Predicting Photometric Redshifts using Machine Learning methods

ASTRON-3705 Class Project

Yasha Kaushal

PhD Candidate (III Year)

Department of Physics and Astronomy

University of Pittsburgh

Ali Beheshti

PhD Student (II Year)

Department of Physics and Astronomy

University of Pittsburgh

Jeffrey Newman

Professor

Department of Physics and Astronomy

University of Pittsburgh

Motivations

What is Photometric Redshift ?
Predicting redshift of a galaxy using its colors, magnitude and some other properties like (morphology, light profile, dust and axis ratio)

Next generation surveys like Euclid, LSST, SKA will rely on Photo-z rather than spectroscopic redshifts.

Expensive to obtain spectra (and then redshift from it) for *billions* of galaxies.

Machine Learning to rescue!
Already existing spectroscopic + photometric data of millions of galaxies can be leveraged to train an ML Model that could make predictions for new photometric data

Data

DESI

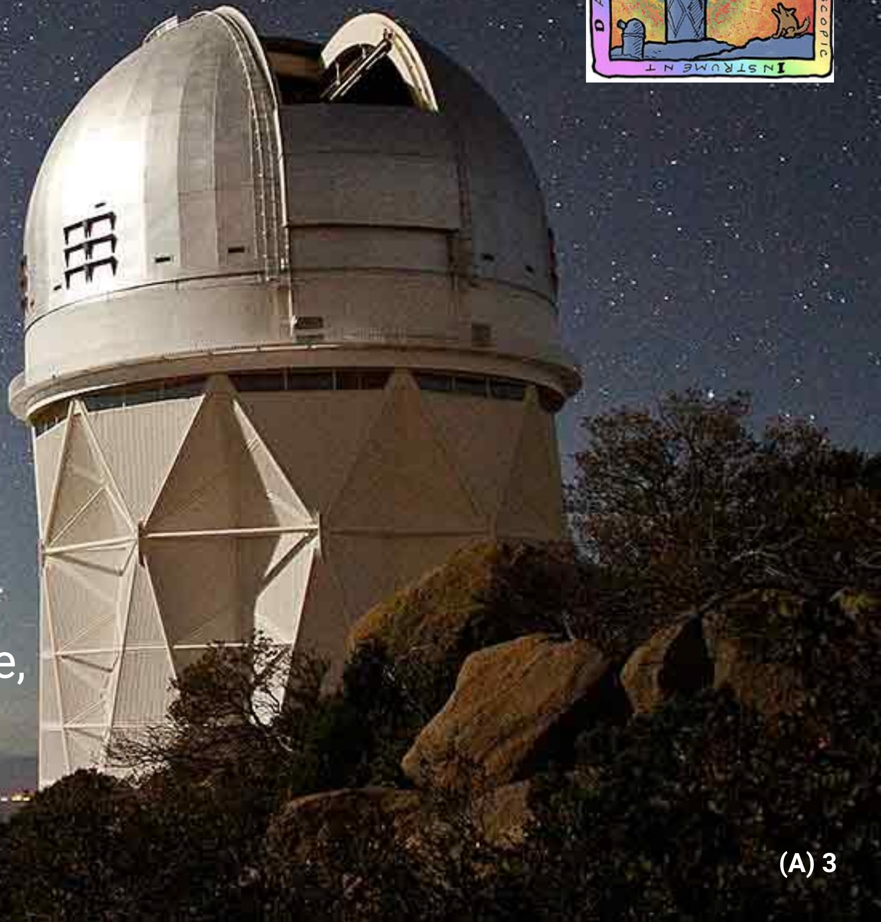
- ~14,000 deg^2
- 0<z<3.5
- >50 millions spectra

Legacy Surveys (Imaging)

- DECaLS, MzLS, BASS, WISE

BGS sample

- ~80,000 objects
- Z_{spec}, Magnitudes, MorphType, Shape, PhotSys, EBV, Sersic, etc.



Machine Learning

- **Regression**
- Classification
- Clustering
- Dimensionality reduction

- **Supervised**
- Unsupervised

- Random forest
- K-nearest neighbour
  - Unweighted (Uniform)
  - Weighted (Distance)
- Gradient Boosting
  - XgBoost
  - CatBoost
- Neural Networks
  - MLP Regressor
  - Keras based ANN
- Gaussian processes
  - KISS GP + LOVE sampling algorithm

Diagnostics for Analysis

1. Different training samples (keeping same test set)
   Scaling relations
   a. NMAD vs Training set size
   b. Outlier fraction vs Training set size

2. Different Features
   a. Colors + Magnitude only
   b. Colors + Magnitude + Half light radius
   c. Colors + Magnitude + Categorical
   d. ALL

3. Different Scalars
   a. Standard Scalar
   b. MinMax Scalar

4. Different Hyperparameters
   a. Layers
   b. Estimators
   c. Neighbours
   d. Depths
   e. Weights (uniform, distance etc.)

Metrics Used

1. NMAD (Normalized Median Absolute Deviation)

$$\sigma_{NMAD} = 1.48 \text{ x median } \frac{|z_{phot} - z_{spec}|}{1 + z_{spec}}$$

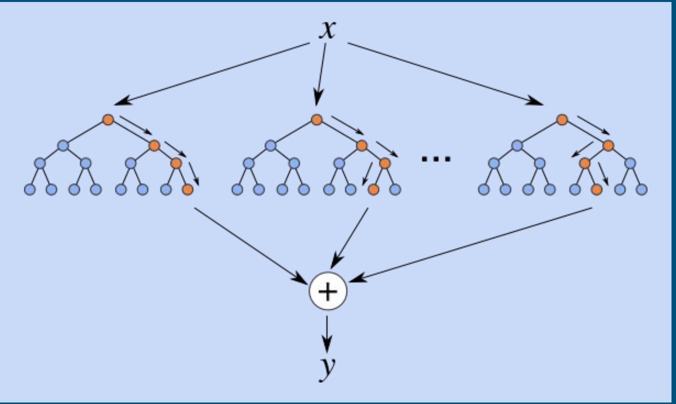
2. Bias $$\text{Bias} = \text{median } \frac{z_{phot} - z_{spec}}{1 + z_{spec}}$$

3. Outliers $$\frac{z_{phot} - z_{spec}}{1 + z_{spec}} > 0.15$$

4. RMSE (Root Mean Square Error)

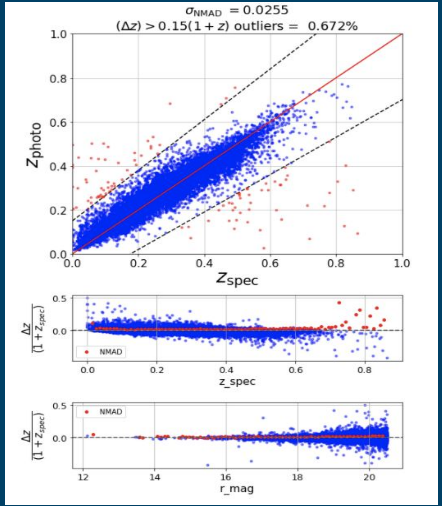
$$\sqrt{\frac{1}{n}\sum\left(\frac{z_{phot} - z_{spec}}{1+z_{spec}}\right)^2}$$

Random Forest



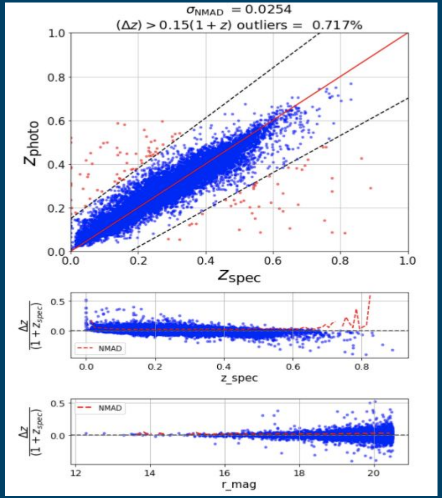
What is RF?

How it works?



MinMaxScalar

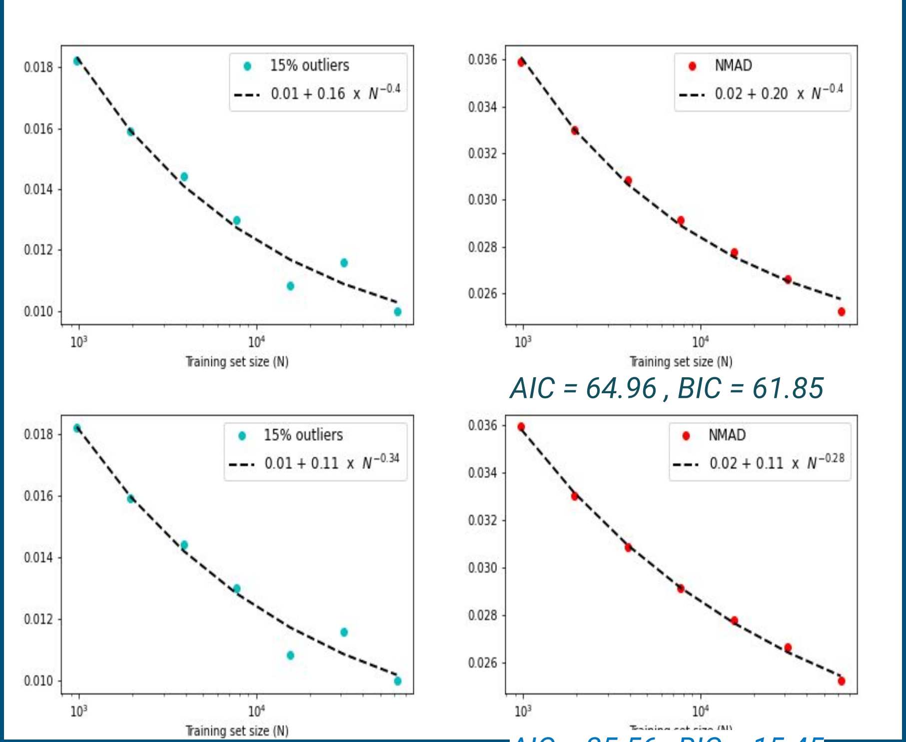
NMAD = 0.0255
Outliers = 0.672 %



StandardScalar

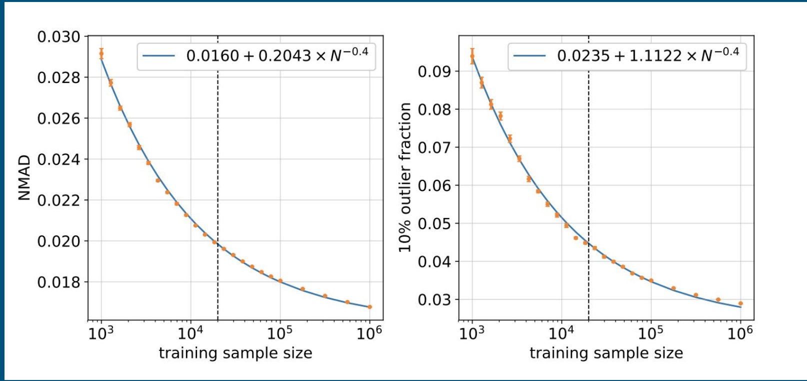
NMAD = 0.0254
Outliers = 0.717 %

Random Forest : What we found for DESI BGS sample?



AIC = 64.96 , BIC = 61.85

AIC = 25.56 , BIC = 15.45



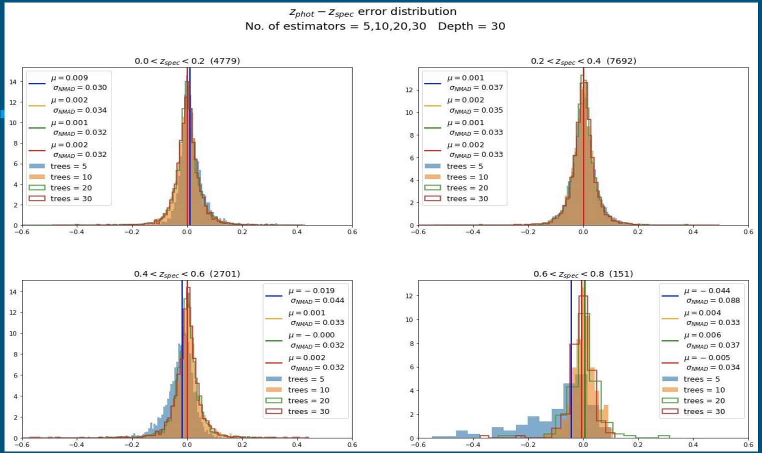
J. Newman + 2020 White Paper

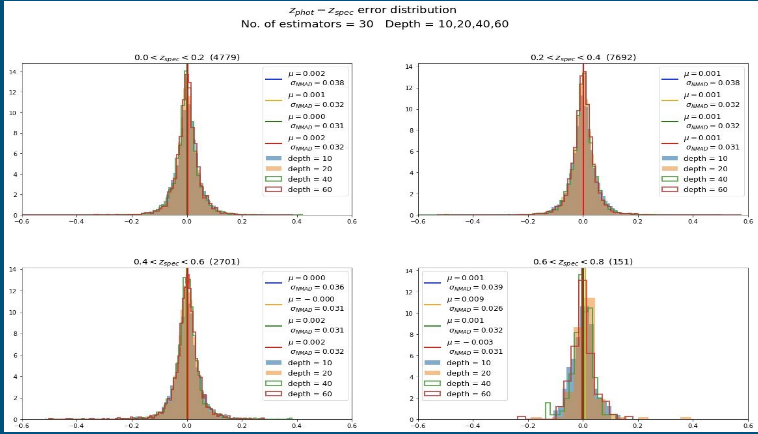
0.1 million simulated LSST galaxies
in 0 < z < 4

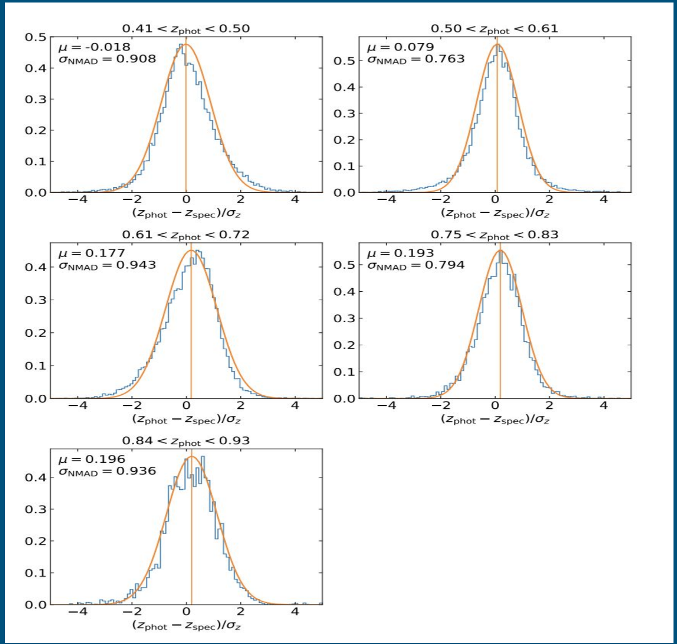
Table 1: Random Forest

	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0283	0.0271	0.0274	0.0250
15% Outliers	1.165 %	1.133 %	1.017 %	0.863 %

Random Forest : What we found for DESI BGS sample?

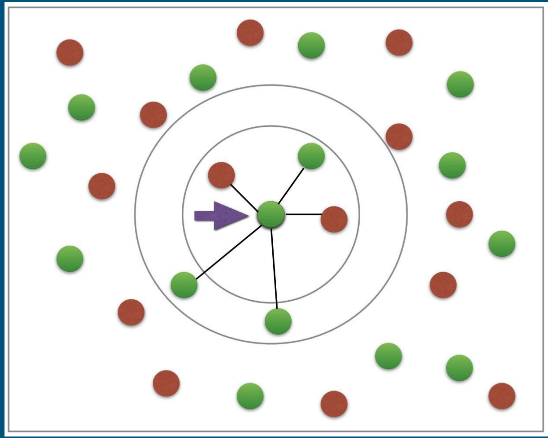






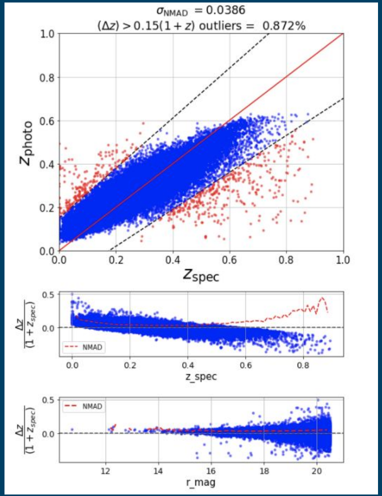
R. Zhou + 2021
2.7 million DESI LRG galaxies
in 0.4 < z < 0.9

KNN - Weighted and Unweighted



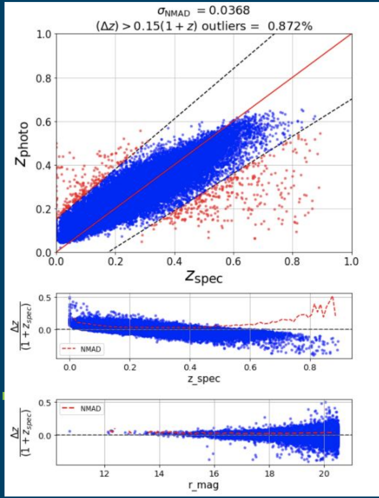
What is KNN?

How it works?



KNN Uniform

NMAD = 0.0386
Outliers = 0.872 %

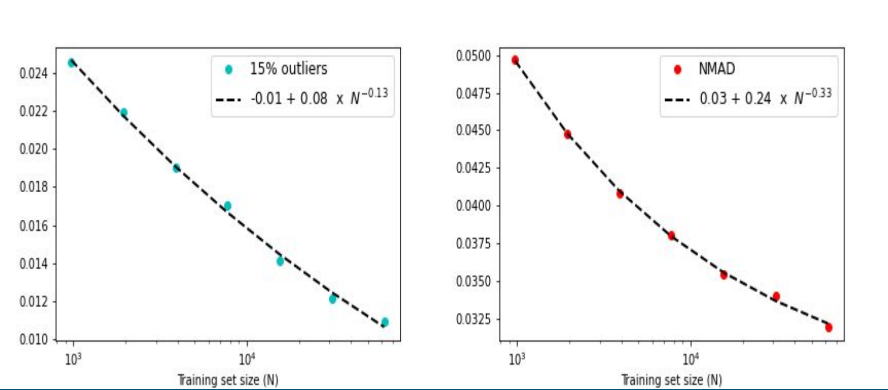


KNN Distance

NMAD = 0.0368
Outliers = 0.872%

KNN- unweighted vs weighted:

What we found for DESI BGS sample?



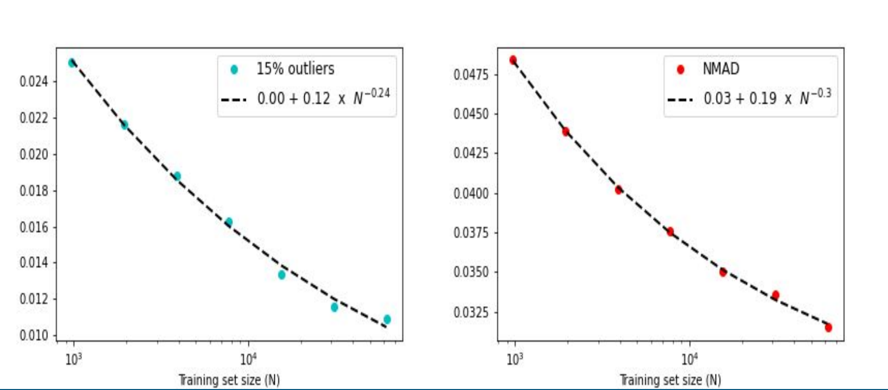


Table 2: Unweighted KNN

	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0289	0.0286	0.0296	0.0323
15% Outliers	1.088 %	0.998 %	1.165%	1.107 %

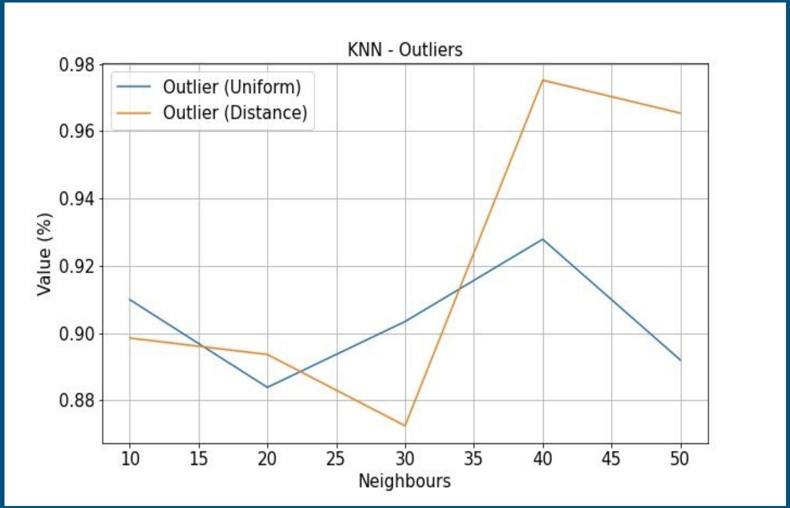
Table 3: Weighted KNN

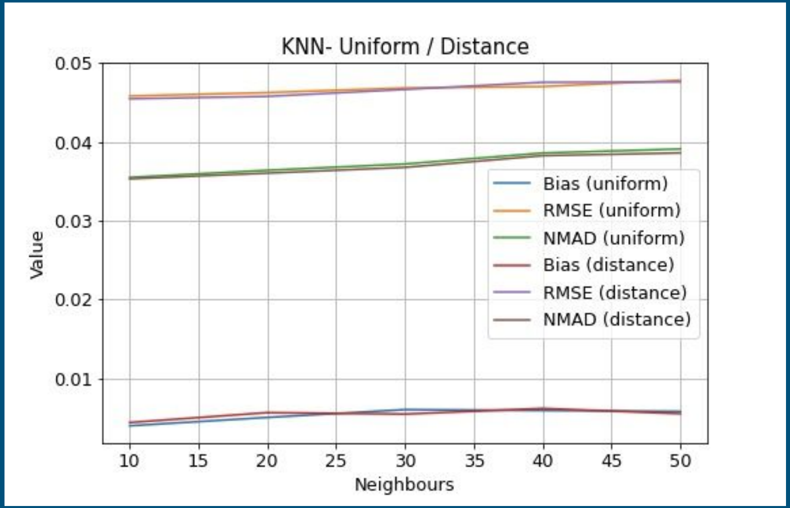
	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0286	0.0281	0.0290	0.0320
15% Outliers	1.101 %	0.991 %	1.191 %	1.088 %

80/20 train test split, 10 neighbours

Z. Gomes+2018 also found improved performance with size information in training

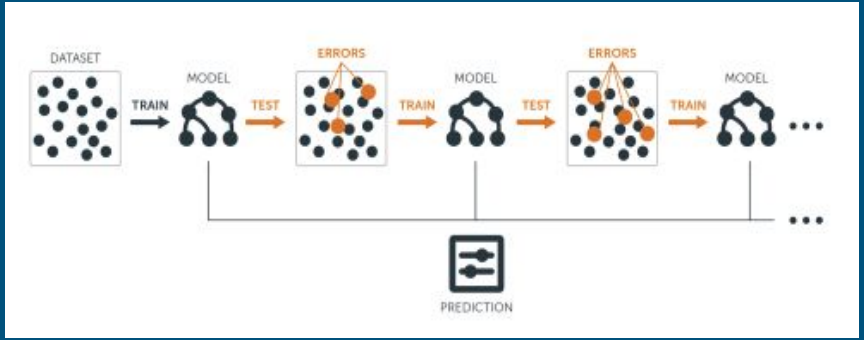
KNN- unweighted vs weighted:
What we found for DESI BGS sample?





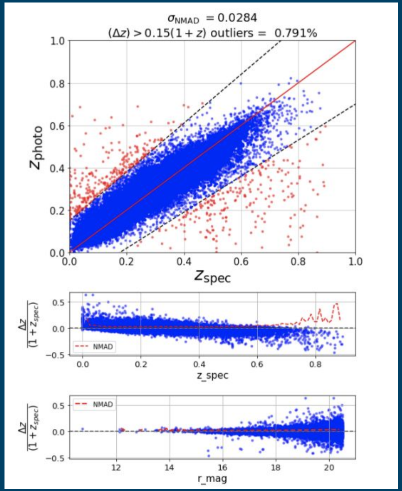
Least Outlier Fraction with :
N = 20 (Uniform)
N = 30 (Distance Weighting)

XGBoost
(Extreme Gradient Boosting)



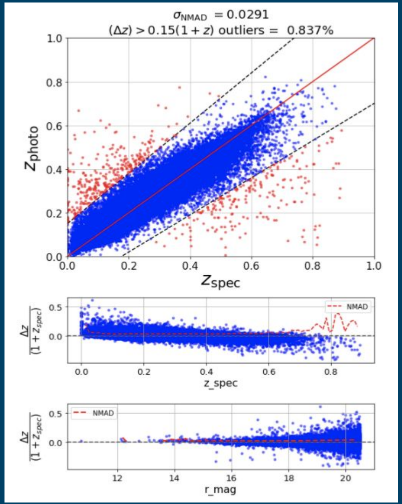
What is XGBoost?

How it works?



Squared Loss

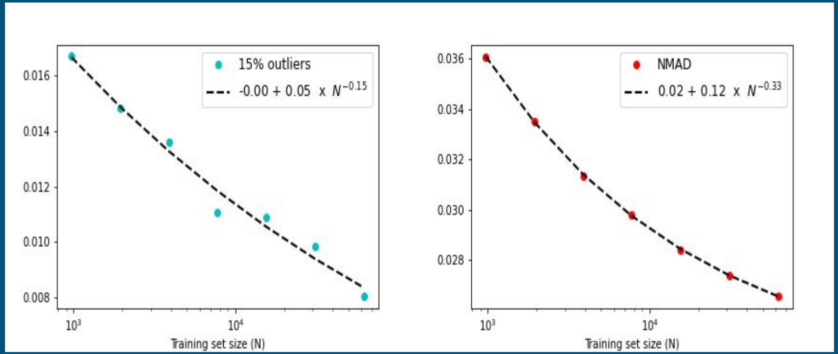
NMAD = 0.0284
Outliers = 0.791 %



Pseudo-Huber Loss

NMAD = 0.0368
Outliers = 0.872%

XGB : What we found for DESI BGS sample?



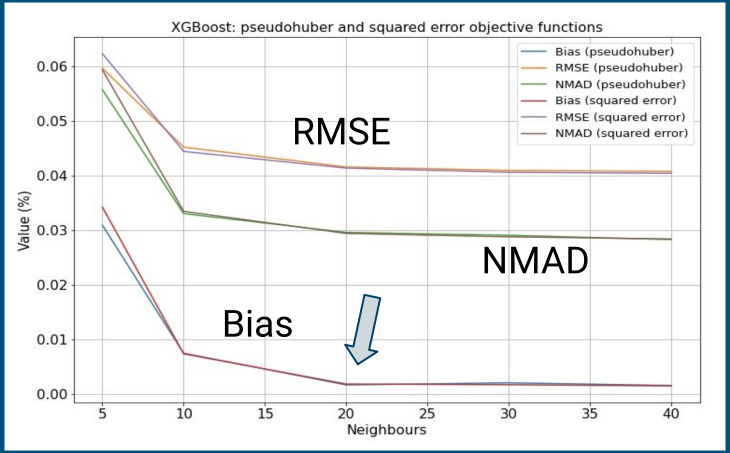
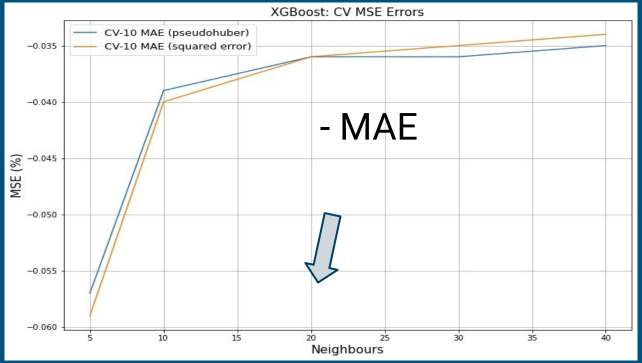


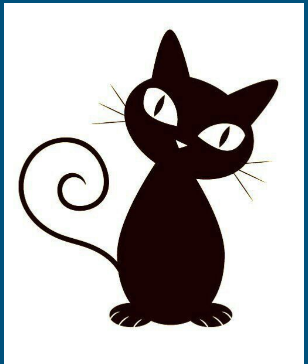
Table 4: XGB

	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0359	0.0353	0.0353	0.0328
15% Outliers	1.449 %	1.223 %	1.178 %	1.204 %



Optimal Neighbours = 20

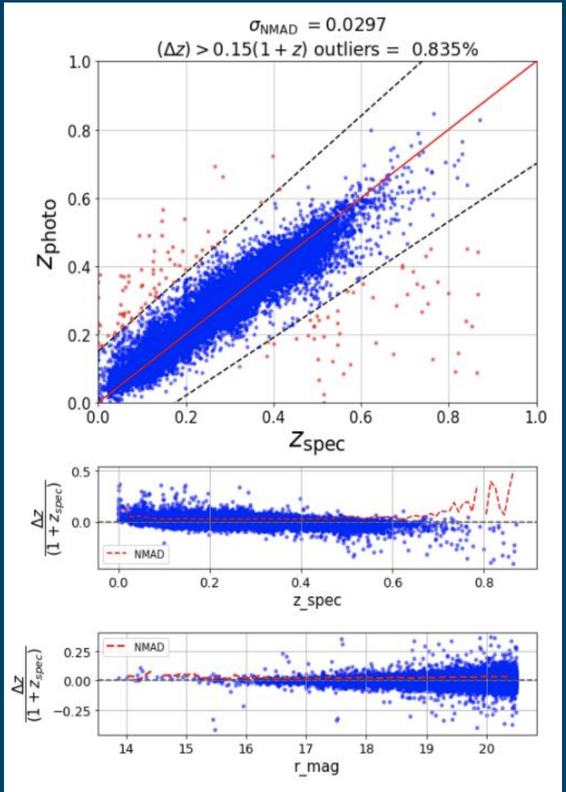
CatBoost
(Categorical Gradient Boosting)



What is CatBoost?

How it works?

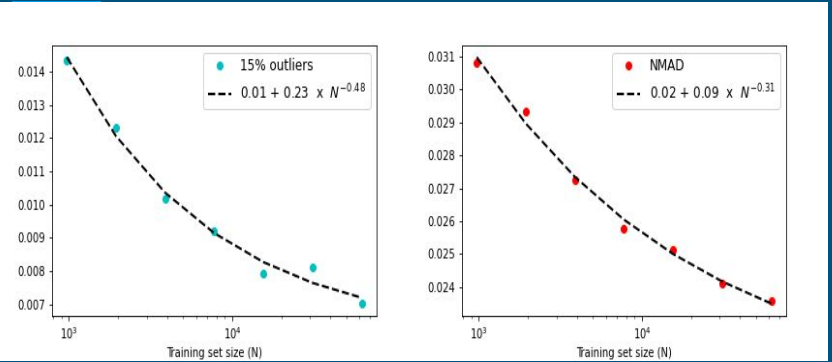
Estimators = 40
NMAD = 0.0297
Outliers = 0.835 %



Much Faster prediction than XGB but takes longer to train.

Utilizes Categorical information in decision trees

CATB : What we found for DESI BGS sample?



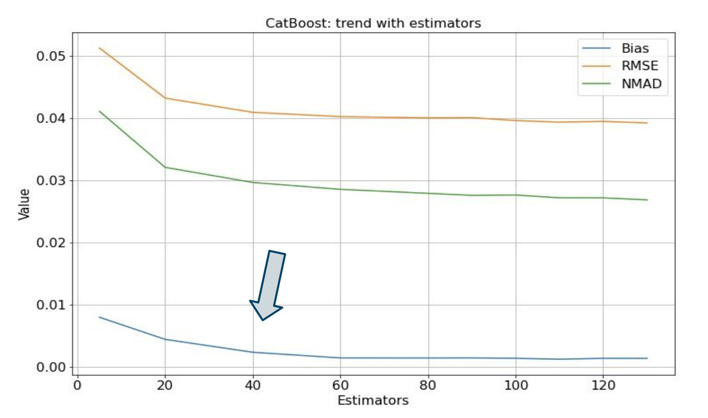
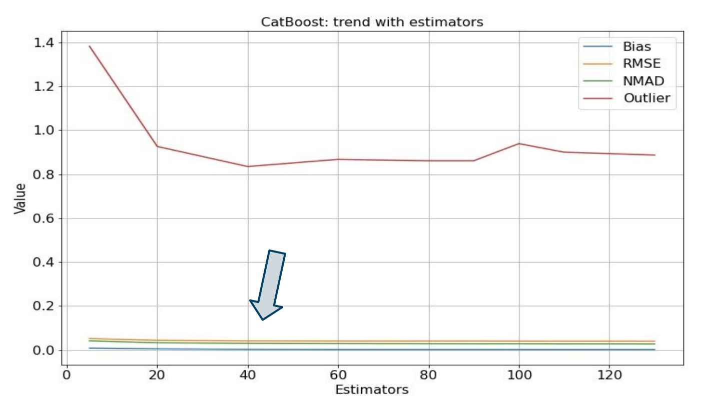


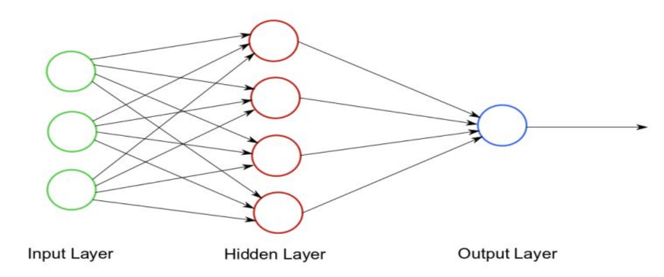
Table 6: CATB

	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0299	0.0290	0.0273	0.0248
15% Outliers	1.281 %	1.185 %	1.114 %	0.927 %



Optimal Estimators = 40
More Accurate than XGBoost

MLP NN
(Multi-Layer Perceptron Neural Network)

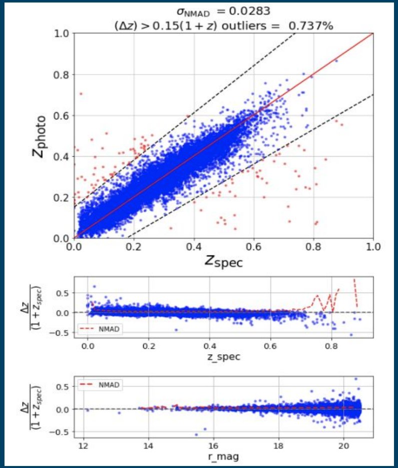


What is MLP?

Feedforward ANN
Can distinguish data that is not linearly separable (can learn a non-linear function approximator)

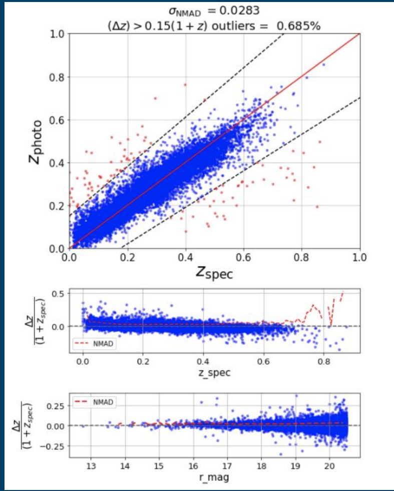
How it works?

Utilizes supervised learning technique called backpropagation for training



Random State = 2

NMAD = 0.0283
Outliers = 0.737 %



Max iterations = 50

NMAD = 0.0283
Outliers = 0.685 %

MLP : What we found for DESI BGS sample?

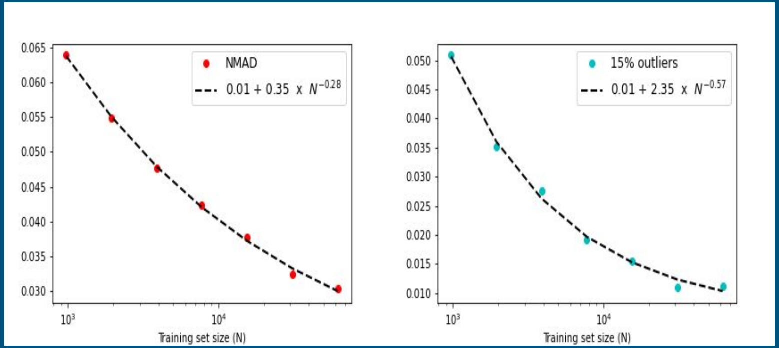
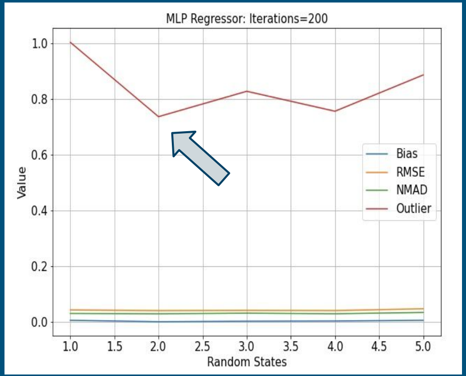


Table 5: MLP

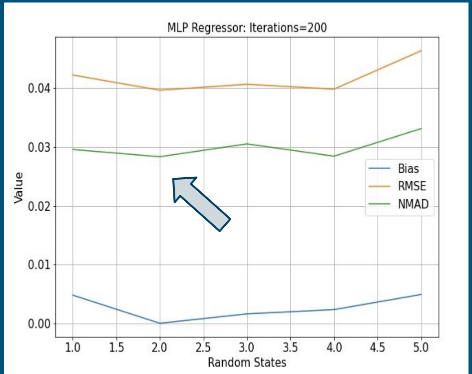
	Col. & Mag.	Col., Mag. & HLR	Col., Mag.'s & Cat.	All
NMAD	0.0329	0.0300	0.0319	0.0308
15% Outliers	1.185 %	1.140 %	0.933 %	1.062 %

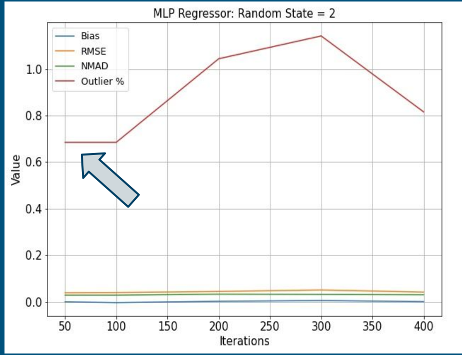
Optimal Iterations = 50

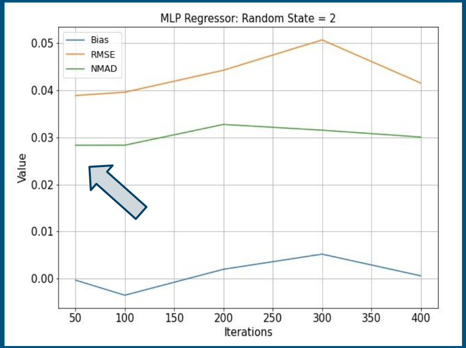
Outlier Fraction



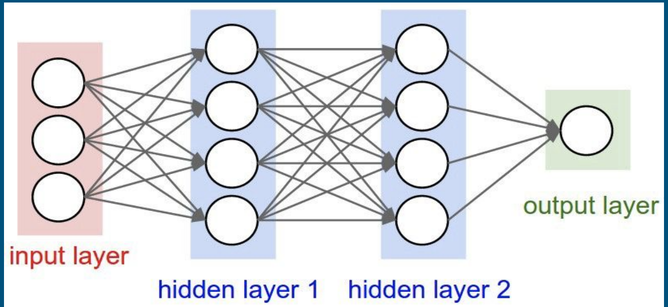
Performance





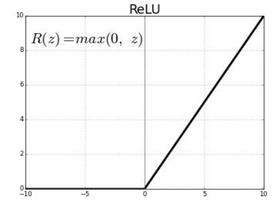


Keras NN



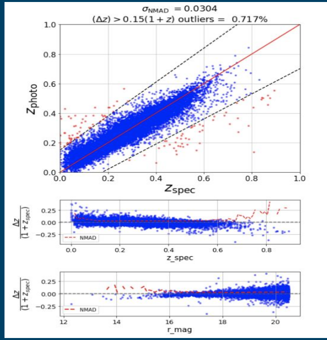
Why Keras NN?

High level API compared to Pytorch
and build over Tensorflow
Easier to Use and Implement
Cons : Can be slower than Pytorch



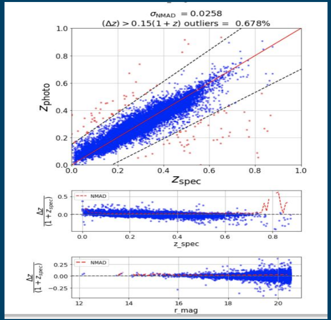
Our Test

2 Hidden Layers
Sequential Model
Rectified Linear Unit (Relu) Activation Function
L2 kernel regularizer



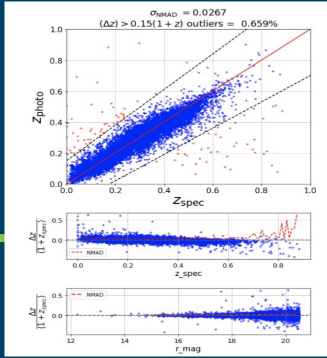
Batch Size = 200

NMAD = 0.0304
Outliers = 0.717 %



Epochs = 40

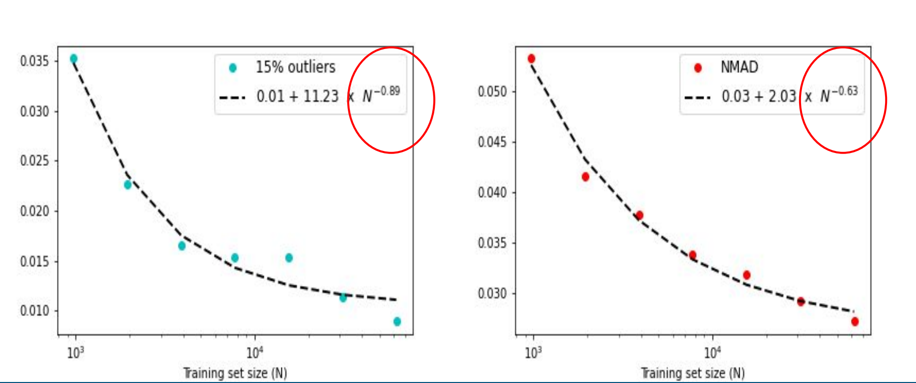
NMAD = 0.0258
Outliers = 0.678 %

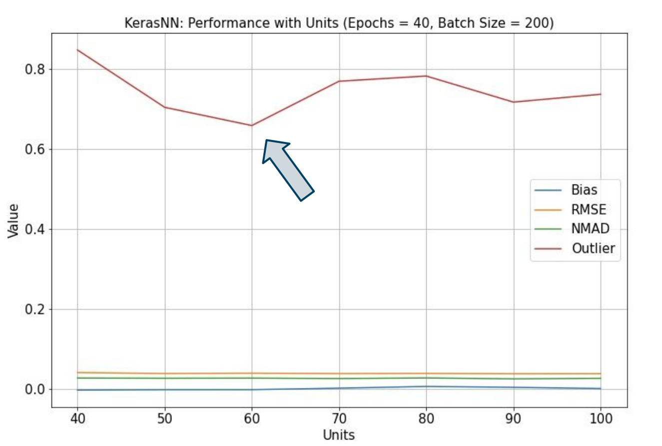


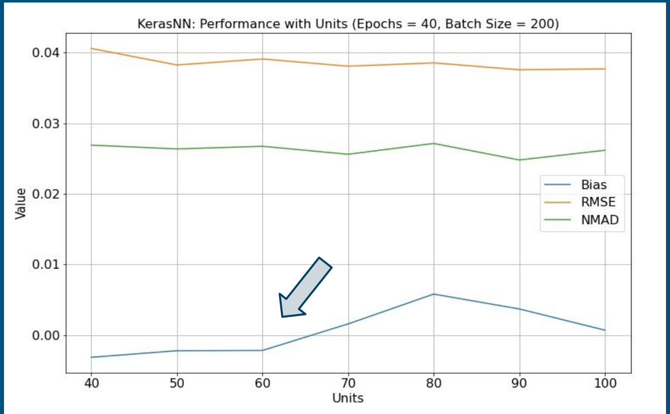
Units = 60

NMAD = 0.0267
Outliers = 0.659 %

KERAS : What we found for DESI BGS sample?

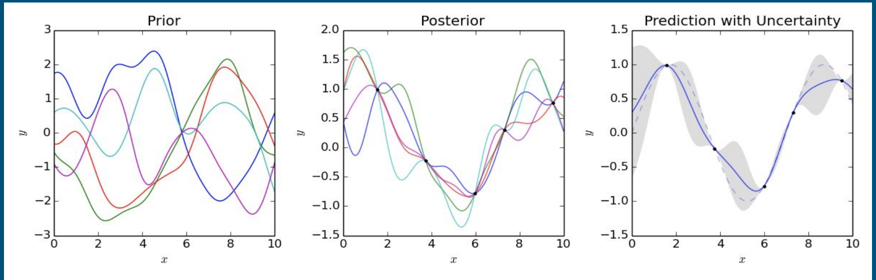






Optimal Parameters
Batch Size = 200
Epochs = 40
Units = 60

GPR
(Gaussian Process Regression)

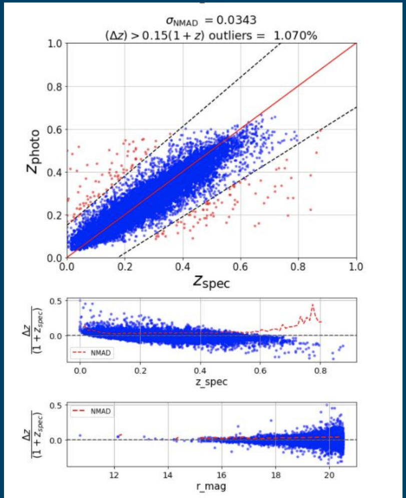


What is GPR?

Nonparametric, Non-linear Bayesian approach to regression that provides well-calibrated posterior distributions.

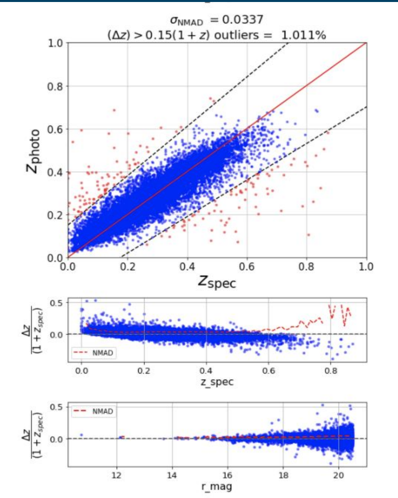
How it works?

Uses Kernels for calculating marginal likelihood + posterior mean (we used KISS)
Needs algorithms for fast posterior sampling and covariance matrix calculations (we used LOVE)



Grid = 100
Training iterations = 30

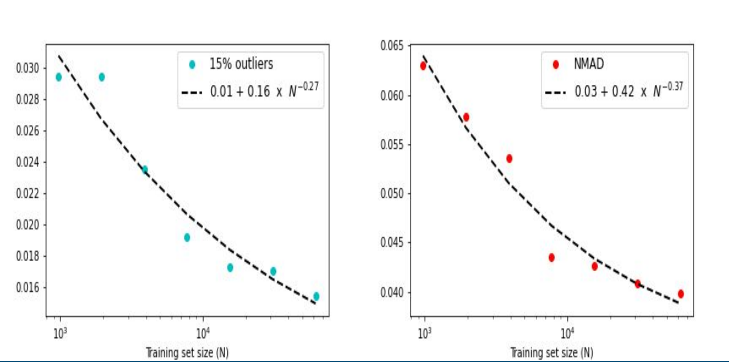
NMAD = 0.0343
Outliers = 1.070 %

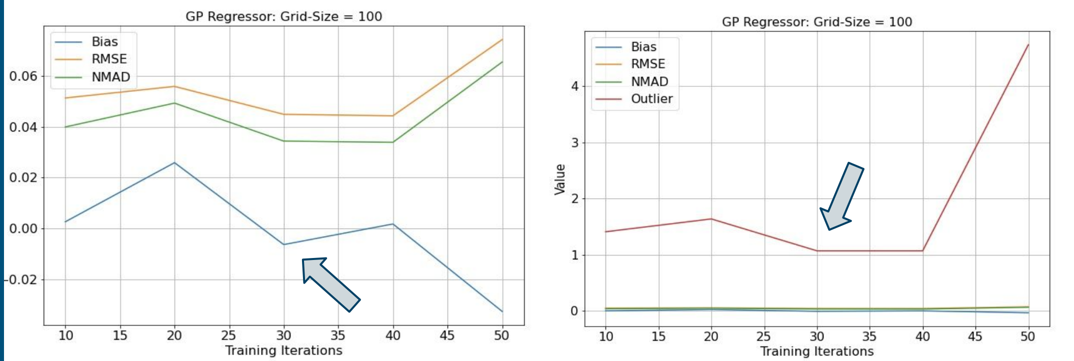


Grid = 200
Training iterations = 30

NMAD = 0.0337
Outliers = 1.011 %

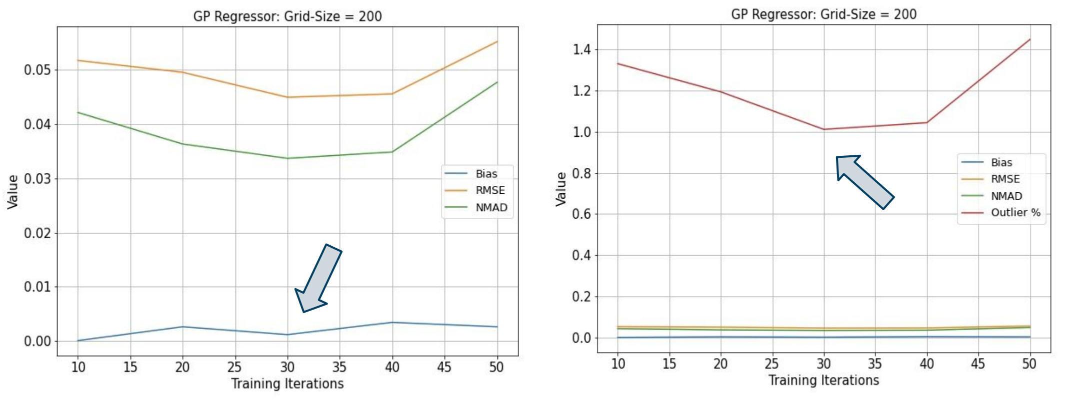
GPR : What we found for DESI BGS sample?





Optimal Params
Training Iterations = 30
Grid Size = 200

*Need to figure out reading covariance matrix to get confidence intervals



Conclusions

- **RF** & **CatBoost** performs best with ALL features included in Training Set.
- **KNN** (both) performed best with Colors + Mags + Half Light Radius Info.
- **XGB** has mixed results:
  NMAD -> ALL features
  Least Outlier % -> Colors + Mags + Categorical Info.
- **MLP** has mixed results.
  NMAD -> Color + Mag + HLR
  Least Outlier % -> Color + Mag + Categorical Info.
- **KERAS** performs best in scaling.

Future Work

- Compare our hyperparameters with scikit learn model optimization routines-
  1. Exhaustive Grid Search
  2. Randomized Parameter Optimization

  Also perform CV extensively for each parameter

- Re-run the scaling relations and different feature-set test with optimized model

- Quantify the computational efficiency of each method (time, GPUs, complexity)

- Obtain confidence intervals for GPR and test that with different kernels.

THANK YOU!