Work in Classification

Simone Albanesi,[1,*] Marina Berbel,[2,†] Marco Cavaglià,[3,‡] Lorena Magaña Zertuche,[4,§] Miquel Miravet-Tenés,[5,¶] Dimitra Tseneklidou,[6,**] and Yanyan Zheng[3,††]

[1] *Dipartimento di Fisica, Università di Torino & INFN Sezione di Torino, via P. Giuria 1, 10125 Torino, Italy*
[2] *Departament de Matemàtiques, Universitat Autònoma de Barcelona, 08193 Bellaterra, Spain*
[3] *Institute of Multi-messenger Astrophysics and Cosmology & Physics Department, Missouri University of Science and Technology, Rolla, MO 65409, USA*
[4] *Department of Physics and Astronomy, University of Mississippi, University, Mississippi 38677, USA*
[5] *Departament d'Astronomia i Astrofísica, Universitat de València, Dr. Moliner 50, 46100, Burjassot (València), Spain*
[6] *Theoretical Astrophysics Department, Eberhard-Karls University of Tübingen, Tübingen 72076, Germany*

(Dated: January 26, 2023)

Abstract goes here.

I. INTRODUCTION

In our introduction we want to start from big picture (LIGO, ML, pipelines) and narrow down to what work we are presenting here and why it is important. We also describe in which ways it is novel and how it compares to previous works like in [**?**]. We may also want to cite [**?**].

The breakthrough observation of the binary neutron star (BNS) merger simultaneously in gravitational and electromagnetic (E/M) waves [**?**] marked a new era of multi-messenger astronomy including gravitational waves (GWs). The event GW170817 gave answers to open questions, such as the origin and production of Gamma-ray bursts (*cite), the production of heavy elements during a merger of BNS (*cite), rule out Equations of States (*cite) and alternative theories of gravity (*cite), +?. Among the challenges that it brings is the early alert of the E/M telescopes. The sources that can produce an E/M counterpart are the coalescences of a neutron star with another neutron star (NSNS) or with a black hole (NSBH). In the case of NSNS: GRBs, neutron star. For NSBH: accretion disk –> short GRBs.

- Mention empirical fits, Foucart?
- Real time inference: What was done in O2, O3 (Deep)
- Probabilities HasNS, HasRemnant
- What we suggest, Random Forest classification (what is it, what are the advantages)

II. DATASET AND LABELING

In this section, we discuss the data conditioning process our study. The data we use for training and testing are the same injections during O2 as in [REF Deep's paper], also aiming to predictict the probability of a detected event of having a neutron star (HasNS) and/or a remnant object (HasREM) that could emit an electromagnetic counterpart. As we utilize supervised ML techniques, we first label the data accordingly.

HasNS is considered true if $m2_inj < 3M\odot$ as in [REF]. For HasREM and following [REF] we use the Focault formula for the remnant mass and if the result is greater than 0 we label the event as true . However, this formula depends on the Equation of State (EoS) as it utilizes the compactness of the NS. The injections used in our study utilize the 2H EoS, but we relabel the data with a different one as explained later, as the EoS used for labelling the training data will affect the ability of predicting correctly a real event.

Predicting the presence of a NS and the creation of a remnant object like this requires a binary classifier for each of the two tasks. We employ a new approach that utilizes the relationship between HasNS and HasREM to train a single multilabel classifier. We relabel the data into 3 mutually exclusive categories: label 0 if there is no NS and no remnant, label 1 if there is a NS but no remnant, and label 2 if there are both. Our labelling is summarized in table I. This labeling eliminates the possibility of an unphysical classification of the event, where $p(HasREM) > p(HasNS)$, as there is no category for HasREM but no NS.

As the categories are mutually exclusive, $p(0) + p(1) + p(2) = 1$. Therefore, p(HasREM)=p(2) and for the NS, p(hasNS)=p(hasNS and hasREM ∪ hasNS and No hasREM)=p(2∪1)=p(2)+p(1)=1-p(0). This approach allows us to use a single classifier while avoiding any unphysical classifications without further treatment of the data or the output.

HasNS	HasRem	Our label
0	0	0
1	0	1
1	1	2

TABLE I. Labelling adopted for classification of having a NS and having a remnant with the same classifier

* simone.albanesi@edu.unito.it
† mberbel@mat.uab.cat
‡ cavagliam@mst.edu
§ lmaganaz@go.olemiss.edu
¶ miquel.miravet@uv.es
** dimitra.tseneklidou@uni-tuebingen.de
†† zytfc@umsystem.edu

In order to avoid conditioning the results of HasREM too heavily to the equation of state (EOS) selected for training, we labeled the dataset with the 23 EOS selected by the LIGO-Virgo collaboration in [ref] for studies of neutron stars. These EOS cover a wide parameter space, making together a robust estimation for the neutron star composition.

An algorithm was trained for each of the 23 EOS and predictions were obtained. The final prediction provided is a weighted average between the results of all of them, with the weights determined by the Bayes factor of each EoS, as explained in [ref2]. This approach allows for the consideration of multiple EOS and their relative likelihoods, resulting in a more robust and accurate prediction.

III. CLASSIFICATION ALGORITHMS

A. K-Nearest Neighbors (KNN)

One of the non-parametric algorithms we have used for classification is the `KNeighborsClassifier` (KNN). This algorithm assumes that similar things are near to each other. It captures the idea of similarity by computing the distance between points in a graph. It is also a lazy learning algorithm, because it does not have a training phase, it uses all the data for training while classification -the dataset is stored and at the time of classification, it acts on the dataset.

The workflow of the KNN algorithm is the following: first, after splitting the data into training and testing sets, we need to select the number of neighbors, K, which can be any integer. Then, for each point of the testing dataset we compute the distance between the point and each row of the training data with a chosen metric (`euclidean, manhattan, chebysev, mahalanobis`), and we sort the points in ascending order based on the distance value. By choosing the top K neighbors from the sorted array, we assign a class to the test point, which is the most frequent class of the chosen neighbors.

B. Random Forest (RF)

A RF is an ensemble of decision trees. One of its major strengths is that every train trains and classifies independently from the rest, while the RF classification joints all the results and assign as category the mode from the trees. The probability of belonging to a category is therefore straightforward, being the number of trees that chose it divided by the total number of trees. Notice that the training and evaluation of a RF can be accelerated by parallelization, as computations inside each tree are independent from the rest.

The training of a RF is usually done with bootstrap, a technique that assigns a random subset of the training dataset to each tree. This prevents overfitting as every individual classifier is not exposed to the same data, and encourages pattern recognition by studying the same data from different subsets. Every decision tree is composed by nodes, where data its splitted until the different categories are separated. At each node, a subset of the features of the data is selected along threshold values that maximize the information gain at the separation. The binary splitting at each node gives the tree its name, as it can be visualized as roots going deeper at separations.

We use the RK implementation in scikitlearn [**?**]. The main hyperparameters to tune in this module are the number of trees, the maximum depth allowed and the information gain criteria used at splitting (two are offered). We have observed that the maximum number of features to be considered in a node can be kept fix as the square root of the total number of features. Given that the aiming of this work is to improve the current low latency classification, the model once trained can occupy a restricted amount of memory. Therefore before searching the optimum hyperparameters for our dataset, we restrict those which make heavier models: the number of trees and their depth. We set to 300 the maximum number of trees the forest may have, and 45 their possible maximum depth.

For the RF we use events with 5 features: the two masses, their corresponding spins and the SNR of the detection. In the tuning of the hyperparameters we measure the performance by its score: the number of events correctly classified against the total number of events in the testing dataset, if threshold is taken as 0.5. As all categories are balanced, this approach is enough to roughly compare models. The best model found achieves a score of *whatever* using *number* of trees, *number* maximum depth and *name* criteria for the information gain.

IV. RESULTS

In order to decide which method gives a better performance in classifying this kind of events, we can apply them over testing data and finally do a comparison between both. A way to see how data is classified we can construct histograms where the number of events that are classified with a label (`HasNS/HasRemnant`) `True` or `False` will change with a given threshold of the probability. For an algorithm with perfect performance, all the events with label `True (False)` should be at $p($`label`$) = 1$ ($p($`label`$)$ $= 0$).

Another way to check the algorithm's performance is by building the so-called *Receiver Operating Characteristic (ROC) Curve*. They show the variation of the true-positive rate (or efficiency) with the false-positive rate given a certain threshold for the probability. An algorithm with a proper performance will give a steeper ROC curve, or in other words, will have a higher eficiency with a lower false-positive rate.

In the ROC curves that we will present in the following

subsections, we highlight three reference EoS in color, from which we show results in more detail. We select BHF_BBB2 because is the model that give the lowest maximum mass, MS1_PP as the model with the bigger maximum mass, and we also include SLy because is the most accepted EoS for NS modeling [ref], and is the one that was used in the injections that are our dataset.

Here are the results for both methods:

A. KNN Results

We apply cross validation in order to fix the different hyperparameters of the algorithm. To do so, we compute the score over a range of different parameters. We consider a number of neighbors between 1 and 20; the different metrics we test are the `euclidean`, `manhattan` and `cityblock`; the algorithms to compute the nearest neighbors can be `BallTree`, `KDTree` and the brute-force search, and the possible weight functions are the uniform weights (all points are weighted equally) and the *distance* weights (points are weighted by the inverse of their distance).

Considering all these possibilities, we apply the `cross val score` function from `scikit-learn` using a 10-fold cross validation to all the 23 datasets with different EOS. We find that the optimal metric, algorithm and weights are the same for all the EOS, being the `manhattan` metric, the `BallTree` algorithm (with a leaf size of 30) and the *distance* weights. The only parameter that differs is the number of neighbors, that goes from 8 to 12. One can find the optimal hyperparameters and the corresponding score for each EOS in Table II. Doing an average weighting by the Bayes factor of each EOS, we finally get an optimal number of neighbors of 10. The mean score from the cross validation goes from 0.941 (for H4 EOS) to 0.953 (for APR4 EPP), as one can see in Table II.

Considering now the SLy EOS, the model with these parameters gives a confusion matrix that is shown in Fig. 1. The probability of having a NS as a function of m_1 and m_2 is shown in Fig. 2. There are no big differences with different values of the spins, but the most remarkable one is that the model classifies better for 0-spin values, especially when m_1 is large. There is also a dependence of the probability of having a remnant on the value of the spin that can be seen in Fig. 3.This dependence is correct, since the probability of having a remnant increases for large values of m_1 at larger spin, but this dependence is given by the EOS.

In Figs. 4, 5 and 6 we depict the histograms of the probabilities p(*HasNS*) (p(*HasRemnant*)) for injections of binaries that had a NS (EM counterpart) and for those that not, for the EOS BHF BB2, MS1 PP and SLy, respectively.

We show in Fig. 7 the ROC curves of the classifier for the two different probabilities we consider. All the EOS are depicted in this figure, but we highlight three cases: BHF BBB2, MS1 PP and SLy.

EOS	Score	$K_{\text{neighbors}}$	Bayes Factor
APR4 EPP	0.953	10	1.526
BHF BBB2	0.951	8	1.555
H4	0.941	10	0.056
HQC18	0.950	10	1.422
KDE0V	0.951	8	1.177
KDE0V1	0.949	10	1.283
MPA1	0.951	14	0.276
MS1 PP	0.948	12	0.001
MS1B PP	0.947	11	0.009
RS	0.945	10	0.176
SK255	0.946	10	0.179
SK272	0.945	10	0.156
SKI2	0.943	12	0.108
SKI3	0.943	12	0.107
SKI4	0.949	10	0.330
SKI5	0.945	9	0.025
SKI6	0.949	10	0.288
SKMP	0.948	10	0.290
SKOP	0.948	10	0.618
SLy	0.950	10	1.000
SLY2	0.950	10	1.028
SLY9	0.949	10	0.370
SLY230A	0.950	10	0.932

TABLE II.

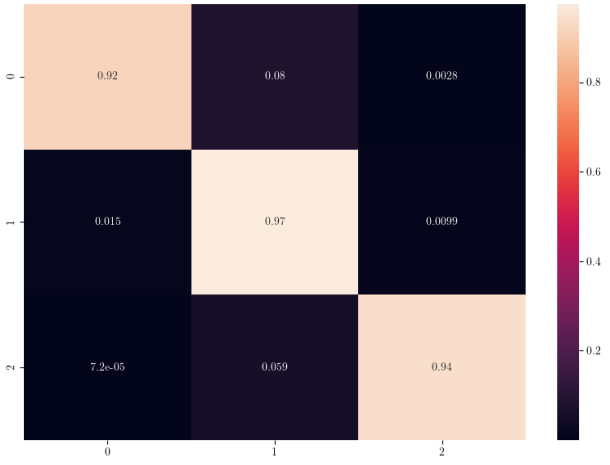

FIG. 1. Confusion matrix SLy KNN

B. RF Results

We apply crossvalidation in the number of trees and depth of the forests for the 23 EoS, fixing the information gain criteria to `entropy`. We also save the second best option for comparison, and save both forests for each EoS in order to compare the file size. As the goal is to provide a model that can run in a low latency pipeline the amount of memory it can take is limited, even more when there will be 23 different model for the EoS generalization.

In table III we present a summary of best and second best hyperparameters found in the crossvalidation for each EoS, along the memory the model occupies and the difference in the score. As we can see, usually a forest

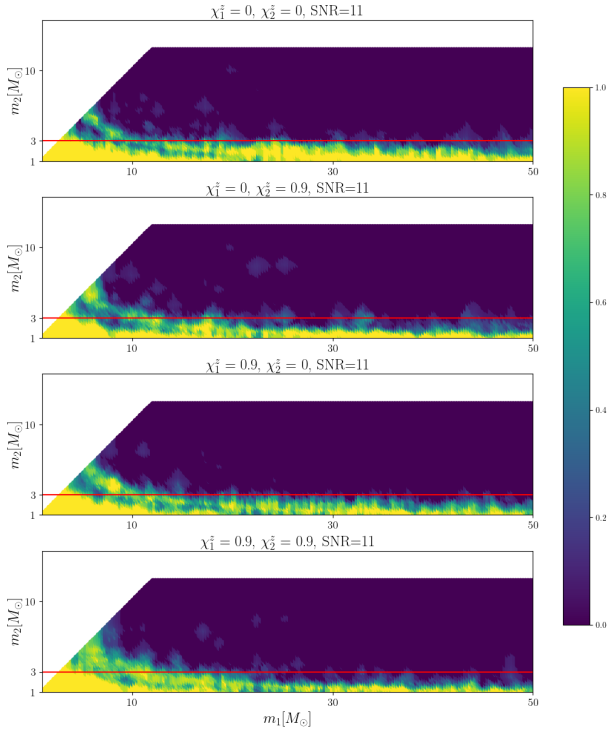

FIG. 2. Parameter sweep NS SLy KNN

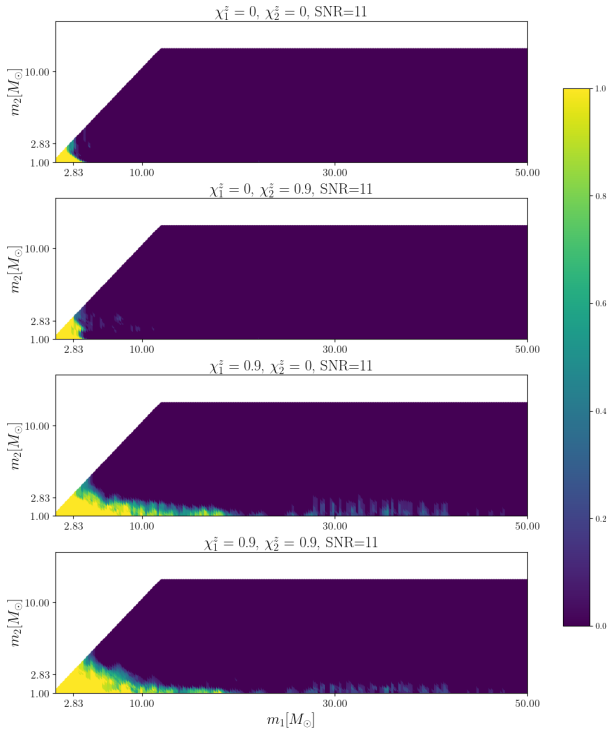

FIG. 3. Parameter sweep REM SLy KNN

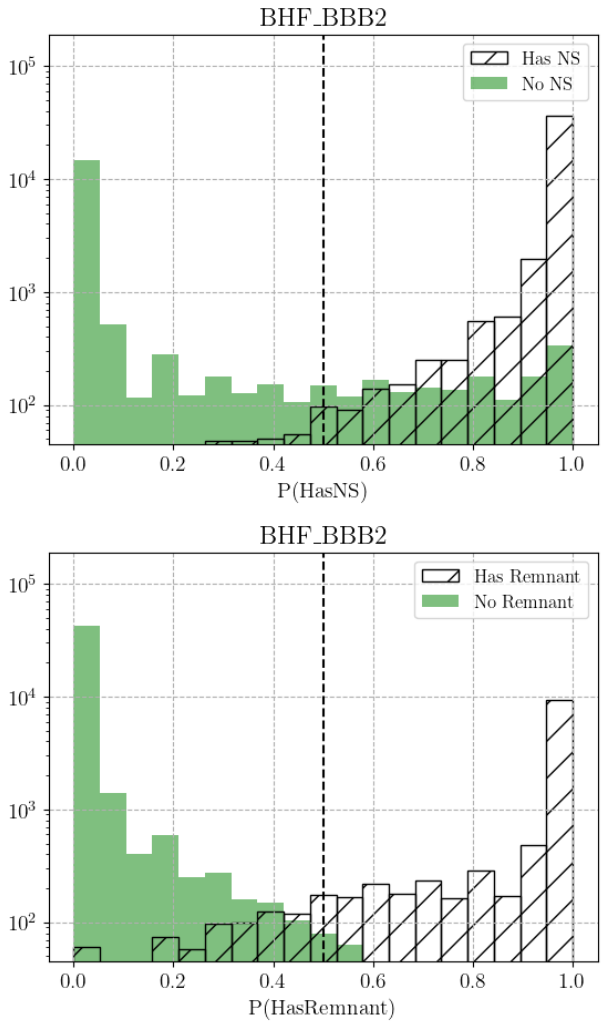

FIG. 4. Histograms BHF BBB2 KNN

with many trees has a second best option with far less that is lighter in memory and achieves a similar performance. The optimum maximum depth is always 15. Also the score achieved for every EoS is similar, and so we check that our accuracy is not NS model dependent.

To simplify the model and according to the results of crossvalidation, we train the final forests for all EoS with 50 trees and 15 maximum depth. In figure 8 we show the ROC curves for all models to give an idea of the performance. Notice that HasREM performs better than HasNS. The ourperformance of HasREM against HasNS in RF is even more noticeable in the histograms in figures 9, 10 and 11 for the highlighted EoS, where the bars of asigned probabilities do not intersect each other and therefore there exists a threshold value for perfect classification in the testing dataset.

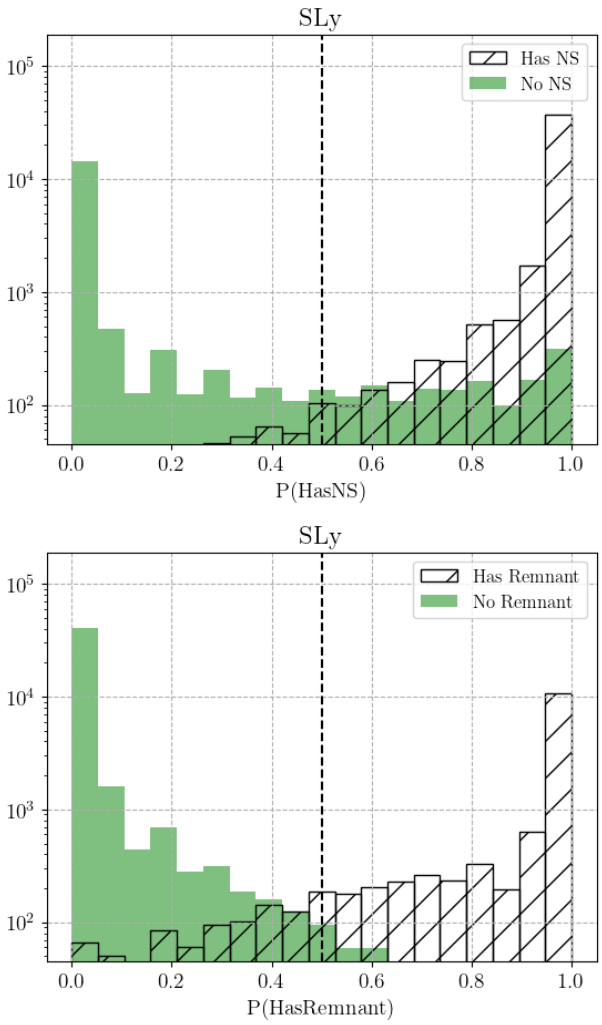

FIG. 5. Histograms SLy KNN

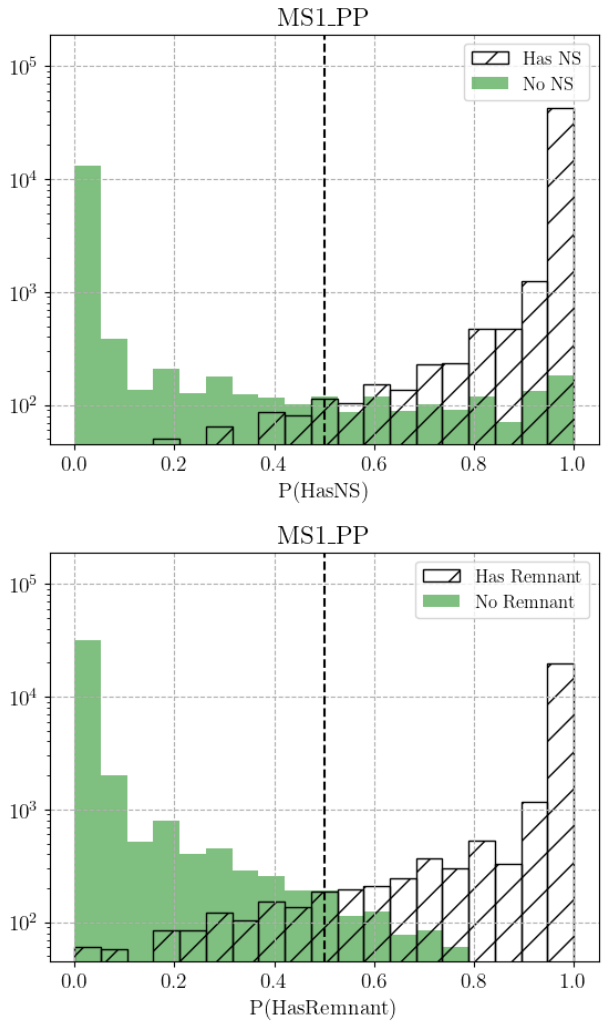

FIG. 6. Histograms MS1 PP KNN

C. Algorithm comparison

To compare quantitatively the results from RF and KNN we compute the true positive and false positive rate for several threshold values, for both HasNS and HasREM, for the three selected EoS. These are tables IV, V and VI. For HasNS the two algorithms perform similarly, with almost the same TP for all threshold values and accross EoSs, although the false positive is smaller always in the RF. For HasREM we obtain that RF performs better than KNN in every case, with not only a smaller false positive rate, but a greater true positive rate.

V. CONCLUSIONS

Reiterate why what we did is important and how it improves current knowledge.

ACKNOWLEDGMENTS

We thank Deep Chatterjee for useful discussions and for sharing his work, which helped us compare our results to those of [**?**]. We also thank Shaon Ghosh for useful discussions and X, Y, and Z for reviewing an earlier version of this manuscript. Part of this research was performed while the authors were visiting the Institute of Pure and Applied Mathematics (IPAM), University of California Los-Angeles (UCLA). The authors would like to thank IPAM, UCLA and the National Science Foundation through grant DMS-1925919 for their warm hospitality during the fall of 2021.

The authors are grateful for computational resources provided by the LIGO Laboratory and supported by the U.S. National Science Foundation Grants PHY-0757058 and PHY-0823459, as well as resources from the Gravitational Wave Open Science Center, a service of the LIGO Laboratory, the LIGO Scientific Collaboration and the Virgo Collaboration.

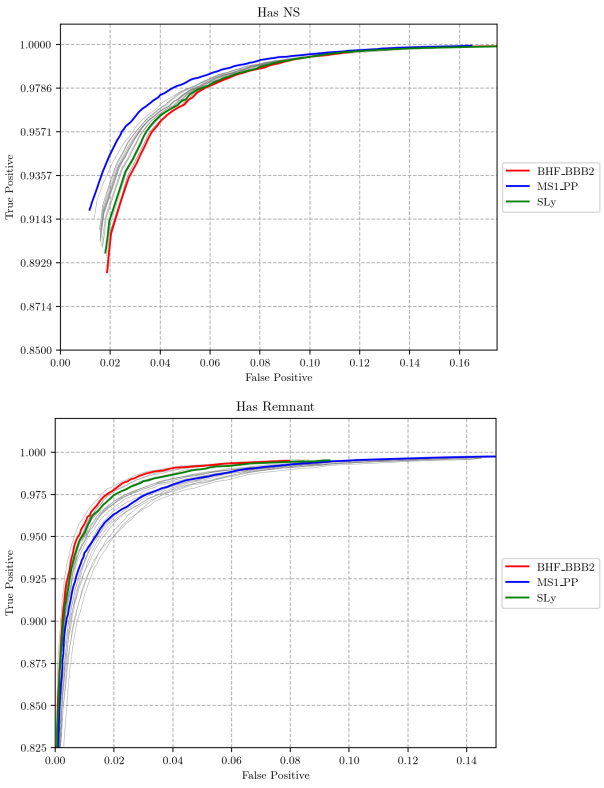

FIG. 7. ROC curves KNN

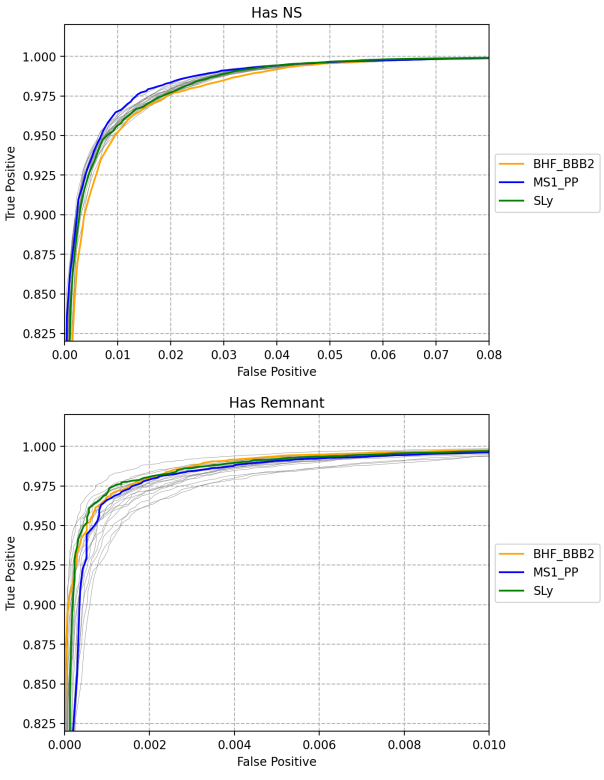

FIG. 8. ROC curves

	Best				Second best		
EOS	Trees	Depth	Size(MB)	Score	Trees	Depth	Size(MB)
APR4_BB	300	15	94.7	0.9683018	50	15	15.7
BHF_BBB2	80	15	24.4	0.9685127	300	15	91.6
H4	80	15	29.6	0.9618587	300	15	111.4
HQC18	300	15	93.7	0.9673755	100	15	31.3
KDE0V	300	15	92.0	0.9673295	80	15	24.5
KDE0V1	100	15	30.9	0.96704954	80	15	24.5
MPA1	80	15	27.2	0.96601225	300	15	102.1
MS1_PP	300	15	113.5	0.96563534	80	15	30.2
MS1B_PP	300	15	114.2	0.96555340	100	15	38.0
RS	300	15	103.8	0.96447350	80	15	27.6
SK255	300	15	105.8	0.96472405	100	15	35.5
SK272	300	15	109.0	0.96401816	100	15	36.4
SKI2	50	15	18.8	0.96242338	300	15	112.8
SKI3	50	15	19.0	0.96174537	100	15	38.1
SKI4	300	15	100.6	0.96598969	30	15	9.8
SKI5	100	15	38.2	0.96343381	80	15	30.4
SKI6	300	15	101.7	0.96586928	30	15	10.0
SKMP	300	15	100.2	0.96544567	80	15	26.9
SKOP	100	15	32.3	0.96610459	300	15	96.2
SLy	80	15	25.3	0.96728884	300	15	95.2
SLY2	100	15	31.8	0.96745868	80	15	25.4
SLY9	300	15	101.6	0.96605993	100	15	34.1
SLY230A	300	15	95.5	0.96714915	100	15	31.9

TABLE III. Comparison of the best and second best RF models obtained during crossvalidation for all EoS. We show the file size in MB of the forest, and the difference in score between the two options.

	Has NS				Has REM			
	RF		KNN		RF		KNN	
Threshold	TP	FP	TP	FP	TP	FP	TP	FP
0.1	0.999	0.107	0.999	0.156	0.998	0.011	0.992	0.051
0.3	0.998	0.068	0.996	0.117	0.993	0.005	0.974	0.017
0.5	0.994	0.042	0.991	0.088	0.985	0.003	0.937	0.006
0.8	0.967	0.014	0.966	0.043	0.957	0.001	0.845	0.001

TABLE IV. BHF_BB2

The work of L.M.Z. was partially supported by the MSSGC Graduate Research Fellowship, awarded through the NASA Cooperative Agreement 80NSSC20M0101. The work of X.Y. was partially supported by NSF Grant No. PHY-20XXXXX. The work of M.M.T. was supported by the Spanish Ministry of Universities through the Ph.D. grant No. FPU19/01750, by the Spanish Agencia Estatal

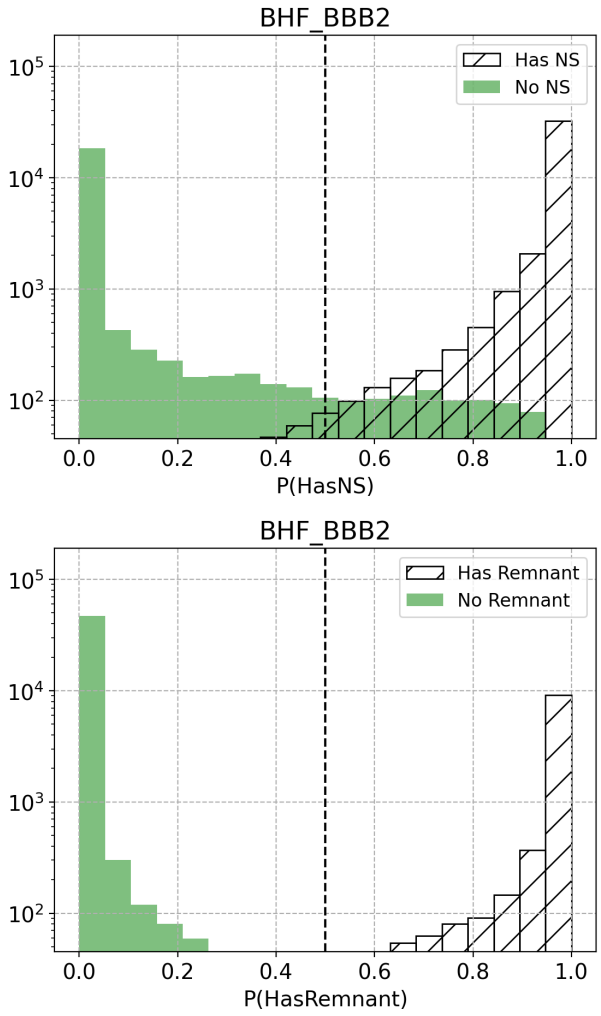

FIG. 9. Histograms BHF BBB2

	Has NS				Has REM			
	RF		KNN		RF		KNN	
Threshold	TP	FP	TP	FP	TP	FP	TP	FP
0.1	1.000	0.107	0.999	0.155	0.998	0.013	0.992	0.059
0.3	0.999	0.064	0.996	0.112	0.993	0.005	0.974	0.020
0.5	0.994	0.038	0.990	0.084	0.986	0.003	0.940	0.007
0.8	0.965	0.012	0.965	0.040	0.958	0.001	0.848	0.001

TABLE VI. SLy

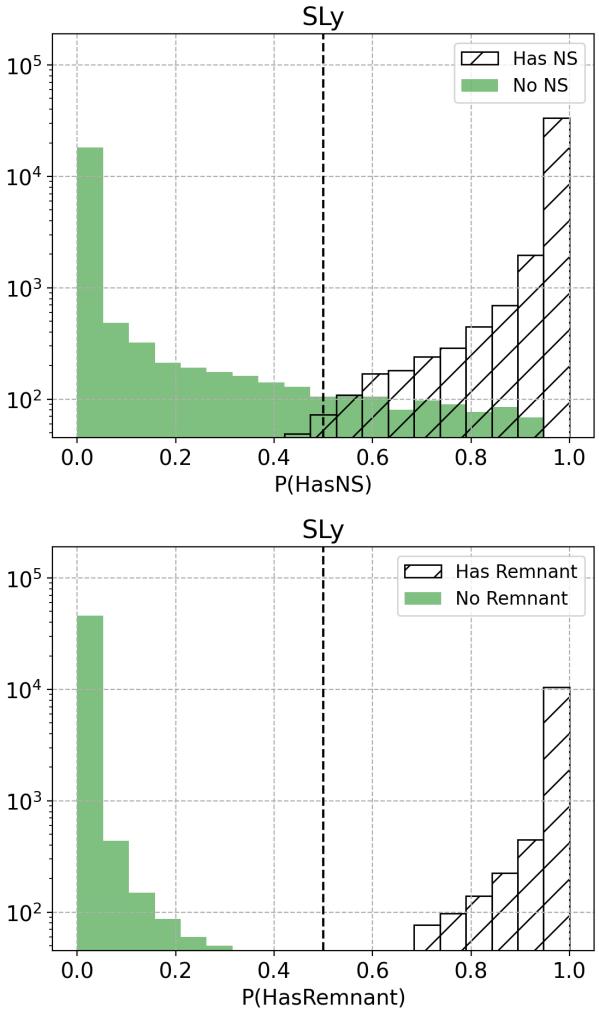

FIG. 10. Histograms SLy

	Has NS				Has REM			
	RF		KNN		RF		KNN	
Threshold	TP	FP	TP	FP	TP	FP	TP	FP
0.1	1.000	0.114	0.999	0.138	0.999	0.023	0.995	0.103
0.3	0.998	0.065	0.995	0.097	0.996	0.010	0.983	0.044
0.5	0.994	0.036	0.989	0.068	0.990	0.004	0.961	0.019
0.8	0.968	0.011	0.967	0.031	0.967	0.001	0.899	0.004

TABLE V. MS1_PP

de Investigación (Grants No. PGC2018-095984-B-I00 and PID2021-125485NB-C21) and by the Generalitat Valenciana (Grant No. PROMETEO/2019/071)

The work of M.B. was supported by the Spanish Agencia Estatal de Investigación (Grants No. PID2020-118236GB-I00).

All plots were made using the python package `matplotlib` [**?**].

This manuscript has been assigned LIGO Document Control Center number LIGO-P22XXXXX.

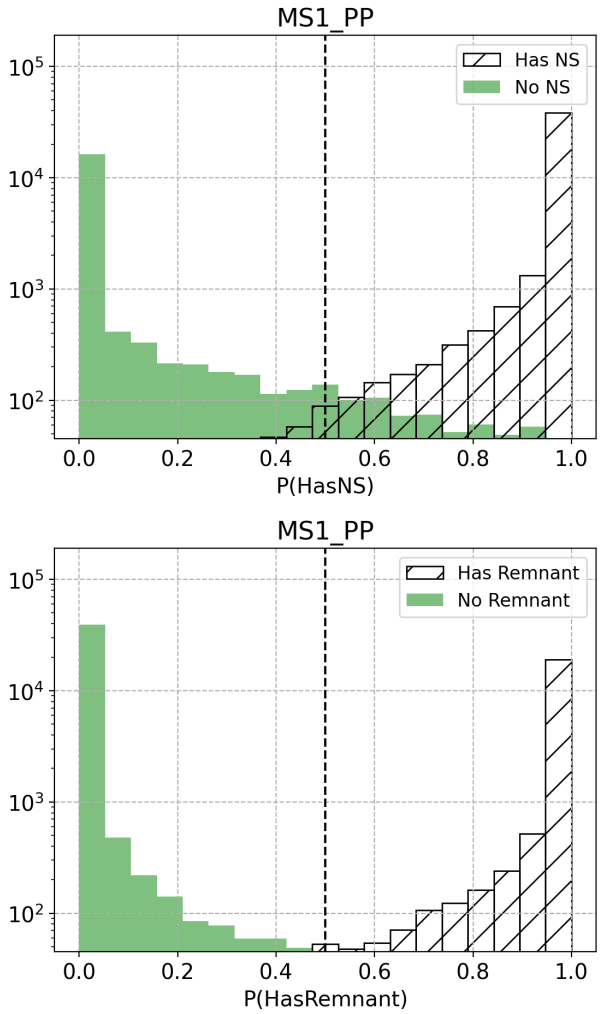

FIG. 11. Histograms MS1 PP