.
Since we require this to construct our decision tree/random forest regressor model, there are no extra steps in doing a feature importance ranking.
We construct a random forest and then we list out the top 20 features with the best predictive power (Fig - 6). Our top 2 most prominent features are `dered_g` and `fiberMag_r` respectively with the `dered_g` feature harshly outweighing the rest. Although we get these as our Most Important Features, if our set of hyperparameters(Max tree depth and Min samples per leaf etc.) is changed, then we get different 'Most Important Features' accordingly, and is seen in [10]

Boruta

We have implemented the Boruta feature selection algorithm with the help of library `BorutaPy`.
Here's how it works :

- Duplicate features called shadow features are created.
- Randomness is then created by shuffling values in each column.
- Trains a regressor/classifier(In our case - Random Forest Regressor) and calculates gini importance.
- Now, each real feature is checked if it has a higher importance(Whether the feature has a higher Z-score than the maximum Z-score of its shadow features)
- At each iteration, important features and unimportant features are determined till the algorithm has reached a limit on number of random forest runs or when all features have been checked

We have followed the python implementation, and we ended up with 41 top ranked features (Fig - 7).
As discussed by [11] in their paper about Boruta,
" One should note that the Boruta is a heuristic procedure designed to find all relevant attributes, including weakly relevant attributes. Following Nilsson et al.(2007), we say that attribute is weakly important when one can find a subset of attributes among which this attribute is not redundant. The heuristic used in Boruta implies that the

attributes which are significantly correlated with the decision variables are relevant, and the significance here means that correlation is higher than that of the randomly generated attributes."
Hence we can say that boruta's output can be valued highly as a suggestion over a definite answer.
We see that the algorithm is completed after 9 iterations.

We find that the top 20 important features via inbuilt procedures for Random Forest Regression is a subset of this so from here on our data set for learning contains these 20 features. Also a notable quality of this feature selection model is that it has an extremely high time complexity

```
Top 20 Features - Gini Importance
dered_g
fiberMag_r
dered_u
fiberMag_i
psfMag_z
psfMag_g
fiberMag_g
fiberMag_z
fiberMag_u
dered_i
dered_z
dered_r
deVRad_g
expAB_r
psfMag_r
psfMag_u
psfMag_i
expRad_u
deVRad_z
```

Fig. 6. We use Gini importance to decide our top 20 features as discussed.

```
BorutaPy finished running.

Iteration:      9 / 100
Confirmed:      41
Tentative:      0
Rejected:       7
```

Fig. 7. Each step yields a similar form, we end up with 41 top important features as discussed.

Regression under ML

Each algorithm we have implemented has been timed by the python library `time`, and although we realise that the time taken completely depends on the server, we use it for the purpose of comparing the learning models.We compare time and accuracy before and after using the selected features.

Decision Tree Regressor

We use the scikit-learn package [19] and the implementation of Decision tree regressors as in [4] The branches are constructed with Pz (The photometric redshift) as the quantity in each leaf. As mentioned before, the quantity MSE must be minimized per leaf.

$$MSE = \frac{1}{N}\sum_{\tau=1}^{l}\sum_{F_i\in\beta_\tau}(Pz_i - \bar{Pz_\tau})^2 \tag{7}$$

Where

$$\bar{Pz_\tau} = \frac{1}{N_\tau}\sum_{F_i\in\beta_\tau} Pz_i \tag{8}$$

Where N_τ is number of objects in each leaf. Each branch of the tree corresponds to a region of the input feature space.The predicted redshift of a new object is obtained by assigning the value of $\bar{Pz_\tau}$ to the last leaf that the data falls upon. In this method, the tree is grown recursively such that the redshift in each is similar to the redshifts of the objects in the leaf. This similarity is measured by the MSE (Eq.8).The selection of the best input feature to split on, and the best splitting point, is determined using an exhaustive search to minimize Eq.1. The tree is grown till until each leaf contains N_τ, where N_τ is a specified quantity by the user. The feature importance score is then decided by minimizing the decrease in MSE(Gini importance) for each feature.

FS	R^2	*MSE*	*RMSE*	*Runtime*
Before	0.9554	0.00181	0.04265	59.56 sec
After	0.9579	0.00171	0.04139	25.80 sec

Table 3. Relevant parameters for Decision Tree Regression before and after feature selection.

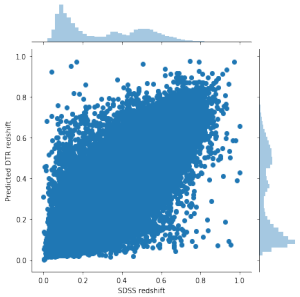

Fig. 8. Our predicted Redshift through Decision Tree Regression vs SDSS Photometric redshift

Random Forest Regressor

Random Forest is an ensemble algorithm designed through bagging (bootstrap aggregating) of the primary learners(Decision Trees).The *Randomness* comes from the fact that we do a random sampling of the set while building the trees and we take random subsets of features for splitting the nodes.In essence, a random forest builds multiple decision trees and merges their predictions together to get a more accurate and stable prediction rather than relying on individual decision trees.
Here, each tree learns from a random bootstrapped sample of the training set(the samples are drawn with replacement).The main idea behind this is that although each tree may have a high variance, the forest as a whole will have a low variance, with the cost of bias.

We follow the same method for each tree as in the decision tree regressor except each tree here acts only on a random subset of the feature space.In practice, for each tree the number of features is taken as the square root of the total number of features.
Here's one way to understand why a random forest would work better than a decision tree. In a random forest, since each decision tree acts on just a subset of the feature space, we have a limited scope of information.But if we could combine many such trees, we would have more detailed information than if a single tree would offer.
The ensemble prediction accuracy is simply the mean of accuracy of predictions of each tree from the previous step. We have followed the `sklearn` regression implementation of [15].

FS	R^2	*MSE*	*RMSE*	*Runtime*
Before	0.9775	0.00091	0.03025	360.50 sec
After	0.9785	0.00087	0.02656	157.34 sec

Table 4. Relevant parameters for AdaBoost Regression before and after feature selection.

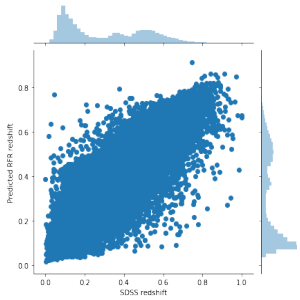

Fig. 9. Our predicted Redshift through Random Forest Regression vs SDSS Photometric redshift

AdaBoost

AdaBoost or Adaptive Boosting, is an ensemble boosting technique through which the output of other week learning algorithms taken into a weighted sum is the output of the boosted regressor.

$$Ada_n = \sum_{i=0}^{n} w_i a_i \tag{9}$$

Where w_i and a_i represent weights and 'weak learners' respectively. Here, our 'weak' learners will be Decision Trees, and weights are chosen based on errors by each learner.
We follow the `sklearn` implementation of AdaBoost[8]. Given below is a brief overview of the working of selecting the weights, described in [6], explained in [10].
Each galaxy is assigned a weight w_i, and a Decision Tree Regressor is trained on a bootstrapped dataset of size N. Each element has a probability of getting selected given by:

$$p_i = \frac{w_i}{\sum_{i=1}^{N} w_i} \tag{10}$$

A new model is thus produced which is added to the ensemble of models. The training set loss L_i , for each element i is calculated as

$$L_i = \frac{| Pz(F_i) - Pz_i |}{sup_j | Pz(F_j) - Pz_j |} \tag{11}$$

Where $Pz(F_i)$ is the function represented by the corresponding tree. L_i is normalized in such a way, that $L_i \in [0, 1]$. Average loss $\bar{L}$ is given by:

$$\bar{L} = \sum_{i=1}^{N} L_i pi \tag{12}$$

Where the sum runs on elements over the whole set. Confidence β is defined as:

$$\beta = \frac{\bar{L}}{1 - \bar{L}} \tag{13}$$

and the weights for each model are iteratively updated by multiplying the weights for each element in the training set by β^{1-L_i}
This weight update procedure gives less weight to elements with a low prediction error L_i and therefore these objects are less likely to be included in the training set drawn in the next boosting iteration. This focuses subsequent learners on elements with a high prediction error. We train a number of Decision Tree Regressors in this fashion and update the weights for the training set. The number of trees M included into the ensemble is decided by the user of the algorithm.If we query a new object with input features F_i , we obtain a prediction $Pz_j(F_i)$ for each tree in the ensemble $j \in 1, ..., M$. The final machine learning redshift prediction $Pz(F_i)$ is then given as the weighted median of the redshift predictions of the models in the ensemble with respect to $\log(\frac{1}{\beta_j})$ (as described in [6]).
As we see, since this model requires to compute many smaller prediction models, it has a large time complexity, which we see in Table 5.

FS	R^2	*MSE*	*RMSE*	*Runtime*
Before	0.9827	0.00070	0.02656	2660.38 sec
After	0.9833	0.00067	0.02603	1194.93 sec

Table 5. Relevant parameters for AdaBoost Regression before and after feature selection.

Outlier, Bias and Precision Analysis

Outliers are defined as the predicted data points that are relatively highly deviant from the true values according to a predefined metric or convention. The convention we

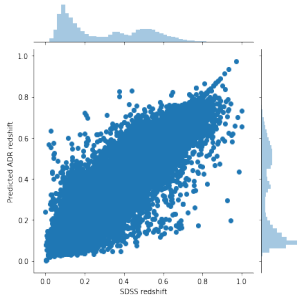

Fig. 10. Our predicted Redshift through AdaBoost Regression vs SDSS Photometric redshift

will use [9]is, a galaxy is an outlier O if :

$$O : \frac{|\, Pz_{pred} - Sz \,|}{1 + Sz} > 0.15 \tag{14}$$

$Precision(\sigma_{Pz})$ gives us an idea of the expected scatter between predictions and actual values. It is defined as :

$$\sigma_{Pz} = \sigma(\frac{Pz_{pred} - Sz}{1 + Sz}) \tag{15}$$

Where σ represents the Standard Deviation. *Bias* characterized the average separation between predicted and true redshifts.

$$Bias =< Pz_{pred} - Sz > \tag{16}$$

Where Pz_{pred} is the predicted photometric redshift and Sz is the corresponding actual spectroscopic redshift of the galaxy.
The above definitions of bias and precision are popular conventions followed in astrophysics. Following is a table with the acquired results:

	ADR	*RFR*	*DTR*
Bias	0.25500	0.25523	0.25526
Precision	0.1597	0.1587	0.1616
No.of outliers	208,259	208,996	209,526
Percentage outliers	55.65%	55.84%	55.99%

Table 6. ADR - AdaBoost Regressor, RFR - Random Forest Regressor, DTR - Decision Tree Regressor.

For this analysis, we use the spectroscopic redshifts that are corresponding to our testing subset, we have used 374,213 data points.

Discussion

The Photo-Z method for estimating galaxy distances has become widely popular in the last few years.
We conduct a simple search for 'Photometric Redshifts' on the ADS[12](Astrophysics Data System), and look at the trend of refereed papers, which is as shown below.
It is apparent that in the last two decades, the number of refereed papers has increased 8-9 fold, and possible reasons for this are the increase in number of multi-wavelength surveys. It is also expected that the use of photometric redshifts will be especially important in the coming years, given that photometric redshifts are becoming major tools in analyses of dark energy.

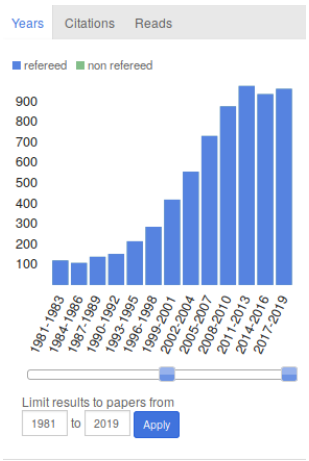

Fig. 11. A search for 'Photometric Redshifts' on the ADS reveals the above.

Photometric Redshifts are also important in applications in weak lensing tomography, which has become one of the main probes in surveys and cosmological experiments including DES[5] (Dark Energy Survey), LSST [21](Large Synoptic Survey Telescope), Euclid[13] etc, all of which are set to be the most important astronomical surveys in the near future.
Here are our results: For the Feature selection, we see that the Boruta algorithm gives us 41 top ranked features after 9 iterations.We see the disadvantages in Linear Correlation functions and we finally get our features ranked through mean decrease impurity (Gini Importance), and see that the top 20 features are a subset of the 41 top features from Boruta. For the Decision Tree Regressor we achieve a 95.54% R^2 accuracy, increase it to 95.89% while decreasing the runtime by more than 50%.Similarly, For the Random Forest Regressor we achieve 97.75% before, 97.85% after and decrease in runtime is again close to 50%.For our AdaBoost Regressor, we achieve 98.27% before, 98.33% after and decrease runtime by more than 55%, which is considerable given that each algorithm has a significant runtime. We have later checked each method for outliers and saw that Decision Tree Regressor gives 55.99% of the considered set as outlier data points, Random Forest Regressor gives 55.84%, and AdaBoost Regressor gives 55.65% which shows a consistent trend with respect to prediction power.

Conclusions

The aim of this project was to study and give a brief introduction to different tree-based learning algorithms, while evaluating performances through various metrics, as well as to understand the importance of photometric redshifts in extra-galactic analyses We first conducted a brief discussion on the SDSS archives and the DR10

schema and then followed up with an introduction to CasJobs. We then queried the relevant data through `MySQL` and stored our dataset as a csv file for further use. We further preprocessed our data and then conducted an analysis with various feature selection algorithms and then applied the learning algorthims and checked performance through some conventional metrics. We finally looked at the outlier rate, precision and bias for the mentioned models, which is useful when comparing different surveys. We thus conclude this paper whilst highlighting the ease of creating Photo-Z models and importance in future surveys.

References

[1] Ahn, C. P., Alexandroff, R., Prieto, C. A., Anders, F., Anderson, S. F., Anderton, T., Andrews, B. H., Aubourg, É., Bailey, S., Bastien, F. A., et al. The tenth data release of the sloan digital sky survey: First spectroscopic data from the sdss-iii apache point observatory galactic evolution experiment. *The Astrophysical Journal Supplement Series 211*, 2 (2014), 17.

[2] Bethapudi, S., and Desai, S. Separation of pulsar signals from noise using supervised machine learning algorithms. *Astronomy and Computing 23* (Apr 2018), 15.

[3] Bisong, E. Google colaboratory. In *Building Machine Learning and Deep Learning Models on Google Cloud Platform*. Springer, 2019, pp. 59–64.

[4] Breiman, L., Friedman, J. H., Olshen, R. A., and Stone, C. J. Classification and regression trees. belmont, ca: Wadsworth. *International Group 432* (1984), 151–166.

[5] Collaboration, D. E. S., et al. The dark energy survey. *arXiv preprint astro-ph/0510346* (2005).

[6] Drucker, H. Improving regressors using boosting techniques. In *ICML* (1997), vol. 97, pp. 107–115.

[7] Eisenstein, D. J., Weinberg, D. H., Agol, E., Aihara, H., Prieto, C. A., Anderson, S. F., Arns, J. A., Aubourg, É., Bailey, S., Balbinot, E., et al. Sdss-iii: Massive spectroscopic surveys of the distant universe, the milky way, and extra-solar planetary systems. *The Astronomical Journal 142*, 3 (2011), 72.

[8] Freund, Y., Schapire, R. E., et al. Experiments with a new boosting algorithm. In *icml* (1996), vol. 96, Citeseer, pp. 148–156.

[9] Hildebrandt, H. e. a. Outliers in redshift predictions.

[10] Hoyle, B., Rau, M. M., Zitlau, R., Seitz, S., and Weller, J. Feature importance for machine learning redshifts applied to sdss galaxies. *Monthly Notices of the Royal Astronomical Society 449*, 2 (2015), 1275–1283.

[11] Kursa, M. B., Jankowski, A., and Rudnicki, W. R. Boruta–a system for feature selection. *Fundamenta Informaticae 101*, 4 (2010), 271–285.

[12] Kurtz, M. J., Eichhorn, G., Accomazzi, A., Grant, C. S., Murray, S. S., and Watson, J. M. The nasa astrophysics data system: Overview. *Astronomy and astrophysics supplement series 143*, 1 (2000), 41–59.

[13] Laureijs, R., Amiaux, J., Arduini, S., Augueres, J.-L., Brinchmann, J., Cole, R., Cropper, M., Dabin, C., Duvet, L., Ealet, A., et al. Euclid definition study report. *arXiv preprint arXiv:1110.3193* (2011).

[14] Li, N., and Thakar, A. R. Casjobs and mydb: A batch query workbench. *Computing in Science & Engineering 10*, 1 (2008), 18–29.

[15] Liaw, A., Wiener, M., et al. Classification and regression by randomforest. *R news 2*, 3 (2002), 18–22.

[16] Marmet, L. On the interpretation of spectral red-shift in astrophysics: A survey of red-shift mechanisms-ii. *arXiv preprint arXiv:1801.07582* (2018).

[17] McKinney, W. pandas: a python data analysis library. *see http://pandas. pydata. org* (2015).

[18] MySQL, A. Mysql, 2001.

[19] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. Scikit-learn: Machine learning in python. *Journal of machine learning research 12*, Oct (2011), 2825–2830.

[20] Salvato, M., Ilbert, O., and Hoyle, B. The many flavours of photometric redshifts. *Nature Astronomy 3*, 3 (2019), 212–222.

[21] Tyson, J. A. Large synoptic survey telescope: overview. In *Survey and Other Telescope Technologies and Discoveries* (2002), vol. 4836, International Society for Optics and Photonics, pp. 10–20.

[22] York, D. G., Adelman, J., Anderson Jr, J. E., Anderson, S. F., Annis, J., Bahcall, N. A., Bakken, J., Barkhouser, R., Bastian, S., Berman, E., et al. The sloan digital sky survey: Technical summary. *The Astronomical Journal 120*, 3 (2000), 1579.