Sharif University of Technology

Physics Department

Machine Learning In Physics

Termpaper

Supervisors: Dr. Sadegh Raeisi & Dr. Shant Baghram

Members: Ali Saraer, Amirhossein Samandar

Fatemeh Farhangian, Armita Kazemi

Fall Semester 00-01

Contents

Abstract

Massive neutrinos have been an item of interest as a candidate for the all or some of the dark matter in the observable universe. But, they are hard to account for in analytical solutions of Einstein and Boltzmann equations, due to the complexities they present in the longitudinal and traceless part of the spatial Einstein equation as a none-zero anisotropic stress, described by the following equation:

$$k^2(\Phi + \psi) = -32\pi G a^2[\rho_\nu \mathcal{N}_2] \tag{1.1}$$

in which k is the wave number of the perturbations in Fourier space, Φ and ψ are perturbations to the FLRW metric, ρ_ν is neutrino density, and $\mathcal{N}_2$ is the quadrupole moment of neutrino temperature.[1] They also make the Friedmann equations much more complicated, since they have a mass of their own, and can contribute to the evolution of Hubble constant. But, most importantly, they have a direct effect on the matter power-spectrum, which in turn changes the evolution of primordial dark matter and changes the structure of today's dark matter distribution quite considerably.

The matter power-spectrum is as follows[1]

$$P_L(k,a) = \frac{8\pi^2}{25}\frac{\mathcal{A}_s}{\Omega_m^2} D_+^2(a) T^2(k) \frac{k^{n_s}}{H_0^4 k_p^{n_s-1}} \tag{1.2}$$

In which $\mathcal{A}_\int$, Ω_m, $D_+(a)$, $T(k)$, n_s, H_0 are the perturbation amplitude, density parameter, growth function, transfer function, spectral index, and Hubble constant in the present, respectively.

The part that we are most focused on, is the growth function and the transfer function. The transfer function can be derived from the considerations of horizon crossing and solving the Einstein and Boltzmann equations in the

[1] In equation 1.1 we have neglected the quadrupole moment of photons, since their effect is negligible.

equality era. The growth function can be derived from the following equation

$$\frac{d^2\delta_m}{da^2} + \frac{d(ln(a^3H))}{da}\frac{d\delta_m}{da} - \frac{3\Omega_m H_0^2}{2a^5H^2}\delta_m = 0 \tag{1.3}$$

where δ_m is matter density contrast and is proportional to the growth function $D_+(a)$. The growth function and the transfer function are highly contingent upon the underlying components of the universe. So, to study the dark matter evolution, neutrinos potentially play a major role and cannot be neglected at the drop of a hat. Therefore, it does not come as a surprise when it is posited that any study of today's dark matter structure is incomplete if neutrinos' effect on matter power-spectrum, especially massive ones, are neglected. In this paper, we tried to outline a project, that will fill the gap in the long-run

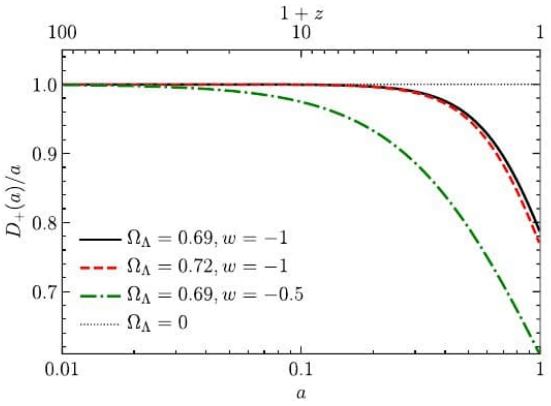

Figure 1.1: Growth function dependence on the components of the universe.

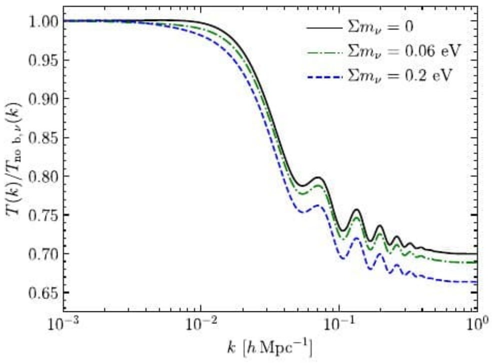

Figure 1.2: Transfer function dependence on the components of the universe.

Introduction

Understanding the structure evolution of dark matter gets quite complicated when neutrinos are involved. All the equations cannot be solved in the linear domain. However, there is an elegant, and yet, simple solution that goes a long way in assisting us in grappling with the much needed intuition before encountering all the intricacies of coding and algorithms. This simple idea was first put forward by Press and Schechter's paper (1974)[2]. In this paper, they answered the following question: What is the fraction of space (in the initial conditions) that is contained in collapsed halos above mass M at redshift z?

Press and Schechter argued that, since the linear density field smoothed on a comoving scale $R_L(M)$ follows a Gaussian with zero mean and variance $\sigma(R_L, z)$, the volume fraction should simply be the integral over this Gaussian from the collapse threshold to infinity:

$$\begin{aligned} F_{coll,PS}(M,z) &= 2 \times \frac{1}{\sqrt{2\pi}\sigma(R_L[M],z)} \int_{\delta_c r}^{\infty} d\delta e^{-\delta^2/2\sigma^2(R_L[M],z)} \\ &= 2 \times \frac{1}{\sqrt{2\pi}} \int_{\delta_c r/\sigma(R_L[M],z)}^{\infty} d\nu e^{-\nu^2/2} \end{aligned} \tag{2.1}$$

This equation will lead to the following, which describes the halo-mass function of the universe's structure.

$$\begin{aligned} \frac{dn(M,z)}{dlnM} = \frac{\rho_m(t_0)}{M} f_{PS}(\frac{\delta_c r}{\sigma(M,z)})|\frac{dln\sigma(M,z)}{dlnM}| \\ , f_{PS}(\nu) = \sqrt{\frac{2}{\pi}} \nu e^{-\nu^2/2} \end{aligned} \tag{2.2}$$

Note that δ_{cr} in equations 2.1 and 2.2 describe the value of which if density contrast at a point in space is greater, the overdensity in that region becomes greater at a later t and collapse happens, and is called the critical density contrast. For a spherical collapse model, we have the following value for the critical density contrast

$$\delta_{cr} \approx 1.686 \tag{2.3}$$

The important consequence of the Press-Schechter(PS) model is that the cold dark matter halo structure can be fully explained only by knowing the density contrast distribution in the initial conditions of the universe, for a redshift of $z > 100$. However, PS might not predict the halo structure when the dark matter contains components other than that of cold particles. Building on PS, we are trying to integrate neutrinos into the picture and investigate whether their velocity field is an important feature in understanding the halo structure. If this turns out to be true, we will get one step closer in comprehending the dark matter present in our universe, and potentially find a good place for neutrinos in the pecking order of different dark matter candidates considered by physicists.

In the following chapters, we first define our problem in the confinements understandable by a computer using machine learning methods. Then, we discuss how G-evolution simulation data were manipulated to extract the features whose vitality were explained above in detail. Then, we go on to discuss our models, which were the fundamental machine learning algorithms such as KNN, SVM, etc., and discuss how neural networks(NN) make life much easier and are viable in a more generalized fashion. Then we conclude by noting that this is by no means a finished work, and it should be considered an on-going research, with which we fully intend to continue.

Defining The Problem

Initial idea of this project comes from the role that massive neutrinos are conjectured to play in the evolution of dark matter halos from an initial condition of $z > 100$, up to the present universe, i.e. $z = 0$. In more accurate terms, this translates to the ability of a machine to identify where each particle from the initial universe structure would end up, namely in a halo or not. Or, to put it another way, the machine must identify that each particle's encompassing halo is massive enough(above a mass threshold of about $10^{11} M_{\odot}$) to be considered a viable dark matter halo.

A more broad generalization of this problem is for the machine to exactly identify to which halo each particle belongs. An even more accurate method, which is the end-game goal of this project, is to identify what the final halo-mass of each particle's encompassing halo would be, and to find the halo-mass function of the present universe. Depending on the preference of the applied method, its algorithm will be binary classification, multi-class classification, or regression, respectively.

Since the complications involved in binary classification are evidently more manageable, their result are the most accurate, and at the same time, the least valuable.

A robust method to evaluate the falsifiability of our models is to run the algorithms developed without considering neutrinos and to compare our models with the results of the the halo-mass function derived by the previous group who studies the effects of Cold Dark Matter(CDM), namely, no neutrinos involved.

Data Collection & Analysis

The bulk of the data are the information about the Cartesian coordinates, and the velocity vector of each particle, which are extracted from the snapshots taken from the redshift of our choosing using G-evolution code developed by Dr. Farbod Hassani.

For instance, for a redshift of $z = 127$, we have a big bulk of information about every single particle, from which we need to find the halo in which they will end-up at $z = 0$.
To identify the halos(which is similar to a clustering problem), we use a halo-finder module. This module takes a snapshot of the universe[1] at $z = 0$ (much like what we described in the previous paragraph) and identifies the separate halos, and lists each halo's features, such as its Virial mass, its center of mass coordinates, number of particles it contains, etc. The halo-finder also identifies in which halo the particle ends up in.[2]

Now, the case for the first algorithm, namely binary classification, is very easy to make. We only need to define a halo-id as follows: we assign a value of 0 to this variable if the particle is not encompassed by a halo, or is contained in a halo with mass less than the threshold mass M_{thr}.[3] and assign a value of 1 whenever a particle is inside a halo with a mass greater than M_{thr}.

For the multi-class classification algorithm, we amend the halo-id variable defined in the previous paragraph so now, instead of taking only zeros and ones as its values, it can take any number from 0 (which is defined similar as before) up to the total number of halos we have found in our halo-finder. It is hence obvious that this algorithm is considerably more complicated than that of binary classification, since we suddenly go from 2 classes to an order of 1000 classes, which is much more complicated without a doubt. Obviously, the same problem casts its shadow on the regression algorithm as well. Also, since there is a bias in the number of generated halos with small mass

[1] here we limited our model to a cubical box of side 160MPc

[2] Note that the particle may also not end up in any halo at all, or end up in a halo with so little mass that we consider that particle not being contained in a halo.

[3] which we should put in before we run the algorithm and it is a hyper-parameter of our problem.

compared to massive halos, we need to designate a threshold mass, similar to the binary algorithm, and to run a different algorithm for particles belonging to halos with mass less than M_{thr} than that of particles in halos with mass greater than M_{thr}.

To train the machine, we need to make our definition of samples and features more accurate. Each sample signifies a particle in our initial snapshot of the universe($z = 127$). And our features are each particle's coordinates x, y, z and its velocity vector v_x, v_y, v_z and density contrast calculated in cubes of side r with different values for r (e.g. $r = 4.9$, 6.8, 8.7, $\dots$) centered around the particle and calculated using the following equation

$$\delta = \frac{\rho(r) - \bar{\rho}}{\bar{\rho}} \tag{4.1}$$

In which δ, $\rho(r)$, and $\bar{\rho}$ represent density contrast, density in a box of side r, and the background density, respectively.

A caveat of calculating the density contrasts is that we need to find every particle that is contained in the box of side r by finding ones that have values of x, y, z greater than $x_0 - r/2$, $y_0 - r/2$, $z_0 - r/2$ and less than $x_0 + r/2$, $y_0 + r/2$, $z_0 + r/2$, in which x_0, y_0, z_0 denotes the coordinates of the particle around which we drew the box. At first glance, the first idea that comes to mind is brute force, which takes time of order n^3, in which n is the number of our particles, which is around 20 million at the very least, and can up to a few days to complete! A better method then is to first sort the particles with respect to one of their coordinates, say their x coordinates, which takes time of order $n\, log\, n$ and then use binary search, which is of order $log\, n$ for this coordinate, and use the brute force for the remaining two. This method takes time of order $O(n\, log\, n + n^2\, log\, n) \propto O(n^2\, log\, n)$, which takes less than an hour to complete.

Also, since there was a large discrepancy between our values, we needed to use a standard scaler before we trained models such as SVM (explained in more detail in the next chapters).

Since there was a significant delay for us to receive the data we needed from the halo-finder, namely the halo-ids, we had to improvise and assign a random id of zero or one and run the binary classification algorithms on our fictitious data in the second phase, and after we received the data, we replaced the random generator with our real data and have ran the algorithms again. Also, we used the data from the previous project(the one that dealt with CDM), so there was not a problem in cross-examination of the models.

Phase One

We have explained this part in full detail here.

In this phase, we had access to less number of particles as our raw data(recall that our data consisted of particles' coordinates and velocity vector). We also knew the halo-mass function at $z = 0$. This was enough to draw the diagrams, and dot the i's and cross the t's.

As we have already exposited in the previous chapter, density contrasts and the velocity magnitude have been extracted as our physical features with which we intend to train our models. There are two pros in using the density contrasts as our features, one is that the problem becomes more physical and makes sense intuitively, and the other is that data are compatible with our physical assumptions, namely, PS and Extended PS theories.

5.1 Diagram Analysis

Figure 5.1 demonstrated the halo-mass histogram from the simulations. In solving the problem with multi-class classification algorithm, we need to identify each class with intervals that put approximately equal number of halos in each class, to make the training more accurate and effective. The big numbers on the axes of figure 5.1 readily suggest that the logarithm of halo masses is a better choice for the class division.

Figure 5.2 shows a box that describes the particle distribution at $z = 127$. Each particle is denoted by a different color depending on their velocity magnitude.

To make the velocity field more visually appealing, we scale the velocity magnitudes and redraw the box like figure 5.3

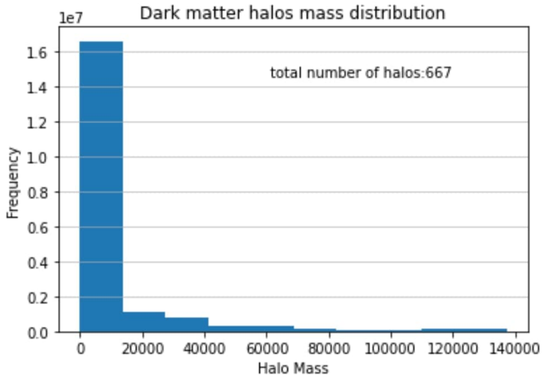

Figure 5.1: Halo-mass histogram

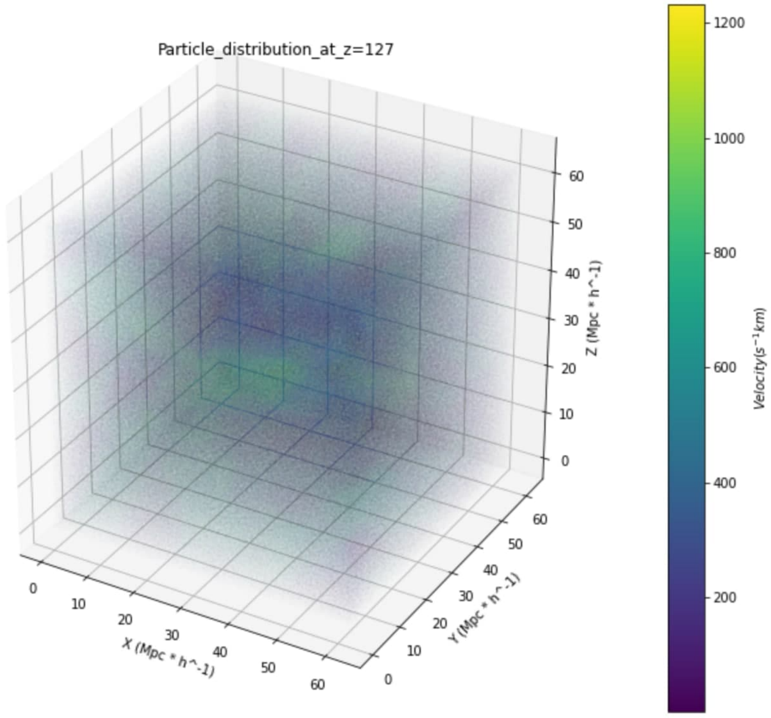

Figure 5.2: particle distribution at initial z ($z = 127$. Note that the universe is homogeneous at large z's)

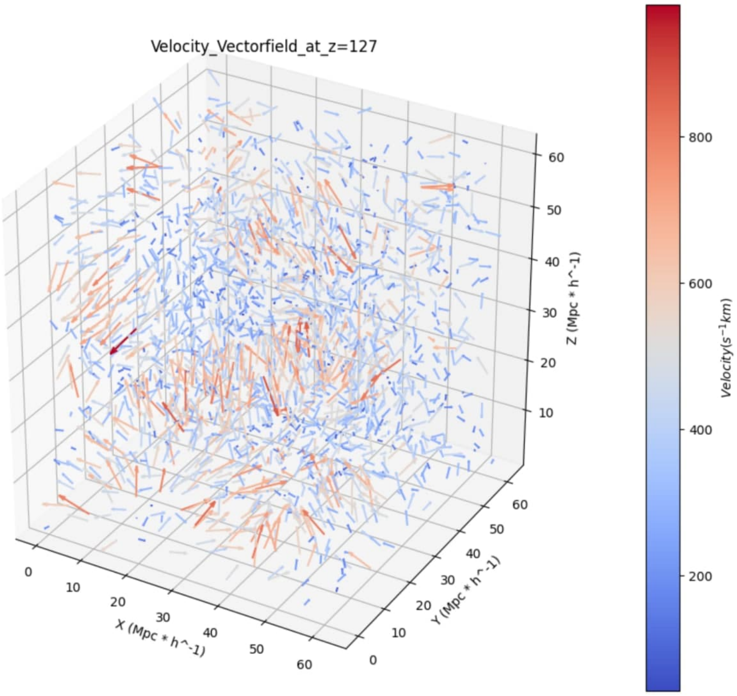

Figure 5.3: Scaled velocity field of particles in the initial universe.

Phase Two

Here, we employed the classic methods of machine learning, namely, SVM, KNN, Logistic Regression, Random Forest, and Decision Tree. As is evident in tables 6.1 and 6.2, Random Forest and Decision Tree had fairly good accuracies and operation times. Conversely, SVM took a long time to finish. Its accuracy did not do justice to the amount of time it took to come to fruition. To train each of the models, we once used the data from the model assuming only CDM and once from the model that assumed neutrino presence.

6.1 Model Analysis

Due to the big range of our data, we used a standard scaler so that our models could handle the data more efficiently. Logistic regression needed the most time to train, but its required prediction time was fairly low. Obviously, a model's superiority is not contingent upon its speed during training time. It depends on the amount of time it consumes to render a solid prediction. Therefore,even though the training time of logistic regression was quite large, it is one of the better models when speed is concerned. However, it falls in the accuracy and f_1 score behind random forest and decision tree algorithms. From the tables, it is observed that random forest is the best algorithm in this category and its prediction speed is also quite good. So, it is fair to say that random forest is the best of the bunch.

Moreover, the classification results show thet neutrino data is harder to classify, because the overall mass is decresed and there are less in-halo particles in the neutrino dataset. Therefore the classification is harder due to unbalanced data.

estimator	train_time	pred_time	accuracy_test	accuracy_train	f1_test
KNN	2.09	58.364	0.59	0.624	0.644
RandomForest	491.004	4.44	0.848	0.99983	0.869
SVM	450.998	98.832	0.584	0.589	0.532
LogisticRegression	2282.755	0.118	0.769	0.77	0.764
DecisionTree	1.604	0.005	0.742	0.916	0.7677

Table 6.1: Model Comparision Table: Neutrino

estimator	train_time	pred_time	accuracy_test	accuracy_train	f1_test
KNN	0.055	6.863	0.586	0.625	0.685
RandomForest	468.946	3.617	0.87	0.99976	0.895
SVM	3987.331	105.331	0.584	0.589	0.532
LogisticRegression	2374.528	0.123	0.766	0.765	0.751
DecisionTree	3.915	0.014	0.774	0.997	0.809

Table 6.2: Model Comparison Table: CDM

To understand models in more detail, we include the validation/Learning curves of all the classic models.

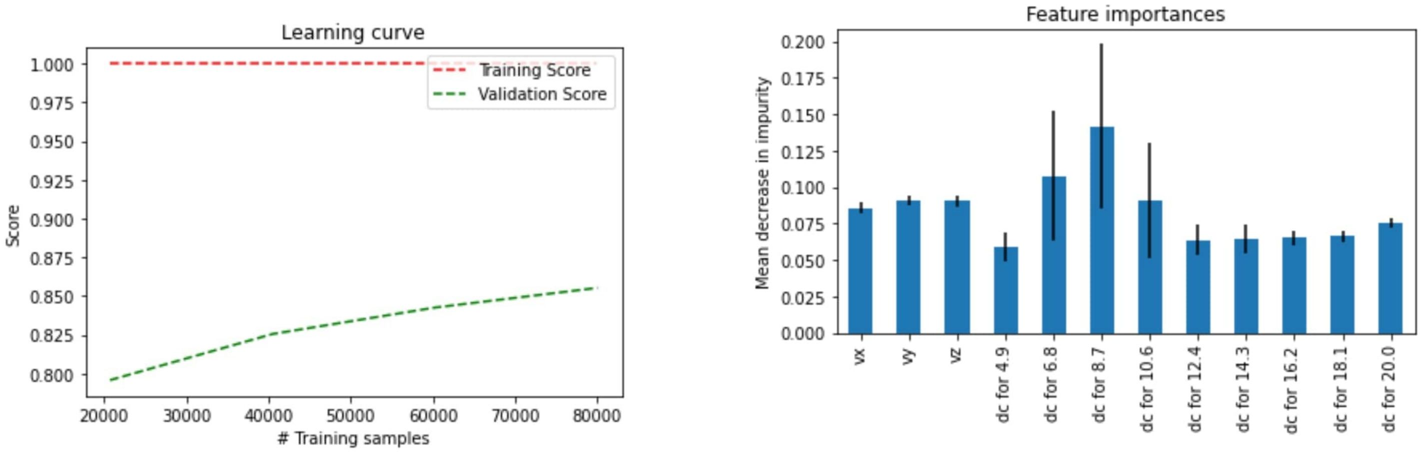

Figure 6.1: Random Forest

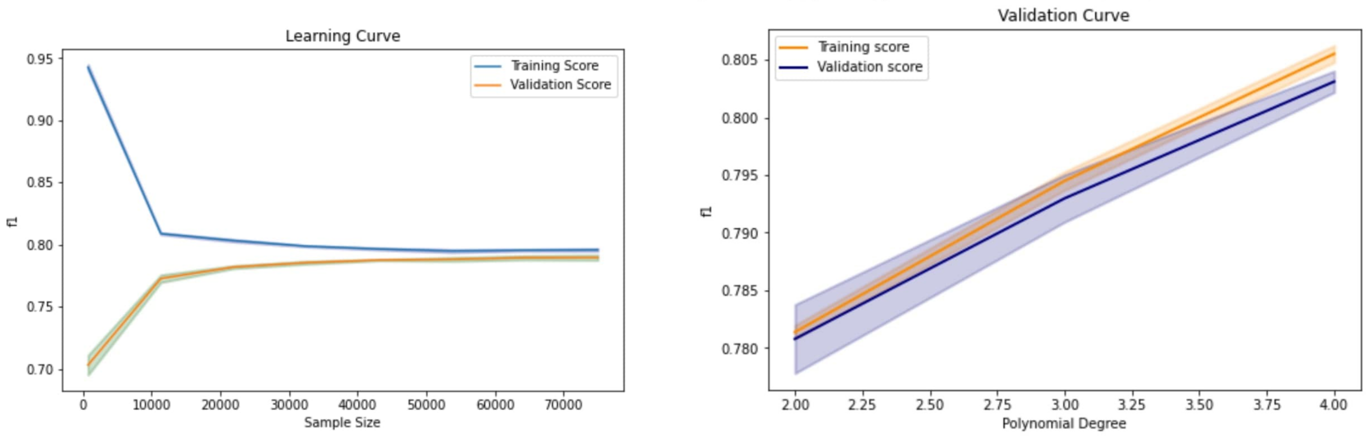

Figure 6.2: Logistic Regression

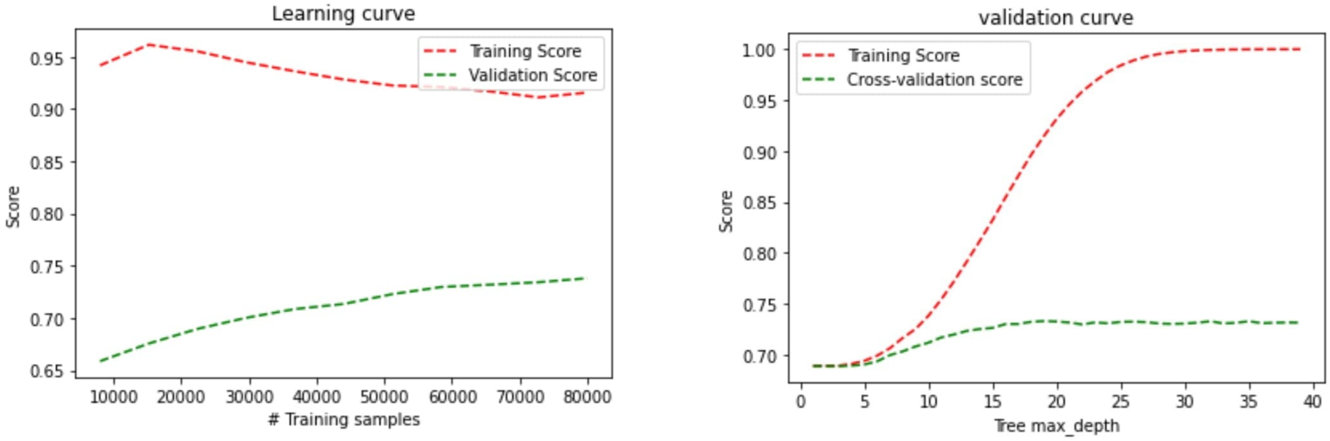

Figure 6.3: Decision Tree

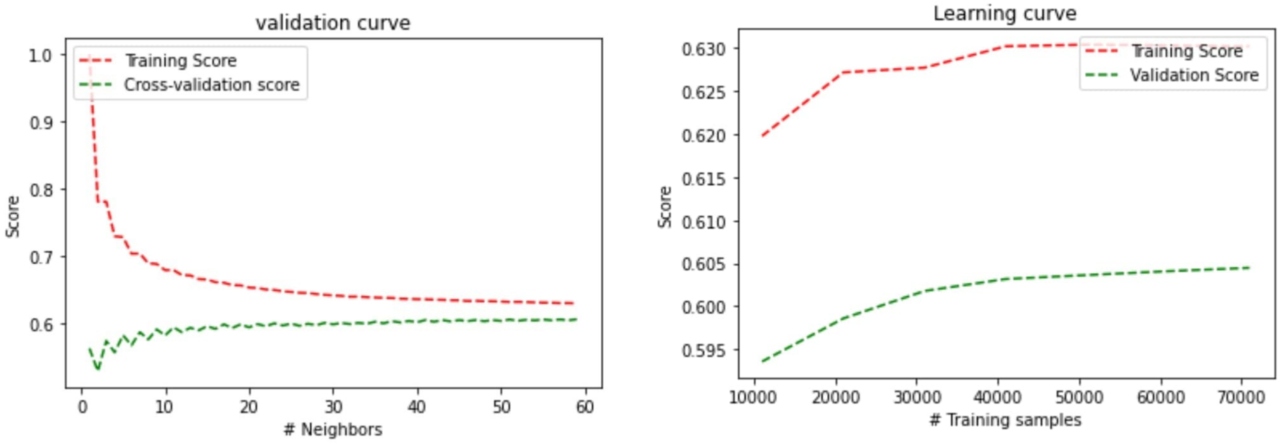

Figure 6.4: KNN

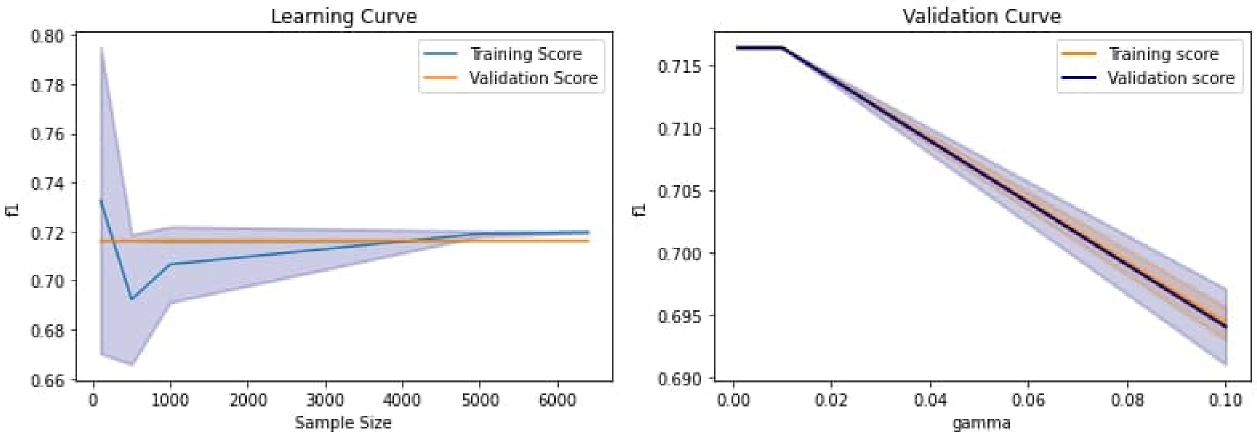

Figure 6.5: SVM

Phase Three

This phase was very similar to the previous one, with the main difference being the difference in the training method employed, namely neural networks(NN). We trained NN for the binary, multi-class classification, and regression problems separately. As was expected, the results are more accurate for the binary case. In the multi-class problem, to reduce the complexities of having many classes, we divided them in five groups, and, each halo falls on either group depending on its mass. We use the PyTorch and Keras libraries to run our NN algorithms. Details and codes are available here

Model Analysis

Firstly, to increase the accuracy, we use a standard scaler to scale our data, since they vary a lot in size. Still, our accuracies do not hold water, as demonstrated in figure 7.3, which is mainly because we have employed data with low resolution. We intend to use a cluster of Dr. Hassani's computer in Geneva to run simulations with better resolutions in the following month.

There is also the question whether the Press-Schechter approximation was valid in this domain because the logistic regression method's accuracy seemed to be improved by increase in the degree of polynomial.

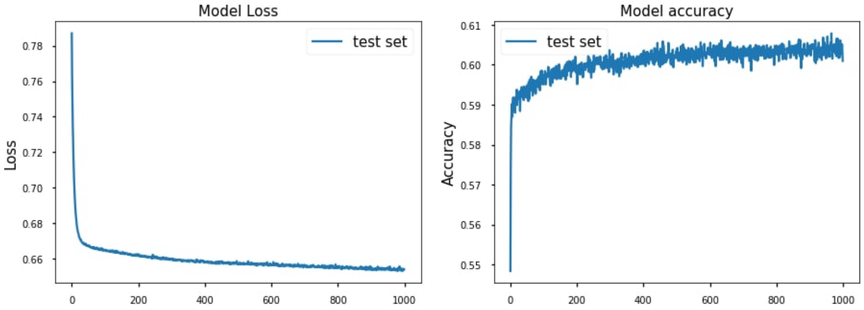

Figure 7.1: Accuracy And Loss Of The Neural Network For The Binary Problem using Keras library

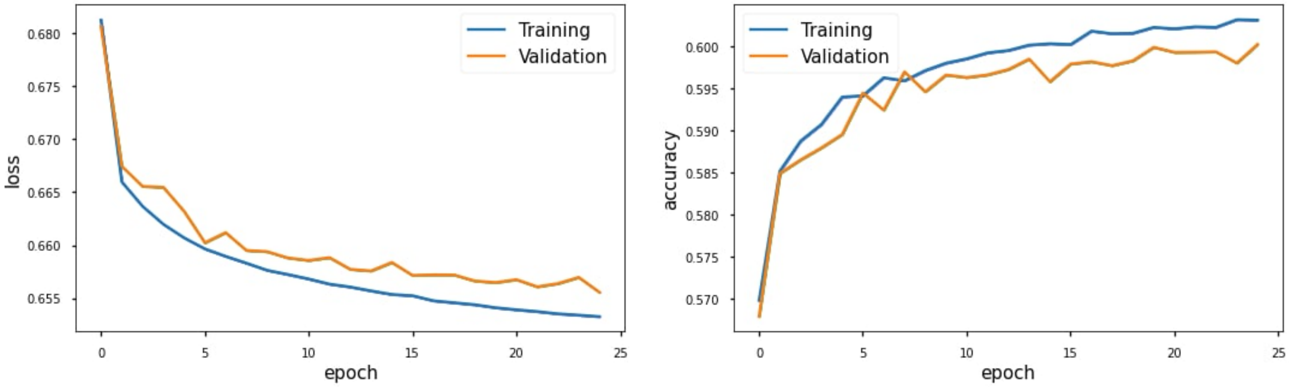

Figure 7.2: Cross-Validation Diagrams Of The Neural Network For The Binary Problem using Keras library

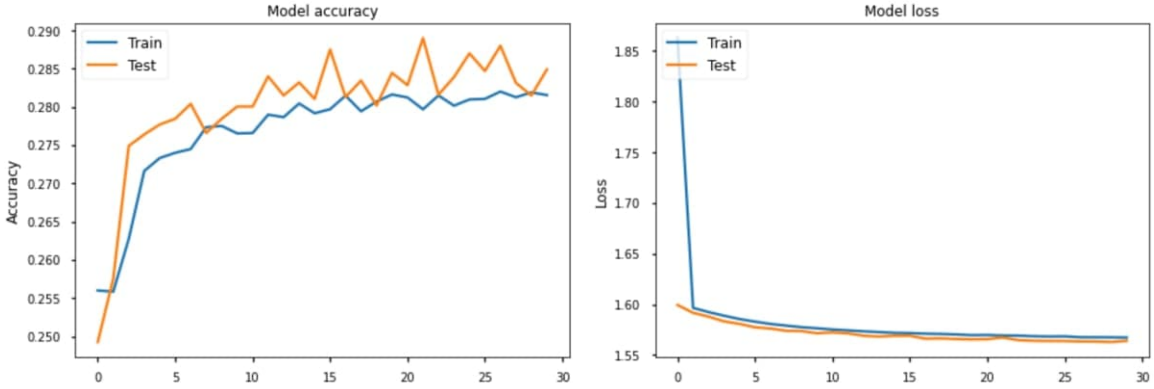

Figure 7.3: NN For Multi-Class Classification (Keras)

The neural network methods were generally less time-consuming but did not result in the best accuracies, comparing to the some of the classic methods. We also did not design NNs with more than six hidden layers, which is two more than what is generally used in these situations. As already mentioned, we have employed the Keras and PyTorch libraries. Surprisingly, their results differed considerably. Keras library resulted in a fairly more composed model with better accuracy – even though its predictions were also sub-optimal at best, but, it was the better of the two.

Conclusions

In this paper, we have outlined the machine learning methods that we employed to eventually predict the halo-mass function of the dark matter halos at $z = 0$. At this point, we are not able to do this with great accuracy, because our data is low in resolution. However, as we have seen, mostly by running the classic algorithms, not only is it possible to use machine learning methods to predict halo-mass distribution, but also it is much faster than running simulations each time we want to experiment with the structure evolution assuming different components in the universe.

Our results are by no means complete, and need further investigation. However, up to now, our results were consistent with the assumption that classical methods work better with our data, however low resolution they might be at this point. But, getting to the hasty conclusion that our classical methods are definitely better than neural networks might lead us to conclude a fallacy. Hence, we reserve those important conclusions for when we have rerun the algorithms with the better and more accurate data that we are going to extract from the simulations. Also, it seems at this point that neural networks are not the final solution to our problems. Better methods include using Convolutional Neural Networks, in which we preserve the shape of our matrix and do not touch the pristine data form the resulted snapshots of the simulations. We highly advise the use of CNNs, and we are going to do exactly that for further studies of the effect of neutrinos on the today's dark matter halo structure and mass function.

Bibliography

[1] Scott Dodelson, Fabian Schmidt. *Introduction to Modern Cosmology, Second edition*. 2020.

[2] Press W. H. Schechter. *Formation of Galaxies and Clusters of Galaxies by Self-Similar Gravitational Condensation*. 1974.

[3] Andrew R. Zentner. *The Excursion Set Theory of Halo Mass Functions,Halo Clustering, and Halo Growth.*. 2006.

[4] M. Bardeen. *THE STATISTICS OF PEAKS OF GAUSSIAN RANDOM FIELDS*. 1986.

[5] ndrew Pontzen Luisa Lucie-Smith Hiranya V. Peiris. *An interpretable machine learning framework for dark matter halo formation*. 2019.

[6] Ali Saraer - Amirhossein Samandar - Fatemeh Farhangian - Armita Kazemi. *Cosmology/MLP github page*. 2021.

[7] Agarwal S., Dave R., Bassett B. A. *2018, MNRAS, 478, 3410*

[8] Ball N. M., Brunner R. J. *2010, Int. J. Modern Phys. D, 19, 1049*

[9] Bardeen J. M., Bond J. R., Kaiser N., Szalay A. S. *1986, ApJ, 304, 15*

[10] Bond J. R., Myers S. T. *1996, ApJS, 103, 1*