Real-time classification of multi-messenger electromagnetic and gravitational-wave observations

Simone Albanesi,[1] Marina Berbel,[2] Marco Cavaglià,[3] Lorena Magaña Zertuche,[4] Miquel Miravet-Tenés,[5] Dimitra Tseneklidou,[6] Yanyan Zheng,[3] and Sushant Sharma Chaudhary[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: February 21, 2023)

Abstract goes here.

I. INTRODUCTION

The first detection of a gravitational-wave (GW) signal from a pair of coalescing black holes in 2015 and the first observation of a coalescing binary neutron system two years later have established multi-messenger astronomy (MMA) as a powerful tool for the exploration of the cosmos TODO: add refs. The third LIGO, Virgo, and KAGRA (LVK) catalog of transient GW signals (GWTC-3) TODO: add ref has shown that GW astronomy has entered its mature phase, becoming a true observational branch of astronomy. In the next few years, MMA will allow scientists to explore in depth the origin and structure of black holes, neutron stars, and gamma-ray bursts; test general relativity; probe the fundamental nature of gravity; and measure the evolution of the universe TODO: add references here?.

Improved GW detector sensitivities will bring a wealth of new detections to achieve these science goals TODO: ref about observing scenario and detectors. The rate of detections in LVK's fourth observation run (O4) is expected to be more than one per day, further increasing in the fifth observation run (O5) TODO: ref about observing scenario. Over the course of these runs, the LVK will analyze hundreds of binary black hole (BBH) and dozens of binary neutron star (BNS) and neutron star and black hole (NSBH) detections, several of which could be MMA sources. Among the challenges that this new phase of GW astronomy brings is the necessity to coordinate the activities of electromagnetic (EM) and GW observatories on a very short time scale.

One of the most interesting areas of study in GW astronomy is the physics of gravity-matter interaction in MMA sources. Tidally disrupted matter in an NSBH system may form a high-temperature accretion disk around the black hole (BH) and trigger the creation of a prompt EM emission in the form of a short gamma-ray burst (GRB). If the system ejecta are unbound, r-process nucleosynthesis may lead to a kilonova (KN) [8–13]. These phenomena could also arise in BNS post-mergers through the expulsion of neutron-rich material even when tidal forces are weak [1, 2, 14–18]. The presence of a post-merger matter remnant, which results in an EM signature or a prompt collapse, is a common factor in all of these scenarios. Determining the potential of a GW source to become an EM emitter and enabling coincident observations of these systems by EM and GW observatories in real time are crucial for the success of MMA.

The LVK employs different matched-filtering pipelines for low-latency GW searches [23–25] TODO: references for pipelines and add spiir. These searches are based on discrete template banks of compact binary coalescence (CBC) waveforms that provide, among other parameters, the component masses and the dimensionless aligned/anti-aligned spins of the objects along the orbital angular momentum. These parameters can be used to determine the *EM properties* of GW candidates in low latency through empirical fits of Numerical relativity (NR) simulations [20, 21]. Low-latency preliminary alerts of candidate GW events in the third observation run (O3) included two EM-property metrics identifying whether the CBC system contains a neutron star (NS), `HasNS`, and a post-merger matter remnant, `HasRemnant`. Alerts with similar content and will continue to be issued in O4 with a latency of the order of one second after the detection of candidate merger events. Additionally, LVK's O4 alerts will include a measure of the `HasMassGap` property, i.e., the likelihood that one of the source compact objects has mass in the lower-mass gap region between NSs and BHs

Classification of GW candidate events in real time poses several challenges as the desire for accuracy contrasts with the desire to issue the information as quickly as possible. The approach taken in O3 was to use a supervised KNeighborClassifier (KNN) [26] machine learning (ML) algorithm with input from the detection pipelines and equation of state (EOS) models to generate independent `HasNS` and `HasRemnant` binary classification scores [22]. The KNN model was trained on a broad set of synthetic CBC signals injected in real detector noise from the second observation run (O2). The advantage of this scheme relies in the capability of the method to handle the statistical uncertainties of the parameters from the search pipelines and the non-stationarity of the detectors' power spectral density. This allowed for a marked improvement in per-

formance compared to the semi-analytic effective Fisher formalism method which was deployed in O2.

In this work, we revisit the problem of real-time production of the `HasNS` and `HasRemnant` metrics with the aim of further improving the latency and performance of the ML-based scheme. To this purpose, we explore a set of new algorithms and classification schemes. In our approach, we first design a scheme that allows for conditional `HasNS` and `HasRemnant` metrics to include the physical requirement that the probability that a system with a post-merger matter remnant must necessarily contain a NS. Then we consider an extended set of EOS and marginalize the results over this set to minimize possible systematics arising from the EOS. Finally, we perform a thorough study of alternative ML algorithms and compare their performance on the synthetic signals in O2 data and confident O3 GW detections from the GWTC-3 catalog.

The structure of the paper is as follows. In Sect. II we describe the data set and the classification schema. In Sect. III, we introduce the classification algorithms. Section IV is dedicated to the results of each method and their comparison. Conclusions and future developments are presented in Sect. V.

II. DATASET AND LABELING

In this section, we discuss the data conditioning process in 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 $\cup$ hasNS and No hasREM)=p(2$\cup$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.In this work, RF and KNN were trained on data labeled as in I, however, genetic programming algorithm used in this work was trained on binary labels for hasNS and hasRemnant separately because of its limitations in performing multi-label classification.

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

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.

Each algorithm presented here 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 [27]. 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.

C. Genetic Programming

Genetic Programming(GP) is a supervised ML algorithm [**?**].It is an evolutionary computation that employs naturally occurring genetic operations, a fitness function, and multiple generations of Darwinian evolution to resolve a user-defined task [**?**]. GP can be used to discover a mathematical relationship between the input variables, also called features, in data (regression) or group data into categories (classification). Similar to biological natural selection, GP learns how well the program functions by comparing the output of its multivariate expression to a *fitness score*. Programs with high fitness score are most likely, but not certain, to propagate to next generation. With subsequent generations, programs are more likely to get better at solving the task at hand [**?**].

GP multivariate expressions are classically represented as a syntax tree, where the trees have a root (top center), nodes (mathematical operators), and leaves (operands). Operators can be arithmetic, trigonometric, boolean, etc. and the operands are place-holder variables related to the problem. The tree depth can be varied aiding in evolution of more complex multivariate expression. GP intitially generates a stochastic poplulation of trees, computes fitness scores and randomly selects trees for comparison. The lead scoring trees are selected and one of the genetic operators (reproduction, mutation, crossover) is randomly applied to carry them forward to next generation. The process repeats until user-defined conditions are met. The run-time parameters like intial population size, tree depth, tournament size for competing trees, number of generations, and termination criterion can be tuned for better algorithmic performance.

The unique aspect of GP algorithm used here is the transparency of its internal workings. Unlike many black box ML algorithms, the evolutionary process can be reviewed at each step which might be quite important to many researchers.[**?**]

For our analysis, we used an open source python code, Karoo GP [**?**]. Karoo GP package is scalable, with multicore and GPU support enabled by the TensorFlow library and has capacity to work with very large datasets. The latest version of Karoo GP can be found in PyPI [**?**].

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 $p(\texttt{label}) = 1$ ($p(\texttt{label}) = 0$).

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 (reference), and is the one that was used in the injections that are our dataset.

Here are the results for the methods:

D. 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.

We show in Fig. 1 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.

E. 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.

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 2 we show the ROC curves for all models to give an idea of the

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

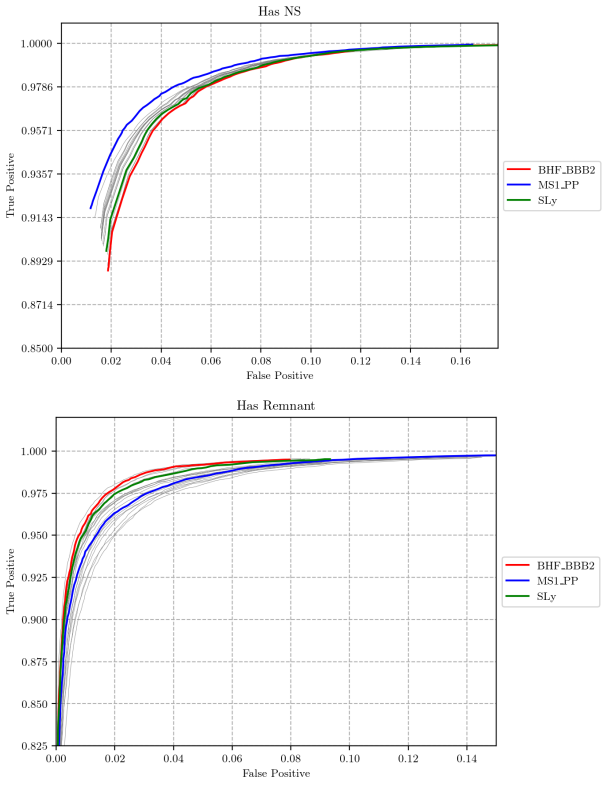


FIG. 1: ROC curves KNN

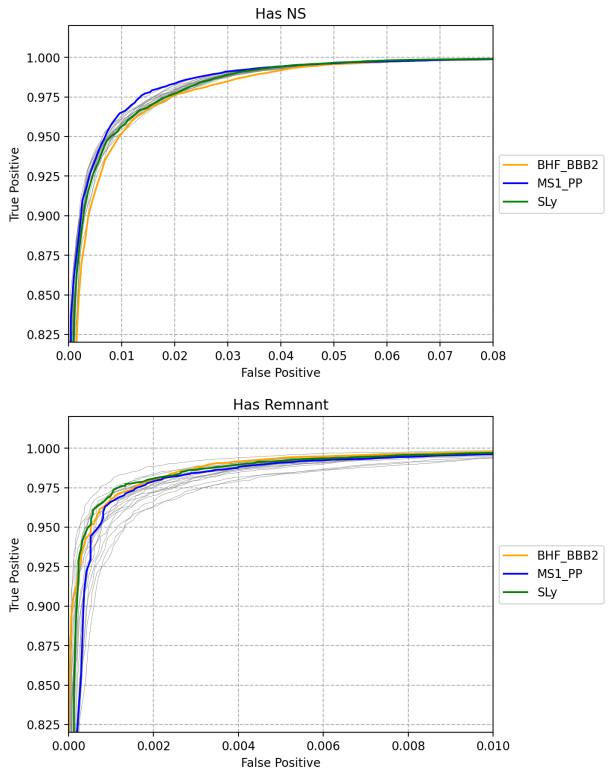


FIG. 2: ROC curves RF

performance. Notice that HasREM performs better than HasNS.

F. GP Results

The end of each Karoo GP training produces a multivariate exression with heighest fitness score. We repeat the training process 200 times to get 200 expressions for each label (hasNS and hasRemnant). Due to the stochastic nature of the GP algorithm, the expressions mostly tend to be unique depending upon the complexity of the classification problem. Each expression can then be used to classify the reconstructed source parameters to determine the labels hasNS and hasRemnant by substituting the values in the expression. If the result is greater or equal to zero, we classify the reconsturcted set of prameter as true and false otherwise. The performace of each of the expressions against the testing set for one EoS is shown in Fig 3. As seen in the plots, the average performace of hasRemnant expressions is better than hasNS expressions. Also the winning expressions are more dispersed in case of hasRemnant. Shall we explain the reasons: hasNS being simpler classification but contaminated by poor reconstructions by the pipeline. HasRemnant is more complex leading to more variable winning trees.
Calculation of Probability: Using testing set, we compute probabilities using bayesian method. HasNS and hasRemnant are not inherintly independent quantities since there can be no hasRemnant without hasNS. Hence, we utilize both hasNS winning expressions and hasRemnant winning expressions for quantifying probabilities $p(hasNS)$ and $p(hasRemnant)$. If XREM denotes the number of hasRemnant winning trees which classifies given set of parameter as true and XNS denotes the number of hasNS winning trees which classifies as true, we compute the probability on the testing set as:

$$p(HasNS|XREM \cap XNS) = \frac{p(XREM \cap XNS|HasNS).p(HasNS)}{p(XREM \cap XNS)} \quad (1)$$

$$p(HasRemnant|XREM \cap XNS) = \frac{p(XREM \cap XNS|HasRemnant).p(HasRemnant)}{p(XREM \cap XNS)} \quad (2)$$

There as some caveats to this calculation. Given a finite testing set, we cannot fully compute the entire parameter space (201 x 201 cases) of the probability since there are some cases which are highly unlikely to occur, like $p(hasNS|XNS = 0 \cap XREM = 200)$. Also, our probability computations might also suffer due to the finite testing set size. The resulting probability distribution is highly rugged and discontinuous or incomplete. Hence, in order to span the entire parameter space of the probability, we use gasussian process regression (GPR).

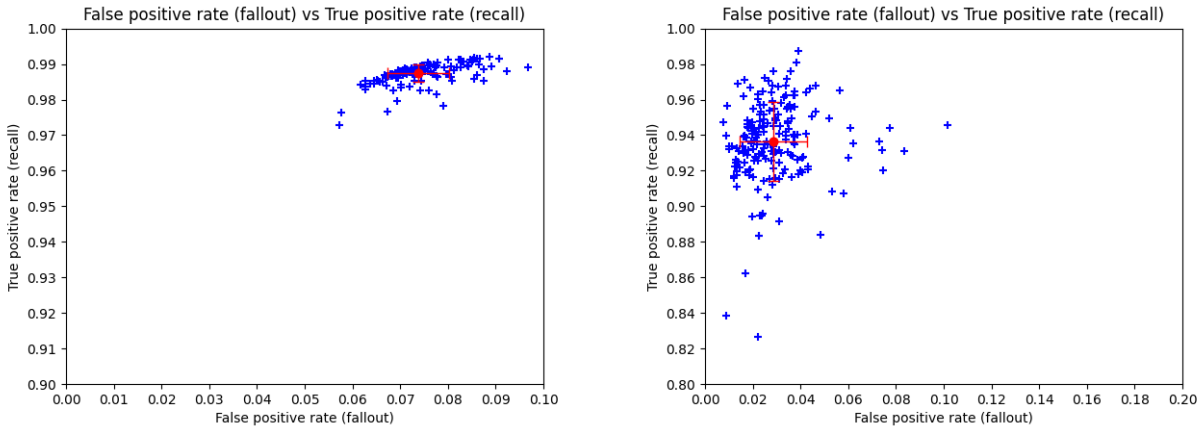
Gaussian Process Regression: We implement GPR using a python package gpytorch [**?**]. Using the Rational Quadratic(RQ) kernel of the gpytorch , we compute a smooth probability distribution over all possible combination of XREM and XNS as shown in Fig 4. The selection of the parameters for GPR was motivated by two major factors. The first being that probability had to be confined between interval [0,1] and the case $P(hasNS)$ or $P(hasRemnant)$ 1 when XNS and $XREM$ approaches maximum number of trees.The second being the condition $p(hasNS|XREM \cap XNS) \geq p(hasRemnant|XREM \cap XNS)$. After multiple trials, we found that using 200 winning expression and using length scale interval $[1X10^{-5}, 1X10^{5}]$ and alpha parameter interval $[1X10^{-5}, 10]$ for GPR, we can obtain desired probability distribution.

Using this probablity, we make the ROC curve using the testing set for all the EoS as shown in 5.

G. Algorithm comparison

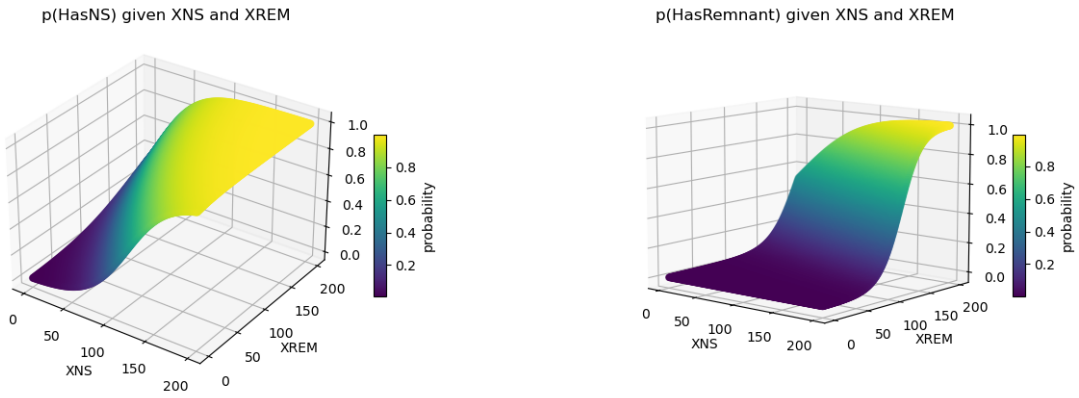
IV. RESULTS

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 III, **??** and



(a) (b)

FIG. 3: The plot on the left is a FPR (False Positive Rate) vs (True Positive Rate) plot for all 200 winning expressions for hasNS classification on MS1-PP EoS. The red dot is the average of TPR and FPR of the expressions and the vetricle and horizontal spread are the one sigma values for TPR and FPR respectively. The plot on the right is similar but for hasRemnant classification.



(a) (b)

FIG. 4: The plot on the left is the probability distrbution of equation 1 after GPR implementation for MS1-PP EoS. The plot on the right is a similar plot of equation 2

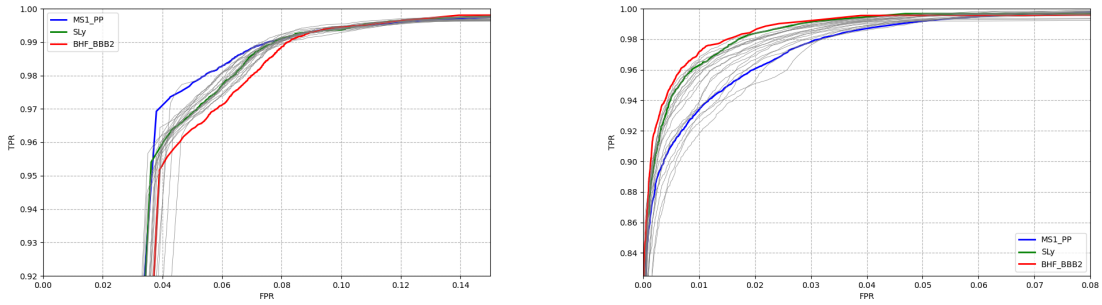
??. For HasNS the two algorithms perform similarly, with almost the same TP for all threshold values and accross EoSs, although the false positive is smaller always in the RF. For HasREM we obtain that RF performs better than KNN in every case, with not only a smaller false positive rate, but a greater true positive rate.

	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 III: BHF_BB2

V. CONCLUSIONS

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



(a) (b)

FIG. 5: The plot on the left is the ROC curve for hasHS label evaluated on the testing set whereas the plot in the right is for hasRemnant for all 23 EoS.

ACKNOWLEDGMENTS

We are very grateful to the authors of Ref. [22] for fruitful discussions and for sharing their work that helped us to directly compare our methods to those used in LVK's O3 observing run. We would also like to thank the many other colleagues of the LIGO Scientific Collaboration and the Virgo Collaboration who have provided invaluable help over the years.

This work is based upon work supported by the LIGO Laboratory which is a major facility fully funded by the National Science Foundation. We 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. L.M.Z. is partially supported by the MSSGC Graduate Research Fellowship, awarded through the NASA Cooperative Agreement 80NSSC20M0101. M.C. and Y.Z. are partially supported by the U.S. National Science Foundation under award PHY-2011334. M.M.T. is supported by the Spanish Ministry of Universities through the Ph.D. grant No. FPU19/01750, by the Spanish Agencia Estatal de Investigación (Grants No. PGC2018-095984-B-I00 and PID2021-125485NB-C21) and by the Generalitat Valenciana (Grant No. PROMETEO/2019/071). M.B. is supported by the Spanish Agencia Estatal de Investigación (Grants No. PID2020-118236GB-I00). Part of this research was performed at the Institute of Pure and Applied Mathematics (IPAM), University of California Los-Angeles (UCLA). IPAM is partially supported by the National Science Foundation through award DMS-1925919. We would like to thank IPAM and UCLA for their warm hospitality.

Software for this analysis is written in `Python` 3.x TODO: add ref and uses standard Open Source libraries and community-contributed modules from the Python Package Index (PyPI) repository TODO: add ref including `numpy`, `scipy`, `pandas`, `matplotlib` [28], and `sklearn`.

This manuscript has been assigned LIGO Document Control Center number LIGO-P23XXXXX.

[1] B. P. Abbott *et al.* (LIGO Scientific, Virgo, Fermi GBM, INTEGRAL, IceCube, AstroSat Cadmium Zinc Telluride Imager Team, IPN, Insight-Hxmt, ANTARES, Swift, AGILE Team, 1M2H Team, Dark Energy Camera GW-EM, DES, DLT40, GRAWITA, Fermi-LAT, ATCA, ASKAP, Las Cumbres Observatory Group, OzGrav, DWF (Deeper Wider Faster Program), AST3, CAASTRO, VINROUGE, MASTER, J-GEM, GROWTH, JAGWAR, CaltechNRAO, TTU-NRAO, NuSTAR, Pan-STARRS, MAXI Team, TZAC Consortium, KU, Nordic Optical Telescope, ePESSTO, GROND, Texas Tech University, SALT Group, TOROS, BOOTES, MWA, CALET, IKI-GW Follow-up, H.E.S.S., LOFAR, LWA, HAWC, Pierre Auger, ALMA, Euro VLBI Team, Pi of Sky, Chandra Team at McGill University, DFN, ATLAS Telescopes, High Time Resolution Universe Survey, RIMAS, RATIR, SKA South Africa/MeerKAT), Multi-messenger Observations of a Binary Neutron Star Merger, Astrophys. J. Lett. **848**, L12 (2017), arXiv:1710.05833 [astro-ph.HE].

[2] I. Arcavi *et al.*, Optical emission from a kilonova following a gravitational-wave-detected neutron-star merger, Nature **551**, 64 (2017), arXiv:1710.05843 [astro-ph.HE].

[3] R. Chornock *et al.*, The Electromagnetic Counterpart of the Binary Neutron Star Merger LIGO/VIRGO GW170817. IV. Detection of Near-infrared Signatures of r-process Nucleosynthesis with Gemini-South, Astrophys. J. Lett. **848**, L19 (2017), arXiv:1710.05454 [astro-ph.HE].

[4] M. R. Drout, A. L. Piro, B. J. Shappee, C. D. Kilpatrick, J. D. Simon, C. Contreras, D. A. Coulter, R. J. Foley, M. R. Siebert, N. Morrell, K. Boutsia, F. Di Mille, T. W. S. Holoien, D. Kasen, J. A. Kollmeier, B. F. Madore, A. J. Monson, A. Murguia-Berthier, Y. C. Pan, J. X. Prochaska, E. Ramirez-Ruiz, A. Rest, C. Adams, K. Alatalo, E. Bañados, J. Baughman, T. C. Beers, R. A. Bernstein, T. Bitsakis, A. Campillay, T. T. Hansen, C. R. Higgs, A. P. Ji, G. Maravelias, J. L. Marshall, C. Moni Bidin, J. L. Prieto,

		p(HasNS)			p(HasREM)		
event ID	grace_id	RF	KNN	GP	RF	KNN	GP
GW170823	G298936	0.000	0.000	0.014	0.000	0.000	0.001
GW170817	G298048	1.000	1.000	0.999	1.000	1.000	0.995
GW170814	G297595	0.000	0.000	0.013	0.000	0.000	0.001
GW170809	G296853	0.002	0.000	0.013	0.000	0.000	0.001
GW190408	G329243	0.000	0.000	0.013	0.000	0.000	0.001
GW190412	G329483	0.000	0.000	0.021	0.000	0.000	0.001
GW190413-052954	G329577	0.000	0.000	0.013	0.000	0.000	0.001
GW190413-134308	G329615	0.000	0.000	0.010	0.000	0.000	0.001
GW190421	G330300	0.000	0.000	0.017	0.000	0.000	0.001
GW190425	G330564	1.000	1.000	0.999	0.999	1.000	0.994
GW190426	G330687	0.996	1.000	0.980	0.009	0.000	0.008
GW190503	G331315	0.000	0.000	0.032	0.000	0.000	0.001
GW190512	G332169	0.000	0.000	0.014	0.000	0.000	0.001
GW190513	G332333	0.000	0.000	0.014	0.000	0.000	0.001
GW190517	G333132	0.000	0.000	0.010	0.000	0.000	0.001
GW190519	G333443	0.000	0.000	0.014	0.000	0.000	0.001
GW190521-074359	G333664	0.000	0.000	0.013	0.000	0.000	0.001
GW190602	G335015	0.000	0.000	0.061	0.000	0.000	0.001
GW190630	G337426	0.000	0.000	0.014	0.000	0.000	0.001
GW190706	G337919	0.015	0.000	0.019	0.000	0.000	0.001
GW190707	G337978	0.000	0.000	0.096	0.000	0.000	0.001
GW190708	G338125	0.000	0.000	0.070	0.000	0.000	0.001
GW190720	G344653	0.004	0.000	0.024	0.000	0.000	0.001
GW190727	G345173	0.011	0.000	0.013	0.000	0.000	0.001
GW190728	G345315	0.000	0.000	0.013	0.000	0.000	0.001
GW190814	G347305	0.098	0.700	0.878	0.000	0.000	0.002
GW190828-063405	G348500	0.002	0.000	0.014	0.000	0.000	0.001
GW190828-065509	G348519	0.001	0.000	0.019	0.000	0.000	0.001
GW190915	G350491	0.000	0.000	0.009	0.000	0.000	0.001
GW190924	G351423	0.037	0.070	0.100	0.000	0.000	0.001
GW190930	G351993	0.000	0.000	0.079	0.000	0.000	0.001
GW191109	G354231	0.000	0.000	0.036	0.000	0.000	0.001
GW191129	G355916	0.004	0.000	0.024	0.000	0.000	0.001
GW191204-171526	G356500	0.000	0.000	0.026	0.000	0.000	0.001
GW191215	G357403	0.000	0.000	0.017	0.000	0.000	0.001
GW191216	G357490	0.000	0.000	0.125	0.000	0.000	0.001
GW191222	G358088	0.000	0.000	0.028	0.000	0.000	0.001
GW200112	G359994	0.000	0.000	0.015	0.000	0.000	0.001
GW200115	G360364	0.997	0.000	0.987	1.000	0.000	0.011
GW200129	G361581	0.000	0.000	0.015	0.000	0.000	0.001
GW200219	G364596	0.000	0.000	0.027	0.000	0.000	0.001
GW200224	G365371	0.014	0.000	0.027	0.000	0.000	0.001
GW200225	G365427	0.001	0.000	0.013	0.000	0.000	0.001
GW200302	G366190	0.000	0.000	0.014	0.000	0.000	0.001
GW200311	G367788	0.000	0.000	0.01	0.000	0.000	0.001
GW200316	G368545	0.000	0.000	0.068	0.000	0.000	0.001
GW200322	G369200	0.012	0.000	0.031	0.000	0.000	0.001

TABLE IV: PROBAB TABLE REAL DATA

K. C. Rasmussen, C. Rojas-Bravo, A. L. Strom, N. Ulloa, J. Vargas-González, Z. Wan, and D. D. Whitten, Light curves of the neutron star merger GW170817/SSS17a: Implications for r-process nucleosynthesis, Science **358**, 1570 (2017), arXiv:1710.05443 [astro-ph.HE].

[5] D. Kasen, B. Metzger, J. Barnes, E. Quataert, and E. Ramirez-Ruiz, Origin of the heavy elements in binary neutron-star mergers from a gravitational-wave event, Nature (London) **551**, 80 (2017), arXiv:1710.05463 [astro-ph.HE].

[6] B. P. Abbott *et al.* (LIGO Scientific, Virgo), GW170817: Measurements of neutron star radii and equation of state, Phys. Rev. Lett. **121**, 161101 (2018), arXiv:1805.11581 [gr-qc].

[7] T. Baker, E. Bellini, P. G. Ferreira, M. Lagos, J. Noller, and I. Sawicki, Strong constraints on cosmological gravity from gw170817 and grb 170817a, Phys. Rev. Lett. **119**, 251301 (2017).

[8] J. M. Lattimer and D. N. Schramm, Black-hole-neutron-star collisions, Astrophys. J. Lett. **192**, L145 (1974).

[9] L.-X. Li and B. Paczynski, Transient events from neutron star mergers, Astrophys. J. Lett. **507**, L59 (1998), arXiv:astro-ph/9807272.

[10] O. Korobkin, S. Rosswog, A. Arcones, and C. Winteler, On the astrophysical robustness of neutron star merger r-process, Mon. Not. Roy. Astron. Soc. **426**, 1940 (2012), arXiv:1206.2379 [astro-ph.SR].

[11] J. Barnes and D. Kasen, Effect of a High Opacity on the Light Curves of Radioactively Powered Transients from Compact Object Mergers, Astrophys. J. **775**, 18 (2013), arXiv:1303.5787 [astro-ph.HE].

[12] M. Tanaka and K. Hotokezaka, Radiative Transfer Simulations of Neutron Star Merger Ejecta, Astrophys. J. **775**, 113 (2013), arXiv:1306.3742 [astro-ph.HE].

[13] D. Kasen, R. Fernandez, and B. Metzger, Kilonova light curves from the disc wind outflows of compact object mergers, Mon. Not. Roy. Astron. Soc. **450**, 1777 (2015), arXiv:1411.3726 [astro-ph.HE].

[14] D. A. Coulter *et al.*, Swope Supernova Survey 2017a (SSS17a), the Optical Counterpart to a Gravitational Wave Source, Science **358**, 1556 (2017), arXiv:1710.05452 [astro-ph.HE].

[15] M. M. Kasliwal *et al.*, Illuminating Gravitational Waves: A Concordant Picture of Photons from a Neutron Star Merger, Science **358**, 1559 (2017), arXiv:1710.05436 [astro-ph.HE].

[16] V. M. Lipunov *et al.*, MASTER Optical Detection of the First LIGO/Virgo Neutron Star Binary Merger GW170817, Astrophys. J. Lett. **850**, L1 (2017), arXiv:1710.05461 [astro-ph.HE].

[17] M. Soares-Santos *et al.* (DES, Dark Energy Camera GW-EM), The Electromagnetic Counterpart of the Binary Neutron Star Merger LIGO/Virgo GW170817. I. Discovery of the Optical Counterpart Using the Dark Energy Camera, Astrophys. J. Lett. **848**, L16 (2017), arXiv:1710.05459 [astro-ph.HE].

[18] N. R. Tanvir *et al.*, The Emergence of a Lanthanide-Rich Kilonova Following the Merger of Two Neutron Stars, Astrophys. J. Lett. **848**, L27 (2017), arXiv:1710.05455 [astro-ph.HE].

[19] M. Ruiz and S. L. Shapiro, General relativistic magnetohydrodynamics simulations of prompt-collapse neutron star mergers: The absence of jets, Phys. Rev. D **96**, 084063 (2017), arXiv:1709.00414 [astro-ph.HE].

[20] F. Foucart, Black Hole-Neutron Star Mergers: Disk Mass Predictions, Phys. Rev. D **86**, 124007 (2012), arXiv:1207.6304 [astro-ph.HE].

[21] F. Foucart, T. Hinderer, and S. Nissanke, Remnant baryon mass in neutron star-black hole mergers: Predictions for binary neutron star mimickers and rapidly spinning black holes, Phys. Rev. D **98**, 081501 (2018), arXiv:1807.00011 [astro-ph.HE].

[22] D. Chatterjee, S. Ghosh, P. R. Brady, S. J. Kapadia, A. L. Miller, S. Nissanke, and F. Pannarale, A Machine Learning

Based Source Property Inference for Compact Binary Mergers, Astrophys. J. **896**, 54 (2020), arXiv:1911.00116 [astro-ph.IM].
[23] S. Sachdev *et al.*, An Early-warning System for Electromagnetic Follow-up of Gravitational-wave Events, Astrophys. J. Lett. **905**, L25 (2020), arXiv:2008.04288 [astro-ph.HE].
[24] A. H. Nitz, T. Dal Canton, D. Davis, and S. Reyes, Rapid detection of gravitational waves from compact binary mergers with PyCBC Live, Phys. Rev. D **98**, 024050 (2018), arXiv:1805.11174 [gr-qc].
[25] T. Adams, D. Buskulic, V. Germain, G. M. Guidi, F. Marion, M. Montani, B. Mours, F. Piergiovanni, and G. Wang, Low-latency analysis pipeline for compact binary coalescences in the advanced gravitational wave detector era, Classical and Quantum Gravity **33**, 175012 (2016), arXiv:1512.02864 [gr-qc].
[26] F. Pedregosa *et al.*, Scikit-learn: Machine Learning in Python, J. Machine Learning Res. **12**, 2825 (2011), arXiv:1201.0490 [cs.LG].
[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, A. Müller, J. Nothman, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, Scikit-learn: Machine Learning in Python, Journal of Machine Learning Research **12**, 2825 (2011), arXiv:1201.0490 [cs.LG].
[28] J. D. Hunter, Matplotlib: A 2D Graphics Environment, Comput. Sci. Eng. **9**, 90 (2007).

In Figs. 6, 7 and 8 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.

Considering now the SLy EOS, the model with these parameters gives a confusion matrix that is shown in Fig. 9. The probability of having a NS as a function of m_1 and m_2 is shown in Fig. 10. 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. 11.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 table V 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 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.

The ourperformance of HasREM against HasNS in RF is even more noticeable in the histograms in figures 12, 13 and 14 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.

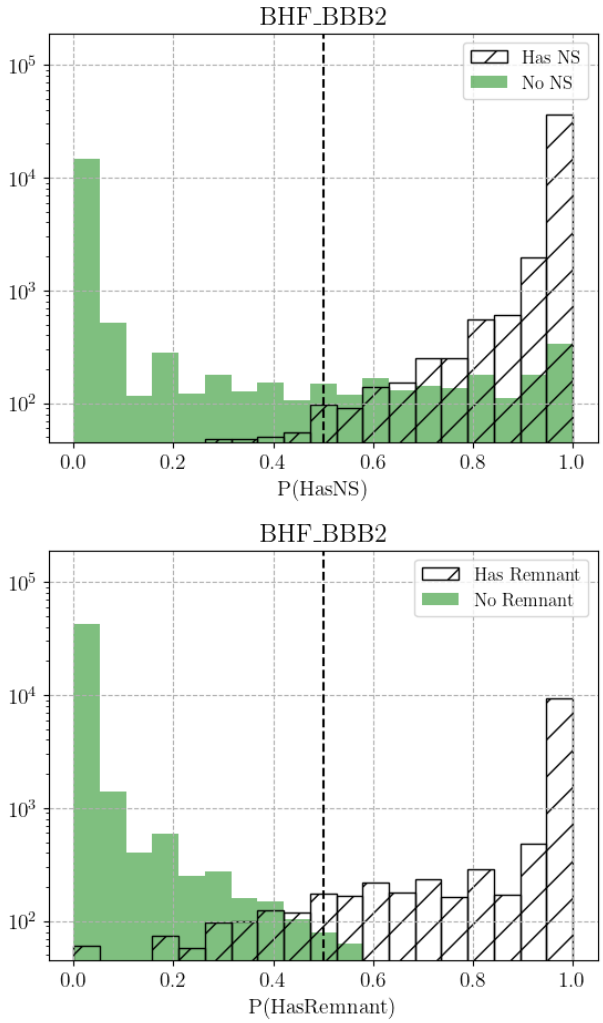


FIG. 6: Histograms BHF BBB2 KNN

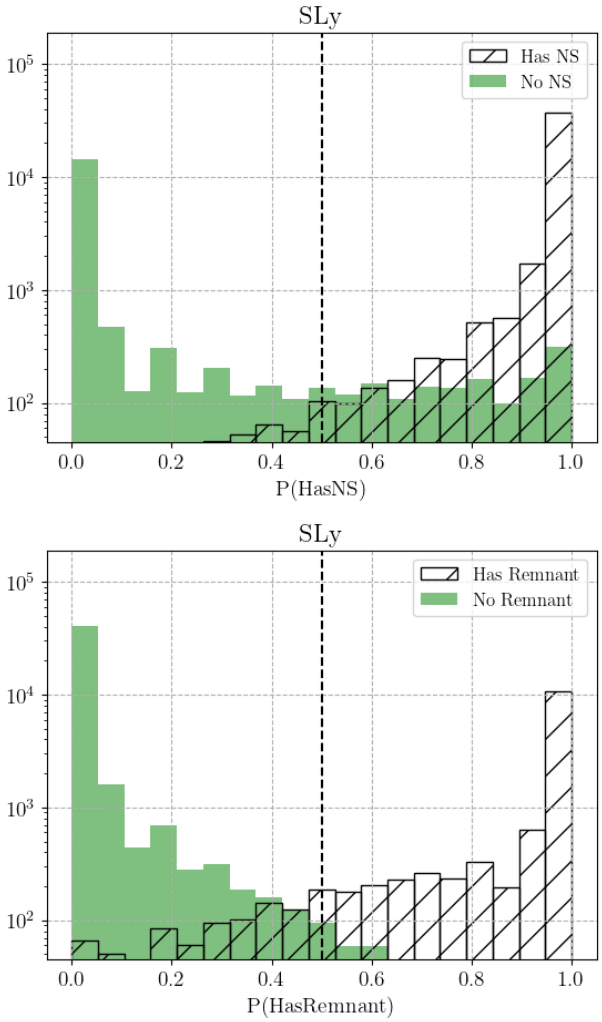


FIG. 7: Histograms SLy KNN

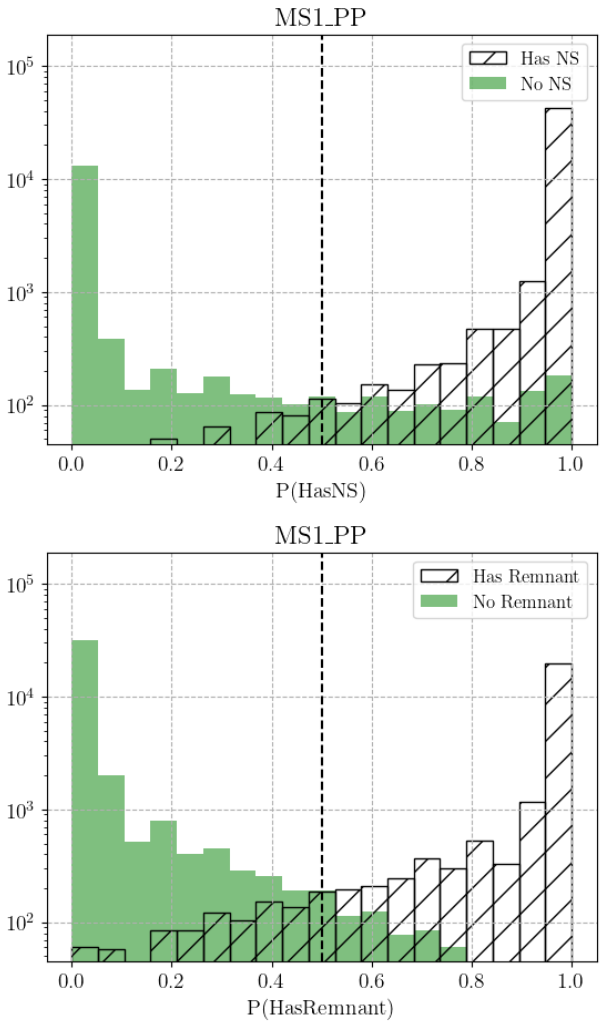


FIG. 8: Histograms MS1 PP KNN

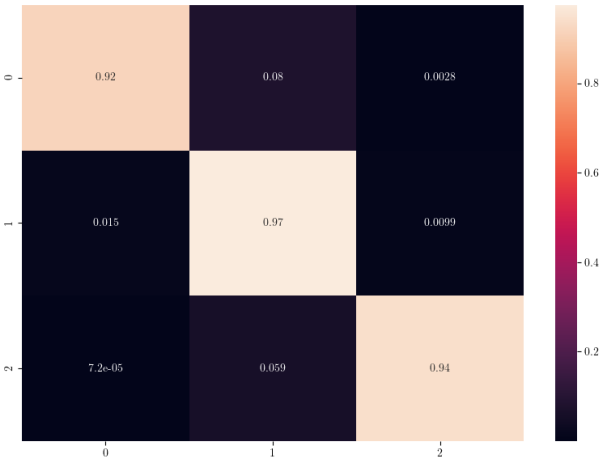


FIG. 9: Confusion matrix SLy KNN

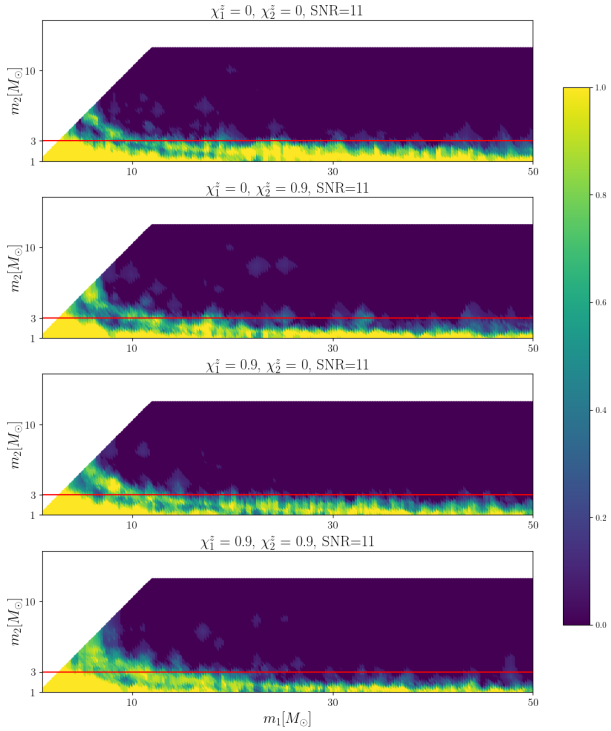


FIG. 10: Parameter sweep NS SLy KNN

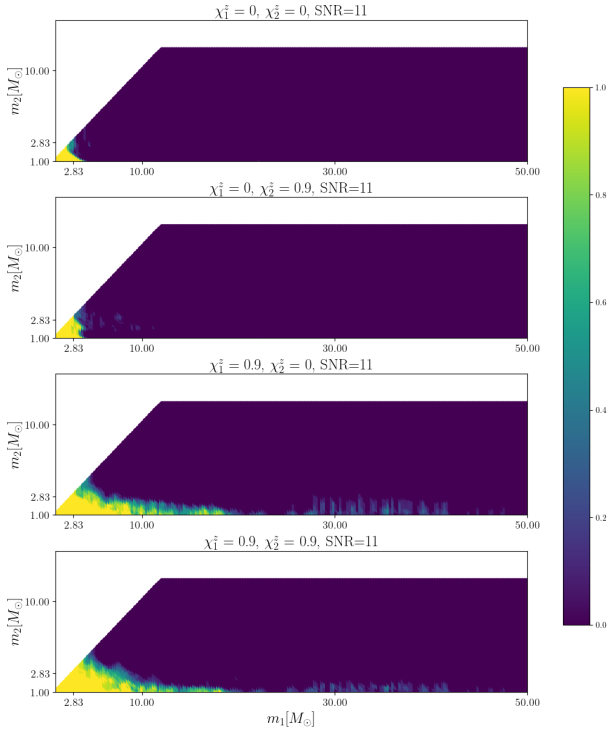


FIG. 11: Parameter sweep REM SLy KNN

	Best				Second best			
EOS	Trees	Depth	Size(MB)	Score	Trees	Depth	Size(MB)	Δscore
APR4_BB	300	15	94.7	0.9683018	50	15	15.7	3.35e-5
BHF_BBB2	80	15	24.4	0.9685127	300	15	91.6	5.16e-5
H4	80	15	29.6	0.9618587	300	15	111.4	1.19e-4
HQC18	300	15	93.7	0.9673755	100	15	31.3	3.06e-4
KDE0V	300	15	92.0	0.9673295	80	15	24.5	2.06e-4
KDE0V1	100	15	30.9	0.96704954	80	15	24.5	3.43e-5
MPA1	80	15	27.2	0.96601225	300	15	102.1	8.19e-5
MS1_PP	300	15	113.5	0.96563534	80	15	30.2	1.15e-4
MS1B_PP	300	15	114.2	0.96555340	100	15	38.0	1.97e-4
RS	300	15	103.8	0.96447350	80	15	27.6	2.36e-4
SK255	300	15	105.8	0.96472405	100	15	35.5	3.69e-4
SK272	300	15	109.0	0.96401816	100	15	36.4	1.99e-4
SKI2	50	15	18.8	0.96242338	300	15	112.8	8.37e-5
SKI3	50	15	19.0	0.96174537	100	15	38.1	6.62e-5
SKI4	300	15	100.6	0.96598969	30	15	9.8	8.37e-5
SKI5	100	15	38.2	0.96343381	80	15	30.4	1.16e-4
SKI6	300	15	101.7	0.96586928	30	15	10.0	2.17e-4
SKMP	300	15	100.2	0.96544567	80	15	26.9	1.69e-4
SKOP	100	15	32.3	0.96610459	300	15	96.2	6.85e-5
SLy	80	15	25.3	0.96728884	300	15	95.2	8.49e-5
SLY2	100	15	31.8	0.96745868	80	15	25.4	2.38e-4
SLY9	300	15	101.6	0.96605993	100	15	34.1	1.51e-4
SLY230A	300	15	95.5	0.96714915	100	15	31.9	2.53e-4

TABLE V: 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.

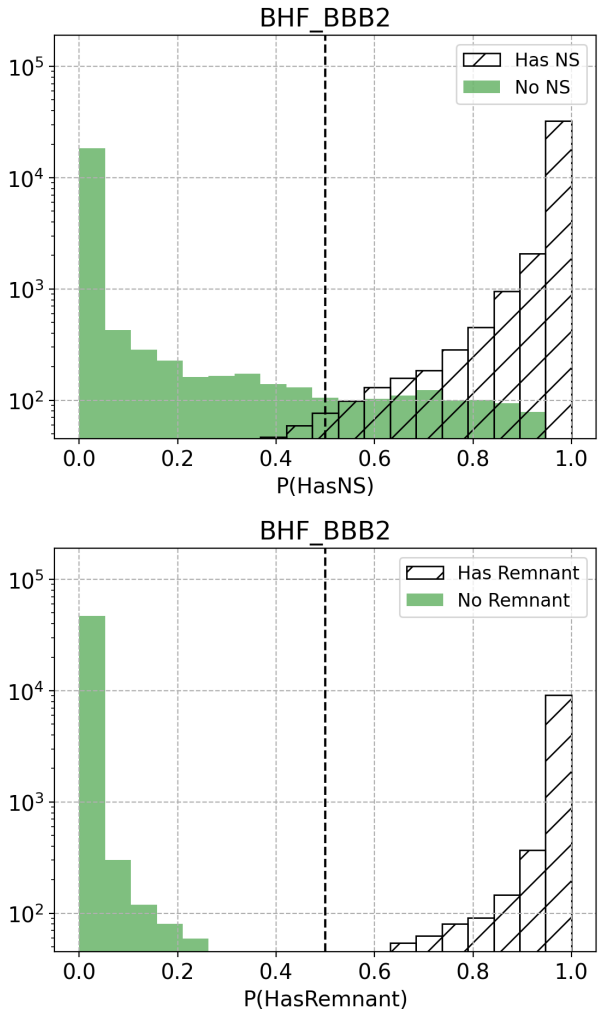


FIG. 12: Histograms BHF BBB2

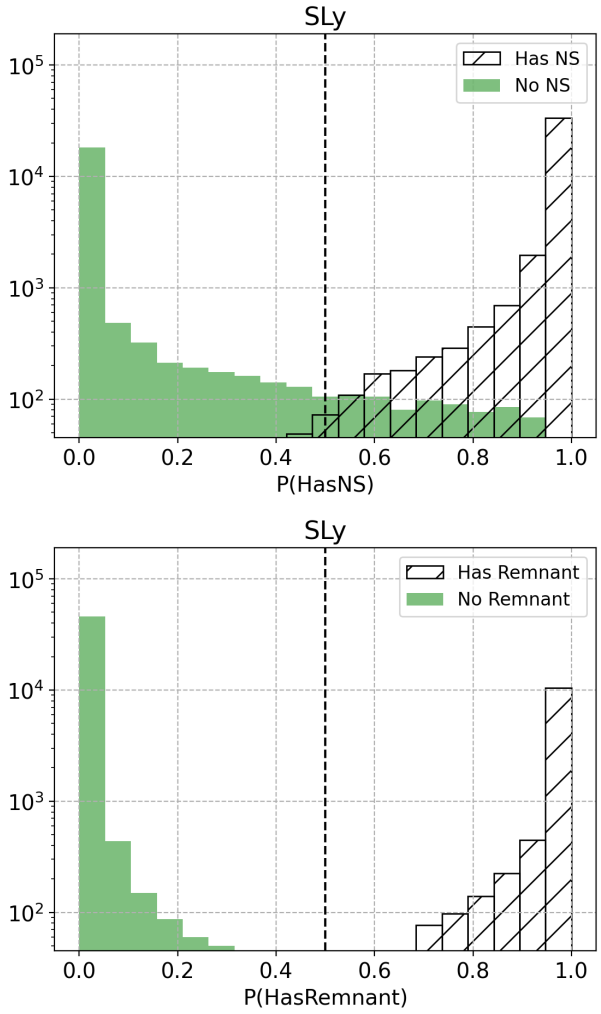


FIG. 13: Histograms SLy

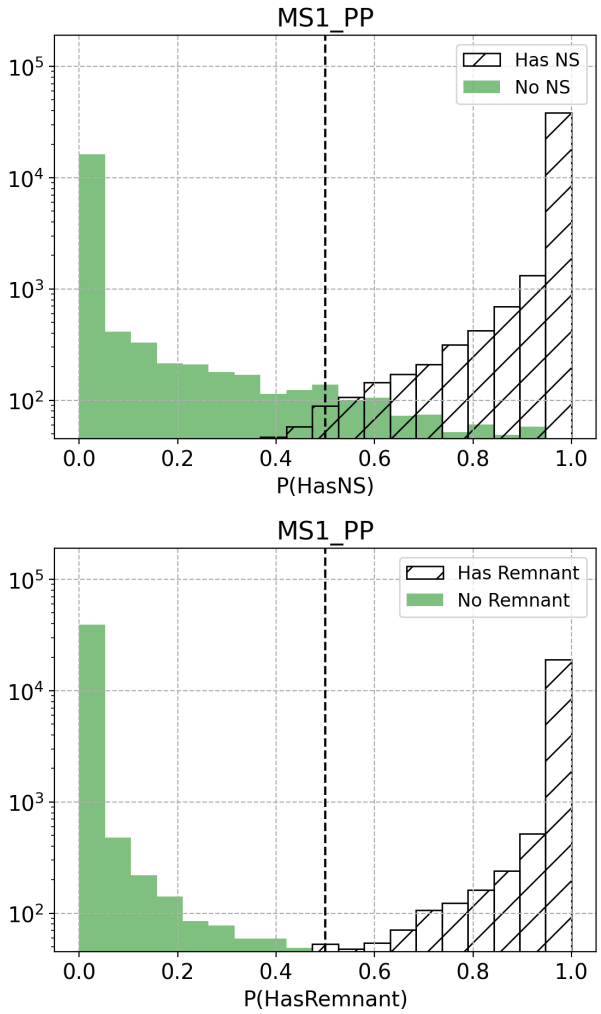


FIG. 14: Histograms MS1 PP