Astronomy and Computing 38 (2022) 100527

Contents lists available at ScienceDirect

Astronomy and Computing

journal homepage: www.elsevier.com/locate/ascom

Full length article

Classification algorithms applied to structure formation simulations

J. Chacón, J.A. Vázquez *, E. Almaraz

Instituto de Ciencias Físicas, Universidad Nacional Autónoma de México, Apdo. Postal 48-3, 62251 Cuernavaca, Morelos, Mexico

ARTICLE INFO

Article history:
Received 15 June 2021
Accepted 20 November 2021
Available online 10 December 2021

Keywords:
Numerical simulations
N-body systems
Machine learning

ABSTRACT

Throughout cosmological simulations, the properties of the matter density field in the initial conditions have a decisive impact on the features of the structures formed today. In this paper we use a random-forest classification algorithm to infer whether or not dark matter particles, traced back to the initial conditions, would end up in dark matter halos whose masses are above some threshold. This problem might be posed as a binary classification task, where the initial conditions of the matter density field are mapped into classification labels provided by a halo finder program. Our results show that random forests are effective tools to predict the output of cosmological simulations without running the full process. These techniques might be used in the future to decrease the computational time and to explore more efficiently the effect of different dark matter/dark energy candidates on the formation of cosmological structures.

1. Introduction

The evidence gathered over the past twenty years consistently points out that the Universe is made up of about 96% of dark energy and dark matter. This conclusion is one of the main foundations of the standard cosmological model (ΛCDM), which despite its success in describing the observations does not provide yet a complete answer on the physical nature of these components. In this regard, the process of structure formation is a very useful tool to characterize the properties of the dark sector and to assess its impact on the historical evolution of the Universe.

Cosmological structure formation is a process determined mainly by the gravitational interaction of dark matter. This process can be broken down into three major stages:

Linear regime. Initially, dark-matter perturbation modes remain frozen, and they start growing once they enter the causal horizon of the Universe. During this stage, density fluctuations remain small enough to be described by linear perturbation theory.

Intermediate regime. As density fluctuations keep growing, a transition to non-linear regime takes place in which perturbations collapse into denser regions called halos. This transition process can be described in its essentials by semi-analytic models, such as the spherical collapse model.

Non-linear regime. Finally, halos group into larger structures that give rise to a cosmic network of filaments and knots. These structures serve as gravitational wells around which visible matter accretes, so by mapping the distribution of clusters of galaxies, quasars, and gas clouds, we expect to reconstruct the underlying skeleton of dark matter.

While the linear evolution and the transition to the non-linear regime can be approached by analytical methods, structure formation in the non-linear regime can only be studied using numerical simulations. These simulations are virtual laboratories by which is possible to study in detail the characteristics of the structures that stem from the dynamics of different candidates of dark matter and dark energy. By comparing the predictions of each model with observations, it is possible to evaluate the feasibility of each one of these scenarios.

Until a few decades ago, numerical simulations were prohibitive in terms of the amount of the computational resources required, but recently the progress in hardware and the development of new algorithms have made these tools accessible to more research groups. Still, the cost remain high and for some tasks they will remain out of reach in the foreseeable future. In this sense, there is an incentive to develop artificial intelligence/machine learning solutions that allow the prediction of important features in numerical simulations without the need of completely executing them or, at most, running a small number of simulations.

The method described in this work is based on a similar approach made in Lucie-Smith et al. (2018), which uses the ability of machine learning algorithms to learn complex relationships in large data sets. We aim to find out how much information provide the features of the initial conditions to determine the formation of dark matter halos in cosmological simulations.

* Corresponding author.
E-mail addresses: jchacon@icf.unam.mx (J. Chacón), javazquez@icf.unam.mx (J.A. Vázquez).

https://doi.org/10.1016/j.ascom.2021.100527
2213-1337/

The content of this paper is as follows. In Section 2 we provide a short review of machine learning fundamentals. We extend the discussion to two supervised learning algorithms: decision trees and random forests. In Section 3 we review some metrics of performance. In Section 4 we discuss the problem of halo formation as a binary classification problem. Then, we present the setup of our simulations, the construction of the training and test sets, and the process of hyperparameter tuning. Finally, in Section 7 we present our conclusions and a brief discussion of the results achieved.

2. Machine learning

Machine Learning (ML) refers to the set of methods used to train computers in order to find patterns in data and make inferences without human intervention. Although this is still a remote possibility for certain applications, currently ML methods are successfully applied to several problems that require the analysis of large volumes of information, for which the cost in terms of human resources needed may represent a serious issue. ML methods are usually classified into two broad categories:

Supervised learning. Supervised methods need a set of labeled samples characterized by some features. In this case computers are trained to find a relationship among the features and the labels, so that the labels of new samples may be predicted based on the learned relation. Examples of these algorithms are: Logistic Regression, Decision Trees, Neural Networks, Support Vector Machines, etc (Gron, 2017). Some works applying these methods in cosmology can be found in Gómez-Vargas et al. (2021), Xu et al. (2013), Hajian et al. (2015), Moster et al. (2020), Kamdar et al. (2016), Buncher and Carrasco Kind (2020), Perraudin et al. (2019).

Unsupervised learning. Unsupervised methods are aimed for making inferences on data sets where samples are not labeled. In this case computers are trained to find hidden patterns in the data, letting the information speak for itself. Examples of these algorithms are: Cluster Analysis, Correlation and Principal Component Analysis (PCA) (Goodfellow et al., 2016). Some applications in cosmology can also be found in Sharma et al. (2020), Agarwal et al. (2018), D'Addona et al. (2021), Cheng et al. (2020), Geach (2011), Hocking et al. (2017), Cheng et al. (2021).

2.1. Supervised learning

Since the main goal of this paper is classification we will make use of supervised learning, whose task is the following:

Given a **training set** of N input–output pairs

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N), \tag{1}$$

where each y_j is computed using a $y = f(x)$ function, the objective is to find a function g that approximates the true function f. The function g is a hypothesis and the variables x and y can take any value and not necessarily a numeric one, that is, it can be an attribute. The learning is carried out through a search, within the space of possible hypotheses, of a function that has a good performance, even when it is fed with new examples beyond the training set. In order to test out the accuracy of the hypothesis, a **test set**, different from the training set, is provided. The hypothesis g is said to generalize well to the function f if it correctly predicts the values of y for several inputs. Furthermore, the dependent variable y can turn out to be categorical, also called qualitative. The values of a categorical variable are mutually exclusive and in that case the learning problem is called a **classification** problem, which in turn is referred as Boolean or binary classification if only two values are possible.

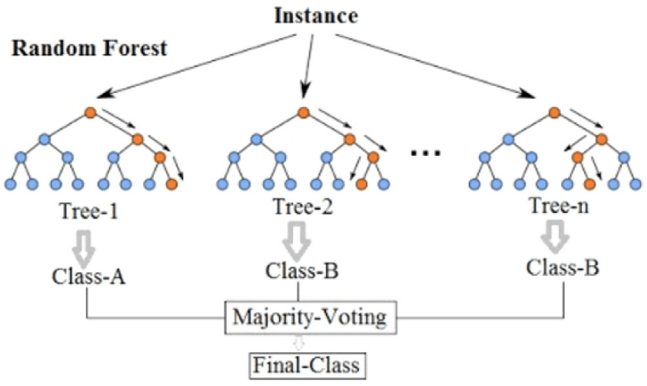

Fig. 1. Sketch of a Random Forest Algorithm. Being an assembly of decision trees, it allows different tests to be carried out on a random selection of attributes, the final class being a vote on a majority obtained in each individual tree.
Source: Figure from Medium.

2.2. Decision trees

These type of algorithms resemble a chart flow for data, where terminal blocks represent classification decisions. The elements of a decision tree are the root (where the data is stored), the branches (the path the tree takes to make decisions) and nodes (consisting of sets of elements that have a determined characteristic after a decision is made). Given a dataset, we can calculate the inconsistency within the set, or in other words, find its entropy in order to divide or split the set until all data is within a given class (Quinlan, 1986).

A decision tree reaches a conclusion by carrying a series of tests. Its nodes perform tests over the attributes on the input values A_i and the branches that come from the node are labeled with the possible values of the attribute $A_i = v_{ik}$. The leaf nodes in the tree specify a value that needs to be computed by the function. A good decision algorithm is developed by splitting the data, so the attribute with the greatest weight or with the highest information gain is obtained, so it is expected to have a correct classification with the least possible number of tests (Louppe, 2014).

2.3. Random Forest

The Random Forest algorithm consists of a large number of decision trees that operate together as an ensemble (Hastie et al., 2009). The "randomness" of the algorithm comes from the fact that operations and predictions from the forest are not hierarchically taken, but a subset of elements (like number of trees, number of attributes, length of data, etc.) is taken in a random way. Each individual tree in the Random Forest chooses a class prediction and the class with the most votes becomes the model prediction. This is due to a simple but powerful concept: *The wisdom of the crowds*. The reason that Random Forest is such a good algorithm is because a large number of relatively uncorrelated trees operating together will perform better than any individual model that constitutes it (Breiman, 2001). The key is the low correlation among models. Uncorrelated models can produce joint predictions that are more accurate than any of the individual predictions that make them up. The trees protect each other from their individual errors (as long as those errors are not in the same direction). If some trees have errors, others may get correct predictions, so that as a group the trees can move to the correct direction (see Fig. 1).

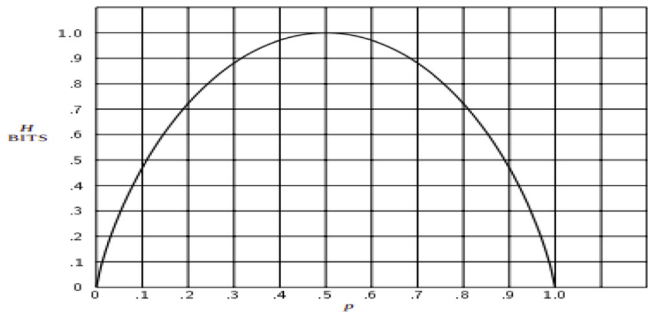

Fig. 2. The entropy for two classes with probability p and $(1-p)$. Shannon's entropy is a way of measuring the relative quantity between the two classes. The entropy value is maximum if there are the same number of classes.
Source: Taken from Shannon (1948).

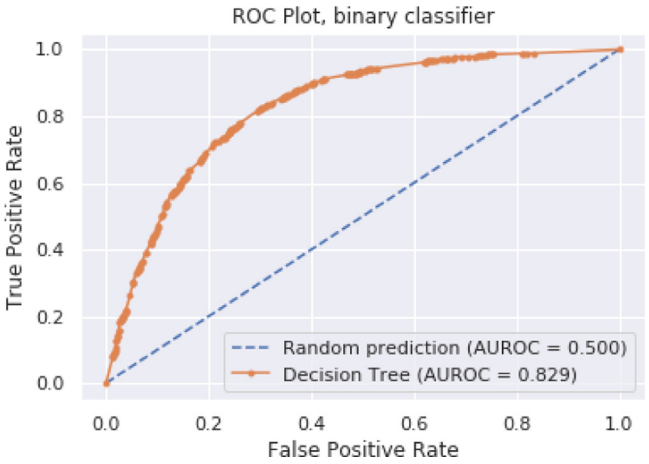

Fig. 3. The ROC curve and the value of the Area Under the Curve (AUC) of a binary classifier. Being a graphic representation, the performance can be evaluated at various prediction thresholds. For different classifiers, the shape of the ROC curve can be very similar, the fairest way to compare them is through the value of the AUC. GitHub ChJazhiel.

2.4. Information and entropy

Decision algorithms like decision trees and random forest perform data division, also called *split* in order to obtain more information after the division is made. This split can be thought of as a way to organize data, thus the learning process should be focused on obtaining a better vision of the analysis process. This comes directly from information theory: the most valuable information comes from unlikely events rather than events that occur frequently. A way to determine the sought information in a more formal and specific way is by considering that the most probable events provide few information, while the least probable events provide the highest amount of information.

The equation that satisfies these conditions is the information content of an event x_i

$$I(x_i) = -\log_2 P(x_i), \tag{2}$$

where $P(x)$ is the probability that event x_i occurs. To account for the whole set of events, a probability distribution is built by using the **Shannon entropy** (Shannon, 1948)

$$H(x) = -\sum_i P(x_i)\log_2 P(x_i), \tag{3}$$

where the sum of i is over all possible events. That is, the Shannon entropy is the expected amount of information in an event of a probability distribution as observed in Fig. 2. The change in the information obtained before and after the division is known as **information gain**. Therefore, the split is made when the information gain is greater.

3. Evaluation of classification models

Evaluating the performance of an algorithm is a fundamental aspect in machine learning. The model must be trained with the **training set** and then evaluated with the **test or validation set**, consisting of totally new data not yet evaluated by the algorithm. During the evaluation is important to measure and understand the quality of the classifier and to tune the parameters in the iterative process of discovering the data.

3.1. ROC curves

Binary classification models are evaluated in the Receiver Operator Characteristics (ROC) space (Fawcett, 2006). A ROC graph is used as a visual representation of the classifier based on its performance. This type of curve shows how the number of correctly classified as true examples changes with respect to the number of incorrectly classified as negative examples. In ROC space we can define the True Positive Rate (TPR) as

$$TPR = \frac{TP}{TP + FN}, \tag{4}$$

and the False Positive Rate (FPR) as

$$FPR = \frac{FP}{FP + TN}. \tag{5}$$

These two quantities are plotted in order to obtain the ROC curve, see Fig. 3 for reference. The FPR measures the fraction of negative examples incorrectly classified as positive, while the TPR measures the fraction of positive examples correctly classified. The convex part of a family of ROC curves can include points located further toward the northwestern boundary of the ROC space. If a line passes through the convex part, then there is no other line with the same slope that passes through another point with a larger TP intersection. In this way, the classifier at that point is optimal under any distribution assumption with that slope (Rokach and Maimon, 2008).

3.2. Area Under Curve (AUC)

Using continuous-type measures such as ROC curves sometimes can lead to misinterpreting the results. In the case of the ROC curves, for example, for two classifiers there may be an overlap in the curves within the ROC space, so that it becomes difficult to determine which model performed better. If there is no dominant model, it cannot be determined which of them is the best.

The Area Under Curve (AUC) is a quite useful metric to visualize the performance of a classifier, since it is independent of the decision criteria and prior probabilities. Given two classifiers, if the ROC curves intersect then the AUC is an average of the comparison between both models. The AUC does not depend on any imbalance of the training data, so comparing this quantity for two classifiers is fairer and more informative than comparing their misclassification rates, for example. We evaluate the performance of an algorithm with this metric with the range values between 0.5 and 1.0. A value of 0.5 is only as good as a random classifier. Then 0.6–0.7 is considered as a regular classification, 0.71–0.8 a good classification, 0.81–0.9 very good classification and 0.91–1.0 an excellent one.

3.3. Overfitting and generalization

Overfitting is a general phenomenon and occurs in all kinds of learning algorithms, even when the target function is not random at all. It becomes more likely as the hypothesis space and the number of input attributes grow, and is less likely as the number of training examples increases.

For random forest and decision tree algorithms, there exist a technique, called *pruning*, that aims to avoid overfitting. Pruning works by removing nodes that are not clearly relevant. The question is, how do you detect that a node is testing an irrelevant attribute? Assuming that a node consists of p positive examples and n negative examples. If the attribute is irrelevant, it would be expected to divide the examples into subsets, so that each one has approximately the same proportion of correctly classified (positive) examples as the complete set, $p/(p+n)$, in this way the information gain would be close to zero.

How big must the information gain be so that it can be divided on a particular attribute? This question is answered using a significant statistical test where the null hypothesis is that no relationship or underlying pattern exists. The actual data is then analyzed to calculate the degree to which it deviates from a perfect absence of a pattern. Given a training set S with input attributes $A = a_1, a_2, \ldots, a_n$ and a nominal target attribute of unknown fixed distribution D on the instance space, the goal is to induce an optimal classifier with minimal generalization error.

In other words, given a training set with a finite number of attributes and a set of classes to be determined, find the algorithm that best generalizes the model with a minimal error.

3.4. Learning curve

These curves are graphs of the learning performance of the model over experience or time. They are a diagnostic tool widely used in machine learning for algorithms that learn incrementally from a training set. The model can be evaluated on the training set and on a validation dataset after each update during the training. Graphs of the measured performance can be viewed to show the learning curves.

Reviewing the learning curves of the models during training can be used to diagnose learning problems, such as an underfitting or overfitting model, as well as whether the training and validation data sets are adequately representative, as seen in Fig. 4.

The evaluation in the validation set offers an idea of how capable the model is of "generalizing". In the space of the learning curve there are two curves:

- Train Learning Curve: computed from the training set that gives an idea of how well the model is learning.
- Validation Learning Curve: computed from the validation set that gives an idea of how good the model is at generalizing.

Because the metrics to evaluate an algorithm are diverse, a simple way to create a learning curve is through accuracy, although it can also be created through an error percentage. To ensure optimal learning, the dataset is divided into subsets of samples called ***k*-Fold Cross-validation**. The procedure has a parameter k that refers to the number of groups the dataset will be divided. It is a simple method to understand and to help the model to decrease variance and avoid bias. This method has the following steps:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups.
3. For each unique group:
   (a) Take a group as a test set.
   (b) Use the remaining $k-1$ groups as training set.
   (c) Adjust the model with the training set and evaluate it with the test set.
   (d) Save the score of evaluation and discard the model.
4. Gather model skills using the model score sample.

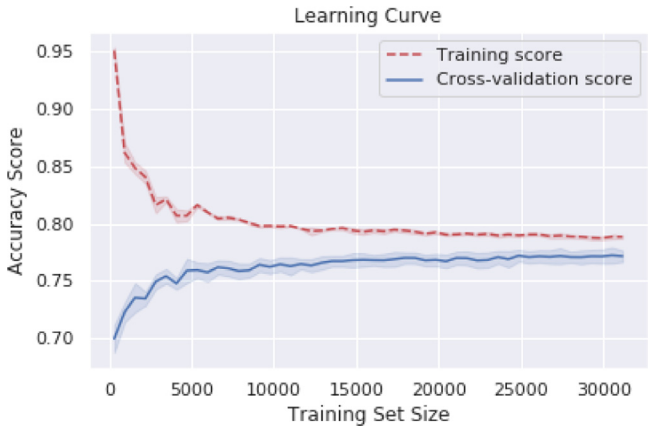

Fig. 4. Learning curves with accuracy metrics of a binary classifier. The algorithm is said to be learning when the validation set curve is close to the training set curve. In this graph, it is observed that the model does not require changing its hyperparameters since the learning set is quite close to the training curve and does not seem to be overfitted. GitHub ChJazhiel.

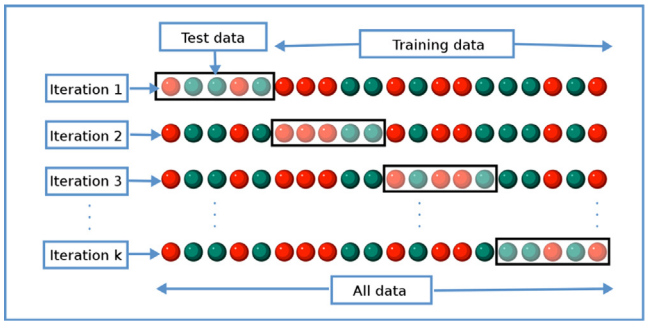

Fig. 5. The k-fold cross validation. The dataset is randomly mixed and a test group is chosen, leaving the rest of the data as training. The iterations are used to carry out this method in a defined way in order to minimize the variance and the bias of the model (Seni and Elder, 2010)..

This approach involves randomly dividing the set of observations into k groups, of approximately the same size. The first group is treated as a validation set and the method fits the remaining $k-1$ groups (James et al., 2014).

Graphically, it can be understood with Fig. 5. From here it is clearly observed how this method mixes and divides the set randomly, so that a small group is the test or validation set and the rest of the data is the training set. This process is carried out recursively to avoid some type of bias or variance of the model.

4. Numerical cosmology as a binary classification problem

After carrying out a series of simulations for various configurations (for more details see Chacón et al., 2020) we obtain a one-to-one relationship of dark matter halos formed at time $z = 0$ with the initial conditions. This allows to identify the dark matter halos, called parents or hosts and in turn we are able to determine substructures or subhalos of the same host. We select a dark matter mass threshold to identify halos, in this way it is possible to identify the particles that end up in a dark matter halo given the mass threshold, as well as those that do not end in a

halo, that is, they are free particles or they belong to halos of lower mass. As it can be deduced, this leads to treat the process of dark matter evolution as a classification problem.

4.1. Data selection

To carry out the process, we chose a cosmological simulation of a ΛCDM Universe made with the cosmological code GADGET-2 (Springel, 2005), with cosmological parameters $\Omega_m = 0.268$, $\Omega_\Lambda = 0.683$, $\Omega_b = 0.049$, $h = 0.7$. The simulation has a gravitational softening of $\epsilon = 0.89$ kpc and it evolves a total of 192^3 particles, each with a mass of 1.3×10^9 $M_\odot$ in a box of comoving length $L = 50h^{-1}$ Mpc from $z = 23$ to $z = 0$. Halos (both host and subhalos) are identified with ROCKSTAR halo finder (Behroozi et al., 2013). We made two classes, [*Not in Halo*, *In Halo*] by selecting a mass threshold of $M \geq 1.2 \times 10^{12}$ $M_\odot$, so that the class *in Halo* will be in halos that exceed this threshold while the particles *Not in Halo* are in halos with mass less than said threshold or they are not linked to any halo. The final snapshot counted a total of 4000 dark matter halos whose masses fall within the range ($10^{11} \leq M/M_\odot \leq 10^{14}$).

Each particle will have a 10 component vector associated with it and a label: 1 for the class *In halo*, 0 for the class *Not in halo*. The properties of the particles are extracted from the initial conditions of the simulation ($z = 23$) and are used as an input data for the decision tree and random forest algorithms. The components are the mass densities centered in each particle linked to the local density of the initial redshift. A subset of all the particles was chosen within the simulation with their respective label. The training was carried out with an 80/20 split of the subset (80% training and 20% test/validation).

Supervised machine learning algorithms require the use of characteristics of a structured database, in this case there is a structured data set with attributes extracted from the density field. This assignment comes from analytical works related to the halo mass function (HMF) by Press–Schechter (Press and Schechter, 1974). This function predicts the density of the number of halos of dark matter depending on their mass and the density field. The density will form a halo of a certain mass M at a redshift z. If it exceeds a critical value $\delta_c(z)$, these values will be called overdensities at a given redshift z.

The main idea is that the matter of a halo will be enclosed in a dense spherical region, where the density contrast will be given by the relation

$$\delta(\mathbf{x}) = \frac{\rho(\mathbf{x}) - \bar{\rho}}{\bar{\rho}}, \tag{6}$$

where $\bar{\rho}$ is the average matter density of the Universe. For a sphere of radius R (Dodelson and A.P.L. 1941–1969, 2003), it is well understood that the overdensity is

$$\delta(\mathbf{x}, R) \equiv \int d^3\mathbf{x}' \delta(\mathbf{x}') W_R(\mathbf{x} - \mathbf{x}'). \tag{7}$$

In Eq. (7), W_R is a window function of the *top hat* model, given by

$$W_R = \begin{cases} \frac{3}{4\pi R^3} & \text{if } |\mathbf{x}| \leq R \\ 0 & \text{if } |\mathbf{x}| > R. \end{cases} \tag{8}$$

A window function with radius R corresponds to a mass scale $M = \bar{\rho} V(R)$. The expected value of the overdensity in Eq. (7) is the normalization term of the power spectrum σ_R

$$\sigma_R^2 = \langle \delta^2(\mathbf{x}, R) \rangle. \tag{9}$$

The choice of attributes of the structured data for the machine learning algorithms reside in the density contrasts calculated with the top hat type window function that is derived from a mass scale M_R in the radius R, centered on the position of a particle, from the initial conditions and the initial redshift $z = 23$. The result is a quantity of 10 overdensities $\delta_1, \ldots \delta_{10}$, each one associated with their respective class or label. This 10-component vector along with their corresponding labels makes a dataset that achieves well performance. That is, when we chose more than 10 component vectors, which means using bigger regions of study, we saw no further improvement in our training. On the other hand, if less vectors were studied we observed a low accuracy performance in the algorithms. The mass range was selected taking into account two main features. First, the data we selected had more massive halos and less massive halos and free particles, the mass range was an average of the halos found. Second, the mass range was also a calculation from the approximation of the spherical collapse model, which gave us the number for the threshold (see Fig. 6).

Table 1
Optimal hyperparameters for the algorithms..

Description	Symbol	Value
Decision criteria	`criterion`	`entropy`
Max depth	`max_depth`	8
Class balance	`class_weight`	`balanced`
No. of estimators	`n_estimators`	2000
Min. No. of particles	`n_particles`	200

4.2. Training

Algorithms used for this section were decision trees and random forest, included in the machine learning package from **Scikit-Learn** (Pedregosa et al., 2011). The initial number of particles was 50,000 randomly selected, but a preprocessing was performed before i.e. labels [*Not in Halo*, *In Halo*] were converted to a set of labels 0 and 1, respectively. After this preprocessing, the total number of particles is reduced to 28,600. The algorithms were tested for both quantities and no reduction in performance was observed when reducing the number of particles. The dataset is selected randomly so there is no bias when performing the classification. The training set, as mentioned above, is 80% of the total particles, so that 22,880 particles served as the training set, while the validation set was the remaining 5720 particles.

Both decision trees and random forest algorithms were fine tuned by making tests in a hyperparameter grid. This grid had elements such as the maximum depth of the tree, the element split criterion, the maximum number of particles per node, the minimum number of particles to make a split, and in case of random forest, the total number of estimators. Starting from the number of estimators in random forest at 100, increasing by 100, the depth of the tree starting at 1 and reaching 20 increasing by 1, the minimum number of particles at 50, up to 200, increasing by 50, thus finding the optimal values in order to avoid blind testing. The optimal hyperparameters are highlighted in Table 1, being the same in almost all values except for the number of estimators, exclusive to random forest. The codes already trained predict the final label of the particles in the test set, which is compared with the real labels in order to obtain the performance of each algorithm. This evaluation was carried out under two tests, the ROC curve along with the AUC of the ROC curve and the learning curve.

5. Dark matter particles classification

Due to the probability distribution obtained for each overdensity range, it is not necessary to perform an extensive preprocessing (see Fig. 7). This figure describes the class distribution (*Not in*

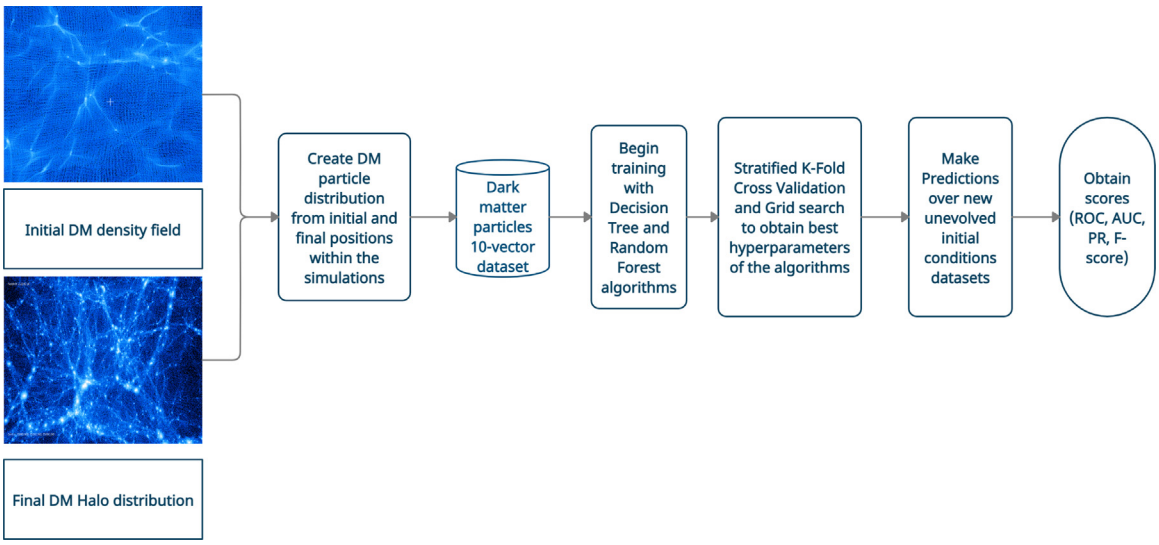

Fig. 6. Diagram of the method to select the properties of the initial density field conditions that will eventually form the structure in the simulation. The process starts from extracting properties of the initial conditions in the local neighborhood of the density field around dark matter particles and associates them to the final position in the halo distribution. The final classification *Not in halo*, *In Halo* depends on the mass threshold chosen to determine whether a dark matter particle will belong in a halo or if it is not bound to any other object Chacón (2021).

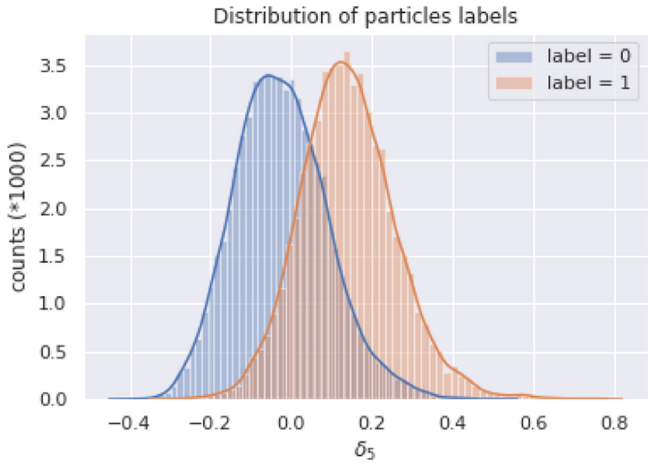

Fig. 7. Histogram of classes for an overdensity obtained in the data preprocessing. The shape of the distribution suggests: (1) It is not necessary to do a data rescaling, since the similarity with a Gaussian curve is evident. (2) The use of the ROC curve metric is sufficient due to the distinction of classes in this range of overdensity values.

Halo: `label = 0`, *In Halo*: `label = 1`) depending on the density contrast δ_i. The overdensities δ_5, δ_6, δ_7 correspond to mass values $1.2 \times 10^{12}\mathrm{M}_\odot$, $2 \times 10^{12}\mathrm{M}_\odot$, $1.1 \times 10^{13}\mathrm{M}_\odot$ respectively and radius R ranging from 3 kpc to 6 kpc, coinciding with the limit that we chose to make a decision ($1.2 \times 10^{12}\mathrm{M}_\odot$). The classification algorithms do not need a rescaling of characteristics since they make decisions through the gain of information, unlike other methods where a subtle difference, for example, the same distance (with different units) can affect the performance of the algorithm. The results are the probability of each class for all particles. That is, the result of belonging to one class or another is determined by a probability threshold value.

After taking this into account, the performance of the algorithms is quantified. A perfect classifier will consist of true positive and true negative values in its confusion matrix. The true positive rate (TPR) and the false positive rate (FPR) are the characteristic quantities of a ROC curve.

The number of particles correctly classified (TPR) and the number of particles incorrectly classified as true (FPR) are shown in Fig. 8. The tests performed for the decision tree gave an accuracy value of 0.77 ± 0.01, with a value of AUC = 0.846. For the random forest, the accuracy was 0.78 ± 0.01 and AUC value = 0.866.

It can be seen in Fig. 8 that TPR decreases as the FPR also decreases. Decision algorithms have been able to predict in a good way whether a particle will end up in a halo or not, depending on the overdensity of the dark matter density field from the initial conditions.

Also as part of the algorithm evaluation, the learning curves of the decision tree and random forest are described in Fig. 9. The upper part corresponds to the decision tree, while the lower part represents the random forest. Both methods adjust their performance well as the number of tests and validation elements increase, reaching a value almost parallel to that reported by the training set. As the training curves neither increase nor the validation curves fall after performing the tests with *cross-validation* it is possible to conclude that both methods are well fitted. Additionally other methods like Logistic Regression and even Naive Bayes were tested, nevertheless the process of decision making of those algorithms does not quite fit the overall result we aim for. The use of Random Forest and Decision Trees has the advantage of being more visual, less biased and with no overfitting when it comes to making decisions for unseen data.

6. Test on new initial conditions

The training and tests sets used on the classification algorithms have been generated during the *N*-body simulations. The advantage is that an evaluation can be carried out on an independent set of initial conditions and the prediction effectiveness can be tested. For this purpose, four new sets of initial conditions were created, these being listed in Table 2.

In three of them the initial seed was changed, which is essentially a pseudo-random creator of numbers that are transmitted to the positions of the particles into the initial conditions. In another simulation, the gravitational smoothing length parameter ϵ was also changed (the first simulation has a value of $\epsilon_1 = 0.89$ kpc). The value of ϵ_1 is not trivial, this arises from the analytical calculation of force and acceleration of a top-hat spherical

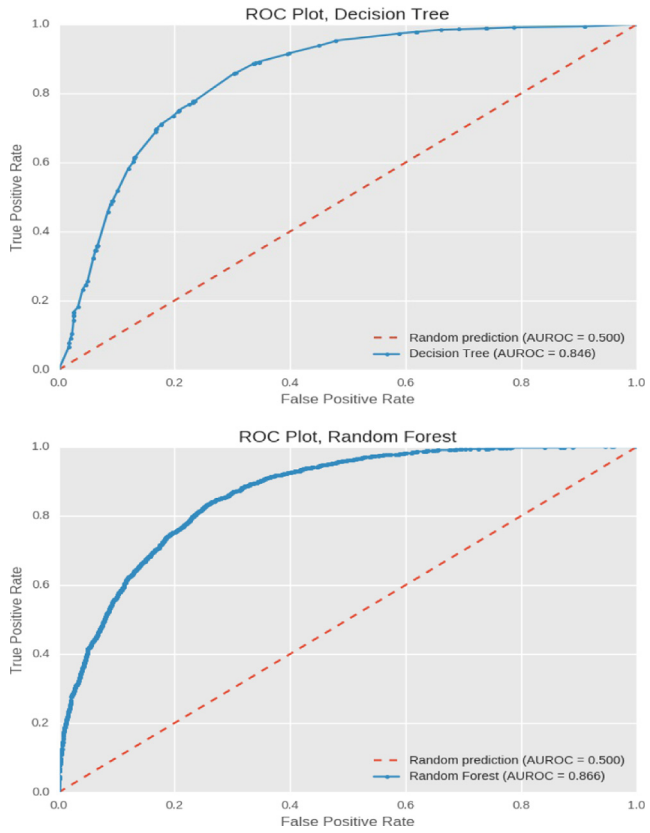

Fig. 8. ROC curves of a decision tree and a random forest algorithm trained in the GADGET simulation. The performance is remarkable given that both have an AUC value ≥ 0.8, highlighting the improvement that random forest has over the decision tree.

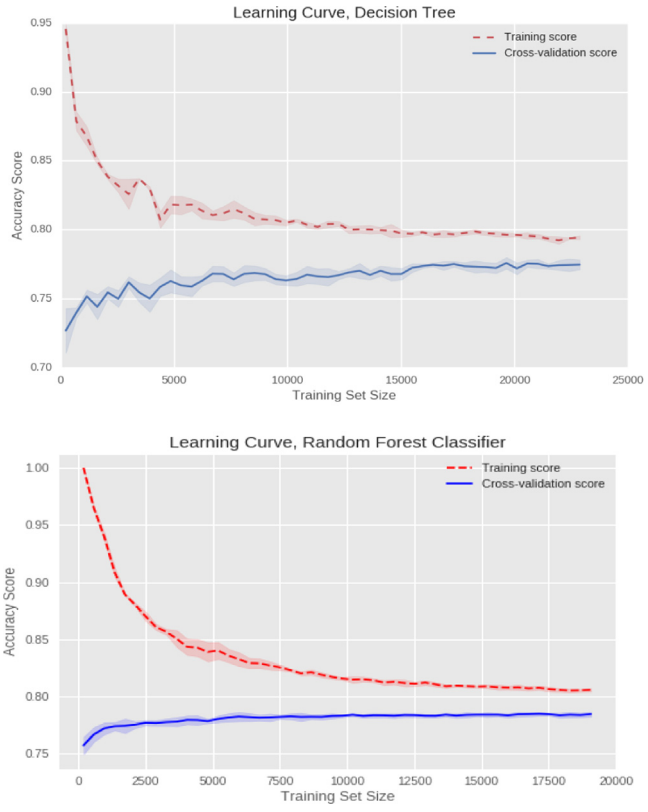

Fig. 9. Learning curves of the decision tree and random forest algorithms. The training curve starts out very high because we have few samples to make a prediction. As the samples increase, the learning curve of the validation set also increases, showing that there is neither overfitting nor underfitting. It is noteworthy that the learning curve of the random forest has less variance, since the low correlation between characteristics prevents a change in this value.

Table 2
Initial conditions for the cosmological simulations.

Description	Symbol	Value
Dark Matter Density	Ω_m	0.268
Dark energy Density	Ω_Λ	0.683
Baryonic Matter Density	Ω_b	0.049
Boxsize	L	50 Mpc
Particle No.	N	192^3
Initial Redshift	z_{init}	23
Final Redshift	z_f	0
Hubble's Parameter	h	0.7
Matter Power Spectrum Normalization	σ_8	0.8
Seed for IC-generator	Seed	100,200,300,400
Another Technical Quantities		
ErrorTolIntAccuracy		0.025
MaxRMSDisplacementFact		0.2
CourantFact		0.15
MaxSizeTimestep		0.03
ErrorTolTheta		0.5
TypeOfOpeningCriterion		1
ErrTolForceAcc		0.005

collapse model. In these calculations, the gravitational softening ϵ_1 widely fits how acceleration between particles in a simulation should be in order to acquire results that are in agreement with other cosmological studies. The three different seeds were determined to add stochasticity to the particle generation in the initial conditions, this stochasticity was required in order to make the decision less biased and to prove the generalization of the decision algorithms.

In the new simulation we changed the smoothing length, by increasing it to $\epsilon_2 = 1$ kpc. Remembering that this length is the minimum distance that two dark matter particles can be together in the simulation (Zhang et al., 2019), the distribution of matter is expected to change, which can be corroborated with the mass power spectrum, observed in Fig. 10. The properties around the particles in the dark matter density field were again extracted and a new evaluation of the performance of the decision tree and random forest was carried out.

Fig. 11 shows the comparative ROC curve of the two algorithms in the realization with the new gravitational smoothing ϵ, when training and testing them with the data from the initial simulation, as well as when doing the test with the new initial conditions, without carrying out a complete computational run. The upper part shows the performance of the decision tree in the previous training and testing set and the prediction for the new initial conditions. The bottom part shows the same for the random forest. Decision algorithms produce consistent ROC curves for the new set of initial conditions. The AUC on both approaches fell $\sim 2\%$ since there was less structure formation.

On the other hand, the change of seed realizations had a performance similar to that described in the previous paragraph. We had the ΛCDM simulation as a training set, and the performance was tested on the new initial conditions. Though the complete simulations were not executed, the algorithms were able to identify the classification of dark matter particles that fell or not into halos of dark matter given a threshold value. Fig. 12 shows the performance of the decision tree in the upper part, and the random forest in the lower part. Both algorithms perform proficiently with their ΛCDM simulation training counterparts. The only change in the initial conditions of these new realizations

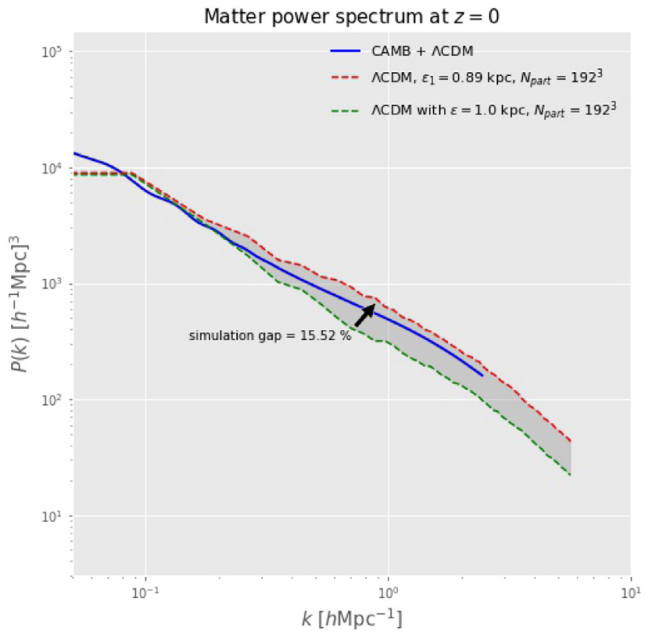

Fig. 10. Matter power spectrum of the new initial conditions ($\epsilon_2 = 1.0$ kpc) and the simulation above; the spectrum obtained with CAMB is shown in the solid line (Lewis and Bridle, 2002). The difference between both simulations is indicated in the figure and is approximately 15%. The power spectrum was obtained in the same way as in the previous realizations. It is evident that the distribution of matter for the new conditions is different, since there is less structure formation.

was in the pseudo-random seed generator, so that the predictive power of the algorithms becomes more evident with this figure.

In Fig. 13 we can see the Precision–Recall scores obtained for the decision algorithms (decision tree and random forest), with Precision defined as

$$Precision = \frac{TP}{TP + FP}, \tag{10}$$

and Recall defined as

$$Recall = \frac{TP}{TP + FN}. \tag{11}$$

We can see that Recall is another name for False Positive Rate. As we know, precision and recall both indicate accuracy of the model. Precision means the percentage of the results which are relevant, while recall refers to the percentage of total relevant results correctly classified by the algorithms. The AUC of the PR curve for both decision tree and random forest are between [0.76, 0.82], and [0.78, 0.84], respectively. The result given suggests that the learning process did have a good trade-off between precision and recall across all different IC seeds. We therefore conclude that the overall accuracy had no impact while making a small change in the initial conditions. Additionally, the F1-score, defined as

$$F_1 = 2\frac{Precision \cdot Recall}{Precision + Recall}, \tag{12}$$

and F_β score defined as

$$F_\beta = (1 + \beta^2)\frac{Precision \cdot Recall}{\beta^2 Precision + Recall}, \tag{13}$$

where $\beta = 0.5$ is chosen such that Recall is considered β times as important as Precision, were calculated. Table 3 lists the averaged weighted F-scores. This result determines the overall good performance by both algorithms and remarks better results obtained by the random forest.

Table 3
F-score of decision algorithms.

Algorithm	F_1	F_β
Decision Tree	0.775	0.777
Random Forest	0.780	0.0.781

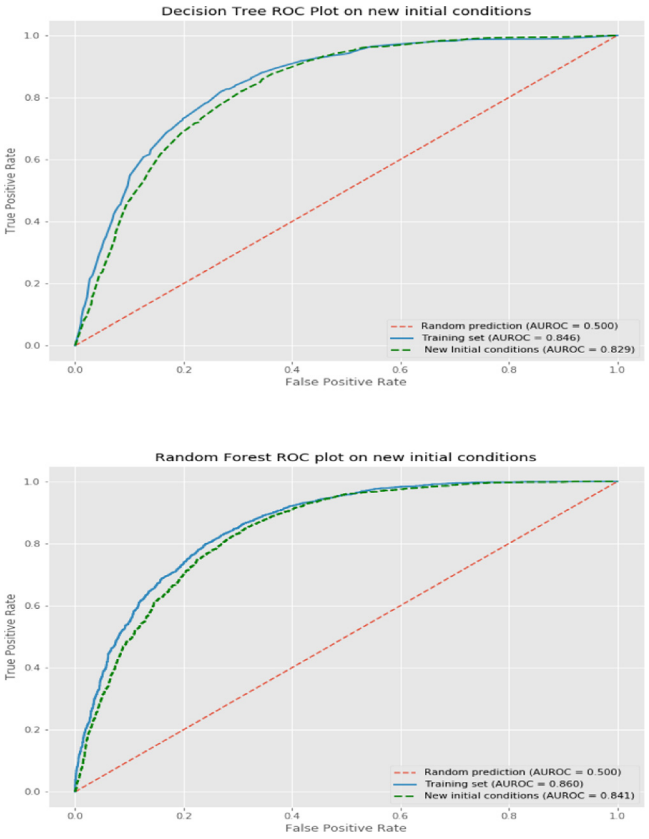

Fig. 11. ROC curves of the decision tree and random forest algorithms of the initial conditions with a new gravitational smoothing ϵ, compared to the performance previously shown. The curves are fairly consistent. The value of the AUC fell $\sim$ 2%. The tests demonstrate the great capacity of the algorithms to predict the final labels of different simulations.

Labels predicted by the machine learning algorithms are computed from the density properties of the initial conditions. In simulations, the algorithms are able to predict the final classification result with fairly good accuracy. By carrying out a new test for the different initial conditions, without running the full simulation, both methods were able to predict the final label accurately. Granted a non negligible change in initial conditions did affect the outcome.

7. Final discussion and future work

Throughout the process, several factors emerged that could be decisive on the results. One of the most relevant was the use of a simulation and its probable deviation or bias due to different numerical parameters independently of the physical processes. As is well known, simulations require knowing many numerical parameters in addition to knowing in depth the code. Even though we were not delved into these aspects, it should be a priority to have a better certainty that the results obtained can be comparable with a workable physical system and have no preference for bias or variance.

It has been emphasized that the algorithm training process has to be more refined, since the number of selected particles

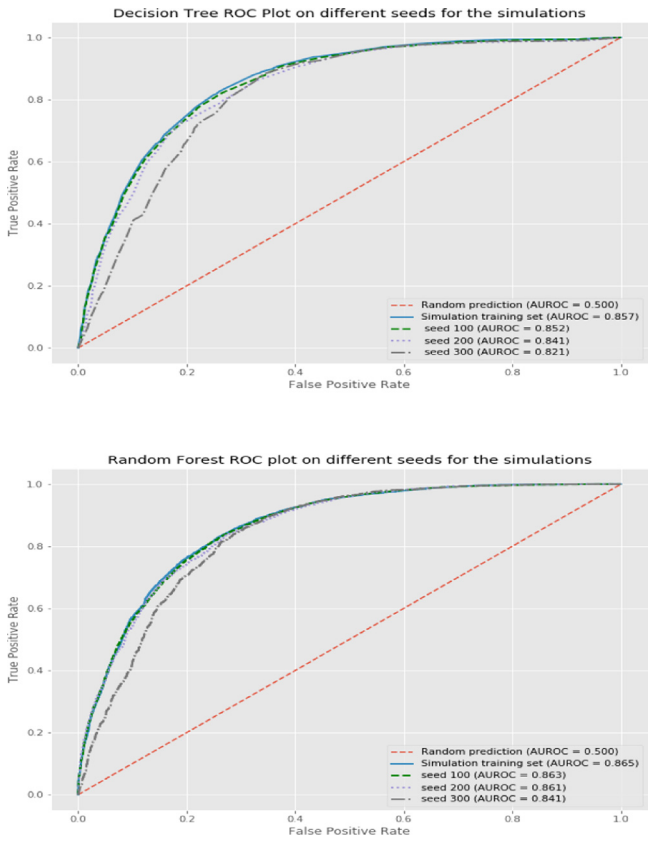

Fig. 12. ROC curves of the decision tree and random forest algorithms of the initial conditions whose seeds were different. The AUC drops an average of 2.2% for new realizations. The generalization of the predictive power of the training is evident since the algorithms are able to decide in a good way the final destination of the dark matter particles from their position at an initial moment.

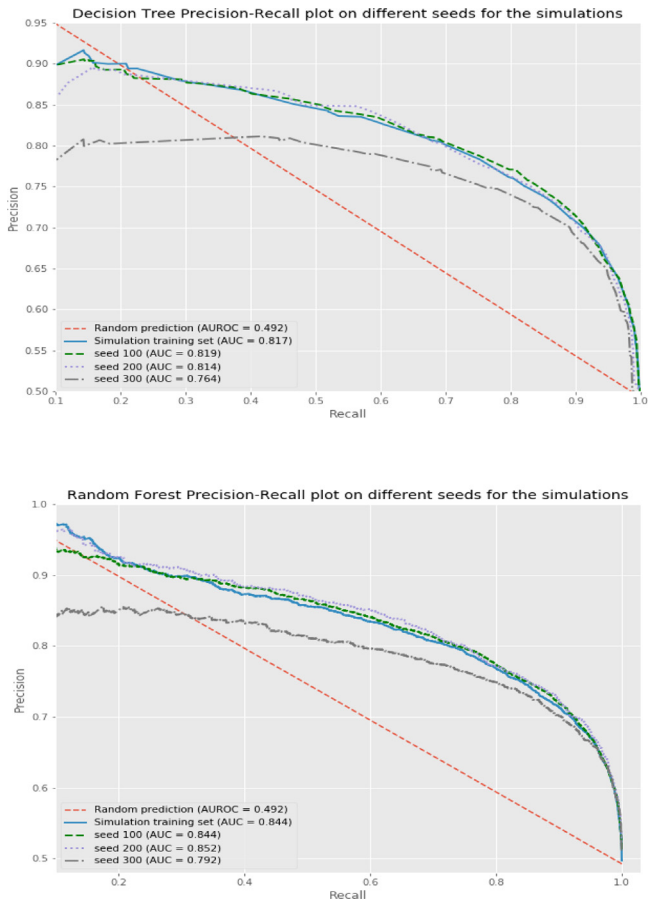

Fig. 13. Precision–Recall scores obtained in the learning process of decision algorithms. We see that for a particular seed, the decision tree Precision drops up to 0.78, this is likely because there were more cases of False Negatives in this dataset, whereas the random forest only drops to 0.85, meaning that the random forest performs generally better than the decision tree. The figure suggests a good accuracy performance on the new seeds generated for the training process, resulting in a reliable algorithm that serves as our binary classifier for dark matter particles within the simulation.

represents a minor contribution of the total number of particles within the simulation. This can certainly be decisive since at the end there was an array of approximately 317,680 elements that should have been used as training data. Even so, another test was performed for a choice of 57,000 particles, performing the procedure described in Section 4.2. The AUC of the algorithms used did not show an improvement since both realizations have similar results (0.85 for decision tree and 0.86 for random forest). This fact shows that the use of a larger volume of particles is not decisive in the identification process.

Finally, the aim of this work was to show how we can use Tree-like decision algorithms to aid cosmological simulations and predict the outcome of a future run without the need of evolving dark matter particles with codes that consume a lot of time, giving results that are similar to the ones obtained in a full run. Furthermore, we saw that using less data, we obtained a good overall result in predictions for new initial conditions datasets, resulting in models that are able to learn the relationship between the initial conditions (position and region of overdensity) of the dark matter particles and their final position within halos given a certain threshold mass. Additionally, the data used for this work can be retrieved from our GitHub repository,[1] in which the data and code necessary to perform our analysis is available. We encourage the reader to visit this site and perform their own tests.

[1] GitHub ChJazhiel.

7.1. Numerical simulations assisted with artificial intelligence

There is another alternative to the complete realization of a numerical simulation, using Generative Adversarial Networks (GAN). These networks basically take databases or images from which the algorithm generates two networks, a generator and a discriminator. The networks begin a competition between each other, both networks were trained with the same data set, but the first must try to create variations of the data that it has already seen. The discriminatory network must identify whether the image created is part of the original training or is a false image that the generative network created. The more datasets generated the better the generative network is at creating them, and the more difficult it is for the discriminatory network to identify whether the image is real or false. The generative network needs the discriminator to know how to create an imitation so realistic that the second one cannot distinguish it from a real image.

In this regard, numerical simulations come in handy, because the displacement density field is shown as a 3-dimensional image with 3 channels, each channel corresponds to the displacement vector, the deep learning model takes the displacements of the low-resolution simulation and generates a possible high-resolution realization, so this result can be seen as a high resolution simulation with more particles and higher mass resolution

(Li et al., 2020). A goal in the future will be understanding and implementing deep learning frameworks that can yield better resolution simulations without requiring more computational time and resources, and obtaining results similar to the ones obtained in the numerical code method.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

J.A.V. acknowledges support from FOSEC SEP-CONACYT Ciencia Básica A1-S-21925, FORDECYT-PRONACES-CONACYT 304001 and UNAM-DGAPA-PAPIIT IA104221.

J. Ch. would like to thank Sebastien Fromentau and Octavio Valenzuela who offered feedback about the process for this work. Additionally, the author acknowledges support from project CONACYT, Mexico 282569, February–July 2019.

References

Agarwal, S., Davé, R., Bassett, B.A., 2018. Painting galaxies into dark matter haloes using machine learning. Mon. Not. R. Astron. Soc. 478, 3410–3422.

Behroozi, P.S., Wechsler, R.H., Wu, H.-Y., 2013. The ROCKSTAR phase-space temporal halo finder and the velocity offsets of cluster cores. Astrophys. J. 762, 109.

Breiman, L., 2001. Random forests. Mach. Learn. 45, 5–32.

Buncher, B., Carrasco Kind, M., 2020. Probabilistic cosmic web classification using fast-generated training data. Mon. Not. R. Astron. Soc. 497, 5041–5060.

Chacón, J., 2021. Algoritmos de clasificación aplicados a simulaciones de formación de estructura cosmológica (Master's thesis). Universidad Nacional Autónoma de México.

Chacón, J., Vazquez, J., Gabbasov, R., 2020. Dark matter with n-body numerical simulations.. arXiv e-prints, arXiv:2006.10203.

Cheng, T.-Y., Huertas-Company, M., Conselice, C.J., Aragón-Salamanca, A., Robertson, B.E., Ramachandra, N., 2021. Beyond the hubble sequence – exploring galaxy morphology with unsupervised machine learning. Mon. Not. R. Astron. Soc. 503, 4446–4465.

Cheng, T.-Y., Li, N., Conselice, C.J., Aragón-Salamanca, A., Dye, S., Metcalf, R.B., 2020. Identifying strong lenses with unsupervised machine learning using convolutional autoencoder. Mon. Not. R. Astron. Soc. 494, 3750–3765.

D'Addona, M., Riccio, G., Cavuoti, S., Tortora, C., Brescia, M., 2021. Anomaly detection in astrophysics: A comparison between unsupervised deep and machine learning on kids data. In: Emergence, Complexity and Computation, pp. 225–244.

Dodelson, S., A.P.L. 1941–1969, 2003. Modern Cosmology. Elsevier Science.

Fawcett, T., 2006. Introduction to roc analysis. Pattern Recognit. Lett. 27, 861–874.

Geach, J.E., 2011. Unsupervised self-organized mapping: a versatile empirical tool for object selection, classification and redshift estimation in large surveys. Mon. Not. R. Astron. Soc. 419, 2633–2645.

Gómez-Vargas, I., Vázquez, J.A., Esquivel, R.M., García-Salcedo, R., 2021. Cosmological reconstructions with artificial neural networks. arXiv e-prints, arXiv:2104.00595.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. The MIT Press.

Gron, A., 2017. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, first ed. O'Reilly Media, Inc..

Hajian, A., Alvarez, M.A., Bond, J.R., 2015. Machine learning etudes in astrophysics: selection functions for mock cluster catalogs. J. Cosmol. Astropart. Phys. 2015, 038.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning: Data Mining, Inference and Prediction, second ed. Springer.

Hocking, A., Geach, J.E., Sun, Y., Davey, N., 2017. An automatic taxonomy of galaxy morphology using unsupervised machine learning. Mon. Not. R. Astron. Soc. 473, 1108–1129.

James, G., Witten, D., Hastie, T., Tibshirani, R., 2014. An Introduction to Statistical Learning: With Applications in R. Springer Publishing Company, Incorporated.

Kamdar, H.M., Turk, M.J., Brunner, R.J., 2016. Machine learning and cosmological simulations – ii. hydrodynamical simulations. Mon. Not. R. Astron. Soc. 457, 1162–1179.

Lewis, A., Bridle, S., 2002. Cosmological parameters from CMB and other data: A Monte Carlo approach. Phys. Rev. D 66, 103511.

Li, Y., Ni, Y., Croft, R.A.C., Di Matteo, T., Bird, S., Feng, Y., 2020. AI-assisted super-resolution cosmological simulations. arXiv e-prints, arXiv:2010.06608.

Louppe, G., 2014. Understanding random forests: From theory to practice. arXiv e-prints, arXiv:1407.7502.

Lucie-Smith, L., Peiris, H.V., Pontzen, A., Lochner, M., 2018. Machine learning cosmological structure formation. Mon. Not. R. Astron. Soc. 479, 3405–3414.

Moster, B.P., Naab, T., Lindström, M., O'Leary, J.A., 2020. GalaxyNet: Connecting galaxies and dark matter haloes with deep neural networks and reinforcement learning in large volumes. arXiv e-prints arXiv:2005.12276.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.

Perraudin, N., Srivastava, A., Lucchi, A., Kacprzak, T., Hofmann, T., Réfrégier, A., 2019. Cosmological n-body simulations: a challenge for scalable generative models.

Press, W.H., Schechter, P., 1974. Formation of galaxies and clusters of galaxies by self-similar gravitational condensation. Astrophys. J. 187, 425–438.

Quinlan, J.R., 1986. Induction of decision trees. Mach. Learn. 1 (1), 81–106.

Rokach, L., Maimon, O.Z., 2008. Data Mining with Decision Trees: Theory and Applications. vol. 69, World Scientific.

Seni, G., Elder, J., 2010.

Shannon, C.E., 1948. A mathematical theory of communication. Bell Syst. Tech. J. 27 (3), 379–423.

Sharma, R., Mukherjee, A., Jassal, H.K., 2020. Reconstruction of late-time cosmology using principal component analysis. arXiv e-prints, arXiv:2004.01393.

Springel, V., 2005. The cosmological simulation code GADGET-2. Mon. Not. R. Astron. Soc. 364, 1105–1134.

Xu, X., Ho, S., Trac, H., Schneider, J., Poczos, B., Ntampaka, M., 2013. A first look at creating mock catalogs with machine learning techniques. Astrophys. J. 772, 147.

Zhang, T., Liao, S., Li, M., Gao, L., 2019. The optimal gravitational softening length for cosmological N-body simulations. Mon. Not. R. Astron. Soc. 487, 1227–1232.