Feature importance for machine learning redshifts applied to SDSS galaxies

Ben Hoyle$^{1,2}$, Markus Michael Rau$^1$, Roman Zitlau$^1$, Stella Seitz$^{1,3}$, Jochen Weller$^{1,2,3}$

$^1$Universitaets-Sternwarte, Fakultaet fuer Physik, Ludwig-Maximilians Universitaet Muenchen, Scheinerstr. 1, D-81679 Muenchen, Germany
$^2$Excellence Cluster Universe, Boltzmannstr. 2, D-85748 Garching, Germany
$^3$Max Planck Institute for Extraterrestrial Physics, Giessenbachstr. 1, D-85748 Garching, Germany.

E-mail: hoyleb@uni-muenchen.de

Accepted —-. Received —-; in original form —-.

ABSTRACT

We present an analysis of importance feature selection applied to photometric redshift estimation using the machine learning architecture Random Decision Forests (RDF) with the ensemble learning routine Adaboost. We select a list of 85 easily measured (or derived) photometric quantities (or ‘features’) and spectroscopic redshifts for almost two million galaxies from the Sloan Digital Sky Survey Data Release 10. After identifying which features have the most predictive power, we use standard artificial Neural Networks (aNN) to show that the addition of these features, in combination with the standard magnitudes and colours, improves the machine learning redshift estimate by 18% and decreases the catastrophic outlier rate by 32%. We further compare the redshift estimate from RDF using the ensemble learning routine Adaboost with those from two different aNNs, and with photometric redshifts available from the SDSS. We find that the RDF requires orders of magnitude less computation time than the aNNs to obtain a machine learning redshift while reducing both the catastrophic outlier rate by up to 43%, and the redshift error by up to 25%. When compared to the SDSS photometric redshifts, the RDF machine learning redshifts both decreases the standard deviation of residuals scaled by 1/(1+z) by 36% from 0.066 to 0.041, and decreases the fraction of catastrophic outliers by 57% from 2.32% to 0.99%.

Key words: galaxies: distances and redshifts, catalogues, surveys.

1 INTRODUCTION

Large scale photometric galaxy surveys require precise redshift information to maximize information about cosmological parameters. Obtaining accurate spectroscopic redshifts are costly and time intensive and are typically only obtained for a small subsample of all galaxies. Conversely, the measurement of multi-band photometric properties of galaxies is much cheaper. The compromise is then to attempt to extract less accurate redshift information from photometrically measured properties, but applied to the full galaxy sample. This paper examines which photometric properties, or ‘features’, of the galaxies are best suited to this task by using feature importance analysis within standard machine learning architectures.

Photometric redshifts are often derived from galaxy Spectral Energy Distribution (hereafter SED) templates. The template redshift approach is well studied and is physically motivation. We know how the measured flux of a fiducial galaxy will change with redshift, and we can employ our knowledge of stellar populations and their evolution to predict how the SED and therefore fluxes and colors of galaxies will change as a function of redshift and galaxy type. However the encoding of the complex stellar physics, the computational time required to generate the templates, coupled with our uncertainty in the stellar population models and observed measurement error, combine to produce redshift estimates which are little better than non-parametric techniques (see e.g., Hildebrandt et al. 2010; Dahlen 2013, for an overview of different techniques).

Machine learning methods offer a non-parametric alternative to template methods. The ‘machine architecture’ learns how to combine, weight, or cluster the photometric galaxy properties to produce a machine learning redshift. The machine then examines the best combinations, clus-
tering, or weighting to minimize the difference between the spectroscopic redshift (of a training sample) and the machine learning redshift.

A major advantage of the template method over the machine learning method is the need of the latter to have a well-defined training set which spans the input feature parameter space of interest. On the other hand, one may view the need to generate templates and, importantly, a non-biased sample of templates, an equally large obstacle in order to produce reasonable template method redshifts.

Non-parametric photometric redshift estimation techniques have been developing since Connolly et al. (1995) and moved into the field of machine learning with the popular artificial Neural Network (aNN) code called aNNz (Collister & Lahav 2004). Since then a plethora of machine learning architectures, including Random Decision Trees, have been applied to the problem of redshift estimation (see e.g. Sánchez 2014 for a list of routine comparisons), and to estimate the full redshift probability distribution function (Gerdes et al. 2010; Carrasco Kind & Brunner 2013). Machine learning architectures have also had success in other fields of astronomy such as galaxy morphology identification, and star & quasar separation (see for example Lahav 1997; Yeche et al. 2009).

Most recently Brescia et al. (2014) applied an advanced type of deep aNN to a subset of galaxies drawn from the Sloan Digital Sky Survey (hereafter SDSS, York & SDSS Collaboration 2000) Data Release 9 (Ahn et al. 2012). The galaxy training sample was selected to be clean of artifacts, and to be confirmed spectroscopically as a galaxy. The resulting machine learning redshift error for this clean subset of galaxies is \( \sigma = 0.023 \), several factors smaller than the photometric redshift available in the SDSS CasJobs interface for the same sample of galaxies.

The machine learning community uses the nomenclature ‘features’ for items which are input into a machine architecture. For our purposes, the features can be any easily measured, or derived, photometric quantities that are available for each galaxy. For example standard machine learning redshift analysis uses a set of input features drawn from either fluxes or magnitudes, or a pivot magnitude and colors.

However one has the freedom to choose other features which are easily measured photometrically and which may also scale with distance. For example it is conceivable that the observed galaxy size, or the iso-photical radius in some (or all) bands may also encode redshift information. One may also find that galaxy inclination, or galaxy type as measured by the SDSS ‘fracDev’ parameter, or Stokes parameters, may also be valuable to identify (perhaps morphological) subsamples which may each have different redshift scalings. Given the quantity of easily obtained photometric parameters, and their ease of accessibility, it seems pertinent to explore if the addition of extra features can indeed improve machine learning redshift estimates.

Indeed early work using aNNs by Tagliaferri et al. (2003) find that the inclusion of radii and fluxes in addition to magnitudes improves the machine learning redshift estimation compared with using just magnitudes. However Sin-gal et al. (2011) use more derived morphological input features and a principle component type analysis to show that the addition of the examined features do not drastically improve machine learning redshift estimation. Furthermore Li et al. (2006) and Brescia et al. (2013) explore different magnitude definitions as input features of aNNs, and find that some magnitudes produce more accurate machine learning redshift estimates than others (e.g. dereddened model magnitudes from the SDSS).

We expand these previous result by compiling 85 standard (e.g. magnitudes) and extended photometric features, and then use feature importance to determine which features have the most predictive power when estimating the galaxy redshift. This is performed using standard feature selection analysis well known to the machine learning community. A full feature importance analysis with so many different features has yet to be applied to machine learning redshift estimation.

We examine the use of the following machine learning architectures: Random Decision Trees and Forests to perform the feature importance and to measure redshifts, and standard aNNs to see how the effect of selecting different features changes the recovered machine learning redshifts. We note that some of the aforementioned machine learning architectures are extremely scalable, and can be performed in a matter of tens of seconds on today’s desktop computers with sample sizes of millions.

We describing the data sample and the list of measured and derived features in §2. We continue by detailing the machine learning architectures and feature importance in §3. We present analysis and results in §4 and conclude and discuss in §5.

2 DATA

The data in this study are drawn from SDSS Data Release 10 (Ahn et al. 2014). The SDSS I-III uses a 4 meter telescope at Apache Point Observatory in New Mexico and has CCD wide field photometry in 5 bands (\( u, g, r, i, z \) Gunn et al. 2006; Smith et al. 2002), and an expansive spectroscopic follow up program (Eisenstein 2011) covering \( \pi \) radians of the northern sky. The SDSS collaboration has obtained 2 million galaxy spectra using dual fiber-fed spectrographs. An automated photometric pipeline performed object classification to a magnitude of \( r \approx 22 \) and measures photometric properties of more than 100 million galaxies. The complete data sample, and many derived catalogs such as the photometric redshift estimates, are publicly available through the CasJobs server.

The SDSS is well suited to the analysis presented in this paper due to the enormous number of photometrically selected galaxies with spectroscopic redshifts to use as training, cross validation and test samples. We select 1,958,727 galaxies from CasJobs with both spectroscopic redshifts and photometric properties. In detail we run the following MySQL query in the DR10 schema:

```
SELECT s.specObjID, s.objid, s.ra,s.dec, p.z AS photoz, p.zerr AS photoz_err, s.z AS specz, s.zerr AS specz_err, s.dered_u,s.dered_g,s.dered_r,s.dered_i, s.dered_z,s.modelMagErr_u,s.modelMagErr_g, s.modelMagErr_r,s.modelMagErr_i,s.modelMagErr_z, s.type as specType, q.type as photpType,
```

\footnote{skyserver.sdss3.org/CasJobs}
Feature importance for machine learning redshifts

2.1 SDSS DR10 photometric redshifts

The SDSS photometric redshifts are generated using a hybrid technique of the template method [Budavári et al. 2000] and a machine learning component using k-nearest neighbours (Csabai et al. 2007) technique as described in Abazajian et al. (2009). We hereafter refer to this combined method as 'template-ml'. The SDSS template-ml photometric redshifts are available from within CasJob by using the above SQL query.

2.2 Input features

Table 1 shows the list of photometric features used in this work. These photometric features are drawn from the following categories. Fluxes: corresponding to magnitudes measured in different bands and apertures, and color combinations created from them; Radii: measurements of sizes in different bands and with differing definitions; Morphology: how much of the light profile is best fit by one profile compared to others; and Shapes: ratio of major and minor ellipses measured in different bands and apertures, and color combinations created from them; and Sizes: linear measurements of sizes in different bands.

| Description | Feature name |
|-------------|--------------|
| Magnitudes  | dered_u dered_g dered_r |
|             | psfMag_u psfMag_g psfMag_r |
|             | fiberMag_u fiberMag_g fiberMag_r |
|             | petroRad_u petroRad_g petroRad_r |
|             | petroRad_i petroRad_z |
| Radii       | expRad_u expRad_g expRad_r |
|             | devRad_u devRad_g devRad_r |
| Colors      | dered_u-dered_g dered_u-dered_r |
|             | dered_g-dered_r |
|             | fiberMag_u-fiberMag_g fiberMag_u-fiberMag_r |
|             | fiberMag_i-fiberMag_z fiberMag_i-fiberMag_r |
| Profile     | fracDeV_u fracDeV_g fracDeV_r |
| Ellipticity | expAB_u expAB_g expAB_r |
|             | devAB_u devAB_g devAB_r |
|             | devAB_i devAB_z |
| Means Stokes| q_u q_v q_g q_u2 |
|             | q_r q_x q_y q_z |

Table 1. The complete list of the input photometric features used in this work. The Means Stokes parameters are shape features. A full description of each of the parameters can be found on the SDSS Sky Server web page.

© 2010 RAS, MNRAS 000, ??
The aim of this work is to identify which of these features provides the most predictive power for redshift estimation. In order to select these features, we perform feature importance, described in Section 3.4. We then obtain a ranking of features, in which the features which have the highest rank have the most predictive power and are selected as inputs into the final model.

2.3 Training, cross-validation and test subsets

We follow traditional machine learning nomenclature and methodology by randomly sub-dividing the galaxy catalog into a training, cross-validation and test samples with proportions 50%, 25%, 25% respectively. The training sample is used to train the machine learning system for a given architecture and hyper-parameter set. One uses the cross-validation sample to select the best values for the hyper-parameters of the learned system. Once the set of hyper-parameters has been decided upon, neither the training nor cross-validation sample provide a bias free estimate of the true error. In these cases the test sample is used to measure the true ability of the learned machine to generalize to a new dataset.

3 MACHINE LEARNING METHODS

Below we describe the two artificial Neural Network programs used in this work and the machine learning frameworks Random Decision Trees and the ensemble learning code Adaboost.

3.1 Artificial Neural Networks

We use two different artificial Neural Network (aNN) architectures. The first is the Java based applet PhotoRaper (Brescia & Cavuoti 2014) and the second is the Fortran code FaNN (Nissen 2003) which is callable from the command line and has wrappers in many languages including Python.

PhotoRaper is a standard multilayer perception with up to two (feed forward) hidden layers which trains the connections between neurons efficiently using a Quasi Newton Algorithm. We follow Brescia et al. (2014) and use an architecture of two hidden layers of size (11,4).

FaNN extends standard aNN architecture by incrementally building hidden layers and determining connections using the Cascade2 training algorithm (Fahlman & Lebiere 1990). Cascade2 training starts with an empty set of hidden layers and incrementally trains and adds one (multiply connected) neuron until either a user-specified maximum number have been added, or until the training error reaches some user-specified threshold.

The hyper-parameters of the aNNs explored in this work are the number of training examples (the training sample size), and the number of input features per training example, the number of hidden layers and neurons, and the method to learn the best connections between neurons.

3.2 Random Decision Trees

We use the Python package scikit-learn (Pedregosa et al. 2011) and the included implementation of Random Decision Trees for regression (Breiman et al. 1984). There exist other public implementations of Random Decision Trees for Classification and Regression, some of which estimate the full shape of the machine learning redshift probability distribution function (e.g. Carrasco Kind & Brunner 2013).

The Decision Tree Regressor sub-divides (or branches) the \(N\) data with respect to the feature space \(f\), into \(\tau\) branches \(B\), which end in \(L\) leaf nodes per tree. The branches are constructed such that the spectroscopic redshift \(z_{\text{spec},i}\) in each leaf has a low mean squared error

\[
MSE = \frac{1}{N} \sum_{\tau=1}^{\tau} \sum_{i \in B_{\tau}} (z_{\text{spec},i} - \langle z_{\text{spec},\tau} \rangle)^2 ,
\]

with respect to the mean spectroscopic redshift

\[
\langle z_{\text{spec},\tau} \rangle = \frac{1}{N_{\tau}} \sum_{i \in B_{\tau}} z_{\text{spec},i} .
\]

Where \(N_{\tau}\) is the number of objects in each leaf, The branches of the tree correspond to regions of input feature space. The machine learning redshift of a new object is obtained by assigning the value of Eq. 2 to the final leaf that the data falls upon. The tree is grown recursively such that the mean spectroscopic redshift in each leaf is similar to the spectroscopic redshift of the objects in the leaf. We measure this similarity using the mean squared error function (Eqn 1).

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 until each leaf node contains \(N_{\tau}\) objects, where \(N_{\tau}\) is a hyper-parameter of the model. We then determine the feature importance score by summing (and normalizing) the decrease in Eq. 1 for each feature.

Computing a Random Decision Tree is incredibly fast. This implementation can partition one million galaxies along 85 feature dimensions in a few tens of seconds using only one CPU core. Predicting a redshift for new data is also very fast, requiring just seconds on a dataset of size half of a million.

3.3 Adaboost

The power of the random decision trees can be enhanced by combining many trees together to make a Random Decision Forest or RDF. One method to achieve this is by using a learning implementation called Adaptive Boosting or Adaboost (Freund & Schapire 1997, Drucker 1997). We give a brief overview of the algorithm below, and refer the reader to Drucker (1997) for a more detailed description of the algorithm used in the scikit-learn routine. We note that Gerdes et al. (2010) also use Adaboost and RDF, but they determine the full shape of the machine redshift probability distribution function using SDSS magnitudes as input features.

The basic idea of boosting is to improve the performance of a base learner, in our case the Decision Tree Regressor, by using multiple models which put more weight on elements in the training set which have large prediction errors.

The algorithm described in Drucker (1997) first assigns equal weight \(w_i = 1\) to each galaxy in the training set. Subsequently one trains a Decision Tree Regressor on a new training set of size \(N\), by bootstrap selecting \(N\) samples with
replacement from the original training set. Each element has a probability to be selected given by
\[ p_i = \frac{w_i}{\sum_{i=1}^{N} w_i} . \]
This produces a new model which is added to the ensemble of models. The training set loss \( L_i \), for each element \( i \) is calculated as
\[ L_i = \frac{|z_{\text{phot}}(f_i) - z_{\text{spec},i}|}{\sup_j |z_{\text{phot}}(f_j) - z_{\text{spec},j}|} , \]
where \( z_{\text{phot}}(f_i) \) is the function represented by the corresponding tree. Note that \( L_i \) is normalized in such a way, that \( L_i \in [0, 1] \). We can then calculate the average loss \( \mathcal{T} \) of the model using
\[ \mathcal{T} = \sum_{i=1}^{N} L_i p_i , \]
where the sum runs over all elements in the training set. The confidence \( \beta \) for a specific model is defined by
\[ \beta = \frac{\mathcal{T}}{1 - \mathcal{T}} , \]
and the weights for each model are iteratively updated by multiplying the weights for each element in the training set by \( \beta^{1 - L_i} \).

The 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 [Freund & Schapire 1997, Drucker 1997]. We train a number of Decision Tree Regressors in this way and update the weights for the training set. The number of trees \( M \) included into the ensemble is a hyper-parameter of the model. If we query a new object with input features \( f \), we obtain a prediction \( z_{\text{phot},j}(f) \) for each tree in the ensemble \( j = 1, \ldots, M \). The final machine learning redshift prediction \( z_{\text{phot}}(f) \) is then given as the weighted median of the redshift predictions of the models in the ensemble with respect to \( \log (1/\beta_j) \) (as described in Drucker 1997).

### 3.4 Feature importance selection

One noteworthy feature of Random Decision Trees is the ability to determine which of the feature dimensions encode the most information about the quantity of interest, which here is the machine learning redshift. The scikit-learn implementation of Random Decision Trees uses the Gini importance to determine the predictive power of each feature.

The Gini importance is constructed from the Gini coefficient, which is the value of the MSE (Eq. 1) of the items on each branch. The larger the MSE the larger the Gini coefficient. As more sub branches are formed the data on each sub branch become more homogeneous which reduces the Gini coefficient. The Gini importance measures the reduction in the Gini coefficient from the parent branch to the child sub branches. For our purposes, the higher the Gini importance, the more the feature is able to separate the training data into similar redshift groups, and therefore the more predictive power the feature has. In summary, the more branches in the RDF that a particular feature dimension has, the more predictive power it has when estimating the redshift compared with other feature dimensions.

### 4 ANALYSIS AND RESULTS

We first determine which are the most important features using Random Decision Trees and Adaboost. We then document how the standard artificial Neural Network machine learning redshift is improved by the addition of the most important features. We then present what effect the size of the training sample has on the recovered machine learning redshift and finally show the further improvement in the photometric redshift of SDSS galaxies when using Random Decisions Forests.

#### 4.1 Feature importance

To determine feature importance, we first construct 25 Random Decision Tree Forests, and for each forest we vary the hyper-parameters. Upon completion of each forest Adaboost returns the list of input features with their representative importance weights. We rank the features by weight and extract the top 1, 2, 3 features for each forest. We construct a results cube which lists how often each feature has made it to a given importance rank.

In Fig. 1 we show the relative importance of different photometric features when determining a machine learning redshift. The first column shows the most important, or top ranked, feature after each iteration. The second column shows the second most important feature. The height of the color bar is the occurrence rate that the feature has been deemed to be the most important. Note that we marginalize over the machine learning architecture hyper-parameters. The x-axis labels corresponds to the ranked importance of the features and we label each feature on the figure. The colors represent the same feature across each column.

Surprisingly, we find that the most important feature is the value of the fiber magnitude measured in the \( g \) band, which, to the authors knowledge, has never been used in machine learning frameworks. We note that the SDSS fiber magnitude is a measure of the flux within a small (2 arc sec) radius around the galaxy center and is measured for all galaxies, not only the spectroscopic subset. We also note that the fiberMag \( g \) is not only the most important feature, but it is often more important by a factor of 3 more than the second most important feature.

To improve statistical significance we generate a further 350 further Random Decision Tree Forests, and repeat the feature analysis. We find fiberMag \( g \) is still the most important, top 1, feature 67\% of the time. However 25 other features also appear at least once in the top 1 feature ranking, and we decide to not include these in Fig. 1 to improve readability.

For fun we examine how the top features scale as a function of redshift and show this illustrative plot in Fig. 2. Recall that each of the input features are pre-scaled by subtracting the mean and dividing by twice the standard deviation, see §2.2. We have arbitrarily re-scaled the features, in a manner such as with some of the machine architectures may use. Note that this is an illustrative explanation of the power of
the most important features at determining galaxy redshift. The top panel shows two of the standard features used in machine learning (see e.g., Brescia et al. 2014). The bottom panel uses the top three most important features as determined by this work. The data in this illustrative example is drawn from the test sample, which has not been used during machine training, and is therefore independent of the analysis. The same sample of galaxies is shown in both plots.

4.2 The effect of the extended features

In this section we show what effect the use of extended features has on the machine learning redshift error estimated using popular frameworks. We choose two types of artificial Neural Networks, and fix their architectures in one approach, and allow it to vary in the other.

In the first approach we use PhotoRapter and fix the size of the training and test sets to be 120139, 360417, and the number of hidden layers to 2 and hidden units to (11,4). We note that here, the sum of the training and cross validation data set corresponds to 25% of the full data sample. Following Brescia et al. 2014 we refer to the following features as ‘standard’: psfMag_g, psfMag_u, psfMag_r, psfMag_i, psfMag_z, psfMag_j, psfMag_r. We use the results from Brescia et al. 2014 to determine which are the most important ‘extended’ features. The extended features are pre-scaled by subtracting the mean and dividing by twice the standard deviation, see §4.1. We note that the top 1 and 2 features have dimension 10, and the top 1, 2 and 3 have dimension 14.

We construct data vectors corresponding to the residual $\Delta z$, between the true redshift $z$ and the machine learning redshift, and also these residuals scaled by $1/(1+z)$. On these vectors we compute and compare measurements of the mean, standard deviation, and the percentage of catastrophic ‘outliers’ defined by $|\Delta z/(1+z)| > 0.15$. The measurements are performed on the following samples: the standard features, the top 1, and 2 best features, the standard & the top 1,2 best features, and the standard & the top 1, 2, and 3 best features. We show the results of this analysis using PhotoRapter in Table. 2.

We find that the values of the mean, standard deviations and percentage of outliers decreases if we choose to use those features deemed to be the most important from §4.1. Furthermore we find that the combination of the standard & top 1,2 (or standard & top 1,2,3) features continue to improve each of the measured values. We find an improvement in the standard deviation of the redshift scaled residuals by 18% and an improvement in the catastrophic outlier rate of 32%. We note that the total training time of PhotoRapter is approximately 4 hours for the stated hyper-parameters, on a single CPU core. The improvement in redshift estimation by combining radii, fluxes and magnitudes has been seen before (Gerdes et al. 2010), however our choice of which additional features to use is motivated by the importance feature selection.

In the second approach we allow the aNN architecture to vary. We calculate the machine learning redshift using the Cascade2 algorithm implemented in FaNN. We choose...
to examine the standard feature sample and the top 1&2 feature sample. For this analysis we marginalize over the machine architecture hyper-parameters by randomizing the number of neurons in the hidden (multiply connected) layers. The learning rates, the desired best error the learning algorithm will try to attain, and the number of galaxies in the training set. We train FaNN using the training set, and calculate the standard deviation of the residuals using the full cross validation set.

We find that the value of the standard deviation of the residuals decreases by 17 ± 8% using the top 1&2 features compared to the standard features. However the value of the standard deviation for the best hyper-parameter configuration is 0.081 which is not competitive with that obtained using PhotoRapier.

Finally, as a further illustrative example of feature importance we determine which are the two worst, or lowest ranking, features from each iteration. The worst features are psfMag_i-psfMag_u, fracDeV_g, fiberMag_j-fiberMag_u, fracDeV_i, dered_z-dered_u, psfMag_r-psfMag_u, fracDeV_r, fracDeV_u, dered_i-dered_u, fracDeV_z. We repeat the above analysis and pass these features to PhotoRapier to measure a machine learning redshift. The value of the standard deviation of the residuals is 0.089 which is larger than using the standard, or top ranked features. We note that the outlier rate is also very high, with a value of 4.5%.

### 4.3 The importance of the size of the training set

In this section we use the computationally fast Random Decision Forest with Adaboost as the machine learning architecture. This allows many different training sample sizes to be examined, and the other hyper-parameters to be explored in a timely manner. We sample the hyper-parameter space and generate a RDF with Adaboost 1100 times.

In Fig.3 we show the effect of the size of the randomly selected training sample on the cross validation mean \( \mu \), and standard deviation of the residuals \( \Delta z \), in four redshift ranges (see legend). The central lines show the mean values, and the shaded regions show the error on the mean. The large star symbols and thick error bars show the mean and standard deviation values of the residuals calculated using the SDSS template-ml redshift calculated on the same cross validation set. The large triangles show the smallest standard deviation of the machine learning method. We have positioned the triangles next to the stars to aid comparison.

**Figure 3**: The effect of the size of the training sample on the mean \( \mu \) value of the redshift residuals \( \Delta z \), calculated using Random Decision Trees and Adaboost. The sample is divided into four redshift ranges (see legend) and the mean, error on the mean and the standard deviation is calculated for the residuals in each bin. The central lines show the mean values, and the shaded regions show the error on the mean. The other hyper-parameters of the machine architecture are marginalized over. The large star symbols and thick error bars show the mean and standard deviation values of the residuals calculated using the SDSS template-ml redshift calculated on the same cross validation set. The large triangles show the smallest standard deviation of the machine learning method. We have positioned the triangles next to the stars to aid comparison.

**Fig.3** shows that the bias \( \mu_{\Delta z} \) on the cross validation sample decreases as the randomly selected training sample increases until the number is \( \approx 2000 \) after which it decreases less significantly and approaches a constant. The mean values are well within one standard deviation of the line \( \mu_{\Delta z} = 0 \). The dispersion of \( \mu_{\Delta z} \), at fixed training sample size is due to the random nature of the Random Decision Forest (RDF), and also the marginalization over the RDF hyper-parameters. At fixed training sample size, we find that the mean and standard deviation in each redshift bin is largely unaffected by our choices of machine architecture hyper-parameters. We find that including more than \( \sim 100,000 \) galaxies in the training set does not drastically improve the machine learning redshift when using the RDF method. Recall that the cross validation sample is not shown to the machine architecture while training, and therefore presents an unbiased estimate of the error.

### 4.4 The effect of the machine learning framework

Random Decision Forests (RDF) are constructed in §4.1 to measure feature importance, but they also provide a precise
measurement of the machine learning redshift, as previously seen by e.g., G erdes et al. (2010); Carrasco Kind & Brunner (2013). The hyper-parameter values of the RDF are the number of trees and the minimum number of training objects on each leaf node. We fix the size of the input features to be all 85 features seen in Table 1. We randomly explore the remaining hyper-parameter space. For each random instance of hyper-parameters the training sample (of size up to 961114) is used to train the RDF. For each instance the residuals are computed on the full cross validation sample (consisting of a different set of 480557 galaxies).

We choose the RDF with the lowest standard deviations as our final set of hyper-parameters, and finally measure the standard deviations on the residuals calculated using the test set (again, another different set of galaxies of size 480557). We reiterate that the cross validation set is not used during training, and the test data set is only used in this final stage. This results in an unbiased estimate of the error when applied to new data.

The top panel of Fig. 4 shows the distribution of the test sample residuals between the true redshift and both the SDSS template-ml redshift, and the RDF redshift. We additionally show this distribution scaled by 1/(1+\text{spec}_z) by the dotted lines, and mark the mean and standard deviation of the residual distribution for each case in the legend. The bottom panel of Fig. 4 shows a redshift scatter plot using 10,000 randomly selected galaxies from the test sample. We show the spectroscopic redshift against the SDSS template-ml redshift and against the RDF machine learning redshift.

Fig. 4 shows that the distribution of residuals using the RDF method (red lines) with 85 input features is more peaked around the true redshift, and declines faster as the residuals increase, than the distribution for the same sample of galaxies using the SDSS DR10 template-ml photometric redshift (blue lines).

We find that mean and standard deviation of the residuals using the RDF machine learning redshift applied to the test sample to be −0.001 and 0.061, and for the residuals scaled by 1/(1+\text{spec}_z) we measure −0.003 and 0.041. We next calculate the mean and standard deviation of the residuals on the test set using the SDSS template-ml photometric redshift and find them to be 0.002 and 0.086, and for the residuals scaled by 1/(1+\text{spec}_z), we measure −0.001 and 0.066. These values are also shown in Fig. 4. We note that the mean and standard deviations of the residuals of the RDF machine learning redshift are comparable, but slightly better than those measured using the artificial Neural Networks in Table 2. We reiterate that the RDF has used the full set of 85 input features, whereas the aNN were restricted to the standard, or top 1,2,3 most important features. A fuller analysis of the ability of an aNN to measure a machine learning redshift with a much larger input features is planned for future work.

We next restrict the input features of the RDF to be the ‘standard’ features defined above, we measure the mean and standard deviation of the residuals \Delta_z to be −0.002 and 0.075. This results in a slight improvement over the SDSS template-ml method, but is not as powerful as using the full 85 input features. We find that the outlier rate using the RDF method and the standard features is 1.74%.

To summarize, we find that the best fitting RDF machine learning redshift using the full 85 input feature list has a decrease in the standard deviation of 29% (38%) for the residual distribution (for the residual distribution scaled by 1/(1+\text{spec}_z) ) when compared to the SDSS DR10 template-ml photometric redshifts. Following Hildebrandt et al. (2010) we define the outlier rate as |\Delta_z/(1+\text{spec}_z)| > 0.15 and find that the outlier rate of the galaxies in the test sample using the SDSS DR10 template-ml redshift to be 2.32% and using the RDF redshift method the value is reduced to 0.99%, which is a 57% reduction. We note that the hyper-parameters values of the final RDF are Number of training examples: 707553, Minimum number of examples per leaf: 6, Number of trees in the forest: 55. We also note that the most important ranked 1,2,3 features of final RDF are fiberMag_g, fiberMag_i and the color dered_z-dered_g. We make the datasets and best trained RDF available on the homepage of the lead author upon journal acceptance.
5 CONCLUSIONS

To exploit large scale photometric surveys the identification of galaxies and measurement of their positions on the sky and in redshift space is paramount. Very accurate spectroscopic redshifts $z$, can only be measured on a small subset of galaxies due to the integration times required to obtain a reliable measurement.

One may determine a less accurate distance measurement using the photometric properties of the galaxy using either template methods, which encode our parametric knowledge of stellar populations and redshift expansion, or machine learning methods which are non-parametric. In machine learning methods, photometrically measured or derived properties, or ‘features’ are chosen and presented to the machine learning architecture in the hope that the input features identify a good scaling with redshift.

In this paper we present a study of the effect on the recovered machine learning redshift of the choice of input photometric features. We select and derive 85 easily obtained photometric features of all galaxies with spectroscopic redshifts found in SDSS DR10 CasJobs. We apply very light quality cuts on the recovered galaxies and obtain a sample of 1.9 million galaxies.

We use the machine learning architecture Random Decision Forests (RDF) with Adaboost (Breiman et al. 1984) Freund & Schapire 1997, Drucker 1997 learning to perform feature importance using the Gini criteria to determine which features produce the most predictive power for redshift estimation, and find that the SDSS $g$ band fiber magnitude is the top ranked, best single feature in 67% of cases. We list the top 1,2,3 features, and show their occurrence fractions after marginalizing over RDF hyper-parameters in Fig. 1. Adaboost and RDF have been used previously to estimate machine learning redshifts but only using standard magnitudes as input features (e.g. Gerdes et al. 2010).

We show how the addition of the top 1,2,3 importance ranked features can improve machine learning redshift estimates using the artificial Neural Networks PhotoRapter and FaNN. PhotoRapter allows a two layer deep network to be trained efficiently, and FaNN implements Cascade2 learning which sequentially adds (multiply connected) hidden neurons to the hidden layer.

We continue by fixing the PhotoRapter machine architecture and present it with the standard and then higher dimensional input features. We find that the recovered machine learning redshift is improved with the addition of important features. This is in agreement with Dagliaferri et al. (2003) who show that the addition of extra features such as radii and fluxes improve the machine learning redshift estimation compared with using just magnitudes alone. However, note that Singal et al. 2011 use more elaborate morphology features and find no real improvements to the machine learning redshifts. In this work we decide which additional features to present to the aNN using the feature importance selection.

We present the standard and top ranked 1,2 import features to FaNN and allow the machine architecture to vary. While the absolute results are not as competitive as with PhotoRapter, the use of the top 1,2 important features does improve the redshift estimate.

To quantify the improvement of the machine learning redshift we measure the residual between the true spectroscopic redshift and the machine learning redshift $\Delta z$. We measure the mean and the standard deviation of $\Delta z$ and the fraction of catastrophic “outliers” defined by $|\Delta z/(1+z)| > 0.15$. We find that the addition of the top ranked features to the standard features decreases the standard deviation of the residuals by 13%, and by 18% for the residual normalized by $1/(1+z)$, and that the outlier fraction is also decreased by 33%. We reiterate that this improvement is due to the addition of easily obtainable photometric features. When using FaNN we note that the standard deviation of the residuals decreases by $17 \pm 8\%$ after marginalizing over the aNN hyper-parameters parameters.

We then show how Random Decision Trees with Adaboost (RDF) can also determine a machine learning redshift (see also Gerdes et al. 2010 Carrasco Kind & Brunner 2013), and document that this technique uses less computational resources, decreases the standard deviation of the residuals and lowers the outlier fraction compared with the aNN architectures implemented here.

We quantify the improvement using RDF and all 85 input features, with respect to the SDSS DR10 photometric redshift available from CasJobs, and with respect to the two aNN architectures. We find that the standard deviation of the residuals distribution decreases by $\approx 29\%$, and by $\approx 38\%$ for the distribution scaled by $1/(1+z)$. We show that the outlier rate of the galaxies in the test sample using the SDSS DR10 photometric redshift is 2.32% and using the forest redshift method the value decreases to 0.99%.

We note that other machine learning architectures are readily available and the use of additional features within these frameworks is an ongoing project. Given the non parametric nature of the machine learning described here, we caution that the results of this analysis are not necessarily easily transported to other surveys or datasets. It would be prudent to perform a similar analysis with different surveys in order to identify their most salient features. However this problem is very tractable. Using Random Decision Trees and boosting routines, one is able to analyze a dataset of millions of galaxies in just a few hundred seconds using a single core machine.

ACKNOWLEDGMENTS

Funding for the SDSS and SDSS-II has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, the U.S. Department of Energy, the National Aeronautics and Space Administration, the Japanese Monbukagakusho, the Max Planck Society, and the Higher Education Funding Council for England. The SDSS Web Site is http://www.sdss.org/

REFERENCES

Abazajian K. N., Adelman-McCarthy J. K., Agüeros M. A., Allam S. S., Allende Prieto C., An D., Anderson K. S. J., Anderson S. F., Annis J., Bahcall N. A., et al. 2009, ApJS, 182, 543
