MACHINE LEARNING IN NONLINEAR STRUCTURE FORMATION

A PREPRINT

Parimah Safarian
Department of Physics
Sharif University of Technology
parimahsafarian@gmail.com

Saba Etezad Razavi
Department of Physics
Sharif University of Technology
s_etezadrazavi@yahoo.com

Erfan Abbasgholinejad
Department of Physics
Sharif University of Technology
erfan77agn@gmail.com

August 15, 2020

ABSTRACT

Dark matter halos have received great attention because of their important role in cosmological structure formation. Methods already used for analysing the evolution of over-densities to form structures are model based and semi-analytical. Despite their successful results they have some problems. In this work we introduce a new density parameter for the study of collapse of over-densities. We trained number of machine learning models to learn structure formation from N-body simulations without no further assumption about the physical processes. We found a relevant scale for each halo mass directly from our trained algorithm and we are able to compare our new scale with ones that were using in previous theories like Press-Schechter formalism. Our model learns to predict whether or not dark matter particles will end up in halos of a given mass range. We have tried to make a better accuracy and also checked the results of our predictions from variety points of view. finally we describe an algorithm to find the halo mass function of dark matter halos and a moving barrier for the collapse which is an important step toward the study of density growth function.

***K*eywords** Cosmic Structures · Structure Formation · Machine Learning · Dark Matter · Halos · Primordial Fluctuations

1 Introduction

The population of dark matter halos and their formation initiated from early universe density fluctuations. They have received attention because of their essential role in cosmological structure formation, specifically galaxies and galactic clusters. These halos act as a potential well for capturing baryonic matter to form massive structures. Furthermore, knowing dark matter halos is an essential step toward testing our cosmological models. Due to the high non-linearity of the halo dynamics, it is a difficult problem to deal with it analytically, and so far, N-body simulations provide the only practical tool to study nonlinear gravitational effects. Analytical approximations also provided an excellent physical interpretation for understanding the halo formation, based on the works of Press and Schechter and Excursion set theory ansatz they provide an analytical tool to predict the halo mass function from an initial condition. Nevertheless, although these models are qualitatively correct, they have some problems with observations and N-body simulations, which we will discuss more genuinely in 3. The high non-linearity of the formation of dark matter halos make it a problem well suited to be treated with machine learning methods. Machine learning is a highly efficient and powerful tool to learn relationships that are too complex for standard statistical techniques (Witten et al. 2016)[1]. As machine learning algorithms are quite sufficient to extract useful information from a wide range of initial conditions properties, they caught the interest of people for treating many analytically complex problems, including dark matter halo formation. Just as previous works in the subject [2] [3], we choose our properties to be aspects of the initial density field, which we believe can capture all the needed information about the gravitational effects. Also, people studied the effect of the tidal shear field on the halo formation [2], which turns out that it has quite low impact. The main focus of these previous works were reducing the computational load of non-linearity studies . They trained initial conditions-to-halo mapping, can be used to predict the mapping for new initial conditions without the need to run a further simulation. Also, they checked the impact of many initial properties on the halo formation to find the most relevant one (which is the density field). Their method had no prediction about the halo mass function and physical process of collapse and concentrated only on one halo mass range. Using their result about the most relevancy of the density field in this work, we provide many initial conditions-to-halo mappings for

different mass ranges, which makes us able to find a halo mass function as well. Our work based on a bigger simulation box than before and a deeper snapshot. Also, we investigated much more machine learning models and applied the method for sufficiently large mass ranges. We introduced a new mass scale and density parameter for studies of structure formation, model independently. We used our method to find a moving collapse threshold, which can be used to evaluating the density growth rate function and encoding many nonlinear features of collapse in terms of this function.

The organization of the paper is as follows: in Sec. 2 we described our used data. Sec. 3 allocated mainly to the theoretical aspects of our work and introduction of new scales used, the definition of a moving barrier, and extracting features from density field. In Sec. 4, we focused on our machine learning method and the results of many different models. The final accuracy of all the models does not exceed 85%, so in Sec. 5 we focused on the error data of our models and analyzed them. Sec. 6 and 7 devoted to describing our algorithm for finding a moving barrier and halo mass function. And finally conclusion and future aspects of our work are in Sec. 8

2 Data

For our data we needed dark matter only particle data with specific labels which able us to trace back particles in time and see their journey from the initial condition to $z = 0$. Fortunately Virgo-Millenium database provide us such information. The simulation was carried out using a modified version of the publicly available code "GADGET-2" (Springel 2005) [4]. The algorithm "SUBFIND" (Springel et al. 2 2001)[5] was used to identify all self-bound halos containing at least 20 particles and all self-bound subhalos within these halos down to the same mass limit. The method used to identify halos in SUBFIND algorithm is FoF (friends of friends).

A millennium run contains 10^{10} particles each with mass $M_* = 8.6\ 10^8 M_o$ within a cubic region of sides $500h^{-1}Mpc$. It follows the evolution of dark matter particles from redshift $z = 127$ (snap number = 67) until $z = 0$ (snap number = 0). The Benchmark model [6] is assumed for this simulation which uses cosmological parameters equal to $\Omega_m = \Omega_{DM} + \Omega_b = 0.25$,$\Omega_b = 0.045$, $\Omega_\Lambda = 0.75$, $h = 0.73$, $\sigma_8 = 0.9$ and $n_s = 1$ (power law power spectrum), with standard definitions for all quantities. The hole millennium run contains of a data about 1TB, in order to make a fair balance between accuracy and run-time, and using the assumption that the distribution of initial fluctuations is Gaussian, we focused only on the "millimil" box of the hole run. A "millimil" box contains of 20 million particles within a cubic region of sides $60h^{-1}Mpc$ and a resolution of $5kpc$. We collected 4GB pure data from 3 different snapshots, ending snapshot $z = 0$, a middle-way snapshot z=16.7 and the initial snapshot $z = 127$. We produced 20 GB data and features usable for our model from it. The data included positions and velocities in every directions for each particle at the 3 snapshots, particle IDs which is unique through out the simulation, merger history for each particle, and the mass and ID of the host halo (if any) for each particle in $z = 0$. With this data we have the ability to follow each particle from the initial condition into the halos. We define halos as object with more than 1600 bounded particles. The simulation contains 706 halos with mass ranging from 1600 to 140000 particles at $z = 0$. You can see the histogram of halo masses in Fig1 also the halo mass distribution of our box can be found in $z = 127$. 2

We used the final snapshot $z = 0$ to label each particle in it's corresponding class. For any halo mass M_h we split the dark matter particles into two classes IN and OUT. class IN is made of particles which will end up in a halo with a mass larger than M_h at $z = 0$ and class OUT is corresponding to the particles in halos with mass smaller than M_h at $z = 0$ and those that do not belong to any halo (all the remaining particles). We traced each particle with it's associated class label to the initial time $z = 127$ to extract relevant features for each of them, there.

3 Density Field

3.1 Physical interpretation

Study of structures in the cosmos mainly devoted to the study of the evolution of density fluctuations. From the linear theory, one can see that the over-density filed grows as $\delta_{(x,t)} \propto D_{(t)}$ with $D_{(t)}$ the linear growth rate. But in the nonlinear regime (which is our case of interest), using the linear growth rate is no longer appropriate. Although even in this regime, the distribution of structures is determined by the initial density filed, so the study of the over-density field can still play an important rule in our understanding of the formation of structures.

An object of mass M in our current universe forms from an over-dense region with volume $M/ < \rho >$ in the initial density field. Such a region will correspond to a peak in the initial density field and a potential well in the gravitational potential field. Such a well will cause all the low-velocity particles in its local neighborhood to come together and form a structure.

For the density contrast at any point of our box we have: $\delta_{(\mathbf{x})} = \frac{\rho_{(t)} - <\rho>}{<\rho>}$ where $< \rho >$ is the background density. We need to smooth our density field; for this purpose, we use a Top-hat window function defined as below, which plays a crucial rule in our future studies, Real space top-hat window function:

$$W_{th}(\mathbf{x}; R) = \begin{cases} \frac{3}{4\pi} R^3, & \text{if } |x| \leq R \\ \\ 0, & |x| > R \end{cases} \tag{1}$$

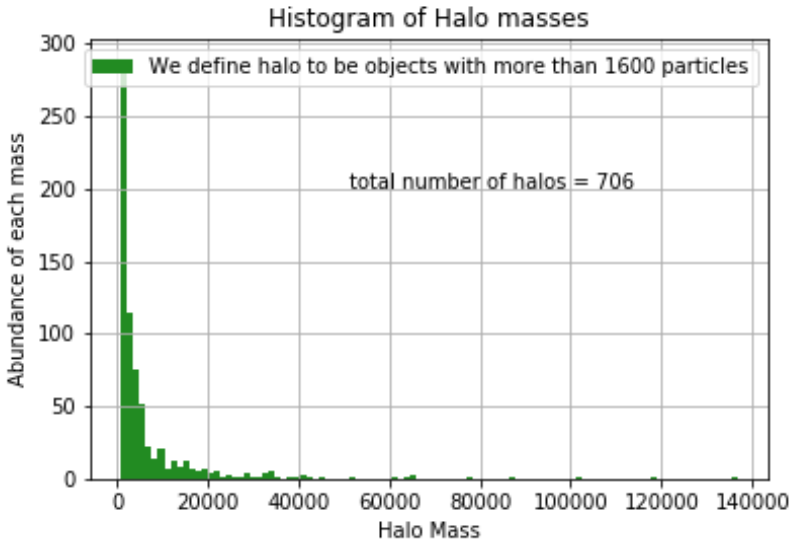
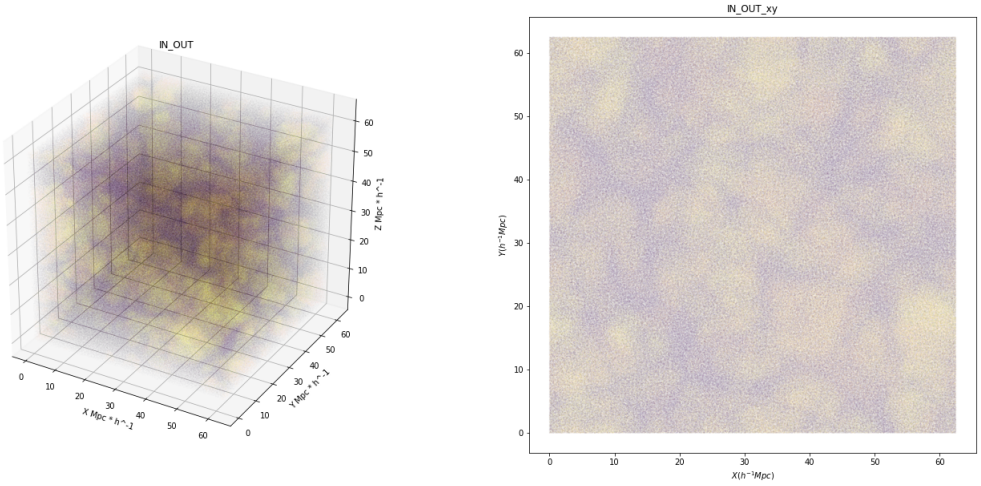


Figure 1: Histogram of halo masses at $z = 0$. x-axis is the halo mass times M_* (particle masses) and y-axis shows the number of halos with this mass



(a) 3D distribution

(b) 2D distribution in x-y plane

Figure 2: Distribution of particles in a "millimil" box of simulation at $z = 127$. Yellow corresponds to the particles which end up to a halo with mass larger than $M_h = 1600M_*$ and purple correspond to remaining.

where R is the smoothing scale of the window function. And for the smoothed over-density field we will have:

$$\delta_s(\mathbf{x}; R) = \int \delta(x') W_{th}(x + x'; R) d^3x' \tag{2}$$

Technically smoothing density field with a top-hat window function and smoothing scale R, means finding the number density of particles in a sphere with radius R around each point of interest in the box. By doing so for all the points in our simulation box, we can create the density field with the smoothing scale R.

As we say before, from the standard theory of structure formation, we believe that halos will form at the density peaks, so the study of them can shed light on the number density and spatial distribution of collapsed dark matter halos.

The field $\delta(x)$ generally contains many peaks (local maxima) and valleys (local minima), namely the over-dense and under-dense regions. Spatial distribution of peaks formally can be written as:

$$n_{peak} = \sum_p \delta^D(x - x_p) \tag{3}$$

Where x_p is the position of peaks. Peaks are simply points in the density filed where its gradient is zero, $\delta(x_p) = 0$. n_{peak} Must be tightly coupled to the spatial distribution of halos in our universe. A well-trained machine learning model should be able to extract the peaks from a density field.

In 1974 Press and Schechter developed a method[7] (PS) for finding the halo mass function of the universe analytically. Their ansatz is that a patch will collapse to form a halo of mass M at redshift z if its linear density contrast exceeds a critical value $\delta_c(z)$.

Consider the over-density field evolving in the linear regime

$$\delta(x, t) = \delta_0(x) D(t) \tag{4}$$

$D(t)$, is the linear growth rate normalized to unity at the present universe. According to the spherical collapse model, the condition for the collapse of any over-density at time t, $\delta(x, t)$, to form a Virialized object is that its value exceeds a critical value computed analytically to be :

$$\delta(x, t) > \delta_c = 1.69$$

Equivalently for an over-density in the initial density field to collapse at a later time t we need that:

$$\delta(x, t) = \delta_0(x) D(t) > \delta_c \rightarrow \delta_0(x) > \frac{\delta_c}{D(t)} \equiv \delta_c(t) \tag{5}$$

So the height of over-densities which will collapse at different times are different from each other, this will correspond to varying masses for halos that were not made at the same time. In hierarchical models, $D(t)$ increases with t and $\sigma(M)$ decreases with M, the characteristic mass increases with time. Thus, as time passes, more and more massive halos will start to form. [8]

What Press and Schechter have done where assigning mass to these collapsing regions. They assumed that the probability that $\delta_s > \delta_c(t)$, where δ_s is the smoothed initial over-density field, is the same as the fraction of mass elements that at time t are contained in a halo with mass greater than M. M is the mass corresponding to the smoothing scale used, R. for top-hat window function it's equal to $3/4\pi < \rho > R^3$. If the density fluctuations are made by an inflationary field, we expect them to obey a Gaussian random distribution. If $\delta_0(x)$ is a Gaussian random field then so is $\delta_s(x)$ and the probability that $\delta_s(x) > \delta_c(t)$ is given by:

$$\begin{aligned} \wp[> \delta_c(t)] &= \frac{1}{\sqrt{2\pi}\sigma(M)} \int_{\delta_c(t)}^{\infty} exp[-\frac{\delta_s^2}{2\sigma^2(M)}] d\delta_s \\ &= \frac{1}{2}\text{erfc}[\frac{\delta_c^2}{\sqrt{2}\sigma(M)}] \end{aligned}$$

And

$$\begin{aligned} \sigma^2(M) &= < \delta_s^2(x; R) > \\ &= \frac{1}{2\pi^2} \int_0^{\infty} P(k) \widetilde{W}^2(kR) k^2 dk \end{aligned} \tag{6}$$

Is the mass variance of the smoothed density field with P(k) the power spectrum of the density perturbations, and the window function written in Fourier space $\widetilde{W}$.

According to Press-Schechter ansatz, the mass fraction of collapsed objects with a mass greater than $M, F(> M)$ is equal to the probability of eq.6. From $F(> M)$ one can easily derive the number density of collapse objects with a mass between M and $M + dM$, the Press-Schechter mass function:

$$\begin{aligned} n(M, t) dM &= \frac{< \rho >}{M} \frac{\partial F(> M)}{\partial M} dM \\ &= \frac{< \rho >}{M^2} f_{ps}(\nu) \left| \frac{dln\nu}{dlnM} \right| dM \end{aligned}$$

Where $\nu = \delta_c(t)/\sigma(M)$ and

$$f_{ps}(\nu) = \sqrt{\frac{2}{\pi}} \nu \, exp(-\frac{\nu^2}{2}) \tag{7}$$

Regardless of the simplicity and beauty of the formalism, Press-Schechter mass function, when tested on simulations, face two problems, it under-estimate the number of most-massive halos in the universe and over-estimate

the abundance of less-massive ones. An alternative to the Press-Schechter mass function is Extended Press Schechter mass function with use of Excursion set formalism(EST) obtained by Bond et al (1991)[9]. In this formalism, we use a shorthand notation $S = \sigma^2(M)$, where M is the mass corresponding to the smoothing scale. Now for every mass element of the box, we can find density contrast corresponding to any smoothing scale, which will be captured in S now. Plotting δ_s versus S will lead to a "trajectory" for each mass element, the largest smoothing scale (first) where one of these trajectories cross the $\delta_s = \delta_c$ line is the collapse scale of the mass element. It's important to know that the EST method gives the collapsing scale only if we look at the mass element at the center of mass of an over-dense region; otherwise, it is just a lower limit.

Motivated by the Press-Schechter formalism our task is to find δ_c as a function of the halo mass which will give us the $\delta_c(t)$ and the halo mass function from a completely model-independent analysis on a simulation box. The advantage of our model in comparison with previous works is that we need no assumption about the collapse model as well as growth function $D(t)$. Physical processes of halo formation and the nonlinear dynamics are learned directly from the N-body simulation rather than fixing a threshold δ_c and employing the excursion set method. We can find the threshold $\delta_c(t)$ without any approximation as an output of our model, comparing our model-independent driven thresholds and halo mass function with analytical model results can illuminate our understanding of cosmic structures formation and evolution.

We provided a Machine learning model with a set of most essential features of dark matter particles in the initial condition, namely density and velocity field. In the absence of isocurvature sources (employing a simple inflationary model for initiating the density fluctuations), We expect them to be the complete set of variables for our study. As can be seen in both PS and EST methods smoothing scale is an essential parameter for our studies and is a measure of the halo masses. The crucial question is, what smoothing scale should we use to see the critical characteristics of our box?

3.2 Feature extraction

We found the smoothed density field for number of smoothing scales; the appropriate scale is learned directly by the model. We found that increasing the number of scales from 10 to 20 does nothing better to our training process, so the order of 10 number of scales that we used, seems sufficient to capture relevant information carried by the density field. We saw that the density contrast goes to zero as the smoothing scale exceed about $4Mpc$ (density become close to the background density), and it goes to infinity with a r^-3 functionality as the radius decrease and the number of particles per volume became constant and equal to unity. So we believe the proper scale is in a range between 1-5 Mpc. It seems reasonable as our less massive halos have a mass of about $1600M_*$ (M_* is the mass of the particles in the simulation), and the corresponding radius for a sphere with this mass and background density is around 1.7 Mpc. It Means the smallest scale that we need for our over-densities to form the halos is in this order. For each particle of simulation, we repeated smoothing for 25 evenly spaced radius scales between 1 to 5 and also used their velocity magnitude as another feature. As we said before, in the context of excursion set theory, the density contrast of a particle as a function of the smoothing scale is called "density trajectory" of the particle. Trajectories give us a proper scheme of whether a particle is found in an under-dense region or an over-dense one; as one approaches the largest mass scales probed by the simulation box, the trajectories start to converge to $\delta_s(x, \infty) = 0$. The ensemble of trajectories constitutes the full feature set we used to train our models. Fig. 3

One can suspect that maybe there is more information in the density of each mass element in a specific direction of the particle. The particle can get attracted to the direction of the densest region around it. So we also made a set of features with 8 distinct directions around each particle and calculated the density contrasts in each of these directions, the number of directional density features for five smoothing scales will be about 40 per particle, but as our stated results in the proceeding sections, directional density features are not much of help in the training process.

3.3 Density classification

The knowledge of the type of over-densities and under-densities is essential for our study. We should classify our density field in two kinds, density related to the regions which end up in a halo with a mass larger than M_h and density related to the regions which are not. It has quite a certain connection with the size of over-densities and under-densities.

We classified our density field by the future they make, and we introduced a critical density δ_c for each halo mass M_h which is the density at "the most important smoothing scale for the training process" of classifying particles in two sets of IN and OUT. The difference of our definition with previous ones is the scale where we compute density classifier measure, δ_c. in the classical accepted models (PS, EST, ...) δ_c is the critical density corresponding to a fixed scale for any M_h which is $(\frac{3}{4\pi}\frac{M_h}{<\rho>})^{1/3}$. There is an assumption for this choice of scale, which states that "all" the particles in the desired region will collapse to form the halo. For model-independency, we decided to import no further assumption to our study; hence we let the machine choose the most appropriate scale itself.

In the EST formalism, we proposed that the method of looking at the size of the over-densities and predicting whether the mass element will end up into a halo or not, only works for the center of masses and peaks in the density field; otherwise, it will only give us a statistical description of particle's future and does not have useful information about each individual. We found out that our model can

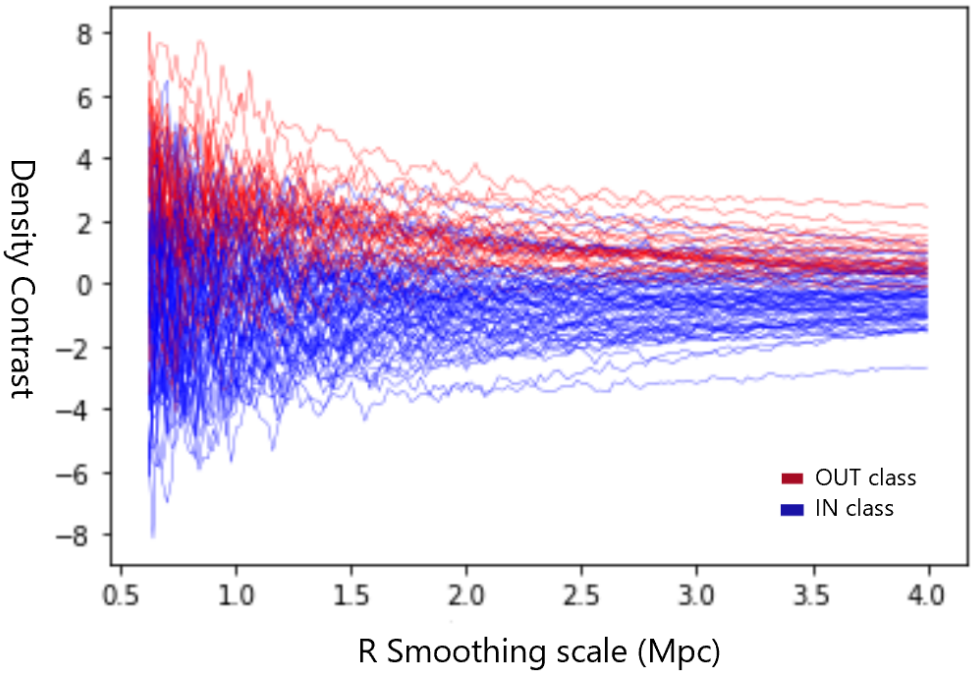
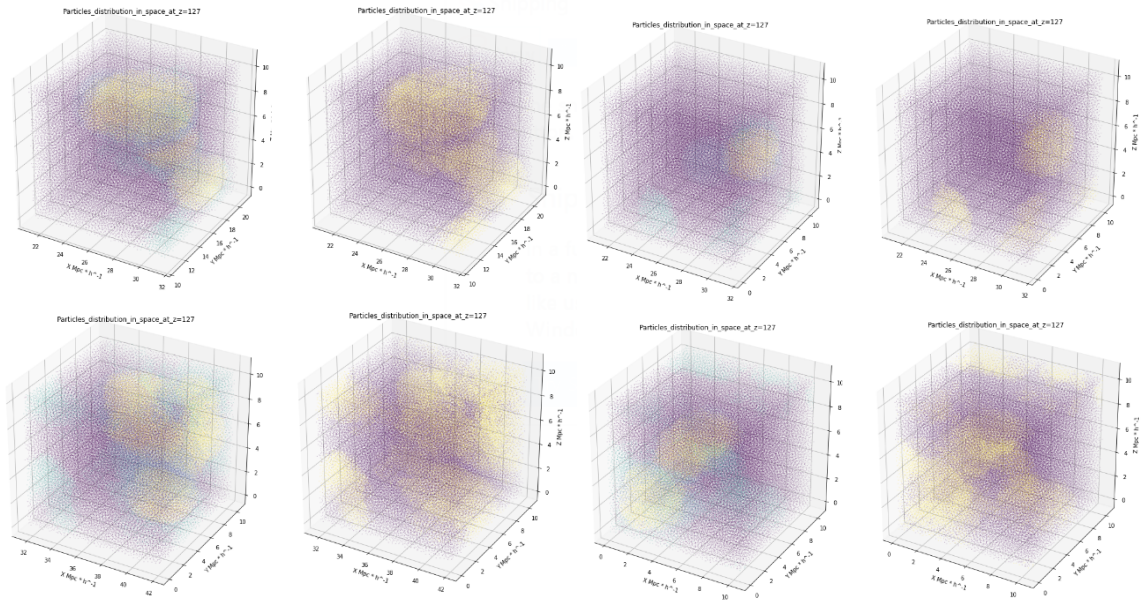


Figure 3: Density trajectories for number of particles in the box, red trajectories correspond to particles which end up in a halo with mass larger than $1600M_*$, and blue trajectories correspond to those who are not ending up to such a halo. The linear density field is smoothed with a real space top-hat filter centred on each particle's initial position.



(a) (b)

Figure 4: Distribution of particles in space at the initial condition ($z = 127$) for 4 sub-space of our box of study with size of $10 * 10 * 10(Mpc)^{-3}$. In each set the right plots correspond to the distribution of IN and OUT particles and left ones shows the boundary particles at each of these plots clearly. Yellow is for IN class particles characterized as signals by DBscan. Blue for IN class particles which are DBscan noises, as we can see most of these particles are at the boundary of the regions. Purple for OUT class particles which are noise.

also have a similar problem. Namely, it will work well for the densest elements (peaks) more precisely cores or signals of a density clustering model and less dense ones, noises (local minima of density field). However, it will face problems while modeling other particles with intermediate size of over-density, boundaries. So we classified our particles with machine learning density based clustering methods in 3 classes of core, boundary, and noise particles. We expect that a more complex model is needed for analyzing boundary particles Fig.4. More discussion on our method for noise searching can be find in section 4.6.

4 Method

We have trained and tested a variety of methods to find out the most appropriate one resulting in a high accuracy. First of all we tried classical classification methods such as Logistic Regression, KNN, SVC and Decision Tree. Then we trained Random Forest and Neural Network and in the end we used DBscan to analysis our errors and classifying the density field. The features are density for a few different smoothing scales and normalized kinetic energy of each particle. It is important to mention that our data is quite high bias. Means the number of OUT class particles is four times larger than IN class particles, so we decided to select sample particles wisely to have equal fraction of each class among 60000 particles in our sample.

We quantify the performance of the algorithm making use of a confusion matrix for binary classification problems as shown in Table 1. Throughout this analysis we always take the positives to be particles of the IN class and negatives to be particles of the OUT class. The perfect classifier consists of true positives and true negatives only. false positives and negatives are the ones which are not predicted correctly.For training our data, tuning the hyper-parameters and choosing training sample size we use the accuracy metric in all methods which is defined as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

Also we measure the true positive rate (TPR), the ratio between the number of particles correctly classified as positives and the total number of positives in the data sample,

$$TPR = \frac{TP}{TP + FN} \tag{9}$$

and the false positive rate (FPR), the ratio between the number of particles incorrectly classified as positives and the total number of negatives in the data sample,

$$FPR = \frac{FP}{FP + TN} \tag{10}$$

Most machine learning algorithms outputs a probabilistic measure of belonging to a class for every particle (in our problem between 0 and 1). For practical use this must be mapped onto a concrete class(0 or 1) for each particle. We consider different probability thresholds at which a particle is considered to belong to a class. In Fig.7(a) TPR and FPR are shown as a function of threshold for Random Forest method. A high or low threshold leads to a pure classification. A ROC curve compares the true positive rate to the false positive rate as a function of decreasing probability threshold. Fig.7(b) The area under the curve (AUC) of a ROC curve is a useful quantity to compare classifiers. The perfect classifier would have an AUC of 1, whereas a random assignment of classes would obtain an AUC of 0.5. Typically, algorithms are considered to be performing well if AUC 0.8. We use AUCs and accuracy to evaluate and compare the performance of different methods.

4.1 Classical methods

We have trained our data using a set of 60,000 randomly selected particles with balance number in the two classes from the simulation, each carrying its own set of density features and corresponding IN or OUT class for $M_h = 1600 M_*$. Methods are compared by some features such as training time and accuracy. The result is given in Table 2.

As can be seen all of these methods have AUC close to 0.88 which is a sufficiently good one. They have a quite low difference in their accuracy. Previous works has obtained an AUC about 0.89 that is close to our result.

4.2 Random Forest(RF)

We make use of the random forest implementation in the scikit-learn Python package. The trained random forest predicts the class label of the particles in the test set, which is then compared to the particles' true labels to assess the algorithm's performance. Like most machine learning algorithms, random forests have hyper-parameters which need to be optimised for a given training set. These include the number of trees and the maximum depth of the forest, the maximum number of particles at the end node of a tree and the size of the subset of features to select at a node split. We used a grid search algorithm to find the best hyper-parameters based on the accuracy of the test sample. Then with these optimal values for hyper-parameters we used learning curves for estimating the appropriate training size. Fig.5 shows that the sample size which, reduce the over-fitting and has a high score, is around $training - sample - size = 35000$. Then we consider confusion matrix and ROC curve. Random Forest has an accuracy of 80.3% and AUC score equal to 0.887. It took $50s$ for it to train and prediction time was $3s$. Random forest gives us a great tool to estimate the importance of the each feature that we are using in the classification problem and this task will give us quite great physical explanation of the most important features in our problem and appropriate physical scales. A histogram of the importance of densities in different smoothing scales is also showed in Fig.8.

As we can see the velocity which is our last feature has small contribution in forming the classification. It also can

Table 1: Confusion matrix for two classes: Positives and Negatives. The positives are particles of the IN class and the negatives are particles of the OUT class.

		True class	
		P	N
Predicted class	P	True Positive (TP)	False Positive (FP)
	N	False Negative (FN)	True Negative (TN)

Table 2: Result of classification for Logistic Regression, KNN, SVC and Decision Tree.

Model	**Training time**	**prediction time**	**ROC AUC**	**Accuracy(%)**
Logistic Regression	102 ms	396 μs	0.882	80.0
KNN	17.7 ms	2.12 s	0.876	79.2
SVC	5 min	10 s	0.882	80.7
Decision Tree	180 ms	2 ms	0.880	80.3

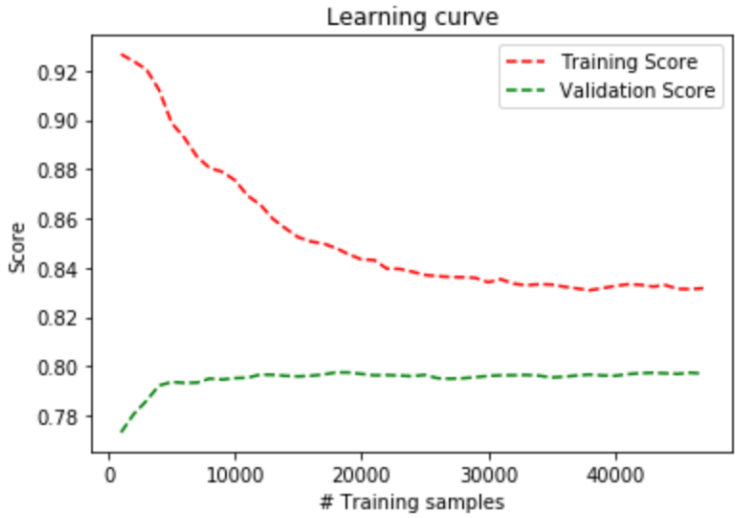
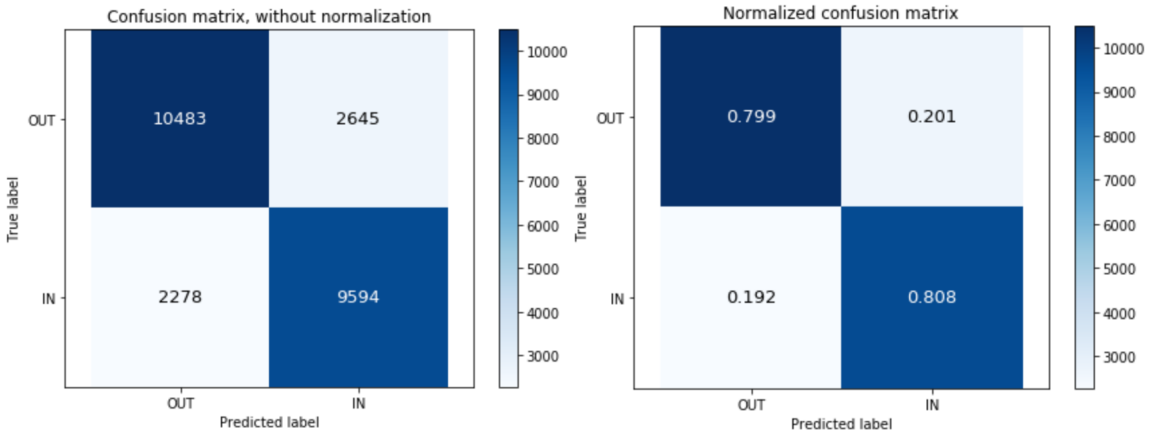
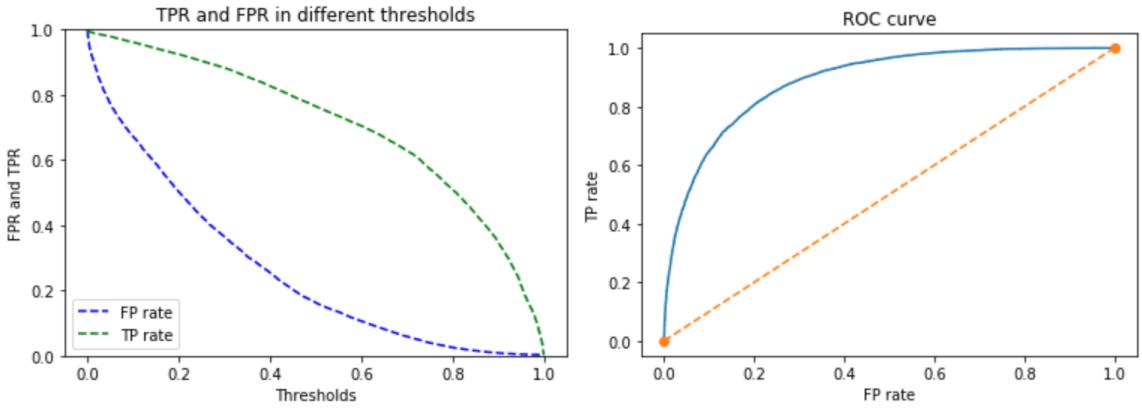


Figure 5: Score (accuracy) as a function of training sample size.we see a bit of bias in the results as can be figured out from the difference between validation score and training score.



(a) (b)

Figure 6: (a) Normalized and (b)without normalization confusion matrix, for Random Forest method. Total number of test samples is 25000.



(a) (b)

Figure 7: (a)TPR and FPR as a function of threshold and (b)ROC curve, for Random Forest method.

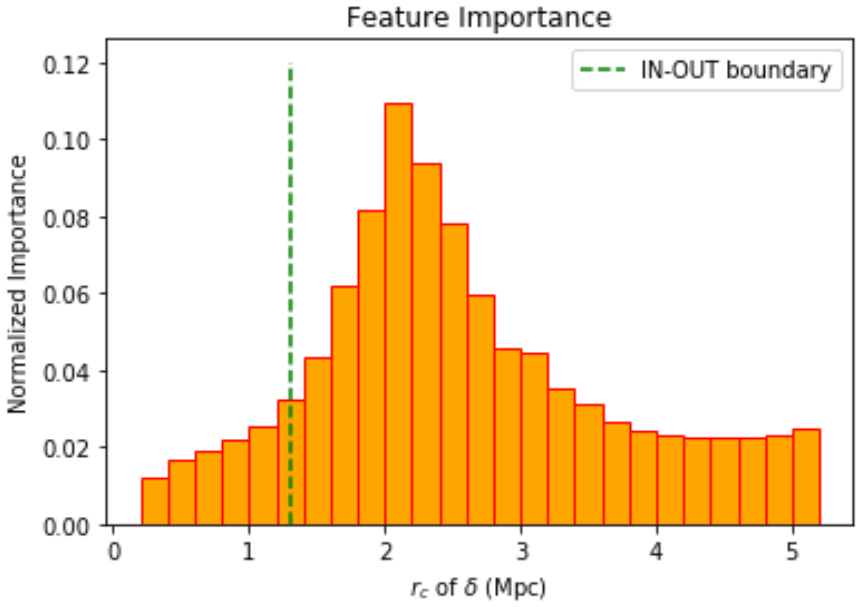
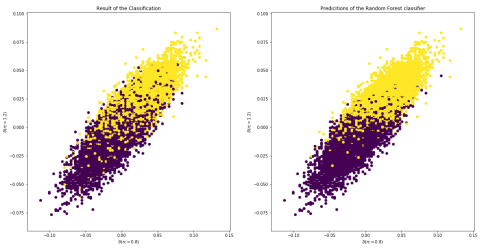
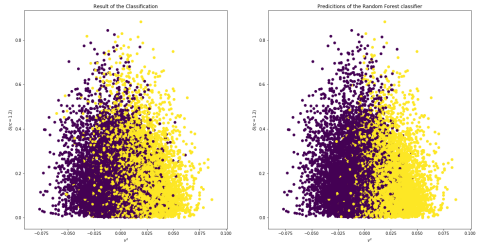


Figure 8: Histogram for the importance of densities in different smoothing scales for random forest. As can be seen density for smoothing scale around 2 Mpc makes an important role in classification.



(a)



(b)

Figure 9: Visualising the prediction result in a 2-D subspace of our feature space: First we showed it in the $\delta(rc = 0.8)$ versus $\delta(rc = 1.2)$ space (a), and then we visualized it in the velocity versus $\delta(rc = 1.2)$ space (b), for random forest classification. yellow and purple points, in order, refer to IN/OUT classes

be seen in the small correlation of velocity magnitude with other features of study.

Note that random forests are robust to correlated features, meaning that the high correlation present in our density features does not affect the predictive performance of the algorithm. Thus it seems that random forest can be a great model for our uses.

4.3 Neural Network(NN)

We have a high number of features for each particle, so at first sight NN would be a good method for training as it works well for complicated data. We used different number of layers and different activation functions but the result didn't change a lot.The Accuracy and loss curves for different architectures of the NN showed us that complex models with many nodes and layers will increase the train accuracy and decrease the train loss but the model will eventually work badly on the evaluation data. Means that accuracy on the validation will be really lower compared to the train one. So we need to use simpler models to prevent the over fitting. The best configuration is shown in Table 3. the optimal NN is not that much complicated and converged in a few epochs so it seems that our data is not very complex. For training we used cross validation which means for every single epoch a random sample of data is trained. Fig.13. accuracy obtained for NN model is 80.2%.

4.4 RF vs NN

We can see Neural network gives the same result as random forest! The results of all the models that we used so far, points out a fact about our data. It seems that we cannot get an accuracy better than 80% with our current information. there are still some works that can be done to augment our accuracy but most likely the system has some irreducible inhere randomness which can be caused by chaos and etc, so as a result we cannot make our prediction better than a constant limit which our task is finding this limit.

4.5 Oriented density

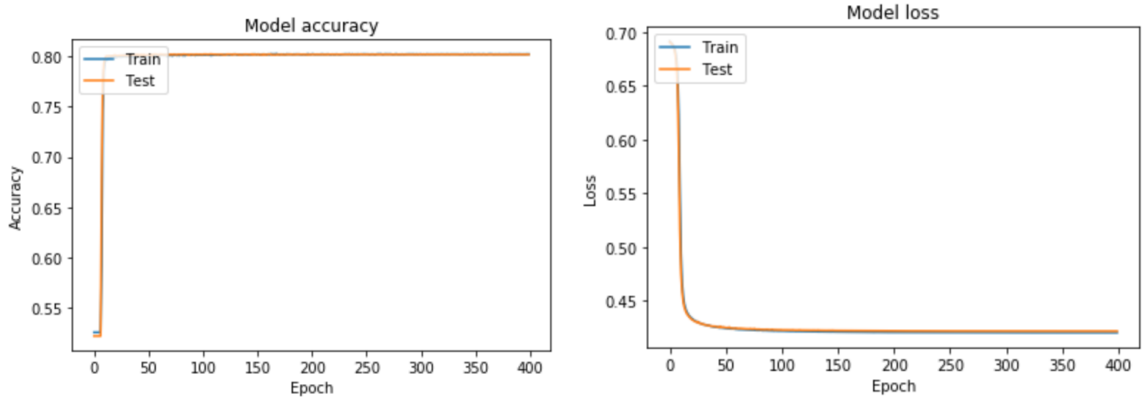
Consider a particle at the center of a cube. Our new features are the density of the eight small cubes surrounding the particle as shown in Fig.11 for different cube sizes. the fluctuation of density around a particle and it's velocity towards or away from the dense regions could play an important part for a particle to be in the IN or OUT class. This made us to use the explained oriented densities plus the velocity vector for training our data using random forest. As a result, accuracy is 81.3% which has increased a little.

4.6 Field analysis and peak searching

For getting a better result we aim to study the particles which are predicted wrong. According to section 5.2 it seems that wrong predicted particles are most considered as borders and boundary particles by DBscan clustering

Table 3: The optimized NN configuration

Layer (Input , Output) shape	Activation	Param#
(12,9)	relu	117
(9,8)	tanh	80
(8,7)	softmax	63
(7,6)	relu	48
(6,5)	relu	35
(5,3)	relu	18
(3,2)	softmax	8
Total params: 369		



(a) (b)

Figure 10: (a)validation accuracy and (b)validation loss as a function of epoch, for Neural Network.

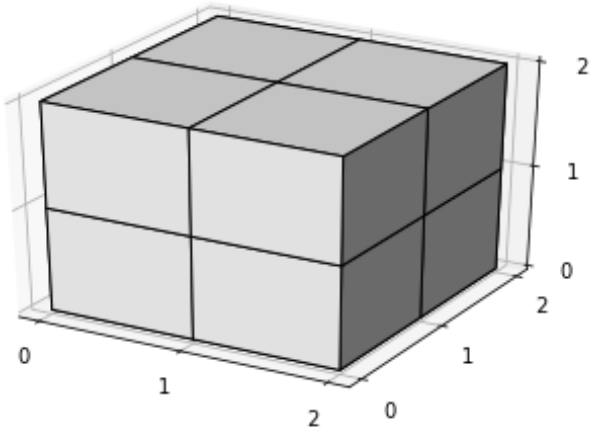


Figure 11: Consider a particle at the center of the cube. Our new features are the density of the eight small cubes surrounding the particle

method with certain hyper-parameters. as a result DBscan can help us to improve our accuracy in training. As DBscan is a density clustering algorithm there should be a relation between it's best hyper-parameters and physical scales of our problem.

The result of a DBscan model is showed in 4 for hyper parameters as follows: number of particles within any radius >1600, radius of clusters = 1.66. these hyper parameters which are the best that we could have done with DBscan are really close to our defined scales M_h and it's corresponding radius.

5 Study of Errors

5.1 In the Features Space

For a better insight from the quality of prediction, wrong and right predicted particles has been plotted in two different colors in the features space. There are 25 different density contrast parameters which are calculated for 25 different smoothing scales. We have too many options for choosing axes and how to look at the particles. In figures12 you can find some of these plots. As can be seen by these plots the accumulation of particles is around intermediate density plane which is possibly around the boundary plane of IN and OUT classes and it was expected. But no other special clustering can be seen in other places.

5.2 In Cartesian Space

To see if there is a common feature between the particles which are predicted wrong, we plotted the false positive and negative ones in the 3D space and studied whether they are mostly in the boundary of IN and OUT regions of the space or not. According to Fig.13 in some sub spaces of our box they are in the border and it would be hard for the machine to distinguish them. More precisely model is able to predict the most and least density regions simply but the borders are challenge for it. If we could be able to find the particles in the border of IN and OUT regions, we can study them more specific and reach into a higher accuracy. This would be our next step.

6 Searching for moving barrier and characteristic collapse scale

We defined a new characteristic density in Sec. 3, As the density which divides our two IN and OUT classes the best, in the smoothing scale that has the largest impact on the classification.

This newly defined scale can directly be found from the feature importance plots of Random Forest model. We expect that this scale has a relation with the M_h used for constructing our classes. The density boundary at this scale should have a relation with the evolution function of density field.

For finding this scale at each M_h we used 25 features for each of our particles from 0.3 to 5.1 $h^{-1}Mpc$. We found the IN and OUT classes for 20 different M_h from 1600 to 20600 M_* . We trained the Random Forest model with the best set of hyper parameters founded by the grid searching method on each of these mass scales. The feature importance plot for all the masses can give us the characteristic scale for each halo mass.

From Fig. 14 it is observable that the characteristic length increase with the increasing of the halo mass. This is in an excellent agreement with our expectations and previous theories, as the characteristic theoretical length $r_c \propto \frac{M_h}{<\rho>}$ also increases with the halo mass.

Characteristic density at each of these scales also founded by looking at the distribution of particles from both classes on the best smoothing scale. The more accurate analysis needed here, so we provide our exact results of these densities in the future.

7 Halo Mass function

We trained our model for 20 different halo masses. Once the model trained, it can be used for all future needs. We can use it to find the appropriate halo mass function of the box. Consider two halo masses M_h and $M_h + dM$; to find the number of halos between these two masses, we coursed two steps. First, we defined the IN and OUT classes based on the specific M_h. Training the model on this new data and then use it for test samples of the box can give us the predicted number of particles which will go into a halos with $Mass > M_h$, doing the same for halo mass $M_h + dM$ and then subtracting the number of particles in the IN class of halo mass $M_h + dM$ from the number of particles in the IN class with the halo mass M_h, gives us the number of particles in halos with a mass between M_h and $M_h + dM$. Dividing this number by the mean number of halo particles $M_h + dM/2$, gives us an order of the number of halos with a mass between M_h and $M_h + dM$.

$$n_{halo}(M_h < M < M_h + dM) = \frac{len[IN(M_h)] - len[IN(M_h + dM)]}{M_h + dM/2} \quad (11)$$

Doing this process for any mass range enables us to predict the number of halos in that specific range. Making ranges narrower will end in finding the halo mass function of dark matter. We did this on our data, but eventually, it over-estimated the number of less massive halos, the more precise analysis should be done to provide the demanded high accuracy halo mass function which we left them for our future studies.

8 Conclusion

We have presented a machine learning approach toward the study of structure formation and investigation of dark

matter particle's destiny. We trained many algorithms on N-body simulations, it predicts whether a mass element will end up into a halo of mass range and whether a region will collapse to form a halo. We used density field aspects as our only relevance features as previous works had been shown that tidal shear does not enhance the result. This works generated a mapping between initial conditions and final halos that would result from non-linear evolution for many halo masses, without need to adopt any further approximations and by a completely model-independent manner. We introduced a new mass scale and density parameter, which can provide a new insight through the study of structure evolution and the collapse of initial fluctuations. We provided an algorithm to find this relevant scales and critical density for any given mass range and then hopefully by use of these, finding the halo mass function. We tested our models by use of cross-validation, but there is still a need for blind analysis, which can be provided by the realizations of different universes by PLANCK collaboration[10]. Our halo mass function and critical densities needs more accurate examinations and studies, and we should test their accuracy by existing halo mass functions[11][12]. Future works will be devoted to improving our accuracy by using other models as Convolutional Neural Network, which can blindly find the appropriate features. Also, regression models will be used for finding the final halo mass of an over-density. Reaching a certain halo mass function, which can be directly tested against existing fitting formulae adopted by analytic approaches, is our primary goal in the following works.

9 Acknowledgements

We thank Dr. Shant Baghram and Dr. Sadegh Raeisi for their useful guide and helps, and Zahra Baghkhani for the good discussions.

References

[1] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. 2016.

[2] Luisa Lucie-Smith, Hiranya V. Peiris, Andrew Pontzen, and Michelle Lochner. *Machine learning cosmological structure formation*. arXiv:1802.04271v2 [astro-ph.CO], 2018.

[3] Luisa Lucie-Smith, Hiranya V. Peiris, and Andrew Pontzen. *An interpretable machine learning framework for dark matter halo formation*. arXiv:1802.04271v2 [astro-ph.CO], 2019.

[4] Volker Springel. *The cosmological simulation code GADGET-2*. Monthly Notices of the Royal Astronomical Society, Volume 364, Issue 4, December 2005, Pages 1105–1134, 2005.

[5] Volker Springel, Naoki Yoshida, and Simon D.M.White. *GADGET: a code for collisionless and gasdynamical cosmological simulations*. New Astronomy,Volume 6, Issue 2, April 2001, 2001.

[6] E. Macaulay et al. *First Cosmological Results using Type Ia Supernovae*. : https://arxiv.org/pdf/1811.02376.pdf, 2019.

[7] William H. Press and Paul Schechter. *Formation of Galaxies and Clusters of Galaxies by Self-Similar Gravitational Condensation*. Astrophysical Journal, Vol. 187, pp. 425-438 (1974), 1974.

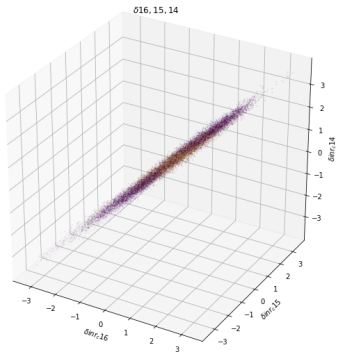
[8] Houjun Mo, Frank van den Bosch, and Simon White. *Galaxy Formation and Evolution*. Cambridge University Press, 2004.

[9] J. R. Bond, S. Cole, G. Efstathiou, and N. Kaiser. *Excursion Set Mass Functions for Hierarchical Gaussian Fluctuations*. Astrophysical Journal v.379, p.440, 1991.

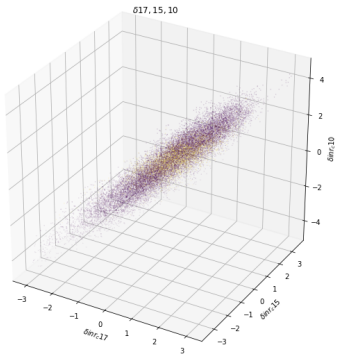
[10] PLANCK collaboration. *Planck 2015 results- XIII. Cosmological parameters*. AA, Volume 594, October 2016, Planck 2015 results, 2015.

[11] M. Maggiore and A. Riotto. 2010.

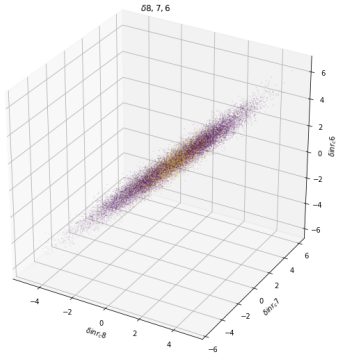
[12] Ravi K. Sheth and Giuseppe Tormen. *Large-scale bias and the peak background split*. Monthly Notices of the Royal Astronomical Society, Volume 308, Issue 1, 1999.



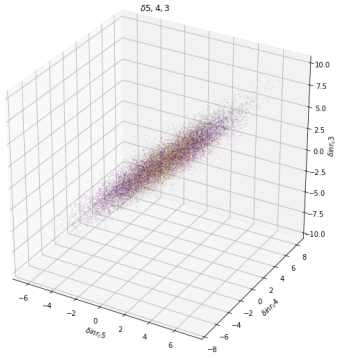
(a)



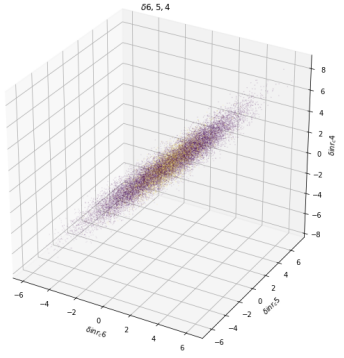
(b)



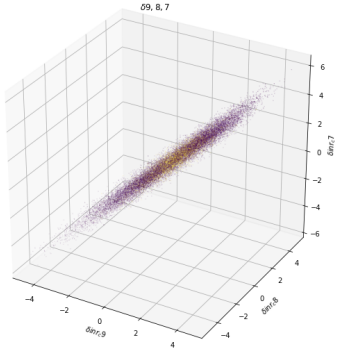
(c)



(d)



(e)



(f)

Figure 12: Wrong and right predicted particles in redshift 127. Particles which predicted wrong are shown by yellow and those which predicted right are shown with purple. most of the wrong predicted particles are clustered around a plane of intermediate densities.

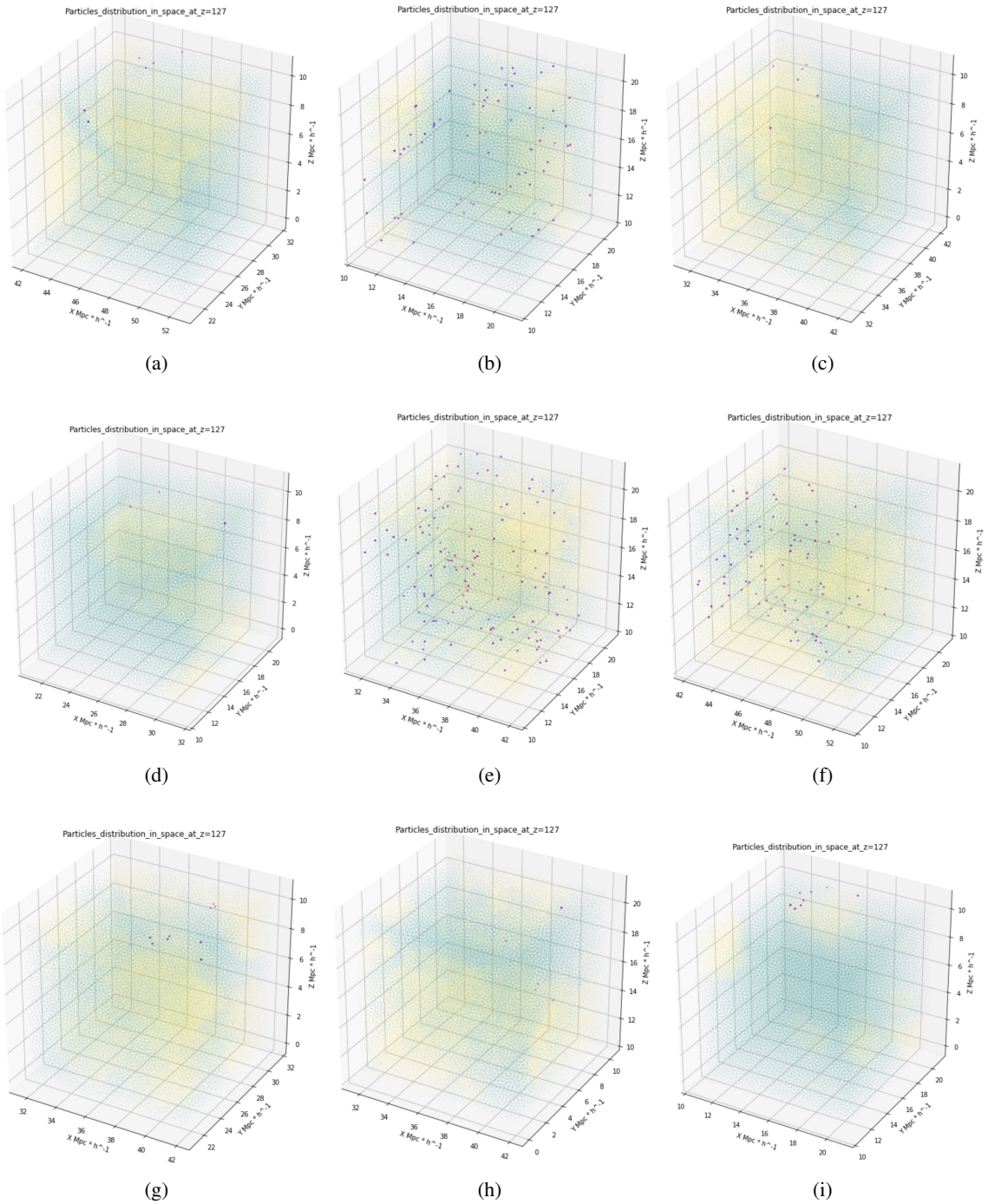
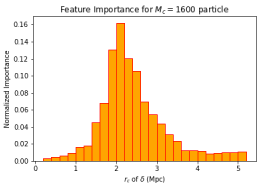
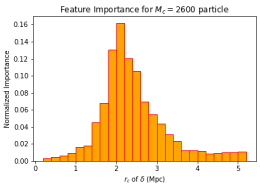


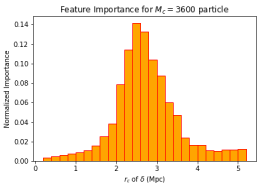
Figure 13: False negative and positive particles distribution in some sub spaces of the total box. yellow particles will end in a halo unlike the blue ones. The wrong predicted particles are colored with purple.



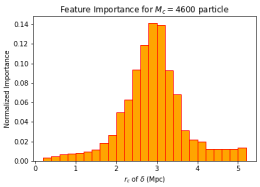
(a) $M_h = 1600M_*$



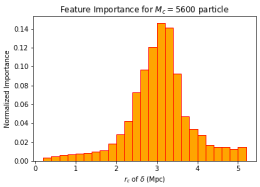
(b) $M_h = 2600M_*$



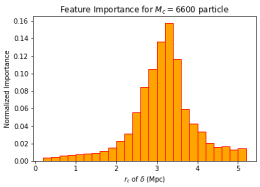
(c) $M_h = 3600M_*$



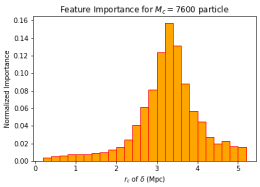
(d) $M_h = 4600M_*$



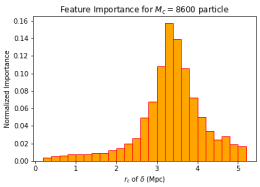
(e) $M_h = 5600M_*$



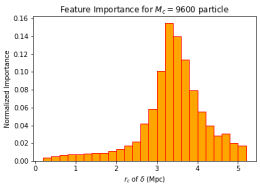
(f) $M_h = 6600M_*$



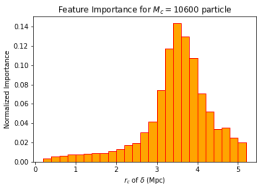
(g) $M_h = 7600M_*$



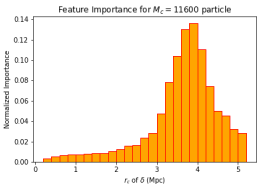
(h) $M_h = 8600M_*$



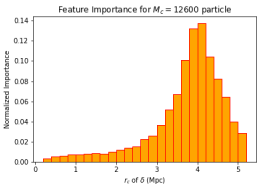
(i) $M_h = 9600M_*$



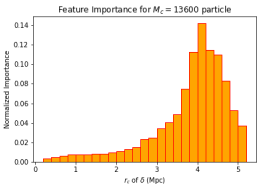
(j) $M_h = 10600M_*$



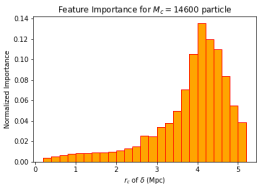
(k) $M_h = 11600M_*$



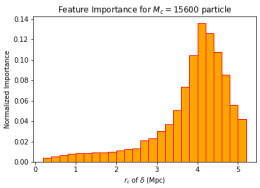
(l) $M_h = 12600M_*$



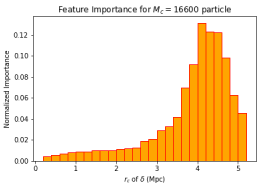
(m) $M_h = 13600M_*$



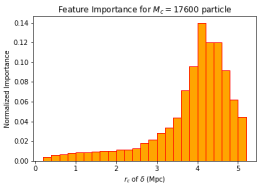
(n) $M_h = 14600M_*$



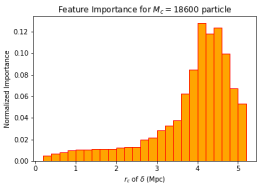
(o) $M_h = 15600M_*$



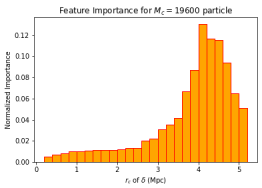
(p) $M_h = 16600M_*$



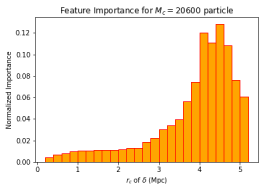
(q) $M_h = 17600M_*$



(r) $M_h = 18600M_*$



(s) $M_h = 19600M_*$



(t) $M_h = 20600M_*$

Figure 14: Feature importance plots for all of our halo mass ranges, as you can see the characteristic scale clearly increase with the halo mass.